UNIVERSITÄT BONN

Modelling Complex Activities from Visual and Textual Data

Dissertation

zur

Erlangung des Doktorgrades (*Dr. rer. nat.*)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich–Wilhelms–Universität Bonn

vorgelegt von

Fadime Sener Merzbach

aus

Darende, Turkey

Bonn 2021

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich–Wilhelms–Universität Bonn

1. Gutachterin und Betreuerin: Angela Yao
2. Gutachter: Juergen Gall
Tag der Promotion: 06.07.2021
Erscheinungsjahr: 2021

Abstract

by Fadime Sener Merzbach

for the degree of

Doctor rerum naturalium

Complex activity videos are long-range videos composed of multiple sub-activities following some temporal structuring and connected purpose. Recognizing human activities in such videos is a long-standing goal with a broad spectrum of applications, such as assistive technologies, robot-human interactions, or security systems. Although extensive efforts have been made to recognize human actions from short trimmed videos, complex activity videos have received attention only recently. This dissertation provides several models and techniques for understanding human activities in these long-range videos. In particular, we focus on the problems of action anticipation and temporal action segmentation with both supervised and unsupervised learning approaches.

Motivated by decreasing the high annotation costs for learning models on complex activity videos, we present two approaches. Given a collection of videos, all of the same complex activity, our temporal action segmentation method partitions videos into sub-activities based on only the visual data in an unsupervised way, following an iterative discriminative-generative approach. Our action anticipation approach generalizes instructional knowledge from large-scale text-corpora and transfers this knowledge to the visual domain using a small scale annotated video dataset. In this work, we challenge ourselves to develop models for describing complex activities with natural language, enabling translation between elements of the visual and textual domains. We also present a complex activity dataset of videos aligned with textual descriptions. We finally present a generic supervised approach for learning representations from long-range videos that we apply to action anticipation and temporal action segmentation. In this work, we investigate the required temporal extent, the representation granularity, and the influence of semantic abstraction with our flexible multi-granular temporal aggregation framework for reasoning from short and long-range observations.

This dissertation advances the state of the art in complex activity understanding, challenges the community with new problems, presents novel models that learn visual and temporal relations between human actions, and contributes a dataset for studying the intersection of vision and language. We thoroughly evaluated our approaches and compared them with the respective state of arts on a set of benchmarks. We conclude this dissertation by reporting the characteristics of future research directions and presenting some open issues on complex activity understanding research.

Keywords: complex activity understanding, temporal action segmentation, action anticipation.

Zusammenfassung

von Fadime Sener Merzbach

zur Erlangung des Doktorgrades

Doctor rerum naturalium

Videos, die komplexe Aktivitäten zeigen, sind mehrere Minuten lang und bestehen aus mehreren Unteraktivitäten mit einer bestimmten zeitlichen Reihenfolge und gemeinsamem Ziel. Das Ziel, menschliche Aktivitäten aus solchen Videos zu erkennen, besteht seit vielen Jahren, und ebnet den Weg für ein breites Spektrum an Anwendungen, z.B. in Form von Assistenzsystemen, Mensch-Roboter-Interaktion oder Sicherheitssystemen. Obwohl es bereits zahlreiche Anstrengungen gibt, menschliche Handlungen anhand kurzer, geschnittener Videos zu erkennen, haben Videos von komplexeren Aktivitäten erst kürzlich Beachtung erlangt. Diese Dissertation bietet verschiedene Modelle und Techniken zur Erkennung menschlicher Aktivitäten aus dieser Art von Videos. In dieser Dissertation konzentrieren wir uns konkret auf die Vorhersage und die zeitliche Segmentierung menschlicher Aktivitäten, sowohl mit überwachten, als auch mit unüberwachten Lernansätzen.

Wir stellen zwei Ansätze vor, beide sind mit einer Reduktion des hohen Annotationsaufwands motiviert, der beim Training auf Videos von komplexen menschlichen Handlungen entsteht. Gegeben eine Videokollektion ein und derselben komplexen Aktivität unterteilt unsere Segmentierungsmethode Videos unüberwacht in Unteraktivitäten. Die Segmentierung basiert dabei rein auf den visuellen Videodaten und folgt einem iterativen diskriminativ-generativen Ansatz. Unser Ansatz zur Aktivitätsvorhersage verallgemeinert Wissen, gelernt aus großen Textkorpora, und überträgt es mithilfe einer deutlich kleineren, annotierten Videodatenbank ins Visuelle. In dieser Arbeit stellen wir uns der Herausforderung Modelle zu entwickeln, durch die komplexe Aktivitäten mittels natürlicher Sprache beschrieben werden können. Dadurch sind wir in der Lage, Wissen zwischen dem Visuellen und dem Textuelle zu übertragen. In diesem Rahmen präsentieren wir außerdem einen Datensatz komplexer Aktivitätsvideos, annotiert mit synchronisierten Textbeschreibungen. Zum Schluss präsentieren wir einen generischen, überwachten Ansatz zum Repräsentationslernen auf längeren Videos, den wir zur Vorhersage und zur zeitlichen Segmentierung von Aktivitäten anwenden. In dieser Arbeit über unser flexibles, multi-granulares Aggregationsframework untersuchen wir das beste Maß an zeitlichem Kontext und dessen optimalen Detailgrad, sowie den Einfluss des semantischen Abstraktionslevels, um kurze und lange Beobachtungen interpretieren zu können.

Diese Dissertation verbessert den Stand der Forschung zur Erkennung und Interpretation komplexer Aktivitäten aus Videos, stellt neue Herausforerungen an die Forschungsgemeinschaft, präsentiert neue Modelle, die visuelle und zeitliche Zusammenhänge menschlicher Handlungen lernen, und liefert einen Datensatz zur Unter-

suchung der Schnittstelle zwischen der maschinellen verarbeitung von visuellen und sprachlichen Daten. Wir führen eine gründliche Evaluation unsere Ansätze durch und vergleichen anhand einer Reihe von Benchmarks mit dem jeweiligen Stand der Forschung. Wir schließen diese Dissertation mit einem Ausblick auf zukünftige Forschungsrichtungen und einigen offenen Fragen zur Forschung über die Interpretation komplexer Handlungen.

Schlagwörter: interpretation komplexer Aktivitäten, zeitgleiche Segmentierung von Aktivitäten, Vorhersage von Aktivitäten.

Acknowledgements

First and foremost, I would like to thank my advisor Angela Yao. I consider myself very lucky to have had the opportunity to work with her. Throughout our many discussions I have learned a lot about how to structure research ideas and, most importantly, how to communicate them. I am deeply thankful to Angela for her constructive and thoughtful criticism, her eye for the detail, challenging me for more, being available for my questions all the time, and most importantly, for pushing me to explore my interests. I would also like to thank for the several opportunities I got through the supervision of master's students and interns. Those were crucial for improving my skills in supervising research topics.

Next, I would also like to thank Juergen Gall for his supervision. His research interests have always been an inspiration to me, particularly the goal of the research unit for anticipating human behavior. I am very happy to have been part of this project lead by him, which guided the topic of this dissertation heavily.

I want to thank my dissertation committee members for kindly accepting to evaluate my work.

I would like to thank my colleagues at the University of Bonn, particularly at the Institute of Computer Science II, both for making my day-to-day office life fun and for our pleasant gatherings. I want to especially thank Soumajit, Moritz, and Linlin. It was a pleasure to share the same office with them. I want to thank the NUS CVML members, particularly Dipika, who made my stay in Singapore so pleasant.

I would like to thank all my friends and their lovely kids, as it is always a pleasure to meet for barbecues, hikes and parties. Special thanks to Ali Can, Alyssa, Sasha and Violet. I would like to thank my family, my parents, brothers, sisters and cousins for being so lovely and fun. Lastly, this dissertation would not have been possible without your support Sebastian, thank you.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

An alphabetically sorted list of abbreviations used in the dissertations:

BERT	bidirectional encoder representations from transformers
BiLSTM	bidirectional long short-term memory
BLEU	bilingual evaluation understudy
C3D	convolutional 3D
CNN	convolutional neural network
CTC	connectionist temporal classification
ECTC	extended connectionist temporal classification
GAN	generative adversarial network
GCNs	graph convolutional networks
GRU	gated recurrent unit
GMM	generalized Mallows model
GT	ground truth
HMM	hidden Markov model
I3D	inflated 3D convolutional neural network
IoU	intersection over union
LSTM	long short-term memory
METEOR	metric for evaluation of translation with explicit ordering
MCMC	Markov chain Monte Carlo
MLP	multilayer perceptron
Mof	mean over frames
NLB	non-local blocks
OCDC	ordered constrained discriminative clustering
RNN	recurrent neural network
ST	skip-thoughts
SVM	support vector machine
TCN	temporal convolutional network
TSN	temporal segment networks
TSM	temporal shift module

Chapter 1

Introduction

Contents

1.1 Motivation

Daily events are high-level semantic concepts. They are composed of different human actions or activities that happen around us simultaneously and / or sequentially without pause. Recognizing such activities in everyday life plays a significant role in human-to-human interaction and relations. As such, humans are experts at recognizing what actions other people are performing, when they started, how far these actions have progressed, what type of transformations are brought to the environment through these actions, and what people will do next. They can effortlessly learn these complicated visual and temporal relations, generalize them to new observations from a few examples through past experience, and construct descriptions about their visual world. Such abilities are similarly highly essential for artificial intelligence applications, including video security or surveillance systems, assistive technologies, and robot-human interactions.

Every day, people are creating a tremendous amount of video data on the internet through streaming platforms such as Youtube, Instagram, etc. People share their daily activities in the form of vlogs, release how-to videos, or teach online courses. This creates a large pool of data for learning intelligent visual systems for tasks closely related to peoples' daily routines, including housework, repairing devices, and hobbies. Naturally, developing methods for automatic understanding of human activities using this abundant source of data could have a broad range of high-impact societal applications. Augmented reality and virtual reality can be used to embody an intelligent assistant and help with cooking, maintenance, and learning new skills. Robots can be trained to help in eldercare and healthcare as an assistant. Future

prediction and intention recognition systems can help in decreasing accidents or unintentional actions.

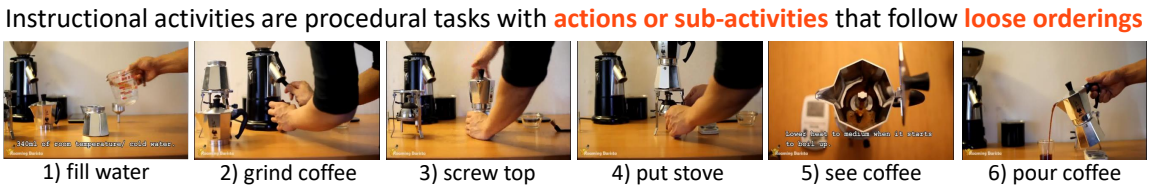
The need for technologies and applications to recognize other persons' activities has made understanding human activities from videos a central problem in computer vision since the early times of the field. The problem is simple: given a sequence of images, the task is to automatically analyze the video's contents and classify human activities. As easy as this question appears, it has been difficult finding a solution. The vision community pursued this problem over the last 20 years. Early research started with experiments in controlled environments with limited data and resources. Over the last years, driven by deep learning and engineering efforts, it evolved into advanced solutions trained on millions of hours of video and can already be used for recognizing daily activities in the wild. Despite this, a vast gap remains between humans and the state-of-the-art methods' performance.

Despite the difficulty of the task, motivated by the potential application and the success of humans who easily accomplish complex visual recognition, in this dissertation, we investigate models and techniques for understanding human activities from long-range untrimmed videos. In particular, we develop frameworks for studying "complex activity" videos, e.g., of instructional or daily activities.

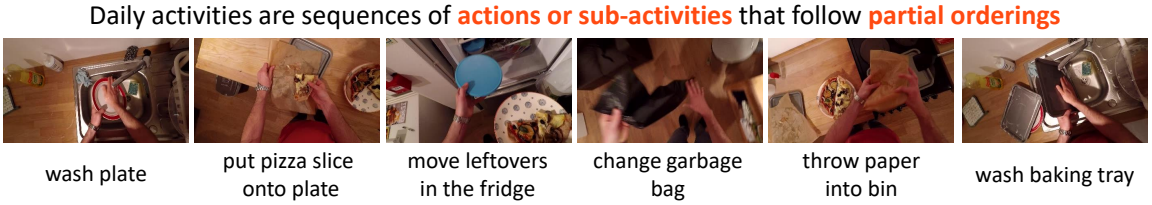
1.2 What is a Complex Activity?

We begin by describing the "complex activities" we will be working with throughout this dissertation. Let us start by presenting the difference between an "action" and "activity". Researchers distinguish the two terms differently, depending on the task they target and the data they use. Some establish a hierarchical categorization (Poppe, 2010; Chaaraoui et al., 2012), starting from action primitives, also known as gestures, moving to actions, and finally, arriving at activities. According to Poppe (2010), an "action" is a collection of action primitives and might correspond to, possibly, cyclic movement of the entire body. He defines an activity as a consecutive execution of several actions. He provides a clear example of a person performing the activity "jumping hurdles", where "left leg forward" is an action primitive, and "running" is an action that is composed of multiple such primitives. Finally, "jumping hurdles" is an activity as the human performs "starting", "running" and "jumping" actions.

To date, there is no agreed definition for action or activity (Poppe, 2010; Chaaraoui et al., 2012; Wang et al., 2016b). A strict separation or definition is difficult as their hierarchy is fluid and depends heavily on the annotation granularity of the dataset of interest (Sener et al., 2020). In terms of the datasets we work on, we are interested in two levels of hierarchy, by considering a complex activity to be composed of multiple "actions" that follow some ordering. First, we take the rather rigid definition of an "action" from Poppe (2010) and relax it to the description from Wang et al. (2016b): an action can affect the state of an object or the environment. We then take the definition of an "activity" from Poppe (2010) and define "complex activities" as the composition of these actions following some ordering and connected with purpose. To better emphasize that "actions" are building blocks of the



(a) Instructional activities



(b) Daily activities

Figure 1.1: In this dissertation, we experiment on two types of complex activity videos. Our primary focus is on *instructional activities* with certain actions or sub-activities that follow a loose ordering. In (a), there are six actions, and a person performs each action following this order with the goal of "making coffee". The second type are videos of *daily activities*, which contain a stream of only partially ordered actions that summarize people's daily routine. The ordering is partial because certain sub-activities can be executed independently of others. In (b), the person could put the leftovers into the fridge at any time. However, to be able to start washing a baking tray, one should first dispose of the baking sheet on top of it.

larger, more complex "activities", we instead use the terms "sub-activity" or "step". In this dissertation, we will thus use the terms "action", "sub-activity" and "step" interchangeably. For example, a complex activity is "making orange juice" with sub-activities "cut oranges", "squeeze oranges", "strain juice" and "pour into glass".

The work in this dissertation is mainly focused on complex instructional activities where a person has a well-defined goal that requires the execution of multiple actions or sub-activities in a loose order, depending on the type of the task. For example, a "changing tire" activity has a more rigid order of actions than "making fruit salad", which is determined by the degree of temporal interdependencies between the individual sub-activities. We also study daily activities which are also composed of multiple steps. However, the occurrence of their sub-activities only needs to be *partially* ordered; for example, a person might "change a garbage bag" at any time, but will have to "heat the oven" before "baking". We show some examples of such complex activities in Figure 1.1. In the instructional activity example (a), "making coffee" is a complex activity. This complex activity is further divided into the following steps: "fill water", "grind the coffee", "screw top", "put stove", "see coffee" and "pour coffee". Similarly, the example daily activity sequence (b) is a composition of steps of "wash plate", "put pizza slide on the plate", "move the pizza in the fridge", "change garbage bin", "throw the used baking paper into the bin", and "wash the baking tray".

To sum up, actions or sub-activities are short, and they transform objects and environments. Complex activities are composed of sequences of actions or sub-activities. Finally, a complex activity *video* either records an instructional task or a daily activity.

1.3 Contributions of the Dissertation

Analyzing motions and actions in videos has been of interest since the early days of computer vision (Hogg, 1983; Rohr, 1994). The focus has mostly been on classifying short video clips, which are manually trimmed to contain only video frames of human actions (Poppe, 2010; Herath et al., 2017). Human action recognition from such videos is a well-studied problem that recently has seen a remarkable performance boost with deep learning based approaches (Lin et al., 2019a; Carreira and Zisserman, 2017; Feichtenhofer et al., 2019). Developing models for such videos assumes observing trimmed clips during inference as well. However, we receive data in continuous streams; we do not get such tidily trimmed short clips in our daily lives. For practical applications, we need to develop algorithms that learn from and perform on long-range untrimmed videos.

The long-range untrimmed nature of complex activities makes the problem significantly more challenging, but the outcomes directly impact humans' daily lives, as we can directly benefit from potential applications. The richness in information present in complex activities lends itself well for us to define many variants of practical applications, such as early action recognition (Ma et al., 2016), action anticipation (Farha et al., 2018; Furnari and Farinella, 2019), verification tasks like compliance (Jaiswal et al., 2019), skill determination (Doughty et al., 2019), task completion (Becattini et al., 2017), generating notifications for missing actions (Soran et al., 2015), temporal action segmentation (Kuehne et al., 2014b), action localization in videos (Gao et al., 2019) and video summarization (Zhang et al., 2018). In this dissertation we mainly work on two different tasks on complex activities: temporal action segmentation and action anticipation. Research on these topics is very young and open for diverse research directions (Sener et al., 2015; Alayrac et al., 2016; Zhao et al., 2017; Farha et al., 2018; Richard et al., 2018b; Farha and Gall, 2019b). We develop novel temporal models with varying supervision levels and experiment on multiple datasets of complex activity videos.

Humans solve complex recognition tasks effortlessly and naturally, making it look straightforward. However, computers fall short in solving such high level-semantic tasks. Despite the progress made in recent years, we are still far from solving the complex activity understanding problem. Working with complex activities poses several difficulties. In the following, we analyze the main challenges of complex activity understanding and present our contributions verified on the action anticipation and temporal action segmentation tasks.

1.3.1 Learning with Less Supervision or Auxiliary Data

Complex activity videos are usually annotated with the temporal boundaries and the types of their sub-activities. Although there is a large amount of video data available on the internet, collecting and annotating such data can be very costly and time-consuming. Moreover, similar

to humans who associate observations with background knowledge, vision algorithms should be able to learn and generalize from a few training examples.

1.3.1.1 Challenges

Using less supervision: Following the trends of other computer vision problems, the first option to develop models for complex activities would be to follow a fully supervised setting. However, collecting and annotating data is expensive and tedious. This is because besides the need for action labels, supervised approaches also require knowledge of where the actions start and end. Alternatively, we might take a weakly supervised approach without using explicit start and end points of the actions, but other information we can acquire more cheaply. This might be narrations, other accompanying text, or an ordered list of sub-activities for the videos. However, approaches using such data assume a rigid order or alignment of textual and visual information. Unsupervised solutions with minimal assumptions are more desirable for removing the costs of labelling.

Generalizing to unseen activities: When considered from a traditional supervised point of view, where multiple demonstrations of the same activity are required, complex activity tasks would require large quantities of temporally segmented videos. Such annotations are costly, do not scale with the number of classes, and will not generalize to new activities. Thus, we require methods that minimize the amount of labelling effort required and do not need to be trained for every single possible complex activity.

1.3.1.2 Contributions

In Chapter 3, we present the first fully unsupervised method for temporal action segmentation on instructional videos without requiring any accompanying textual data or temporal annotations. Given a video collection of the same complex activity as input, we propose an iterative discriminative-generative approach to parse videos into their sub-activities. For this, we alternate between discriminatively learning the appearance of sub-activities and generatively modelling their temporal structure.

Our work in Chapter 4 is based on knowledge transfer, which allows avoiding the costly annotation of large video datasets and is the first work to explore zero-shot learning for the task of action anticipation. Our main contribution in this work is a learning paradigm: we prove that it is possible to learn models from text and transfer their knowledge to video to solve a very challenging vision problem. We decouple the problem into learning sequence semantics from text and appearance from video. By first aggregating information from a large text corpus, we learn the procedural structure of complex activities. This stage is unsupervised as the direct usage of text does not require any labelling. Based on the knowledge learned from the textual data, we develop models that learn future prediction on a small-scale video dataset. This stage is fully-supervised as we are using labeled video data. Finally, at inference time, we predict future steps for unseen complex activities. Decoupling the problem allows us to leverage text data for learning a richer semantic model, and thus minimizes the need for large-scale labeled video datasets.

1.3.2 Temporal Modelling of Complex Activities

Complex activity videos are *untrimmed* sequences where multiple sub-activities take place over time. Most importantly, these sub-activities are not independent but are related to each other through sequence dynamics. For example, Figure 1.1 shows two sequences of sub-activities connected with temporal relations. This makes temporal modelling more critical for understanding complex activities, compared to simple action recognition (Huang et al., 2018b).

1.3.2.1 Challenges

Unsupervised temporal modelling of complex activities: Prior works in unsupervised video parsing have two main weaknesses. First, they assume that some text, e.g., narrations, accompany the video (Sener et al., 2015; Alayrac et al., 2016). Second, and most importantly, they enforce strict ordering of sub-activities. This conflicts with the characteristic of complex activities which might include missing steps and deviations in orderings. We thus require temporal models that do not assume additional data sources, and can account for temporal variations.

Modelling instructional knowledge using language and vision: Learning procedural knowledge from text data and transferring it to visual data both requires approaches that can model instructional data and can combine text and visual modalities. Although sequence-to-sequence learning (Sutskever et al., 2014) has been shown to successfully generate continuous text, these approaches tend to perform poorly for instructional sequences, as they cannot fully capture their underlying structure. Even though there is a large body of work for relating visual and textual data in the captioning and visual grounding domains (Venugopalan et al., 2015; Rohrbach et al., 2014), such approaches are tailored for short videos, less for complex activities. We thus require methods that can model linguistic knowledge and integrate it into the visual recognition process of instructional data.

Modelling long-range temporal relationships: When built into the architecture, temporal models tend to fall short for complex activities. Growing memory requirements limit the temporal horizon that can be processed and therefore prevent using long-range information. For anticipation, models restricted to reason only from the most recent observations might miss helpful cues from the longer-range past. This can ultimately lead to a loss of accuracy in the predictions. We thus require methods that can capture long-range temporal relations. Moreover, succeeding in integrating long-range information raises questions about the necessary temporal extent and scale of context.

1.3.2.2 Contributions

In this dissertation, we present three different ways of modelling the temporal relations between sub-activities with respect to our three targets in Chapters 3, 4, 5.

Our first contribution is a novel way of handling temporal modelling for unsupervised temporal video segmentation, presented in Chapter 3. We treat the ordering as a sequence

of permutable activities. In this way, we can then model a probabilistic distribution over the permutations with a Mallows model (Fligner and Verducci, 1986). We assume a canonical ordering, and given an observed ordering, we can then define the ordering likelihood as an exponential function based on the distance between the canonical and observed ordering with some dispersion parameter that controls how much the distance influences the likelihood. The advantage of using a Mallows model is that it enforces global ordering constraints while allowing non-strict orderings and missing steps.

Our second contribution is an action anticipation framework which learns multi-step procedures with text and visual context, presented in Chapter 4. We construct a hierarchical system that is composed of four recurrent neural networks (RNNs). A sentence encoder RNN maps text data into context vectors. The context is given as input to another RNN, which models the sequential structure of the instructional activity and predicts the following steps, which are subsequently decoded back into sentences using a sentence decoder RNN. For training on videos, the sentence encoder is replaced with a video encoder RNN that maps video data into context vectors. The text encoder is used only for training, while the video encoder is applied at test time. Our hierarchical model trained on textual data allows for capturing the underlying sequential nature of the instructions and transferring this knowledge to visual data.

Our third contribution is a framework that pairs recent observations with long-range past or future information at different scales and granularities, presented in Chapter 5. This form of coupling has not been considered for minute-long videos. We split video streams into snippets of equal length and max-pool the frame features within the snippets over multiple scales. We then relate the max-pooled recent observations to max-pooled past information using attention mechanisms. In our experiments we demonstrate simple max-pooling, together with attention, to be effective tools for aggregating multi-scale past and recent information to represent long-range videos that span tens of minutes. We show that the extent of long-range context is beneficial, however, that it depends on the nature of the activity and label granularity. Our framework is very flexible and can be used for different tasks with slight changes. We successfully evaluate our model for next action anticipation, dense anticipation, complex activity recognition, action recognition, and temporal action segmentation with minimal modifications.

1.3.3 Visual Representations

Working with complex activities poses several difficulties. They inherit the typical computer vision problems, like variations in camera motion, viewpoint, illumination, and often involve further challenges.

1.3.3.1 Challenges

Fine-grained visual recognition of sub-activities: Low inter-class and high intra-class variations are naturally challenging. In the complex activities we are working with, this becomes more prominent, as there are considerable variations in the visual appearance of the

same sub-activities across different videos, while the different sub-activities in the same video tend to be visually similar. To be able to distinguish sub-activities, we need discriminative representations. This is more critical for unsupervised than for supervised approaches, as we cannot rely on sub-activity labels.

Irrelevant content: In instructional videos, it is common to see some video segments where the actor is talking or highlighting previous or subsequent sub-activities and giving recommendations. These frame groups, which we call "background frames", are not related to the sub-activities and can occur with various durations and at arbitrary locations. We thus need models that work under these uncertainties due to background frames.

Influence of different modalities and semantic abstraction: Videos can be represented in various ways, ranging from low-level visual features to high-level semantic representations across different modalities, or their combinations. Although appearance-based video features are frequently used for complex activities, methods would also benefit from different modalities to get more discriminative video representations.

1.3.3.2 Contributions

We examine the problem of representing fine-grained action classes and irrelevant frames in Chapter 3. Our first contribution is a discriminative approach for representing fine-grained sub-activities. We propose a discriminative appearance model via clustering by grouping frames according to their semantic similarity with respect to their sub-activity rather than relying entirely on visual appearance. Our second contribution is explicit background handling. We introduce a variant of our model for handling background to account for frames unrelated to the actual sub-activities.

Our temporal aggregates model in Chapter 5 is a highly flexible framework that handles low-level features or high-level semantic abstractions as input. We experiment with ensembles of different features and compare the influence of video-based features such as appearance and motion features and high-level semantic labels from actions and objects. With our simple feature ensembles we obtain state-of-the-art performances. Our target task in this chapter is action anticipation which is a task that might benefit from high-level representations of past observations, e.g., action labels. One way to achieve this is to assign labels with a temporal action segmentation algorithm and then use them for anticipation. This approach is adopted by the state of the art in action anticipation. However, temporal action segmentation is a challenging and not fully solved task itself. We evaluate our model with the outputs of different segmentation algorithms, their combination with visual features, and ground truth labels. Our experiments confirm that higher abstraction levels, such as frame-wise action labels are more beneficial for anticipation. However, there is still a large gap between using the outputs of temporal action segmentation algorithms in comparison to ground truth labels.

1.3.4 Natural Language Descriptions of Complex Activities

Many complex activity datasets are annotated with action categories composed of verb-noun-pairs; e.g. "boil water". However, working with such categories has several disadvantages.

The number of categories grows exponentially with the number of actions and objects, which leads to very long-tailed distributions. Moreover, the end-users of vision systems are humans who use sentences, which makes working with sentence-based outputs more desirable, as it removes the need for translations between natural language and category labels. Instead of using category-based labels, another scope of this dissertation is shifting the focus to describing human activities with sentences.

1.3.4.1 Challenges

Lack of sentence-based datasets with diverse complex activities: Typical existing complex activity datasets consist of multiple demonstrations of the same activity and are labeled with category-based labels. There are a few datasets that have sentence-based labels. However, they either lack diversity in their complex activities (Zhou et al., 2018b), which is essential to show the zero-shot / few-shot recognition or anticipation performance, or they lack temporal alignments between text and video (Malmaud et al., 2015). Thus, there is a demand for a video dataset annotated with aligned sentences, over a diverse set of complex activities.

Evaluating sentences: Category-based label predictions are straightforward to evaluate, as we can precisely measure the output quality. Sentences are, on the other hand, difficult to evaluate. Although there are widely used sentence evaluation methods, they often fail in capturing linguistic fluency and coherence. Moreover, tasks like action anticipation are even more difficult to evaluate because there might not always be a single outcome but multiple possible future actions. Thus we need reliable evaluation systems for assessing the correctness of the generated sentences.

1.3.4.2 Contributions

We address these challenges in Chapter 4. We collect a new dataset which is annotated with sentences and contains a large number of different complex activities. Our dataset is composed of videos of around 2.500 different cooking recipes accompanied by temporally aligned recipe steps. In this work, we design a zero-shot action anticipation framework using sentences. We avoid the dependence on category labels by training with and predicting sentences, i.e., "add water" vs. "Gradually add the water to the bowl until combined". Using sentences adds richness to the instructions. It also allows for anticipation of not only actions but also objects and attributes. Finally, it makes data labelling manageable. We are the first to predict the future in the form of sentences. In some sense, this is analogous to video captioning except that our sentences describe a possible future instead of current observations.

Our last contribution are the various ways of evaluating our sentences. We use several state-of-the-art sentence evaluation methods along with ingredient and verb recalls. Since so far, there is no well-established method for assessing the quality of anticipation predictions, apart from using the aforementioned automatic metrics, we also consult human raters to evaluate our predictions.

1.4 Organization of the Dissertation

In this dissertation, we develop new models for understanding human activities in complex activity videos and verify them on the action anticipation and temporal action segmentation tasks. The dissertation is organized as follows.

We present a thorough analysis of the methods on complex activity understanding in videos and relate our contributions to prior work in **Chapter 2, Background**.

Chapter 3 presents our **unsupervised method for temporal action segmentation** on videos. We parse video streams into semantically meaningful sub-activities and also capture the temporal relationships that occur between these sub-activities. Specifically, we work with instructional activities. The content of this chapter corresponds to our CVPR 2018 publication, *Unsupervised Learning and Segmentation of Complex Activities from Video* (Sener and Yao, 2018).

Chapter 4 introduces the new problem of **zero-shot action anticipation** for instructional activities, and presents our transfer learning-based solution. Contrary to standard action recognition, where the task is to assign action labels to videos, our interest in this work is to predict future actions based on current and past observations, i.e., action anticipation. Here, "zero-shot" refers to the anticipation of future steps for complex activities previously unseen during training. The content of this chapter corresponds to our ICCV 2019 publication, *Zero-shot Anticipation for Instructional Activities.* (Sener and Yao, 2019).

Chapter 5 addresses the lack of methods that can handle long-range videos and introduces our **temporal aggregate representations model**. We design a generic framework for long-range video representation and verify it on several tasks, including action anticipation and temporal action segmentation. The content of this chapter corresponds to our ECCV 2020 publication, *Temporal Aggregate Representations for Long-Range Video Understanding.* (Sener et al., 2020).

Finally, in **Chapter 6**, we summarize the addressed and remaining challenges, and conclude this dissertation.

Chapter 2

Background

Contents

In this chapter, we first provide a preliminaries section where we present an overview of video representation methods in Section 2.1.1 and review state-of-the-art action recognition approaches in Section 2.1.2. In Section 2.2, we provide a detailed overview of complex activity datasets, action anticipation and temporal action segmentation approaches. This section also includes an overview on temporal action detection and key-step localization approaches which are compared against in the proposed works in Chapters 3, 4, 5. Finally, we summarize joint video and text modelling methods in Section 2.3.

2.1 Preliminaries

2.1.1 Video Representation

Video representation is an important part of the research presented in this dissertation, where we represent long videos with some generic frame/snippet-based features and build our models on top of them. In this section we overview several common image/video representation approaches. We first review the hand-crafted representations and then focus on deep learning based representations.

2.1.1.1 Hand-crafted Representations

Conventional approaches for computing video representations are composed of three main stages. The first stage is detecting sparse (Laptev, 2005) or dense (Wang et al., 2011) spatio-temporal interest points. In the second stage, these interest points are described using local spatio-temporal features such as histograms of oriented gradients (Dalal and Triggs, 2005), histograms of optical flow, and motion boundary histogram (Dalal et al., 2006). The final stage is computing a high dimensional encoding of these features and combining them to a fixed-size video-level description vector. For example, Laptev et al. (2008) quantize spatio-temporal features using a learned k-means-based "visual vocabulary" over these features and compute bags of features over varying spatio-temporal grids. These fixed-length representations are then used to train a classifier, such as an SVM.

Wang et al. (2011) track interest points using optical flow fields to form dense trajectories. Then, they compute spatio-temporal features along these trajectories. Wang and Schmid (2013) improve this approach using camera motion to correct the trajectories and present improved dense trajectories. They use Fisher vector (Perronnin et al., 2010) encoding along trajectories, which achieves state-of-the-art performance. Different than bag of visual words, Fisher vector encoding employs a Gaussian mixture clustering and benefits from soft-assignment. State-of-the-art hand-crafted video representations are built on improved dense trajectories and Fisher vector encoding to compute video features.

2.1.1.2 Deep Learning Based Representations

Deep learning refers to machine learning algorithms that learn representations for inputs by passing them through a hierarchy of transformations. A notable example are convolutional neural networks (CNN), which are composed of multiple layers where the inputs are convolved with filter weights, and non-linear transformations are applied to the filter responses. Such networks usually contain millions of parameters and require large-scale labeled datasets to support their learning process.

Convolutional networks learn feature hierarchies ranging from low-level representations like edges to higher-level semantic concepts such as objects. They implicitly learn powerful and interpretable (Zeiler and Fergus, 2014) representations where the upper-layers can be used for extracting high-level feature representations. Such features demonstrate a strong ability to generalize to images outside the dataset they are trained on, making them very valuable as generic representations. In this section, we first overview standard CNNs trained for image classification, followed by a review on architectures trained for action classification.

Convolutional Neural Networks Trained for Image Classification:

Image classification aims at automatically annotating images with pre-defined categories. ImageNet (Russakovsky et al., 2015) is a standard benchmark for large-scale image classification, which is used for the ImageNet large scale visual recognition challenge (ILSVRC). The goal is to classify images into one of the 1000 object categories selected from humans' daily lives, including animals, household objects, vehicles, etc. AlexNet (Krizhevsky et al.,

2012) was the winner of ILSVRC in 2012 by a significant margin, which made CNNs receive massive attention. Later a variety of CNN architectures have been proposed for image classification with architectures improved in terms of accuracy (He et al., 2016) and computational costs (Szegedy et al., 2015). For this, researchers both developed hand-crafted CNN architectures and explored neural architecture search (Liu et al., 2018). In the following, we list the most prominent CNN architectures to date, which we also refer to throughout this dissertation. Although these CNNs are trained for image classification, they can be used for high-level feature extraction from video frames.

AlexNet (Krizhevsky et al., 2012) contains five convolutional layers, which are followed by an activation function and max pooling respectively, and three fully-connected layers. Several novelties made AlexNet the state of the art. Instead of sigmoids – a standard in traditional neural networks, it uses rectified linear unit (ReLU) activations as non-linearity. ReLUs are computationally more efficient compared to sigmoids, as they do not involve expensive evaluations of the exponential function. Most importantly, the sigmoid's derivative becomes very small for saturating inputs, which leads to vanishing gradients. AlexNet also uses dropout (Srivastava et al., 2014) layers after every fully connected layer to reduce overfitting.

VGGNet (Simonyan and Zisserman, 2014b) replaces the large convolutional filters of Alexnet, 11×11 and 5×5, with small-sized 3×3 filters. Stacking smaller-sized filters allows for deeper networks, e.g., 16 and 19 layers, which enables learning more complex functions. The disadvantages of VGGNet are its huge time and memory requirements.

GoogLeNet (Inception-v1) (Szegedy et al., 2015) introduces the inception module, which, instead of going deeper, grows wider using multiple parallel filters of different sizes, to capture fine-grained features at varying scales. Additionally, it introduces several novelties, which both increase the accuracy and computational efficiency. It uses 1×1 convolutions for dimensionality reduction and faster computations. Instead of using fully connected layers, which contain a large number of parameters, e.g., around 90% of those in Alexnet, GoogLeNet uses global average pooling across feature maps after the last convolution layer, which decreases the total number of parameters significantly.

Residual Networks (Resnet) (He et al., 2016): Deep CNNs up to 30 layers (Simonyan and Zisserman, 2014b; Szegedy et al., 2015; He et al., 2015; Ioffe and Szegedy, 2015) have been shown to improve the performance. However, going deeper in these CNNs is shown to drop their generalization capability due to optimization difficulties, which even increases the training error. Instead of learning a direct mapping, $h(x) = y$, from an input x to an output y, He et al. (2016) propose learning a residual function, $f(x) = h(x) - x$ as the difference between a mapping of the input, $h(x)$, and the input, x. Therefore a "skip connection" from the previous layer is used to let the following layers approximate such a residual function. This has been proven to successfully train networks with varying depths of up to 152 layers. Similar to VGGnet, Resnets use 3×3 filters, and like GoogLeNet, they use global average pooling to reduce parameters. With their compelling results, Resnets are frequently used by many vision applications.

CNNs trained for Action Recognition on Videos:

The previously mentioned CNN architectures are trained on Imagenet and could be used for extracting frame-wise features for videos. However, it is shown that these features do not lead to better performances than the state-of-the-art hand-crafted features (Wang et al., 2013) for video-based action recognition (Karpathy et al., 2014). Several works extend these CNN architectures for action recognition by modelling the temporal structure of videos using motion in a second stream (Simonyan and Zisserman, 2014a) or extending 2D convolutions to the temporal dimension (Tran et al., 2015; Carreira and Zisserman, 2017). These models can be pre-trained on large video datasets for action classification and used to extract features on other datasets. Since they are trained for action classification, the features extracted from such networks also generalize better to action recognition related tasks.

The two-stream (Simonyan and Zisserman, 2014a) and convolutional 3D (C3D) (Tran et al., 2015) networks are early architectures used for extracting video features. The two-stream network combines a spatial stream, where action recognition is performed on single frames, and a temporal stream for action recognition on motion using stacks of optical flow images from consecutive frames. C3D is one of the early 3-dimensional convolutional networks, extending convolutions to the temporal domain. It includes eight convolutional and two fully connected layers and can process 16 frames per video with a spatial resolution of 112×112.

Inflated 3D ConvNet (I3D) (Carreira and Zisserman, 2017) is currently the state-of-the-art architecture to extract generic features for video understanding. It builds on the state-of-the-art image classification architectures and inflates their filters and pooling kernels to 3D. Motivated by the success of 2D CNNs trained on very large image datasets, it is trained on the Kinetics dataset (Kay et al., 2017). Kinetics covers 400/600/700 human action classes in different dataset versions, where at least 400/600/700 video clips for each action are collected from Youtube. These actions are human-centric and cover various interactions with other humans and objects. I3D takes 64 frames as input with 224×224 spatial resolution.

2.1.2 Action Recognition on Videos

In this dissertation, we work on complex activities that are composed of multiple actions. Research on complex activities has already considerably benefited from solutions for action recognition (Furnari and Farinella, 2019; Huang et al., 2016; Farha and Gall, 2019b). Advancements in this area will lead to further gains in complex activity research. We now continue our overview with simple action classification in videos containing a single action.

In a broad perspective, action recognition refers to classifying an input video into a pre-defined action category. This section focuses on approaches and recent developments in recognizing actions on short and often trimmed videos using deep models. Such videos are composed of a sequence of frames that are associated with a single action class. We restrict our discussion mainly to fully supervised learning based approaches. We start our overview with two-stream-based solutions and continue with 3D CNNs. Finally, we review the approaches that aim at capturing long-range dependencies for action recognition.

2.1.2.1 Two-stream Networks

CNNs' success on image classification promptly made them the method of choice in video-based action recognition. Initial approaches are built on 2D CNNs using only 2D convolution and pooling operations. Karpathy et al. (2014) propose one of the first approaches for action recognition using CNNs with detailed analysis of several architectures to capture local spatio-temporal information in videos. Their main CNN architecture is composed of two streams where one works on low-resolution frames, while the other works on high-resolution crops from the middle-portion of these frames. However, their results are significantly worse than using hand-crafted representations (Wang and Schmid, 2013) (85.9% vs. 65.4%). This lack of performance has been accounted for by other two-stream approaches, where one stream processes RGB frames, and the other optical flow images to account for motion in the videos.

Some actions can easily be identified from the appearance contained in a single frame, e.g., "eating apple" or "slicing apple". However, single frames can be confusing for others, where motion information might be crucial, e.g. "closing lid" or "opening lid". Simonyan and Zisserman (2014a) propose a two-stream network where a 2D CNN is used to model appearance of actions from single frames and another to capture their motion from stacks of optical flow images. Although this network outperforms using hand-crafted representations (Wang and Schmid, 2013), it does not relate the motion in the the optical flow stream to the appearance stream. To address the missing relations, e.g., "*what* is moving *where*", Feichtenhofer et al. (2016) fuse the two streams. They report the best results with a fusion after the last convolutional layer. Their improvement over Simonyan and Zisserman (2014a) suggests the importance of jointly reasoning on appearance and temporal features for action recognition. Feichtenhofer et al. (2017) further investigate adding residual connections between the two streams.

Similar to Simonyan and Zisserman (2014a), Wang et al. (2016a) learn motion and appearance information in two streams in their temporal segment networks (TSNs). Differently, TSNs partition videos into segments and separately process each in two streams, to finally aggregate their outputs using a consensus function. This allows for incorporating more temporal information from the videos, which increases the accuracy over the two-stream model of Simonyan and Zisserman (2014a) by 10%. Kazakos et al. (2019) build on TSNs, but instead of using late fusion, apply mid-level fusion, which is shown to improve the performance in two-stream networks (Feichtenhofer et al., 2016).

Two-steam approaches are frequently used in action recognition. Chéron et al. (2015) fuse motion and appearance of human body parts for action recognition. Weinzaepfel et al. (2015); Gkioxari and Malik (2015) employ such architectures for action localization. Recently, with audio becoming a popular modality in newly collected datasets (Damen et al., 2018; Zhukov et al., 2019), a third stream is included for audio (Kazakos et al., 2019; Xiao et al., 2020).

2.1.2.2 3D Convolutional Neural Networks

Two-steam approaches are computationally efficient, as they build on 2D convolutions. However, their ability to encode temporal order or relationships is limited to the optical flow

stream. Now we will briefly overview related works on 3D CNNs, which apply convolution and pooling operations over the temporal dimension as well.

An early neural network using 3D convolutions is proposed by Ji et al. (2012) who train this network for action recognition on regions detected to contain humans. Tran et al. (2015) design a 3D CNN, namely "C3D", trained over frames of a large-scale action recognition dataset (Karpathy et al., 2014). To find the best 3D architecture for video processing Tran et al. (2017) employ neural architecture search. However, due to the large number of parameters of 3D CNNs, these early 3D CNNs are based on relatively shallow custom architectures. Instead, Carreira and Zisserman (2017) propose I3D, which utilizes established successful 2D architectures by inflating their filter and pooling kernels to 3D. This approach is shown to obtain state-of-the-art performances in all benchmarks. Recently, Feichtenhofer et al. (2019) propose SlowFast networks using two-streams of 3D ResNets, where one stream operates at a low frame rate but high resolution to capture spatial semantics, and the other on high frame rates but low resolution to capture motion. The two streams are fused by connections at multiple layers and achieve state-of-the-art results.

3D CNNs are successful at spatial and temporal representation learning, however, such architectures are large and expensive. Therefore, several approaches seek a balance between speed and accuracy by replacing 3D with low-cost 2D convolutions. Xie et al. (2018) apply this replacement in the initial layers while keeping the 3D convolutions in deeper layers. Their model, S3D, which stands for "separable 3D CNN", achieves good performance, indicating that a temporal modelling is more important for higher-level features. Another way of decreasing the computational costs is factorizing the 3D convolutions into 2D spatial and 1D temporal convolutions. Sun et al. (2015) use 2D spatial kernels in the early, and temporal kernels in the deeper layers of their network. Tran et al. (2018) propose an architecture, R(2+1)D, where the 3D convolutions are factorized to successively operated 2D spatial and 1D temporal convolutions for *each* layer.

Recently, Lin et al. (2019a) propose their temporal shift module (TSM) with similar motivations. This module when incorporated to a 2D CNNs achieves the superior performance of 3D CNNs, while maintaining the low complexity of 2D CNNs. Given a video, TSM shifts channels along the temporal dimension, which blends the information from neighboring frames. It can be inserted into 2D CNNs with no additional costs and is shown to have strong spatio-temporal modelling ability compared to 3D networks (Damen et al., 2020a).

Optical flow is used as an additional modality by several state-of-the-art architectures which train separate models on flow and RGB images (Carreira and Zisserman, 2017; Feichtenhofer et al., 2019; Lin et al., 2019a; Damen et al., 2020a). Although combining the results from two networks improves the performance, it also adds the costs of computing and storing optical flow images. Moreover, since optical flow is a hand-designed representation, such networks are often not learned end-to-end and introduce extra inference time. Some recent works (Crasto et al., 2019; Stroud et al., 2020) focus on incorporating flow information into 3D networks using knowledge distillation. A teacher network, which recognizes actions from optical flow sequences, transfers knowledge into an RGB-based student network, which recognizes actions from videos.

Even though 2D CNNs cannot capture temporal information in videos, their competitive performance implies that most of the action information is contained in the appearance features. Huang et al. (2018b) propose two models for analyzing how important the temporal modelling in 3D CNNs is for accurate action recognition. Their first model synthesizes a video from a subset of frames and the other chooses key-frames from videos independent of motion. They show that C3D (Tran et al., 2015) trained with either synthesized frames or key-frames performs comparably to using the original video for training. This suggests a tighter upper bound for the influence of motion for recognition. They provide a detailed evaluation on the UCF (Soomro et al., 2012) and Kinetics (Carreira and Zisserman, 2017) datasets, where they show motion to be necessary only for some classes. Even for these classes, they show that the entire video is not necessary. They also show that motion is dataset-specific; for example, it is less critical for Kinetics. In a later work, Sevilla-Lara et al. (2019) shuffle frames in time to evaluate the influence of temporal order on action recognition. They define a new set of actions from the classes which cannot be recognized without the correct order. Performance on this set measures a network's ability to capture temporal information. They also show that inflated convolutions are biased towards classes where motion is not important. Such observations and experiments become further interesting for recognizing actions from the daily lives of humans, where human-object interactions are abundant. Li et al. (2018c) argue that there are object-, scene-, or person-centric biases in existing action recognition datasets. They claim that most of the time, the action class is determined merely by an accurate recognition of objects, scenes, or person-centric captures. They collect a new dataset of *diving* videos, where they try to minimize such biases, so that models can focus on understanding the temporal dynamics.

2.1.2.3 Capturing Long-range Dependencies

CNNs are inefficient in modelling long-range interdependencies between frames, as they are limited by their convolution operators, which only capture local features and relations. Moreover, 3D CNNs are usually limited to 8 to 32 frames as input, as increasing the temporal extent increases memory requirements and computations (He and Sun, 2015), and runs risk of over-fitting. These reduce chances of learning long-range temporal relations in videos. Different methods have been proposed to tackle these issues.

Several works model temporal relationships across video frames using 2D CNNs and posthoc fusion. Such fusion can be implemented with recurrent neural networks. Yue-Hei Ng et al. (2015) extract frame-wise features from a two-stream framework and propose several ways of aggregating them. They show that max-pooling over frame features performs comparably to RNNs. Similarly, Donahue et al. (2015) employ RNNs to learn representations from 2D CNNs for several tasks, such as activity recognition, image and video description. Li et al. (2018d) and Sharma et al. (2015) show improvements by incorporating attention mechanisms into RNNs. Fernando et al. (2016) learn ranking functions to capture the temporal evolution of sequences. Zhou et al. (2018a) propose a module, called temporal relation network (TRN), which reasons about the temporal interactions between frames at multiple time scales. They sparsely sample frames from different temporal segments and feed them into different frame

relation modules. Although TRNs are good at capturing high-level temporal structures, they cannot capture low-level temporal relations, as those are lost during feature extraction.

A direct way for temporal modelling is to use 3D CNNs. However, these could be limited in modelling long-range interdependencies due to the locality of convolutional operations. Wang et al. (2018c) propose non-local neural networks, which exhaustively correlate features between all locations using attention mechanisms, to relate long-range dependencies in both space and time. Chen et al. (2019) project features into a latent interaction space and perform graph convolutions to extract relation-aware features. The features are then projected back to the original coordinate space. Recently, Wu et al. (2019) incorporate long-range feature information from object regions into the 3D CNNs using non-local operations.

Humans can understand what happens in a video by looking at the appearance, states, and locations of objects. Reasoning on what happened to objects is crucial for understanding actions that are dominated by human-object interactions. To model finer-grained interactions, a large body of works employs pre-computed object proposals, given by pre-trained detectors. Wang and Gupta (2018) construct spatio-temporal graphs, where nodes correspond to object region proposals from the entire video. They perform reasoning on this graph representation via graph convolutions. Baradel et al. (2018) use a pre-trained object segmentation network to extract object regions and model their relationships through space and time. They model inter-frame interactions between all objects using a pairwise mapping, which is then fed into an RNN for action classification. A similar solution is used by Tekin et al. (2019) on 3D poses of hands and objects. Ma et al. (2018) model the co-occurrence of object regions using a module which selects object subsets that are important for discriminating human actions. Wang et al. (2020d) explore the relations between objects and motion.

Although the methods discussed so far show quite impressive results, they are mostly tested for action recognition on short *trimmed* videos. For example, 3D CNNs do not allow processing more than a limited amount of frames per video at a time. Several approaches use sampling (Wang et al., 2016a) and striding (Feichtenhofer et al., 2017) to process larger spans of video. However, their extend is still limited, and sampling might miss important frames in *untrimmed* videos. One solution to enable modelling longer *untrimmed* sequences for simple action recognition is pre-computing frame-wise features and applying temporal modelling on them. Several works have been evaluated on the Youtube8M dataset (Li et al., 2017), which provides audio and appearance features pre-extracted for its *untrimmed* sequences. These works are focused on developing temporal models and aggregating features to arrive at a robust and discriminative video representation. Miech et al. (2017) pass the features through some learnable pooling modules (Arandjelovic et al., 2016), and use a two-stream architecture for aggregating audio and visual features. Tang et al. (2018) use non-local operations with pooling for constructing more representative video descriptors. Skalic and Austin (2018) employ model distillation over an ensemble of numerous models, such as LSTMs, which improves the performance while reducing the model size.. However, since these works operate on pre-extracted features, they are not end-to-end trainable and thus cannot optimize the features for the target task.

2.2 Understanding Complex Activities

Human activities exhibit various potential directions to explore, including action recognition, detection, anticipation, captioning, and temporal action segmentation. This dissertation is particularly focused on two of these, namely temporal action segmentation and action anticipation. However, in this section, we also overview several other tasks carried out on complex activity videos, as we either provide comparisons to models targeting such tasks, e.g., key-step localization, or refer to these works frequently in our related works. We first overview datasets frequently used in complex activity research. We then provide a detailed review of related works on action anticipation and temporal action segmentation. Finally, we briefly summarize temporal temporal action localization and key-step localization on complex activities.

2.2.1 Datasets

The need for developing algorithms to understand complex activities in real-world scenarios has prompted the construction of standardized datasets. This section provides an overview of complex activity datasets that are composed of untrimmed videos with multiple actions.

We divide complex activity datasets into two categories. The first category are those datasets which are composed of instructional activity videos (Stein and McKenna, 2013; Kuehne et al., 2014b; Alayrac et al., 2016; Zhou et al., 2018b), where the videos are compositions of sub-activities following some loose orderings. The second category consists of datasets that are composed of sequences of daily activities (Pirsiavash and Ramanan, 2012; Damen et al., 2018; Sigurdsson et al., 2016), where the sub-activities follow partial ordering, as some sub-activities can be executed independently of others. In this section, we also present several other long-range video datasets where the order of sub-activities does not matter (Gorban et al., 2015), or they are mostly employed for tasks where the temporal structure is not leveraged (Caba Heilbron et al., 2015).

2.2.1.1 Instructional Activities Datasets

In instructional activity videos, a person executes a sequence of actions, following some order, to arrive at a goal, such as "making a dish" or "assembling furniture". Instructional activities can be divided into two groups, where some are recorded datasets (Fathi et al., 2011b; Kuehne et al., 2014b; Li et al., 2018b), and some are collected from online video sharing platforms such as YouTube (Zhou et al., 2018b; Tang et al., 2019). In Table 2.1, we provide detailed statistics and comparisons for each dataset. In this table, we can see that the recorded datasets are either captured from only an egocentric point of view (Fathi et al., 2011b; Li et al., 2018b), or from a third-person view (Rohrbach et al., 2012; Kuehne et al., 2014b; Toyer et al., 2017). The collected datasets are usually composed of multi-view videos. Finally, the majority of instructional datasets focus on cooking activities, which might create biased solutions towards one domain.

Instructional activity datasets are usually composed of multiple demonstrations of the same complex activities, e.g., YouCookII includes 89 different complex activities, with

around 22 sample videos for each of them. Different than these datasets, our Tasty Videos dataset (Sener and Yao, 2019) (see Chapter 4) includes a diverse set of complex activities. The majority of the instructional datasets are annotated with category-based labels such as "pour milk", while several others contain sentences (Malmaud et al., 2015; Zhou et al., 2018b) including our newly collected Tasty Videos dataset.

In the following, we present a list of instructional activity datasets sorted by the year they are published.

Georgia Tech Egocentric Activities (GTEA) (Fathi et al., 2011b) is a recorded dataset that contains videos from seven complex activities of making: "hotdog sandwich", "instant coffee", "peanut butter sandwich", "jam & peanut butter sandwich", "sweet tea", "coffee & honey", and "cheese sandwich". The videos are recorded with a camera mounted on a cap, worn by four participants. It includes 28 videos with 560 sub-activity segments from 71 classes.

MPII-Cooking Activities (Rohrbach et al., 2012) is a recorded dataset with 44 videos from 14 cooking activities such as "making fruit salad" or "making cake". There are 65 fine-grained sub-activities. Twelve participants prepare dishes in a single kitchen using a static camera. As a result, the dataset does not exhibit any scene and motion biases. The participants are instructed verbally, but frequently divert from the instructions using different utensils or ingredients. This dataset is later extended to MPII Cooking 2 (Rohrbach et al., 2015) which contains 273 videos from 30 participants. There are 67 fine-grained sub-activities from 59 diverse dishes. The extension also includes a large number of textual scripts that describe how to prepare certain dishes. These scrips are collected independently of the videos and do not have any temporal association. Even though the number videos are limited, using these scrips enable knowledge transfer for recognizing unseen sub-activities.

TACoS (Regneri et al., 2013) includes 185 videos from the MPII Cooking 2 Dataset from 26 dishes and provides multi-sentence descriptions for 8.8K segments.

50Salads (Stein and McKenna, 2013) is composed of 50 recorded videos of 25 participants making two different mixed salads. There are 17 sub-activities in total. The average duration of the videos is 6.4 minutes. The videos are captured by a camera with a top-down view onto the work-surface. The participants are provided with recipe steps which are randomly sampled from a statistical recipe model. The dataset provides RGB-D videos and data from wireless accelerometers which are attached to the kitchen utensils.

Breakfast Actions (Kuehne et al., 2014b) targets recording videos "in the wild", in 18 different kitchens, as opposed to the controlled lab environments in the previous datasets. The participants are not given any scrips and the recordings are unrehearsed and undirected. The dataset is composed of the 10 complex activities of breakfast-related tasks, of making "coffee", "orange juice", "chocolate milk", "tea", "bowl of cereals", "fried eggs", "pancakes", "fruit salad", "sandwich", and "scrambled eggs". Each video is composed of a set of sub-activities from 48 different classes such as "pour water" or "stir coffee". This dataset is recorded with 52 participants with multiple cameras, all from a third-person point of view. The number of cameras per participant varies from 3 to 5. There are 1712 videos, when accounting for the

multi-camera views, and around 8500 sub-activity segments in total. The average duration of the videos is 2.3 minutes.

What's Cooking (Malmaud et al., 2015) is the largest collected dataset with 180K videos. For every video, there is a textual cooking recipe along with the ingredients. Additionally, YouTube's automatic speech recognition output is provided, which provides loose alignments between the textual recipe steps and the video. The dataset does not provide sub-activity labels or their locations in the videos.

Inria Instructional Videos (Alayrac et al., 2016) is a collected dataset, composed of the activities of "changing a tire", "performing CPR", "repotting a plant", "making coffee" and "jumping car". There are 30 videos for each complex activity and 47 sub-activities in total.

IKEA Furniture Assembly (IKEA-FA) (Toyer et al., 2017) is a dataset recorded from third person view. It is composed of 101 videos of assembling an IKEA table. There are 12 sub-activity classes, such as "pick leg", "attach leg" and "flip table". The dataset provides the temporal locations of the sub-activities along with the bounding box for the table. This dataset is one of the rare examples of recorded instructional datasets without cooking activities.

EGTEA Gaze+ (Li et al., 2018b) is a recorded egocentric dataset which contains 86 sequences from 32 subjects preparing seven different meals in a naturalistic kitchen environment. The tasks are making "pasta salad", "turkey sandwich", "bacon and eggs", "continental breakfast", "cheeseburger", "Greek salad", and "pizza." There are 106 sub-activity classes with an average duration of 3.2 seconds. This dataset provides audio, gaze tracking data, and pixel-level hand masks.

YouCookII (Zhou et al., 2018b) is a dataset collected from YouTube. There are 89 complex activities, all from cooking recipes. Each recipe has around 22 videos, in total there are 2000 videos. Each video is annotated with temporal boundaries of the recipe steps and their textual description.

CrossTask (Zhukov et al., 2019) is a dataset collected from YouTube. It has two types of tasks: 18 primary tasks which are fully annotated with the sub-activities' temporal locations, and 65 related tasks without any temporal annotations, which are collected to complement the primary tasks. There are 4.7K videos, 2.7K of which have temporal annotations. There are 107 sub-activities in the primary tasks.

COIN (Tang et al., 2019) includes videos collected from the 12 domains of "nursing and caring", "vehicles", "leisure and performance", "gadgets, "electric appliances", "household items", "science and craft", "plants and fruits", "snacks and drinks dishes", "sports", and "housework". There are 11.8K videos from 180 different complex activities from these twelve domains. There are 46K annotated segments for 778 sub-activity classes. It is the largest instructional activity dataset with temporal sub-activity annotations.

2.2.1.2 Daily Activities Datasets

Such datasets capture individuals' daily routines, usually in their homes. In Table 2.1, we present detailed statistics and comparisons with instructional activity datasets. The daily activity datasets are usually composed of recorded videos and are not limited to specific domains like cooking (Pirsiavash and Ramanan, 2012; Sigurdsson et al., 2018). Usually, these datasets are collected with an egocentric view (Pirsiavash and Ramanan, 2012; Damen et al., 2018). The Charades-EGO dataset (Sigurdsson et al., 2018) stands out with recordings that include paired third-person and egocentric view videos.

In the following, we present an overview of daily activity datasets ordered by the year of publication.

Activities of Daily Living (ADL) (Pirsiavash and Ramanan, 2012) is a dataset with egocentric recordings where 20 participants' daily activities are recorded in their homes. These recordings are not scripted. The dataset is annotated with 18 sub-activities such as "brushing teeth", "watching TV" and "cleaning house". The dataset also is annotated with 42 different objects and their interaction status, indicating whether they are being interacted with or not.

Charades (Sigurdsson et al., 2016) is a dataset recorded from a third-person view by around 260 participants in their own homes. The participants are asked to act out a script in front of the camera. Since the storyline is pre-defined, this dataset is considered to contain less natural looking actions. This is because in real-life scenarios, people make mistakes or look for things. The dataset includes around 9800 videos with 30 second duration on average. There are 66K action segments from 157 classes.

Charades-EGO (Sigurdsson et al., 2018) is a large-scale dataset of paired first-person and third-person videos. 112 participants are asked to record two videos using the scripts from the Charades (Sigurdsson et al., 2016) dataset. One video captures the person enacting the script from a third-person view. The other captures the same person repeating the same script through a camera fixed to their forehead. This dataset is interesting, as it could be used for modelling actions jointly from the first and third person's perspective. The recordings are not identical but follow the same semantic structure. The collection has 4000 pairs of videos.

Epic-Kitchens-55 (Damen et al., 2018) is a large-scale egocentric dataset with 432 videos from 32 participants in their kitchens. The participants record their daily routine over three days. The dataset has 55 hours of recording where the video durations range from 1 to 55 minutes. Each video is densely annotated with actions, e.g., "take plate", "open fridge". The dataset leads to a long-tailed distribution where only 149 action classes have 50 or more sample clips available. There are, in total, 125 verbs, 331 nouns, and 2513 actions.

Epic-Kitchens-100 (Damen et al., 2020a) is a recently extended version of Epic-Kitchens-55 (Damen et al., 2018) where the total duration of the dataset is increased to 100 hours with newly collected videos. The number of action classes is increased to 4025, the total number of segments to 70K.

2.2.1.3 Other Datasets

Finally, we overview other long-range video datasets, which include multiple actions with irrelevant order or are used for tasks where the temporal dynamics of these actions are not leveraged.

ActivityNet (Caba Heilbron et al., 2015) is a large-scale video dataset with 203 actions such as "painting furniture" or "vacuuming floor". The collected videos, ranging between 5 and 10 minutes, are annotated with the temporal boundaries of these 203 action classes. On average, there are 137 untrimmed videos per class, and there might be multiple actions annotated in some videos. The dataset is mainly used for the tasks of temporal action localization and captioning events.

Thumos (Gorban et al., 2015) is a dataset collected from YouTube for temporal action localization in long untrimmed videos. It consists of 7700 temporally untrimmed videos that may include any action from the 101 action classes of UCF101 (Soomro et al., 2012).

2.2.2 Action Anticipation

Predicting the future is a goal of many areas of computer vision. There has been a variety of research on predicting future frames (Liang et al., 2017; Lee et al., 2018a; Castrejon et al., 2019; Wu et al., 2020), human poses (Corona et al., 2020; Hernandez et al., 2019; Gui et al., 2018; Martinez et al., 2017), trajectories (Rasouli et al., 2019; Amirian et al., 2019; Sadeghian et al., 2019), action tubes (Singh et al., 2018, 2017), or active objects (Furnari et al., 2017). Others predict semantic (Luc et al., 2017) or instance segmentation (Luc et al., 2018), or optical flow (Jin et al., 2017) of future frames. Another task is predicting locations, e.g., of events in sports (Wei et al., 2014; Felsen et al., 2017), or of hands and objects (Fan et al., 2018). Further research directions are the anticipation of instruments in surgery (Rivoir et al., 2020), or even the prediction of future actions of dogs (Ehsani et al., 2018). However, in this section, we restrict our discussion mainly to predicting future human actions.

We now continue our overview with the topic of action anticipation, where the task is to predict the upcoming actions in videos. Anticipation is a perception task that makes an inference about the future given a sequence of observations. It is highly entangled with the advancements in action recognition, since the perception part is similar to the one required for recognition. It is also closely related to temporal action segmentation, as such algorithms provide high-level abstractions for visual data that could be directly used for anticipation.

In this overview, we study anticipation methods according to the prediction horizon they target, i.e., short and long-term. We first summarize early prediction approaches, which are focused on predicting ongoing, but not yet completed, actions as early as possible. Next, we present works that target "short-term anticipation" where the task is to predict the next action seconds before it starts. Finally, we review approaches that perform "long-term anticipation" into the future.

Dataset	Duration	#Seq.	#Seg.	Domain	Source	#C.A	#Actions	Year	View
GTEA	0.4h	28	0.5K	cooking	R	7	71	2011	egocentric
MPII Cooking	9.5h	44	5.5K	cooking	R	14	65	2012	3rd-person
TACoS	16h	185	8.8K	cooking	R	26	-	2013	3rd-person
50Salads	5.5h	50	0.9K	cooking	R	1	17	2013	top-view
Breakfast	77h	1712	11K	cooking	R	10	48	2014	3rd person
MPII Cooking 2	27h	273	14K	cooking	R	59	67	2015	3rd-person
Ikea-FA	4h	101	1.9K	furniture	R	1	12	2017	3rd person
EGTEA Gaze+	29h	86	10K	cooking	R	7	106	2018	egocentric
What's Cooking	3,000h	180K	-	cooking	C	-	-	2015	mixed
Inria Instructional	7h	150	-	mixed	C	5	47	2016	mixed
YouCookII	176h	2K	15K	cooking	C	89	-	2018	mixed
CrossTask	376h	4.7K	34K	mixed	C	83	107	2019	mixed
COIN	476h	11.8K	46K	mixed	C	180	778	2019	mixed
Tasty Videos	37h	2.5K	21K	cooking	C	2.5K	-	2019	top-view
ADL	10h	20	0.4K	mixed	R	-	18	2012	egocentric
Charades	69h	8000	68K	mixed	R	-	157	2016	3rd-person
Charades-EGO	82h	9848	66K	mixed	R	-	157	2018	mixed
Epic-Kitchens-55	55h	432	39K	cooking	R	-	2043	2018	egocentric
Epic-Kitchens-100	100h	700	70K	cooking	R	-	4025	2020	egocentric
Thumos	430h	7.7K	-	mixed	C	-	101	2015	mixed
ActivityNet	849h	27.8K	39.2K	mixed	C	-	203	2015	mixed

Table 2.1: We present comparisons of existing complex activity video datasets. We first present the recorded instructional activity datasets: GTEA (Fathi et al., 2011b), MPII-Cooking Activities (Rohrbach et al., 2012), MPII Cooking Activities 2 (Rohrbach et al., 2015), TACoS (Regneri et al., 2013), 50Salads (Stein and McKenna, 2013), Breakfast Actions (Kuehne et al., 2014b), Ikea Furniture Assembly (IKEA-FA) (Toyer et al., 2017), EGTEA Gaze+ (Li et al., 2018b), and collected instructional activity datasets: What's Cooking (Malmaud et al., 2015), Inria Instructional Videos (Alayrac et al., 2016), YouCookII (Zhou et al., 2018b), CrossTask(Zhukov et al., 2019), COIN (Tang et al., 2019), and Tasty Videos (Sener and Yao, 2019). We then list the faily activity datasets: Activities of Daily Living (ADL) (Pirsiavash and Ramanan, 2012), Charades (Sigurdsson et al., 2016), Charades-EGO (Sigurdsson et al., 2018), Epic-Kitchens-55 (Damen et al., 2018) and Epic-Kitchens-100 (Damen et al., 2020a). Finally we present other datasets where order is irrelevant: ActivityNet (Caba Heilbron et al., 2015) and Thumos (Gorban et al., 2015) datasets. The datasets are ordered based on the year they are released. Here "#Seq" corresponds to the number of videos, "#Seg." corresponds to the number of action segments, "Source" is either recorded, "R", or collected, "C", "#C.A" corresponds to the total number of different complex activities such as "making coffee", "#Actions" corresponds to the total number of action classes in the dataset. The datasets might include egocentric, 3rd person, top-view, or mixed views. The majority of the datasets are from the cooking domain.

2.2.2.1 Early Prediction

The goal of early prediction approaches is to enable early recognition of unfinished actions. Although they target inference on unfinished actions, early action prediction approaches are often wrongly referred to as "anticipation". Nevertheless, the two tasks are highly related; progress made in one can be useful for the other.

A variety of approaches for recognizing the current action/activity as early as possible have been proposed in the literature. In the following, we first summarize early action recognition methods, then we list approaches assessing the completeness of actions. Finally, we present methods that perform intention anticipation in long, complex sequences.

Early action recognition aims at recognizing ongoing actions from partially observed videos. One of the earliest attempts at formulating the concept of early action recognition is proposed by Ryoo (2011), who bases his solution on temporally integrated bag-of-words representations and models variations in the distribution of these histograms over time. Yu et al. (2012) implement a 3D extension of the implicit shape model (Leibe et al., 2008) to capture the spatio-temporal structure of local features and then perform pattern matching between testing and training videos. Hoai and De la Torre (2012) target early localization of starting frames of incomplete events by training a structured-output SVM on sequentially arriving observations. They constrain the SVM to ensure monotonically increasing classification confidence, so that partial observations are not classified more confidently than the more complete ones. Instead of directly classifying partial observations, Xu et al. (2015b) explore ranking all potential actions by utilizing auto-completion methods on prefix-candidate pairs. They temporally segment videos by clustering discriminative frame patches and use these segments to construct the prefix and candidate action pairs. Although small-scale datasets with limited action variations (Ryoo et al., 2010) are used in these eary works, with the advancement of deep learning, they are replaced by large-scale realistic video datasets (Soomro et al., 2012; Caba Heilbron et al., 2015) for early recognition.

Recurrent neural networks have been widely used to encode the sequential structure of videos (Ma et al., 2016; Zhou et al., 2016; Aliakbarian et al., 2017; Kong et al., 2018; Shi et al., 2018; Wang et al., 2019b; Gammulle et al., 2019b; Sun et al., 2019b; Wang et al., 2020a). To capture the progression of observations, Ma et al. (2016) use LSTMs with two ranking losses to encourage the rank of the correct action to increase, as the video progresses. One loss forces the classification score for the correct class to be monotonically non-decreasing with time, while the other ensures a non-decreasing margin between the detection score of the correct and the maximum score of the wrong classes. To encourage the correct predictions as early as possible, Aliakbarian et al. (2017) introduce a loss to penalize false negatives at the beginning of the sequence. They use a multi-stage LSTM architecture to extract context-aware features from the entire image and action-aware features using weighted activation maps (Zhou et al., 2016). These maps correspond to the image regions that are discriminative w.r.t. the action category. With a similar motivation to improve the recognition accuracy for early observations, Kong et al. (2018) use LSTMs augmented with a memory module to remember hard-to-predict samples. Their memory module averages latent features from the LSTM to create key-value pairs for querying the ongoing action class.

Instead of directly predicting action labels from partial observations, early recognition performance can be boosted by exploiting the future. One way of achieving this is predicting a future representation. Given partial observations in the form of a motion image for time t, Rodriguez et al. (2018) use a convolutional auto-encoder to hallucinate motion representations for subsequent frames. The hallucinated motion images are then fed to a CNN for classification. With a similar motivation, Shi et al. (2018) employ an RNN-based architecture to generate future features and feed them to a multi-layer perceptron (MLP) to predict the action class. Instead of learning future representations and their classification in two stages, Gammulle et al. (2019b) jointly learn action prediction and the synthesis of future representations using generative adversarial networks (GANs). They show that learning a context descriptor that is shared between early action recognition and representation prediction leads to good results. Another way of benefiting from the future is by transferring knowledge from complete to incomplete observations. Kong et al. (2017) reconstruct features of complete from those of incomplete sequences. Wang et al. (2019b) employ a knowledge distillation framework where a teacher model with access to complete videos transfers knowledge to a student model that only observes partial videos.

While Hoai and De la Torre (2012) and Ma et al. (2016) predict the starting points of actions after processing partial observations, De Geest et al. (2016) introduce the problem of online action detection, where the task is to classify frames in untrimmed, streaming videos as soon as they arrive. For the same online task, Xu et al. (2019b) incorporate future information by simultaneously predicting current and future actions. Similarly, Eun et al. (2020) only accumulate past information if it is relevant to the ongoing action to learn a more discriminative representation.

Early human activity recognition has also been investigated for actors and their interactions (Sun et al., 2019b) or group activities (Chen et al., 2020a; Yao et al., 2018). Some works rely on different modalities, such as skeleton data (Li et al., 2020; Ke et al., 2019a). In a very recent approach, Wang et al. (2020a) study early recognition in environments with multiple cameras, assuming that, due to some system limitations, data can only be received from one active camera at a time. They solve camera selection and early action recognition problems, considering missing or unobserved frames, using reinforcement learning.

Action completeness is introduced by Heidarivincheh et al. (2016), who propose a method that classifies sequences into complete and incomplete categories. In a concurrent work, Li et al. (2016) propose a joint classification and regression framework that predicts the class, start, and end time of the ongoing action. Later works propose solutions for regressing the time left to finish the action. Becattini et al. (2017) predict the action progress, which corresponds to the percentage of the action that is completed until the partial observation. They also localize the actions in the videos. Similarly, Twinanda et al. (2018) predict remaining surgery durations from partial videos. Several works attempt detecting the "completion moment" of actions, i.e., the exact onset of an action, e.g., turning the handle during a door-opening action. Heidarivincheh et al. (2018) predict the completion moment by accumulating frame-level votes. In a later work, Heidarivincheh et al. (2019) address the same problem in a weakly supervised setting using sequence labels, e.g., "complete" and "incomplete". Most of

these works rely on RNNs to encode sequences.

Intention prediction targets early prediction on sequences with multiple actions. For example, if a subject wears shoes and takes the house keys, this indicates that the person will go out.

In one of the early approaches, Pei et al. (2011) hierarchically decompose videos into activities using and-or-graphs over objects, actors and their interactions, which are then used to predict the intended actions, e.g., "using a microwave". Li et al. (2012) propose using "actionlets" – the atomic actions in activities, e.g., "making a phone call" which can be decomposed into the atomic actions of "reaching", "picking up", "answering" and "hanging up the phone". They detect actionlet segments in partial observations by discovering motion velocity peaks and use probabilistic suffix trees to learn actionlet patterns. Inferring others' intentions is an important subject for human-robot interactions and has been investigated for playing tennis (Wang et al., 2012), helping humans with daily activities (Koppula and Saxena, 2015), or predicting harmful events (Ryoo et al., 2015).

Recently, several works studied intention prediction on dedicated newly collected datasets. Wu et al. (2017) use on-wrist sensors, including a camera and an accelerometer, to observe human actions and collect an "intention" dataset. They use an RNN to encode the sequence over accelerometer and camera data. While the accelerometer data is cheap to store and compute, the camera frames are more expensive and are thus only triggered on demand by a policy network. Rhinehart and Kitani (2017) collect a dataset of videos of participants performing goal-seeking behavior in home-, office- or laboratory-environments using scripts like "obtain a snack and plate in kitchen, eat at dining room table." They aim to forecast the participants' long-term goals via an online inverse reinforcement learning approach. The goals in this dataset are rooms, e.g., bathroom or living room in homes, and lounge or kitchen for offices.

2.2.2.2 Short-term Anticipation

We now present a detailed overview of the related works on action anticipation, where the task is to predict upcoming actions before they occur. More formally, let τ_α be the "anticipation time", i.e., how many seconds in advance to anticipate the next action. Then the task of action anticipation is to predict upcoming action, τ_α seconds before it starts. In many recent approaches, τ_α is considered to be 1 second (Miech et al., 2019a; Furnari and Farinella, 2019) but could vary between several zero (Lan et al., 2014) and several seconds (Koppula and Saxena, 2015).

Early works are focused on anticipating simple movement primitives, such as "reaching", "moving", "placing", to plan ahead for robotic responses. Koppula and Saxena (2013a,b, 2015) model the interactions between human pose, environment and object affordances using spatio-temporal graphs. Lan et al. (2014) target anticipating human interactions such as "hugging". They use a multi-layer "moveme" hierarchy, which represents human movements at multiple levels of semantic and temporal granularities, and use a max-margin learning framework to relate information across different movemes for predicting future interactions.

One of the most popular sequence encodings are recurrent neural networks. The framework of Mahmud et al. (2017) jointly learns predicting the label of the next action and its starting time. Predictions are made based on the combined output of three branches, where two fully connected layers encode the object and recent action information, and an LSTM encodes the sequence. Pirri et al. (2019) use LSTMs with internal dynamic memory, which contributes to taking into account long-term past observations. Recently, Furnari and Farinella (2019) propose an anticipation architecture with two LSTMs, where one summarizes the past observations, and the other conditions on the hidden and cell states of the first to make predictions about the future at multiple anticipation times.

Similar to the early recognition literature, several action anticipation works exploit the future instead of making predictions based on past observations only. A majority of these works predict future features and perform next action classification on those. Vondrick et al. (2016) predict the visual representations of future frames by training a CNN on large-scale unlabelled video data. They later learn to recognize next actions from these features. To predict future frame features, they only rely on the last observed frame. Differently, Gao et al. (2017c) take a *sequence* of past frames as input, and output a sequence of future frame features. Their framework is composed of a video feature extractor, an encoder-decoder that predicts future features from past observations, a classification module to anticipate the action class, and a reinforcement module to reward the network for making a correct prediction as early as possible. Similarly, Zeng et al. (2017b) formulate predicting future representations as a inverse reinforcement problem. Zolfaghari et al. (2019) address the ambiguous nature of the future and predict multiple feature vectors corresponding to possible future outcomes along with their uncertainties. They fuse these multiple feature representations and feed them into an LSTM for predicting the future action. Different than predicting the future features, Tran et al. (2019) propose a knowledge distillation model. They train a teacher network for recognizing the future action, and a student network for next action anticipation. The teacher guides the student to attend relevant information needed for predicting the next action.

Predicting whether an action will occur or not is a more straightforward task but could be very critical for real-world scenarios like autonomous driving and accident prevention. Zhou and Berg (2015) propose one of the most straightforward scenarios for future prediction. Given an observation and snippets of two actions, they select the snippet most likely to happen next. Zeng et al. (2017a) focus on the accident anticipation problem where the task is to predict if a person will encounter an accident or not. In their LSTM-based framework, they also localize the region where the accident might happen. Suzuki et al. (2018) collect a dataset with a large number of near-miss accidents. They base their solution on a quasi-recurrent neural network (Bradbury et al., 2016) with a new loss that gradually learns early anticipation and outputs the probability of a future accident. Similarly, Neumann et al. (2019) present a method tested on two tasks: if and when a car will stop, and if and when a player will shoot a basketball. They study several representations and losses to make accurate predictions up to 10 seconds before an action happens. Manglik et al. (2019) focus on the "when" question and forecast the exact time-to-collision in the following six seconds.

The future is uncertain, as given some observation, there might be multiple future actions. Several works note that action labels in the standardized datasets are not complete to train models for action anticipation and propose dedicated losses or transferring knowledge from other sources. Furnari et al. (2018) consider anticipation as a multi-label learning problem with missing labels, as there might be multiple potential future actions for a given observation, but each is associated with only one action in the ground truth. They compare several loss functions and evaluation measures and conclude that Top-K losses (Berrada et al., 2018) produce encouraging results, as they are designed to produce small errors when the correct label is among the top-K predictions. Following the same multi-label missing labels definition, Camporese et al. (2020) use additional data sources to reduce the prediction uncertainty. They propose a knowledge distillation framework where a teacher network injects semantic prior information, e.g., word embeddings (Pennington et al., 2014), via label smoothing (Szegedy et al., 2016). Miech et al. (2019a) use additional data sources in the form of visual attribute classifiers. They fuse two approaches, one to anticipate the next action directly from the observations, the other to first predict visual attributes of the current frames and then to anticipate the next action based on these predictions.

Recent interest in egocentric vision has started a new line of approaches dedicated to such recordings. The main idea is to benefit from hand and object regions and their interactions. Shen et al. (2018) use a temporal attention module directed by different egocentric modalities such as gaze and hand masks in an LSTM architecture. Liu et al. (2020) propose a deep network that jointly predicts future hand trajectories, hand-object interaction regions on the last observable frame, and future action labels. In a very recent work, Dessalene et al. (2020) build object manipulation graphs using hand and object interaction regions to produce state representations, which model relations between hands and objects. They are then fed to an LSTM to model long-term video dynamics for anticipation. All these models require supervisory signals such as gaze, hand trajectories, interaction hotspots or hand masks during training.

All these approaches make future predictions in the form of categorical action labels. In Chapter 4, we present our anticipation work which generates sentence descriptions for upcoming actions. Our model can be used for next action anticipation and can also be extended for multiple step predictions into the future. Similar to us, recently, Mahmud et al. (2020) output sentence-based generations for the next actions by extending action anticipation framework from Mahmud et al. (2017).

2.2.2.3 Long-term Anticipation

The approaches we reviewed until now target short-term and mostly next action anticipation. This type of anticipation is shown to be successfully performed by processing only the recent observations (Furnari and Farinella, 2019). In recent years, research on predicting multiple future actions gained attention, as e.g., exercised in "dense anticipation", which refers to predicting multiple future actions and their durations for long horizons of up to several minutes. However, for this type of anticipation, models need to understand long-range past in complex activities. We now review approaches which predict long-term future by exploiting

long-term past observations.

One of the first approaches for dense anticipation is presented by Farha et al. (2018). Their method is composed of two stages. The first stage is assigning action labels to each frame using some temporal action segmentation algorithm. The second stage uses these labels as input for two future prediction algorithms, one using RNNs, the other CNNs. The RNN conditions on the past action sequence and predicts the remaining duration and label of the current action, and it recursively predicts future actions and their durations. The CNN-based method makes future predictions in the form of a matrix. Gammulle et al. (2019a) point out the limitations of LSTMs for capturing long-term dependencies and propose using memory networks with LSTMs. They use two memory modules, one for their CNN-based frame-level features and the other for action labels of sequences. Ke et al. (2019b) argue that recursive predictions might accumulate and propagate errors and that the CNN-based solution from Farha et al. (2018) introduces many parameters and requires a predefined scale for the predicted matrix, which limits the prediction horizon. They propose an approach that, given a future time point, makes a time-conditioned future action prediction. As such, their model only relies on observations to make future predictions and thus does not accumulate anticipation errors.

Unlike methods mentioned earlier, which work in a two-stage setting and either use the outputs of a segmentation algorithm (Farha et al., 2018; Ke et al., 2019b) or the ground truth action labels (Gammulle et al., 2019a), our work, presentend in Chapter 5, performs dense anticipation in a single stage (Sener et al., 2020). Similarly, Farha et al. (2020) propose a sequence-to-sequence model that encodes visual features to predict future actions and their durations recursively. They also employ a cycle consistency module that, given the predictions for future actions, predicts past actions and their durations, which is shown to improve the prediction performance.

Given an observation, multiple possible actions might occur in the future. Uncertainty in the future led researchers to develop models that can make multiple dense future predictions. Mehrasa et al. (2019) propose a variational auto-encoder to predict a distribution over the future actions and their starting times. Using these distributions, they can sample multiple sequences for the future. Similarly, Farha and Gall (2019a) employ the RNN from Farha et al. (2018) and separately model probability distributions of future actions and their durations, and use them to generate possible future predictions. In a very recent work, Zhao and Wildes (2020) propose a discrete sequential generative adversarial network (GAN) for predicting realistic future sequences in the form of action labels and their durations. They combine the GAN with a distance regularizer that encourages the generator to produce diverse sequences.

Different than the aforementioned dense anticipation methods, which predict future action labels and their durations, Nagarajan et al. (2020) only predict a list of actions that are likely to occur in the long-term future. They first construct a topological graph of the environment from egocentric videos, where nodes correspond to action zones and edges to people visiting these zones. Using graph convolutional operations, they relate information across zones and create a video representation to predict multiple future actions in egocentric videos.

2.2.3 Temporal Action Segmentation

Temporal action segmentation from videos aims at dense labelling of video frames with action classes. The datasets frequently used for segmentation are instructional activity datasets like Breakfast Actions (Kuehne et al., 2014b), GTEA (Fathi et al., 2011b; Li et al., 2018b), 50Salads (Stein and McKenna, 2013) or Inria Instructional Videos (Alayrac et al., 2016), where the long-range temporal modelling is important. In videos there might be many garbage or background frames that are irrelevant to actions, i.e., no action of interest occurs. Related works in our review achieve dense labelling by using additional mechanisms such as explicit modelling of a background class (Hoai et al., 2011; Cheng et al., 2014; Richard and Gall, 2016; Lei and Todorovic, 2018; Sener and Yao, 2018) or implicit modelling of the background as the group of frames that are not covered during an action (Fathi and Rehg, 2013).

The outputs from temporal action segmentation algorithms could be used as a prior step for several applications. For example, Soran et al. (2015) propose generating notifications about missing actions given unsegmented egocentric stream. They segment the input video stream to issue notifications about these actions. Similarly, in action anticipation, several approaches utilize segmentation methods to represent past observations with action labels (Farha et al., 2018; Ke et al., 2019b; Gammulle et al., 2019a). This is because such labels contain high-level semantic information which is more preferable than visual features for anticipation tasks (Sener et al., 2020).

The success of temporal action segmentation methods is closely related to advancements in traditional action recognition, particularly to feature extraction and action representation. The pipeline of temporal action segmentation is usually composed of two stages. First, frame- or snippet-wise features are extracted, and then a long-range temporal modelling approach is used to model the sequence of actions. Such temporal models capture high-level patterns. In the literature, a variety of such models are utilized for temporal action segmentation, including conditional random fields (Hoai et al., 2011), hidden Markov models (Lea et al., 2016), recurrent neural networks (Singh et al., 2016) and recently temporal convolutions (Lea et al., 2017).

We now continue our overview by summarizing temporal action segmentation methods that have been proposed in the literature. Two particularly important factors, when comparing temporal action segmentation strategies, are the type of the employed temporal models and the annotations provided during training. Since in this dissertation, we propose both an unsupervised and a supervised segmentation method, we now summarize related works based on the extent of training annotations they use. In the following, we overview supervised, weakly supervised, and unsupervised temporal action segmentation strategies.

2.2.3.1 Supervised Approaches

Supervised temporal action segmentation methods require frame-wise action labels during training. Although there are several early works that are tested on short-term activity videos (Hoai et al., 2011), the majority of the following approaches are tested on at least one of the instructional complex activity datasets like Breakfast, GTEA, or 50Salads.

Early approaches rely on hand-crafted features and bag-of-words-like representations. Usually, these works involve extensive pre-processing and might be slow, especially for long-range videos. Fathi et al. (2011a) represent actions as relations between objects and hands. They first assign action scores to frames using classifiers learned on features capturing such relations. Given the action scores, they learn further classifiers to recognize complex activities, like "making tea" and train a conditional random field for each complex activity to assign action labels to sequences. Although, their action segmentation solution is prone to error propagation due to wrong complex activity recognition, they show that objects-wise representations are very beneficial for action recognition and segmentation. In a similar spirit to using object-centric representations, Fathi and Rehg (2013) focus on changes in object states through manipulation. They first discover changing regions in frames based on motion and color changes. Next, they train SVM-based state-specific detectors over these regions to represent each frame by a vector of detector responses. Finally, they segment videos by localizing the initial and final segment frames with transition constraints between actions, e.g., "pour milk" cannot occur after "close milk". Hoai et al. (2011) treat the segmentation problem as finding a set of change points for time series, given action models learned using a multi-class SVM. They segment videos based on maximizing the difference between the SVM scores of the top two action classification scores. However, this model does not capture the temporal relations between actions. Contrarily, the work of Cheng et al. (2014) models long-term relations in sequences. They split videos into snippets, each represented with visual words, and use a Bayesian non-parametric language model to capture temporal dependencies among the sequence of snippets (words), independent of their action classes. They integrate this temporal model and action classification into a framework to jointly segment and classify video sequences.

Another line of early methods use probabilistic models to predict the most probable sequence of actions. Vo and Bobick (2014) use a stochastic context-free grammar to represent the temporal structure of complex activities. Given the grammar, they construct a graphical model, where the outputs of action detectors are the observed nodes, and the action durations are the hidden variables. They use message passing to parse a video by computing the start and end times of all actions if they occur. Kuehne et al. (2014b, 2016) combine a set of hidden Markov models, which model the actions, with a context-free grammar to determine the most probable sequence of actions. Richard and Gall (2016) propose a probabilistic model composed of three components, where an action model maps visual features to action probabilities, a language model provides sequence level action probabilities for action segments, and a length model predicts the duration of these segments. They use dynamic programming to obtain the most likely segmentation that maximizes the joint probability of these components.

Deep learning-based features coupled with temporal models have proven to be successful in recent years. Lea et al. (2016) propose a CNN which uses large 1D convolutional filters for capturing the spatio-temporal feature relations. They segment videos with a semi-Markov model using the features from this CNN. However, this model cannot capture long-range temporal structure, as an action is only conditioned on its previous segment. Singh et al.

(2016) propose using an RNN-based temporal modelling method. They split videos into snippets and pass them through a multi-stream network with two appearance and two motion streams (whole frame and cropped by human tracker) to compute visual representation. These representations are then fed into a bi-directional LSTM, which predicts action labels for each snippet considering temporal dependencies.

In recent years, temporal convolutional networks (TCNs) gained attention in temporal action segmentation, as they can capture long-term dependencies and are faster than RNN-based models. Lea et al. (2017) are the first to use temporal convolutional networks for action segmentation. They propose a hierarchy of 1D temporal convolutional and deconvolutional kernels in an encoder-decoder framework. The work from Ding and Xu (2017) replaces the decoder in the framework of Lea et al. (2017) with a bi-directional LSTM. By that they end up with a hybrid temporal convolutional and temporal recurrent network. However, this network yields high computation costs due to the recurrences. Lei and Todorovic (2018) also build on top of the work of Lea et al. (2017) by replacing the temporal convolutions with deformable temporal convolutions and adding a residual stream to the encoder-decoder model. The residual stream processes videos in full temporal resolution, while the other stream captures temporal context at different scales. Although the aforementioned approaches based on TCNs work on the entire video – referred to as *full resolution* – these methods downsample the videos to a temporal resolution of a few frames per second beforehand, which might cause a loss of fine-grained details. In contrast, Farha and Gall (2019b) propose a multi-stage hierarchical temporal convolutional network, MS-TCN, that works on exact full resolution video. Each stage consists of multiple temporal convolutional layers with 1D dilated temporal convolutions and outputs an initial prediction, which is refined by the following stages. This work improves the segmentation performance compared to Lea et al. (2017) and Lei and Todorovic (2018) by a large margin.

As we remarked before, most temporal action segmentation networks operate on two-stages: feature extraction and temporal modelling. Although several approaches (Lea et al., 2016; Singh et al., 2016) propose improving the feature quality with their CNN models, the focus of all these works is mainly to enhance temporal modelling. Conversely, Mac et al. (2019) propose a method purely for learning spatio-temporal feature representations better capable of modelling fine-grained actions. Instead of using optical flow in a second stream to model the temporal relations, they use locally-consistent deformable convolutions to capture motion information. Using the features from this network with existing segmentation works (Lea et al., 2017; Lei and Todorovic, 2018) improves the performance. With a similar motivation, Gammulle et al. (2019c) use a recurrent segmentation architecture composed of two GANs to improve visual representations. While one GAN takes RGB frames as input, the other works on supplementary modalities such as depth maps or optical flow. Both GANs are trained to produce similar vector representations for the current action, called action codes, which are used for classifying the video frames. The videos are processed sequentially by feeding the action codes from the previous frame as input to both GANs at every step to provide context.

A very recent collection of methods targets improving the performance of existing tem-

poral action segmentation algorithms. They develop special modules that can either be integrated into a backbone segmentation method or applied to its outputs to improve the segmentation quality. In a very recent work, Chen et al. (2020b) argue that spatio-temporal variations of human actions – referred to as *different domains* – are the reason for low performance in supervised action segmentation, as training a model in one domain and testing in another will fail due to the variations across videos. They improve a source model's performance on target data using two self-supervised auxiliary tasks to decrease differences between the feature spaces of the source and target domains. One task predicts which domain a single frame's feature vector comes from, while the other predicts domain labels for a shuffled *sequence* of source and target segments. They integrate their self-supervised model to the MS-TCN of Farha and Gall (2019b) and, without using additional labeled data, significantly improve the action segmentation performance. Similarly, Zhang et al. (2019) propose a bilinear pooling module that can be integrated into TCNs to serve as a computationally efficient feature fusion operation, e.g., by replacing the last 1×1 convolution layer in the first stage of MS-TCN (Farha and Gall, 2019b). Given an initial segmentation, e.g., from some existing action segmentation algorithm, Huang et al. (2020) refine it using graph convolutional networks (GCNs) by modelling temporal relations between actions. However, the improvement of their outcome strongly depends on the quality and degree of fragmentation of the initial segmentation.

Another emerging idea to improve the performance of existing segmentation algorithms is to correct the segmentation results at the boundaries. Wang et al. (2020e) raise concerns over the boundary ambiguity and over-segmentation issues in existing works and specifically propose a module that could be used with multi-stage segmentation algorithms (Farha and Gall, 2019b). Their module enables the later stages to focus on ambiguous frames. Additionally, a newly introduced pooling operator smooths noisy boundary predictions with confident ones. A similar boundary refining model is proposed by Ishikawa et al. (2020) that could be used with any action segmentation algorithm.

2.2.3.2 Weakly Supervised Approaches

As discussed earlier, there is a diverse set of fully-supervised approaches for temporal action segmentation. However, such approaches require action labels and boundaries to be provided during training, which can be tedious to obtain.

In this section, we provide an overview of weakly-supervised approaches that do not need the action boundaries. We divide such methods into three categories. Methods in the first one receive supervision as an ordered list of sub-activities, which are called *transcripts*. In the second one, an unordered list of sub-activities are used, which are called *action sets*. The last category is the group of segmentation methods that use complementary *textual data* such as narrations to provide temporal constraints for segmentation.

Transcripts: ordered list of sub-activities:

Given a video and its transcript, the task is to locate start and end times of the transcript actions in the video. This type of supervision significantly reduces the cost of annotating

videos. Bojanowski et al. (2014) are one of the first to use transcript-based supervision. However, they target the task of alignment between frames and transcript actions, as they assume the transcripts will be provided both during training and testing (see Section 2.3.5). Later, several approaches adopt transcript-based supervision for temporal action segmentation. We divide them into iterative two-stage and single-stage methods.

We start with two-stage approaches, where the segmentation is achieved with iterative refinement of previous predictions. Kuehne et al. (2017) extend a supervised approach from an early work of theirs (Kuehne et al., 2016) to a weakly supervised setting. In their framework, the actions are modeled by a set of hidden Markov models (HMM), while the observations are modeled by Gaussian mixture models (GMM). They start with segments uniformly initialized based on the transcripts and iteratively refine them. Richard et al. (2017) build on this idea and replace GMMs with recurrent neural networks. They also further divide the actions into snippets to capture their finer-detailed characteristics. Ding and Xu (2018) extend the temporal convolutional networks from Lea et al. (2017) by adding lateral connections between the encoder and decoder layers. They use a soft labelling mechanism at the segment boundaries and iteratively refine the segmentation.

The two-step approaches are sensitive to initialization and might suffer from an oscillation effect, as these models are learned incrementally. Single-stage approaches allow for direct learning of segmentation. Huang et al. (2016) propose an extended version of connectionist temporal classification (CTC) (Graves et al., 2006) for aligning the transcripts with video frames with consistency constraints. They enforce the frame-wise similarities to be consistent with the action alignments. This reduces the space of possible paths and avoids degenerate segmentations, which might occur due to a large number of frames in long videos. Richard et al. (2018b) propose a method, Neuralnetwork-Viterbi, which considers Viterbi decoding as part of the loss function to train a segmentation network. The Viterbi algorithm generates pseudo ground truths over the output probabilities of the network, which are then used for computing the loss. This method provides significant improvements over the previous methods, however, it is costly to train due to the Viterbi decoding. Chang et al. (2019) propose a differentiable alignment loss to discriminatively model positive and negative transcripts. A similar discriminative training is proposed by Li et al. (2019) who build their solution on Neuralnetwork-Viterbi with ordering constraints. Unlike the randomly selected negative transcripts by Chang et al. (2019), Li et al. (2019) generate candidate valid and invalid segmentations using a segmentation graph, where invalid candidates violate the transcripts. They recursively estimate each candidate's segmentation energy and formulate a new loss based on the energy differences between valid and invalid candidates. They improve over previous works by a significant margin, however, the training gets more expensive. Criticizing the state of the art being costly to train, Souri et al. (2019) employ a sequence-to-sequence network that performs comparably to earlier works, but is much faster. Their network is composed of two branches, where one predicts transcripts and action durations, while the other outputs frame-wise predictions. The predictions from the two branches are used to compute a mutual consistency loss to enforce similar predictions.

Action sets:

In this setting, the methods assume that a set of action labels is provided during training, without knowing the temporal location, the order, or how often they occur. This type of labelling could arise in the form of meta-tags e.g., from video-sharing platforms. Richard et al. (2018a) are the first to propose a weak segmentation model using action sets. Their framework is composed of three components similar to Richard and Gall (2016), with action, length, and sequence models, and aim to find the most likely segmentation. They generate multiple transcripts using context-free grammars to restrict the search space and convert the problem into a weakly supervised segmentation setting with multiple transcripts. The most likely segmentation is achieved by using a Viterbi algorithm. However, this work cannot generate all possible sequences of a set of action labels, which might degrade segmentation quality. Fayyaz and Gall (2020) learn a segmentation network that directly uses provided annotations for learning. They start by splitting videos into regions and estimate action probabilities and temporal lengths for them in one branch, and use a second branch to produce frame-wise predictions. They measure the consistency of the frame-wise predictions w.r.t. region predictions, which significantly improves the model's accuracy. They also define several losses and regularizers for purposes, such as encouraging temporally consistent predictions for neighboring regions, or regularizing region lengths. Li and Todorovic (2020) use a set-constrained Viterbi algorithm to generate more accurate pseudo ground truths and an n-pair loss to minimize the distance between pairs of training videos sharing action classes in their action sets.

Textual data:

Text data is straightforward to obtain, as it comes for free with videos in the form of scripts, subtitles or narrations. It is frequently used for video and text alignment (Malmaud et al., 2015; Bojanowski et al., 2015) and step localization (Alayrac et al., 2016; Zhukov et al., 2019). There are also several works that make use of such data for weakly supervised action segmentation.

Duchenne et al. (2009) propose using subtitles and scripts as weak supervision to segment actions from movies. They align the subtitles and scripts to provide coarse temporal localization of actions. Then they segment the videos based on the aligned regions with an additional margin, assuming that there will be a single action in each segment. Finally, they use discriminative clustering to temporally localize the actions in each segment, labelling the frames outside the localized area as background. As opposed to Duchenne et al. (2009), who do not use any temporal modelling, Sener et al. (2015) and Fried et al. (2020) capture the temporal relations between the actions. Sener et al. (2015) first generate object proposal segments from a collection of videos of the same complex activity and compute visual vocabularies. Using those together with textual vocabularies, which are computed over narrative text, they represent each frame by a binary histogram over visual and textual words. To identify the sub-activities shared among the videos, they utilize the generative beta process mixture model from Fox et al. (2014), but on binary observations. Fried et al. (2020) propose an approach that uses canonical step ordering and transcribed narrations in videos as super-

vision for segmentation. Canonical step ordering refers to the sequence in which the steps of complex activities are typically performed. It is the same for each video from a complex activity. They propose modelling the segment duration, location, order, and features by a semi-Markov model. Although they do not use the narrations during testing, they use the canonical ordering, as these constraints affect their model parameters.

The main disadvantage of models employing textual data is the assumption that *all* the videos are accompanied by temporally aligned text. However, text data might not always be aligned or even completely missing.

2.2.3.3 Unsupervised Approaches

The related works on activity segmentation we discussed earlier either require full or weak supervision in the form of sub-activity lists or textual data. In this section, we discuss unsupervised action segmentation approaches that neither require any action labels or their temporal boundaries, nor any textual data.

Very early approaches on unsupervised segmentation do not consider global temporal modelling. The majority of them are focused only on change-point detection using sliding windows in the time dimension, e.g., segmenting music signals (Harchaoui et al., 2009) or financial data (Xuan and Murphy, 2007). Zelnik-Manor and Irani (2001) segment videos by clustering frames using normalized cuts.

Unsupervised segmentation of sequences has been popular on motion capture data, which is less complex than video data. Barbič et al. (2004) propose a solution based on probabilistic principal component analysis and place a cut when the distribution of poses is observed to change. Zhou et al. (2008, 2012) propose aligned cluster analysis, which combines k-means with dynamic time alignment. They measure the similarities between two temporally aligned segments through dynamic time warping kernels. These measures are then used with k-means to determine the segment clusters. Fox et al. (2014) examine multiple motion capture recordings at the same time to extract global features of actions that are modeled with HMMs. The feature selection is achieved via a beta process model. They define a segment when the recordings change from one to another HMM. Similarly, Wu et al. (2015) propose a topic-modelling-based approach for segmenting human actions using skeleton data and object features. They divide sequences into overlapping splits and use k-means to cluster them for computing action-words. Frames are then represented with action-words and complex activities with a set of action-topics. They use Gibbs sampling to infer complex activities from action-words.

We propose an unsupervised temporal action segmentation approach (Sener and Yao, 2018) that works solely with visual data without any supervision, presented in Chapter 3. Following our segmentation approach on instructional activity videos, several new approaches have been proposed. In our setting, we assume we are given a collection of videos, all of the same complex activity, and the number of sub-activities as input. The recent related works also follow similar settings and provide further improvements. For example, Kukleva et al. (2019) propose an extension of their method to work on collections of *different* complex activities, while Aakur and Sarkar (2019) do not require the number of sub-activities as

input.

Rivoir et al. (2019) propose using our segmentation framework as a preliminary task for predicting the remaining time of long surgery videos. They extend our work (Sener and Yao, 2018) by replacing linear mapping for discriminative appearance modelling with a CNN-LSTM-based mapping. They later either use the features from this mapping or the segmentation output to predict the remaining surgery time. Our model has a notable limitation in that it cannot model repeated sub-activities, as we treat the ordering of sub-activities as a sequence of permutable steps. Goel and Brunskill (2018) propose a generative hierarchical Bayesian model that allows for repeated sub-activities. They use Gibbs sampling to perform the inference of the latent variables. However, this model assumes that all the videos follow the same underlying ordering. They also introduce several new metrics to evaluate repeated structures, as well as over- and under-segmentation.

Recent works propose exciting new directions and significant improvements to unsupervised segmentation. Kukleva et al. (2019) propose an approach where they first learn continuous temporal embeddings of frame-wise features. They then cluster the embedded features and decode videos based on ordered clusters of embedded frame-wise features using the Viterbi algorithm. Different than our work (Sener and Yao, 2018) which gets the complex activity label as input, Kukleva et al. (2019) propose a version of their method without this requirement by grouping videos into complex activity clusters in a pre-processing stage. Aakur and Sarkar (2019) propose a self-supervision-based approach to detect action boundaries for unsupervised segmentation. They recurrently predict features of the next frame and compute the difference to the observed features to determine action boundaries. Instead of using pre-computed visual features, they propose joint training with a CNN (Simonyan and Zisserman, 2014b) to encode the frames. Different than other unsupervised works (Sener and Yao, 2018; Kukleva et al., 2019), this method automatically determines the number of sub-activities from boundary transitions. VidalMata et al. (2021) propose a joint visual-temporal learning model for unsupervised action segmentation. They separately train the temporal embedding from Kukleva et al. (2019) and an encoder-decoder visual embedding network which predicts the features of subsequent frames. These two embedding networks are in turn trained in a joint framework to learn useful representations of visual and temporal attributes. The embedding space is then used for clustering to form the action segments. They achieve significant improvements in performance on the Breakfast Actions dataset.

Recently, Fried et al. (2020) proposed a method that can be used for supervised, weakly supervised and unsupervised action segmentation of videos. They train a hidden semi-Markov model in an unsupervised way to maximize the feature likelihoods for all videos. In their work, they systematically evaluate how much models improve with the degree of supervision, e.g., using canonical ordering, transcripts from narrations or full supervision. They only report results for the CrossTask dataset (Zhukov et al., 2019), as it includes narrations and canonical orderings for complex activities, which allow for their systematic evaluation.

2.2.4 Temporal Action Detection / Localization

The goal of temporal action localization is to search actions throughout a video and localize them. It is closely related to temporal action segmentation, however, action detection approaches do not necessarily target dense labelling of video frames. Moreover, these methods usually target localizing actions in untrimmed videos from the Thumos (Gorban et al., 2015) and ActivityNet (Caba Heilbron et al., 2015) datasets, and typically do not consider the temporal structure of sub-activities. Temporal action localization tasks on instructional datasets are mostly focused on discovering key steps (sub-activities) in the form of single timestamps. In this section, we first give a brief overview of temporal action localization works in general. Then, we overview the approaches on key-step localization.

2.2.4.1 Temporal Action Localization

Temporal action localization in untrimmed long videos usually does not consider temporal connections between actions but focuses on localizing proposal-based predictions for individual actions. In the following, we overview supervised, weakly supervised, and unsupervised works, respectively.

Supervised Approaches:

Early works produce proposals based on sliding windows and focus on designing hand-crafted feature representations (Gaidon et al., 2013; Jain et al., 2014), while more recent works use deep networks.

Many of the recent methods adopt a two-stage approach that generates candidate action proposals in a first, and classifies them in a second stage. A common way to generate proposals is using sliding windows. For example, Shou et al. (2016) rank sliding-window-based proposals and use non-maximum suppression for post-processing. However, such generic proposals do not produce precise temporal boundaries of actions. As such several works propose refinement strategies. Shou et al. (2017) predict at the frame level and combine frames with proposals for better localization. Xiong et al. (2017) group snippets based on actionness scores. Gao et al. (2017b) employ temporal coordinate offset regression. Since sliding window-based approaches produce redundant proposals, several works focus on better methods for proposal generation. Buch et al. (2017) densely predict proposals in a single pass using a recurrent architecture. Chao et al. (2018) utilize 2D object detection architectures for proposal generation in the temporal domain. Lin et al. (2019b) target predicting temporally precise proposals based on boundary probabilities. Instead of improving proposal generation, several works focus on accurate action classification. Zhao et al. (2017) model the temporal structure of actions using structured temporal pyramids. Heilbron et al. (2017) utilize action-object and action-scene relationships to prune out unrelated actions.

Some works perform proposal generation and classification simultaneously in a single stage. Yeung et al. (2016) use reinforcement learning with an RNN-based agent. Long et al. (2019) optimize the temporal scales of proposals using Gaussian kernels. Xu et al. (2020) employ graph convolutional networks to fully exploit the video context. Several works model

the evolution of actions explicitly. Yuan et al. (2017) use a two-stream network to generate frame-wise scores and search for the structured maximal sum of these scores. Hou et al. (2017) cluster videos into temporally connected snippets and discover actions as sequences of ordered snippets.

Weakly supervised approaches:

These works aim at learning models on untrimmed videos annotated only with action classes, without relying on their temporal locations. Some works split videos into short clips and perform clip selection. Wang et al. (2017) use a ranking module to select the important clips, and Nguyen et al. (2018) use an attention module to identify a sparse set of clips which minimizes the classification loss. Shou et al. (2018) predict temporal action boundaries by encouraging action scores outside the action segments to be smaller than on the inside. Nguyen et al. (2019) show that explicitly modelling background frames improves the performance in the weakly supervised setting.

Unsupervised approaches:

Gong et al. (2020) address temporal action localization without using any annotation. They alternate between a clustering stage to get pseudo-action labels, which are then used as input for a localization stage to detect the actions temporally.

2.2.4.2 Temporal Action Localization on Daily and Instructional Videos

Contrary to the setting of common action localization datasets, several works attempt localizing actions on instructional and daily video datasets. Zhou et al. (2018b) localize procedural segments based on their visual appearance and temporal relations, irrelevant of their action class, on the instructional YouCookII dataset. Their model comprises a context encoder based on bi-directional LSTMs, a proposal generation network utilized from object detection works, and an LSTM-based sequential prediction module that selects the final proposals. Damen et al. (2020a) generate action proposals (Lin et al., 2019b) for the daily activity videos in the Epic-Kitchens-100 dataset, and use SlowFast (Feichtenhofer et al., 2019) for classifying and selecting the final list of action proposals. Damen et al. (2020a) also start a temporal action localization challenge on this dataset, which might motivate future research to employ the sequential relationships in such activities rather than independent localization of actions.

2.2.4.3 Key-step Localization

Different than temporally localizing actions with their start and end times, these works propose identifying single timesteps for these actions, which are also referred to as key-steps.

Alayrac et al. (2016) target finding such steps in a collection of narrated videos of the same complex activity. Their method is composed of two discriminative clustering-based solutions, one on text and the other on video. The first clustering is performed over the narrations to find the total number of sub-activities, K, and their canonical ordering per complex activity in an unsupervised way, by formulating it as a multiple sequence alignment problem. The second

clustering is over visual features and is formulated as a discriminative clustering approach using the narrations as temporal constraints. Their model predicts K steps in each video which follow a strict ordering. Similarly, Zhukov et al. (2019) assume narrated instructional videos will be provided during training. Different than Alayrac et al. (2016), they do not parse the narrations, but assume that a canonical ordering of sub-activities per complex activity is already provided. Their goal is to discover a set of classifiers for objects, verbs, and prepositions extracted from the categorical tags of sub-activities in this ordering. Their objective is to simultaneously optimize key-step locations and classifiers over all complex activities using constraints from the narrations. Their predictions, for each complex activity, follow a canonical ordering, provided both during training and testing.

Subset selection aims at discovering a small subset of the most informative data points and is frequently used in document summarization. It is recently also employed for key-step localization in instructional videos using varying degrees of supervision. Elhamifar and Naing (2019) employ subset selection for unsupervised localization on a collection of videos of the same complex activity. They partition videos into segments using superframes segmentation (Zhang et al., 2016). They learn an HMM with latent states, some of which correspond to key steps, while the rest corresponds to background. Using dynamic programming, they find a subset of hidden states and their assignments to video segments. They then perform multiple sequence alignment, similar to Alayrac et al. (2016), to obtain a final sequence of key steps. Different than early works, this method allows for missing steps, repetitions or deviations from the canonical ordering. Unlike existing key-step localization works, which assume that the videos are from the same complex activity, Elhamifar and Huynh (2020) develop an unsupervised method to discover the key-steps from a collection of videos of multiple complex activities. In their deep neural network, a spatial attention module learns attending informative regions in frames, and an unsupervised subset selection component localizes key-steps, whose outputs are used as pseudo labels for a key-step localization and a complex activity classification module. This network learns complex activity-dependent attention features and discovers and localizes key-steps. Naing and Elhamifar (2020) learn key-step localization given videos partially annotated with a small subset of key-steps. Given training labels in the form of one frame from each sub-activity, Xu and Elhamifar (2019) develop a supervised subset selection framework.

2.3 Joint Video and Text Modelling

When reasoning about what they do and will do, vision and language are primary references for humans to derive knowledge from. The predominant approach for video analysis applications is to model the visual recognition problem as a task of classifying videos or frames into some number of fixed visual categories. However, with rapid advances in neural networks and availability of large-scale data in recent years, research in computer vision has focused on developing approaches that combine the visual domain with natural language to enable language-based predictions. In this section, we first overview language representation approaches, then we give a short overview of modelling instructional textual data, and present

methods that model video and text jointly. Finally we briefly summarize video captioning on complex activities and temporal video-text aligment methods.

2.3.1 Representing Textual Data

Throughout the years, various approaches have been used to represent sentences, including one-hot encodings, recurrent neural networks, and, more recently, transformer architectures. In visual tasks, pre-training models is shown to offer significant improvements over learning models from scratch (Karpathy et al., 2014; Carreira and Zisserman, 2017). Similarly, to improve language processing tasks, several works propose employing language model pre-training. In the following, we first give an overview of standard sequence modelling architectures, and then review pre-trained language models that can be used for computing generic sentence representations.

2.3.1.1 Recurrent Neural Networks (RNNs)

Recurrent layers (Elman, 1990) are powerful building blocks for modelling natural language and are also actively used in video representation (Donahue et al., 2015; Furnari and Farinella, 2019). The central units in RNNs are the input and hidden states at each time step. RNNs handle sequential input (sentences) by having a recurrent rule that takes the input and hidden state at time t (word) and updates the hidden state at time $t+1$. The hidden states act as a running summary of all information in the previous steps until that time step t. This rule is shared among all time steps.

Mikolov et al. (2010) applied RNNs in language modelling and brought substantial advancements across a wide range of language tasks. However, RNNs suffer from short-term memory and are unable to associate information over long sequences. As a solution, to address the limitations of vanilla RNNs, two variants have been proposed: Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014). The LSTM is designed to avoid the long-term dependency problem of RNNs using a gating mechanism that controls memoizing. In addition to a hidden state, LSTMs maintain a memory cell state. At each time step, the LSTM can choose to write to, read from, or reset the cell using gating units (input, output, and forget), which control how much information will flow into and out of the memory. GRUs do not use separate memory cells, but instead, use gating units to modulate the information flow inside each recurrent unit. GRUs contain fewer training parameters and use less memory, while LSTMs lead to better results, particularly for long sequences.

Although RNNs can handle variable-length sequence input and the model size does not increase with the size of the input, they are slow in computation due to recurrences, as they do not allow for parallel computation. Moreover, even with the variants such as LSTM, RNNs have difficulties in accessing information from a long time ago.

2.3.1.2 Transformer Language Model

The transformer architecture (Vaswani et al., 2017) is a non-recurrent encoder-decoder design that uses a series of self-attention layers. In this architecture, a self-attention mechanism computes similarity scores between inputs in a sequence by attending to all other positions. To retain the temporal order in sequences, it uses positional embeddings. It reduces the training time for learning sequence models via parallel computation and reduces the performance drop for long dependencies as every input is connected to all other inputs. Transformer architectures have become the de-facto standard for natural language processing tasks and are recently getting popular in computer vision (Zhou et al., 2018c; Dosovitskiy et al., 2020). Their parallelization ability enables training on larger datasets, which allows developing and pre-training models such as BERT (Devlin et al., 2019), or VideoBERT (Sun et al., 2019a).

2.3.1.3 Pre-training Language Representations

Instead of training sentence representations from scratch, several works propose learning fixed-length vector representations for sentences, pre-trained on large text corpora.

Learning representations in an unsupervised manner has been a popular approach in NLP. A naïve approach is computing sentence representations through word embeddings using neural networks. For example, Le and Mikolov (2014) map sentences or documents to vectors, to act as a memory of the context. These vectors are then concatenated with vectors of the surrounding words to predict the next words. The other direction is learning generic representations through sentence-level embeddings. In their work, skip-thought vectors, Kiros et al. (2015) encode sentences using recurrent neural networks and use their embeddings to reconstruct the previous and next sentences. Recent advancements rely on transformer-based models. BERT (bidirectional encoder representations from transformers) (Devlin et al., 2019) is a multi-layer bidirectional transformer encoder, and is trained using two unsupervised tasks. The first task is predicting randomly masked words using a bi-directional context. In the second one, the next sentence is replaced with a random sentence. Here the task is a binary prediction of whether the next sentence succeeds the first sentence. Pre-trained BERT models could be used for feature extraction or fine-tuned for new tasks.

Several works explore learning supervised representations. Conneau et al. (2017) compare standard recurrent models, trained on a natural language inference task, and show that the sentence representations extracted from these models can be transferred to other tasks successfully. Similarly, McCann et al. (2017) train a sequence-to-sequence model for machine translation and use the encoder for transferring knowledge to other NLP tasks.

2.3.2 Modelling Sequential Instructions

In the NLP literature, there is a large body of work for modelling and generating long text, including summarization (Cohan et al., 2018), poetry generation (Wang et al., 2016c), continuous text generation (Holtzman et al., 2018), dialogue generation (Cho et al., 2014), or generating biographies (Lebret et al., 2016). In this section, we focus on modelling long instructional text with instructive flow, particularly cooking recipes.

Recipes are popular data sources in natural language processing used to model instructional data. There have been many attempts to understand them. Early works focus on parsing the recipes to extract actions and ingredients (Tenorth et al., 2010; Malmaud et al., 2014; Kiddon et al., 2015; Jermsurawong and Habash, 2015). For example, Tenorth et al. (2010) generate plans from textual instructions, Kiddon et al. (2015) map recipes to action graphs, Beetz et al. (2011) use parsed instructions to make robots cook pancakes.

In recent years, neural network based solutions gained attention. Sequence-to-sequence learning (Sutskever et al., 2014; Kiros et al., 2014) has made it possible to successfully generate continuous text and build dialogue systems (Cho et al., 2014; Vinyals and Le, 2015). However, for recipe texts, such representations tend to perform poorly and suffer from incoherence, since they do not fully capture the underlying sequential nature of the instruction set. A number of solutions have been proposed to improve the coherence in such generations. Kiddon et al. (2016) propose a neural recipe generation model, assuming that the title and ingredients of the recipes are given as input. They train an encoder-decoder language model (Sutskever et al., 2014) with a checklist mechanism to keep track of the ingredients. Bosselut et al. (2018) develop a reinforcement learning based solution with discourse-aware rewards which encourage generating instructions with a correct order. H. Lee et al. (2020) produce personalized recipes by fusing users' previously consumed recipes with an attention mechanism.

In computer vision, generating recipes from images is becoming popular. Salvador et al. (2019) generate cooking recipes from food images. They first predict ingredients in food images and use these as input to a transformer-based decoder to generate sentences. However, since they generate an entire recipe as a block of long continuous text, their model is limited to the length of the decoder model which might generate incomplete paragraphs as recipes. Instead, Wang et al. (2020b) split recipes into several chunks and predict the instructions for each chunk guided by position encoders. Similarly, Wang et al. (2020c) propose a structure-aware generation network. Using an unsupervised approach, they first parse the recipes to obtain sentence-level tree structures. Using these trees as labels, they train an RNN to generate recipe trees from images. They then incorporate these trees into their model via attention to provide global guidance during recipe generation. Instead of using a single food image, Nishimura et al. (2019) generate instructional text given a photo sequence. Conversely, Zhu and Ngo (2020) propose an image generation method from recipe text using an instruction encoder to represent fine-grained details.

2.3.3 Learning Video Representations Using Text

Strong visual representations are essential for the performance of machine learning models. Videos, particularly instructional ones, naturally come with auxiliary modalities like sound and narrations. Those are valuable sources for learning video representations from multiple modalities. In this section, we overview two types of approaches for combining visual and textual data. The first one is learning a joint embedding space for text and visual data, the second is cross-modal learning.

2.3.3.1 Multi-Modal Embeddings

Many computer vision applications rely on joint understanding of visual and textual cues to get better representations, including captioning (Karpathy and Fei-Fei, 2015), retrieval (Kiros et al., 2014; Salvador et al., 2017), question answering (Malinowski et al., 2015), or video summarization (Plummer et al., 2017). A common approach is to learn a mapping of text and video in a shared embedding space where semantically similar data from different modalities is mapped to similar locations.

Frome et al. (2013) introduce a model to map images, encoded by CNNs, and categorical labels, encoded using word2vec (Mikolov et al., 2013), to a common space using a ranking loss. Kiros et al. (2014) extend this work for learning a common embedding space between image and sentences for image-caption retrieval. They encode sentences using an LSTM and extract image features from CNNs. The image features are then projected to the embedding space of the LSTM's hidden states. A pairwise ranking loss is used to map the correct target closer to the query than the other instances in the dataset. Faghri et al. (2017) extend this work by incorporating hard negatives in the loss function. Instead of embedding images or sentences into a common space, Karpathy and Fei-Fei (2015) learn multi-modal embedding spaces between sentences and image regions to generate descriptions of these regions.

Although it is easy to represent a single sentence, either using the average of word2vec vectors, or using RNNs, these representations would not perform well for long text. Salvador et al. (2017) learn joint embeddings between long instructional text from cooking recipes and food images. They represent their sentences using skip-thought vectors (Kiros et al., 2015) and use an LSTM on top of these vectors to model the textual instructions. They learn a shared embedding space between food images, textual instructions, and cooking ingredients, using a pairwise ranking loss. Carvalho et al. (2018) extend this work using a triplet loss. Instead of giving equal importance to all samples during training, Wang et al. (2019a) leverage a hard sample mining approach with a triplet loss. They also introduce an adversarial loss to enforce modality alignment and a consistency loss to ensure recipe-to-image and image-to-ingredients translation consistency.

Learning a common embedding space between videos and textual data has also been explored. To make the semantic relationships between videos and sentences stronger for the task of caption generation, Pan et al. (2016) propose learning a common embedding space for video and text and generating captions jointly. To learn powerful representations for zero-shot retrieval, Dong et al. (2019) first learn multi-level encodings of video and text to represent global, local, and temporal patterns. These multi-level encodings are then combined and projected to a common embedding space to learn joint embeddings, similar to Faghri et al. (2017). Miech et al. (2018) propose learning a joint embedding space between text and a varying number of video descriptors like global appearance, motion, audio, and face, any of which might be missing for some videos. Their mixture of embedding experts learn an expert model for each descriptor and combine them using weighting to obtain a final similarity score between the video and input sentence.

Instead of learning such embedding spaces from small-scale annotated datasets, some works focus on large-scale instructional video datasets. Miech et al. (2019b) collect a video

dataset with 130M video clips, extracted from 1.2M narrated web videos. They show that the visual representations they obtain through this joint space perform superior to previous works on tasks like text-to-video retrieval and action localization. However, their model does not handle misaligned narrations in the videos and also requires pre-trained CNNs to extract visual features. Miech et al. (2020) propose a multiple instance learning-based method to enable learning from such misaligned data sources. They also train their joint video-text embedding model from scratch without relying on pre-trained models.

2.3.3.2 Cross-Modal Learning

Early works use sound for learning better representations of videos by employing deep convolutional architectures to produce sound from image sequences (Owens et al., 2016) or videos (Aytar et al., 2016).

Recent works focus on using textual and visual data from large-scale instructional videos, which usually come with narrations. Sun et al. (2019a) propose using text from automatic speech recognition systems to provide supervision for learning representations on cooking videos. Their method, VideoBERT, is based on the BERT architecture (Devlin et al., 2019) and uses visual and textual words together, where the visual words are computed from video frames via vector quantization. Instead of the next sentence prediction task, VideoBERT is trained with a linguistic-visual alignment task by predicting whether the sentences are aligned with the visual data. They show that the high-level features learned by their model achieve significant improvements on dense video captioning (Zhou et al., 2018c) and activity recognition tasks. However applying vector quantization to video frames leads to a loss of fine-grained visual content. Zhu and Yang (2020) propose, ActBERT, which benefits from global actions and local regional objects. It encodes each modality with separate transformers that are tangled with cross-modal interactions.

2.3.4 Video Captioning

While the early works on video captioning are based on Markov models (Yu and Siskind, 2013) and ontologies (Das et al., 2013), later works mostly use deep learning. There is an extensive body of works on generating captions for short videos. Usually, in these works, videos are encoded into feature representations, which are then fed into a recurrent neural network to generate sentences (Venugopalan et al., 2015; Yao et al., 2015; Gan et al., 2017). There are many techniques used to improve the quality of generated captions, including attention (Yao et al., 2015), spatio-temporal graphs (Pan et al., 2020), complementary features (Zhang et al., 2017), joint embeddings for text and video (Pan et al., 2016), attributes or concepts (Pan et al., 2017), or reconstruction losses (Wang et al., 2018a). However, in this section we only focus on works targeting dense captioning of long videos as we present comparisons to these approaches in Chapter 4 where we generate sentences for the future actions.

Initial works focus on generating paragraph descriptions on the TACoS (Regneri et al., 2013) dataset. Rohrbach et al. (2014) temporally segments videos using hierarchical clustering based on the outputs of attribute classifiers. They then use phrase-based statistical

machine translation (Koehn et al., 2007) to generate sentences for these segments. Yu et al. (2016) generate sentences on ground truth segments by combining a sentence and a paragraph generator, with the latter encoding the temporal dependency among sentences.

The majority of later works are focused on dense captioning on more complex datasets like ActivityNet (Caba Heilbron et al., 2015), first tackled by Krishna et al. (2017) on this dataset. Similar to temporal action detection approaches, Krishna et al. (2017) first generate temporal proposals and then combine them with a captioning module that exploits the past and future with an attention mechanism. Wang et al. (2018b) use a bidirectional LSTM to improve the quality of event proposals. However, generic proposals are usually inconsistent and redundant, and Krishna et al. (2017) and Wang et al. (2018b) fail to consider temporal dependencies between captions. Several works thus select a sequence of coherent proposals and sequentially generate captions for them (Xiong et al., 2018; Xu et al., 2019a; Mun et al., 2019). Weakly supervised captioning is also explored on ActivityNet, by only providing captions without their temporal locations during training. Shen et al. (2017) describe video regions with lexical labels and form temporal regions based on their outputs. Rahman et al. (2019) use an attention-based multimodal fusion model on text, audio, and video data to generate temporal regions. Both methods use these temporal regions to generate captions. Duan et al. (2018) alternate between sentence localization for the captions and caption generation for the localized segments, using captioning and reconstruction losses.

Instead of separating the captioning problem into the two stages of proposal generation and captioning, Zhou et al. (2018c) produce proposals and descriptions simultaneously and mainly work on the YouCookII (Zhou et al., 2018b) dataset. Their work is composed of a transformer-based (Vaswani et al., 2017) video encoder for context-aware features, a proposal decoder similar to Zhou et al. (2018b) that localizes action proposal candidates, and finally a transformer-based decoder that generates captions. Since fine-grained differences in instructional videos are difficult to distinguish from visual features alone, some works also use narrations with transformer-based models (Hessel et al., 2019; Shi et al., 2019).

2.3.5 Temporal Video-Text Alignment

Given videos along with natural language descriptions, the goal of temporal alignment methods is to find the start and end times of these descriptions. The descriptions could be coming from a set of instructions, narrations, subtitles, or summaries. The alignment task assumes access to descriptions, both during training and testing.

There is a recent and growing body of work focused on grounding natural language descriptions in videos. Given an untrimmed video and a textual query, they aim at determining the start and end times of the segment that matches the query the most. The pioneering works are proposed by Anne Hendricks et al. (2017), who learn a shared embedding space for video and language and use a ranking loss for matching them, and Gao et al. (2017a), who use sliding windows to generate candidate regions and rank them using alignment scores and location regressors. Soon after, grounding textual queries gained attention, and different strategies were proposed to tackle the problem, including using reinforcement learning (He et al., 2019), generating discriminative proposals (Chen and Jiang, 2019), attention-based

localization models (Yuan et al., 2019; Opazo et al., 2019), or visual-textual graphs (Chen and Jiang, 2020). However, these methods assume a single description and operate on short untrimmed videos instead of the complex activity videos we work with.

In this section, we overview works that target aligning or grounding multiple textual descriptions in long videos, given a video and a sequence of sentence descriptions. Note that these approaches either directly use the sentences for alignment or convert them to object and verb pairs.

The early works perform alignment without using location supervision during training. One of the early works for alignment between videos and transcripts is proposed by Bojanowski et al. (2014) and is based on discriminative clustering. Bojanowski et al. (2015) extend this work for aligning video with natural language on cooking activities. Naim et al. (2014, 2015) match sentences with video frames and nouns in sentences with scene objects using hierarchical HMMs and CRFs, respectively. Song et al. (2016) extend these works by incorporating alignment of verbs to actions.

Temporal alignment is also linked to weakly supervised action segmentation, as such approaches can be employed for video and transcript-based alignments. Several weakly-supervised action segmentation works are utilized to align videos with transcripts, which are also provioded during testing (Chang et al., 2019; Richard et al., 2018b; Huang et al., 2016).

In the cooking domain, finding instructions and corresponding videos is easy, where instructions correspond to the recipes. Malmaud et al. (2015) target alignment of textual instructions with videos using narrations. They first align the recipes to narrations in the videos using HMMs. They then refine these alignments on videos using visual food detectors. Hahn et al. (2018) propose a video-recipe alignment framework. They extract visual features from frames and decide with an LSTM whether frames belong to an action or background. Action frames are then used to extract objects; similarly, sentences are parsed to extract action-object pairs. Using a similarity score, the action segments are aligned to recipe steps. Lin et al. (2020) first learn pairwise text-text, text-video, and video-video alignments from recipe pairs of the same dish. To derive joint alignments across multiple recipes of the same dish, they then use these pairwise alignments to construct a graph for each dish. Instead of relying on raw narrations, Huang et al. (2017) focus on referring expressions and propose a method for resolving noise in the narrations. In a follow-up work Huang et al. (2018a) resolve such visual-linguistic ambiguities to perform visual grounding in cooking videos.

Instead of using instructional activities, Zhu et al. (2015) propose a method for aligning books to their movie releases, where they represent sentences using skip-thought vectors (Kiros et al., 2015). They first match sentences from the books to subtitles in the movies and then learn a joint embedding space for videos and sentences (Kiros et al., 2014). Initial alignments based on the similarity between movie clips and sentences from books are subsequently smoothed using a pairwise CRF. Dogan et al. (2018) propose a supervised neural architecture based on four LSTMs encoding movie clips, text, actions, and previous alignments. In another supervised setting, given a list of sentence descriptions,

Chapter 3

Unsupervised Segmentation of Complex Activities

Contents

This chapter addresses the problem of temporal action segmentation in instructional videos. The content of this chapter corresponds to our CVPR 2018 publication, *Unsupervised Learning and Segmentation of Complex Activities from Video* (Sener and Yao, 2018). Given

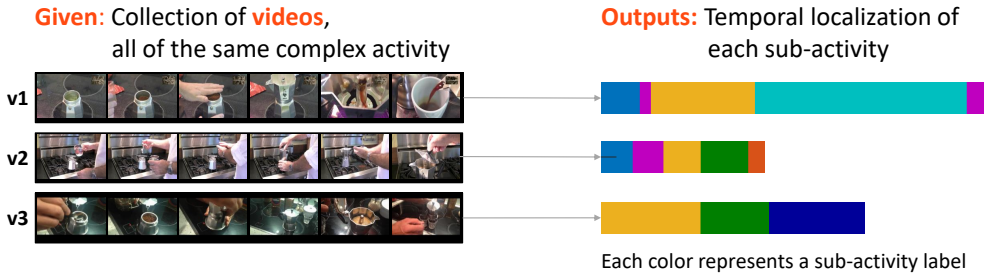


Figure 3.1: We aim to discover and segment the sub-activities of a complex activity from a collection of videos. We assume we are given a collection of videos, all of the same complex activity, and we specify the number of sub-activities, K. The goal is to find and temporally localize these sub-activities.

a collection of videos, all from a complex activity, we aim to find and temporally localize the instructional steps (see Figure 3.1).

We present a new method for unsupervised segmentation of complex activities from video into multiple steps, or sub-activities, without any textual input. We propose an iterative discriminative-generative approach which alternates between discriminatively learning the appearance of sub-activities from the videos' visual features to sub-activity labels and generatively modelling the temporal structure of sub-activities using a Generalized Mallows Model. In addition, we introduce an extension of our model to account for background frames unrelated to the actual activities. We validate our approach on the challenging Breakfast Actions and Inria Instructional Videos datasets.

3.1 Introduction

This work addresses the problem of understanding complex activities from video sequences. Complex activities can be found in instructional videos; YouTube hosts hundreds of thousands of such videos on activities as common as *"making coffee"* to the more obscure *"weaving banana fibre cloths"*. Another relevant domain for complex activity understanding is assistive robotics; a robot that can understand and parse the steps of a household task such as *'doing laundry'* can anticipate and support upcoming steps or sub-activities.

Complex activity understanding has received little attention in the computer vision community compared to the more popular simple action recognition task. In simple action recognition, short, trimmed clips are classified with single labels, e.g., of sports, playing musical instruments (Karpathy et al., 2014; Soomro et al., 2012), and so on. Performance on simple action recognition has seen a remarkable boost with the use of deep architectures (Karpathy et al., 2014; Simonyan and Zisserman, 2014a; Tran et al., 2015). Such methods, however, are rarely applicable for temporally localizing and/or classifying actions from longer, untrimmed video sequences, usually due to the lack of temporal consideration. Even works which do incorporate some modelling of temporal structure (Fernando et al., 2015; Sharma et al.,

2015; Srivastava et al., 2015; Tran et al., 2015) do little more than capturing frame-to-frame changes, which is why the state of the art still relies on either optical flow (Simonyan and Zisserman, 2014a) or dense trajectories (Tran et al., 2015; Wang et al., 2013). Moving towards understanding complex activities then becomes even more challenging, as it requires not only parsing long video sequences into semantically meaningful sub-activities but also capturing the temporal relationships that occur between these sub-activities.

We aim to discover and segment the steps of a complex activity from collections of videos in an unsupervised way based purely on visual inputs. Labelling video sequences is tedious and expensive; it is much easier to collect videos of the same complex activity by querying of platforms such as YouTube. Within the same activity class, it is likely that videos will share common steps and follow a similar temporal ordering. Before our work, works in a similar vein of unsupervised learning all require inputs from narration; the sub-activities and sequence information are extracted either entirely from (Alayrac et al., 2016) or rely heavily on text (Malmaud et al., 2015; Sener et al., 2015). Such works assume that the text is well-aligned with the visual information of the video so that visual representations of the sub-activities are learned from within the text's temporal bounds. This is not always the case for instructional videos, as it is far more natural for the human narrator to first speak about what will be done and then carry out the action. Carefully checking many videos, most of the time, we see people talk about an action before/after performing it, or they talk about an alternative action/object. Sometimes people talk about an action but do not show it. Such choices cause various misalignments between video and narrations. Finally, reliably parsing spoken natural language into scripts[1] is an unsolved and open research topic in itself. As such, it is in our interest to rely only on visual inputs.

Temporal segmentation of complex activities purely from video data is a challenging task, see Figure 3.2. First of all, in these complex activities, the ordering is loose but not fixed. In other words, one can accomplish the same task with a slightly different order or even skip some steps. There might be many garbage or background frames that are not relevant to what we are interested in. There might be significant appearance variations amongst the frames of the same sub-activities from different videos. Moreover, appearance variations between different sub-activities from the same video can have very fine-grained differences.

We propose an iterative model which alternates between learning a discriminative representation of a video's visual features to sub-activities and a generative model of the sub-activities' temporal structure. By combining the sub-activity representations with the temporal model, we arrive at a segmentation of the video sequence, which is then used to update the visual representations (see Figure 3.3). We represent the sub-activities by learning linear mappings from visual features to a low dimensional embedding space with a ranking loss. The mappings are optimized such that visual features from the same sub-activity are pushed together, while different sub-activities are pulled apart.

Temporally, we treat a complex activity as a sequence of permutable sub-activities and model the distribution over permutations with a Generalized Mallows Model (GMM) (Fligner

[1]Here, we refer to the NLP definition of a script as *"a predetermined, stereotyped sequence of actions that define a well-known situation"* (Schank and Abelson, 1975).

Figure 3.2: What makes temporal action segmentation in complex activities difficult? **(a)** First of all, due to the fine-grained nature of the data, the visual appearance variations can be quite confusing; for example, the two sub-activities "put stove" and "remove stove" have very fine-grained differences, while overall looking very similar. **(b)** In instructional videos, there might be a lot of garbage or background frames, which are not relevant to what we are interested in. **(c)** There might be huge appearance variations between the same sub-activities, e.g., "filling water" frames from different videos can look very different. **(d)** In some complex activities, some steps can be skipped. **(e)** Finally, and most importantly, in complex activities, the ordering is loose but not fixed.

and Verducci, 1986). GMMs have been used in combination with topic models in the natural language processing community to model document structures (Chen et al., 2009) and script knowledge (Frermann et al., 2014). In our method, the GMM assumes that a canonical sequence ordering is shared among videos of the same complex activity, as per the feasibility assumption of doing unsupervised learning. There are several advantages of using the GMM for modelling temporal structure. First and foremost, the canonical ordering enforces

a global ordering constraint over the activity – something not possible with Markovian models (Kuehne et al., 2014b; Richard et al., 2017; Sener et al., 2015) and recurrent neural networks (RNNs) (Yeung et al., 2016). Secondly, considering the temporal structure as a permutation offers flexibility and richness in modelling. We can allow for missing steps and deviations, all of which are characteristic of complex activities, but cannot be accounted for with works which enforce a strict ordering (Alayrac et al., 2016). Finally, the GMM is compact – parameters grow linearly with the number of sub-activities, versus quadratic growth in parameterization in pairwise relationships e.g., in HMMs.

Within a video, it is unlikely that every frame corresponds to a specified sub-activity. They may be interspersed with unrelated segments of the actor talking or highlighting previous or subsequent sub-activities. Depending on how the video is made, such segments can occur arbitrarily with varying lengths. It becomes difficult to maintain a consistent temporal model under these uncertainties, which in turn affects the quality of visual representations. In this work, we extend our segmentation method to represent such "background frames", and therefore be robust against such frames. In this variant, we explicitly learn about and represent such frames so that we can exclude them from the temporal model.
Our contributions can be summarized as follows:

- We are the first to explore a fully unsupervised method for temporal understanding of complex activities in video without requiring any text. We design a discriminative appearance learning model to enable the use of GMMs on visual data with visual features (Sánchez et al., 2013; Tran et al., 2015; Wang et al., 2013).
- We verify our method on real-world videos of complex activities which do not follow strict orderings and are heavily interspersed with background frames.
- We demonstrate that our method achieves competitive results comparable or better than the state of the art on two challenging complex activity datasets, Breakfast Actions (Kuehne et al., 2014b) and Inria Instructional Videos (Alayrac et al., 2016).

3.2 Related Work

Modelling temporal structures in activities has been focused predominantly at a frame-wise level (Fernando et al., 2015; Sharma et al., 2015; Srivastava et al., 2015; Tran et al., 2015). Existing works on complex activity understanding typically require fully annotated video sequences with start and end points of each sub-activity (Kuehne et al., 2014b; Richard and Gall, 2016; Rohrbach et al., 2012). Annotating every frame in videos is expensive and makes it difficult to work at a large scale. Instead of annotations, a second line of work tries to use cues from accompanying narrations (Alayrac et al., 2016; Malmaud et al., 2015; Sener et al., 2015). These works assume that the narrative text is well-aligned with the visual data, with performance governed largely by the quality of the alignment. For example, Alayrac et al. (2016) use instruction narrations as temporal boundaries of sub-activities for discriminative clustering. Sener et al. (2015) represent every frame as a concatenated histogram of textual

and visual words, which are used as input to a probabilistic model. The applicability of these methods is limited because neither the existence of accompanying text, nor their proper alignment to the visual data can be taken for granted.

More recent works focus on developing weakly-supervised solutions, i.e., where the orderings of the sub-activities are provided either only during training (Huang et al., 2016; Richard et al., 2017) or testing as well (Bojanowski et al., 2014). These methods try to align the frames to the given ordered sub-activities. Similar to us, the work of Bojanowski et al. (2014) includes a *"background"* class. However, they assume that the background appears only once between every consecutive pair of sub-activities, while our model does not force any constraints on the occurrence of background. Others (Huang et al., 2016; Richard et al., 2017) borrow temporal modelling methods from speech recognition such as connectionist temporal classification, RNNs and HMMs.

In the bigger scope of temporal sequences, several previous works have also addressed unsupervised segmentation (Fox et al., 2014; Zhou et al., 2012; Krüger et al., 2017) on motion capture data. Similar to us in spirit is the work of Fox et al. (2014), who propose a Bayesian nonparametric approach to model multiple sets of time series data concurrently. However, it has been applied only to motion capture data. Since skeleton poses are lower-dimensional and exhibit much less variance than video, it is unlikely for such a model to be directly applicable to video without a strong discriminative appearance model. To our knowledge, we are the first to tackle the problem of complex activity segmentation working solely with visual data without any supervision.

3.3 Probability Theory

3.3.1 Preliminaries

A random variable x represents an uncertain quantity which could be the results of an experiment or a measurement. Some values are observed frequently, and some rarely. A probability distribution, $P(x)$, of random variable x, provides the probabilities of occurrence of different possible outcomes in a sample space.

We distinguish between discrete and continuous random variables. A discrete random variable, x, takes values from a pre-defined set, e.g., head or tail, in a coin flip experiment. We use a probability mass function to assign a probability to each possible outcome. The associated probabilities must be all positive, and the sum of the probabilities of all outcomes is 1.

$$P(x = x_i) > 0, \qquad \sum_i P(x = x_i) = 1. \tag{3.1}$$

A continuous random variable takes an uncountable infinite number of possible values. The probability of a random variable taking any particular value is 0. So instead, we define the probability that falls in some interval. The function that represents a continuous probability distribution is called a probability density function. Its integral sums to 1.

3.3.1.1 Joint and Conditional Probabilities

Given two random variables, x and y, their joint probability can be written as $P(x, y)$. A joint probability is composed of two or more variables. It might relate only discrete or only continuous variables, as well as a mixture of both. The total joint probability sums up to 1.

The process of recovering any random variable from a joint distribution via summing (discretely) or integrating (continuously) over other random variables is called marginalization. For continuous variables the marginal distributions can be computed as follows:

$$P(x) = \int P(x, y)dy, \qquad P(y) = \int P(x, y)dx. \tag{3.2}$$

The conditional probability of $P(x|y = y^*)$, usually abbreviated as $P(x|y)$, shows us the tendency of x taking different outcomes given that the random variable y is fixed to the value y^*. Here, the vertical line | is read as "given". Fixing y, we can compute the values of x. However, this won't sum up to 1. Thus, we also need to normalize using the marginal probability distribution:

$$P(x|y) = \frac{P(x, y)}{\int P(x, y)dx} = \frac{P(x, y)}{P(y)}, \tag{3.3}$$

which can be rearranged as

$$P(x, y) = P(x|y)P(y) = P(y|x)P(x). \tag{3.4}$$

3.3.1.2 Independence and Conditional Independence

If knowing x does not provide further information about y and vice-versa, we say x and y are independent:

$$P(x, y) = P(x|y)P(y) = P(x)P(y). \tag{3.5}$$

When more random variables are introduced, the definition of independence also gets complex. For example,

$$P(x|y, z) = P(x|y)P(z), \tag{3.6}$$

where x is conditionally independent of z given y. If we know y, then z does not provide further information on x. Such independence is not symmetric. Also, this does not mean that x and z are independent. Conditional independence allows us to remove redundancies and describe distributions with fewer parameters. For example, if given y, x and z are independent, then we can simplify the following joint distribution:

$$P(x, y, z) = P(x|y, z)P(y|z)P(z) \tag{3.7}$$

$$= P(x|y)P(y|z)P(z). \tag{3.8}$$

3.3.1.3 Relevant Probability Distributions

In this section, we define several probability distributions occurring in this thesis.

The **Bernoulli distribution** is a univariate, binary, discrete distribution used to model binary trials where the random variable takes the value of 1 with probability λ and 0 with probability $1-\lambda$, where $\lambda \in [0,1]$. There are two possible outcomes, $x \in \{0,1\}$, which correspond to "failure" and "success". It can be used to represent a (possibly biased) coin toss with outcomes heads and tails, or it can be used to represent the probability of an image containing a cat or not. The probability distribution can be written as follows

$$P_{\text{Bern}}(x) = \lambda^x (1-\lambda)^{(1-x)}, \tag{3.9}$$

λ is the only parameter to govern the distribution. It determines the probability of success such that $P_{\text{Bern}}(x=1) = \lambda$ and failure $P_{\text{Bern}}(x=0) = 1-\lambda$.

The **beta distribution** is a univariate continuous distribution defined on a single variable bounded between $\lambda \in [0,1]$ with parameters $\alpha, \beta \in [0,\infty]$. The probability distribution can be written as follows:

$$P_{\text{beta}}(\lambda) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \lambda^{(\alpha-1)}(1-\lambda)^{\beta-1}, \tag{3.10}$$

where $\Gamma(\cdot)$ is the gamma function, an extension of the factorial to real numbers, $\Gamma(z) = \int_0^\infty x^{z-1}e^{-x}dx$ and for any positive integer it is $\Gamma(z) = (z-1)!$. P_{beta} is a suitable distribution for representing the uncertainty in parameter λ in the Bernoulli distribution. Its generalization to multiple variables is a Dirichlet distribution.

The **categorical distribution** is a univariate, discrete distribution used to represent the probability of observing one of K possible outcomes. It is a generalization of the Bernoulli distribution with multiple outcomes. For example, it could be used to describe the probability of rolling a dice, or the probability of an image containing a cat, human, car, or bird. The probability of observing K outcomes are modeled with vector, $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_K]$ where $\lambda_i \in [0,1]$ and $\sum_{i=1}^K \lambda_i = 1$. The categorical distribution simply returns the probability of a given event, $P_{\text{cat}}(x=i) = \lambda_i$.

The **multinomial distribution** models observing the values $\{1, ..., K\}$ with probabilities $\boldsymbol{\lambda} = [\lambda_1, ..., \lambda_K]$ for N repeated trials, where each trial has a discrete set of possible outcomes. Here $\lambda_i \in [0,1]$ and $\sum_{i=1}^K \lambda_i = 1$. It is a generalization of the categorical distribution with $N = 1$. For example, it can be used to model the probability of getting every time an outcome of 2 in an experiment where a dice is rolled N=3 times. It is also a generalization of a binomial experiment where each trial can result in two possible outcomes. The probability function can be written as

$$P_{\text{mult}}(x) = \frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i+1)} \prod_{i=1}^K \lambda_i^{x_i}, \quad \text{where} \quad \sum_i x_i = N. \tag{3.11}$$

The **Dirichlet distribution** is a multivariate, continuous distribution defined over K con-

tinuous values $[\lambda_1, ..., \lambda_K]$ where $\lambda_i = [0, 1]$ and $\sum_{i=1}^{K} \lambda_i = 1$. For K dimensions it has K parameters $[\alpha_1, ..., \alpha_K]$ where $\alpha_i > 0$. The probability distribution can be written as follows:

$$P_{\text{Dir}}(\lambda_{1...K}) = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} \lambda_i^{\alpha_i - 1}. \tag{3.12}$$

The Dirichlet distribution can be used to express our prior beliefs about the parameters of a categorical or multinomial distribution.

3.3.2 Bayes' Theorem

Bayes' theorem describes how to update the probabilities when given evidence. For example, we can find the probability of a person having cancer from incidence rates in a population. However, given additional evidence of this person being a smoker, we need to update the probability, as smoking increases the risk of getting cancer.
Using equation 3.4, we can define the following relationship between the conditional distributions of P(x|y) and P(y|x):

$$P(x|y)P(y) = P(y|x)P(x). \tag{3.13}$$

We can write Bayes' rule as follows:

$$\begin{aligned} P(x|y) &= \frac{P(y|x)P(x)}{P(y)} && (3.14)\\ &= \frac{P(y|x)P(x)}{\int P(x,y)dx} && (3.15)\\ &= \frac{P(y|x)P(x)}{\int P(y|x)P(x)dx}, && (3.16) \end{aligned}$$

where $P(x|y)$ is the posterior that we are trying to estimate. It corresponds to what we know about x given some evidence, y. $P(y)$ is the evidence, while $P(x)$ refers to the prior, representing all we know about x before some evidence is taken into account. $P(y|x)$ describes the likelihood, which is the probability of observing new evidence given x.

3.3.3 Conjugate Prior

When we fit models to data, we also want to measure how uncertain we are about a fit. Some probability distributions are commonly used to model data, e.g., Bernoulli or categorical, while others, such as beta or Dirichlet, are used to represent the uncertainty over the parameters of the fitted models. The parameters of this second distribution are called hyperparameters.

In Bayesian theory, if the posterior and the prior probability distributions are in the same distribution family, the prior distribution is called conjugate prior for the likelihood function. For example, the beta distribution can be used to define a distribution over the

parameter of the Bernoulli distribution, while Dirichlet can model the parameters of the categorical distribution. Therefore the beta distribution is a conjugate prior to the Bernoulli distribution, and the Dirichlet is conjugate prior to the categorical distribution.

For some likelihood functions, if we choose a certain prior, the posterior ends up being distributed according to the same type of distribution as the prior, but with different parameters. For example, using beta as the prior distribution of the Bernoulli:

$$\begin{aligned} P(\lambda|x) &\propto P_{\text{Bern}}(x|\lambda)P_{\text{beta}}(\lambda) \\ &= \lambda^x(1-\lambda)^{(1-x)}\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\lambda^{(\alpha-1)}(1-\lambda)^{\beta-1} \\ &= \lambda^{x+\alpha-1}(1-\lambda)^{(\beta-x)}\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \end{aligned} \tag{3.17}$$

Expanding with the following constant $c = \frac{\Gamma(x+\alpha+\beta-x+1)}{\Gamma(x+\alpha)\Gamma(\beta-x+1)}$ allows rearranging to:

$$P(\lambda|x) \propto \hat{c} \cdot \frac{\Gamma(\hat{\alpha}+\hat{\beta})}{\Gamma(\hat{\alpha})\Gamma(\hat{\beta})}\lambda^{(\hat{\alpha}-1)}(1-\lambda)^{\hat{\beta}-1} \tag{3.18}$$

where $\hat{c} = \frac{\Gamma(x+\alpha)\Gamma(\beta-x+1)}{(\alpha+\beta)\Gamma(\alpha)\Gamma(\beta)}$ is a constant, and $P(\lambda|x)$ is another beta distribution with parameters $\hat{\alpha} = \alpha + x$ and $\hat{\beta} = \beta + 1 - x$.
Let N denote the number of observations. The posterior hyperparameters can be computed as $\hat{\alpha} = \alpha + \sum_{i=1}^{N} x_i$ and $\hat{\beta} = \beta + \sum_{i=1}^{N}(1 - x_i)$. This allows us to easily compute the posterior, as the conjugate priors allow us to integrate out the actual parameters of a distribution. It also allows for measuring how much our beliefs change after observing new data.

3.3.4 Fitting Probability Models

We can use maximum likelihood or maximum a posteriori for predicting model parameters, θ, for a dataset composed of $\{x_1, \ldots, x_N\}$.
Maximum likelihood, ML, estimation is concerned with finding a set of parameters $\hat{\theta}$ under which the data is most likely. Assuming each data point is independently drawn, the likelihood function $P(x_1, \ldots, x_N|\theta)$ can be written as the product of individual likelihood functions at a single data point $P(x_i|\theta)$, which can be easily maximized:

$$\hat{\theta} = \arg\max_{\theta} \left[\prod_{i=1}^{N} P(x_i|\theta) \right]. \tag{3.19}$$

Maximum a posterior, MAP, estimation utilizes prior information on parameters θ and maximizes the posterior probability $P(\theta|x_1, \ldots, x_N)$:

$$\hat{\theta} = \arg\max_{\theta} P(\theta|x_1, \ldots, x_N). \tag{3.20}$$

By using Bayes' rule we get

$$\hat{\theta} = \arg\max_{\theta} \left[\frac{\prod_{i=1}^{N} P(x_i|\theta)P(\theta)}{P(x_1, \ldots, x_N)} \right], \tag{3.21}$$

where the $P(x_1, \ldots, x_N)$ is constant w.r.t. the model parameters, and as such we can omit. This makes the fitting similar to maximum likelihood estimation with an additional prior term.

3.3.5 Sampling

When computing the posterior is not tractable, an alternative is drawing samples from the posterior distribution. A Markov chain Monte Carlo (MCMC) method is a popular way of generating samples from any complex high-dimensional probability distribution. It generates a chain of samples from the distribution where each sample depends on the previous one (Markov chain), with a non-deterministic generation process (Monte Carlo).

3.3.5.1 Gibbs sampling

A common MCMC sampling approach is Gibbs sampling (Griffiths and Steyvers, 2004) which is used to draw samples from multivariate probability distributions. It generates a chain of samples by cycling through each variable in turn in any order. It draws a sample from the conditional distribution $P(x_i|\boldsymbol{x}_{\setminus i})$, where $\boldsymbol{x}_{\setminus i}$ corresponds to the the set of all variables except x_i. After repeating this process for a burn-in period, such that the initial conditions are forgotten, a sample is considered to be drawn from the joint distribution. The sampling process can be summarized as in Algorithm 3.1. The conditional distribution of one variable given all others is proportional to the joint distribution:

$$P(x_i|\boldsymbol{x}_{\setminus i}) = \frac{P(x_1, \ldots, x_N)}{P(\boldsymbol{x}_{\setminus i})} \propto P(x_1, \ldots, x_N). \tag{3.22}$$

Algorithm 3.1 Gibbs sampling.

1: Initialize the chain $x_i^{(0)}$
2: **for** iterations $t = 1, 2, 3, \ldots$ **do**
3: **for** variables $x_i^{(t)}$ **do**
4: $x_i^{(t+1)} \propto P(x_i^{(t+1)}|x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_{i+1}^{(t)}, x_{i+2}^{(t)}, \ldots)$

3.3.5.2 Slice sampling

Slice sampling (Neal, 2003; MacKay, 2003) is another MCMC sampling approach. Consider the goal of obtaining samples from a univariate distribution $P(x)$. We introduce a new uniform random variable u, conditioned on x $P(u|x) \sim \text{uniform}([0, P(x)])$. We fix x and

sample u uniformly from the region under the curve of the density function $P(x)$. We then fix u and sample x from the slice through the distribution, which is constructed by stepping into both directions with a fixed step size and narrowed down by rejecting those x that are sampled above $P(x)$. The algorithm alternates between sampling u uniformly in the vertical direction and uniform sampling from the horizontal "slice" defined by the current vertical position, u. Slice sampling is an alternative to Gibbs sampling that avoids the need to sample from non-standard distributions.

3.4 The Generalized Mallows Model (GMM)

Permutations appear naturally in a wide variety of domains such as information retrieval (Farah and Vanderpooten, 2007) or classification (Cheng and Hüllermeier, 2009). The Mallows model (MM) is a distribution that models orderings or permutations.

In the standard Mallows model (Mallows, 1957), the probability of observing some ordering $\boldsymbol{\pi}$ is defined by a dispersion parameter ρ and a canonical ordering $\boldsymbol{\sigma}$,

$$P_{MM}(\boldsymbol{\pi}|\boldsymbol{\sigma},\rho) = \frac{e^{-\rho \cdot d(\boldsymbol{\pi},\boldsymbol{\sigma})}}{\psi(\rho)}, \tag{3.23}$$

where any distance metric for rankings or orderings can be used for $d(\cdot,\cdot)$. A frequently used metric is the Kendall τ distance which measures the minimum number of swaps between adjacent positions required to transform $\boldsymbol{\pi}$ to the canonical ordering $\boldsymbol{\sigma}$. As such, orderings close to the canonical ordering will have high probability and the probability of an ordering $\boldsymbol{\pi}$ decreases exponentially with an increased distance to the canonical ordering. The extent to which the probability decreases as $\boldsymbol{\pi}$ differs from $\boldsymbol{\sigma}$ is controlled by a dispersion parameter $\rho > 0$; $\psi(\rho)$ serves as a normalization constant. Similarly, an increasing value of ρ implies strong penalization of the distance. There are several extensions of the Mallows model, such as non-parametric models (Lebanon and Mao, 2008) and mixture models (D'Elia and Piccolo, 2005). The Generalized Mallows Model (GMM) is the most popular extension among all.

The GMM, first introduced by Fligner and Verducci (1986), extends the standard Mallows model by introducing a set of dispersion parameters $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_{K-1}]$, each affecting a particular position of the permutation, to allow individual parameterization of the K elements in the ordering. This allows for modeling a distribution with more emphasis on the consensus of certain positions in the ordering while at the same time having more uncertainty in the others.

3.4.1 The Distance Function in GMM

The distance function between orderings now can be considered as a set of independent components. The GMM represents permutations as a vector of inversion counts $\mathbf{v} = [v_1, \ldots, v_{K-1}]$ with respect to the canonical ordering, where element v_k corresponds to the total number of elements in $(k+1, \ldots, K)$ that occur before k in the ordering $\boldsymbol{\pi}$. We assume the canonical ordering as a parameter to the distribution and set it as the identity permutation $(1, ..., K)$.

Note that only $K-1$ elements are needed since v_K is 0 by definition, as there cannot be any elements greater than K.

We present the algorithms for converting an inversion count vector, $\mathbf{v}$ computed with respect to the identity permutation to the ordering $\boldsymbol{\pi}$ in Algorithm 3.2 and $\boldsymbol{\pi}$ to $\mathbf{v}$ conversion in Algorithm 3.3. For example, given an ordering of $\boldsymbol{\pi} = [4\ 2\ 1\ 3\ 6\ 5]$, the inversion count vector is computed as $\mathbf{v} = [2\ 1\ 1\ 0\ 1]$. v_1 is 2 as 4 and 2 occur before 1 in $\boldsymbol{\pi}$ as elements which are greater than 1. Similarly, v_2 is 1, as only 4 appears before 2 as a greater element. The sum of all positions in the inversion count corresponds to the ordering's Kendall τ distance from the canonical ordering.

Algorithm 3.2 Algorithm for converting the vector of inversion counts $\mathbf{v}$ to ordering $\boldsymbol{\pi}$.

Require: $\mathbf{v}, K$
1: $\boldsymbol{\pi}[0, ..., K-1] \leftarrow 0$
2: $\boldsymbol{\pi}[0] \leftarrow K$
3: **for** $i = K-1$ down to 1 **do**
4: **for** $j = K - i$ down to $\mathbf{v}[i-1] + 1$ **do**
5: $\boldsymbol{\pi}[j] \leftarrow \boldsymbol{\pi}[j-1]$
6: $\boldsymbol{\pi}[\mathbf{v}[i-1]] \leftarrow i$
7: **return** $\boldsymbol{\pi}$

Algorithm 3.3 Algorithm for converting the ordering $\boldsymbol{\pi}$ to the vector of inversion counts $\mathbf{v}$.

Require: $\boldsymbol{\pi}, K$
1: $\mathbf{v}[v_1, ..., v_{K-1}] \leftarrow 0$
2: **for** $i = K-1$ down to 1 **do**
3: **for** $j = 0$ to $i-1$ **do**
4: **if** $\boldsymbol{\pi}[j] > \boldsymbol{\pi}[i]$ **then**
5: $\mathbf{v}[\boldsymbol{\pi}[i]-1] \leftarrow \mathbf{v}[\boldsymbol{\pi}[i]-1] + 1$
6: **return** $\mathbf{v}$

3.4.2 The GMM

GMM introduces the dispersion parameters $\boldsymbol{\rho} = [\rho_1, ..., \rho_{K-1}]$ to MM where a dispersion parameter is assigned to each position which defines individual levels of tolerance for the distance function. The probability of an ordering becomes

$$P_{GMM}(\boldsymbol{\pi}|\boldsymbol{\sigma}, \boldsymbol{\rho}) = \frac{e^{-\sum_{k=1}^{K-1} \rho_k \cdot d(\pi_k, \sigma_k)}}{\psi(\rho)}. \tag{3.24}$$

If we assume that $\boldsymbol{\sigma}$ is the identity permutation, then a natural distance, $d(\boldsymbol{\pi}, \boldsymbol{\sigma})$, with an inversion count vector $\mathbf{v}$, can be defined as $\sum_k \rho_k v_k$, leading to

$$P_{GMM}(\mathbf{v}|\boldsymbol{\rho}) = \frac{e^{-\sum_{k=1}^{K-1}\rho_k v_k}}{\psi_k(\boldsymbol{\rho})} = \prod_{k=1}^{K-1}\frac{e^{-\rho_k v_k}}{\psi_k(\rho_k)}, \tag{3.25}$$

with $\psi_k(\rho_k) = \frac{1-e^{-(K-k+1)\rho_k}}{1-e^{-\rho_k}}$ as the normalization. We omit this constant in the rest of the chapter as we work with the unnormalized GMM distribution.
GMM allows for the factorization of the distribution into individual components:

$$P_{GMM_k}(v_k|\rho_k) \propto e^{-\rho_k v_k}. \tag{3.26}$$

This allows for position-specific penalty parameters, and therefore individual degrees of temporal flexibility for ordering positions. A high value of ρ_k indicates fewer inversions and, therefore, less tendency to deviate from the canonical ordering.

As the GMM is a member of the family of exponential distributions, a conjugate prior can be defined. The conjugate prior can be factorized into element-wise components like GMM. The natural prior (Fligner and Verducci, 1990) for each element ρ_k is the conjugate:

$$P_{GMM_0}(\rho_k|v_{k,0},\nu_0) \propto e^{(-\rho_k v_{k,0} - log\psi_k(\rho_k))\nu_0}, \tag{3.27}$$

with hyperparameters $v_{k,0}$ and ν_0, which need to be defined manually. Intuitively, the prior states that over ν_0 previous trials, $\nu_0 \cdot v_{k,0}$ inversions will be observed (Chen et al., 2009). The distribution can be updated with the observed v_k. We manually set the sample size prior, ν_0, which corresponds to the weight of the prior information. However, the hyperparameter $v_{k,0}$, which encodes the distance of an element from its canonical position in trials, is difficult to set. For simplicity, we do not set multiple priors for each k and instead introduce a common prior ρ_0 as per (Chen et al., 2009), such that

$$v_{k,0} = \frac{1}{e^{\rho_0}-1} - \frac{K-k+1}{e^{(K-k+1)\rho_0}-1}. \tag{3.28}$$

We again set the prior dispersion hyperparameter, ρ_0, manually. Higher values for ρ_0 indicate an ordering closer to the canonical ordering.

3.5 Proposed Model

Assume we are given a collection of M videos, all of the same complex activity, and that each video is composed of an ordered sequence of multiple sub-activities. A single video i with J_i frames can be represented by a design matrix of features $\mathbf{F}_i \in \mathbb{R}^{J_i \times D}$ where D is the feature dimension. We further define $\mathbf{F}$ as the concatenated design matrix of features from all M videos, and $\mathbf{F}_{\setminus i}$ as the features excluding video i.

In Figure 3.3, we present an overview of our full model. Our goal is to assign a sub-activity label to each frame in the video collection. We represent the sub-activity assignment as a bag of frame counts, $\mathbf{a}_i$. Since our algorithm is unsupervised, we start with uniform initialization, e.g., in Figure 3.3 the initial sub-activity count is 4 for all sub-activities. The order, $\boldsymbol{\pi}_i$, for our K sub-activities is initialized to the canonical ordering. Together the counts and the ordering form the sub-activity assignments to video frames. With our initial assignments, we first discriminatively learn a joint embedding space between the sub-activity labels. This is necessary because naïve clustering of the video frames leads to groupings according to video rather than sub-activity, see Figure 3.5 (left). Section 3.5.1 describes how we discriminatively learn the features $\mathbf{F}$, as a better representation for video frames. We then continue with our generative process to update the sub-activity count and orderings in the second step in Figure 3.3. We provide the details of the generative process of our temporal model in Section 3.5.2 and our full model, which models background frames in Section 3.5.3.

An overview of our iterative process for our standard model without background can be found in Figure 3.4. According to this we first learn video frame embeddings w.r.t. the pseudo-labels obtained from an initialization or previous segmentation step. We then sample, for each video; first, the activity counts $\mathbf{a}$, and then the ordering $\boldsymbol{\pi}$, both via Gibbs sampling. For the entire collection we then sample the dispersion parameter, $\boldsymbol{\rho}$, with slice sampling. Then we construct the new segmentation, $\mathbf{z}$, for each video through $\mathbf{a}$ and $\boldsymbol{\pi}$. We iterate over these steps until our stopping criterion is reached.

3.5.1 Sub-Activity Representation

Within a video collection of a complex activity, there may be huge variations in visual appearance, even with state-of-the-art visual feature descriptors (Sánchez et al., 2013; Tran et al., 2015; Wang et al., 2013). Suppose for frame j of video i we have video features X_{ij} with dimensionality V. These features, if clustered naively, are most likely to group together according to video rather than sub-activity, see Figure 3.5 (left). To cluster the features more discriminantly, we learn a linear mapping of these features into a latent embedding space, i.e., $\Phi_f(X_{ij}) : \mathbb{R}^V \rightarrow \mathbb{R}^E$. We also define in the latent space K anchor points, with locations determined by a second mapping $\Phi_a(k) : \{1, \ldots, K\} \rightarrow \mathbb{R}^E$. More specifically,

$$\Phi_f(X_{ij}) = \mathbf{W}_f X_{ij}, \quad \mathbf{W}_f \in \mathbb{R}^{E \times V}, \tag{3.29}$$

$$\Phi_a(k) = \mathbf{W}_a(k), \quad \mathbf{W}_a \in \mathbb{R}^{E \times K}, \tag{3.30}$$

where $\mathbf{W}_f$ and $\mathbf{W}_a$ are the learned embedding weights and E is the dimensionality of the joint latent space. Here, $\mathbf{W}_a(k)$ is the k-th column of $\mathbf{W}_a$, which corresponds to the location of anchor k in the latent space. Together, $\mathbf{W}_f$ and $\mathbf{W}_a$ make up the parameter $\mathbf{W}$. We use the similarity of the video feature with respect to these anchor points as a visual feature descriptor, i.e.,

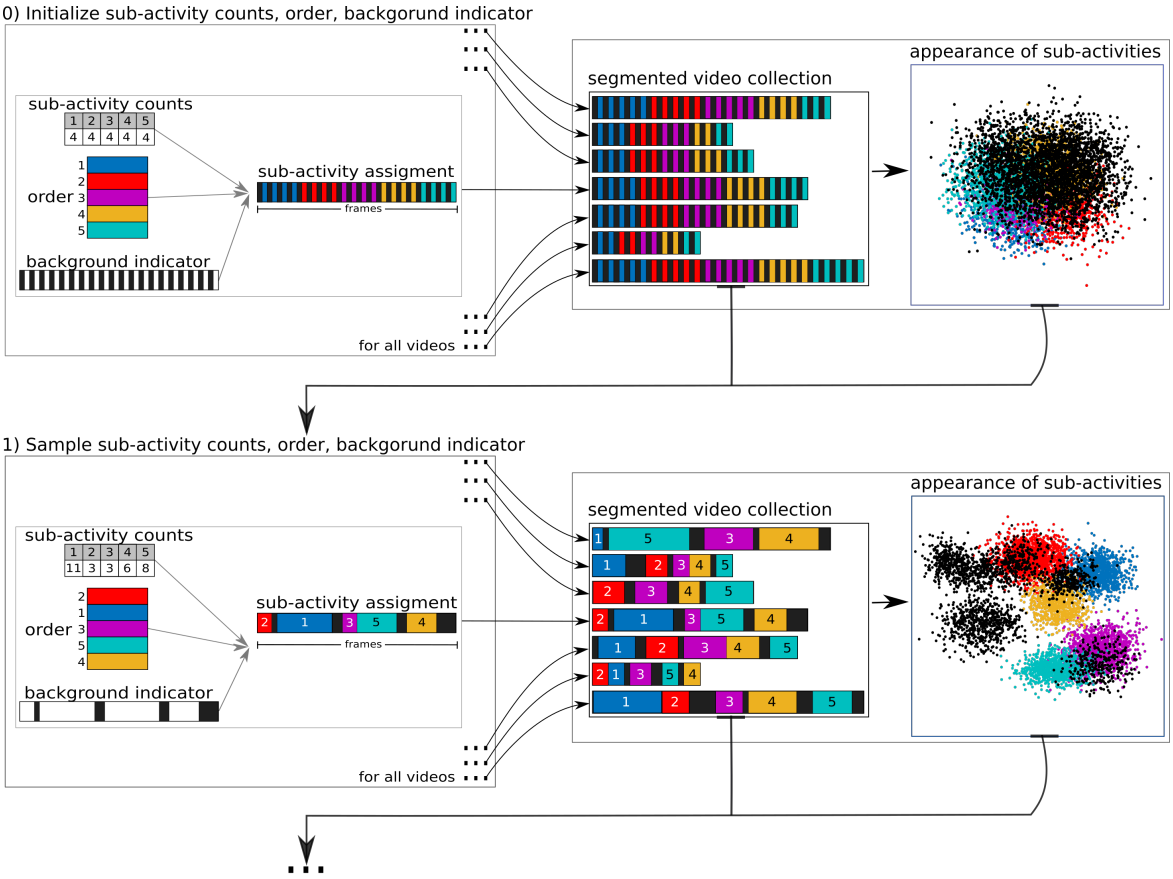


Figure 3.3: Overview: Our iterative model alternates between learning the visual appearance and temporal structure of the sub-activities. We combine the visual appearance with the temporal model to obtain a segmentation of the video sequence which is then used to update the visual appearance representation for the next iteration.

$$\mathbf{F}_{ij} = \mathbf{W}_a{}^\intercal \mathbf{W}_f X_{ij}, \tag{3.31}$$

where $\mathbf{F}_{ij} = [f^1, ..., f^K]_{ij}$. Each element f^k_{ij} is inversely proportional to the distance between X_{ij} and anchor point k in the latent space. By using K anchor points, this implies that $D = K$.

Our objective in learning the embeddings is to cluster the video features discriminatively, see Figure 3.5 (right). We achieve this by encouraging the X_{ij} belonging to the same sub-activity to cluster closely around a single anchor point while being far away from the other anchor points. If we assign each anchor point to a given sub-activity, then we can learn $\mathbf{W}$ by minimizing a pair-wise ranking loss L, where

$$L = \sum_{i,j}^{M,J_i} \sum_{k=1,k\neq k^*}^{K} \max[0, f^k_{ij} - f^{k^*}_{ij} + \Delta] + \gamma ||\mathbf{W}||^2_2. \tag{3.32}$$

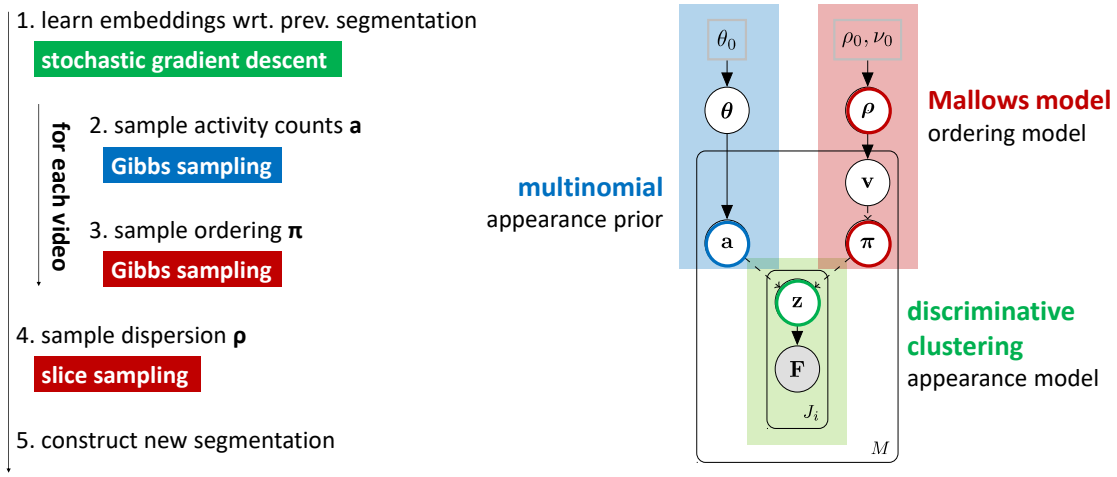


Figure 3.4: We present a plate model representation for our standard model without background along with the inference procedure. Given a segmentation $\mathbf{z}$, we can define an appearance model. The segmentation is composed of $\mathbf{a}$, the bag of activity counts, which we model as a multinomial, as well as $\boldsymbol{\pi}$, the ordering, based on the Mallows model. For segmenting videos, we iterate over the following: Given some initial or previous segmentation, we first embed the videos as a group, learned via stochastic gradient descent. For each video we then sample (Gibbs) activity counts, $\mathbf{a}$, and ordering, $\boldsymbol{\pi}$. Finally, the dispersion parameter, $\boldsymbol{\rho}$, is obtained via slice sampling, and a new segmentation, $\mathbf{z}$, is constructed for each video through $\mathbf{a}$ and $\boldsymbol{\pi}$.

In this loss, k^* is the anchor point associated with the true sub-activity label for F_{ij}, Δ is a margin parameter and γ the regularization constant for the l_2 regularizer of $\mathbf{W}$. The loss in Eq. 3.32 encourages the distance of X_{ij} in the latent space to be closer to the anchor point k^* associated with the true sub-activity than any other anchor point by a margin Δ.

The above formulation assumes that the right anchor point k^*, i.e., the true sub-activity label, is known. This is not the case in an unsupervised scenario so we follow an iterative approach where we learn $\mathbf{W}$ at each iteration from an assumed sub-activity based on the segmentation of the previous iteration. More details on learning are given in Section 3.5.4.

3.5.2 Standard Temporal Model

Given a collection of M videos of the same complex activity, we would like to infer the sub-activity assignments to video frames, $\mathbf{z} = \{\mathbf{z}_i\}, i \in \{1...M\}$. For video i, $\mathbf{z}_i = \{z_{ij}\}, j \in \{1...J_i\}$, $z_{ij} \in \{1...K\}$ can be assigned to one of K possible sub-activities. Note that for convenience, we overload the use of K for both the number of elements in the ordering for the GMM model as well as the number of sub-activities, as the two are equal when applying the GMM.

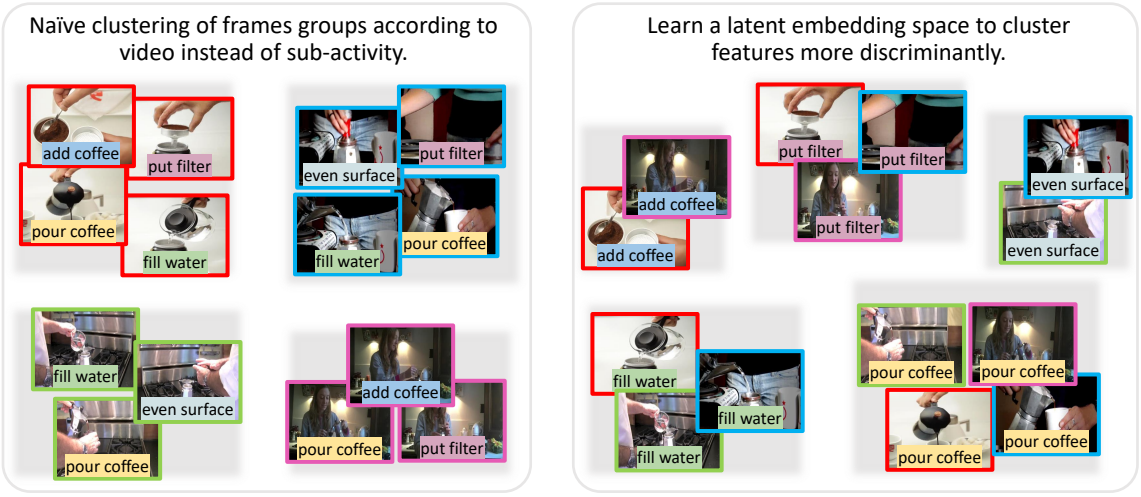


Figure 3.5: We compare the naïve clustering of frames with our discriminative clustering. Here, border colors correspond to videos, while sub-activity labels are shown as text. If we cluster frame features naïvely (left), what we end up with are clusters grouped according to the frames of each video, rather than according to the sub-activity of interest. Instead, we apply a discriminative clustering technique where we try to group frames according to their semantic similarity with respect to sub-activity rather than purely visual appearance (right).

We introduce $\mathbf{a}_i$, a bag of sub-activity labels for video i, i.e., the collection of elements in $\mathbf{z}_i$ but without consideration for the temporal frame ordering. The ordering is then described by $\boldsymbol{\pi}_i$. $\mathbf{a}_i$ is expressed as a vector of counts of the K possible sub-activities, while $\boldsymbol{\pi}_i$ is expressed as an ordered list. Together, $\mathbf{a}_i$ and $\boldsymbol{\pi}_i$ determine the sub-activity label assignments $\mathbf{z}_i$ to the frames of video i. $\mathbf{a}$ and $\boldsymbol{\pi}$ are redundant to $\mathbf{z}$; the extra set of variables gives us the flexibility to separately model the sub-activities' visual appearance (based on $\mathbf{a}$) and the temporal ordering (based on $\boldsymbol{\pi}$). We model $\mathbf{a}$ as a multinomial, with parameter $\boldsymbol{\theta}$ and a Dirichlet prior with hyperparameter θ_0. For the ordering $\boldsymbol{\pi}$, we use a GMM with the prior from Eq. 3.27 and hyperparameters ρ_0 and ν_0. The joint distribution of the model factorizes as follows:

$$\begin{aligned} P(\mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\rho}, \mathbf{F} | \theta_0, \rho_0, \nu_0) \\ &\propto P(\mathbf{F}|\mathbf{z}) P(\mathbf{a}|\boldsymbol{\theta}) P(\boldsymbol{\pi}|\boldsymbol{\rho}) P(\boldsymbol{\theta}|\theta_0) P(\boldsymbol{\rho}|\rho_0, \nu_0) \\ &= \Big[\prod_{i,j=1}^{M,J_i} P(\mathbf{F}_{ij}|z_{ij}) \Big] \Big[\prod_{i=1}^{M} P(\mathbf{a}_i|\boldsymbol{\theta}) P(\boldsymbol{\pi}_i|\boldsymbol{\rho}) \Big] \\ &= \Big[\prod_{k=1}^{K} P(\theta_k|\theta_0) \Big] \Big[\prod_{k=1}^{K-1} P(\rho_k|\rho_0, \nu_0) \Big], \end{aligned} \tag{3.33}$$

based on the assumption that each frame of each video as well as each video are all indepen-

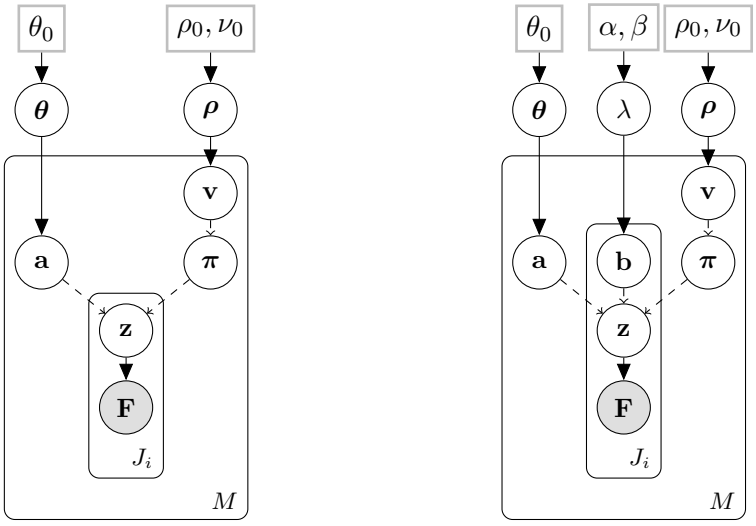


Figure 3.6: Plate diagrams of our model. Shaded nodes are observed variables; rectangles denote fixed hyperparameters with fixed values, and dashed arrows indicate deterministically constructed variables. Left: our standard model. Right: our full model with background.

dent observations. We show a diagram of the model in Figure 3.6 (left).

When using the GMM, performing maximum likelihood estimation to find a consensus or canonical ordering over a set of observed orderings is an NP hard problem, though several approximations have been proposed. Our case is the reverse, in which we assume that a canonical ordering is already given and we would like to find a (latent) set of orderings. Our interest is to infer the posterior $P(\mathbf{z}, \boldsymbol{\rho}|\mathbf{F}, \theta_0, \rho_0, \nu_0)$ for the entire video corpus. Directly working with this posterior is intractable, so we make Markov chain Monte Carlo (MCMC) sampling-based approximations. We integrate out all but three sets of hidden variables: sub-activity $\mathbf{a}$, orderings $\boldsymbol{\pi}$, and permutation inversion parameters $\boldsymbol{\rho}$. Specifically, we are using slice sampling (Neal, 2003) for $\boldsymbol{\rho}$ and Gibbs sampling (Griffiths and Steyvers, 2004) for $\mathbf{z}$. Since $\mathbf{z}$ is fully specified by $\mathbf{a}$ and $\boldsymbol{\pi}$, it is equivalent to sample $\mathbf{a}$ and $\boldsymbol{\pi}$. After a burn-in period, we treat the last samples of sub-activity $\mathbf{a}$, orderings $\boldsymbol{\pi}$, as a draw from the posterior. Before elaborating on the sampling equations, we first detail how we model the video likelihood $P(\mathbf{F}_i|\mathbf{z}_i)$.

3.5.2.1 Video Likelihood

Video likelihood $P(\mathbf{F}_i|\mathbf{z}_i)$ can be broken down into the product of frame likelihoods, since each frame is conditionally independent given the frame's sub-activity, i.e.,

$$P(\mathbf{F}_i|\mathbf{z}_i, \mathbf{F}_{\setminus i}, \mathbf{z}_{\setminus i}) = \prod_{j=1}^{J_i} P(\mathbf{F}_{ij}|z_{ij}, \mathbf{F}_{\setminus i}, \mathbf{z}_{\setminus i}). \tag{3.34}$$

Since our temporal model is generative, we need to make some assumptions about the generating process behind the video features. We directly model the frame likelihoods and use

K mixtures of Gaussians, one for each sub-activity k. Each mixture has Q components with weights ω_k, means $\boldsymbol{\mu}_k$ and covariances $\boldsymbol{\Sigma}_k$, with likelihood scores for each mixture selected according to the assignments z_{ij}:

$$P(\mathbf{F}_{ij}|z_{ij}=k,\mathbf{F}_{\setminus i},\mathbf{z}_{\setminus i}) \sim \sum_{q=1}^{Q} \omega_k^q \cdot \mathcal{N}(\boldsymbol{\mu}_k^q, \boldsymbol{\Sigma}_k). \tag{3.35}$$

3.5.2.2 Sampling Sub-activity Counts

Sampling sub-activity $\mathbf{a}_i$ is done with collapsed Gibbs sampling. Recall that $\mathbf{a}$ is modelled as a multinomial with K outcomes parameterized by $\boldsymbol{\theta}$. We sample a_{ij}, the j-th frame for video i, from the posterior conditioned on all other variables. Without the redundant terms, this posterior is expressed as

$$P(a_{ij}=k|\dots) \propto P(a_{ij}=k|\mathbf{a}_{\setminus ij},\theta_0) \cdot P(\mathbf{F}_i|\mathbf{z}_i,\mathbf{F}_{\setminus i},\mathbf{z}_{\setminus i}), \tag{3.36}$$

where the second term is the video likelihood from Eq. 3.34. The first term is a prior over the sub-activities, and can be estimated by integrating over $\boldsymbol{\theta}$. The integration is done via the collapsed Gibbs sampling and as we assumed a Dirichlet prior, i.e., $\boldsymbol{\theta} \sim Dir(\theta_0)$, this results in

$$P(a_{ij}=k|a_{\setminus ij},\theta_0) = \frac{N_k+\theta_0}{\sum_{k=1}^{K} N_k + K\theta_0}, \tag{3.37}$$

where N_k is the total number of times the sub-activity k is observed in all the sequences and $\sum_{k=1}^{K} N_k$ is the total number of sub-activity assignments. We compute a_{ij} over all possible topic assignments and select the new value for a_{ij} by sampling from this probability.

Note that sampling for a_{ij} does not correspond to the sub-activity assignment to the j-th frame. The assignment is given by z_{ij} which can only be computed after sampling a_{ij} for all J_i frames of video i and then re-ordering the bag of frames according to $\boldsymbol{\pi}_i$.

3.5.2.3 Sampling Orderings

Sampling ordering $\boldsymbol{\pi}_i$ is done via Gibbs sampling. Recall that the ordering follows a GMM as described in Section 3.4 and is parameterized for elements in the ordering individually via inversion count vector $\mathbf{v}_i$. As such, we sample a value for each position in the inversion count vector from $k=1$ to $K-1$ independently. The value states the mobility of the position according to:

$$P(v_{ik}=c|\mathbf{z},\boldsymbol{\rho},\mathbf{F}) \propto P(v_{ik}=c|\rho_k) \cdot P(\mathbf{F}_i|\mathbf{z}_i,\mathbf{F}_{\setminus i},\mathbf{z}_{\setminus i}), \tag{3.38}$$

where c indicates the inversion count assignment to v_{ik}. Again, the second term is the video likelihood from Eq. 3.34, while the first term corresponds to $P_{GMM}(v_{ik}=c;\rho_k)$, and is computed according to Eq. 3.25. We estimate the probability of every possible value of v_{ik}, which ranges from 0 to $K-k$, and sample a new inversion count value c based on these probabilities.

3.5.2.4 Sampling the Dispersion Parameter

We draw our ordering parameters $\boldsymbol{\rho}$ from the conjugate prior distribution of the GMM in Eq. 3.27. Sampling GMM dispersion parameter ρ_k is done for each sub-activity $k = 1$ to $K - 1$ independently. The number of trials, ν_0 and the number of previously encountered inversions $v_{k,0}$ are updated as follows

$$(\nu_0, v_{k,0}) = (N + \nu_0, \frac{\sum_i v_{ik} + v_{k,0}\nu_0}{N + \nu_0}). \tag{3.39}$$

Since the normalizing constant is unknown, it is not straightforward to sample from this distribution. The distribution is univariate and unimodal, we therefore draw ρ_k using slice sampling (Neal, 2003) from the conjugate prior distribution P_{GMM_0} according to Eq. 3.27 with the updated parameters. As GMM parameters do not have a direct effect on sub-activity assignments $\mathbf{z}$, we sample those parameters once after Gibbs sampling is done.

3.5.3 Background Modelling

To consider background, we extend the label assignment vector $\mathbf{z}$ with a binary indicator variable $b_{ij} \in \{0, 1\}$ for each frame. The indicator b_{ij} follows a Bernoulli variable parameterized by λ, with a beta prior, i.e., $\lambda \sim$ Beta(α, β). In this setting, $\mathbf{z}_i$ is determined by the bag of sub-activities $\mathbf{a}_i$, the ordering $\boldsymbol{\pi}_i$, and background vector $\mathbf{b}_i = \{b_{ij}\}$, where $\mathbf{b}_i$ indicates the frames to be excluded from sub-activity consideration.

As an example, for video i, given $\mathbf{a}_i = [6\ 3\ 5]$, $\boldsymbol{\pi}_i = [2\ 3\ 1]$ and $\mathbf{b}_i = [1\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1]$, the sub-activity assignment is $\mathbf{z}_i = [2\ 2\ 2\ 0\ 0\ 3\ 3\ 3\ 0\ 0\ 3\ 3\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1]$. We show a diagram of this model in Figure 3.6 (right).

The joint distribution of the model can be expressed as

$$P(\mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\rho}, \mathbf{F}|\theta_0, \alpha, \beta, \rho_0, \nu_0,) \propto P(\mathbf{a}|\boldsymbol{\theta}, \theta_0) \cdot P(\boldsymbol{\pi}|\boldsymbol{\rho}, \rho_0, \nu_0) \cdot P(\mathbf{b}|\lambda, \alpha, \beta) \cdot P(\mathbf{F}|\mathbf{a}, \boldsymbol{\pi}, \mathbf{b}). \tag{3.40}$$

Drawing samples from this full model requires a small modification to the sub-activity sampling $\mathbf{a}_i$. More specifically, we need a Gibbs sampler that samples a_{ij} and b_{ij} jointly while integrating over $\boldsymbol{\theta}$ and λ.

3.5.3.1 Sampling Background

Sampling background $\mathbf{b}_i$ is done from the joint conditional

$$P(b_{ij}, a_{ij}| \dots) \propto P(b_{ij}|\alpha, \beta) \cdot P(a_{ij}|\mathbf{a}_{\setminus ij}, \theta_0) \cdot P(\mathbf{F}_i|\mathbf{z}_i, \mathbf{F}_{\setminus i}, \mathbf{z}_{\setminus i}). \tag{3.41}$$

This is equivalent to the following for a sub-activity frame:

$$P(b_{ij}=1, a_{ij}=k|\dots) \propto \frac{N_f+\alpha}{N_f+N_b+\alpha+\beta} \cdot \frac{N_k+\theta_0}{\sum_{k=1}^{K} N_k + K\theta_0} \cdot P(\mathbf{F}_i|b_{ij}=0, a_{ij}=k, \mathbf{F}_{\setminus i}, \mathbf{z}_{\setminus ij}), \tag{3.42}$$

where N_f and N_b are the total number of sub-activity frames and background frames in the corpus respectively. For a background frame, the joint distribution is equal to

$$P(b_{ij}=0, a_{ij}|\dots) \propto \frac{N_b+\alpha}{N_f+N_b+\alpha+\beta} \cdot P(\mathbf{F}_i|b_{ij}=1, a_{ij}, \mathbf{F}_{\setminus i}, \mathbf{z}_{\setminus ij}). \tag{3.43}$$

The video likelihood in Eqs. 3.42 and 3.43 are computed in a similar way as defined in Eqs. 3.34 and 3.35, with the exception that we now iterate over the joint states of background and sub-activity labels for the frame likelihoods. Note that this only adds one extra probability in being computed, i.e., $b=1$, since the state of a_{ij} is then irrelevant. The rest of the the Gibbs sampling remains the same.

Algorithm 3.4 The algorithm for our full model.

Require: K, Q, $\mathbf{X}, \theta_0, \alpha, \beta, \rho_0, \nu_0$
Ensure: $\mathbf{z}$ and $\boldsymbol{\rho}$

initialize $\mathbf{a}, \mathbf{b}, \boldsymbol{\pi}$ and construct $\mathbf{z}$
randomly initialize $\mathbf{W}$
for each iteration **do**
  learn $\mathbf{W}$ with $\mathbf{z}$
  for $k = 1$ to K **do**
    learn $\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
  for $i = 1$ to M **do**
    for $j = 1$ to J_i **do**
      for $k = 1$ to K **do**
        $P(b_{ij} = 1, a_{ij} = k|\dots) \leftarrow$ Eq. 3.42
      $P(b_{ij} = 0, a_{ij}|\dots) \leftarrow$ Eq. 3.43
    $\{a_{ij}, b_{ij}\} \leftarrow$ draw from $P(b_{ij}, a_{ij}|\dots)$
    for $k = 1$ to $K - 1$ **do**
      $P(v_{ik}|\dots) \leftarrow$ Eq. 3.38
    $v_{ij} \leftarrow$ draw from $P(v_{ik}|\dots)$
  for $k = 1$ to $K - 1$ **do**
    $P(\rho_k|\dots) \leftarrow$ Eq. 3.27
  construct $\mathbf{z}$ with new $\mathbf{a}, \mathbf{b}, \boldsymbol{\pi}$

3.5.4 Inference Procedure

Our model's inputs are the frame features $\mathbf{X}$, the number of sub-activities K and the number of Gaussian mixtures Q. We iterate between solving for $\mathbf{F}$ and sampling $\mathbf{z}$ and $\boldsymbol{\rho}$

from the posterior $P(\mathbf{z}, \boldsymbol{\rho}|\mathbf{F}, \theta_0, \alpha, \beta, \rho_0, \nu_0)$.

The inference process is summarized in Algorithm 3.4. To initialize $\mathbf{z}_i$ for each video i, the sub-activity counts $\mathbf{a}_i$ are split uniformly over K sub-activities; $\boldsymbol{\pi}_i$ is set to the canonical ordering; $\mathbf{b}_i$ is set with every other frame being background (see schematic in Figure 3.3). Using the current assignments $\mathbf{z}$, we first learn $\mathbf{W}$ of the latent embeddings to solve for $\mathbf{F}$ and then for each sub-activity k, (step 4). Then using the same $\mathbf{z}$ and $\mathbf{F}$ we learn K mixture of Gaussian, one for each sub-activity with the Gaussian mixture components $\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$, (step 6).

Next, for each video i, we proceed to re-sample $\{\mathbf{a}_i, \mathbf{b}_i\}$, $\boldsymbol{\pi}_i$, in that order, using Gibbs sampling using equations 3.42, 3.43 and 3.38 respectively (steps 10, 11, 14). These variables are used to construct $\mathbf{z}_i$. After repeating for each video, we can then re-sample the dispersion parameter $\boldsymbol{\rho}$ according to Eq. 3.27, step 17. From the new $\mathbf{z}$ and $\boldsymbol{\rho}$, we then repeat the steps between 4-18 until our stopping criteria is reached.

3.5.5 Implementation Details

To optimize Eq. 3.32 for learning $\mathbf{W}$, we use Stochastic Gradient Descent (SGD) with mini-batches of 200 and momentum of 0.9. We set the hyperparameters $\rho_0 = 1$, $\alpha = 0.2$, $\beta = 0.2$, $\nu_0 = 0.1$ times the number of videos, $\theta_0 = 0.1$. The learning rate we use for the Instructional Videos Dataset (Alayrac et al., 2016) is 1^{-5} and for the Breakfast Actions Dataset (Kuehne et al., 2014b) it is 1^{-7}.

3.6 Experimentation

3.6.1 Datasets & Evaluation Metrics

We analyze our model's performance on two challenging datasets, Breakfast Actions (Kuehne et al., 2014b) and Inria Instructional Videos (Alayrac et al., 2016). Breakfast Actions has 1,712 videos of 52 participants performing ten breakfast preparation activities such as *"making coffee"* and *"scrambling eggs"*. There are 48 sub-activities; videos vary according to the participants' preference of preparation style and orderings. We use the visual features from Kuehne et al. (2016) based on improved dense trajectories (Wang and Schmid, 2013), encoded with Fisher vectors (Sánchez et al., 2013) and reduced to 64 dimensions via PCA. This dataset has no background and sub-activities transition directly from one to another.

Inria Instructional Videos contains 150 narrated videos of 5 instructional activities collected from YouTube using the activity name as the search term. The videos are on average 2 minutes long with 47 sub-activities. We use the visual features provided by Alayrac et al. (2016), which are based on improved dense trajectories and VGG-16 (Simonyan and Zisserman, 2014b) conv5 layer responses taken over multiple windows per frame. The trajectory and CNN features are each encoded with bag-of-words into vocabularies of 2000 and 1000 respectively and concatenated into a 3000-dimensional vector for each frame. The videos are labelled with background, i.e., frames in which the sub-activity is not visually discernible, usually when the person stops to explain upcoming steps. As such, the sub-activities are

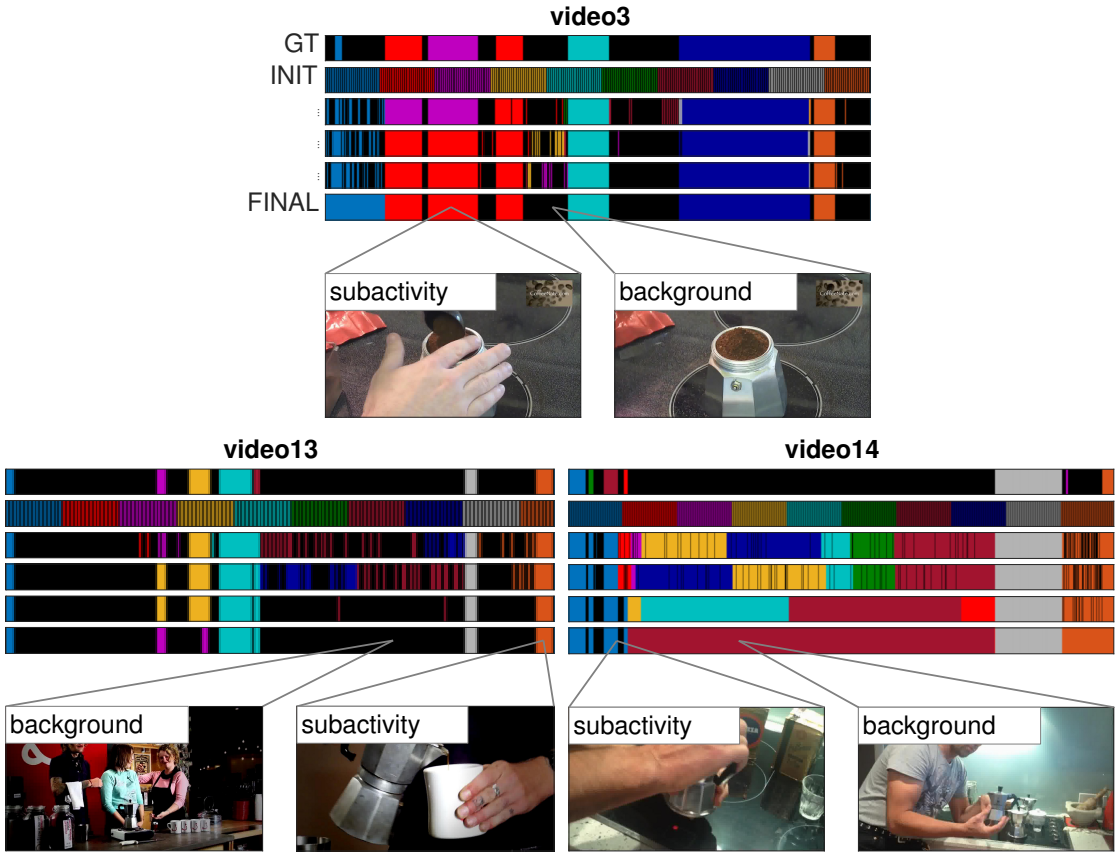


Figure 3.7: Segmentation outputs on three *"making coffee"* examples from Inria Instructional Videos Dataset (Alayrac et al., 2016), from video 3, 13 and 14 respectively. Colors indicate different sub-activities, black the background frames. Since our algorithm is fully unsupervised, we established one-to-one color mappings between the ground truth and our outputs for visualization purposes. The horizontal axis is time and below we show some example sub-activity and background frames. The first row (GT) is the ground truth; the second until the last rows show the progression from the initialization (INIT), some iterations, and the (FINAL) segmentation. Our method performs well when the appearance of the sub-activities is discriminative, e.g., for video 3, occurrence of a hand during a sub-activity vs. none during the background frames, or people talking for video 13. We fail in detecting background when there are also interactions with objects of interest, e.g., in video 14. Our model does not enforce continuity over the background frames and may result in fragmentation, but as shown, with good appearance modelling, the background clusters naturally. Furthermore, the final segmentations may contain a different number of sub-activities while still maintaining a global order, e.g., the orange sub-activity tends to appear last and follows the grey one.

separated by hundreds of background frames (73% of all frames). We evaluate our standard model without background modelling by removing these frames from the sequence as well as

our model with background modelling on the original sequences.

To evaluate our segmentation results in the fully unsupervised setting, we need one-to-one mappings between the segment and ground truth labels. In line with (Alayrac et al., 2016; Sener et al., 2015), we use the approach described by Liao (2005) who use the Hungarian method to find the mapping that maximizes the evaluation scores. We use three evaluation metrics. The mean over frames (MoF) evaluates the quality of temporal localization of sub-activities with the percentage of frames correctly labelled. The Jaccard index, computed as intersection-over-detections, quantifies the difference between ground-truth and predicted segmentations. Finally, we use the F1 score that ranges between between 0 and 1 to compare against the state of the art on the Instructional Videos Dataset (Alayrac et al., 2016). With all three measures, higher values indicate better performance.

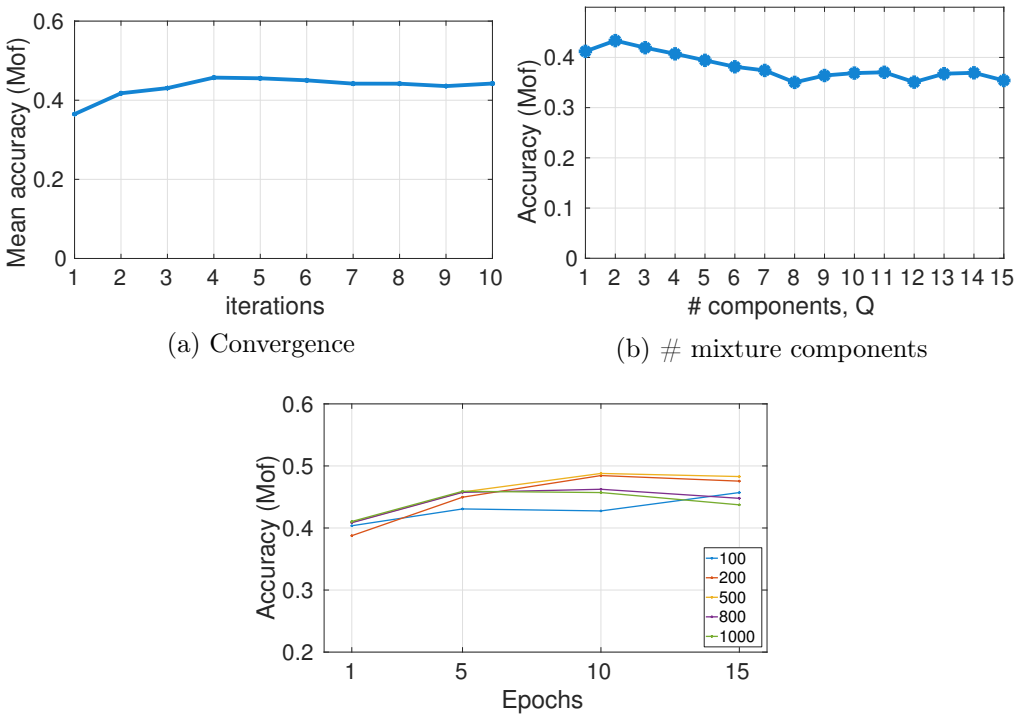
We also show a partly supervised baseline in which we use ground truth sub-activity labels for learning $\mathbf{F}$ but learn the temporal alignments unsupervised. This can be thought of as an upper bound on performance for our fully unsupervised version, in which we iteratively learn the temporal alignment and discover the visual appearance of the sub-activities. We refer to these to as *"ours GT"* and *"ours iterated"* respectively in the experimental results.

3.6.2 Sub-Activity Representation Modelling

By projecting the frames' visual features and the sub-activity labels into a joint feature space, we learn a visual appearance model for the sub-activities. We first consider our standard model on Inria Instructional Videos with the background frames removed. The plot in Figure 3.8(a) tells us that the appearance model can be learned successfully in an iterative fashion and begins to stabilize after approximately 5 iterations between learning the sub-activity appearance and the GMM.

Our model's performance depending on the the number of Gaussian mixture components Q is shown in Figure 3.8(b). The resulting sub-activity representations are very low dimensional and highly separable so that we achieve higher MoF with a few number of components. We use $Q = 3$ mixture components for our iterative and $Q = 1$ for the ground truth experiments.

In Figure 3.8(c), we use our iterated method to show the MoF for different values of E, or embedding dimensionality, over the training epochs. We find only small differences in MoF for different E values. We fix the embedding size $E = 200$ with 12 epochs of training and 5 iterations of sub-activity and GMM learning for subsequent experiments on both datasets. The run time of a single iteration of our algorithm is proportional to the number of frames J_i in each video and the assumed number of sub-actvities K. We evaluate the performance of our system, on a computer with an Intel Core i7 3.30 GHz CPU. Our model, for a single iteration, takes approximately 115 seconds (109 for learning the sub-activity appearance model and 6 seconds for estimating the temporal structure).



(a) Convergence

(b) # mixture components

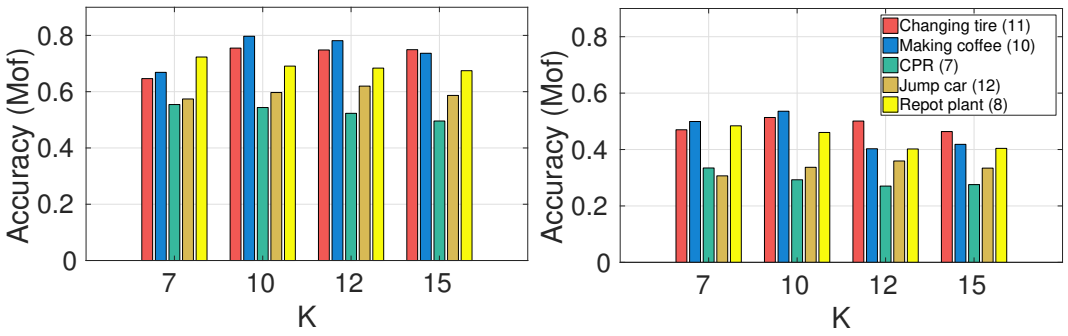
(c) Dimensionality

Figure 3.8: Influence of our model's parameters are tested on the Instructional Videos Dataset (Alayrac et al., 2016) without background frames. We set K to the ground truth sub-activity number of the all five activities. Our method's performance over the iterations is shown in (a), using different numbers of Gaussian mixture components in (b) and dimensionality of embedding space in (c).

3.6.3 Temporal Structure Modelling

The GMM models temporal ordering – without it, one can only classify each frame's sub-activity label based on the visual appearance. Even if these appearance models are trained on ground truth, the segmentation results would be very poor. On Inria Instructional Videos without background, the MoF is 32.2 without versus 69.2 with the GMM (see Figure 3.9).

The only GMM parameter is K, the number of assumed sub-activities. We again consider the Inria Instructional Videos without background and show the MoF as a function of K, once partially unsupervised (sub-activity appearance model from ground truth) and once fully unsupervised in Figure 3.9(a) and (b) respectively. As can be expected, the MoF drops when moving from the partially to the fully unsupervised case. This drop can be attributed to the fact that the Inria Instructional Videos dataset is extremely difficult and exhibits a lot of variation across the videos. In both partially and fully unsupervised scenarios, however, the MoF remains stable with respect to K, demonstrating that our method is quite robust with respect to varying K. This is also the case once background is considered in the full



(a) No background, GT (b)No background, iterated

Figure 3.9: Our results on Instructional Videos (Alayrac et al., 2016) without background frames with varying K. The legend gives the ground truth K for each subactivity in braces, for example "changing tire (11)", where the ground truth number of sub-activities is 11. Each color correspond to a different activity. Results are reported as mean over frames (MoF).

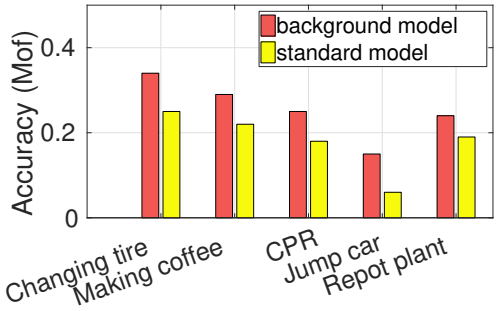


Figure 3.10: Our standard model vs. background model on original Inria Instructional Videos sequences. The fractions of background are *changing tire (0.46), making coffee (0.71), perform CPR (0.56), jump car (0.83) and repot plant (0.66).*

model with the original sequences (see Figure 3.11).

3.6.4 Background Modelling

In Figure 3.10, we demonstrate the effectiveness of our full model in capturing the background in the original sequences in Inria Instructional Videos. Figure 3.10 shows the improvement in F1 score once the background is accounted for as a part of the model; there are improvements on every activity, with the most significant being a three-fold increase for *'Jump car'* despite the sequences being 83% background. In Figure 3.7, we show qualitative examples of how our model copes with background, where it succeeds and where it fails.

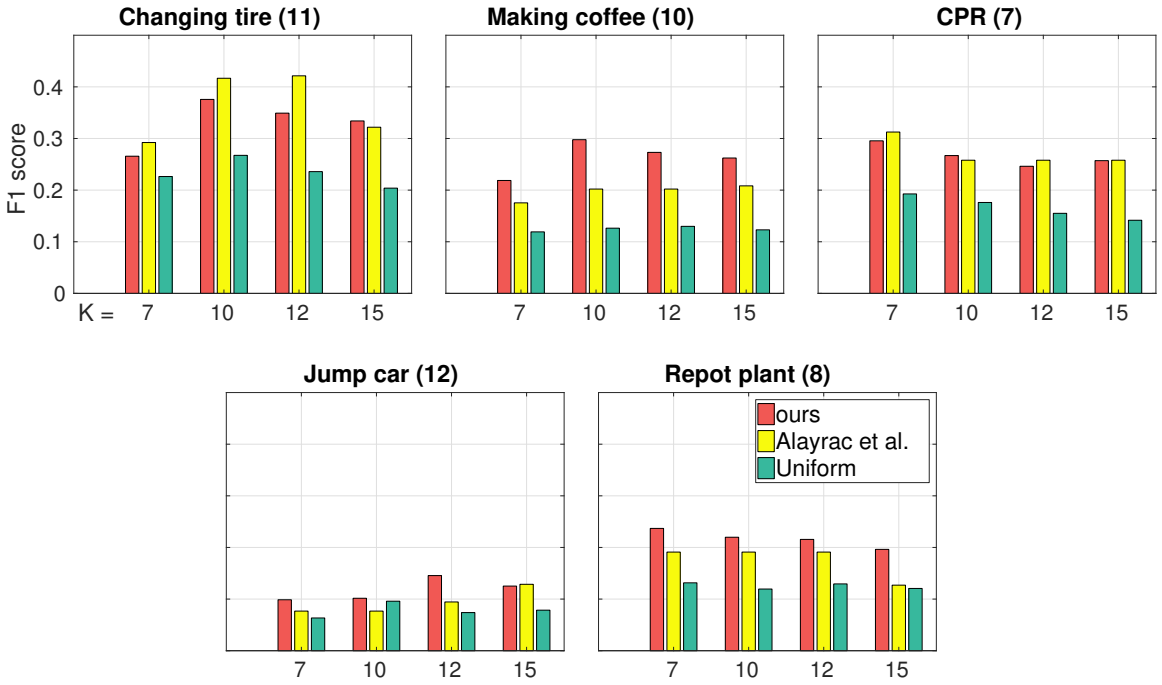


Figure 3.11: Comparison of our method with Alayrac et al. (2016) on the Instructional Videos Dataset. To be compatible to the key step detection of Alayrac et al. (2016), we report the mean over 15 randomly selected frame from each segment.

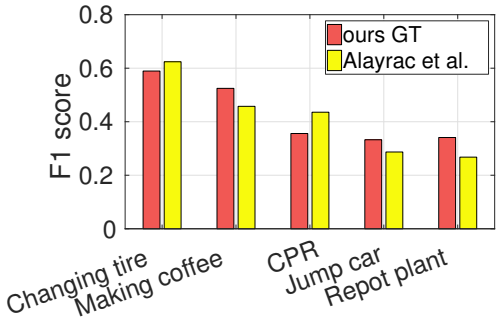


Figure 3.12: Comparison of our supervised setting against Alayrac et al. (2016)'s supervised method on the Instructional Videos Dataset. Here, our model learns the sub-activity appearance from the ground truth annotations. Alayrac et al. (2016) use the ground truth annotations as constraints for their discriminative clustering based algorithm.

3.6.5 Comparison to the State of the Art

3.6.5.1 Inria Instructional Videos

We compare our full model to Alayrac et al. (2016) in Figure 3.11 on the Inria Instructional Videos dataset. The method of Alayrac et al. (2016) outputs a single representative frame for

each sub-activity and reports an F1 score on this single frame. To make a valid comparison, since our work is aimed at finding entire segments, we randomly select a frame from each segment and then find a one-to-one mapping between our prediction set and the ground truth annotations based on the method proposed by Liao (2005). We report our results over this prediction set by the F1 score and compare against Alayrac et al. (2016).

Our performance across the five activities is consistent and varies much less than Alayrac et al. (2016). We have stronger performance in three out of five activities, while we are worse on *'perform cpr'* and *'changing tire'*. The GMM is a distribution on permutations and orderings; it is by definition unable to account for repeating sub-activities but in *'perform CPR'*, *'give breath'* and *'do compression'* are repeated multiple times and account for more than 50% of the sequence frames. In general, we attribute our stronger performance to the fact that the GMM can model flexible sub-activity orderings, while Alayrac et al. (2016) enforces a strict ordering. The GMM parameter $\boldsymbol{\rho}$ has a prior with hyperparameter ρ_o (Sec. 3.4). A smaller ρ_0 allows more flexible orderings, while a larger ρ_0 encourages the ordering $\boldsymbol{\pi}$ to remain similar to the canonical ordering $\boldsymbol{\sigma}$. In all of our reported results, we fixed $\rho_0 = 1$. We find that for an activity such as *'change tire'*, which follows a strict ordering, a larger ρ_0 is more appropriate; with $\rho_0 = 5$ we are comparable to Alayrac et al. (2016) (0.41 vs. 0.42 F1 score). For *'jump car'* our method outperforms Alayrac et al. (2016), however our overall performance is the lowest as our model struggles with separating the visually very similar *'remove cable A'* and *'remove cable B'*. We observe a similar trend when we learn the sub-activity appearances using ground truth labels (see Figure 3.12).

Supervision Level	Method	MoF	Jaccard
Fully Supervised	SVM (Huang et al., 2016)	15.8	-
	HTK (Kuehne et al., 2014b)	19.7	-
Weakly Supervised	OCDC (Bojanowski et al., 2014)	8.9	23.4
	HTK (Kuehne et al., 2014b)	25.9	-
	ECTC (Huang et al., 2016)	27.7	-
	Fine2Coarse (Richard et al., 2017)	33.3	47.3
Unsupervised	ours iterated	**34.6**	47.1

Table 3.1: Comparisons on the Breakfast Actions dataset (Kuehne et al., 2014b). Methods are evaluated according to mean over frames (MoF) and Jaccard index. For both, a higher result indicates better performance.

3.6.5.2 Breakfast Actions

This dataset has no background labels so we apply our standard model and compare with other fully supervised and semi-supervised approaches in Table 3.1. Of the supervised methods, the SVM method (Huang et al., 2016) classifies each frame individually without any temporal consideration and achieves an MoF of 15.8%. This shows the strength (or necessity) of temporal information.

Our full model, *'ours iterated'*, is the only fully unsupervised method. We set K, the number of sub-activities in the complex activity, based on ground truth. In comparison, the weakly supervised methods (Huang et al., 2016; Richard et al., 2017; Bojanowski et al., 2014) require both K as well as an ordered list of sub-activities as input. OCDC (Bojanowski et al., 2014) is based on discriminative clustering, while ECTC (Huang et al., 2016) and Fine2Coarse (Richard et al., 2017) are RNN and HMM-based methods. We find that our fully unsupervised approach has performance that is the state of the art compared to all previous methods. We also tested our method with I3D (Carreira and Zisserman, 2017) features and obtained a MoF of 42.5% which shows us that there is room for improvement for our method using better feature representations.

3.7 Conclusion and Future Works

In this work we present an unsupervised method for partitioning complex activity videos into coherent segments of sub-activities. We learn a function assigning sub-activity scores to a video frame's visual features and we model the distribution over sub-activity permutations by a Generalized Mallows Model (GMM). Furthermore, we account for background frames not contributing to the actual activity.

We successfully test our method on two datasets of this challenging problem and are either comparable to or out-perform the state of the art, even though our method is completely unsupervised, in contrast to the existing work. Our method is able to produce coherent segments, at the same time being flexible enough to allow missing steps and variations in ordering. Our model's performance drops slightly for complex activities including repetitive sub-activities, as the GMM does not allow for such repeating structures.

As GMM cannot handle repetitions in the ordering of sub-activities, a potential future work could be approaching GMM in a hierarchical manner to handle repeating blocks as a single step, which can then be further subdivided. Additionally, our framework could be extended to generate segmentation outputs for multi-modal data, which could be in the form of multi-view video data as in the Breakfast Actions dataset (Kuehne et al., 2014b) or video, audio, and motion capture data as in the CMU-Kitchen dataset (Torre et al., 2008).

Chapter 4

Zero-Shot Anticipation for Instructional Activities

Contents

In this chapter, we present our method for zero-shot action anticipation on instructional activity videos. The content of this chapter corresponds to our ICCV 2019 publication, *Zero-shot Anticipation for Instructional Activities* (Sener and Yao, 2019). Given instructional observations, action anticipation aims at predicting future actions without seeing the future frames. Within the action recognition community, the models developed for instructional tasks mainly follow a traditional supervised setting, where the test time tasks are available as multiple demonstrations during training. The key to gain human-level intelligence is to generalize from a few or zero demonstrations. To learn a new task, humans do not need

significantly many demonstrations as they can transfer experience from previous tasks to new environmental conditions. Motivated by the abilities of humans who quickly learn and adapt to new tasks using prior experience, we develop our models for predicting future actions on never-before-seen instructional tasks (see Figure 4.1).

Figure 4.1: Action anticipation methods tackle the problem of predicting future actions before observing their frames. In our work, we are interested in zero-shot action anticipation where our models might be trained on making all sorts of cookie videos but making chocolate chip cookies. During inference, our models make future predictions on never-before-seen chocolate chip cookies videos.

How can we predict what will happen next to an activity we have never seen before? We address this problem of zero-shot anticipation by presenting a hierarchical model that generalizes instructional knowledge from large-scale text-corpora and transfers the knowledge to the visual domain. Given a portion of an instructional video, our model predicts coherent and plausible actions multiple steps into the future, all in rich natural language. To demonstrate the anticipation capabilities of our model, we collected a new dataset, the *Tasty Videos dataset*, a collection of cooking recipes that could be used for zero-shot learning, recognition, and anticipation.

4.1 Introduction

Imagine a not-so-distant future, where a kitchen is serviced by a robot chef[1]. How should we teach robots to cook? By reading all the recipes on the web? By watching all the cooking videos on YouTube? The ability to learn and generalize from a set of instructions, be it in text, image, or video form, is a highly challenging and open problem faced by those working in machine learning and robotics.

In this work, we limit our scope of training the next robo-chef to predict subsequent steps as it watches a human cook a never-before-seen dish. We frame our problem as one of future action prediction in a zero- and/or few-shot learning scenario. This best reflects the situation under which service robots will be introduced (Finn et al., 2017; Sünderhauf et al., 2018). The robot is pre-trained extensively, but not necessarily with knowledge matching exactly the deployment environment, thereby forcing it to generalize from prior knowledge. At the same time, the robot needs to anticipate what will happen in the future, to ensure a safe and smooth collaborative experience with the human (Koppula and Saxena, 2015; Wu et al., 2016).

[1]Robots cooking specific recipes (Moley, 2018; Beetz et al., 2011; Tenorth et al., 2013) already exist!

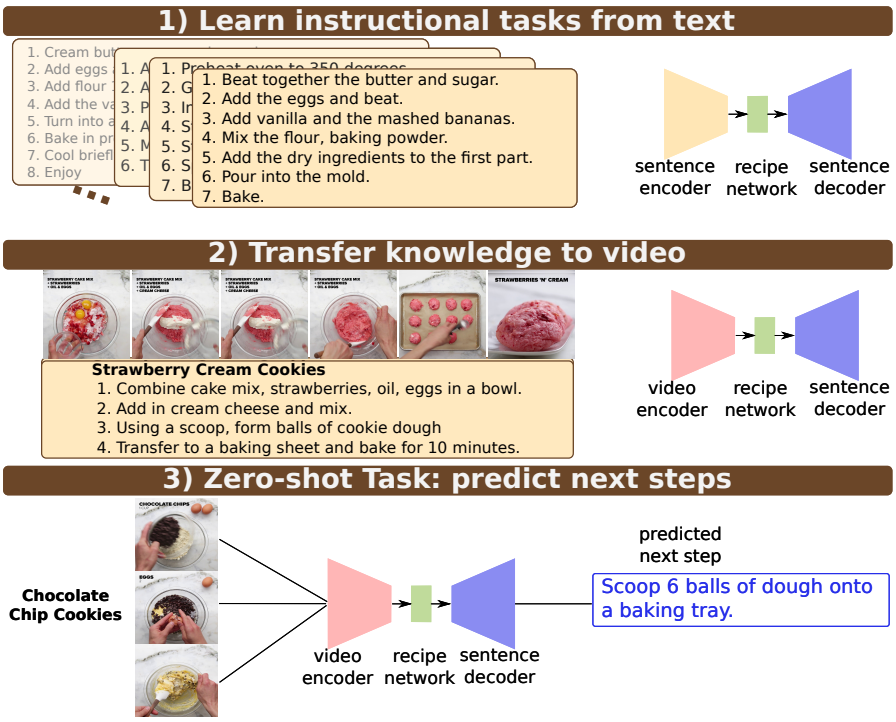


Figure 4.2: An overview of our model. We first learn procedural knowledge from large text corpora and transfer it to the visual domain to anticipate the future. Our system is composed of four Recurrent Neural Networks (RNNs): a sentence encoder, a sentence decoder, a video encoder, and a recipe network.

Instructional data and, in particular, cooking recipes can be readily found on the web (Instructables, 2005; Wikihow, 2005). The richest forms are multi-modal, e.g., images plus text, or videos with narrations. Such data fits well into our scenario in which the service robot visually recognizes the current context and makes future predictions. However, learning complex, multi-step activities requires significant amounts of data, and despite their online abundance, it is still challenging to find sufficient examples in multi-modal form. Furthermore, learning the visual appearance of specific steps would require temporally aligned data, which is less common and/or expensive to obtain. Our strategy is, therefore, to separate the procedural learning from visual appearance learning. Procedural knowledge is learned from text, which is readily available in large corpora on the scale of millions (Salvador et al., 2017). This knowledge is then transferred to video so that the learning of visual appearances can then be simplified to only a grounding model done via aligned video and text. More specifically, we encode text and/or video into fixed-length context vectors. The context is fed to a recipe network, which models the sequential structure of the recipe and makes following step predictions in vector form, which are then decoded back into sentences using a sentence decoder. The text encoder is used only for training, while the video encoder is applied at test time. An overview is shown in Figure 4.2.

Our work is highly novel in two key regards. First and foremost, we are working with

zero-shot action anticipation under a semi-supervised setting, as we target prediction for never-before-seen dishes. We achieve this by generalizing cooking knowledge from large-scale text corpora and then transferring the knowledge to the visual domain. This relieves us of the burden and impracticality of providing annotations for a domain in which there are virtually unlimited categories (dishes) and sub-categories (instructional steps). We are the first to tackle such a problem in this form; prior works in complex activity recognition are severely limited in the number of categories and steps (Alayrac et al., 2016; Kuehne et al., 2014b,a; Rohrbach et al., 2012), while works in action anticipation rely on strong supervision (Farha et al., 2018; Lan et al., 2014; Zhou and Berg, 2015).

Second, we do not work with closed categories derived from word tags; instead, we train with and also predict full sentences, e.g., *"Cook the chicken wing until both sides are golden brown."* vs. *"cook chicken"*. This design choice makes our problem significantly more challenging, but also offers several advantages. First of all, it adds richness to the instruction, since natural language conveys much more information than simple text labels (Lin et al., 2015; Zhou et al., 2018b). It also allows for anticipation of not only actions but also objects and attributes. Finally, as a byproduct, it facilitates data collection, as the number of class-based annotations grows exponentially with the number of actions, objects, and attributes and leads to very long-tailed distributions (Damen et al., 2018).

When transferring knowledge from text recipes to videos, we need to ground the two domains; video with temporally aligned captions. To the best of our knowledge, YouCookII (Zhou et al., 2018b) is currently the only dataset with such labels. However, it lacks diversity in the number of dishes and, therefore, different recipe steps. As such, we collect and present our new *Tasty Videos dataset*, a diverse set of 2511 different cooking recipes[2] accompanied by a video, ingredient list, and temporally aligned recipe steps. Video footage is taken from a fixed birds-eye view and focuses almost exclusively on the cooking instructions, making it well-suited for understanding the procedural steps.

We summarize our main contributions as follows:

- We are the first to explore zero-shot action anticipation by generalizing knowledge from large-scale text-corpora and transferring it to the visual domain.
- We propose a modular hierarchical model for learning multi-step procedures with text and visual context.
- Our model generalizes cooking knowledge and is able to predict coherent and plausible instructions for multiple steps into the future. The predictions, in rich natural language, score higher in standard NLP metrics than state-of-the-art video captioning methods, which observe the visual data on both YouCookII and Tasty Videos.
- We demonstrate how the proposed approach can be useful for making future step predictions in a zero-shot scenario compared to a supervised setting.

[2]Collected from the website `https://tasty.co/`

- We present a new and highly diverse dataset of cooking recipes which is publicly available[3] and will be of interest for those working in anticipation, complex activity recognition and video captioning.

4.2 Related Works

Complex Activity Understanding:

Understanding complex activities and their sub-activities have been addressed typically as a supervised temporal video segmentation and recognition problem (Kuehne et al., 2014b; Richard and Gall, 2016; Rohrbach et al., 2012). However, annotating every frame in video can be expensive, making it difficult to work at a large scale. Newer alternative lines of work are either weakly-supervised, using cues from accompanying narrations (Alayrac et al., 2016; Malmaud et al., 2015; Sener et al., 2015) or known orderings of the sub-activities (Huang et al., 2016; Richard et al., 2017; Bojanowski et al., 2014), or are fully unsupervised (Sener and Yao, 2018; Kukleva et al., 2019). Our work is similar to those using text cues; however, we do not rely on aligned visual text data for learning the activity models (Alayrac et al., 2016; Sener et al., 2015). Instead, we use a large corpus of unlabeled data in the text domain and apply only a very small set of aligned data for grounding the visual evidence.

Action Anticipation:

Action prediction is a new but fast-growing topic within computer vision. In the past, the closely related task of early event recognition, in which on-going or incomplete actions are recognized, was sometimes referred to in literature as action prediction (Ryoo, 2011; Hoai and De la Torre, 2014). However, this term is a misnomer since at least a portion of the action has already been observed and models (Ryoo, 2011; Hoai and De la Torre, 2014; Xu et al., 2015a) are simply performing inference with incomplete data.

Prior work in forecasting activities before making *any* observations have been limited to simple movement primitives such as *reaching, moving, placing, etc.* (Koppula and Saxena, 2015), or personal interactions such as *hand-shaking, hugging etc.* (Lan et al., 2014; Vondrick et al., 2016). Single predictions are made, and the anticipated actions typically occur within a few seconds time frame. Recently, Farha et al. (2018) predicts multiple actions into the future; our method also predicts multiple steps into the future, but unlike Farha et al. (2018), we do not work in a fully supervised framework and do not require repetitions of activity sequences for training.

Cooking Data:

The cooking domain is popular in NLP research since recipes are rich in natural language yet are reasonably limited in scope. Modelling the procedural aspects of the text and generating coherent recipes span several decades of work (Dale, 1989; Hammond, 1986; Morris

[3]Tasty Video Dataset https://cvml.comp.nus.edu.sg/tasty

et al., 2012; Mori et al., 2014; Kiddon et al., 2016; Bosselut et al., 2018; Salvador et al., 2019). In multimedia, recipes are involved in tasks such as food recognition (Herranz et al., 2018), recommender systems (Min et al., 2017) and indexing and retrieval (Carvalho et al., 2018; Salvador et al., 2017).

In computer vision, cooking has been well-explored for complex and fine-grained activity recognition (Damen et al., 2018; Kuehne et al., 2014a; Rohrbach et al., 2014, 2012; Zhou et al., 2018b), temporal segmentation (Kuehne et al., 2014a; Zhou et al., 2018b) and captioning (Rohrbach et al., 2013; Regneri et al., 2013; Zhou et al., 2018c). Several cooking and kitchen-related datasets have been presented (Damen et al., 2018; Malmaud et al., 2015; Sener et al., 2015; Kuehne et al., 2014b; Zhou et al., 2018b) and feature a wide variety of labels depending on the task. What's Cooking (Malmaud et al., 2015) and YouCookII (Zhou et al., 2018b) are similar to our new dataset, in that they include recipe texts and accompanying videos. However, YouCookII has limited diversity in activities with only 89 dishes, and What's Cooking is larger in scale, but lacks temporal alignments between recipe texts and videos.

4.3 Modelling Sequential Instructions

Sequence-to-sequence learning (Sutskever et al., 2014) has made it possible to successfully generate continuous text and build dialogue systems (Cho et al., 2014; Vinyals and Le, 2015). Recurrent neural networks (RNNs) are used to learn rich representations of sentences (Hill et al., 2016; Ba et al., 2016a; Kiros et al., 2015) in an unsupervised manner, using the extensive amount of text that exists in book and web corpuses. Examples include skip-thoughts vectors (Kiros et al., 2015) and FastSent (Hill et al., 2016), both of which are highly effective for generation tasks. However, for instructional text such as cooking recipes, such representations tend to do poorly, and suffer from coherence from one time step to the next, since they do not fully capture the underlying sequential nature of the instruction set. As such, we propose a hierarchical model with four components, where two dedicated RNNs represent the sentences and the steps of the recipe: the sentence encoder and the recipe RNN respectively. A third RNN decodes predicted recipe steps back into sentence form for human-interpretable results (sentence decoder). These three RNNs are learned jointly as an auto-encoder in an initial training step. A fourth RNN encoding visual evidence (video encoder) is then learned in a subsequent step to replace the sentence encoder to enable interpretation and future prediction from video data. An overview is shown in Figure 4.3, while details of the RNNs are given in Sections 4.3.1 to 4.3.3.

4.3.1 Sentence Encoder and Decoder

The sentence encoder produces a fixed-length vector representation of each textual recipe step. We use a bi-directional LSTM, but instead of representing a sentence by the last hidden vector, we follow the highly effective approach of Conneau et al. (2017) and apply a max-pooling over each dimension of the hidden units. This type of architecture and, in particular, the max-pooling step has been shown to perform very well on sentence encoding

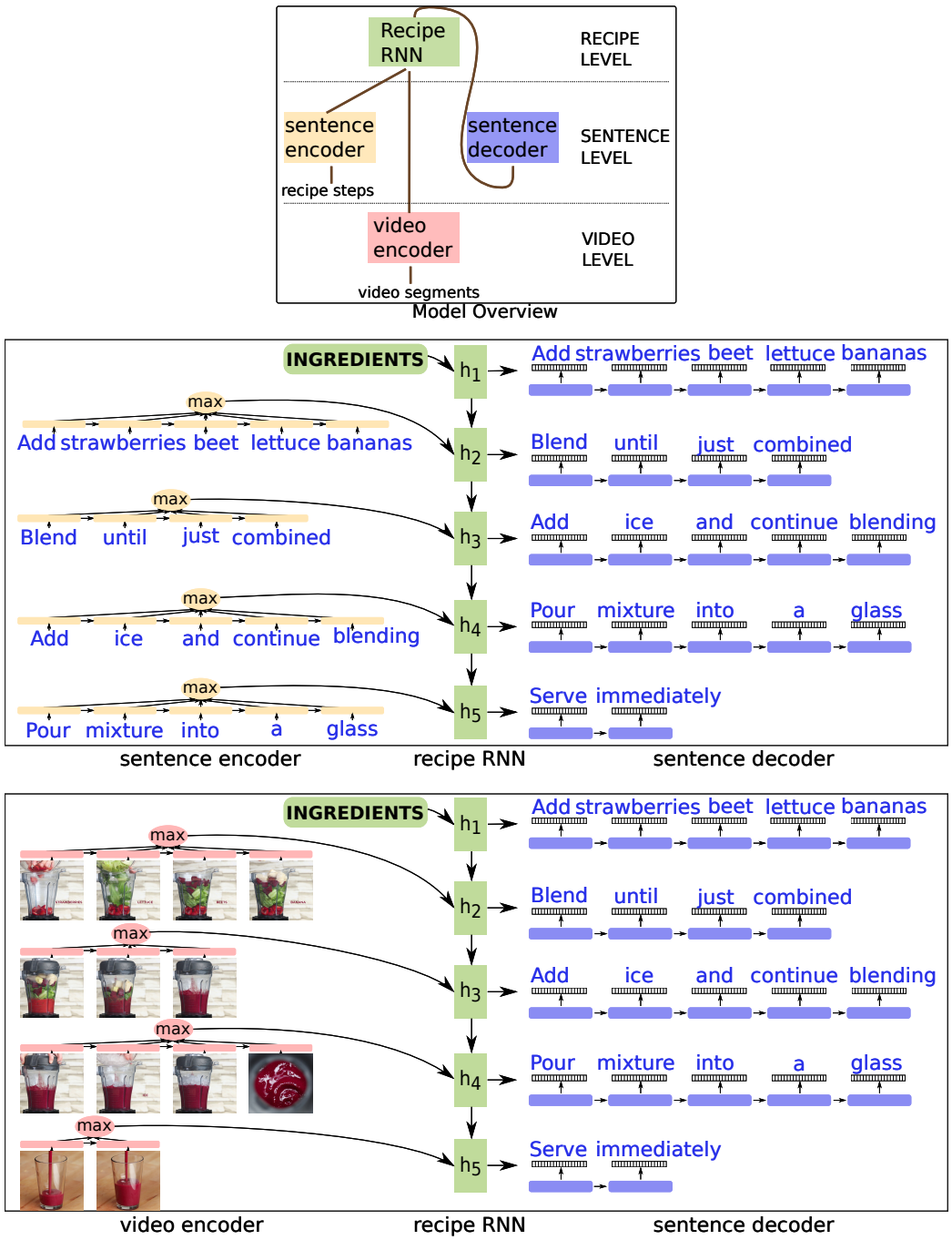


Figure 4.3: Our system is composed of four RNNs: a sentence encoder and a decoder, a video encoder, and a recipe RNN. Given the ingredients as initial input and context in either text or visual form, the recipe RNN recurrent predicts future steps. The sentence decoder converts predicted future steps back into natural language. We continue predicting future steps by repeatedly feeding the next steps encoded by the sentence or video encoder. We first learn the sentence encoder, decoder, and recipe RNN jointly as an auto-encoder on textual data. Then we replace the sentence encoder with the video encoder and jointly train video encoder, sentence decoder, and recipe RNN on video data.

tasks (Conneau et al., 2017). More formally, let sentence s_j from step j of a recipe (we assume each step is one sentence) be represented by M words, i.e., $s_j = \{w_j^t\}_{t=1...M}$ and $\boldsymbol{x}_j^t$ be the word embedding of word w_j^t. For each sentence j, at each (word) step t, the bi-directional LSTM_{se} outputs $\boldsymbol{y}_j^t$, where

$$\boldsymbol{y}_j^t = \left[\text{LSTM}_{\text{se}}\left(\{\boldsymbol{x}_j^1, ..., \boldsymbol{x}_j^t\}\right), \text{LSTM}_{\text{se}}\left(\{\boldsymbol{x}_j^M, ..., \boldsymbol{x}_j^t\}\right)\right], \tag{4.1}$$

which is a concatenation of the hidden states from the forwards and backwards pass of LSTM_{se}. The overall sentence representation $\boldsymbol{r}_j$ is determined by a dimension-independent max-pooling over the time steps, i.e.,

$$(\boldsymbol{r}_j)_d = \max_{t \in \{1,...,M\}} (\boldsymbol{y}_j^t)_d, \tag{4.2}$$

where $(\cdot)_d$, $d \in \{1, ..., D\}$, indicates the d-th element of the D-dimensional bi-directional LSTM outputs $\boldsymbol{y}_j^t$.

The sentence decoder is an LSTM-based neural language model that converts the fixed-length representation of the steps back into human-interpretable sentences. More specifically, given the vector prediction $\hat{\boldsymbol{r}}_j$ from the recipe RNN of step j, it decodes the sentence $\hat{s}_j$

$$\hat{s}_j = \text{LSTM}_{\text{d}}(\hat{\boldsymbol{r}}_j) = \{\hat{w}_j^1, ..., \hat{w}_j^{\hat{M}}\}. \tag{4.3}$$

4.3.2 Recipe RNN

We model the sequential ordering of recipe steps with an LSTM, which takes as input $\{\boldsymbol{r}_j\}_{j=1,...,N}$, i.e., fixed-length representations of the steps of a recipe with N steps, where j indicates the step index. At each (recipe) step, the hidden state of the recipe RNN $\boldsymbol{h}_j$ can be considered a fixed-length representation of all recipe steps $\{s_1, ..., s_j\}$ seen up to step j; we directly use this hidden state vector as a prediction of the sentence representation for step $j+1$, i.e.,

$$\hat{\boldsymbol{r}}_{j+1} = \boldsymbol{h}_j = \text{LSTM}_{\text{r}}(\{\boldsymbol{r}_0, ..., \boldsymbol{r}_j\}). \tag{4.4}$$

The hidden state of the last step $\boldsymbol{h}_N$ can be considered as a representation of the entire recipe. Due to the standard recursion of the hidden states in LSTM_{r}, each hidden state vector and, therefore, each future step prediction is conditioned on the previous steps. This allows predicting recipe steps that are plausible and coherent with respect to previous steps.

Recipes usually include an ingredient list, a rich source of information that can also serve as a strong modelling cue (Kiddon et al., 2016; Salvador et al., 2017, 2019). To incorporate the ingredients, we form an ingredient vector $\boldsymbol{I}$ for each recipe in the form of a one-hot encoding over a vocabulary of ingredients. $\boldsymbol{I}$ is then transformed with a separate fully connected layer in the recipe RNN to serve as the initial input, i.e., $\boldsymbol{r}_0 = f(\boldsymbol{I})$.

4.3.3 Video Encoder

For inference, we would like the recipe RNN to interpret sentences from text inputs and also visual evidence. We can conveniently replace the sentence encoder with an analogous video encoder due to the modular nature of our proposed model. Suppose the j^{th} video segment c_j is composed of L frames, i.e., $c_j = \{\boldsymbol{f}_j^t\}_{t=1,...,L}$. Each frame f_j^t is represented as a high-level CNN feature vector – we use the last fully connected layer output of ResNet-50 (He et al., 2016) before the softmax layer. Similar to the sentence encoding $\boldsymbol{r}_j$ in Eqs. 4.1 and 4.2, we determine the video encoding vector, $\boldsymbol{v}_j$, by applying a dimension-independent max pooling over the time steps of the segment representations, $\boldsymbol{z}_j^t$, where :

$$\boldsymbol{z}_j^t = \left[\text{LSTM}_{\text{ve}}\left(\{\boldsymbol{f}_j^1, ..., \boldsymbol{f}_j^t\}\right), \text{LSTM}_{\text{ve}}\left(\{\boldsymbol{f}_j^L, ..., \boldsymbol{f}_j^t\}\right)\right], \tag{4.5}$$

$$(\boldsymbol{v}_j)_d = \max_{t \in \{1,...,L\}} (\boldsymbol{z}_j^t)_d, \tag{4.6}$$

The video encoder, LSTM_{ve}, is trained such that $\boldsymbol{v}_j$ can directly replace $\boldsymbol{r}_j$, as detailed in the following.

4.3.4 Model Learning and Inference

Our full model is learned in two stages. First, the sentence encoder (LSTM_{se}), recipe RNN (LSTM_{r}) and sentence decoder (LSTM_{d}) are jointly trained end-to-end. Given a recipe of N steps, a loss can be defined as the negative log probability of each reconstructed word:

$$L(s_1, ..., s_N) = -\sum_{j=1}^{N} \sum_{t=1}^{M_j} \log P(w_j^t | w_j^{t'<t}, \hat{\boldsymbol{r}}_j), \tag{4.7}$$

where $P(w_j^t | w_j^{t'<t}, \hat{\boldsymbol{r}}_j)$ is parameterised by a softmax function at the output layer of the sentence decoder to estimate the distribution over the words, w, in our vocabulary V. The overall objective is then summed over all recipes in the corpus. The loss is computed only when the LSTM is learning to decode a sentence. This first training stage is unsupervised, as the sentence encoder and decoder and the recipe RNN require only text inputs that can easily be scraped from the web without human annotations.

In a second step, we train the video encoder (LSTM_{ve}) while keeping the recipe RNN (LSTM_{r}) and sentence decoder (LSTM_{d}) fixed. We simply replace the sentence encoder with the video encoder while applying the same loss function as defined in Eq. 4.7. This step is supervised, as it requires video segments of each step temporally aligned with the corresponding sentences.

During inference, we provide the ingredient vector $\boldsymbol{r}_0$ as an initial input to the recipe RNN, which then outputs the predicted vector $\hat{\boldsymbol{r}}_1$ for the first step of the video (see Figure 4.3). We use the sentence decoder and generate the corresponding first sentence, $\hat{s}_1$. Then, we sample a sequence of frames from the video and apply the video encoder to generate $\boldsymbol{v}_1$, which we again provide as an input to the recipe RNN. The output prediction of the recipe RNN, $\hat{\boldsymbol{r}}_2$,

is for the second step of the video. We again use the sentence decoder and generate the corresponding sentence $\hat{s}_2$.

Our model is not limited to one step ahead predictions: for further predictions, we can simply apply the predicted output $\hat{\boldsymbol{r}}_j$ as contextual input $\boldsymbol{r}_j$. During training, instead of always feeding in the ground truth $\boldsymbol{r}_j$, we sometimes (with 0.5 probability after the 5th epoch) use our predictions, $\hat{\boldsymbol{r}}_j$, as the input for the next step predictions that helps us with being robust to feeding in bad predictions (Bengio et al., 2015).

4.3.5 Implementation and Training Details

We use a vocabulary V of 30171 words provided by the Recipe1M dataset (Salvador et al., 2017); words are represented by 256-dimensional embeddings shared between the sentence encoder and decoder. Our ingredients vocabulary has 3769 ingredients selected from the training set of Recipe1M; the one-hot ingredient encodings are mapped into a 1024 dimensional vector $\boldsymbol{r}_0$. The RNNs are all single-layer LSTMs implemented in PyTorch; LSTM_{se}, LSTM_{ve}, LSTM_{d} have 512 hidden units while LSTM_{r} has 1024. We train our model using the Adam optimizer (Kingma and Ba, 2014) with a batch size of 50 recipes and a learning rate of 0.001. We train our text-based model (LSTM_{se}, LSTM_{r}, LSTM_{de}) for 50 epochs and the visual model (LSTM_{ve}, LSTM_{r}, LSTM_{de}) for 25 epochs.

4.4 Tasty Videos Dataset

Our new *Tasty Videos Dataset* has 2511 unique recipes collected from the Buzzfeed website `https://tasty.co`. Our dataset is publicly available `https://cvml.comp.nus.edu.sg/tasty`. Each recipe has an ingredient list, step-wise instructions, and a video demonstrating the preparation of the dish. The recipes are drawn from three major culinary locations (Americas, Europe, and Asia), spanning a variety of meals (breakfast, lunch, dinner, snacks), desserts, and drinks from 185 categories such as smoothies, pies, soups. We define a split ratio of 8:1:1 for training, validation, and testing, each containing different recipes. Our test setting is, therefore, zero-shot, as we make predictions on unseen recipes. We further divide the test set into recipes with similarities in the training set (183 videos), e.g., *"Strawberry Breakfast Muffins"* vs. *"Carrot Cake Muffins"* and *"Pizza Muffins"* and those without any similarities (72 videos) e.g., *"Pigs In A Blanket"*.

The Tasty Videos are captured with a fixed overhead camera and focus entirely on the preparation of the dish (see Figure 4.3, 4.8). This viewpoint removes the added challenge of distractors and irrelevant actions. While we are aware that it may not exactly reflect the visual environments one may find in the home, this simplification allows us to focus the scope of our work on modelling the sequential nature of instructional data, which is already a highly challenging and open research topic. Videos are designed to be primarily visually informative without any narrations. Unlike other datasets with text data that are crowd-sourced (Zhou et al., 2018b; Damen et al., 2018), the recipes in our dataset are written by experts, which ensures specificity and richness in the instruction.

The videos are well-suited for learning complex instructional activities because they are short (on average, 1551 frames / 54 seconds), yet they contain a challenging number of steps (9 on average). For each recipe step, we annotated the temporal boundaries in which the step occurs within the video, omitting those without visual correspondences, such as alternative recommendations, non-visualized instructions such as *"Preheat oven."* and stylistic statements such as *"Enjoy!"*.

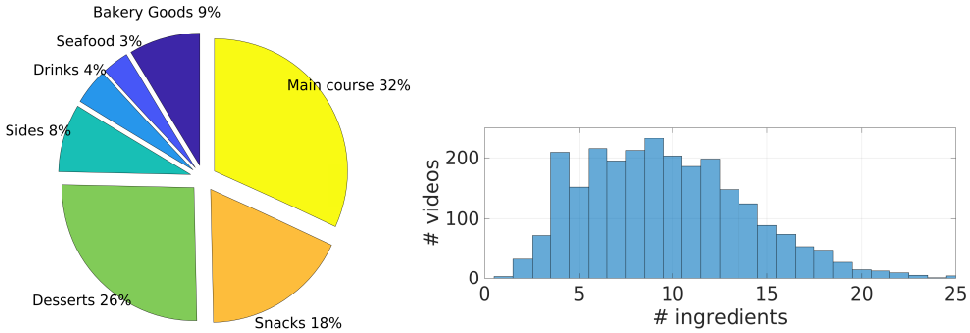


Figure 4.4: (left) Rough categorization of the recipes in our dataset. (right) Distribution of the number of ingredients over all our recipes (out of 1199 unique ingredients).

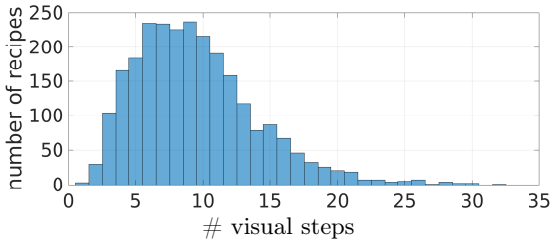


Figure 4.5: Distribution of the number of visual steps in our dataset. The x-axes indicate the number of visual steps. There are nine visual steps on average.

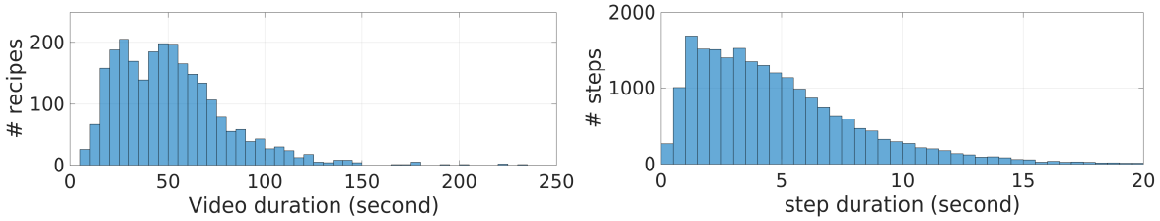


Figure 4.6: Distribution of video durations (left) and annotated visual step durations (right).

Our dataset has a large variety of meals, including main courses, snacks, sides etc., (see Figure 4.4, left). In our dataset, there are 1199 unique ingredients, and the average number of ingredients per recipe is nine, (see Figure 4.4, right). In comparison, the number of unique ingredients in the Recipe1M dataset is 3769 (2.5K vs. 1M recipes).

In Figure 4.5, we show the distribution of the number of visual recipe steps over all recipes in our dataset. The average number of visual recipe steps is 9, and there are 21236 visual recipe steps in total. In Figure 4.6 (left), we report the distribution of the duration of our videos. Our shortest video lasts 6 seconds while the longest lasts 233 seconds. The average video duration is 1551 frames (54 seconds). In Figure 4.6 (right), we report the distribution of the duration of the visual steps over all recipes. The average visual step duration is 144 frames (5 seconds).

4.5 Experiments

4.5.1 Datasets and Evaluation Measures

We train and evaluate our method with Recipe1M (Salvador et al., 2017), YouCookII (Zhou et al., 2018b), and our own Tasty Videos datasets. Recipe1M is a large-scale dataset with, as the name suggests, approximately one million recipes with a recipe name, list of ingredients, a sequence of instructions, and images of the final dish for each recipe. YouCookII is a collection of cooking videos from YouTube with around 2000 videos of 89 dishes. Each video is captured from a third-person viewpoint. Each dish has, on average, 22 videos annotated with the temporal boundaries of each step and their corresponding descriptions. The average number of steps per video is 7.70.

We use the ingredients and instructions from the training split of the Recipe1M dataset to learn our sentence encoder, sentence decoder, and recipe RNN. To learn the video encoder, we use the aligned instructions and video data from the training split of either YouCookII or Tasty Videos datasets. We evaluate our model's prediction capabilities with text inputs from Recipe1M and video and text inputs from YouCookII and Tasty Videos.

Our predictions are in sentence form; evaluating the quality of generated sentences is known to be difficult in captioning and natural language processing (Vedantam et al., 2015; Lopez, 2008). We apply a variety of evaluation measures in order to offer a broad assessment. First, we target the matching of important keywords, specifically, ingredients and verbs, since they indicate the next active objects and actions and are analogous to the assessments made in action anticipation (Damen et al., 2018). Second, we evaluate with sentence matching scores BLEU (BiLingual Evaluation Understudy) (Papineni et al., 2002) and METEOR (Metric for Evaluation of Translation with Explicit ORdering) (Banerjee and Lavie, 2005) scores which are also used for video captioning methods (Regneri et al., 2013; Rohrbach et al., 2013; Zhou et al., 2018c). BLEU computes an n-gram based precision for predicted sentences w.r.t. the ground truth sentences. METEOR creates an alignment between the ground truth and predicted sentence using the exact word matches, stems, synonyms, and paraphrases; then, it computes a weighted F-score with an alignment fragmentation penalty.

For the uninformed reader, we note that sentence scores are best at indicating sentences with precise word matches to the ground truth (GT). In natural spoken language, much variation may exist between sentences conveying the same ideas. Automated scores might fail in matching sentences a human would consider equivalent. This is the case even in text with very specific language such as cooking recipes. For example, for the ground truth

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	bacon, brown sugar, cooking spray, breadsticks						
step1	Preheat oven to 325 degrees F (165 degrees C).	Preheat oven to 400 degrees F.	36.0	0.0	26.0	1.5	1.5
step2	Line 2 baking sheets with aluminum foil or parchment paper and spray with cooking spray.	Line a baking sheet with aluminum foil.	23.0	0.0	23.0	1.0	1.0
step3	Wrap 1 bacon strip around each breadstick, leaving about 1 inch uncovered on each end.	Place bacon strips in a single layer on the prepared baking sheet.	13.0	0.0	9.0	0.5	1.5
step4	Place wrapped breadsticks on the prepared baking sheet.	Place rolls on a baking sheet.	48.0	0.0	30.0	1.5	1.5
step5	Sprinkle brown sugar evenly over breadsticks.	Bake in the preheated oven until breadsticks are golden brown, about 15 minutes.	15.0	0.0	13.0	0.0	1.5
step6	Bake in the preheated oven until bacon is crisp and browned, 50 to 60 minutes.	Bake in preheated oven until bacon is crisp and breadsticks are golden brown, about 15 minutes.	63.0	43.0	36.0	1.0	1.0
step7	Cool breadsticks on a piece of parchment paper or waxed paper sprayed with cooking spray.	Remove from oven and let cool for 5 minutes.	6.0	0.0	4.0	0.5	1.5

Figure 4.7: Predictions of our text-based method for *"Candied Bacon Sticks"* along with the automated scores and human ratings. For *"HUMAN1 (HUM1)"* we ask the raters to directly assess how well the predicted steps match the corresponding Ground Truth (GT) sentences, for *"HUMAN2 (HUM2)"* we ask to judge if the predicted step is still a plausible future prediction (see Sec. 4.5.9). Our prediction for step6 matches the GT well, while step5 does not. However, according to *"HUMAN2 (HUM2)"* score, our step5 prediction is still a plausible future action.

sentence *"Garnish with the remaining wasabi and sliced green onions."*, our method predicts *"Transfer to a serving bowl and garnish with reserved scallions."*. For a human reader, this is half correct, especially since *"scallions"* and *"green onions"* are synonyms, yet this example would have only a BLEU1 score of 30.0, BLEU4 of 0.0 and METEOR of 11.00. For another, for the ground truth sentence *"Place patties on the grill, and cook for 5 minutes per side."* versus a prediction by our model *"Place on the grill, and cook for about 10 minutes, turning once."*, we would have a BLEU1 score of 65.0, BLEU4 of 44.0, and METEOR of 29.0.

As such, human evaluations are still the gold standard in dialogue generations (Liu et al., 2016) and captioning (Li et al., 2018a). We, therefore, conduct a user study and ask people to assess how well the predicted step matches the ground truth in meaning; if it does not match, we ask if the prediction would be plausible for future steps. This gives flexibility in case predictions do not follow the exact aligned order of the ground truth, e.g., due to missing steps not predicted, or steps that are slightly out of order (see Figures 4.7 and 4.8).

4.5.2 Learning of Procedural Knowledge

We first verify the learning of procedural knowledge with a text-only model, i.e., the sentence encoder, sentence decoder, and the recipe RNN by evaluating on Recipe1M's test set of 51K

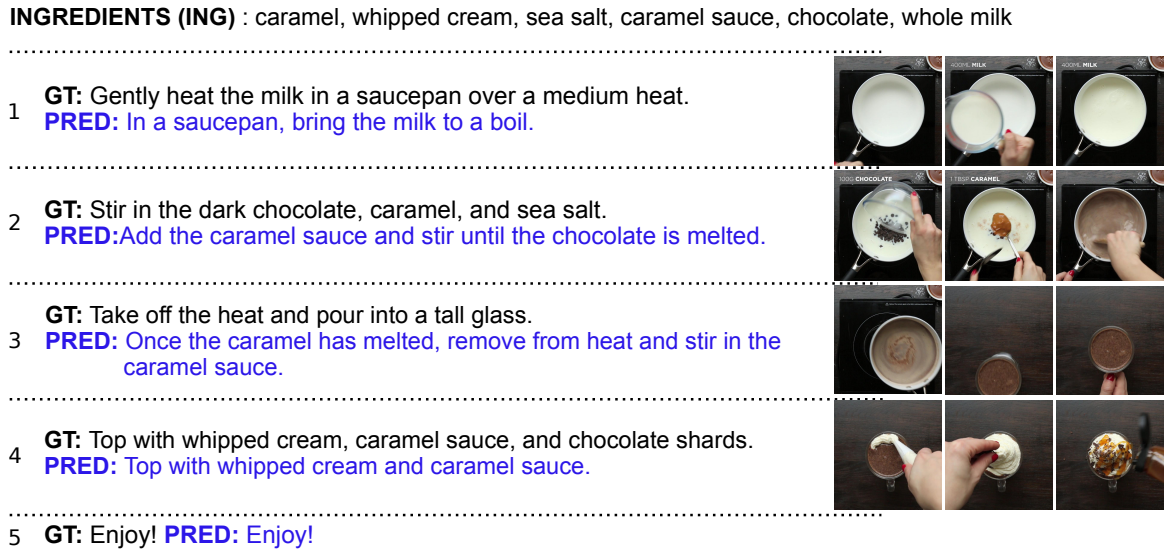


Figure 4.8: Next step predictions from our visual model for *"Salted Caramel Hot Chocolate"*. The blue sentences are our model's predictions. Note that our model predicts the next steps before seeing these segments!

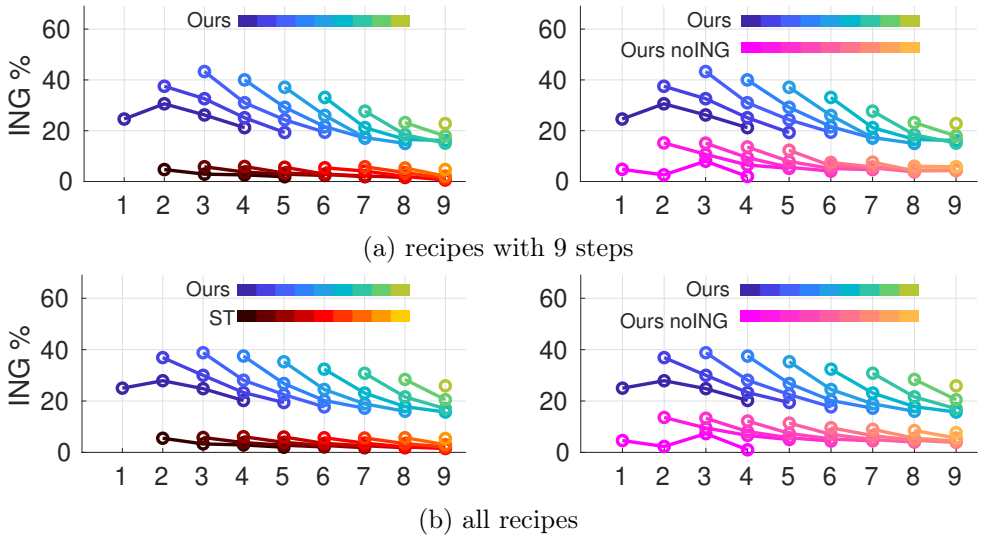
recipes. For a recipe of N steps, we evaluate our model's ability to predict steps $j+1$ to N, conditioning on steps 1 to j as input context. N varies from recipe to recipe, so we separately tally recipes with N=9 steps (4.3K recipes; 9 is also the average number of recipe steps in the test set) and recipes in the entire test set.

For comparison, we look at the generations from a common sequence-to-sequence model, specifically a skip-thoughts (ST) model (Kiros et al., 2015). Skip-thoughts models are trained to decode temporally adjacent sentences from a current encoding, i.e., given step j to the encoder, the decoder predicts steps $j+1$ and $j-1$, and have been shown to be successful in generating continuous text (Cho et al., 2014; Vinyals and Le, 2015; Kiddon et al., 2016). We train the skip-thoughts model on the training set of the Recipe1M dataset. Because the skip-thoughts model generations are not trained to accept an ingredient list as a 0^{th} or initialization step, we make skip-thoughts predictions only from the second step onwards.

We report our results over the recipes with 9 steps and the entire test set of the Recipe1M dataset in Figures 4.9 (ingredients), 4.11 (verbs), 4.12 (sentences). We report scores of the predicted steps averaged over multiple recipes. Only those recipes which have at least j steps contribute to the average for step j. Compared to the recipes with exactly 9 steps, results over the entire test set are not significantly different in trends.

4.5.2.1 Key Ingredients

We first look at our model's ability to predict important keywords, i.e., ingredients and verbs on Recipe1M. We compare the recall of ingredients in our predictions to a skip-thoughts (ST) model and a variation of our model trained without ingredients, (*"Ours noING"*). We train our model without any ingredients and compute the recall of the ingredients with this variant.



(a) recipes with 9 steps

(b) all recipes

Figure 4.9: The recall of ingredients predicted by our model (*"Ours"*), by our model trained without the ingredients (*"Ours noING"*) and the by the skip-thoughts model (*"ST"*) over the recipes with 9 steps (a) and entire test set (b) of the Recipe1M dataset. The x-axes in the plots indicate the step number being predicted in the recipe; each curve begins on the first prediction, i.e., the $(j+1)$th step after having received steps 1 to j as input.

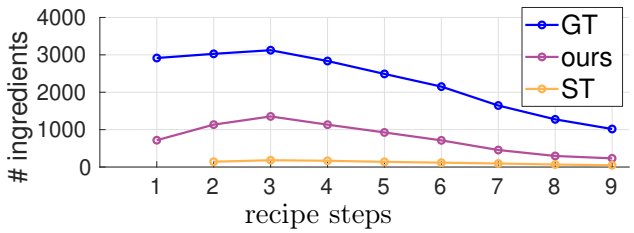


Figure 4.10: The absolute number of ingredients detected in the ground truth steps (*"GT"*), steps predicted by our model (*"ours"*), and the skip-thoughts model (*"ST"*) computed over recipes with exactly 9 steps. The number of ingredients detected in a recipe decreases towards the end of the recipe.

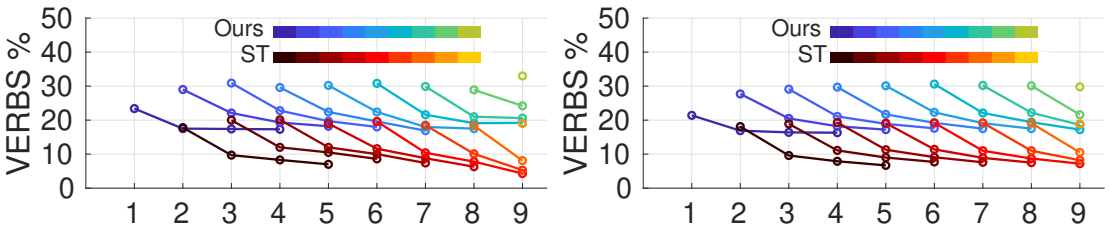
Rather than directly cross-referencing the ingredient list, we limit the evaluation to ingredients mentioned explicitly in the recipe steps. This is necessary to avoid ambiguities that may arise from specific instructions such as *"add chicken, onion, and bell pepper"* versus the more vague *"add remaining ingredients"*. Furthermore, the ingredient lists in Recipe1M are often automatically generated and may be incomplete.

In Figure 4.9, we compare the recall of the ingredients detected in our predicted steps versus steps generated by the skip-thoughts method and our model trained without ingredient inputs. We present our comparisons on the recipes with 9 steps and all recipes. On the recipes with 9 steps, we can see that our model's predictions successfully incorporate relevant ingredients with recall rates as high as 43.3% with the predicted next step, 31.0% with second,

24.3% with third and 19.4% with the predicted fourth step. The overall recall decreases with the latter steps, likely due to increased difficulty once the overall number of ingredient occurrences decreases, which tends to happen in later steps. Based on the ground truth, we observe that the majority of the ingredients occurs in the early and middle steps and decreases in the last steps, see Figure 4.10. The last steps are usually related to the already completed dish and do not explicitly mention as many ingredients as the earlier steps. Compared to the recipes with 9 steps, the scores over the recipes in the entire test set are not significantly different in trends.

Compared to the skip-thoughts model, our predictions' ingredient recall is higher regardless of whether or not ingredients are provided as an initial input. Without ingredient input, the overall recall is lower, but after the initial step, our model's recall increases sharply, i.e., once it receives some context. Our model without the ingredient input still performs better than the skip-thoughts predictions. We attribute this to the strength of our model to generalize across related recipes so that it is able to predict relevant co-occurring ingredients. Our predictions include common ingredients such as *salt, butter, eggs* and *water* and also recipe-specific ones such as *couscous, zucchini*, or *chocolate chips*. While the skip-thoughts model predicts some common ingredients, such as *water* and *butter*, it fails to predict recipe-specific ingredients.

4.5.2.2 Key Verbs



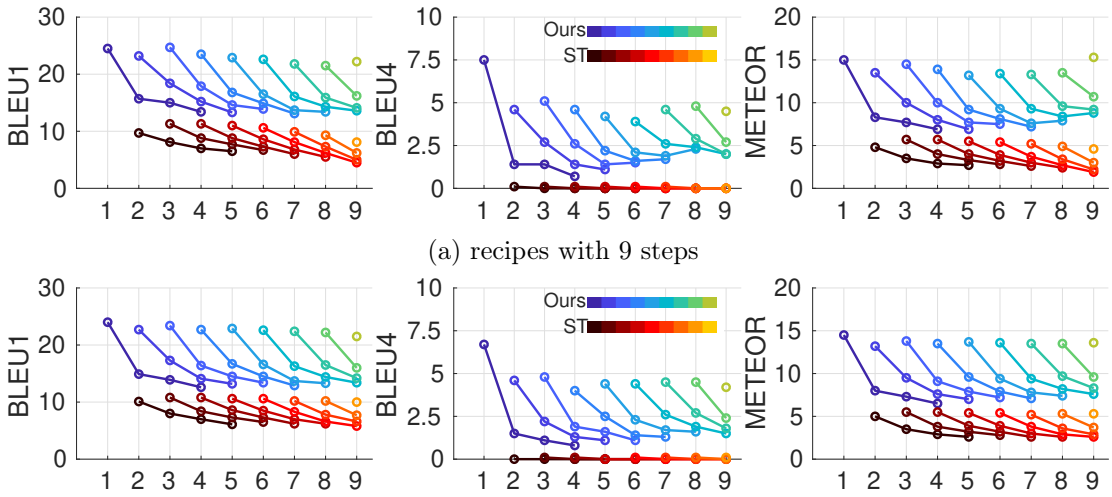
(a) recipes with 9 steps (b) all recipes

Figure 4.11: The recall of verbs computed between the predicted and the ground truth sentences. We compare the recall of our model (*"Ours"*) and the skip-thoughts (*"ST"*) model over the recipes with 9 steps (a) and the entire test set (b) of the Recipe1M dataset. The x-axes in the plots indicate the step number being predicted in the recipe; each curve begins on the first prediction, i.e., the $(j+1)$th step after having received steps 1 to j as input.

Key verbs indicate the main action for a step and are also cues for future steps both immediate (e.g., after *"adding"* ingredients into a bowl, a common next step may be to *"mix"*) and long-term (e.g., after *'preheating'* the oven, one expects to *"bake"*). We tag the verbs in the training recipes with the Natural Language Toolkit (NLTK, 2018) and select the 250 most frequent verbs for evaluation. Similar to ingredients, we check for recall of these verbs only if they appear in the ground truth steps. In the ground truth steps, there are between 1.55 and 1.85 verbs per step, i.e., steps often include multiple verbs such as *"add and mix"*.

Figure 4.11 shows that our model recalls up to 30.9% of the verbs with the predicted next step on the recipes with 9 steps (a). Our model's performance is worst in the first steps, due to ambiguities when given only the ingredients without any further knowledge of the recipe. After the first steps, our model's performance quickly increases and stays consistent across the remaining steps. In comparison, the ST model's best recall is only 20.1% for the next step prediction. Compared to the recipes with 9 steps, the scores over the entire test set are similar in trends with a slight decrease.

4.5.2.3 Sentences



(a) recipes with 9 steps

(b) all recipes

Figure 4.12: The sentence scores, BLEU1, BLEU4, and METEOR, are computed between the predicted and the ground truth sentences for our model (*"Ours"*) and the skip-thoughts (*"ST"*) model over the recipes with 9 steps (a) and entire test set (b) of the Recipe1M dataset. The x-axes in the plots indicate the step number being predicted in the recipe; each curve begins on the first (relative) prediction, i.e., the $(j+1)$th step after having received steps 1 to j as input.

Key ingredients and verbs alone do not capture the rich instructional nature of recipe steps, compare e.g., *"whisk"* and *"egg"* to *"Whisk the eggs till light and fluffy"*. As such, we also evaluate the quality of the entire predicted sentences.

We evaluate our predicted sentences as a whole and compare to skip-thoughts predictions based on BLEU1, BLEU4, and METEOR in Figure 4.12. For our results on recipes with 9 steps, the BLEU1 scores are consistently high, at around 25.0 for the next step predictions, with a slight decrease towards the end of the recipes. Predictions further than the next step have lower scores, though they stay above 15.0. The BLEU4 scores are highest in the very first step, around 7.5, and range between 1.0 and 5.0 over the remaining steps. The high performance, in the beginning, is because many of the recipes start with common instructions such as *"Preheat oven to X degrees"* or *"In a large skillet, heat the oil"*. For similar reasons,

we also tend to do well towards the end of recipes, where instructions with respect to serving and garnishing are common, e.g., *"Season with salt and pepper."*. Trends for the METEOR score are similar to our BLEU1 scores. METEOR scores are between 15.0 for the next step predictions and do not go lower than 7.5 for the further step predictions.

Our proposed method outperforms skip-thoughts predictions across the board. In fact, predictions up to four steps into the future surpass the predictions made by skip-thoughts predictions only one step ahead. This can be attributed to the dedicated long-term modelling of the recipe RNN; as such, we are able to incorporate the context from all sentence inputs up to the present. In contrast, skip-thoughts are Markovian in nature and can only take the current step into account.

One interesting and unexpected outcome of our model is that it can also make recommendations. In cooking recipes, one does not only find strict recipe steps but also suggestions based on the writer's experience, e.g., *"If using wooden skewers, make sure to soak in water for about 30 minutes."*. Our learned model is also able to generate such suggestions. For example, for the ground truth *"If it's too loose at this point, place it in the freezer for a little while to let freeze."*, our model predicts *"If you freeze it, it will be easier to eat"*.

Our model is able to predict coherent and plausible instructional sentences, as shown in Figure 4.7. We show more examples of predictions of our text-based method in Figures 4.20, 4.21, 4.23, 4.24, 4.22, 4.25, 4.26, along with the automated scores and human ratings.

4.5.3 Ablation Studies on Textual Model

Since our method is modular, we conduct an ablation study to check the interchangeability of the sentence encoder on the Recipe1M dataset. Instead of using our own sentence encoder, we represent the sentences using skip-thoughts vectors trained on the Recipe1M dataset, as provided by Salvador et al. (2017). These vectors have been shown to perform well for their recipe retrieval task. For this experiment, we train the recipe RNN and sentence decoder jointly using the pre-trained skip-thoughts vectors as sentence representations. Note that in all experiments for the ablations, the recipe RNN and sentence decoder have been trained with the same parameter settings.

Figure 4.13 compares sentence scores of our joint model (*"Ours"*), our joint model trained without ingredient inputs (*"Ours noING"*) and our model when the sentence encoder is replaced with pre-trained skip-thoughts vectors (*"ST vectors"*) when *"X%"*, $X = \{0, 25, 50, 75\}$, of a recipe is observed. We can see that our sentence encoder performs on par with skip-thoughts encodings. Moreover, our encoder and decoder can all be trained jointly and do not require a separate pre-training of a sentence auto-encoder.

Similar to our observations in Figure 4.9 for ingredient scores, we see that ingredient information is very important for predicting sentences, especially for the initial steps. In subsequent steps, when 25%, 50% of the recipe steps (enough context) are observed, the model's performance starts to improve.

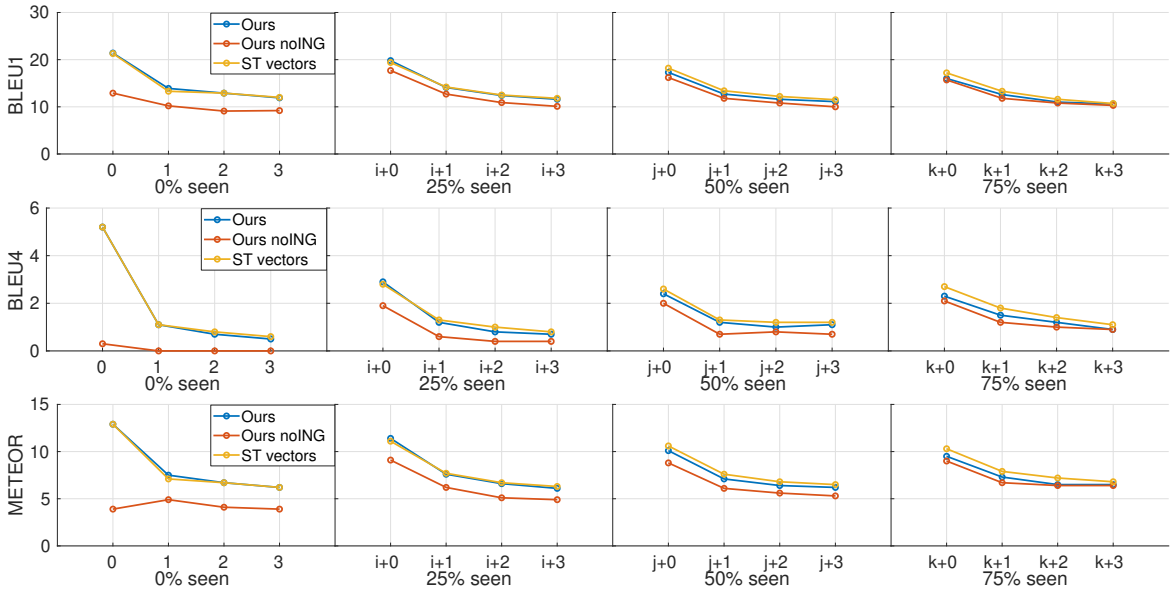


Figure 4.13: Ablation study to check the interchangeability of the sentence encoder and show the influence of ingredient inputs for sentence generation. We present our results evaluated on the entire test set of Recipe1M. We compare the sentence scores of our joint model (*"Ours"*), our joint model without ingredient inputs (*"Ours noING"*), and our model when sentence encoder is replaced with pre-trained skip-thoughts vectors (*"ST vectors"*). *"X% seen"* refers to the number of steps the model receives as input, while predicting the remaining $(100-X)\%$.

4.5.4 Recipe Visualization using tSNE

Our method can model recipes, as the output of the recipe RNN, especially after seeing all N steps, serves as a feature vector representing the entire recipe. For validating these representations, we conduct a recipe visualization experiment. We select recipes from the nine most common recipe categories in the test set of the Recipe1M dataset and encode them with our model by taking the final hidden output of the recipe RNN. For comparison, similar to Salvador et al. (2017), we compute the mean of the skip-thoughts vectors across the recipe steps. We visualize a two-dimensional representation computed using tSNE (Maaten and Hinton, 2008) of both recipe representations in Figure 4.14. We find that with our method, the recipes are better separated according to category.

4.5.5 Video Predictions

We evaluate our model for making predictions on video inputs on YouCookII's validation set and Tasty Videos' test set. We test two video segmentation settings for inference: one according to ground truth (*"Ours Visual (GT)"*) and one based on fixed windows (*"Ours Visual (window)"*). In both settings, we sample every fifth frame in GT or window-based segments and feed the visual features of these frames into the video encoder. The representations from the video encoder are then fed to recipe RNN as context vectors. Through the video encoder,

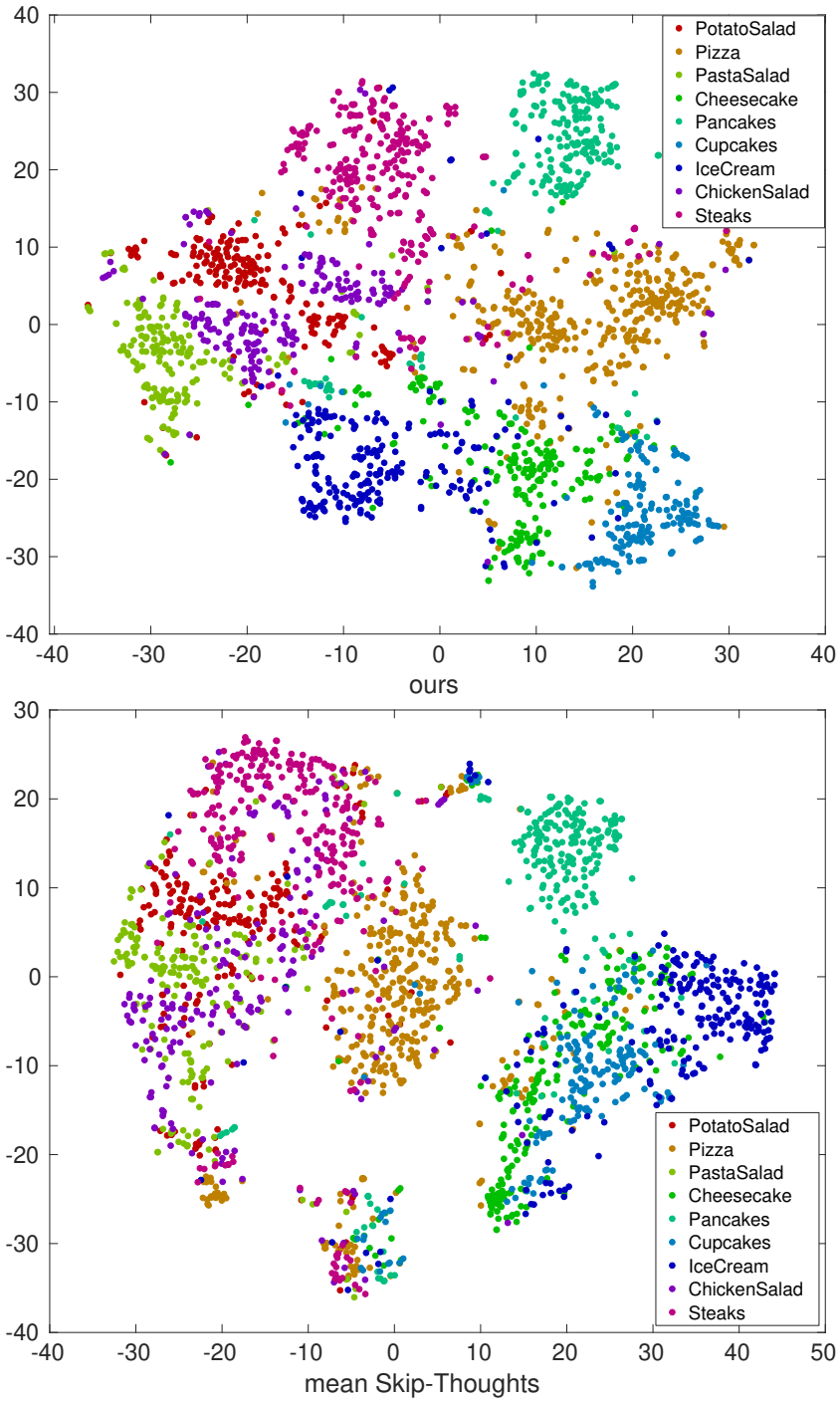


Figure 4.14: Recipe encoding visualization with tSNE (Maaten and Hinton, 2008) over a subset of recipes from the nine most common categories in the Recipe1M test set. Our representations successfully create clusters of recipes from the same categories. The mean of the skip-thoughts vectors fails in separating recipes from similar categories such as *"IceCream"* and *"Cupcakes"*.

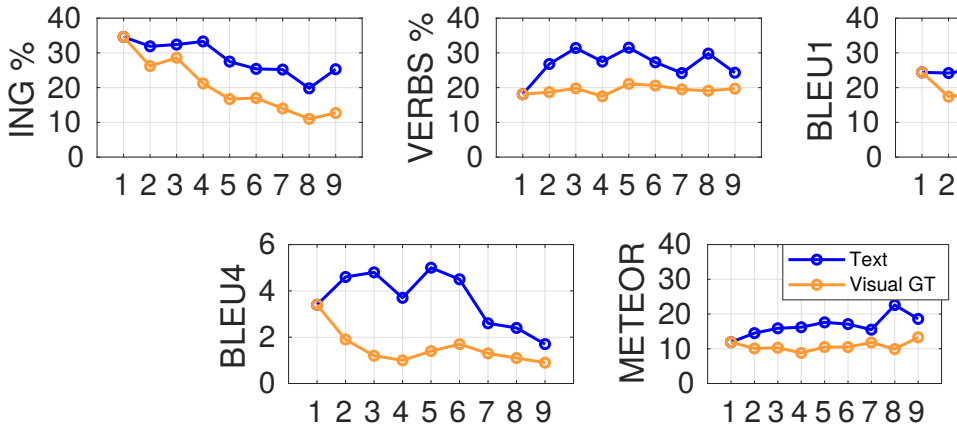


Figure 4.15: We compare our visual model when tested with GT segments, *"Visual GT"*, and our textual model, *"Text"*, on the Tasty Videos dataset for next step predictions using the recall of predicted ingredients and verbs, and sentence scores. Compared to our text-based model, our visual model has lower performance, but follow similar trends. The x-axes in the plots indicate the step number being predicted in the recipe.

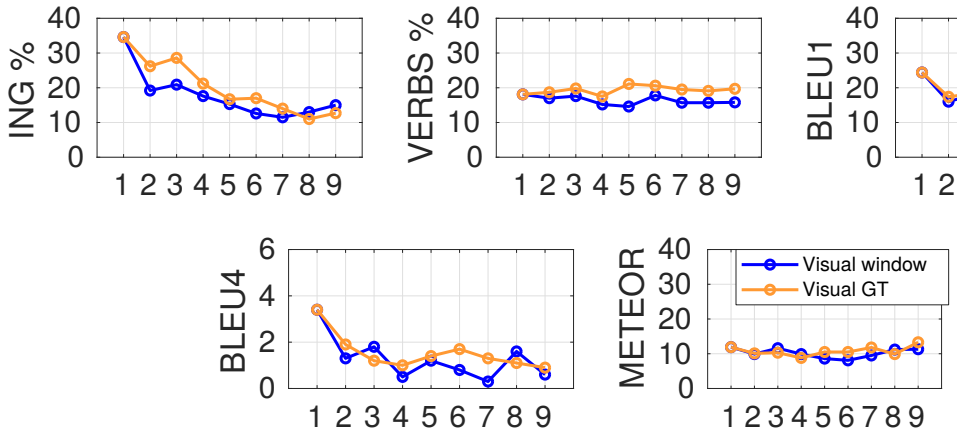


Figure 4.16: We compare the performance of our visual models when tested with GT segments, *"Visual GT"*, and fixed window-sized segments, *"Visual window"*, using the recall of predicted ingredients and verbs, and sentence scores. Compared to using GT segments, using fixed window-sized segments show lower results. The x-axes in the plots indicate the step number being predicted in the recipe.

our model can interpret visual evidence, and make plausible predictions of the next steps that can be seen in examples in Figures 4.8, 4.29, 4.28, 4.27, 4.30). In Figure 4.18, we show an example of our visual model, which corrects itself after observing new evidence.

We present our results for our visual model in Table 4.1 for Tasty Videos dataset and in Table 4.2 for YouCookII dataset. Compared to using ground truth segments, (*"Ours Visual (GT)"*), using fixed window segments, (*"Ours Visual (window)"*), result in a decrease in performance, with the most extreme drop on the most challenging sentence score, BLEU4

Method	ING	VERBS	BLEU1	BLEU4	METEOR
S2VT (Venugopalan et al., 2015) (GT)	7.59	19.18	18.03	1.10	9.12
S2VT (Venugopalan et al., 2015), next (GT)	1.54	10.66	9.14	0.26	5.59
End-to-end (Zhou et al., 2018c)	-	-	-	0.54	5.48
Ours Visual (GT)	20.40	19.18	19.05	1.48	11.78
Ours Visual (window)	16.66	17.08	17.59	1.23	11.00
Ours Text	26.09	27.19	26.78	3.30	17.97
Ours Text noING	9.04	22.00	20.11	0.92	13.07
Ours joint video-text	22.27	23.35	21.75	2.33	14.09

Table 4.1: Evaluations on the Tasty Videos dataset for our visual and text-based model along with comparisons against video captioning methods (Venugopalan et al., 2015; Zhou et al., 2018c). Performance drops when fixed-sized windows based segments (*"Ours Visual (window)"*) are used compared to using GT segments (*"Ours Visual (GT)"*). Ingredient inputs are important for our model's success. Our method performs better than video captioning.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
End-to-end (GT) (Zhou et al., 2018c)	-	-	-	0.87	8.15
TempoAttn (GT) (Yao et al., 2015)	-	-	-	1.42	11.20
Ours Visual (GT)	21.36	27.55	23.71	1.66	11.54
End-to-end (Zhou et al., 2018c)	-	-	-	0.08	4.62
TempoAttn (Yao et al., 2015)	-	-	-	0.30	6.58
Ours Visual (window)	17.64	25.11	22.55	1.38	10.71
Ours Text	24.60	29.39	26.49	2.66	13.31

Table 4.2: Evaluation for our visual and text-based models and comparison against two video captioning methods (Yao et al., 2015; Zhou et al., 2018c) on YouCookII's validation set. We perform better than the state-of-the-art captioning method (Zhou et al., 2018c).

(around 17% for both datasets), and ingredient scores (around 18% for both datasets). For the verb, BLEU1, and METEOR scores, the decrease is not as big (lower than 10%). We compare the prediction performance of our visual model with the GT segments vs. fixed-window based segment in Figure 4.16 for different recipe steps. Overall, using fixed windows show lower results for different steps, especially for ingredient and BLEU4 scores, but the results follow similar trends across recipe steps.

In Tables 4.1 and 4.2, we present our text-based results as upper-bound for both datasets. Given that our model is first trained on text and then transferred to video, the drop in performance from text to video is as expected, though video results still follow similar trends. This can be seen in Figure 4.15, where we provide step-wise comparisons of our textual and visual models. Comparing the performance of our visual with the textual model, the textual results are better overall on the Tasty Videos than YouCookII.

We further investigate the influence of the ingredients on the performance of our method on the Tasty videos dataset in Table 4.1. When ingredients are not provided, our method fails to make plausible predictions. The performance decrease is mainly noticeable in the ingredient scores and the BLEU4 scores.

In some instructional scenarios, there may be semi-aligned text that accompanies the video, e.g., narrations. We test such a setting by training the sentence and video encoder, as well as sentence decoder and recipe RNN jointly, for making future step predictions. We concatenate the sentence and video context vectors and then pass them through a linear layer before feeding them as input to the Recipe RNN. We observe that the results are better than our video alone results but not better than our text alone results (see *"Ours joint video-text"* in Table 4.1). Even with joint training, it is still challenging to make improvements, which we attribute to the diversity in our videos and variations in the text descriptions for similar visual inputs. On the other hand, when there is accompanying text, our model can be adapted easily and improve prediction performance.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
Ours Visual (window 70)	12.40	13.26	14.73	0.93	8.31
Ours Visual (window 90)	13.15	13.99	15.97	1.06	9.24
Ours Visual (window 110)	14.06	15.58	16.84	1.18	10.05
Ours Visual (window 130)	14.97	15.40	17.94	1.09	10.32
Ours Visual (window 150)	16.70	17.59	18.94	1.07	11.25
Ours Visual (window 170)	16.66	17.08	17.59	1.23	11.00
Ours Visual (window 190)	17.14	17.38	17.18	1.09	11.60
Ours Visual (window 210)	15.99	15.90	17.43	1.19	10.85
Ours Visual (window 230)	15.40	15.48	16.19	1.06	10.31

Table 4.3: Window size selection for our visual model on the Tasty Videos dataset.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
Ours Visual (window 30)	15.18	20.38	19.99	0.60	9.21
Ours Visual (window 50)	15.86	22.60	21.29	1.10	10.02
Ours Visual (window 70)	17.64	25.11	22.55	1.38	10.71
Ours Visual (window 90)	18.13	26.31	22.87	1.32	10.93
Ours Visual (window 110)	18.86	26.91	22.93	1.32	10.83
Ours Visual (window 130)	18.21	26.05	22.51	1.28	10.83
Ours Visual (window 210)	18.05	26.40	21.83	1.20	9.83

Table 4.4: Window size selection for our visual model on the YouCookII dataset.

Fixed-sized Window Selection: For *"Ours Visual (window)"* experiments, we first partition the video until the latest observation into chunks of fixed-sized windows. We sequentially feed the representations of these chunks, which we obtain from the video encoder, into our recipe RNN. Overall, our method is relatively robust to window size. We report our results for different window sizes for Tasty in Table 4.3 and for YouCookII in Table 4.4. We select a window of 70 frames for YouCookII and 170 for Tasty Videos.

4.5.6 Supervised vs. Zero-Shot Learning

We compare the differences between supervised and zero-shot learning on the YouCookII dataset. The provided splits for this dataset are not zero-shot. Therefore we create new

Method	ING	VERBS	BLEU1	BLEU4	METEOR
Supervised Visual (GT)	20.93	24.76	22.11	1.21	10.66
Supervised Visual	18.90	23.15	21.09	1.03	10.22
Supervised Visual no pre-train	2.69	19.43	15.05	0.15	5.89
Supervised Text	24.56	27.24	24.94	1.99	12.50
Zero-shot Visual (GT)	17.77	23.11	20.61	0.84	9.51
Zero-shot Visual	6.04	23.19	20.30	0.76	9.27
Zero-shot Visual no pre-train	1.58	17.83	14.54	0.01	5.03
Zero-shot Text	19.90	24.86	23.06	1.47	10.98

Table 4.5: Comparison of our zero-shot and supervised setting on the YouCookII dataset computed using 4-fold cross-validation. Supervised results are better overall. Without pre-training, the performance drop is significant.

splits over the training and validation sets of YouCookII. We divide the dataset into four splits based on the 89 dishes, 22 dishes per split. We create our splits based on distinct dishes. The first split includes all videos from dish labels between 1 and 125, second from 126 and 222, third from 223 and 316, and fourth split from 317 and 425. We use three splits for training and half of the videos in the fourth split for testing. In the zero-shot setting, the videos from the other half of the fourth split are unused, while in the supervised setting, they are included as part of the training. We report results averaged over the four cross-folds in Table 4.5.

As expected, the predictions are better when the model is trained under a supervised setting in comparison to a zero-shot setting. This is true for all inputs, with the same drop as observed previously when moving from text to video inputs and when moving from ground truth video segments to fixed window segments. However, the difference between the supervised versus zero-shot setting (see Table 4.5 *"Supervised Visual"* vs. *"Zero Visual"*) is surprisingly much smaller than the difference between a supervised setting with and without pre-training on Recipe1M (*"Supervised Visual"* vs. *"Supervised Visual no pre-train"*). This suggests that having a large corpus for pre-training is more useful than repeated observations for a specific dish.

The YouCookII dataset has 22 videos per dish on average. We use 11 for testing; for supervised-training, we use the other 11 for training, but exclude them for a comparable zero-shot scenario. In Figure 4.17, we show a detailed experiment where we start with the zero-shot scenario (no videos about the evaluated dish in the training set), and we continue with one-shot learning (only one similar video) and so on until the fully supervised case (on average 11 videos from the same dish in the training set). One can see that the more videos are added, the more the performance increases. This indicates that the model is, in fact, learning and that more than 11 videos (current supervised setting) will further improve the supervised performance.

While the test set of Tasty Videos is fully zero-shot, 183 videos are of recipes that occur with some variations in the training set, while 72 are without any variations. As expected, when comparing the predictions on these subsets separately (see Table 4.6), we observe higher

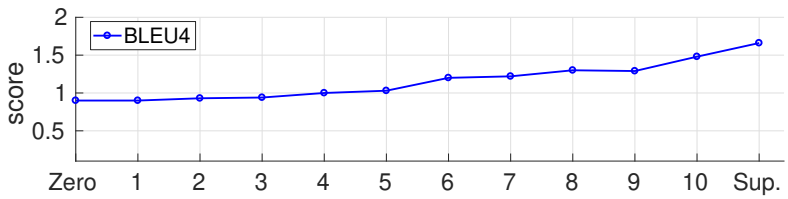


Figure 4.17: Zero-shot (*"Zero"*) vs. Supervised (*"Sup."*) comparison when the number of training videos from the same dish is increased. When more videos from the same dish are added into the training set, the BLEU4 score increases.

Method	ING	VERBS	BLEU1	BLEU4	METEOR
without variations	14.20	17.08	16.67	0.76	10.00
with variations	25.40	20.41	20.54	2.16	13.00
all test videos	22.24	19.47	19.45	1.77	12.15

Table 4.6: Evaluations on the test set of the Tasty Videos with and without variations in the training set. Out of 255 test videos, 183 are variations of training videos, while 72 are unrelated. We do better on variations.

performance on videos with variations, especially for the challenging BLEU4 score. For the videos with no variations, the decrease from the ingredient recall and BLEU4 scores are 45% and 65%, respectively. This suggests that our method generalizes better when it receives visually similar recipes.

4.5.7 Knowledge Transfer

Method	ING	VERBS	BLEU1	BLEU4	METEOR
Ours Text (100%)	26.09	27.19	26.78	3.30	17.97
Ours Text (50%)	23.01	24.90	25.05	2.42	16.98
Ours Text (25%)	19.43	23.83	23.54	2.03	16.05
Ours Text (0%)	5.80	9.42	10.58	0.24	6.80

Table 4.7: Evaluations on the Tasty Videos dataset for the textual model when the number of training recipes varies. Performance drops when the amount of pre-training decreases.

At the core of our method is the transfer of knowledge from text resources to solve a challenging visual problem. We evaluate the effectiveness of the knowledge transfer by varying the amount of training data from Recipe1M to be used for pre-training. We train our method with 100%, 50%, and 25% of the Recipe1M training set. We present our results in Table 4.7. Looking at the averaged scores over all the predicted steps on the Tasty Videos dataset, we observe a decrease in all evaluation measures as we limit the amount of data from Recipe1M (see *"Ours Text"* 100%, 50%, 25% and 0%), with the most significant decrease occurring for the BLEU4 score. When using less text data, our method's performance decreases from 3.30 of BLEU4 score to 2.42 when half of the Recipe1M is used, and to 2.03 when a quarter of the dataset is used. While we observe a similar decrease in the ingredient detection scores,

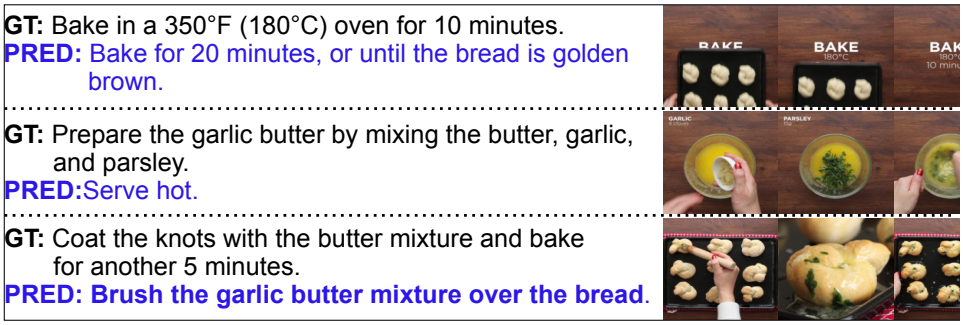


Figure 4.18: Next step prediction of our visual model. Blue sentences are our model's predictions. After baking, our model predicts that the dish should be served, but after visually observing the butter parsley mixture, it predicts that the knots should be brushed. Note that our model predicts the next steps before seeing these segments.

the decrease in BLEU1, METEOR, and verb scores remains less significant. If there is no pre-training, i.e., when the model is learned only on text from Tasty Videos (*"Ours Text (0%)"*), the decrease in scores, is noticeable for all evaluation criteria. These results verify that pre-training has a significant effect on our method's performance.

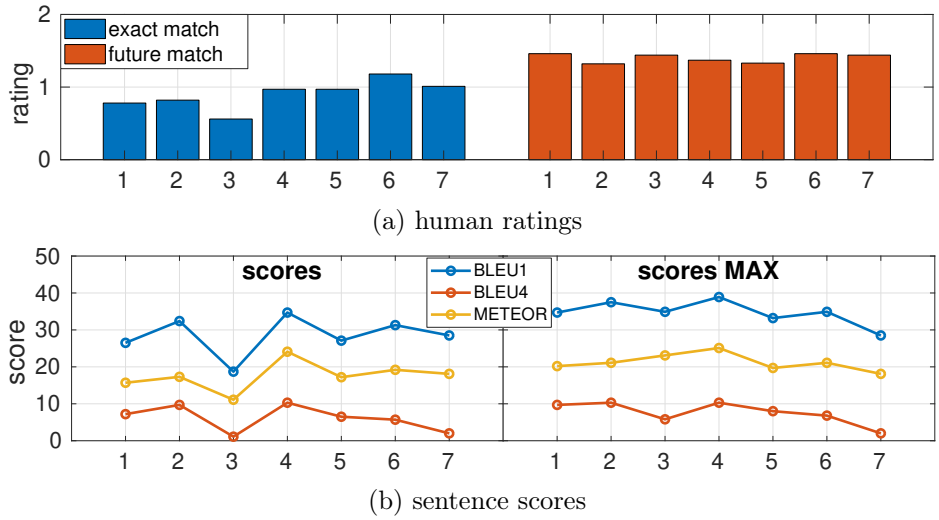
4.5.8 Comparisons to Video Captioning

We show that knowledge transfer considerably improves our method's predictions. To validate our claims further, we compare our method against different video captioning methods in Tables 4.1 and 4.2 for the Tasty Videos and YouCookII datasets respectively. Unlike predicting future steps, captioning methods generate sentences after observing their visual data, which makes it a much easier task than predicting the future. We train and test S2VT (Venugopalan et al., 2015), an RNN based encoder-decoder approach, on the ground truth segments of the Tasty Videos dataset. Our visual model outperforms this baseline, especially for ingredient recall, by 13%, and with an improvement of 0.3 in BLEU4 score in Table 4.1. To highlight the difficulty of predicting future steps compared to captioning, we train S2VT for predicting the next step from the observation of the current step (see Table 4.1 *"S2VT Venugopalan et al. (2015) next (GT)"*). Our visual model outperforms this variation with a big margin for all scores. We also test a state-of-the-art video captioning method, End-to-end Masked Transformer (Zhou et al., 2018c), on the Tasty Videos dataset and get a BLEU4 and METEOR score of 0.54 and 5.48, respectively (vs. our future prediction scores 1.23 / 11.00). The poor performance is likely due to the increased dish diversity and difficulty of our dataset compared to YouCookII.

We compare our model on the validation set of the YouCookII dataset against two captioning methods (Yao et al., 2015; Zhou et al., 2018c) in Table 4.2. End-to-end Masked Transformer (Zhou et al., 2018c) performs dense video captioning by both localizing steps and generating descriptions for these steps, while TempoAttn (Yao et al., 2015) is an RNN-based encoder-decoder approach with attention. TempoAttn is tested on YouCookII by Zhou et al. (2018c) after several changes were made to the model, including adding a Bi-LSTM con-

text encoder and adding temporal attention. Again, even though our task is more difficult than captioning, our method outperforms both of the captioning methods in BLEU4 and METEOR scores. Compared to the state-of-the-art video captioning method, End-to-end Masked Transformer (Zhou et al., 2018c), our visual model achieves a METEOR score twice as high, and a BLEU4 score four times higher. We attribute the better performance of our method compared to the captioning methods to the pre-training on the Recipe1M dataset, which allows our model to generalize. Note that for YouCookII, as we use all the videos in the training set, our training is no longer a zero-shot but a supervised scenario.

4.5.9 Human Ratings



(a) human ratings

(b) sentence scores

Figure 4.19: We conduct a user study and ask human raters to asses how well the predicted sentences matches the ground truth sentences. We present the comparison of human ratings (a) versus automated sentence scores (b) over a subset of 30 recipes.

As automated scores such as BLEU and METEOR are not fully representative of the correctness of the predicted steps, we also ask humans to evaluate our model's predictions. We ask human raters to directly assess how well the predicted steps match the ground truth with scores 0 (*"not at all"*), 1 (*"somewhat"*), or 2 (*"very well"*). If the prediction receives a score of 0, we additionally ask the human to judge if the predicted step is still a plausible future prediction, again with the same scores of 0 (*"not at all"*), 1 (*"somewhat"*), or 2 (*"very likely"*). We conduct this study with three people on a subset of 30 recipes from the test set of the Recipe1M dataset, each with seven steps, and present their ratings in Figure 4.19 while comparing them to automated sentence scores.

In Figure 4.19, the upper graph (a) shows the results of the human raters. In this plot, *"exact match"* corresponds to humans assessing if the predicted steps match the ground truth, Raters report a score close to 1 for the initial step predictions indicating that our method, even by only seeing the ingredients, can start predicting plausible steps. Scores increase

towards the end of the recipe and are lowest at step 3. *"future match"* corresponds to humans assessing if that step a plausible future prediction. The average score of the predicted steps being a possible future prediction is consistently high across all steps. Even if the predicted step does not exactly match the ground truth, human raters still consider it possible for the future, including the previously low rating for step 3. Overall, the ratings indicate that the predicted steps are plausible.

The lower graph (b) in Figure 4.19 shows automated scores for the same set of recipes used in our user study. The left plot shows the standard scores for the predicted sentences matching the ground truth. Overall, trends are very similar to the user study, including the low-scoring step 3. To match the second setting of the user study, we compute the sentence scores between the predicted sentence $\hat{s}_j$, and the next four future ground truth steps $\{s_j, s_{j+1}, s_{j+2}, s_{j+3}\}$ and select the step with the maximum score as our future match. These scores are plotted in the lower right plot of Figure 4.19. Similar to the second setting in the human study, the sentence scores increase overall.

To assess the reliability of agreement between our human raters, we use Fleiss's kappa (Fleiss et al., 1971) measure. It is used to analyze how much the annotators agree in their decisions. A high level of agreement (at most 1) indicates that the human rating study was reliable. Our, inter-rater agreement, measured via Fleiss's kappa by aggregating across all rating tasks, is 0.43, which is statistically significant at $p < 0.05$.

4.6 Conclusion and Future Works

In this chapter, we present a method for zero-shot action anticipation in videos. Our model learns to generalize instructional knowledge from the text-domain. Applying this knowledge to videos allows us to tackle the challenging task of predicting steps of complex tasks from visual data, which is otherwise ruled out because of the scarcity of or difficulty in annotating training data. We present a new, diverse dataset of cooking videos that is of high interest to the community. We successfully validate our method's performance on both text and video data. We show that our model can produce coherent and plausible future steps. We conclude that our knowledge transfer strategy works much better than captioning methods and generalizes well on different datasets.

We extended our Tasty dataset to 4000 videos, all annotated with the temporal boundaries of the textual steps. In our current work, we use sliding windows to split the videos into fixed segments. We are planning to incorporate proposal generation and segmentation methods for partitioning the past observations in the videos. For richer representations of observations, we also want to replace our current video encoder with our temporal aggregates model presented in Chapter 5. Our framework could be further improved by replacing our current RNN-based encoders and decoder with state-of-the-art sentence architectures such as transformers (Devlin et al., 2019). To improve our evaluations, we also plan to use the newly presented, BERT-based metric, BLEURT (Sellam et al., 2020).

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	salt, pepper, onion, egg, bacon, ketchup, ground beef, Worcestershire sauce, parmesan cheese, hamburger buns, shredded cheddar cheese						
step1	Preheat a grill for high heat.	In a large bowl, combine the first six ingredients.	11.0	0.0	3.0	0.0	1.5
step2	In a large bowl, mix together the Cheddar cheese, Parmesan cheese, onion, egg, ketchup, Worcestershire sauce, salt and pepper.	In a large bowl, mix together the ground beef, onion, Worcestershire sauce, egg, salt	66.0	38.0	33.0	1.0	1.0
step3	Crumble in the ground beef, and mix together by hand.	Shape into 4 patties.	0.0	0.0	0.0	0.0	1.0
step4	Form into 6 patties, and wrap two slices of bacon around each one in both directions.	Form into 4 patties.	4.0	0.0	10.0	0.5	1.5
step5	Secure bacon with toothpicks.	Place on the grill, and cook for about 10 minutes, or until the bacon is cooked through	6.0	0.0	9.0	0.0	1.0
step6	Place patties on the grill, and cook for 5 minutes per side, or until well done.	Place on the grill, and cook for about 10 minutes, turning once.	48.0	32.0	22.0	1.5	1.5
step7	Remove toothpicks before serving on hamburger buns.	Serve on buns with lettuce, tomato, and ketchup.	25.0	0.0	14.0	0.5	1.5

Figure 4.20: Predictions of our text-based method for *"Cheddar Bacon Wrapped Hamburgers"* along with the automated scores and human ratings. step4 prediction is half correct. step7 is a plausible prediction.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	milk, carrots, poultry seasoning, fresh ground black pepper, chicken bouillon cubes, celery ribs, chicken breasts						
step1	Place chicken in a slow cooker.	Place chicken in a large pot and cover with water.	40.0	26.0	29.0	0.5	1.0
step2	Heat broth in microwave; dissolve bouillon in broth.	Add celery, carrots, and celery.	0.0	0.0	0.0	0.0	2.0
step3	Add next 4 ingredients to broth.	Pour over chicken.	0.0	0.0	0.0	0.0	1.5
step4	Pour over chicken.	Pour over chicken.	100	3.0	100	2.0	2.0
step5	Cover and cook on low for 6-8 hours, until chicken falls apart when poked with a fork.	Cover and cook on low for 8 hours.	28.0	23.0	26.0	1.5	1.5
step6	Combine buttermilk biscuit mix and milk, then drop spoonfuls over chicken to form dumplings.	Remove chicken from broth, cool and shred.	11.0	0.0	5.0	0.0	1.0
step7	Cover and cook on high for 35 minutes or until dumplings are done.	Cover and cook on low for 8 to 10 hours or until chicken is tender.	47.0	20.0	26.0	0.5	0.5

Figure 4.21: Predictions of our text-based method for *"Slow Cooker Chicken and Dumplings"* along with the automated scores and human ratings. step2 prediction is a plausible future step. step5 is correct.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	sugar, celery, black olives, chopped tomatoes, green olives						
step1	Boil your noodles as directed on box.	Mix all ingredients together.	0.0	0.0	0.0	0.0	0.5
step2	Do not overcook!	Drain and rinse with cold water.	0.0	0.0	0.0	0.0	2.0
step3	Drain.	Drain and rinse with cold water.	17.0	0.0	26.0	1.5	1.5
step4	Add veggies to pasta in large bowl.	In a large bowl, mix together the noodles, celery, tomatoes, olives, and sugar.	23.0	0.0	16.0	1.5	1.5
step5	Add Italian dressing and Splenda or sugar and ground pepper.	Add all other ingredients.	6.0	0.0	6.0	1.5	1.5
step6	Mix well, chill and enjoy!	Mix well.	22.0	0.0	24.0	1.5	1.5
step7	Add more dressing the next day as needed, if put in the fridge overnight!	I like to add a little bit of olive oil to the salad and I add a little more	16.0	0.0	6.0	0.5	1.0

Figure 4.22: Predictions of our text-based method for *"My Mom's Summer Pasta Salad"* along with the automated scores and human ratings. step2 is a plausible future step. step4 is correct as veggies refer to celery tomatoes and olives. step7 is a plausible suggestion even though it does not match the GT step.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	pineapple, strawberries, oranges, flaked coconut, fresh mint leaves, vanilla yogurt, kiwi fruits						
step1	Quarter pineapple lengthwise; remove core.	In a large bowl, combine the strawberries, kiwi, oranges, and pineapple.	9.0	0.0	7.0	0.0	1.0
step2	Cut crosswise into small chunks.	Cut into 1/2-inch cubes.	39.0	0.0	12.0	1.0	1.0
step3	Place in large serving bowl.	In a large bowl, combine fruit, kiwi, and pineapple.	33.0	0.0	20.0	0.0	1.0
step4	Add clementine segments, strawberries, kiwifruit, and coconut; gently toss.	Add fruit and nuts.	14.0	0.0	7.0	0.5	0.5
step5	Spoon into dessert glasses.	Serve immediately or store in refrigerator up to 3 days.	0.0	0.0	0.0	0.0	0.5
step6	Top with a dollop of vanilla yogurt or sweetened sour cream.	Garnish with orange slices and mint.	7.0	0.0	2.0	0.0	1.0
step7	Garnish with mint sprigs if desired.	Garnish with orange slices and mint.	50.0	0.0	22.0	1.0	1.0

Figure 4.23: Predictions of our text-based method for *"Ambrosia Fruit Salad"* along with the automated scores and human ratings. step1, step3, and step6 are plausible future step predictions.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	butter, cheese, eggs, salt and pepper						
step1	Whisk the eggs till light and fluffy.	Preheat oven to 350 degrees.	0.0	0.0	0.0	0.0	0.0
step2	Add cheese, salt and pepper.	Add the cheese and season with salt and pepper.	56.0	0.0	43.0	2.0	2.0
step3	Mix well.	Melt butter in a pan.	0.0	0.0	0.0	0.0	1.5
step4	Heat a 7 inch non-stick skillet and add butter.	Melt butter in a pan.	18.0	0.0	11.0	1.0	1.0
step5	As the butter begins to smoke, lower heat and pour the beaten egg mixture.	Pour in egg mixture.	6.0	0.0	14.0	1.0	1.0
step6	Cook covered for a minute.	Let cook until eggs are set.	17.0	0.0	10.0	1.5	1.5
step7	Fold over and serve immediately.	Turn the heat down to low and cook for about 5 minutes.	8.0	0.0	3.0	0.0	1.0

Figure 4.24: Predictions of our text-based method for *"Cheese Omelette"* along with the automated scores and human ratings. step5 is a correct prediction. step7 is a plausible prediction.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	salt and pepper, parmesan cheese, garlic cloves, extra virgin olive oil, diced tomatoes, fusilli, pesto sauce, fresh basil leaves						
step1	Heat oil in a medium skillet over medium heat.	Heat the oil in a large skillet over medium heat.	80.0	39.0	43.0	2.0	2.0
step2	Saute garlic for 2 minutes, then add tomatoes and simmer for 10 minutes.	Add garlic and cook until fragrant, about 30 seconds.	21.0	0.0	9.0	0.5	1.5
step3	Meanwhile, bring a large pot of lightly salted water to a boil.	Add pesto and cook for 1 minute.	0.0	0.0	4.0	0.0	1.5
step4	Add pasta and cook for 8 to 10 minutes or until al dente; drain and return to pot.	Add pasta and cook for 8 to 10 minutes or until al dente; drain.	75.0	75.0	54.0	2.0	2.0
step5	Season tomato sauce with salt and pepper; stir in pesto.	Add pesto and basil to pasta and toss to combine.	20.0	0.0	7.0	0.5	1.5
step6	Pour sauce over hot cooked pasta and mix well.	Pour sauce over pasta and toss to coat.	55.0	0.0	25.0	1.0	1.0
step7	Sprinkle with cheese, garnish with basil leaves and serve immediately.	Serve immediately.	2.0	0.0	14.0	1.0	1.0

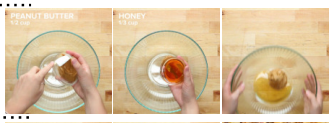
Figure 4.25: Predictions of our text-based method for *"Easy Fusilli with Tomato Pesto Sauce"* along with the automated scores and human ratings. step3 is a plausible future prediction. step5 is half correct.

	Ground Truth (GT)	*Prediction*	*BLEU1*	*BLEU4*	*MET.*	*HUM1*	*HUM2*
ING	salt, butter, cheese, cream cheese, scallions, cauliflower, black pepper, bacon bits, whole milk						
step1	Preheat oven to 350	Preheat oven to 350 degrees F.	67.0	51.0	50.0	2.0	2.0
step2	Chop cauliflower and boil in salted water for about 3-4 mins until a little tender	Place cauliflower in a large pot and cover with water.	30.0	0.0	10.0	1.0	1.0
step3	Strain cauliflower and mash with a potato masher.	Add the butter, salt, and pepper, and stir to combine.	10.0	0.0	2.0	0.0	1.5
step4	Place in bowl and mix all ingredients	Melt butter in a large skillet over medium heat.	11.0	0.0	3.0	0.0	1.0
step5	Add ingredients to a buttered 8x8 dish and top with extra cheese and bacon.	Pour mixture into a baking dish and bake for 20 minutes	21.0	0.0	7.0	0.5	1.5
step6	Bake for 30 mins until nice and bubbly.	Bake for 30 minutes.	27.0	0.0	28.0	2.0	2.0
step7	Enjoy!	Serve with a side of buttered toast.	0.0	0.0	0.0	0.5	1.5

Figure 4.26: Predictions of our text-based method for *"Baked Loaded Cauliflower"* along with the automated scores and human ratings. step5 is half correct.

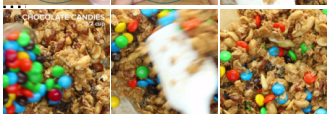
ING: chocolate, honey, almonds, raisins, candy, peanuts, rolled oats, peanut butter

1 **GT:** Melt peanut butter and honey together in the microwave, stirring every 15 seconds.
PRED: In a large bowl, combine the oats chocolate chips, **honey**, and **peanut butter**

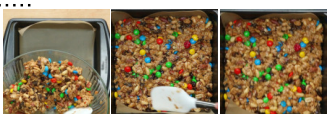


2 **GT:** Add **oats**, almonds, **peanuts**, and raisins into the bowl and mix until well coated.
PRED: Stir in **oats** and **peanuts**.

3 **GT:** Add chocolate candies and mix just until combined.
PRED: Press into a greased 9x9-inch pan.



4 **GT:** Line a baking dish with parchment paper and pour the cooled mixture into the pan.
PRED: Using a small, sharp knife, cut into squares.



5 **GT:** Cool in the refrigerator for at least 2 hours.
PRED: Enjoy !

6 Enjoy! Enjoy!

Figure 4.27: Next step prediction of our visual model for *"Trail Mix Bars"*. The blue sentences are our model's predictions. Our model's prediction for step3 is a plausible future step prediction as the mixture is placed in a pan in step4. Note that our model predicts the next steps before seeing these segments!

ING: salt, pepper, onion, milk, egg, meatballs, ground beef, Worcestershire sauce, parmesan cheese, canola oil, beef broth, fresh parsley, seasoned bread crumbs

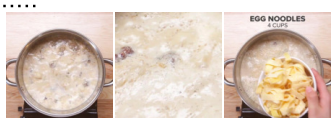
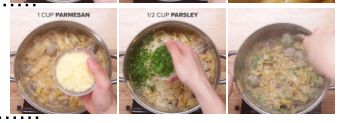
1	**GT:** In a large bowl, **combine** the **ground beef**, **breadcrumbs**, **onion**, **egg**, salt, and pepper, mixing until evenly combined. **PRED:** In a large bowl, **combine** the ground **beef**, **bread crumbs**, milk, **egg**, parsley, **onion**	
2	**GT:** Heat the canola oil in a pot over medium-high heat. **PRED:** Pour mixture into a loaf pan and bake for about 1 hour.	
3	**GT:** Place the **meatballs** into the pot, **cooking** for one minute. **PRED:** Add the **meatballs** to the pan and **cook** for about 5 minutes on each side.	
4	**GT:** Flip the meatballs. **PRED:** In a medium bowl, combine the meatballs, sauce, and parsley.	
5	**GT:** Add the beef broth, milk, salt, pepper, and Worcestershire sauce and give it a stir. **PRED:** Transfer the meatballs to a baking sheet lined with parchment paper.	
6	**GT:** Bring the liquid to a boil, then add the egg noodles.. **PRED:** Pour the meatballs into the baking dish.	
7	**GT: Stir** constantly until the pasta is cooked and the liquid has reduced to a sauce that coats the noodles, about seven to eight minutes.. **PRED:** Add the meatballs to the sauce and **stir** to combine.	
8	**GT:** Add the parmesan and the parsley, stirring until the cheese is melted. **PRED:** Enjoy!	
9	Enjoy! Enjoy!	

Figure 4.28: Next step prediction of our visual model for *"One-pot Swedish Meatball Pasta"*. The blue sentences are our model's predictions. Our prediction for step3 matches the GT step4. Prediction for step4 is somehow plausible as a future step as the GT in step5 suggests mixing the sauce and meatballs. Note that our model predicts the next steps before seeing these segments!

ING: salt, butter, flour, garlic, olive oil, cloves, warm water, bread flour, fresh parsley, dry yeast

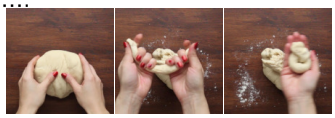
1 **GT:** For the dough, mix together the **flour**, **yeast**,and **salt**, and make a well in the middle.
PRED: In a small bowl, combine the **yeast**, water and **salt**.

2 **GT:** Pour the water and 1 teaspoon of olive oil into the well and **mix together**.
PRED: Add the yeast mixture and **mix** until the dough comes **together**.

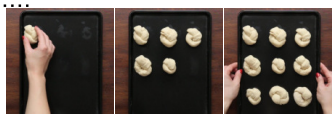
3 **GT:** **Knead** for about 7-10 minutes until the dough is springy.
PRED: **Knead** the dough for a few minutes, then cover and leave to rise for 1 hour.

4 **GT:** Place the dough in the bowl, cover, leave to rise in a warm place for 1 hour.
PRED: Knead the dough for 5 minutes, then knead for a few minutes until smooth and elastic.

5 **GT:** Tear the dough into 9 separate balls and then **roll** into long tubes, folding over to make a knot.
PRED: Once the dough has risen, **roll** it out on a floured surface to a thickness of about 1/4.



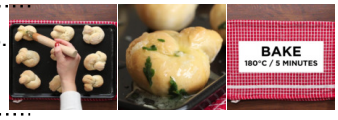
6 **GT:** **Place** the knots on a **baking tray**.
PRED: **Place** on a **baking tray** lined with parchment paper.



7 **GT:** **Bake** in a 350°F (180°C) oven for 10 minutes.
PRED: **Bake** for 20 minutes, or until the bread is golden brown.

8 **GT:** Prepare the garlic butter by mixing the butter, garlic, and parsley.
PRED: Serve hot.

9 **GT:** Coat the knots with the **butter mixture** and bake for another 5 minutes.
PRED: Brush the garlic **butter mixture** over the bread.

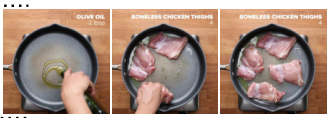


10 Enjoy! Enjoy!

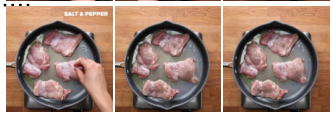
Figure 4.29: Next step prediction of our visual model for *"Garlic Knots"*. The blue sentences are our model's predictions. After baking, in step7, our model predicts that the dish should be served, but after visually seeing the butter parsley mixture in step8, it correctly predicts that the knots should be brushed in step9. Note that our model predicts the next steps before seeing these segments!

ING: salt, pepper, olive oil, tomatoes, green beans, chicken thighs, basil pesto

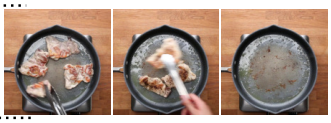
1 **GT:** In a large pan, heat olive oil and add chicken thighs.
PRED: Preheat oven.



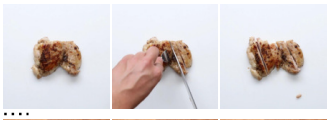
2 **GT: Season with salt and pepper**.
PRED: Season with salt and pepper.



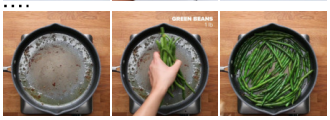
3 **GT:** When the chicken is completely **cooked through**, remove from pan.
PRED: Cook for about 5 minutes on each side, or until **cooked through.**



4 **GT:** Slice into strips, and set aside.
PRED: Add the green beans to the pan and cook for 2 minutes.



5 **GT:** Add green beans and cook until crisp tender.
PRED: Place the chicken thighs on top of the pesto.



6 **GT:** Return the chicken strips to the pan, then **add tomatoes** and **pesto**.
PRED: Add the **pesto** and **tomatoes**.

7 **GT:** Stir until fully incorporated.
PRED: Pour the remaining sauce over the chicken and vegetables.

8 **GT:** Divide into 4 food storage containers and store in the refrigerator.
PRED: Enjoy!

9 Or serve immediately! Enjoy!

Figure 4.30: Next step prediction of our visual model for *"Weekday Meal-prep Pesto Chicken and Veggies"*. The blue sentences are our model's predictions. Our model's prediction for step4 is a plausible future step prediction as it happens in step5. Predictions for step8 and step9 are plausible recommendations. Note that our model predicts the next steps before seeing these segments!

Chapter 5

Temporal Aggregate Representations for Long-Range Videos

Contents

In this chapter, we present our method for learning representations for long-range activity videos. The content of this chapter corresponds to our ECCV 2020 publication, *Temporal Aggregate Representations for Long-Range Video Understanding.* (Sener et al., 2020). Figure 5.1 shows an example video where a person is preparing coffee. There are five sub-activities, including the background, SIL, with varying lengths of frames. Modelling such long-range videos is crucial for tasks like temporal action segmentation, action anticipation, and action

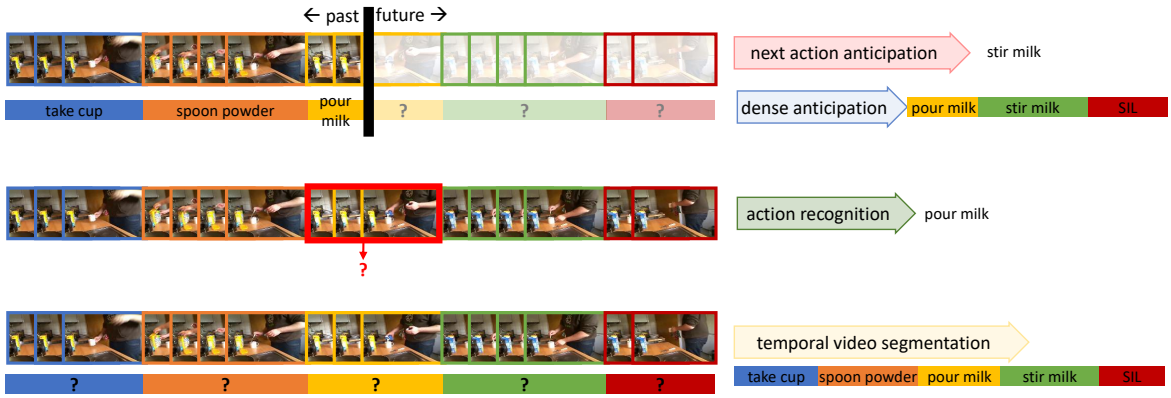


Figure 5.1: Our temporal aggregation model is designed to represent minutes-long videos. Our model can encode the recent and long-term observations to predict actions a few seconds before they start and perform dense anticipation. It can also be employed to perform action recognition and temporal action segmentation in videos with slight modifications.

recognition. We present a temporal aggregation model that could be used for these tasks with slight modifications.

In our work, we mainly focus on validating our approach on action anticipation on instructional and daily activities. Future prediction requires reasoning from current and past observations and raises several fundamental questions. How much past information is necessary? What is a reasonable temporal scale to process the past? How much semantic abstraction is required? We address all of these questions with a flexible multi-granular temporal aggregation framework. We show that it is possible to achieve state-of-the-art results in both next action and dense anticipation using simple techniques such as max-pooling and attention. We conduct experiments on the Breakfast Actions, 50Salads, and Epic-Kitchens datasets to demonstrate the anticipation capabilities of our model, where we achieve state-of-the-art or comparable results. We also show that our model can be used for temporal action segmentation and action recognition with minimal modification.

5.1 Introduction

We tackle long-range video understanding, specifically anticipating not-yet-observed but upcoming actions. When developing intelligent systems, one needs not only to recognize what is *currently* taking place – but also predict what will happen *next*. Anticipating human actions is essential for many applications such as smart surveillance, autonomous driving, assistive robotics, and other human-computer interfaces.

While action anticipation is a niche (albeit rapidly growing) area, the key issues that arise are germane to long-range video understanding as a whole. How should temporal or sequential relationships be modelled? What temporal extent of information and context needs to be processed? At what temporal scale should they be derived, and how much

semantic abstraction is required? The answers to these questions are not only entangled with each other but also depend very much on the videos being analyzed. Here, one needs to distinguish between clipped actions, e.g., of UCF101 (Soomro et al., 2012), versus the multiple actions comprising complex activities in long video streams, e.g., of the Breakfast Actions dataset (Kuehne et al., 2014b). In the former, the actions and video clips are on the order of a few seconds, while in the latter, it is several minutes. As such, temporal modelling is usually not necessary for simple action recognition (Huang et al., 2018b), but more relevant for understanding complex activities (Kuehne et al., 2014b; Huang et al., 2016; Lea et al., 2017; Richard et al., 2017; Sener and Yao, 2018).

Temporal models that are built into the architecture (Ding and Xu, 2018; Farha and Gall, 2019b; Huang et al., 2016; Kuehne et al., 2014b; Richard et al., 2017) are generally favored because they allow frameworks to be learned end-to-end. However, this means that the architecture also dictates the temporal extent that can be accounted for. This tends to be short, either due to difficulties in memory retention, as is the case with LSTMs and transformers, or due to model size, as with 3D CNNs, temporal convolution networks (TCNs), and transformers. As a result, the context for anticipation can only be drawn from a limited extent of recent observations, usually on the order of seconds (Lan et al., 2014; Vondrick et al., 2016; Miech et al., 2019a). This, in turn, limits the temporal horizon i.e., only the next (few) action(s) and granularity of the prediction i.e., next action label vs. dense predictions. However, observations from further temporal contexts may contain some critical cues. For example, a person making scrambled eggs can add salt at any time from start to end; if we are able to ascertain from (further) past actions that salt has already been added, the next action is less likely *"adding salt"*. In our model, we attempt to make use of the entire observation to make more accurate predictions.

One way to ease the computational burden, especially under longer temporal extents, is to use higher-level but more compact feature abstractions, e.g., by using already detected objects and people (Wu et al., 2019) or even sub-activity labels (Farha et al., 2018; Ke et al., 2019b) based on the outputs of temporal action segmentation algorithms (Richard et al., 2017; Farha and Gall, 2019b). Such an approach places a strong semantic load on the initial task of segmentation and is often difficult to train in a single-stage. Furthermore, since labelling and segmenting actions from videos are difficult tasks and open areas of research themselves, their errors which may then propagate onwards when anticipating future actions.

Motivated by these questions of temporal modelling, extent, scaling, and level of semantic abstraction, we aim to perform a systematic study for long-range video understanding. Based on conclusions drawn from these studies, we design and propose a general framework for encoding long-range video. We aim for flexibility in frame input, i.e., ranging from low-level visual features to high-level semantic labels, and the ability to meaningfully integrate recent observations with long-range context, all in a computationally efficient way.

To do so, we split video streams into snippets of equal length and max-pool the frame features within the snippets. We then create ensembles of multi-scale feature representations that are aggregated bottom-up based on scaling and temporal extent. Our motivation for creating video snippets is sampling from long videos, where the samples are supposed to be

as representative of the actions as possible. However, the duration of actions varies a lot in complex activities. Furthermore, for different complex activities, it might be necessary to use a different number of snippets. To achieve this, we use an ensemble of different scales for long-range and recent observations. In this way, depending on the duration of the snippets, we expect to capture actions at different granularities, ranging from gestures such as "reach for milk", over actions such as "pour milk", to compositional actions such as "open fridge and get the milk out of fridge".

Temporal aggregation (Kline and Snodgrass, 1995) is a form of summarization used in temporal database systems. Our framework is loosely analogous as it summarizes the past observations through aggregation, so we name it "temporal aggregates". Temporal aggregate features can be applied to several video understanding tasks; in addition to action anticipation, it can also be used for recognition and temporal action segmentation.

In particular, we aim to integrate and relate recent observations with the long-range observations at various granularities to predict the future, recognize current actions, or to perform temporal action segmentation and do so in a computationally efficient manner. Our temporal aggregates model is similar in spirit to the long-term feature banks (Wu et al., 2019) in that we couple the recent with the long-range past using attention. However, one key difference is that we work with an ensemble of multiple scalings and granularities, whereas Wu et al. (2019) work at a single (frame-level) granularity. As such, we can handle long streams of videos that can span tens of minutes, while they are only able to work on short video clips. Furthermore, their method requires a preliminary step of object and person detection while we can work directly on frame-wise features, making our inputs much simpler.

We summarize our main contributions as follows:

- We propose a flexible and simple single-stage framework of multi-scale temporal aggregates for long-range video understanding by relating recent observations to the long-range past. It can encode long video durations and attend to salient segments to predict the future or recognize the current actions.

- Our representations can be applied to several video understanding tasks; in addition to action anticipation, it can be used for recognition and temporal action segmentation with minimal modifications and is able to achieve competitive results.

- Our model has minimal constraints regarding the type of anticipation (dense or next action), type of the dataset (instructional or daily activity datasets), and type of input features (visual features or frame-level labels).

- We show that our method either outperforms or is on par with the state of the art for anticipation, recognition, and segmentation with experiments conducted on three datasets: Breakfast Actions (Kuehne et al., 2014b), 50Salads (Stein and McKenna, 2013) and Epic-Kitchens (Damen et al., 2018).

5.2 Related Works

Early activity recognition: The earlier future prediction works were mostly focused on predicting ongoing but not yet completed activities as early as possible. The problem is formulated as inference under incomplete observations (Ryoo, 2011; Hoai and De la Torre, 2014), and recently it has been tackled mostly with RNNs (Ma et al., 2016; Kong et al., 2018). Methods for early event recognition require partial observations of the already ongoing actions. Unlike these, we address the more challenging task of action *anticipation*, seconds before the action starts.

Action recognition has advanced significantly with deep networks in recent years. Many directions have been explored successfully with new feature representations, backbones, etc. Notable works include two steam networks (Simonyan and Zisserman, 2014a; Wang et al., 2016a), 3D convolutional networks (Tran et al., 2015; Xie et al., 2018; Carreira and Zisserman, 2017), and recurrent neural networks (Donahue et al., 2015; Yue-Hei Ng et al., 2015). These methods have been designed to encode short clips of a few seconds and are typically applied to the classification of *trimmed* videos containing a single action such as those found in the UCF (Soomro et al., 2012) and Kinetics (Kay et al., 2017) datasets. In contrast, in our work, we are working with long *untrimmed* sequences of complex activities where the actions are not cleanly segmented. Such long videos are not simply a composition of short independent actions, as the composing segments are related to each other with sequence dynamics. Various models for complex activity understanding have been addressed before (Ding and Xu, 2018; Richard et al., 2017; Sener and Yao, 2018); these approaches are designed to work on instructional videos by explicitly modelling their sequence dynamics. These models are not flexible enough to be extended to daily activities with loose orderings. Moreover, when only partial observations are provided, e.g., for anticipation, these models cannot be trained in a single-stage.

Action anticipation, a fast-growing field, aims to forecast actions before they occur. We distinguish between immediate anticipation of the next action label and dense anticipation.

Prior works in immediate anticipation were initially limited to movement primitives like *reaching* or *moving* (Koppula and Saxena, 2015) or interactions such as *hugging* and *handshaking*. Vondrick et al. (2016) classify predicted future frame features to anticipate interactions. Mahmud et al. (2017) present a model for predicting both the next action and its starting position. Damen et al. (2018) present a large daily activities dataset, along with a challenge for anticipating the next action one second before occurrence. Miech et al. (2019a) propose direct next action anticipation from recent observations. Recently, Furnari and Farinella (2019) proposed using an LSTM to summarize the past and another LSTM for future prediction. All these works assume current or near-past information, whereas we make use of long-range past for anticipation.

Dense anticipation predicts actions multiple steps into the future. Previous methods (Farha et al., 2018; Ke et al., 2019b) to date have all relied on having already segmented temporal observations. For example, Farha et al. (2018) uses outputs of a temporal action segmentation method (Richard et al., 2017) to iteratively predict future actions using

an RNN. Ke et al. (2019b) bypass the iterative approach and proposes a time-conditioned method that directly anticipates actions for specific future times. They also require an auxiliary segmentation method for generating input data. Different than these, our model can perform dense anticipation in a single-stage on visual features without any pre-segmented nor labelled inputs.

Datasets for action anticipation should include videos of real-life environments and scenarios. As such, we use three datasets: Breakfast Actions (Kuehne et al., 2014b), 50Salads (Stein and McKenna, 2013), and Epic-Kitchens (Damen et al., 2018), which reflect the actual durations and orderings of actions. This is crucial for real-world deployment of anticipations models. Datasets frequently used in action localization (Caba Heilbron et al., 2015) are not suitable for our purposes, as they contain videos repeating a single action. In contrast, the videos we work with contain a sequence of different actions. Similarly, sports datasets (Felsen et al., 2017), which involve multiple people, make anticipation more challenging, as it requires explicit detection of people and their interactions. In our datasets, a single person performs a sequence of different actions, which provides a nicely constrained scenario where we can test our model.

Motion and temporal dynamics: The role of motion and temporal dynamics has been well-explored for video understanding, though the focus has been on short video clips (Lin et al., 2019a; Donahue et al., 2015; Carreira and Zisserman, 2017; Huang et al., 2018b; Miech et al., 2017). Some works were able to use longer-term temporal contexts in short videos using pre-computed features (Li et al., 2017; Tang et al., 2018). Wu et al. (2019) proposed integrating long-term features with 3D CNNs for action recognition in short videos and showed the importance of incorporating temporal context for action recognition. Recently Feichtenhofer et al. (2019) proposed slow-fast networks, which similar to our model, encode time-wise multi-scale representations. Using a two-stream architecture working on differently sampled versions of the video, they aim to capture slow spatial semantics and fast motion features that are successively concatenated. In contrast, our multi-scale spanning past is built on standard features extracted at a fixed sampling rate. Our spanning features of *all* scales are processed w.r.t. *all* recent scales via an attention mechanism. Most importantly, these approaches are limited to short videos and cannot be extended to minute-long videos due to computational constraints. Our temporal modelling enables encoding minute-long videos while being flexible enough to be applied to any dataset for several tasks.

5.3 Representations

We begin by introducing the representations which are given as inputs to the building blocks of our framework. A schematic of the representations can be found in Figure 5.2. We had two rationales when designing our network. First, the coupling blocks relate recent observations to long-range past, since some actions directly determine what future actions can or cannot be. Second, to represent recent and long-range past at various granularities, we pool snippets over multiple scales.

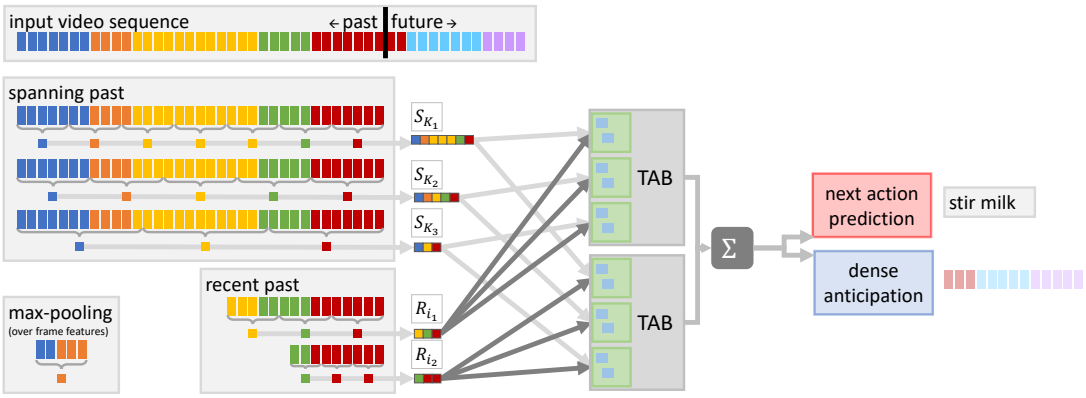


Figure 5.2: Model overview: in this example we use 3 scales for computing the "spanning past" snippet features $\mathbf{S}_{K_1}, \mathbf{S}_{K_2}, \mathbf{S}_{K_3}$, and 2 starting points to compute the "recent past" snippet features, $\mathbf{R}_{i_1}, \mathbf{R}_{i_2}$, by max-pooling over the frame features in each snippet. Each recent snippet is coupled with all the spanning snippets in our Temporal Aggregation Blocks (TABs). An ensemble of TAB outputs is used for dense or next action anticipation. Best viewed in color.

5.3.1 Video Representations

For a video of length T, we denote the feature representation of a single video frame indexed at time t as $f_t \in \mathbb{R}^D, 1 \leq t \leq T$. f_t can be derived from low-level features, such as IDT (Wang and Schmid, 2013) or I3D (Carreira and Zisserman, 2017), or high-level abstractions, such as sub-activity labels derived from segmentation algorithms. To reduce computational load, we work at a snippet-level instead of a frame level. We define a snippet feature $\mathbf{F}_{ij;K}$ as the concatenation of max-pooled features from K snippets, where snippets are partitioned consecutively from a range of frames starting at i and ending at j:

$$\begin{aligned} \mathbf{F}_{ij;K} &= [F_{i,i+k}, F_{i+k+1,i+2k}, ..., F_{j-k+1,j}], \text{ where} \\ (F_{p,q})_d &= \max_{p \leq t \leq q} \{f_t\}_d \,, \; 1 \leq d \leq D, \quad k = {}^{(j\,-\,i)}/_{K}. \end{aligned} \tag{5.1}$$

Here, $F_{p,q}$ indicates the maximum over each dimension d of the frame features in a given snippet between frames p and q, though it can be substituted with other alternatives. In the literature, methods representing snippets or segments of frames range from simple sampling and pooling strategies to more complex representations such as learned pooling (Lin et al., 2018; Miech et al., 2017), 3D CNNs (Carreira and Zisserman, 2017) and LSTMs (Ostyakov et al., 2018). Especially for long snippets, it is often assumed that a learned representation is necessary (Girdhar et al., 2017; Lee et al., 2018b), though their effectiveness over simple pooling is still controversial (Wang et al., 2016a). The learning of novel temporal pooling approaches goes beyond the scope of this work and is an orthogonal line of development. In this work, we verify established methods (see Section 5.5.5) and find that a simple max-pooling is surprisingly effective and sufficient.

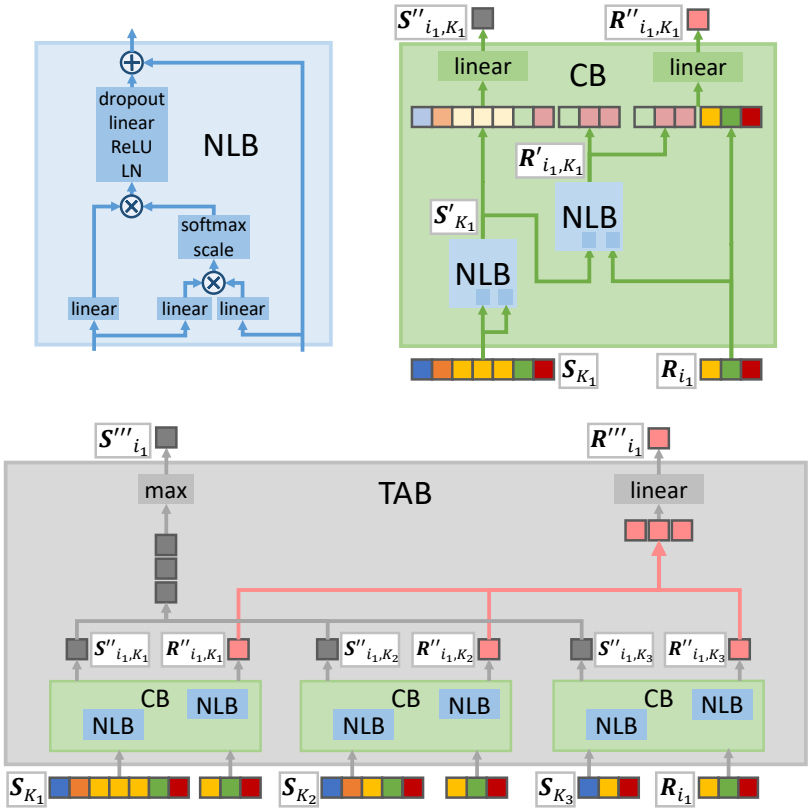


Figure 5.3: Overview of model components: non-local blocks (NLB) compute interactions between two representations via attention (see Section 5.4.1). Two such NLBs are combined in a Coupling Block (CB) which calculates the attention-reweighted spanning and recent past representations (see Section 5.4.2). We couple each recent with all spanning representations via individual CBs and combine their outputs in a Temporal Aggregation Block (TAB) (see Section 5.4.3). The outputs of multiple such TABs are combined to perform anticipation, see Figure 5.2. Colors in the input video sequence refer to the action. Best viewed in color.

5.3.2 Recent vs. Spanning Representations

Based on different start and end frames i and j and number of snippets K, we define two types of snippet features: *'recent'* features $\{\mathcal{R}\}$ from recent observations and *"spanning"* features $\{\mathcal{S}\}$ drawn from the entire video. The recent snippets cover a couple of seconds (or up to a minute, depending on the temporal granularity) before the current time point, while spanning snippets refer to the long-range past and may last up to ten minutes. For *"recent"* snippets, the end frame j is fixed to the current time point t and the number of snippets is fixed to K_R. Recent snippet features $\mathcal{R}$ can be defined as a feature bank of snippet features with different start frames i, i.e.,

$$\mathcal{R} = \{\mathbf{F}_{i_1 t;K_R}, \mathbf{F}_{i_2 t;K_R}, ..., \mathbf{F}_{i_R t;K_R}\} = \{\mathbf{R}_{i_1}, \mathbf{R}_{i_2}, ..., \mathbf{R}_{i_R}\}, \tag{5.2}$$

where $\mathbf{R}_i \in \mathbb{R}^{D \times K_R}$ is a shorthand to denote $\mathbf{F}_{i\,t;K_R}$, since endpoint t and number of snippets K_R are fixed. In Figure 5.2 we use two starting points to compute the "recent past" snippet features and represent each with $K_R = 3$ number of snippets (&).

For *"spanning"* snippets, i and j are fixed to the start of the video and current time point, i.e., $i = 0, j = t$. Spanning snippet features $\mathcal{S}$ are defined as a feature bank of snippet features with varying number of snippets K, i.e.,

$$\mathcal{S} = \{\mathbf{F}_{0\,t;K_1}, \mathbf{F}_{0\,t;K_2}, ..., \mathbf{F}_{0\,t;K_S}\} = \{\mathbf{S}_{K_1}, \mathbf{S}_{K_2}, ..., \mathbf{S}_{K_S}\}, \tag{5.3}$$

where $\mathbf{S}_K \in \mathbb{R}^{D \times K}$ is a shorthand for $\mathbf{F}_{0\,t;K}$. In Figure 5.2 we use three scales to compute the "spanning past" snippet features with $K = \{7, 5, 3\}$ (, &).

Key to both types of representations is the ensemble of snippet features from multiple scales. We achieve this by varying the number of snippets K for the spanning past. For the recent past, it is sufficient to keep the number of snippets K_R fixed, and vary only the start point i, due to redundancy between $\mathcal{R}$ and $\mathcal{S}$ for the snippets that overlap. For our experiments, we work with snippets ranging from seconds to several minutes.

5.4 Framework

In Figure 5.3 we present an overview of the components used in our framework, which we build in a bottom up manner, starting with the recent and spanning features $\mathcal{R}$ and $\mathcal{S}$, which are coupled with non-local blocks (NLB) (Section 5.4.1) within coupling blocks (CB) (Section 5.4.2). The outputs of the coupling blocks from different scales are then aggregated inside temporal aggregation blocks (TAB) (Section 5.4.3). Outputs of different TABs can then be chained together for either next action anticipation or dense anticipation (Sections 5.5.6, 5.5.8).

5.4.1 Non-Local Blocks (NLB)

We apply non-local operations to capture relationships amongst the spanning snippets and between spanning and recent snippets. Non-local blocks (Wang et al., 2018c) are a flexible way to relate features independently from their temporal distance and thus capture long-range dependencies. We use the modified non-local block from Wu et al. (2019), which adds layer normalization (Ba et al., 2016b) and dropout (Srivastava et al., 2014) to the original one by Wang et al. (2016a). Figure 5.3 (left-top) visualizes the architecture of the block, the operation of which we denote as NLB$(\cdot, \cdot)$.

5.4.2 Coupling Block (CB)

Armed with the NLB operation, we define attention-reweighted spanning and recent outputs as:

$$\mathbf{S}'_K = NLB(\mathbf{S}_K, \mathbf{S}_K) \tag{5.4}$$

$$\mathbf{R}'_{i,K} = NLB(\mathbf{S}'_K, \mathbf{R}_i). \tag{5.5}$$

The coupling is done by concatenating $\mathbf{R}'_{i,K}$ with either $\mathbf{R}_i$ or $\mathbf{S}'_K$ and passed through linear layers. This results in the fixed-length representations $\mathbf{R}''_{i,K}$ and $\mathbf{S}''_{i,K}$, where i is the starting point of the recent snippet and K is the scale of the spanning snippet.

5.4.3 Temporal Aggregation Block (TAB)

The final representation for recent and spanning past is computed by aggregating outputs from multiple CBs. For the same recent starting point i, we concatenate $\mathbf{R}''_{i,K_1}, ..., \mathbf{R}''_{i,K_S}$ for all spanning scales and pass the concatenation through a linear layer to compute $\mathbf{R}'''_i$. The final spanning past representation $\mathbf{S}'''_i$ is a max over all $\mathbf{S}''_{i,K_1}, ..., \mathbf{S}''_{i,K_S}$. We empirically find that taking the max outperforms other alternatives like linear layers and/or concatenation for the spanning past (see 5.5.5).

TAB outputs, by varying recent starting points $\{i\}$ and scales of spanning snippets $\{K\}$, are multi-granular video representations that aggregate and encode both the recent and long-range past. We name these **temporal aggregate representations**. Figure 5.2 shows an example with 2 recent starting points and 3 spanning scales. Temporal aggregate representations are generic and can be applied in various video understanding tasks (see Section 5.4.4) from long streams of video. We find they are especially well-suited for anticipation, as they are designed to encode long video durations while attending to salient snippets (see Section 5.5.5.4). Next, we describe how we use temporal aggregates as inputs to various prediction models.

5.4.4 Prediction Model

5.4.4.1 Classification

For single-label classification tasks such as next action anticipation, temporal aggregate representations can be used directly with a classification layer (linear + softmax). A cross-entropy loss based on ground truth labels Y can be applied to the predictions $\hat{Y}_i$, where Y is either the current action label for recognition, or the next action label for next action prediction, see Figure 5.4.

When the individual actions compose a complex activity (e.g., "take bowl" and "pour milk" as part of "making cereal" in Breakfast (Kuehne et al., 2014b)), we can add an additional loss based on the complex activity label Z. We postulate that predicting Z as an auxiliary task will boost the performance. For this we concatenate $\mathbf{S}'''_{i_1}, \ldots, \mathbf{S}'''_{i_R}$ from all TABs and again pass them through a classification layer to obtain $\hat{Z}$. Our final loss formulation is the sum of the cross entropies over the action and complex activity labels respectively:

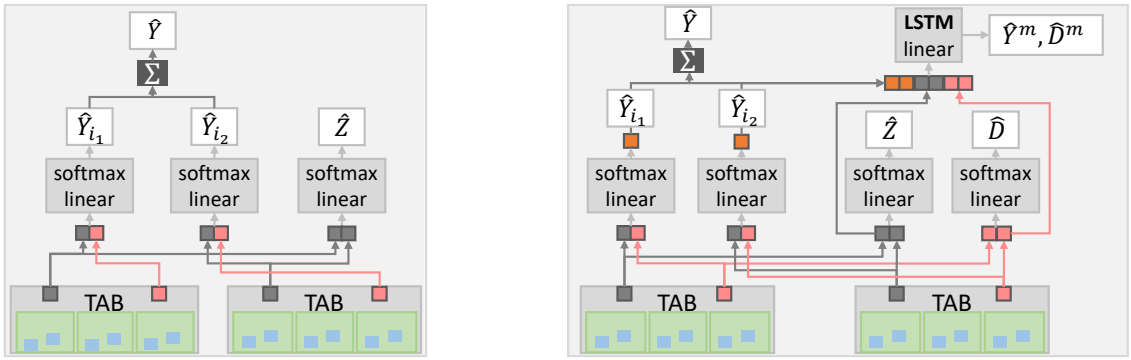


Figure 5.4: Prediction models for classification (left) and sequence prediction (right).

$$\mathcal{L}_{\text{cl}} = \mathcal{L}_{\text{comp}} + \mathcal{L}_{\text{action}} = -\sum_{n=1}^{N_Z} Z_n \log(\hat{Z})_n - \sum_{r=1}^{R}\sum_{n=1}^{N_Y} Y_n \log(\hat{Y}_{i_r})_n, \tag{5.6}$$

where i_r is one of the R recent starting points, and N_Y and N_Z are the total number of actions and complex activity classes respectively. During inference, the predicted scores, post softmax for all starting points i_r, are summed for a final prediction, i.e., $\hat{Y} = \max_n(\sum_{r=1}^{R} \hat{Y}_{i_r})_n$.

We also frame sequence segmentation as a classification task. Here the task is predicting frame-level action labels of complex activity videos. We generate multiple sliding windows with start and end times t_s and t_e. Our goal is to classify each window into an action using the loss in Equation 5.6.

5.4.4.2 Sequence prediction

The dense anticipation task predicts frame-wise action labels of the entire future sequence. Others (Farha et al., 2018) have formulated this task as predicting future segment labels and regressing their durations. We adopt a similar approach, but also estimate the duration via classification. We discretize the sequence duration into N_D intervals and represent durations as one-hot encodings $D \in \{0,1\}^{N_D}$.

For dense predictions, we perform multi-step estimates, distinguishing between the current action and its duration versus future actions and durations. We first estimate the current action and complex activity label, as described in Equation 5.6. The current duration D is then estimated via a classification layer applied to the concatenation of recent temporal aggregates $\mathbf{R}'''_{i_1}, ..., \mathbf{R}'''_{i_R}$.

For future actions, we concatenate all recent and spanning temporal aggregates $\mathbf{R}'''_{i_1}, ..., \mathbf{R}'''_{i_R}$ and $\mathbf{S}'''_{i_1}, ..., \mathbf{S}'''_{i_R}$, as well as the classification layer outputs $\hat{Y}_{i_1}, ..., \hat{Y}_{i_R}$, and pass the concatenation through a linear layer before feeding the output to a one-layer LSTM. The LSTM consecutively predicts M future action labels and their durations. For this, at each iteration m, the LSTM predicts an action vector $\hat{Y}^m$ and a duration vector $\hat{D}^m$. At each step the future action and its duration are combined and fed to the LSTM to predict the

next future action and length (see Figure 5.4).

We formulate the dense anticipation loss as the sum of the cross-entropies over the current action, its duration, future actions and durations, and task labels respectively:

$$\mathcal{L}_{\text{dense}} = \mathcal{L}_{\text{cl}} - \sum_{n=1}^{N_D} D_n \log(\hat{D})_n - \frac{1}{M} \sum_{m=1}^{M} \left(\sum_{n=1}^{N_Y} Y_n^m \log(\hat{Y}^m)_n + \sum_{n=1}^{N_D} D_n^m \log(\hat{D}^m)_n \right). \quad (5.7)$$

During inference we sum the predicted scores (post soft-max) for all starting points i_r to predict the current action as $\max_n(\sum_{r=1}^{R} \hat{Y}_{i_r})_n$. The LSTM is then applied recursively to predict subsequent actions and durations by feeding previous predictions until the full sequence length is reached.

5.4.5 Implementation Details

We train our model using the Adam optimizer (Kingma and Ba, 2014) with batch size 10, learning rate 10^{-4} and dropout rate 0.3. We train for 25 epochs and decrease the learning rate by a factor of 10 every 10^{th} epoch. We use 1024 dimensions for all non-classification linear layers for the Breakfast Actions and 50Salads datasets and 512 dimensions for the Epic-Kitchens dataset. The LSTMs in dense anticipation have one layer and 512 hidden units.

5.5 Experiments

5.5.1 Datasets and Features

We conduct experiments on two instructional activity datasets, Breakfast Actions (Kuehne et al., 2014b) and 50Salads (Stein and McKenna, 2013), and one daily activity dataset, Epic-Kitchens (Damen et al., 2018).

The Breakfast Actions dataset (Kuehne et al., 2014b) contains 1712 videos and shows complex activities related to breakfast preparation of 10 high-level tasks like "making coffee", "making tea" and so on. Each video is composed of multiple actions. Overall, there are 48 different actions, such as "pouring water" or "stirring coffee", with approximately 11.3K samples, including around 3.6K "silence (SIL)" samples. The average duration of the videos is 2.3 minutes. There are, on average, six actions per video. There are four splits, and we report our results averaged over these predefined splits. We use two types of frame-wise features: Fisher vectors computed as in by Farha et al. (2018) and I3D features (Carreira and Zisserman, 2017). The Fisher vectors are computed over the improved dense trajectories (Wang and Schmid, 2013), over which a 64 dimensional PCA is applied.

The 50Salads dataset (Stein and McKenna, 2013) includes 50 videos and 17 different actions for a single task, namely making mixed salads. Since 50Salads has only a single complex activity (making salad), we omit complex activity prediction for it. On average, 50Salads has 20 actions per video due to repetitions. The average video duration is 6.4

Dataset	segment duration median, mean ±std	# actions	# verbs	# noun	# segments
Breakfast(@15fps)	15.1s, 26.6s ±36.8	48	14	34	11.3K
50Salads(@30fps)	29.7s, 38.4s ±31.5	17	6	12	0.9K
EPIC(@60fps)	1.9s, 3.7s ± 5.6	2513	125	351	39.6K

Table 5.1: Detailed statistics about the three datasets used in our work.

Dataset	$\{i\}$(in seconds)	spanning scope	K_R	$\{K\}$
Breakfast(@15fps)	$\{t-10, t-20, t-30\}$	entire past	5	$\{10, 15, 20\}$
50Salads(@30fps)	$\{t-5, t-10, t-15\}$	entire past	5	$\{5, 10, 15\}$
EPIC(@60fps)	$\{t-1.6, t-1.2, t-0.8, t-0.4\}$	6 seconds	2	$\{2, 3, 5\}$

Table 5.2: Our model parameters for the datasets used in our work.

minutes. There are five splits, and we again average our results over them. We represent the frames using Fisher vectors (Farha et al., 2018) and I3D features (Carreira and Zisserman, 2017). Note that since the 50Salads is a very small dataset, for our next action and dense anticipation on the 50Salads, we augment our training data by shifting the starting frame by several seconds.

The Epic-Kitchens dataset (Damen et al., 2018) is a large first-person video dataset which contains 432 sequences and 39,594 action segments recorded by participants performing non-scripted daily activities in their kitchen. The average duration of the videos is 7.6 minutes ranging from 1 minute to 55 minutes. An action is defined as a combination of a verb and a noun, e.g. "boil milk". There are in total 125 verbs, 351 nouns, and 2513 actions. The dataset provides a training and test set which contains 272 and 160 videos, respectively. The test set is divided into two splits: Seen Kitchens (S1), where sequences from the same environment are in the training data, and Unseen Kitchens (S2) where complete sequences of some participants are held out for testing. The labels for the test set are not shared, as there is an action anticipation [1] and action recognition challenge [2]. We use the appearance (rgb), motion (optical flow), and object-based features provided by Furnari and Farinella (2019). They independently train the spatial and motion CNNs using the TSN (Wang et al., 2016a) framework for action recognition on Epic-Kitchens. They also train object detectors to recognize the 352 object classes of the Epic-Kitchens dataset. The feature dimensions are 1024, 1024, and 352 for appearance, motion, and object features, respectively. We use a validation set provided by Furnari and Farinella (2019), which is constructed using 40 videos from the training set for selecting Epic-Kitchens parameters. For Epic-Kitchens, we report our results both on this validation set and on the test set of the anticipation challenge.

5.5.2 Statistics, Evaluations Measures & Parameters

The sequences in these datasets reflect realistic durations and orderings of actions, which is crucial for real-world deployment of anticipation models. We provide relevant statistics about the datasets used in our work to show a broader comparison of their scale and label granularity in Table 5.1. One notable difference between these datasets is the label granularity; it is very fine-grained for Epic-Kitchens, hence their 2513 action classes, versus the coarser 48 and 17 actions of Breakfast and 50Salads. As a result, the median action segment duration is 8-16x shorter.

Evaluation measures we are using in this work are class accuracy (Accuracy (%)) for next action prediction and mean over classes accuracy (Farha et al., 2018) for dense prediction. For Epic-Kitchens, we report Top-1 and Top-5 accuracies to be consistent with (Miech et al., 2019a; Furnari and Farinella, 2019).

Parameters: The spanning scales $\{K\}$, recent scale K_R, recent starting points $\{i\}$ and spanning scope for each dataset are given in Table 5.2. We cross-validated the parameters on different splits of 50Salads and Breakfast and used a validation set provided by Furnari and Farinella (2019) for selecting Epic-Kitchens parameters. We use intervals of 5 and 20 seconds for Breakfast and 50Salads for discretizing the durations in dense anticipation. The starting points for Epic-Kitchens are much smaller than the others due to its fine label granularity as the average action duration is on the order of seconds vs. minutes for the other datasets. We use a spanning scope of 6 seconds for Epic-Kitchens while we are using the entire past as our "long range" past for the other datasets. "Long-range" and "entire past" are relative concepts. For instructional datasets, e.g., the Breakfast and 50Salads, the average action duration is around 15 and 30 seconds, and the annotations are coarse. Also, instructional activities follow a loose ordering for completing a task, and therefore, models benefit from observing the entire past. For such datasets, we use the entire past. People in daily activities act unscripted, and thus the long-range past might be less relevant. Moreover, Epic-Kitchens is a special daily dataset where the action annotations are very fine-grained, lasting 3.7 seconds on average. An example sequence of annotations in Epic-Kitchens is "take kettle - put down kettle - pour water - watch boiling water", while in the Breakfast the same group of actions would be annotated as "boil water". In 20 seconds, one observes, on average, 5-6 actions in Epic-Kitchens videos. Therefore, in Epic-Kitchens, we define the long-range duration in terms of seconds (see Section 5.5.7 for detail).

5.5.3 Model Validation

For validating our method's capabilities in modelling sequences, we make baseline comparisons. The most straightforward approach for solving the next action anticipation task is using a transition matrix (TM) (Miech et al., 2019a), which encodes the transition from one action to the next. A more sophisticated solution is building a lookup table (LUT) of varying length sequences, which allows encoding the context in a more explicit manner. The problem

[1] https://competitions.codalab.org/competitions/20071
[2] https://competitions.codalab.org/competitions/20115

	cereal	coffee	f.egg	juice	milk	panc.	salat	sand.	s.egg	tea	mean±std
TM	77.8	50.8	57.2	57.2	40.1	39.6	57.9	52.4	59.4	54.2	54.6±10.8
LUT	57.5	**59.9**	56.2	58.8	56.1	57.3	55.1	49.6	**61.2**	60.1	57.2±3.1
LSTM	**79.8**	47.2	52.9	61.2	72.7	**73.9**	**64.3**	46.9	60.5	**68.7**	62.8±11.3
ours	69.8	54.7	**62.5**	**65.7**	**72.9**	66.2	63.6	**64.6**	58.0	64.1	**64.2**±5.2

Table 5.3: Model validation using GT labels for next action anticipation on the Breakfast Actions, presented are accuracies. We compare transition matrices (TM), lookup tables (LUT), LSTMs, and our temporal aggregates model (without complex activity prediction).

ours, (obj+flow+rgb)	ours (GT)	TM (GT)	BiLSTM 5 (GT)	BiLSTM 10 (GT)
15.5	**17.9**	17.4	17.3	15.1

Table 5.4: Model validation using GT labels for next action anticipation on the Epic-Kitchens dataset. We report Top-1 action prediction accuracies. We compare our temporal aggregates model trained on the object, appearance and flow features, transition matrices (TM), our model trained on GT action labels, lookup tables (LUT), and LSTMs.

with LUTs is that their completeness depends on the coverage of the training data, and they rapidly grow with the number of actions. So far, for next step prediction, RNNs achieve good performance (Farha et al., 2018), as they learn modelling the sequences.

For our baseline comparisons, instead of frame features, we use the frame-level ground truth labels as input to our model. We compute the TM, LUT, and LSTMs on the ground truth segment-level labels. In Table 5.3 we present comparisons on the Breakfast Actions for the next action anticipation per complex activity. Overall, transition matrices provide the worst results. LUTs improve the results, as they incorporate more contextual information. Both the LSTMs and our method outperform the other alternatives, while our method still performs better than the LSTMs on average. However, applying LSTMs requires parsing the past into action sequences (Farha et al., 2018), which turns the problem into separate segmentation and prediction phases. Our model, on the other hand, can be trained in one stage and can represent the long-range observations good enough to outperform LSTMs. We show that our model is doing better than simply learning pairwise statistics of the dataset.

We conduct similar experiments using frame-level ground truth labels as input to our model on Epic-Kitchens. For the RNN experiments, we split the video sequences in the training set into sequences of $L = \{5, 10\}$ actions with a sliding window using a stride of 1. We train a BiLSTM on these short sequences and predict the next action. Table 5.4 presents our comparisons. Our method outperforms both TM and BiLSTM. Compared to the Breakfast dataset, TM performs better than BiLSTM and the difference between the accuracy of TM and our model is marginal. We also see that when more past actions are incorporated, "BiLSTM 10 (GT)", the BiLSTM gets less accurate. These might be due to the dataset size, which falls short compared to its large number of actions. Therefore, our model and LSTMs cannot learn all the variants of long-term relationships. Compared to using visual features, "ours, (obj+flow+rgb)", we observe a difference of 2.4% in accuracy, which indicates that there is still room for improvement for our model.

Note that for our experiments on the Breakfast Actions and Epic-Kitchens datasets, we use the hyper-parameters reported in Table 5.2. For our experiments using ground truth labels, we use one-hot vector encoding and use this vector as input to our model.

5.5.4 Recognizing Long-range Complex Activities

Method	Fine-tuning	Accuracy (%)
I3D	no	64.3
I3D + Timeception (Hussein et al., 2019)	no	69.3
I3D + ours	no	**80.8**
I3D	yes	80.6
I3D + Timeception (Hussein et al., 2019)	yes	86.9
I3D+ PIC (Hussein et al., 2020)	yes	**89.8**

Table 5.5: Comparisons to methods developed for recognizing long-range complex activities, Timeception (Hussein et al., 2019) and PIC (Hussein et al., 2020) on the Breakfast Actions dataset. Our method outperforms Timeception (Hussein et al., 2019) by a significant margin showing the superiority of our method modelling long-range activities.

To validate our model further, we present comparisons on classifying long-range complex activities. Since these videos include multiple actions and are several minutes long, it becomes more challenging to model their temporal structure compared to short-term single action videos. Recently, Hussein et al. (2019) proposed a neural layer, "Timeception", which uses multi-scale temporal-only convolutions for modelling minutes-long complex activity videos, such as "cooking a meal". Placed on top of backbone CNNs, the permutation invariant convolution layer, PIC (Hussein et al., 2020), also aims at modelling only the temporal dimension. PIC is invariant to temporal permutations as it models their correlations regardless of their order, which helps to handle different action orderings in videos. It also uses pairs of key-value kernels to learn the most representative visual signals in long and noisy videos.

We report our comparisons in Table 5.5 on the Breakfast Actions dataset using two types of I3D features, where one is the features from an I3D model trained on Kinetics only, and the other is the features from an I3D model fine-tuned on the Breakfast dataset. Our method outperforms Timeception (Hussein et al., 2019) by 11.4%, and the I3D backbone by 16.5%. Hussein et al. (2020) use the fine-tuned I3D features on Breakfast and show a 3.1% improvement over Timeception (Hussein et al., 2019). Fine-tuning improves the accuracy by 16.3% and shows that there is room for improvement for our method using better feature representations.

5.5.5 Component Validation

The design choices for our framework are inspired by trends in the state of the art, such as attention (Wang et al., 2018c; Wu et al., 2019) and careful experimentation. We verify each component's utility via a series of ablation studies which are summarized in Tables 5.6.

As our main motivation was to develop a representation for long video streams to perform anticipation, we experiment on the Breakfast Actions dataset for next action anticipation. Our full model gets a performance of 40.1% accuracy averaged over actions.

5.5.5.1 Video Representation

Pooling type	Frame sampling	GRU	BiLSTM	Mean-pooling	Max-pooling
Accuracy (%)	32.1	37.9	38.7	36.6	**40.1**

Table 5.6: Ablations on the influence of different snippet representations on the Breakfast Actions dataset.

Several short-term spatio-temporal feature representations have been proposed for video, e.g., 3D convolutions (Tran et al., 2015), or combining CNNs and RNNs for sequences (Yue-Hei Ng et al., 2015; Donahue et al., 2015). For long video streams, it becomes difficult to work with all the raw features. Selecting representative features can be as simple as sub-sampling the frames, as e.g., in the SlowFast Networks (Feichtenhofer et al., 2019; Xiao et al., 2020), or pooling (Wang et al., 2016a), to more complex RNNs (Yue-Hei Ng et al., 2015). Current findings in the literature are not in agreement. Some propose learned strategies (Miech et al., 2017; Lee et al., 2018b), while others advocate pooling (Wang et al., 2016a). Our experiments align with the latter, showing that max-pooling is superior to both sampling (+8%) and the GRU (+2.2%) or bi-directional LSTM (Conneau et al., 2017) (+1.4%). The performance of GRU and bi-LSTM are comparable to average-pooling but significantly increases the training and inference time. Of the variants of pooling, max-pooling works best. This is in contradiction to the findings of Wang et al. (2016a). We attribute this to the fact that we pool over minutes-long snippets, and it is likely that mean-pooling smooths away salient features that are otherwise preserved by max-pooling. The minimum and maximum snippet durations, over which we apply pooling, are 0.4s and 115.3s for 50Salads, 0.1s and 64.5s for Breakfast, and 1.2s and 3.0s for Epic-Kitchens. Note that we conducted a similar pooling ablation between mean- and max-pooling on Epic-Kitchens, where we observed a 1.3% increase with max-pooling.

5.5.5.2 Recent and Spanning Representations

In our ablations, unless otherwise stated, an ensemble of 3 spanning scales $K = \{10, 15, 20\}$ and 3 recent starting points $i = \{t-10, t-20, t-30\}$ are used.

Table 5.7 **(a)** compares single starting points for the recent snippet features versus an ensemble. With a single starting point, points too near to and too far from the current time decrease the performance. The worst individual result is with $i_4 = 0$, i.e., using the entire sequence; the peak is at $i_2 = t - 20$, though an ensemble is still best. In Table 5.7 **(b)**, we show the influence of spanning snippet scales. These scales determine the temporal snippet granularity; individually, results are not significantly different across the scales, but as we begin to aggregate an ensemble, the results improve. The ensemble with 4 scales is best

(a)	starting points, i	$\{i_1, i_2, i_3\}$	$i_1 = t - 10$	$i_2 = t - 20$	$i_3 = t - 30$	$i_4 = 0$		
	Acc.	40.1	36.9	37.7	37.2	35.1		
(b)	spanning scales, K	{5,10,15,20}	{10,15,20}	{10, 15}	{5}	{10}	{15}	{20}
	Acc.	40.2	40.1	39.0	37.4	38.0	37.5	37.4
(c)	recent scales, K_R	1	3	5	10			
	Acc.	38.7	39.5	40.1	38.6			

Table 5.7: Ablations on the influence of recent and spanning representations on the Breakfast Actions dataset, reported are accuracies. The highlighted cells in each row are the same model trained with the parameters reported in Table 5.2.

but only marginally better than 3, at the expense of a larger network, so we choose $K = \{10, 15, 20\}$. In Table 5.7 **(c)**, we show the influence of recent snippet scales, we find $K_R = 5$ performs best.

5.5.5.3 Block Ablations

We report our results for ablations on our model blocks in Table 5.8.

Influence of	Changes in components	Acc.(Drop)
Non-Local Blocks (NLB)	replace all NLBs with concatenation + linear layer	33.7 (6.4%)
Coupling Blocks (CB)	only couple the $\mathbf{S}_K$ and $\mathbf{S}_K$ in CBs	35.1 (5.0%)
	only couple the $\mathbf{R}_i$ and $\mathbf{R}_i$ in CBs	34.2 (5.9%)
	replace CBs with concatenation + linear layer	33.4 (6.7%)
Temporal Aggregation Blocks (TAB)	a single CB is used in TABs	38.0 (2.1%)
	three CBs are used in a single TAB	37.7 (2.4%)
	a single a CB is used without any TABs	32.1 (8.0%)

Table 5.8: Ablations on the influence of different blocks in our model on the Breakfast Actions dataset, reported are accuracies.

Non-local Blocks: Previous studies on simple (single action, several seconds long duration) video understanding have shown the benefits of using features from both the recent and long-term past (Li et al., 2017; Wu et al., 2019). A naïve way to use both recent and long-term features is to simply concatenate the two. However, combining the two in a learned way, e.g., via attention, is superior (+6.4%). To incorporate attention, we apply non-local blocks (NLBs) (Wang et al., 2018c), which is an adaptation of the attention mechanism that is popularly used in machine translation.

Coupling Blocks: Different than the simple feature vector concatenations like in SlowFast (Feichtenhofer et al., 2019; Xiao et al., 2020), we couple the max-pooling outputs in our Coupling Blocks (CBs) which are very essential components of our model. When we replace our CBs with concatenation + linear layer, we observe a drop of 6.7%. When we do not use coupling but separately pass the $\mathbf{R}_i$ and $\mathbf{S}_K$ through concatenation + linear layer, we observe a drop of 7.5%. We find that coupling the recent $\mathbf{R}_i$ and long range $\mathbf{S}_K$ information is critical. Coupling only recent information (-5.9%) does not keep sufficient context,

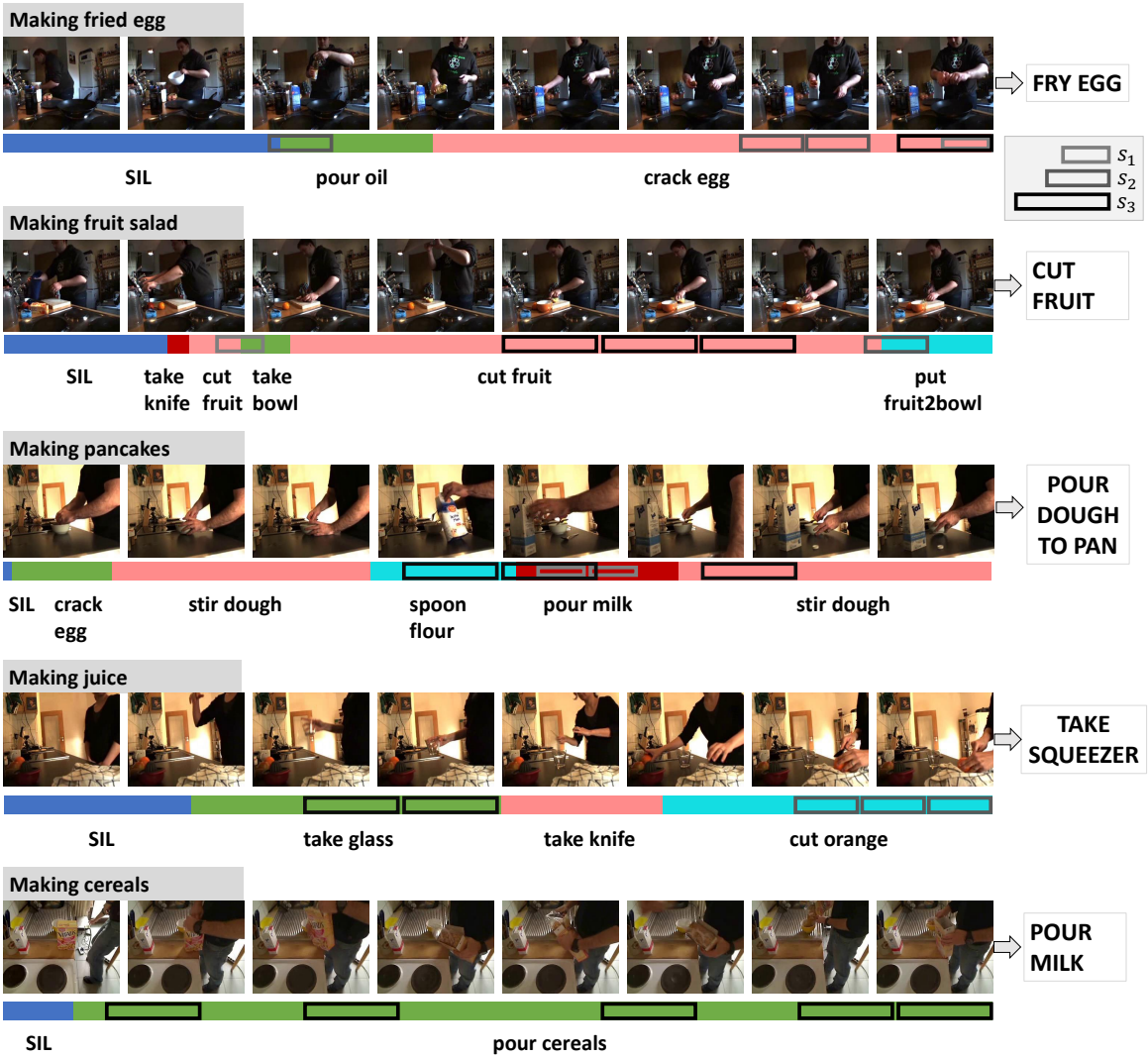


Figure 5.5: Attention visualization on the Breakfast Actions dataset for next action anticipation. Rectangles are the top 5 five spanning snippets (at different granularities where K = 10,15,20), weighted highest by the attention mechanism in the non-local blocks (NLB). Best viewed in color.

whereas coupling only long-range past (-5%) does not leave sufficient representation for the more relevant recent aspects.

The **Temporal Aggregation Blocks** (TAB) are the most critical components for our model. Omitting them and directly classifying a single CB's outputs significantly decrease accuracy (-8%). The strength of the TAB comes from using an ensemble of coupling blocks as input (single, -2.1%) and using the TABs in an ensemble (single, -2.4%).

5.5.5.4 Additional Ablations

When we omit the auxiliary complex activity prediction, i.e., removing the Z term from Equation 5.6 ("no Z"), we observe a slight performance drop of 1.1%. In our model we max pool over all $\mathbf{S}''_{i,K_1}, ..., \mathbf{S}''_{i,K_S}$ in our TABs. When we replace the max-pooling with concatenation + linear, we reach an accuracy of 37.4%. We also try to disentangle the ensemble effect from the use of multi-granular representations. When we fix the spanning past scales K to $\{15, 15, 15\}$ and all the starting points to $i = t - 20$, we observe a drop of 1.2% in accuracy which indicates the importance of our multi-scale representation. We also trained a simple attention based model that enables learning and selecting important frame features. For next action anticipation, this baseline drops the performance by 9% drop on Breakfast and 4.2% on EPIC. This validated the importance of our multi-granular pairing of the recent with long-range past.

Our method can encode long video durations while attending to salient snippets. In Figure 5.5, we present example visualizations of regions attended by our model when performing action anticipation. We show the five highest weighted spanning snippets (at different granularities). Our model attends different regions over the videos; for example, for predicting 'fry egg' when making fried eggs, it attends regions both when pouring oil and cracking eggs.

5.5.6 Anticipation on Procedural Activities: Breakfast & 50Salads

Method	Input	Segmentation Method & Feature	Accuracy
Vondrick et al. (2016)	FC7 features	-	8.1
Miech et al. (2019a)	R(2+1)D	-	32.3
RNN (Farha et al., 2018)	segmentation output	Richard et al. (2017), Fisher	**30.1**
CNN (Farha et al., 2018)	segmentation output	Richard et al. (2017), Fisher	27.0
ours "no Z"	Fisher	-	29.2
ours	Fisher	-	29.7
ours	I3D	-	40.1
ours "no Z"	I3D	-	39.0
ours	segmentation output	ours, I3D	43.1
ours	segmentation output + I3D	ours, I3D	**47.0**
ours	frame GT	-	64.7
ours	frame GT + I3D	-	63.1

Table 5.9: Performance of next action anticipation on the instructional Breakfast Actions dataset and comparisons against the state of the art, given different frame inputs: GT action labels, Fisher vectors, I3D features.

5.5.6.1 Next Action Anticipation

For this task, we predict the action that occurs 1 second from the current time t. We compare to the state of the art in Table 5.9 with two types of frame inputs: spatio-temporal features (Fisher vectors or I3D) and frame-wise action labels (either from ground truth or via a separate segmentation algorithm) on Breakfast actions. Compared to previous methods

Method	Input	Segmentation Method & Feature	Accuracy
Vondrick et al. (2016)	FC7 features	-	6.2
RNN (Farha et al., 2018)	segmentation output	Richard et al. (2017), Fisher	30.1
CNN (Farha et al., 2018)	segmentation output	Richard et al. (2017), Fisher	29.8
ours "no Z"	Fisher	-	**31.6**
ours	I3D		40.7
ours	frame GT	-	63.8

Table 5.10: Next action anticipation on the instructional 50Salads dataset and comparisons to the state of the art, given frame inputs as GT action labels, Fisher vectors, I3D features.

using only visual features as input, we outperform CNN features (Vondrick et al., 2016) (FC7 features (Krizhevsky et al., 2012)) and spatio-temporal features R(2+1)D (Miech et al., 2019a) by a large margin (+32.3% and +8.1%). While the inputs are different, R(2+1)D features were shown to be comparable to I3D features (Tran et al., 2018). Given that Miech et al. (2019a) uses only recent observations, we conclude that incorporating the spanning past into the prediction model is essential.

As can be expected, our method degrades when we replace I3D with the weaker Fisher vectors (40.1% vs. 29.7%). Nevertheless, this result is competitive with methods that use action labels (Farha et al., 2018) (30.1% with RNN) derived from segmentation algorithms (Richard et al., 2017) using Fisher vectors as input. For a fair comparison, we report a variant of our model without the complex activity prediction ("no Z"), which has a slight performance drop of 0.5% and 1.1% when Fisher vectors and I3D features are used respectively.

If we use action labels as inputs instead of visual features, our performance improves from 40.1% to 43.1%; merging labels, and visual features give another 4% boost to 47%. In this experiment, we use segmentation results from our own framework, adapted for the task of temporal action segmentation (see Section 5.5.9). However, if we substitute ground truth labels instead of segmentation labels, we still find a 21.6% gap. This suggests that the quality of the segmentation matters. When the segmentation is very accurate, then adding additional features does not help and actually slightly deteriorates results (see Table 5.10 "frame GT" vs. "frame GT + I3D"). Note that for using the ground truth and segmentation labels as inputs to our model, we represent each frame with a one-hot vector encoding. Each vector has a length of the total number of classes that all zero values except the index of the correct class.

In Table 5.10, we also report our results for 50Salads. Using Fisher vectors, we both outperform the state of the art (Richard et al., 2017) by 1.8% and the baseline with CNN features (Vondrick et al., 2016) by 25.4%. Similar to our observations on the Breakfast Actions, our method's performance increases when we replace Fisher vectors. We observe an accuracy of 23.1% when we use the ground truth labels instead of I3D features.

5.5.6.2 Dense Anticipation

Dense anticipation predicts frame-wise actions; accuracies are given for specific portions of the remaining video (Pred.) after observing a given percentage of the past (Obs.). Competing

Obs.	20%				30%			
Pred.	10%	20%	30%	50%	10%	20%	30%	50%
A	**Labels** (GT)							
RNN (Farha et al., 2018)	60.4	50.4	45.3	40.4	61.5	50.3	45.0	41.8
CNN (Farha et al., 2018)	58.0	49.1	44.0	39.3	60.3	50.1	45.2	40.5
Ke (Ke et al., 2019b)	64.5	**56.3**	**50.2**	**44.0**	66.0	55.9	**49.1**	**44.2**
ours	**65.5**	55.5	46.8	40.1	**67.4**	**56.1**	47.4	41.5
B	**Features** (Fisher)							
CNN (Farha et al., 2018)	12.8	11.6	11.2	10.3	17.7	16.9	15.5	14.1
ours	**15.6**	**13.1**	**12.1**	**11.1**	**19.5**	**17.0**	**15.6**	**15.1**
C	**Labels** (Fisher + Richard et al. (2017) (Frame-wise Acc. 36.8/42.9))							
RNN (Farha et al., 2018)	18.1	**17.2**	15.9	15.8	21.6	20.0	19.7	19.2
CNN (Farha et al., 2018)	17.9	16.4	15.4	14.5	22.4	20.1	19.7	18.8
Ke (Ke et al., 2019b)	18.4	**17.2**	16.4	**15.8**	22.8	**20.4**	19.6	**19.8**
ours	**18.8**	16.9	**16.5**	15.4	**23.0**	20.0	**19.9**	18.6
	Concatenate **B** and **C**							
ours	**25.0**	**21.9**	**20.5**	**18.1**	**23.0**	**20.5**	19.8	**19.8**
D	**Features** (I3D)							
ours	24.2	21.1	20.0	18.1	30.4	26.3	23.8	21.2
E	**Labels** (I3D + our seg.(Frame-wise Acc. 54.7/57.8))							
ours	**37.4**	31.2	30.0	26.1	39.5	34.1	31.0	**27.9**
	Concatenate **D** and **E**							
ours	37.1	**31.8**	**30.1**	**27.1**	**39.8**	**34.2**	**31.9**	27.8

Table 5.11: Dense anticipation mean over classes on the instructional Breakfast Actions dataset, given different frame inputs GT action labels, Fisher vectors, I3D features.

methods Farha et al. (2018) and Ke et al. (2019b) have two stages; they first apply temporal action segmentation and then use the segmentation outputs (Richard et al., 2017), i.e., frame-wise action labels, as inputs for anticipation. We experiment with both action labels and visual features.

For the Breakfast Actions dataset (Table 5.11), when using GT frame labels, we outperform the others, for shorter prediction horizons. For the 50Salads dataset (Table 5.12), we outperform the state of the art for the observed 20%, and our predictions are more accurate on long-range anticipation (Pred. 50%). We outperform Farha et al. (2018) when we use visual features as input (B Features (Fisher)). When using the segmentation outputs (from Richard et al. (2017), which has a frame-wise temporal segmentation accuracy of 36.8% and 42.9% for the observed 20% and 30% of video respectively), we are comparable to the state of the art (Ke et al., 2019b).

We further merge visual features with action labels for dense anticipation. With Fisher vectors and the frame labels obtained from Richard et al. (2017), we observe a considerable performance increase in performance compared to only using the frame labels (up to +7%) in the Breakfast Actions dataset. In 50Salads, this increase is not significant nor consistent. This may be due to the better performing segmentation algorithm on the 50Salads dataset (frame-wise accuracy of 66.8% and 66.7% for 20% and 30% observed respectively). We observe further improvements on Breakfast Actions once we substitute Fisher vectors with I3D features and segmentations from our own framework (I3D + ours seg.), which has a

Obs.	20%				30%			
Pred.	10%	20%	30%	50%	10%	20%	30%	50%
A	**Labels** (GT)							
RNN (Farha et al., 2018)	42.3	31.2	25.2	16.8	44.2	29.5	20.0	10.4
CNN (Farha et al., 2018)	36.1	27.6	21.4	15.5	37.4	24.8	20.8	14.1
Ke (Ke et al., 2019b)	45.1	33.2	27.6	17.3	**46.4**	**34.8**	**25.2**	13.8
ours	**47.2**	**34.6**	**30.5**	**19.1**	44.8	32.7	23.5	**15.3**
B	**Features** (Fisher)							
CNN (Farha et al., 2018)	-	-	-	-	-	-	-	-
ours	25.5	19.9	18.2	15.1	30.6	22.5	19.1	11.2
C	**Labels** (Fisher + Richard et al. (2017) (Frame-wise Acc. 66.8/66.7))							
RNN (Farha et al., 2018)	30.1	25.4	18.7	13.5	30.8	17.2	14.8	9.8
CNN (Farha et al., 2018)	21.2	19.0	16.0	9.9	29.1	20.1	17.5	10.9
Ke (Ke et al., 2019b)	32.5	**27.6**	21.3	**16.0**	**35.1**	**27.1**	22.1	15.6
ours	**32.7**	26.3	**21.9**	15.6	32.3	25.5	**22.7**	**17.1**
	Concatenate **B** and **C**							
ours	**34.7**	25.9	**23.7**	15.7	34.5	26.1	19.0	15.5

Table 5.12: Dense anticipation mean over classes on the instructional 50Salads dataset, given different frame inputs GT action labels, Fisher vectors.

frame-wise temporal segmentation accuracy of 54.7% and 57.8% for the observed 20% and 30% of video respectively. Similar to next action anticipation, our performance drops when using only visual features as input (I3D is better than Fisher vectors). When using I3D features and the frame label outputs of our segmentation method, we obtain our model's best performance, with a slight increase over using only frame label outputs.

In Figure 5.6, we provide qualitative results from our method for dense anticipation on the Breakfast Actions dataset. We show our method's predictions after observing 30% of the video. We compare our results when we use the GT labels and I3D features as input. It can be seen that using the GT labels are leading to better predictions.

5.5.7 How much spanning past is necessary?

We vary the duration of spanning snippets (Equation 5.3) with start time i as fractions of the current time t; $i = 0$ corresponds to the full sequence, i.e., 100% of the spanning past, while $i = t$ corresponds to none, i.e., using only recent snippets since the end points j remain fixed at t. Using the entire past is best for the Breakfast Actions dataset (Figure 5.7 left). Interestingly, this effect is not observed on Epic-Kitchens (Figure 5.7 right). Though we see a small gain by 1.2% until 40% past for the appearance features (rgb), beyond this, performance saturates.

We believe this has to do with the fine granularity of labels in Epic-Kitchens; given that the median action duration is only 1.9s, one could observe as many as 16 actions in 30 seconds. Given that the dataset has only 28.5K samples (excluding the test and validation sets) split over 2513 action classes, we speculate that the model cannot learn all the variants of long-term relationships beyond 30 seconds. Therefore, increasing the scope of the spanning past does not further increase the performance, see Figure 5.7 (right). Based on experiments

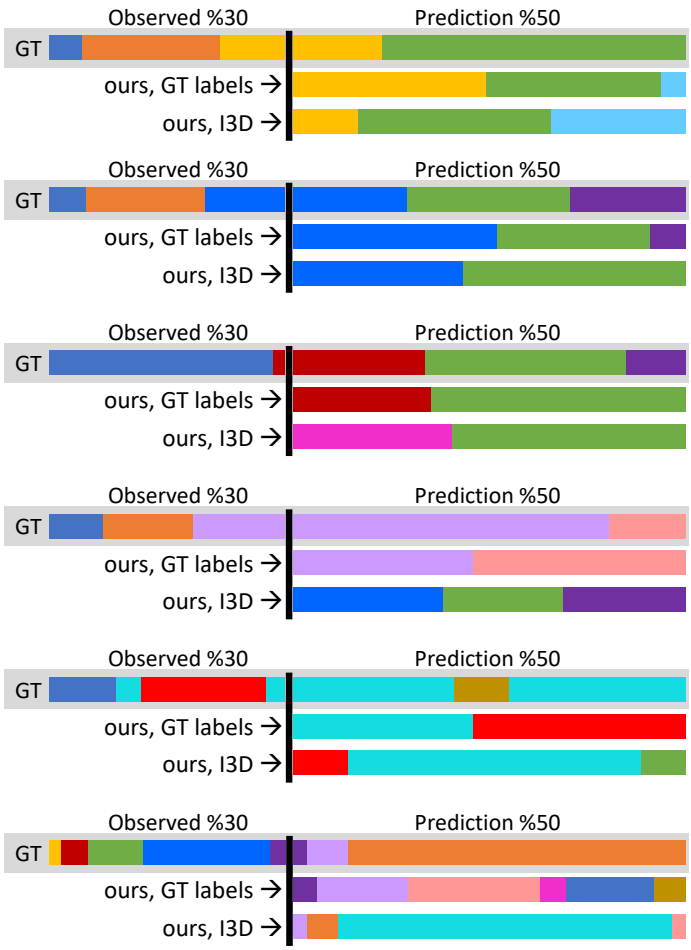


Figure 5.6: Qualitative results for dense anticipation on Breakfast Actions dataset when using the GT labels and I3D features. Best viewed in color.

on the validation set, we set the spanning scope to 6 seconds for Epic-Kitchens for the rest of the chapter. In Figure 5.7, we report our results for appearance (rgb), optical flow and object-based features and the late fusion of the predictions from these modalities. Overall the appearance and object-based features outperform the optical-flow features.

5.5.8 Anticipation and Recognition on Daily Activities: Epic-Kitchens

5.5.8.1 Action Anticipation

The **anticipation** task of Epic-Kitchens requires anticipating the future action $\tau_\alpha = 1$s before it starts. For fair comparison to the state of the art, (Furnari and Farinella, 2019), denoted by "RU", we use the same features (appearance, motion, and object) provided by the authors. We train our model separately for each feature modality with the same hyper-parameters; during inference, we apply a late fusion of the predictions from the different modalities by

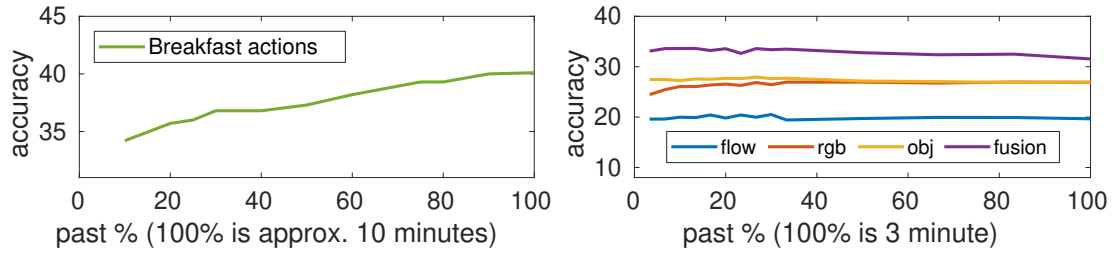


Figure 5.7: Effect of spanning scope on instructional vs. daily activities. For Epic-Kitchens (right) we report Top-5 accuracy on the validation set with rgb, optical-flow and object features and late fusion.

		Top-1 Accuracy (%)			Top-5 Accuracy (%)		
		Verb	Noun	Action	Verb	Noun	Action
S1	TSN (Damen et al., 2018)	31.8	16.2	6.0	76.6	42.2	28.2
	Miech et al. (2019a)	30.7	16.5	9.7	76.2	42.7	25.4
	RU (Furnari and Farinella, 2019)	33.0	22.8	14.4	79.6	50.9	33.7
	ours	31.4	22.6	16.4	75.2	47.2	**36.4**
	ours v+n	**37.9**	**24.1**	**16.6**	**79.7**	**54.0**	36.1
S2	TSN (Damen et al., 2018)	25.3	10.4	2.4	68.3	29.5	6.6
	Miech et al. (2019a)	28.4	12.4	7.2	69.8	32.2	19.3
	RU (Furnari and Farinella, 2019)	27.0	15.2	8.2	69.6	34.4	21.1
	ours	27.5	**16.6**	10.0	66.8	32.8	23.4
	ours v+n	**29.5**	16.5	**10.1**	**70.1**	**37.8**	**23.4**

Table 5.13: Action anticipation comparisons on Epic tests sets, seen (S1) and unseen (S2)

average voting. Note that for experiments on this dataset, we do not use the entire past for computing our spanning snippet features as increasing the scope of the spanning past does not further increase the performance (see Section 5.5.7).

We report our results for hold-out test data of Epic-Kitchens in Table 5.13 both for seen kitchens (S1) with the same environments as in the training data and unseen kitchens (S2) of held out environments. We outperform the temporal segment networks (TSN) (Damen et al., 2018) for both Top-1 and Top-5 accuracy by a great margin. Miech et al. (2019a) performs anticipation by combining two transitional models: one is based on recognizing the current action, and the other is based on visual attributes. We outperform this model in the Top-1 action accuracies by 6.7% and 2.8% on S1 and on S2, respectively. We also outperform the state of the art, RU (Furnari and Farinella, 2019), in the Top-1 and Top-5 action accuracies by 2% and 2.7% on S1 and by 1.8% and 2.3% on S2. When we add verb and noun classification to our model as auxiliary tasks to help with anticipation, "ours v+n", our performance improves for action and especially for noun and verb scores.

Besides predicting the next actions for $\tau_\alpha = 1$s, Furnari and Farinella (2019) also report prediction results at multiple anticipation times between 0.25s and 2s on Epic-Kitchens. We present our comparisons to this setting in Table 5.14 on the validation set. Our prediction scores are better than Furnari and Farinella (2019) for all time points, where our improvements are more significant when the anticipation time decreases.

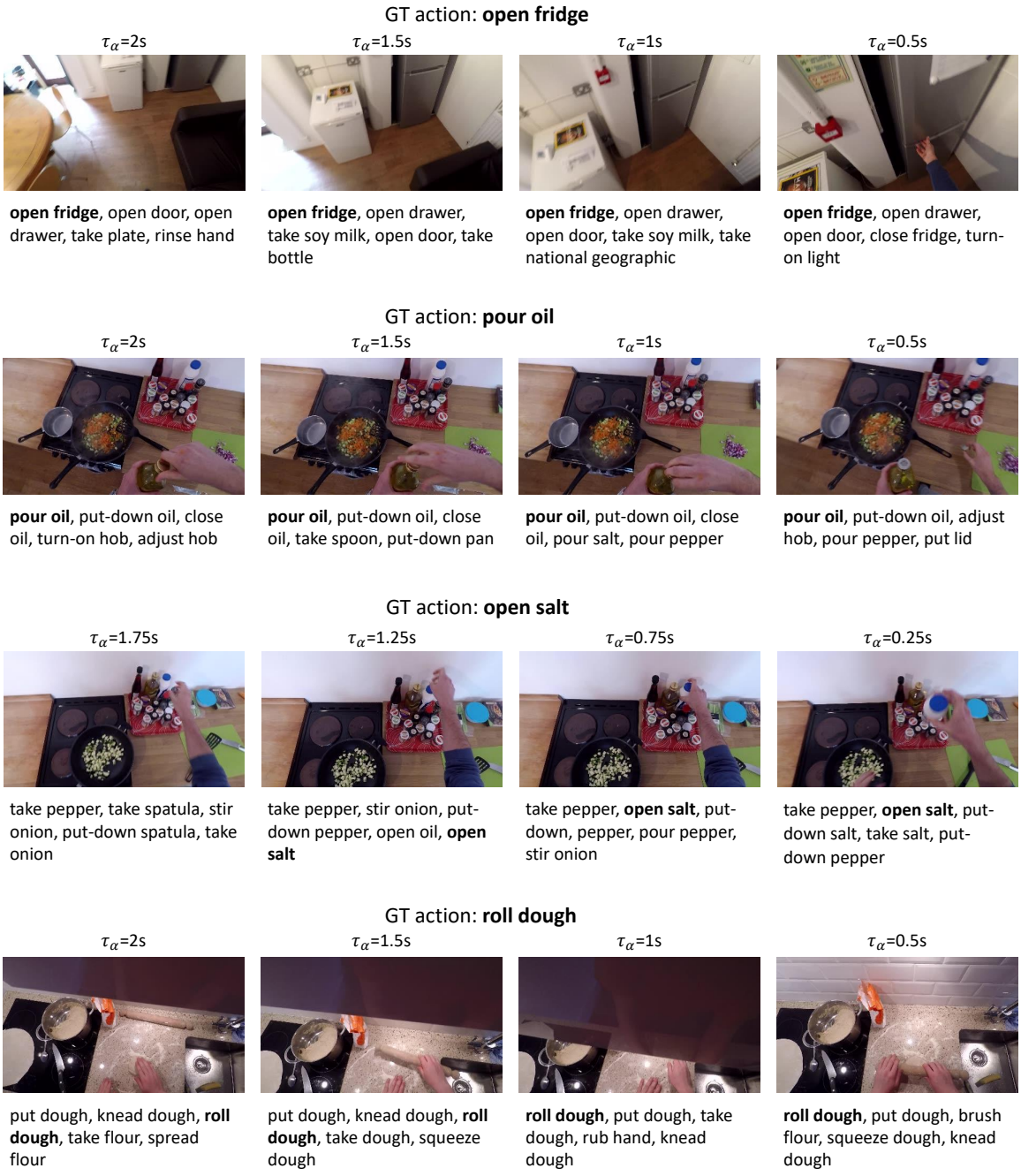


Figure 5.8: Exemplary qualitative results for next action anticipation on Epic-Kitchens dataset, showing the success of our method. We list our Top-5 predictions at different anticipation times, τ_α. The closer τ_α is to the next action, the better are our model's predictions. Best viewed in color.

	Top-5 Accuracy%							
τ_a	2s	1.75s	1.5s	1.25s	1.0s	0.75s	0.5s	0.25s
RU	29.4	30.7	32.2	33.4	35.3	36.3	37.4	39.0
ours	**30.9**	**31.8**	**33.7**	**35.1**	**36.4**	**37.2**	**39.5**	**41.3**

Table 5.14: Action anticipation on Epic-Kitchens validation set for different anticipation times τ_α compared to RU (Furnari and Farinella, 2019).

			Top-1 Accuracy (%)			Top-5 Accuracy (%)		
Set	Rank	Team	Verb	Noun	Action	Verb	Noun	Action
S1	1st	**NUS_CVML**	37.87	24.10	16.64	79.74	53.98	36.06
	2nd	VI-I2R	36.72	24.61	16.02	80.39	54.90	37.11
	3rd	Ego-OMG	32.20	24.90	16.02	77.42	50.24	34.53
	4th	UNIPD-UNICT	36.73	24.26	15.67	79.87	53.76	36.31
	5th	GT-WISC-MPI	36.25	23.83	15.42	79.15	51.98	34.29
S2	1st	Ego-OMG	27.42	17.65	11.81	68.59	37.93	23.76
	2nd	VI-I2R	28.71	17.21	10.11	71.77	40.49	23.46
	3rd	**NUS_CVML**	29.50	16.52	10.04	70.13	37.83	23.42
	4th	GT-WISC-MPI	29.87	16.80	9.94	71.77	38.96	23.69
	5th	UNIPD-UNICT	28.51	16.59	9.32	71.66	37.97	23.28

Table 5.15: Action anticipation results on Epic-Kitchens egocentric action anticipation challenge (2020) for seen (S1) and unseen (S2) test sets. Our submission (Team "NUS_CVML") is ranked first on S1 and third on S2 sets.

In Figure 5.8, we present qualitative results from our method for next action anticipation on the Epic-Kitchens dataset for multiple anticipation times τ_α between 0.25 and 2 seconds. We show examples where our method is certain about the next action for all different times. We also show examples where our method's prediction gets more accurate when the anticipation time is closer.

5.5.8.2 Action Anticipation Challenge

For **anticipation**, we follow the protocol of the Epic-Kitchens egocentric action anticipation challenge (2020) and participate in this challenge (Damen et al., 2020b). We report our results and our ranking for the hold-out test datasets for seen kitchens (S1) and unseen kitchens (S2) in Table 5.15. The official ranking on the challenge is based on the Top-1 action accuracy. Our submission (Team "NUS_CVML") is ranked first on S1 and third on S2.

In Table 5.15, team "VI-I2R", extends the rolling-unrolling LSTMs (Furnari and Farinella, 2019) by extracting better feature representations for appearance, optical-flow, and objects and adding a past action prediction module to improve future action prediction performance. Team "Ego-OMG"(Dessalene et al., 2020) localizes hand and object interaction regions and builds object manipulation graphs which are fed to an LSTM to model long-term sequences. This method performs very well on the unseen kitchens, which indicates the significance of focusing on object regions for action anticipation in new environments. "UNIPD-UNICT" (Camporese et al., 2020) utilizes knowledge distillation with using the

		Top-1 Accuracy%			Top-5 Accuracy%		
		Verb	Noun	Action	Verb	Noun	Action
S1	TSN (Damen et al., 2018)	48.2	36.7	20.5	84.1	62.3	39.8
	RU (Furnari and Farinella, 2019)	56.9	43.1	33.1	85.7	67.1	55.3
	LFB Wu et al. (2019)	60.0	45.0	32.7	88.4	71.8	55.3
	TBN (Kazakos et al., 2019)	64.8	46.0	34.8	**90.7**	71.3	56.7
	SlowFast + audio (Xiao et al., 2020)	65.7	46.4	35.9	89.5	71.7	57.8
	ours	63.2	49.5	41.3	87.3	70.0	63.5
	ours v+n	**66.7**	**49.6**	**41.6**	90.1	**77.0**	**64.1**
S2	TSN (Damen et al., 2018)	39.4	22.7	10.9	74.3	45.7	25.3
	RU (Furnari and Farinella, 2019)	43.7	26.8	19.5	73.3	48.3	37.2
	LFB Wu et al. (2019)	50.9	31.5	21.2	77.6	57.8	39.4
	TBN (Kazakos et al., 2019)	52.7	27.9	19.1	79.9	53.8	36.5
	SlowFast + audio (Xiao et al., 2020)	**55.8**	32.7	24.0	**81.7**	58.9	43.2
	ours	52.0	31.5	26.2	76.8	52.7	45.7
	ours v+n	54.6	**33.5**	**27.0**	80.4	**61.0**	**46.4**

Table 5.16: Action recognition comparisons on Epic tests sets, seen (S1) and unseen (S2)

rolling-unrolling LSTMs architecture (Furnari and Farinella, 2019) where a teacher network provides semantic prior information through glove-based word embeddings (Pennington et al., 2014) and sequence transition probabilities. Team "GT-UWISC-MPI" proposes a method composed of three modules that allow for joint prediction of hand motions, interaction regions, and future actions.

To conclude, these teams show that utilizing hand-object interactions and regions is essential for action anticipation, especially when the environment is unseen, where the appearance features tend to perform poorly compared to seen environments. The influence of hand-object regions on anticipation could be evaluated in detail on the newly released extension of Epic-Kitchens (Damen et al., 2020a), where hands and interacting objects are localized using bounding boxes which are made publicly available. Moreover, using additional information sources such as word embeddings could further be evaluated, considering the long-tailed actions distribution in EPIC.

5.5.8.3 Action Recognition

For **recognition**, we follow the protocol of the Epic-Kitchens Action Recognition Challenge to classify pre-trimmed action segments. For this task, we adjust the scope of our spanning and recent snippets according to the action start and end times t_s and t_e. Spanning snippet features are computed on a range of $[t_s - 6, t_e + 6]$; the first recent snippet scope is fixed to $[t_s, t_e]$ and the rest to $[t_s - 1, t_e + 1], [t_s - 2, t_e + 2]$ and $[t_s - 3, t_e + 3]$. Remaining hyper-parameters are kept the same.

In Table 5.16, we compare our results to the state-of-the-art methods. We show that our model outperforms the other recognition methods, including the state of the art SlowFast networks with audio data (Xiao et al., 2020) (+5.4% on S1, +2.2% on S2 for Top-1 Accuracy). TBN (Kazakos et al., 2019) utilizes RGB, flow, and audio, along with an approach for temporal binding of modalities based on temporal segment networks. Both TBN (Kazakos et al.,

	Verb Acc.%	Noun Acc.%	Action Acc.%
ours verb & ours noun	63.09	49.15	35.95
ours action	60.53	47.33	41.21

Table 5.17: We compare our model when we directly predict the actions to a variant of our model where we predict the verbs and nouns separately on the validation set of Epic-Kitchens, reported are Top-1 accuracies.

2019) and the SlowFast network by Xiao et al. (2020) use audio data. We outperform both of these works without audio and think that including audio modality will further improve our model's performance. We also outperform Long-term Feature Banks (Wu et al., 2019), which also uses non-local blocks (+8.6% on S1, +5% on S2 for Top-1). We show that our method outperforms Furnari and Farinella (2019) by approximately +7% on both S1 and S2. Together with the anticipation results from the previous paragraph, we can conclude that our method generalizes to both anticipation and recognition tasks and is able to achieve state-of-the-art results on both, while Furnari and Farinella (2019) performs very well on anticipation but poorly on recognition. Similar to our anticipation results, when we add verb and noun classification to our model, "ours v+n", we observe improvements in our action, verb, and noun scores.

Epic-Kitchens has a long-tailed action distribution, which means "head" classes have a large number of samples, and "tail" classes have a small number of samples per class. Moreover, in Epic-Kitchens, there are action classes in the test set which do not have any training instances. Therefore several approaches, TBN (Kazakos et al., 2019), SlowFast (Xiao et al., 2020), and LFB (Wu et al., 2019), are trained for predicting verbs and nouns separately, and the outputs of the verb and noun models are later combined for action recognition.

Like Furnari and Farinella (2019), we directly predict actions and marginalize the action scores to get verb and noun scores. We compute a variant of our model where we separately predict the verbs and nouns on the validation set of Epic-Kitchens. We present our comparisons to our model when we directly predict the action on the validation in Table 5.17. When predicting verbs and nouns separately, our verb accuracy improves by 3.5%, and noun accuracy improves by 1.9%. However, the action accuracy drops to 36.0% while predicting directly the actions has an accuracy of 40.9%. We believe this is due to the verbs and nouns not being independent. Similar observations were made by Furnari et al. (2018).

5.5.8.4 Action Recognition Challenge

We participated in the Epic-Kitchens egocentric action recognition challenge (2020). We report our ranking and comparisons in Table 5.18 for seen kitchens (S1) and unseen kitchens (S2). The official ranking on the challenge is based on the Top-1 action accuracy. Our submission (Team "NUS_CVML") is ranked second on S1 and third on S2.

In table 5.18, team "UTS-Baidu" presents the best performing model on both the seen and unseen kitchens (Wang et al., 2020d). For top-K detected object bounding boxes obtained from an object detector, they extract features from pre-trained flow and appearance-based

			Top-1 Accuracy%			Top-5 Accuracy%		
Set	Rank	Team	Verb	Noun	Action	Verb	Noun	Action
S1	1st	UTS-Baidu	70.41	52.85	42.57	90.78	76.62	63.55
	2nd	**NUS_CVML**	66.70	49.55	41.59	90.10	76.98	64.11
	3rd	SAIC-Cambridge	69.43	49.71	40.00	91.23	73.18	60.53
S2	1st	UTS-Baidu	60.43	37.28	27.96	83.07	63.67	46.81
	2nd	GT-WISC-MPI	60.02	37.49	27.38	82.55	63.47	45.10
	3rd	**NUS_CVML**	54.56	33.46	26.97	80.40	60.98	46.43

Table 5.18: Action recognition results on Epic-Kitchens egocentric action recognition challenge (2020) for seen (S1) and unseen (S2). Our submission (Team "NUS_CVML") is ranked second on S1 and third on S2 sets.

models and use an attention mechanism to enable interactions between them to select the most action-relevant features for classifying nouns and verbs. Team "SAIC-Cambridge" proposes a spatio-temporal attention network (Perez-Rua et al., 2020) aiming at focusing on action related regions. Motivated by the relations of gaze and actions for egocentric vision (Li et al., 2018b), team "GT-WISC-MPI" proposes utilizing gaze as a probabilistic distribution of attention in the context of an action. They perform well on the unseen kitchens indicating the importance of focusing on the action regions for better feature representations.

5.5.8.5 Influence of Modality

We use appearance (rgb), motion (flow), and object-based features provided by Furnari and Farinella (2019) for our anticipation and recognition experiments. In Table 5.19, we report our results, "ours", separately for anticipation and recognition for different modalities. Overall the most important features are appearance-based features for both anticipation and recognition. Object-based features seem to be very important for anticipation, while flow does not significantly influence anticipation, 11.6% vs. 7.7%. For action recognition, object and flow-based features perform similarly. When we apply late fusion of the predictions from all these modalities by average voting, we get a 3% accuracy improvement over the best performing modality in anticipation and a 7.8% improvement in recognition.

In Eq. 5.6, we show that we can add complex activity classification, Z, as an auxiliary task. Epic-Kitchens (Damen et al., 2018) videos are recorded with 32 participants in 32 different kitchens. We can use the kitchen prediction as an auxiliary task similar to using complex activity classification, Z. We report our results, "ours + kitchen ", in Table 5.19. We observe slight improvements for both anticipation and recognition.

	Anticipation				Recognition			
	obj	flow	rgb	obj+flow+rgb	obj	flow	rgb	obj+flow+rgb
ours	11.6	7.7	12.5	15.5	28.6	28.1	33.5	41.2
ours + kitchen	11.5	8.0	12.9	**15.8**	28.7	28.3	33.4	**41.4**

Table 5.19: Influence of different modalities, experimented on the validation set of Epic-Kitchens for anticipation and recognition. Reported are Top-1 accuracies.

# samples s	# classes	%validation instances	Accuracy (rgb)	Accuracy (flow)	Accuracy (obj)	Accuracy (rgb+flow+obj)	Accuracy GT nouns	Accuracy GT verbs
$s > 350$	2	4.2	79.2	82.5	67.3	**87.7**	82.5	46.9
$300 < s \leq 350$	5	6.5	64.6	63.0	51.9	**77.0**	74.2	47.2
$250 < s \leq 300$	5	6.2	49.2	52.1	34.4	63.0	**71.1**	54.7
$200 < s \leq 250$	7	5.9	66.7	64.3	53.7	**73.5**	60.9	47.4
$150 < s \leq 200$	10	7.3	42.0	43.4	43.7	56.1	**68.0**	32.3
$100 < s \leq 150$	16	7.4	39.6	39.6	34.7	55.7	**72.7**	13.4
$s > 100$	45	37.5	54.7	55.2	46.0	67.1	**71.0**	38.9
$75 < s \leq 100$	11	4.0	50.5	37.9	47.0	61.6	**68.2**	17.2
$50 < s \leq 75$	29	6.7	39.6	20.7	36.7	51.4	**64.3**	17.1
$s > 50$	85	48.1	52.3	49.0	44.8	64.5	69.8	**34.1**
$40 < s \leq 50$	23	4.2	32.2	17.3	25.5	39.0	**56.3**	14.4
$30 < s \leq 40$	38	5.1	28.9	14.2	23.7	31.6	**54.9**	16.2
$20 < s \leq 30$	77	8.1	24.9	16.5	25.2	33.3	**48.2**	11.6
$15 < s \leq 20$	78	5.7	17.8	12.6	17.1	22.7	**41.6**	10.5
$10 < s \leq 15$	94	4.7	15.0	7.3	13.3	21.8	**29.9**	11.1
$s > 10$	395	76.0	41.8	36.1	36.2	51.7	**61.2**	26.2
$5 < s \leq 10$	231	6.1	14.6	4.3	6.6	13.3	**28.5**	7.0
$1 < s \leq 5$	787	8.1	10.4	5.2	8.2	12.9	**19.1**	7.2
$s = 1$	878	2.9	2.1	0.7	1.4	2.1	**14.8**	3.5
$s = 0$	222	7.0	0.0	0.0	0.0	0.0	0.0	0.0
$s > 0$	2513	100	33.5	28.1	28.6	41.2	**50.2**	21.0

Table 5.20: Our accuracy reported on different groups of classes on the validation set (Furnari and Farinella, 2019) of Epic-Kitchens for recognition. Each row starts with the range of the number of instances in the training set. For example, in the first row, we are interested in the classes with more than 350 cases in the training set. There are two such classes. These classes comprise 4.2% of the validation instances. We report the average Top-1 action accuracy over subsets of the validation set, which include only the instances of the classes that fall into the range reported in the "# samples s" column. For example, in the first row, the average is calculated only over the instances in the validations set that belong to the two largest classes, containing more than 350 instances of the training set.

5.5.8.6 Epic-Kitchens – A Long-Tailed Action Distribution

We compute our model's performance on action recognition as the mean accuracy over all the instances in the validation set in Table 5.19. Since the Epic-Kitchens dataset has a long-tailed action distribution, we further explore our model's performance on different action groups in Table 5.20. We first divide the validation set into several groups of classes based on the number of instances in the training set. For example, in the first row of Table 5.20, we show the two most common classes, namely "open door" and "open drawer", which each include more than 350 instances in the training set. These two classes comprise 4.2% of the validation instances. Similarly, we have five classes, which are "take plate", "take spoon", "put plate", "open fridge", "close door", that have between 300 and 350 instances in the training set. These five classes comprise 6.5% of the validation instances.

We report our results for six different feature types / combinations in Table 5.20. In each

row, we average our model's accuracy over the subset of instances in the validation set that only includes the classes in the "# classes s" column. We report our accuracies for appearance, motion and object-based features separately, in columns "Accuracy(rgb)", "Accuracy(flow)", "Accuracy(obj)" respectively. We report our late fusion results in "Accuracy(rgb+flow+obj)". We also use ground truth noun and verb labels to represent each frame using a one-hot vector encoding. The results are reported in columns "Accuracy GT nouns" and "Accuracy GT verbs".

Late fusion improves average accuracy over the entire validation by 7.7% over using rgb, by 13.1% over flow, and by 12.6% over object-based features. When we use ground truth nouns to form frame-level features, we get an improvement of 9% over using late fusion. Using the verb labels to form frame-level features performs poorly, even worse than flow-based action recognition accuracy (7.1% decrease).

Our model with GT nouns outperforms our late fusion based model for every range but for "$s > 350$" (5.2% drop), "$300 < s \geq 350$" (2.8% drop) and with the largest gap at range "$200 < s \geq 250$" (12.6% drop) where there are several actions sharing the same object such as "turn off tap", "close tap", "put down pan", "wash pan". This shows that when there is enough data to train a model, visual features will perform better and should be preferred. The object-based features achieve half of the performance compared to using GT nouns, indicating the challenges of training a state-of-the-art object detector on fine-grained datasets.

Our model's performance degrades towards the classes with fewer and fewer examples. Especially for classes with less than 50 instances in the training set, our performance drops by more than 10% when using late fusion. For classes with only one instance, one-shot classes, our late fusion-based model's performance drops to 2.1%. When we use GT nouns to form our frame-level features, we observe an increase of 12.7% accuracy. This shows that object labels are more useful when the training data is scarce, as nouns provide high-level discriminative information. Using the GT verbs to form our frame-level features does not improve accuracy. For zero-shot classes, our performance is zero for all our models, as we directly predict actions, and these classes are simply unknown to our models.

One-shot and zero-shot classes comprise 10% of the validation dataset. Tail classes comprise 24% of the dataset (classes with less than ten examples in the training set). Possible solutions could be utilizing models that work on long-tail distributions, using complementary data in the form of text, or retrieving similar actions from different datasets or the web to increase the amount of the training data. For the newly collected Epic-Kitchens extension (Damen et al., 2020a), the authors compute two types of accuracies, calculated over all classes vs. tail classes, which contain either a tail verb class or a tail noun class. These new evaluations might motivate future research to explore long-tail distributions (Kang et al., 2020) or generalized few-shot-learning methods (Schonfeld et al., 2019) on Epic-Kitchens.

5.5.8.7 Epic-Kitchens - Anticipation Using GT Labels

We also experimented using ground truth nouns, verbs, and action labels on the Epic-Kitchens dataset for next action anticipation. We report our results in Table 5.21. Our model's upper bound is 17.9 % when using the ground truth action label, "GT actions", for forming the

obj+flow+rgb	GT nouns	GT verbs	GT actions
15.5	14.0	5.8	17.9

Table 5.21: The influence of using ground truth labels for action anticipation, evaluated on the validation set of Epic-Kitchens. Reported are Top-1 accuracies.

frame-wise feature representations. Using verb labels, "GT verbs", reduces the accuracy significantly, by 12.1%. Using only the noun labels, "GT nouns" decreases the accuracy by 3.9%, supports our previous findings that object-based representations are more important than verbs for action anticipation.

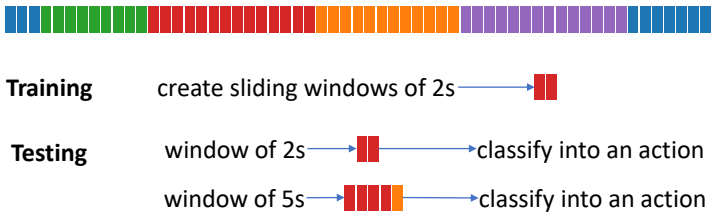
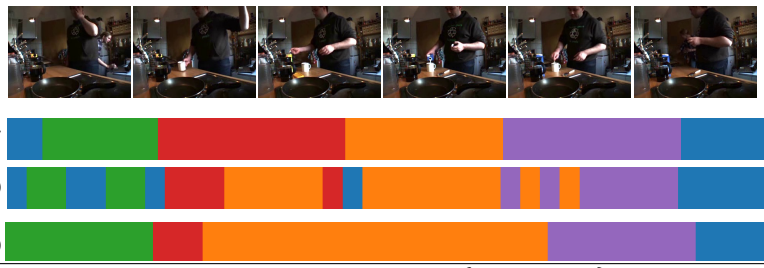


Figure 5.9: For temporal action segmentation, the task is predicting frame-level action labels in videos. We use a sliding window of 2 seconds to generate multiple windows. We train our model to learn labelling each window with an action. We test our model with 2 and 5-second windows separately.

5.5.9 Temporal Action Segmentation

Here the task is predicting frame-level action labels of complex activity videos. We use a sliding window of 2 seconds and generate multiple windows with start and end times, t_s and t_e. Our goal is to classify each window into an action. For this task we only change the end time of our recent and spanning snippets. We use start and end point ranges $[t_s, t_e], [t_s - 2, t_e + 2], [t_s - 5, t_e + 5]$ for computing the recent, and the entire video for computing the spanning snippet features. The rest of the parameters are kept the same. We compare our performance against the state of the art, MS-TCN (I3D) (Farha and Gall, 2019b), in Table 5.22 on Breakfast actions. We test our model with 2s and 5s windows (see Figure 5.9).

We report the frame-wise accuracy (Acc), segmental edit distance and F1 scores at overlapping thresholds of 10%, 25% and 50%. In the example sequences and also in the F1 scores and edit distances in Table 5.22, we observe more fragmentation in our segmentation for 2s than for 5s windows. However, for 2s, our model produces better accuracies, as the 5s windows are smoothing the predictions at action boundaries. We perform comparably to the state of the art in terms of accuracies and we outperform it in the F1 scores when we use 5s windows for testing. Additionally we provide our model's upper bound, "ours (I3D) pre-trimmed action segments", for which we classify GT action segments instead of sliding windows. The results indicate that there is room for improvement, which we leave as future work. We show that we are able to easily adjust our method from its main application and



	F1@{10, 25, 50}			Edit	Acc.
MS-TCN (I3D) (Farha and Gall, 2019b)	52.6	48.1	37.9	**61.7**	**66.3**
ours (I3D) 2s	52.3	46.5	34.8	51.3	65.3
ours (I3D) 5s	**59.2**	**53.9**	**39.5**	54.5	64.5
ours (I3D) pre-trimmed action segments	-	-	-	-	**75.9**

Table 5.22: Exemplary temporal action segmentation and comparison to the state of the art on the Breakfast Actions dataset.

already get close to the state of the art, with slight modifications. Currently, we segment naively by classifying snippets aggregated with long-range information. Our segmentations will likely improve with stronger sequence models.

5.6 Discussion & Conclusion

In this chapter, we presented a temporal aggregate model for long-range video understanding. Our method computes recent and spanning representations pooled from snippets of video that are related via coupled attention mechanisms. Validating on three complex activity datasets, we show that temporal aggregates are either comparable or outperform the state of the art on three video understanding tasks: action anticipation, recognition and temporal video segmentation.

In developing our framework, we faced questions regarding temporal extent, scaling, and level of semantic abstraction. Our experiments show that max-pooling is a simple and efficient yet effective way of representing video snippets; this is the case even for snippets as long as two minutes. For learning temporal relationships in long video, attention mechanisms relating the present to long range context can successfully model and anticipate upcoming actions. The extent of context that is beneficial, however, may depend on the nature of activity (instructional vs. daily) and label granularity (coarse vs. fine) of the dataset.

We found significant advantages to using ensembles of multiple scales, both in recent and spanning snippets. Our aggregates model is flexible and can take as input either visual features or frame-wise action labels. We achieve competitive performance with either form of input, though our experiments confirm that higher levels of abstraction such as labels are more preferable for anticipation. Nevertheless, there is still a large gap between what can be anticipated with inputs from current segmentation algorithms in comparison to ground truth labels, leaving room for segmentation algorithms to improve.

Chapter 6

Conclusion

Contents

6.1 Overview

Let us imagine a future where we have robot assistants capable of recognizing what we are doing and predicting our future actions to help us getting things done without expecting to take any orders. The state-of-the-art technologies might not be there yet, but they have been improved over time, with the intention to be good enough that machines will be able to work alongside humans.

This dissertation deals with complex activity understanding in videos. Such activities are rich sources for understanding humans' daily lives and developing algorithms to make our lives easier. They could be used in a wide variety of ambitious problem settings relavant to smart systems to assist humans. Despite the difficulties of working with long-range untrimmed videos, there has been a growing interest in working on them in recent years (Rohrbach et al., 2012; Kuehne et al., 2014b; Sener et al., 2015; Malmaud et al., 2015; Alayrac et al., 2016; Huang et al., 2016; Richard and Gall, 2016; Farha et al., 2018; Ke et al., 2019b; Damen et al., 2020a). We present a detailed overview of complex-activity-associated tasks in Chapter 2, Background.

In this dissertation, we developed models and techniques for understanding human activities in complex activity videos and verified our solutions on the two challenging tasks of temporal action segmentation (Sener and Yao, 2018; Sener et al., 2020) and action anticipation (Sener and Yao, 2019; Sener et al., 2020). In Chapter 3, we presented our first work, where we investigated temporal action segmentation in complex activity videos. In contrast to fully supervised approaches, which rely on temporal action annotations, we addressed

unsupervised learning from easily accessible online videos. In Chapter 4, we presented the details of our second work where we challenged ourselves with the new task of zero-shot action anticipation. In Chapter 5, we presented a generic framework for learning long-range video representations which can be employed for action anticipation and temporal action segmentation.

In this chapter we conclude this dissertation. The remainder of this chapter is organized as follows. We first present a summary of our works and contributions. We then conclude this dissertation with a broad review of the challenges of developing algorithms for understanding complex activities along with perspectives for future research.

6.2 Summary of Contributions

The overall goal of this dissertation was to develop novel approaches to model complex activities under different supervision levels, on different datasets and for different tasks. Towards this goal we investigated several challenges related to working with such activities and presented our contributions in Chapters 3, 4, 5. In the following, we summarize our contributions to the complex activity domain individually with respect to our solutions. We mainly focused on developing novel temporal models, using less supervision, utilizing auxiliary data, measuring the influence of visual representations, and investigating the role of natural language in complex videos.

The potential reward for developing algorithms requiring little or no supervision becomes more prominent as the amount of online data grows rapidly. Even though there have been several works on unsupervised learning from complex activity videos (Sener et al., 2015; Alayrac et al., 2016), they assume videos with textual data in the form of narrations. It is common that people talk when teaching or instructing. However, the alignment between the spoken and visual content is not always guaranteed. In Chapter 3, we presented a method which does not require any text, and automatically segments videos based on visual data only. Following our work, there has been recent interest in unsupervised temporal action segmentation in videos in the vision community (Goel and Brunskill, 2018; Kukleva et al., 2019; Aakur and Sarkar, 2019; Mounir et al., 2020).

In computer vision, the majority of complex activity datasets are composed of multiple demonstrations of the same activities. However, this does not match human experience, as humans do not need many demonstrations to learn new tasks and are experts at transferring previous experience. Moreover, relying on multiple demonstrations does not scale well to all complex activities, and data collection gets costly. Reducing the number of demonstrations needed for individual tasks is also known to be important for teaching robots using a few demonstrations (Finn et al., 2017). In Chapter 4, we considered the transfer of knowledge as the ability to make predictions on tasks that we have never seen before, similar to how humans perform. We learned the structure of instructions from text data in an unsupervised way and transferred it to the visual domain to anticipate the future actions on unseen videos. Transferring the knowledge to the video domain is supervised and requires a small-scale video dataset annotated with temporal labels. Our work shows promising results for a challenging

task that, without transferring knowledge, would otherwise require collecting orders of magnitude more video data and thus a much more costly labelling process. With our zero-shot problem description on complex activity videos, we believe that we started a new challenge for the vision community.

In this dissertation, we proposed three temporal models for understanding human activities in untrimmed complex activity videos. In Chapter 3, we separated the unsupervised temporal action segmentation problem into two components of discriminative learning of sub-activity representations and their temporal modelling. We generatively modeled the temporal structure of sub-activities using a generalized Mallows model (Fligner and Verducci, 1986), which models the distribution over sub-activity permutations. Since our algorithm is unsupervised, we start with some initial label assignment to video frames. We then alternate between discriminatively learning the appearance of sub-activities and generatively modelling their temporal structure. We showed that our method is flexible enough to allow missing steps and variations in ordering and is able to produce coherent segments. In Chapter 4, we presented an action anticipation framework for learning the structure of instructions from text and transferring it to video data. In particular, we worked on cooking recipes where one can find a vast amount of instructional data both in the text and video domain. We introduced a hierarchical system that is composed of four recurrent neural networks (RNN), where two RNNs encode the sentences and videos, one generates sentences, and another RNN captures the sequential structure of the instructional activity. Our hierarchical model allowed us to separate the procedural learning from the visual appearance learning and predict coherent and plausible future instructions. Finally, in Chapter 5, we proposed a novel way to learn representations of long videos and showed its strength through successful experiments on several tasks, including action anticipation and temporal action segmentation. Our main novelty is our multi-granular pairing of recent observations with long-range past/future observations. We modelled the relationships between recent observations and long-range context using simple techniques such as max-pooling and attention. Our engineering efforts, when interpreted with extensive experiments, give insight for long-range video understanding, including topics of temporal extent and level of semantic abstraction. We showed that our model can be used for multiple tasks with minimal modifications and does not pose any constraints on the type of activity, i.e., it can be used for daily and instructional activities alike. We believe that our method's success is a meaningful contribution that will spawn more research in the future.

To improve the robustness of our temporal models to visual representations, we presented several solutions. In Chapter 3, we proposed a discriminative appearance model and an extension of our model to handle the frames which are irrelevant to the complex activities. In this work, we separated the modelling of appearance and their temporal relations. We discriminatively modelled the appearance of sub-activities via clustering, where we group frames according to their semantic similarity with respect to their sub-activity rather than purely visual features. We also showed that explicitly handling the irrelevant frames within the model improves the performance. In Chapter 5, we experimented with different types of visual representations such as appearance or flow-based features as well as higher-level abstractions such as object labels. We showed that an ensemble of such complementary

features yields improvements.

Different from related works (Farha et al., 2018; Damen et al., 2018; Ke et al., 2019b), in Chapter 4, we did not make category-based predictions but instead predict the future in the form of sentences. Our motivation for using sentences is to decrease the load for annotating the video datasets and adding richness to our predictions, as sentences convey information about the objects, their states, and the actions. Additionally, as humans are the end-users of such systems, it is natural to present the results of such systems as sentences for smoother interactions. We collected a new video dataset annotated with sentences where we have a single demonstration of around 2.500 different cooking recipes. We think that our publicly available dataset will be of interest to those working in anticipation, complex activity recognition, and video captioning. We also presented various ways of evaluating our generated sentences, including using human raters.

6.3 Discussion and Outlook

Human activities exhibit various potential research directions to explore, ranging from action recognition, detection, anticipation, retrieval, captioning, to temporal action segmentation. In this dissertation, we closely worked on long-range complex activity videos and proposed solutions for temporal action segmentation and action anticipation from videos. Throughout our works, we faced many challenges and introduced novel solutions.

In the following, we present an overview of the challenges of developing vision-based solutions for complex activities, along with our contributions and future directions. First, we start with an analysis of the complex activity datasets. Next, we highlight the importance of temporal modelling in complex videos. We then review feature representations for videos. Finally, we conclude with an overview of annotations and evaluations on complex activity datasets.

6.3.1 Datasets

There are several frequently used standardized complex activity video datasets. We split them into instructional and daily activities. For the instructional activities (Kuehne et al., 2014b; Zhukov et al., 2019) the actors usually have a well-defined task and an end goal, and the sub-activities follow some *loose* order. Daily activities (Damen et al., 2018), on the other hand, are composed of multiple actions that are following some *partial* order, as certain sub-activities can be executed independently of others. In the following, we compare such datasets based on how they are collected and annotated, as well as their limitations.

- Although there are several *recorded* datasets (Kuehne et al., 2014b; Damen et al., 2018), recently, the trend is to *collect* videos from online resources such as YouTube. This is due to the large number and variety of videos on such platforms (Malmaud et al., 2015; Zhou et al., 2018b; Zhukov et al., 2019; Tang et al., 2019; Sener and Yao, 2019). However, collected datasets have limitations regarding the applicability to every task.

For example, action anticipation and augmented reality-related scenarios might require dedicated, realistic recording settings, as collected videos tend to be edited, e.g., with fast-forwarding, annotated frames, or changing viewpoints. Among the recorded datasets, the Epic-Kitchens (Damen et al., 2018, 2020a) is the most recent and largest daily activity dataset. For instructional activities, the largest recorded dataset, the Breakfast Actions dataset (Kuehne et al., 2014b), was created in 2014, and is much smaller than Epic-Kitchens.

- Recorded datasets have either third-person or egocentric views. Recently, egocentric datasets gained attention. The Epic-Kitchens (Damen et al., 2018, 2020a) is the largest dataset in egocentric vision that captures untrimmed daily activities. There are also several instructional activity datasets with egocentric view (Fathi et al., 2011b; Li et al., 2018b), but on a much smaller scale. Activity understanding research can benefit from datasets with multi-view recordings, including both third-person and egocentric views. The Breakfast Actions dataset (Kuehne et al., 2014b) contains multi-view recordings. However, all the views are from a third-person perspective. Egocentric views do not capture the physical environment as a well as a third-person view would. The third-person view, on the other hand, might not be as good as the egocentric view in capturing objects, tools and hand-object interaction spots, which are essential ingredients for recognizing fine-grained actions correctly.

- The diversity in complex activity datasets is limited. Although the collected datasets recently started including diverse domains (Tang et al., 2019), almost all the recorded datasets with a few exceptions (Toyer et al., 2017) cover only cooking activities.

6.3.2 Temporal Modelling

Action recognition from short videos has seen a considerable boost with the use of deep architectures (Simonyan and Zisserman, 2014a; Wang et al., 2016a; Carreira and Zisserman, 2017; Feichtenhofer et al., 2019; Lin et al., 2019a). However, complex activity videos tend to be long, ranging from minutes to hours. They exhibit arbitrarily-long temporal durations depending on the type of activity (Kuehne et al., 2014b; Zhou et al., 2018b; Damen et al., 2018). As a result, it becomes challenging to process an entire complex activity video at once. Therefore, taking state-of-the-art action recognition models that usually work on seconds-long videos and applying them directly to complex videos becomes challenging.
We consider the challenges of modelling complex activity videos as the following:

- Complex activities are *untrimmed* sequences where multiple sub-activities take place over time. Most importantly, these sub-activities are not independent but are related to each other through sequence dynamics. This makes temporal modelling more critical for understanding complex activities compared to simple action recognition (Huang et al., 2018b). A variety of such models are already utilized for modelling complex activities, including context-free grammars (Kuehne et al., 2014b), hidden Markov models (Lea et al., 2016; Elhamifar and Naing, 2019), recurrent neural networks (Singh et al., 2016;

Farha et al., 2018; Sener and Yao, 2019), Mallows model (Sener and Yao, 2018) and temporal convolutions (Lea et al., 2017; Farha and Gall, 2019b). Temporal models should capture long and short-range temporal dependencies, account for variable orderings, missing or repeating sub-activities in videos. Ideally, the the developed models should be generic and applicable to both daily and instructional videos, which is a lacking feature in recent approaches, as the majority of them assume that the data will be in the form of instructional videos (Sener et al., 2015; Alayrac et al., 2016; Huang et al., 2016; Sener and Yao, 2018; Elhamifar and Naing, 2019).

- There is a large amount of video data available on the internet. However, collecting and annotating video data can be very costly and time-consuming. Therefore, particularly for video data, unsupervised methods are preferred over supervised ones. Similarly, research directions toward weakly supervised and transfer learning-based solutions are favored.

- In complex activity videos, a considerable portion of the data might be composed of irrelevant content, i.e., frames that cannot be classified into any action class of interest. Therefore, models should be supplied with additional mechanisms to handle such noise (Sener and Yao, 2018; Elhamifar and Naing, 2019).

- Standard action recognition datasets (Soomro et al., 2012; Kay et al., 2017) have usually balanced action class distributions. However, complex activities might contain long-tailed distributions (Damen et al., 2018). This is because the number of category-based annotations grows arbitrarily with the number of new complex activities, as more and more objects will be in interaction. For example, there are 10 different complex activities and 52 action classes in the Breakfast Actions dataset (Kuehne et al., 2014a) while there are 432 different complex sequences with in total 2513 action classes in Epic-Kitchens (Damen et al., 2018). There is a large body of work on long-tailed (Kang et al., 2020; Zhou et al., 2020) and few-shot learning (Schonfeld et al., 2019) that the complex activity research can benefit from for modelling actions.

6.3.3 Representation

When working with videos their representation is a fundamental question. Dut to the long-range videos, so far, the research on complex activities (Kuehne et al., 2014b; Richard et al., 2017; Farha and Gall, 2019b) is focused on representing video frames with standard image or video-based features, both handcrafted (Wang and Schmid, 2013) and deep learning-based (He et al., 2016; Carreira and Zisserman, 2017), and only then applying temporal models on top of these representations.
When representing videos, we need to account for the following aspects:

- First of all, we are faced with all typical computer vision challenges, like significant variations in camera motion, changes in viewpoint, illumination conditions, or background. We need features that are robust to all these variations. Additionally, there might be considerable appearance variations between videos of the same complex activity due

to diverse scenery. In contrast, appearance changes between different sub-activities in the same video might be very subtle. Ideally, a good representation should reflect similarities between the same sub-activities in different videos and provide discriminative cues for different sub-activities in the same video. Recently several works have been proposed to learn strong representations for video data by using joint embedding spaces between text and video (Miech et al., 2019b, 2020) and cross modal learning (Sun et al., 2019a). These works can be utilized for obtaining robust representations.

- Besides appearance-based video features, we could use auxiliary modalities to get more discriminative video representations from motion, audio, text, and high-level features such as object labels. Motion-based features might better represent the actions at different speeds (Simonyan and Zisserman, 2014a). Although it might fail at distinguishing fine-details, audio can contribute discriminative cues for actions such as "frying" or "washing". Text obtained through narrations (Alayrac et al., 2016) or additional sources (Malmaud et al., 2015) can be a helpful cue, as it is high-level and directly related to action labels. However, it should be noted that the performance gain is largely governed by the quality of the alignment between text and video. Since complex activities are composed of fine-grained and mostly object-centric actions, explicitly detecting objects (Ren et al., 2015) in videos and using object-based features can provide discriminative information.

6.3.4 Labelling

Complex activity videos are annotated with the temporal boundaries and the types of their sub-activities. In the following we present differences among annotations in existing datasets:

- Depending on the task of interest, temporal annotations in datasets can exists at different granularities. For example, in one dataset (Kuehne et al., 2014b), a sequence labeled with "pour oil" might be annotated in another dataset (Damen et al., 2018) with five sub-activities of "taking oil bottle", "opening the bottle", "pouring oil", "closing the oil lid" and "placing it in the cupboard". The labelling granularity affects the temporal extent of actions. The duration of the same action may vary significantly across different datasets. Such differences make the generalizability of models to different datasets very challenging.

- Although the majority of datasets in the literature are presented with category-based labelling (Kuehne et al., 2014b; Damen et al., 2018; Tang et al., 2019), there are several datasets with sentence descriptions (Malmaud et al., 2015; Zhou et al., 2018b), including our Tasty Videos dataset (Sener and Yao, 2019). On the one hand, natural language descriptions of activities convey a lot more information than simple category-based labels and allow for better understanding of actions and objects (Lin et al., 2015; Zhou et al., 2018b). On the other hand, modelling language and evaluating it is an open research topic itself. One direction when annotating datasets could be following the recent Epic-Kitchens (Damen et al., 2020a) dataset, which provides both narrations and category-based annotations.

6.3.5 Evaluation

The mode of evaluation in complex activity related tasks might differ from standard metrics used in recognition.

- Usually, for complex activity related tasks such as temporal action segmentation or action detection, the labels are predicted per frame. As such, many methods use metrics such as frame accuracy (Kuehne et al., 2014b) or mean average precision (Caba Heilbron et al., 2015) to assess the quality of the results. However, as a new but fast-growing topic without well established evaluation methods, action anticipation is still difficult to evaluate accurately. Currently, the standard is to evaluate using conventional recognition metrics such as accuracy and recall (Farha et al., 2018; Furnari and Farinella, 2019). However, these do not allow assessing the *plausibility* of outcomes, which is necessary, as the future is not always fixed and might lead to alternative actions. Although it might be very costly, one solution could be annotating the datasets with multiple possible future labels.

- For evaluating sentence-based predictions, standard challenges in natural language processing remain (Vedantam et al., 2015; Lopez, 2008). Although there are widely used sentence evaluation metrics (Papineni et al., 2002; Banerjee and Lavie, 2005), human evaluations are still considered the gold standard.

- Existing works are mainly focused on improving their accuracy and thus are only focused on their correctly predicted outcomes. An interesting future direction could be to more closely investigate negative outcomes (Negative, 2018) to better understand potential shortcomings of temporal methods and problems of complex activity datasets.

Bibliography

Sathyanarayanan N Aakur and Sudeep Sarkar. A perceptual prediction framework for self supervised event segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1197–1206, 2019. 37, 38, 148

Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016. viii, ix, xv, 4, 6, 19, 21, 24, 31, 36, 40, 41, 51, 53, 71, 72, 73, 74, 75, 76, 77, 82, 83, 147, 148, 152, 153

Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging lstms to anticipate actions very early. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 280–289, 2017. 25

Javad Amirian, Jean-Bernard Hayet, and Julien Pettré. Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 23

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*, pages 5803–5812, 2017. 47

Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 18

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pages 892–900, 2016. 46

Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016a. 84

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016b. 121

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005. 90, 154

Fabien Baradel, Natalia Neverova, Christian Wolf, Julien Mille, and Greg Mori. Object level visual reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 105–121, 2018. 18

Jernej Barbič, Alla Safonova, Jia-Yu Pan, Christos Faloutsos, Jessica K Hodgins, and Nancy S Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface 2004*, pages 185–194. Citeseer, 2004. 37

Federico Becattini, Tiberio Uricchio, Lorenzo Seidenari, Lamberto Ballan, and Alberto Del Bimbo. Am i done? predicting action progress in videos. *arXiv preprint arXiv:1705.01781*, 2017. 4, 26

Michael Beetz, Ulrich Klank, Ingo Kresse, Alexis Maldonado, Lorenz Mösenlechner, Dejan Pangercic, Thomas Rühr, and Moritz Tenorth. Robotic roommates making pancakes. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 529–536. IEEE, 2011. 44, 80

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1171–1179, 2015. 88

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Smooth loss functions for deep top-k classification. *arXiv preprint arXiv:1802.07595*, 2018. 29

Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 35, 48, 54, 77, 78, 83

Piotr Bojanowski, Rémi Lajugie, Edouard Grave, Francis Bach, Ivan Laptev, Jean Ponce, and Cordelia Schmid. Weakly-supervised alignment of video with text. In *Proceedings of the IEEE international conference on computer vision*, pages 4462–4470, 2015. 36, 48

Antoine Bosselut, Asli Celikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. Discourse-aware neural rewards for coherent text generation. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 173–184, 2018. 44, 84

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-recurrent neural networks. *arXiv preprint arXiv:1611.01576*, 2016. 28

Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2911–2920, 2017. 39

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–970, 2015. xv, 19, 23, 24, 25, 39, 47, 118, 154

Guglielmo Camporese, Pasquale Coscia, Antonino Furnari, Giovanni Maria Farinella, and Lamberto Ballan. Knowledge distillation for action anticipation via label smoothing. *arXiv preprint arXiv:2004.07711*, 2020. 29, 139

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6299–6308, 2017. 4, 14, 16, 17, 42, 78, 117, 118, 119, 124, 125, 151, 152

Micael Carvalho, Rémi Cadène, David Picard, Laure Soulier, Nicolas Thome, and Matthieu Cord. Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 35–44, 2018. 45, 84

Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7608–7617, 2019. 23

Alexandros André Chaaraoui, Pau Climent-Pérez, and Francisco Flórez-Revuelta. A review on vision techniques applied to human behaviour analysis for ambient-assisted living. *Expert Systems with Applications*, 39(12):10873–10888, 2012. 2

Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Niebles. D3tw: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3546–3555, 2019. 35, 48

Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018. 39

Harr Chen, SRK Branavan, Regina Barzilay, David R Karger, et al. Content modeling using latent permutations. *Journal of Artificial Intelligence Research*, 36(1):129–163, 2009. 52, 62

Junwen Chen, Wentao Bao, and Yu Kong. Group activity prediction with sequential relational anticipation model. In *In European Conference on Computer Vision (ECCV)*, 2020a. 26

Min-Hung Chen, Baopu Li, Yingze Bao, Ghassan AlRegib, and Zsolt Kira. Action segmentation with joint self-supervised temporal domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9454–9463, 2020b. 34

Shaoxiang Chen and Yu-Gang Jiang. Semantic proposal for activity localization in videos via sentence query. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8199–8206, 2019. 47

Shaoxiang Chen and Yu-Gang Jiang. Hierarchical visual-textual graph for temporal activity localization via language. In *In European Conference on Computer Vision (ECCV)*, 2020. 48

Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Yan Shuicheng, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 433–442, 2019. 18

W. Cheng and Eyke Hüllermeier. A simple instance-based approach to multilabel classification using the mallows model. In *In Proceedings MLD-2009 1st International Workshop on Learning from Multi-Label Data, Bled, Slovenia*, page 28–38, 2009. 60

Yu Cheng, Quanfu Fan, Sharath Pankanti, and Alok Choudhary. Temporal sequence modeling for video event detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2227–2234, 2014. 31, 32

Guilhem Chéron, Ivan Laptev, and Cordelia Schmid. P-cnn: Pose-based cnn features for action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3218–3226, 2015. 15

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014. 42, 43, 44, 84, 92

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*, 2018. 43

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680, 2017. 43, 84, 86, 129

Enric Corona, Albert Pumarola, Guillem Alenya, and Francesc Moreno-Noguer. Context-aware human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6992–7001, 2020. 23

Nieves Crasto, Philippe Weinzaepfel, Karteek Alahari, and Cordelia Schmid. Mars: Motion-augmented rgb stream for action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7882–7891, 2019. 16

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005. 12

Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *European conference on computer vision*, pages 428–441. Springer, 2006. 12

Robert Dale. Cooking up referring expressions. In *Proceedings of the 27th Annual Meeting on Association for Computational Linguistics (ACL)*, 1989. 83

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. xv, 15, 19, 22, 24, 82, 84, 88, 90, 116, 117, 118, 124, 125, 137, 140, 142, 150, 151, 152, 153

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020a. xv, 16, 22, 24, 40, 140, 144, 147, 151, 153

Dima Damen, Evangelos Kazakos, Jian Ma Will Price, and Hazel Doughty. Epic-kitchens-55 - 2020 challenges report. `https://epic-kitchens.github.io/Reports/EPIC-KITCHENS-Challenges-2020-Report.pdf`, 2020b. 139

Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641, 2013. 46

Roeland De Geest, Efstratios Gavves, Amir Ghodrati, Zhenyang Li, Cees Snoek, and Tinne Tuytelaars. Online action detection. In *European Conference on Computer Vision*, pages 269–284. Springer, 2016. 26

Angela D'Elia and Domenico Piccolo. A mixture model for preferences data analysis. *Computational Statistics & Data Analysis*, 49(3):917–934, 2005. ISSN 0167-9473. 60

Eadom Dessalene, Michael Maynord, Chinmaya Devaraj, Cornelia Fermuller, and Yiannis Aloimonos. Egocentric object manipulation graphs. *arXiv preprint arXiv:2006.03201*, 2020. 29, 139

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 43, 46, 106

Li Ding and Chenliang Xu. Tricornet: A hybrid temporal convolutional and recurrent network for video action segmentation. *arXiv preprint arXiv:1705.07818*, 2017. 33

Li Ding and Chenliang Xu. Weakly-supervised action segmentation with iterative soft boundary assignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 6508–6516, 2018. 35, 115, 117

Pelin Dogan, Boyang Li, Leonid Sigal, and Markus Gross. A neural multi-sequence alignment technique (neumatch). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8749–8758, 2018. 48

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2625–2634, 2015. 17, 42, 117, 118, 129

Jianfeng Dong, Xirong Li, Chaoxi Xu, Shouling Ji, Yuan He, Gang Yang, and Xun Wang. Dual encoding for zero-example video retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9346–9355, 2019. 45

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 43

Hazel Doughty, Walterio Mayol-Cuevas, and Dima Damen. The pros and cons: Rank-aware temporal attention for skill determination in long videos. In *CVPR*, 2019. 4

Xuguang Duan, Wenbing Huang, Chuang Gan, Jingdong Wang, Wenwu Zhu, and Junzhou Huang. Weakly supervised dense event captioning in videos. In *Advances in Neural Information Processing Systems*, pages 3059–3069, 2018. 47

Olivier Duchenne, Ivan Laptev, Josef Sivic, Francis Bach, and Jean Ponce. Automatic annotation of human actions in video. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1491–1498. IEEE, 2009. 36

Kiana Ehsani, Hessam Bagherinezhad, Joseph Redmon, Roozbeh Mottaghi, and Ali Farhadi. Who let the dogs out? modeling dog behavior from visual data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4051–4060, 2018. 23

Ehsan Elhamifar and Dat Huynh. Self-supervised multi-task procedure learning from instructional videos. In *In European Conference on Computer Vision (ECCV)*, 2020. 41

Ehsan Elhamifar and Zwe Naing. Unsupervised procedure learning via joint dynamic summarization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6341–6350, 2019. 41, 151, 152

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990. 42

Hyunjun Eun, Jinyoung Moon, Jongyoul Park, Chanho Jung, and Changick Kim. Learning to discriminate information for online action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 809–818, 2020. 26

Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improving visual-semantic embeddings with hard negatives. *arXiv preprint arXiv:1707.05612*, 2017. 45

Chenyou Fan, Jangwon Lee, and Michael S Ryoo. Forecasting hands and objects in future frames. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 23

Mohamed Farah and Daniel Vanderpooten. An outranking approach for rank aggregation in information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 591–598, 2007. 60

Yazan Abu Farha and Juergen Gall. Uncertainty-aware anticipation of activities. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1197–1204. IEEE, 2019a. 30

Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3575–3584, 2019b. 4, 14, 33, 34, 115, 145, 146, 152

Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what? Anticipating temporal occurrences of activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018. 4, 30, 31, 82, 83, 115, 117, 123, 124, 125, 126, 127, 132, 133, 134, 135, 147, 150, 152, 154

Yazan Abu Farha, Qiuhong Ke, Bernt Schiele, and Juergen Gall. Long-term anticipation of activities with cycle consistency. In *German Conference on Pattern Recognition*, 2020. 30

Alireza Fathi and James M Rehg. Modeling actions through state changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2586, 2013. 31, 32

Alireza Fathi, Ali Farhadi, and James M Rehg. Understanding egocentric activities. In *2011 international conference on computer vision*, pages 407–414. IEEE, 2011a. 32

Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011b. xv, 19, 20, 24, 31, 151

Mohsen Fayyaz and Jurgen Gall. Sct: Set constrained temporal transformer for set supervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–510, 2020. 36

Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016. 15

Christoph Feichtenhofer, Axel Pinz, and Richard P Wildes. Spatiotemporal multiplier networks for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4768–4777, 2017. 15, 18

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6202–6211, 2019. 4, 16, 40, 118, 129, 130, 151

Panna Felsen, Pulkit Agrawal, and Jitendra Malik. What will happen next? forecasting player moves in sports videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3342–3351, 2017. 23, 118

Basura Fernando, Efstratios Gavves, Jose M Oramas, Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 50, 53

Basura Fernando, Efstratios Gavves, José Oramas, Amir Ghodrati, and Tinne Tuytelaars. Rank pooling for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):773–787, 2016. 17

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning*, pages 357–368, 2017. 80, 148

J.L. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971. 106

Michael A Fligner and Joseph S Verducci. Distance based ranking models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 359–369, 1986. 7, 51, 60, 149

Michael A Fligner and Joseph S Verducci. Posterior probabilities for a consensus ordering. *Psychometrika*, pages 359–369, 1990. 62

Emily B Fox, Michael C Hughes, Erik B Sudderth, Michael I Jordan, et al. Joint modeling of multiple time series via the beta process with application to motion capture segmentation. *The Annals of Applied Statistics*, 8(3):1281–1313, 2014. 36, 37, 54

Lea Frermann, Ivan Titov, and Manfred Pinkal. A hierarchical bayesian model for unsupervised induction of script knowledge. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 49–57, 2014. 52

Daniel Fried, Jean-Baptiste Alayrac, Phil Blunsom, Chris Dyer, Stephen Clark, and Aida Nematzadeh. Learning to segment actions from observation and narration. In *Proceedings*

of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2569–2588, 2020. 36, 38

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013. 45

Antonino Furnari and Giovanni Maria Farinella. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. xvii, 4, 14, 27, 28, 29, 42, 117, 125, 126, 136, 137, 139, 140, 141, 142, 143, 154

Antonino Furnari, Sebastiano Battiato, Kristen Grauman, and Giovanni Maria Farinella. Next-active-object prediction from egocentric videos. *J. of Visual Communication and Image Representation*, 49:401–411, 2017. 23

Antonino Furnari, Sebastiano Battiato, and Giovanni Maria Farinella. Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 29, 141

Adrien Gaidon, Zaid Harchaoui, and Cordelia Schmid. Temporal localization of actions with actoms. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2782–2795, 2013. 39

Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Forecasting future action sequences with neural memory networks. In *Proceedings of the 30th British Machine Vision Conference 2019, BMVC 201*, 2019a. 30, 31

Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Predicting the future: A jointly learnt model for action anticipation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5562–5571, 2019b. 25, 26

Harshala Gammulle, Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. Coupled generative adversarial network for continuous fine-grained action segmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 200–209. IEEE, 2019c. 33

Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. Semantic compositional networks for visual captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5630–5639, 2017. 46

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275, 2017a. 47

Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE international conference on computer vision*, pages 3628–3636, 2017b. 39

Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Red: Reinforced encoder-decoder networks for action anticipation. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017c. 28

Mingfei Gao, Mingze Xu, Larry S Davis, Richard Socher, and Caiming Xiong. Startnet: Online detection of action start in untrimmed videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5542–5551, 2019. 4

Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 971–980, 2017. 119

Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 759–768, 2015. 15

Karan Goel and Emma Brunskill. Learning procedural abstractions and evaluating discrete latent temporal structure. In *International Conference on Learning Representations*, 2018. 38, 148

Guoqiang Gong, Xinghan Wang, Yadong Mu, and Qi Tian. Learning temporal co-attention models for unsupervised video action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9819–9828, 2020. 40

A. Gorban, H. Idrees, Y.-G. Jiang, A. Roshan Zamir, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://www.thumos.info/`, 2015. xv, 19, 23, 24, 39

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006. 35

Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *PNAS*, 101(suppl 1):5228–5235, 2004. 59, 67

Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José MF Moura. Adversarial geometry-aware human motion prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 786–803, 2018. 23

Helena H. Lee, Ke Shu, Palakorn Achananuparp, Philips Kokoh Prasetyo, Yue Liu, Ee-Peng Lim, and Lav R Varshney. Recipegpt: Generative pre-training based cooking recipe generation and evaluation system. In *Companion Proceedings of the Web Conference 2020*, pages 181–184, 2020. 44

Meera Hahn, Nataniel Ruiz, Jean-Baptiste Alayrac, Ivan Laptev, and James M Rehg. Learning to localize and align fine-grained actions to sparse instructions. *arXiv preprint arXiv:1809.08381*, 2018. 48

Kristian J. Hammond. Chef: A model of case-based planning. In *AAAI Conference on Artificial Intelligence*, 1986. 83

Zaid Harchaoui, Félicien Vallet, Alexandre Lung-Yut-Fong, and Olivier Cappé. A regularized kernel-based approach to unsupervised audio segmentation. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1665–1668. IEEE, 2009. 37

Dongliang He, Xiang Zhao, Jizhou Huang, Fu Li, Xiao Liu, and Shilei Wen. Read, watch, and move: Reinforcement learning for temporally grounding natural language descriptions in videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8393–8400, 2019. 47

Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In *CVPR*, pages 5353–5360, 2015. 17

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 13

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016. 13, 87, 152

Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Beyond action recognition: Action completion in rgb-d data. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016. 26

Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Action completion: A temporal model for moment detection. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018. 26

Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Weakly-supervised completion moment detection using temporal attention. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 26

Fabian Caba Heilbron, Wayner Barrios, Victor Escorcia, and Bernard Ghanem. Scc: Semantic context cascade for efficient action detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3175–3184. IEEE, 2017. 39

Samitha Herath, Mehrtash Harandi, and Fatih Porikli. Going deeper into action recognition: A survey. *Image and vision computing*, 60:4–21, 2017. 4

Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7134–7143, 2019. 23

Luis Herranz, Weiqing Min, and Shuqiang Jiang. Food recognition and recipe analysis: integrating visual content, context and external knowledge. *arXiv preprint arXiv:1801.07239*, 2018. 84

Jack Hessel, Bo Pang, Zhenhai Zhu, and Radu Soricut. A case study on combining asr and visual features for generating instructional video captions. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 419–429, 2019. 47

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pages 1367–1377, 2016. 84

Minh Hoai and Fernando De la Torre. Max-margin early event detectors. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2863–2870. IEEE, 2012. 25, 26

Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107(2):191–202, 2014. 83, 117

Minh Hoai, Zhen-Zhong Lan, and Fernando De la Torre. Joint segmentation and classification of human actions in video. In *CVPR 2011*, pages 3265–3272. IEEE, 2011. 31, 32

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997. 42

David C. Hogg. Model-based vision: a program to see a walking person. *Image Vis. Comput.*, 1:5–20, 1983. 4

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. *arXiv preprint arXiv:1805.06087*, 2018. 43

Rui Hou, Rahul Sukthankar, and Mubarak Shah. Real-time temporal action localization in untrimmed videos by sub-action discovery. In *BMVC*, volume 2, page 7, 2017. 40

De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 14, 35, 48, 54, 77, 78, 83, 115, 147, 152

De-An Huang, Joseph J Lim, Li Fei-Fei, and Juan Carlos Niebles. Unsupervised visual-linguistic reference resolution in instructional videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2183–2192, 2017. 48

De-An Huang, Shyamal Buch, Lucio Dery, Animesh Garg, Li Fei-Fei, and Juan Carlos Niebles. Finding it. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5948–5957, 2018a. 48

De-An Huang, Vignesh Ramanathan, Dhruv Mahajan, Lorenzo Torresani, Manohar Paluri, Li Fei-Fei, and Juan Carlos Niebles. What makes a video a video: Analyzing temporal information in video understanding models and datasets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7366–7375, 2018b. 6, 17, 115, 118, 151

Yifei Huang, Yusuke Sugano, and Yoichi Sato. Improving action segmentation via graph-based temporal reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14024–14034, 2020. 34

Noureldien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Timeception for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 254–263, 2019. xvi, 128

Noureldien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Pic: Permutation invariant convolution for recognizing long-range activities. *arXiv preprint arXiv:2003.08275*, 2020. xvi, 128

Instructables. Instructables. `https://www.instructables.com/`, 2005. 81

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 13

Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating over-segmentation errors by detecting action boundaries. *arXiv preprint arXiv:2007.06866*, 2020. 34

Mihir Jain, Jan Van Gemert, Hervé Jégou, Patrick Bouthemy, and Cees GM Snoek. Action localization with tubelets from motion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 740–747, 2014. 39

Mayoore Jaiswal, Frank Liu, Anupama Jagannathan, Anne Gattiker, Inseok Hwang, Jinho Lee, Matthew Tong, Sahil Dureja, Soham Shah, Peter Hofstee, et al. Video-text compliance: Activity verification based on natural language instructions. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 4

Jermsak Jermsurawong and Nizar Habash. Predicting the structure of cooking recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 781–786, 2015. 44

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1): 221–231, 2012. 16

Xiaojie Jin, Huaxin Xiao, Xiaohui Shen, Jimei Yang, Zhe Lin, Yunpeng Chen, Zequn Jie, Jiashi Feng, and Shuicheng Yan. Predicting scene parsing and motion dynamics in the future. In *Advances in Neural Information Processing Systems*, pages 6915–6924, 2017. 23

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations*, 2020. 144, 152

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015. 45

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1725–1732, 2014. 14, 15, 16, 42, 50

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 14, 117, 152

Evangelos Kazakos, Arsha Nagrani, Andrew Zisserman, and Dima Damen. Epic-fusion: Audio-visual temporal binding for egocentric action recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5492–5501, 2019. 15, 140, 141

Qiuhong Ke, Mohammed Bennamoun, Hossein Rahmani, Senjian An, Ferdous Sohel, and Farid Boussaid. Learning latent global network for skeleton-based action prediction. *IEEE Transactions on Image Processing*, 29:959–970, 2019a. 26

Qiuhong Ke, Mario Fritz, and Bernt Schiele. Time-conditioned action anticipation in one shot. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019b. 30, 31, 115, 117, 118, 134, 135, 147, 150

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 982–992, 2015. 44

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 329–339, 2016. 44, 84, 86, 92

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 88, 124

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 44, 45, 48

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, page 3294–3302, 2015. 43, 45, 48, 84, 92

Nick Kline and Richard T Snodgrass. Computing temporal aggregates. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 222–231. IEEE, 1995. 116

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007. 47

Yu Kong, Zhiqiang Tao, and Yun Fu. Deep sequential context networks for action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1481, 2017. 26

Yu Kong, Shangqian Gao, Bin Sun, and Yun Fu. Action prediction from videos via memorizing hard-to-predict samples. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 25, 117

Hema Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. In *Robotics: Science and Systems (RSS)*, 2013a. doi: 10.15607/RSS.2013.IX.006. 27

Hema Koppula and Ashutosh Saxena. Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation. In *International conference on machine learning*, pages 792–800, 2013b. 27

Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):14–29, 2015. 27, 80, 83, 117

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017. 47

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 12, 13, 133

Björn Krüger, Anna Vögele, Tobias Willig, Angela Yao, Reinhard Klein, and Andreas Weber. Efficient unsupervised temporal segmentation of motion data. *IEEE Transactions on Multimedia (TMM)*, 19(4):797–812, 2017. 54

H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2014a. 82, 84, 152

Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2014b. xv, 4, 19, 20, 24, 31, 32, 53, 71, 77, 78, 82, 83, 84, 115, 116, 118, 122, 124, 147, 150, 151, 152, 153, 154

Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8. IEEE, 2016. 32, 35, 71

Hilde Kuehne, Alexander Richard, and Juergen Gall. Weakly supervised learning of actions from transcripts. *Computer Vision and Image Understanding*, 163:78–89, 2017. 35

Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 12066–12074, 2019. 37, 38, 83, 148

Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 689–704. Springer, 2014. 27, 82, 83, 115

Ivan Laptev. On space-time interest points. *International journal of computer vision*, 64 (2-3):107–123, 2005. 12

Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 12

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014. 43

Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. Segmental spatiotemporal cnns for fine-grained action segmentation. In *European Conference on Computer Vision*, pages 36–52. Springer, 2016. 31, 32, 33, 151

Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017. 31, 33, 35, 115, 152

Guy Lebanon and Yi Mao. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9(10), 2008. 60

Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016. 43

Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018a. 23

Joonseok Lee, Walter Reade, Rahul Sukthankar, George Toderici, et al. The 2nd youtube-8m large-scale video understanding challenge. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018b. 119, 129

Peng Lei and Sinisa Todorovic. Temporal deformable residual networks for action segmentation in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6742–6751, 2018. 31, 33

Bastian Leibe, Aleš Leonardis, and Bernt Schiele. Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1-3):259–289, 2008. 25

Fu Li, Chuang Gan, Xiao Liu, Yunlong Bian, Xiang Long, Yandong Li, Zhichao Li, Jie Zhou, and Shilei Wen. Temporal modeling approaches for large-scale youtube-8m video understanding. *arXiv preprint arXiv:1707.04555*, 2017. 18, 118, 130

Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10820–10829, 2020. 36

Jun Li, Peng Lei, and Sinisa Todorovic. Weakly supervised energy-based learning for action segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6243–6251, 2019. 35

Kang Li, Jie Hu, and Yun Fu. Modeling complex temporal composition of actionlets for activity prediction. In *European conference on computer vision*, pages 286–299. Springer, 2012. 27

Tianjiao Li, Jun Liu, Wei Zhang, and Lingyu Duan. Hard-net: Hardness-aware discrimination network for 3d early activity prediction. In *In European Conference on Computer Vision (ECCV)*, 2020. 26

Yanghao Li, Cuiling Lan, Junliang Xing, Wenjun Zeng, Chunfeng Yuan, and Jiaying Liu. Online human action detection using joint classification-regression recurrent neural networks. In *European Conference on Computer Vision*, pages 203–220. Springer, 2016. 26

Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. Jointly localizing and describing events for dense video captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018a. 91

Yin Li, Miao Liu, and James M Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 619–635, 2018b. xv, 19, 21, 24, 31, 142, 151

Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528, 2018c. 17

Zhenyang Li, Kirill Gavrilyuk, Efstratios Gavves, Mihir Jain, and Cees GM Snoek. Videolstm convolves, attends and flows for action recognition. *Computer Vision and Image Understanding*, 166:41–50, 2018d. 17

Xiaodan Liang, Lisa Lee, Wei Dai, and Eric P Xing. Dual motion gan for future-flow embedded video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1744–1752, 2017. 23

T Warren Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874, 2005. 73, 77

Angela S Lin, Sudha Rao, Asli Celikyilmaz, Elnaz Nouri, Chris Brockett, Debadeepta Dey, and Bill Dolan. A recipe for creating multimodal aligned datasets for sequential tasks. *arXiv preprint arXiv:2005.09606*, 2020. 48

Dahua Lin, Chen Kong, Sanja Fidler, and Raquel Urtasun. Generating multi-sentence lingual descriptions of indoor scenes. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2015. 82, 153

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 7083–7093, 2019a. 4, 16, 118, 151

Rongcheng Lin, Jing Xiao, and Jianping Fan. Nextvlad: An efficient neural network to aggregate frame-level features for large-scale video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 119

Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3889–3898, 2019b. 39, 40

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018. 13

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2122–2132, 2016. 91

Miao Liu, Siyu Tang, Yin Li, and James Rehg. Forecasting human object interaction: Joint prediction of motor attention and egocentric activity. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 29

Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 344–353, 2019. 39

Adam Lopez. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):8, 2008. 90, 154

Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 648–657, 2017. 23

Pauline Luc, Camille Couprie, Yann Lecun, and Jakob Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 584–599, 2018. 23

Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. Attend and interact: Higher-order object interactions for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6790–6800, 2018. 18

Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1942–1950, 2016. 4, 25, 26, 117

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9:2579–2605, 2008. xi, 97, 98

Khoi-Nguyen C Mac, Dhiraj Joshi, Raymond A Yeh, Jinjun Xiong, Rogerio S Feris, and Minh N Do. Learning motion in feature space: Locally-consistent deformable convolution networks for fine-grained action detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6282–6291, 2019. 33

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003. 59

Tahmida Mahmud, Mahmudul Hasan, and Amit K Roy-Chowdhury. Joint prediction of activity labels and starting times in untrimmed videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5773–5782, 2017. 28, 29, 117

Tahmida Mahmud, Mohammad Billah, Mahmudul Hasan, and Amit K Roy-Chowdhury. Prediction and description of near-future activities in video. *arXiv preprint arXiv:1908.00943*, 2020. 29

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *Proceedings of the IEEE international conference on computer vision*, pages 1–9, 2015. 45

Colin L Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957. 60

Jonathan Malmaud, Earl Wagner, Nancy Chang, and Kevin Murphy. Cooking with semantics. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 33–38, 2014. 44

Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nicholas Johnston, Andrew Rabinovich, and Kevin Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 143–152, 2015. xv, 9, 20, 21, 24, 36, 48, 51, 53, 83, 84, 147, 150, 153

Aashi Manglik, Xinshuo Weng, Eshed Ohn-Bar, and Kris M Kitani. Forecasting time-to-collision from monocular video: Feasibility, dataset, and challenges. *arXiv preprint arXiv:1903.09102*, 2019. 28

Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2891–2900, 2017. 23

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305, 2017. 43

Nazanin Mehrasa, Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. A variational auto-encoder model for stochastic point processes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3165–3174, 2019. 30

Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *arXiv preprint arXiv:1706.06905*, 2017. 18, 118, 119, 129

Antoine Miech, Ivan Laptev, and Josef Sivic. Learning a text-video embedding from incomplete and heterogeneous data. *arXiv preprint arXiv:1804.02516*, 2018. 45

Antoine Miech, Ivan Laptev, Josef Sivic, Heng Wang, Lorenzo Torresani, and Du Tran. Leveraging the present to anticipate the future in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) Workshops*, pages 0–0, 2019a. 27, 29, 115, 117, 126, 132, 133, 137

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE international conference on computer vision*, pages 2630–2640, 2019b. 45, 153

Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9879–9889, 2020. 46, 153

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010. 42

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 45

Weiqing Min, Shuqiang Jiang, Shuhui Wang, Jitao Sang, and Shuhuan Mei. A delicious recipe analysis framework for exploring multi-modal recipes with various attributes. In *Proceedings of the 25th ACM International Conference on Multimedia*, 2017. 84

Moley. The world's first robotic kitchen. `http://www.moley.com/`, 2018. 80

Shinsuke Mori, Hirokuni Maeta, Tetsuro Sasada, Koichiro Yoshino, Atsushi Hashimoto, Takuya Funatomi, and Yoko Yamakata. FlowGraph2Text: Automatic sentence skeleton compilation for procedural text generation. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, 2014. 84

Richard G Morris, Scott H Burton, Paul M Bodily, and Dan Ventura. Soup over bean of pure joy: Culinary ruminations of an artificial chef. *Computational Creativity*, 2012. 83

Ramy Mounir, Roman Gula, Jörn Theuerkauf, and Sudeep Sarkar. Temporal event segmentation using attention-based perceptual prediction model for continual learning. *arXiv preprint arXiv:2005.02463*, 2020. 148

Jonghwan Mun, Linjie Yang, Zhou Ren, Ning Xu, and Bohyung Han. Streamlined dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6588–6597, 2019. 47

Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 163–172, 2020. 30

Iftekhar Naim, Young Chol Song, Qiguang Liu, Henry Kautz, Jiebo Luo, and Daniel Gildea. Unsupervised alignment of natural language instructions with video segments. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1558–1564, 2014. 48

Iftekhar Naim, Young Chol Song, Qiguang Liu, Liang Huang, Henry Kautz, Jiebo Luo, and Daniel Gildea. Discriminative unsupervised alignment of natural language instructions with corresponding video segments. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 164–174, 2015. 48

Zwe Naing and Ehsan Elhamifar. Procedure completion by learning from partial summaries. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2020. 41

Radford M Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003. 59, 67, 69

Positively Negative. The first workshop on negative results in computer vision. cvpr 2017. `http://negative.vision/`, 2018. 154

Lukas Neumann, Andrew Zisserman, and Andrea Vedaldi. Future event prediction: If and when. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 28

Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6752–6761, 2018. 40

Phuc Xuan Nguyen, Deva Ramanan, and Charless C Fowlkes. Weakly-supervised action localization with background modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5502–5511, 2019. 40

Taichi Nishimura, Atsushi Hashimoto, and Shinsuke Mori. Procedural text generation from a photo sequence. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 409–414, 2019. 44

NLTK. Natural language toolkit: Nltk 3.3 documentation. `http://www.nltk.org/`, 2018. 94

Cristian Rodriguez Opazo, Edison Marrese-Taylor, Fatemeh Sadat Saleh, Hongdong Li, and Stephen Gould. Proposal-free temporal moment localization of a natural-language query in video using guided attention. *arXiv preprint arXiv:1908.07236*, 2019. 48

Pavel Ostyakov, Elizaveta Logacheva, Roman Suvorov, Vladimir Aliev, Gleb Sterkin, Oleg Khomenko, and Sergey I Nikolenko. Label denoising with large ensembles of heterogeneous neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 119

Andrew Owens, Phillip Isola, Josh McDermott, Antonio Torralba, Edward H Adelson, and William T Freeman. Visually indicated sounds. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2405–2413, 2016. 46

Boxiao Pan, Haoye Cai, De-An Huang, Kuan-Hui Lee, Adrien Gaidon, Ehsan Adeli, and Juan Carlos Niebles. Spatio-temporal graph for video captioning with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10870–10879, 2020. 46

Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4594–4602, 2016. 45, 46

Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. Video captioning with transferred semantic attributes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6504–6512, 2017. 46

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics (NAACL)*, pages 311–318. Association for Computational Linguistics, 2002. 90, 154

Mingtao Pei, Yunde Jia, and Song-Chun Zhu. Parsing video events with goal inference and intent prediction. In *2011 International Conference on Computer Vision*, pages 487–494. IEEE, 2011. 27

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 29, 140

Juan-Manuel Perez-Rua, Antoine Toisoul, Brais Martinez, Victor Escorcia, Li Zhang, Xiatian Zhu, and Tao Xiang. Egocentric action recognition by video attention and temporal context. *arXiv preprint arXiv:2007.01883*, 2020. 142

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010. 12

Fiora Pirri, Lorenzo Mauro, Edoardo Alati, Valsamis Ntouskos, Mahdieh Izadpanahkakhk, and Elham Omrani. Anticipation and next action forecasting in video: an end-to-end model with memory. *arXiv preprint arXiv:1901.03728*, 2019. 28

Hamed Pirsiavash and Deva Ramanan. Detecting activities of daily living in first-person camera views. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2847–2854. IEEE, 2012. xv, 19, 22, 24

Bryan A Plummer, Matthew Brown, and Svetlana Lazebnik. Enhancing video summarization via vision-language embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5781–5789, 2017. 45

Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010. 2, 4

Tanzila Rahman, Bicheng Xu, and Leonid Sigal. Watch, listen and tell: Multi-modal weakly supervised dense event captioning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8908–8917, 2019. 47

Amir Rasouli, Iuliia Kotseruba, Toni Kunic, and John K Tsotsos. Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6262–6271, 2019. 23

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 2013. xv, 20, 24, 46, 84, 90

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 153

Nicholas Rhinehart and Kris M Kitani. First-person activity forecasting with online inverse reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3696–3705, 2017. 27

Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 3131–3140, 2016. 31, 32, 36, 53, 83, 147

Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2017. 35, 53, 54, 77, 78, 83, 115, 117, 132, 133, 134, 135, 152

Alexander Richard, Hilde Kuehne, and Juergen Gall. Action sets: Weakly supervised action segmentation without ordering constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5987–5996, 2018a. 36

Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7386–7395, 2018b. 4, 35, 48

Dominik Rivoir, Sebastian Bodenstedt, Felix von Bechtolsheim, Marius Distler, Jürgen Weitz, and Stefanie Speidel. Unsupervised temporal video segmentation as an auxiliary task for predicting the remaining surgery duration. In *OR 2.0 Context-Aware Operating Theaters and Machine Learning in Clinical Neuroimaging*, pages 29–37. Springer, 2019. 38

Dominik Rivoir, Sebastian Bodenstedt, Isabel Funke, Felix von Bechtolsheim, Marius Distler, Jürgen Weitz, and Stefanie Speidel. Rethinking anticipation tasks: Uncertainty-aware

anticipation of sparse surgical instrument usage for context-aware assistance. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, 2020. 23

Cristian Rodriguez, Basura Fernando, and Hongdong Li. Action anticipation by predicting future dynamic images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 26

Karl Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP: Image understanding*, 59(1):94–115, 1994. 4

Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In *German conference on pattern recognition*, pages 184–195. Springer, 2014. 6, 46, 84

Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2012. xv, 19, 20, 24, 53, 82, 83, 84, 147

Marcus Rohrbach, Wei Qiu, Ivan Titov, Stefan Thater, Manfred Pinkal, and Bernt Schiele. Translating video content to natural language descriptions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013. 84, 90

Marcus Rohrbach, Anna Rohrbach, Michaela Regneri, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision*, pages 1–28, 2015. ISSN 0920-5691. doi: 10.1007/s11263-015-0851-8. URL `http://dx.doi.org/10.1007/s11263-015-0851-8`. xv, 20, 24

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 12

Michael S Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011. 25, 83, 117

Michael S Ryoo, Chia-Chih Chen, JK Aggarwal, and Amit Roy-Chowdhury. An overview of contest on semantic description of human activities (sdha) 2010. In *International Conference on Pattern Recognition*, pages 270–285. Springer, 2010. 25

MS Ryoo, Thomas J Fuchs, Lu Xia, Jake K Aggarwal, and Larry Matthies. Robot-centric activity prediction from first-person videos: What will they do to me? In *2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 295–302. IEEE, 2015. 27

Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2019. 23

Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 3020–3028, 2017. 45, 81, 84, 86, 88, 90, 96, 97

Amaia Salvador, Michal Drozdzal, Xavier Giro-i Nieto, and Adriana Romero. Inverse cooking: Recipe generation from food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10453–10462, 2019. 44, 84, 86

Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105 (3):222–245, 2013. 53, 63, 71

Roger C Schank and Robert P Abelson. Scripts, plans, and knowledge. In *Proceedings of the 4th international joint conference on Artificial intelligence-Volume 1*, pages 151–157, 1975. 51

Edgar Schonfeld, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. Generalized zero-and few-shot learning via aligned variational autoencoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8247–8255, 2019. 144, 152

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020. 106

Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8368–8376, 2018. 10, 31, 37, 38, 49, 83, 115, 117, 147, 152

Fadime Sener and Angela Yao. Zero-shot anticipation for instructional activities. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 862–871, 2019. xv, 10, 20, 24, 79, 147, 150, 152, 153

Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *European Conference on Computer Vision*, pages 154–171. Springer, 2020. 2, 10, 30, 31, 113, 147

Ozan Sener, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 4, 6, 36, 51, 53, 73, 83, 84, 147, 148, 152

Laura Sevilla-Lara, Shengxin Zha, Zhicheng Yan, Vedanuj Goswami, Matt Feiszli, and Lorenzo Torresani. Only time can tell: Discovering temporal data for temporal modeling. *arXiv preprint arXiv:1907.08340*, 2019. 17

Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov. Action recognition using visual attention. In *Neural Information Processing Systems (NIPS) Time Series Workshop*, December 2015. 17, 50, 53

Yang Shen, Bingbing Ni, Zefan Li, and Ning Zhuang. Egocentric activity prediction via event modulated attention. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 197–212, 2018. 29

Zhiqiang Shen, Jianguo Li, Zhou Su, Minjun Li, Yurong Chen, Yu-Gang Jiang, and Xiangyang Xue. Weakly supervised dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1916–1924, 2017. 47

Botian Shi, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and Ming Zhou. Dense procedure captioning in narrated instructional videos. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6382–6391, 2019. 47

Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 301–317, 2018. 25, 26

Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016. 39

Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5734–5743, 2017. 39

Zheng Shou, Hang Gao, Lei Zhang, Kazuyuki Miyazawa, and Shih-Fu Chang. Autoloc: Weakly-supervised temporal action localization in untrimmed videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 154–171, 2018. 40

Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 2016. xv, 19, 22, 24

Gunnar A Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. Actor and observer: Joint modeling of first and third-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7396–7404, 2018. xv, 22, 24

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014a. 14, 15, 50, 51, 117, 151, 153

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014b. 13, 38, 71

Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1961–1970, 2016. 31, 32, 33, 151

Gurkirt Singh, Suman Saha, Michael Sapienza, Philip HS Torr, and Fabio Cuzzolin. Online real-time multiple spatiotemporal action localisation and prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3637–3646, 2017. 23

Gurkirt Singh, Suman Saha, and Fabio Cuzzolin. Predicting action tubes. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 23

Miha Skalic and David Austin. Building a size constrained predictive models for video classification. In *European Conference on Computer Vision*, pages 297–305. Springer, 2018. 18

Young Chol Song, Iftekhar Naim, Abdullah Al Mamun, Kaustubh Kulkarni, Parag Singla, Jiebo Luo, Daniel Gildea, and Henry Kautz. Unsupervised alignment of actions in video with text descriptions. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2025–2031, 2016. 48

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 17, 23, 25, 50, 115, 117, 152

Bilge Soran, Ali Farhadi, and Linda Shapiro. Generating notifications for missing actions: Don't forget to turn the lights off! In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4669–4677, 2015. 4, 31

Yaser Souri, Mohsen Fayyaz, Luca Minciullo, Gianpiero Francesca, and Juergen Gall. Fast weakly supervised action segmentation using mutual consistency. *arXiv preprint arXiv:1904.03116*, 2019. 35

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 13, 121

Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning (ICML)*, pages 843–852, 2015. 51, 53

Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738. ACM, 2013. xv, 19, 20, 24, 31, 116, 118, 124

Jonathan Stroud, David Ross, Chen Sun, Jia Deng, and Rahul Sukthankar. D3d: Distilled 3d networks for video action recognition. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 625–634, 2020. 16

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473, 2019a. 43, 46, 153

Chen Sun, Abhinav Shrivastava, Carl Vondrick, Rahul Sukthankar, Kevin Murphy, and Cordelia Schmid. Relational action forecasting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 273–283, 2019b. 25, 26

Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4597–4605, 2015. 16

Niko Sünderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, et al. The limits and potentials of deep learning for robotics. *The International Journal of Robotics Research*, 37(4-5):405–420, 2018. 80

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112, 2014. 6, 44, 84

Tomoyuki Suzuki, Hirokatsu Kataoka, Yoshimitsu Aoki, and Yutaka Satoh. Anticipating traffic accidents with adaptive loss and large-scale incident db. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3521–3529, 2018. 28

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 13

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 29

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1207–1216, 2019. xv, 19, 21, 24, 150, 151, 153

Yongyi Tang, Xing Zhang, Lin Ma, Jingwen Wang, Shaoxiang Chen, and Yu-Gang Jiang. Non-local netvlad encoding for video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 18, 118

Bugra Tekin, Federica Bogo, and Marc Pollefeys. H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4511–4520, 2019. 18

Moritz Tenorth, Daniel Nyga, and Michael Beetz. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *2010 ieee international conference on robotics and automation*, pages 1486–1491. IEEE, 2010. 44

Moritz Tenorth, Alexander Clifford Perzylo, Reinhard Lafrenz, and Michael Beetz. Representation and exchange of knowledge about actions, objects, and environments in the roboearth framework. *IEEE Transactions on Automation Science and Engineering*, 10(3): 643–651, 2013. 80

F. Torre, J. Hodgins, Adam W. Bargteil, X. Martin, J. Macey, A. Collado, and Pep Beltran. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. 2008. 78

Sam Toyer, Anoop Cherian, Tengda Han, and Stephen Gould. Human pose forecasting via deep Markov models. In *DICTA*, 2017. xv, 19, 21, 24, 151

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497, 2015. 14, 16, 17, 50, 51, 53, 63, 117, 129

Du Tran, Jamie Ray, Zheng Shou, Shih-Fu Chang, and Manohar Paluri. Convnet architecture search for spatiotemporal feature learning. *arXiv preprint arXiv:1708.05038*, 2017. 16

Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6450–6459, 2018. 16, 133

Vinh Tran, Yang Wang, and Minh Hoai. Back to the future: Knowledge distillation for human action anticipation. *arXiv preprint arXiv:1904.04868*, 2019. 28

Andru Putra Twinanda, Gaurav Yengera, Didier Mutter, Jacques Marescaux, and Nicolas Padoy. Rsdnet: Learning to predict remaining surgery duration from laparoscopic videos without manual annotations. *IEEE transactions on medical imaging*, 38(4):1069–1078, 2018. 26

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 43, 47

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2015. 90, 154

Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. xv, 6, 46, 100, 104

Rosaura G VidalMata, Walter J Scheirer, Anna Kukleva, David Cox, and Hilde Kuehne. Joint visual-temporal embedding for unsupervised learning of actions in untrimmed sequences. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1238–1247, 2021. 38

Oriol Vinyals and Quoc V. Le. A neural conversational model. In *ICML Deep Learning Workshop*, 2015. 44, 84, 92

Nam N Vo and Aaron F Bobick. From stochastic grammar to bayes network: Probabilistic parsing of complex activity. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2641–2648, 2014. 32

Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016. 28, 83, 115, 117, 132, 133

Bairui Wang, Lin Ma, Wei Zhang, and Wei Liu. Reconstruction network for video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7622–7631, 2018a. 46

Boyu Wang, Lihan Huang, and Minh Hoai. Active vision for early recognition of human actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1081–1091, 2020a. 25, 26

Hao Wang, Doyen Sahoo, Chenghao Liu, Ee-peng Lim, and Steven CH Hoi. Learning cross-modal embeddings with adversarial networks for cooking recipes and food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11572–11581, 2019a. 45

Hao Wang, Guosheng Lin, Steven CH Hoi, and Chunyan Miao. Decomposed generation networks with structure prediction for recipe generation from food images. *arXiv preprint arXiv:2007.13374*, 2020b. 44

Hao Wang, Guosheng Lin, Steven CH Hoi, and Chunyan Miao. Structure-aware generation network for recipe generation from images. In *In European Conference on Computer Vision (ECCV)*, 2020c. 44

Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3551–3558, 2013. 12, 15, 71, 119, 124, 152

Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR 2011*, pages 3169–3176. IEEE, 2011. 12

Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103(1):60–79, 2013. 14, 51, 53, 63

Jingwen Wang, Wenhao Jiang, Lin Ma, Wei Liu, and Yong Xu. Bidirectional attentive fusion with context gating for dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7190–7198, 2018b. 47

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 20–36, 2016a. 15, 18, 117, 119, 121, 125, 129, 151

Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4325–4334, 2017. 40

Xiaohan Wang, Linchao Zhu, Yu Wu, and Yi Yang. Symbiotic attention for egocentric action recognition with object-centric alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020d. 18, 141

Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European conference on computer vision (ECCV)*, pages 399–417, 2018. 18

Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions˜ transformations. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2658–2667, 2016b. 2

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7794–7803, 2018c. 18, 121, 128, 130

Xionghui Wang, Jian-Fang Hu, Jian-Huang Lai, Jianguo Zhang, and Wei-Shi Zheng. Progressive teacher-student learning for early action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3556–3565, 2019b. 25, 26

Zhe Wang, Wei He, Hua Wu, Haiyang Wu, Wei Li, Haifeng Wang, and Enhong Chen. Chinese poetry generation with planning based neural network. *arXiv preprint arXiv:1610.09889*, 2016c. 43

Zhenzhi Wang, Ziteng Gao, Limin Wang, Zhifeng Li, and Gangshan Wu. Boundary-aware cascade networks for temporal action segmentation. In *In European Conference on Computer Vision (ECCV)*, 2020e. 34

Zhikun Wang, Marc Peter Deisenroth, Heni Ben Amor, David Vogt, Bernhard Schölkopf, and Jan Peters. Probabilistic modeling of human movements for intention inference. *Proceedings of robotics: Science and systems, VIII*, 2012. 27

Xinyu Wei, Patrick Lucey, Stephen Vidas, Stuart Morgan, and Sridha Sridharan. Forecasting events using an augmented hidden conditional random field. In *Asian Conference on Computer Vision*, pages 569–582. Springer, 2014. 23

Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Learning to track for spatio-temporal action localization. In *Proceedings of the IEEE international conference on computer vision*, pages 3164–3172, 2015. 15

Wikihow. How to do anything. `http://www.wikihow.com/`, 2005. 81

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 18, 115, 116, 118, 121, 128, 130, 140, 141

Chenxia Wu, Jiemi Zhang, Silvio Savarese, and Ashutosh Saxena. Watch-n-patch: Unsupervised understanding of actions and relations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4362–4370, 2015. 37

Chenxia Wu, Jiemi Zhang, Bart Selman, Silvio Savarese, and Ashutosh Saxena. Watch-bot: Unsupervised learning for reminding humans of forgotten actions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016. 80

Tz-Ying Wu, Ting-An Chien, Cheng-Sheng Chan, Chan-Wei Hu, and Min Sun. Anticipating daily intention using on-wrist motion triggered sensing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 48–56, 2017. 27

Yue Wu, Rongrong Gao, Jaesik Park, and Qifeng Chen. Future video synthesis with object motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5539–5548, 2020. 23

Fanyi Xiao, Yong Jae Lee, Kristen Grauman, Jitendra Malik, and Christoph Feichtenhofer. Audiovisual slowfast networks for video recognition. *arXiv preprint arXiv:2001.08740*, 2020. 15, 129, 130, 140, 141

Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018. 16, 117

Yilei Xiong, Bo Dai, and Dahua Lin. Move forward and tell: A progressive generator of video descriptions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 468–483, 2018. 47

Yuanjun Xiong, Yue Zhao, Limin Wang, Dahua Lin, and Xiaoou Tang. A pursuit of temporal accuracy in general activity detection. *arXiv preprint arXiv:1703.02716*, 2017. 39

Chengguang Xu and Ehsan Elhamifar. Deep supervised summarization: Algorithm and application to learning instructions. In *Advances in Neural Information Processing Systems*, pages 1109–1120, 2019. 41

Huijuan Xu, Boyang Li, Vasili Ramanishka, Leonid Sigal, and Kate Saenko. Joint event detection and description in continuous video streams. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 396–405. IEEE, 2019a. 47

Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020. 39

Mingze Xu, Mingfei Gao, Yi-Ting Chen, Larry S Davis, and David J Crandall. Temporal recurrent networks for online action detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5532–5541, 2019b. 26

Zhen Xu, Laiyun Qing, and Jun Miao. Activity auto-completion: Predicting human activities from partial videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015a. 83

Zhen Xu, Laiyun Qing, and Jun Miao. Activity auto-completion: Predicting human activities from partial videos. In *Proceedings of the IEEE international conference on computer vision*, pages 3191–3199, 2015b. 25

Xiang Xuan and Kevin Murphy. Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th international conference on Machine learning*, pages 1055–1062, 2007. 37

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015. xv, 46, 100, 104

Taiping Yao, Minsi Wang, Bingbing Ni, Huawei Wei, and Xiaokang Yang. Multiple granularity group interaction prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2246–2254, 2018. 26

Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016. 39, 53

Gang Yu, Junsong Yuan, and Zicheng Liu. Predicting human activities using spatio-temporal structure of interest points. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 1049–1052, 2012. 25

Haonan Yu and Jeffrey Mark Siskind. Grounded language learning from video described with sentences. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 53–63, 2013. 46

Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016. 47

Yitian Yuan, Tao Mei, and Wenwu Zhu. To find where you talk: Temporal sentence localization in video with attention based location regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9159–9166, 2019. 48

Zehuan Yuan, Jonathan C Stroud, Tong Lu, and Jia Deng. Temporal action localization by structured maximal sums. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3684–3692, 2017. 40

Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4694–4702, 2015. 17, 117, 129

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 12

Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001. 37

Kuo-Hao Zeng, Shih-Han Chou, Fu-Hsiang Chan, Juan Carlos Niebles, and Min Sun. Agent-centric risk assessment: Accident anticipation and risky region localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2222–2230, 2017a. 28

Kuo-Hao Zeng, William B Shen, De-An Huang, Min Sun, and Juan Carlos Niebles. Visual forecasting by imitating dynamics in natural sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2999–3008, 2017b. 28

Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *European conference on computer vision*, pages 766–782. Springer, 2016. 41

Ke Zhang, Kristen Grauman, and Fei Sha. Retrospective encoders for video summarization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 383–399, 2018. 4

Xishan Zhang, Ke Gao, Yongdong Zhang, Dongming Zhang, Jintao Li, and Qi Tian. Task-driven dynamic fusion: Reducing ambiguity in video description. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, pages 3713–3721, 2017. 46

Yan Zhang, Krikamol Muandet, Qianli Ma, Heiko Neumann, and Siyu Tang. Frontal low-rank random tensors for fine-grained action segmentation. *arXiv preprint arXiv:1906.01004*, 2019. 34

He Zhao and Richard P Wildes. On diverse asynchronous activity anticipation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 30

Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2914–2923, 2017. 4, 39

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 25

Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018a. 17

Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2020. 152

Feng Zhou, Fernando De la Torre, and Jessica K Hodgins. Aligned cluster analysis for temporal segmentation of human motion. In *2008 8th IEEE international conference on automatic face & gesture recognition*, pages 1–7. IEEE, 2008. 37

Feng Zhou, Fernando De la Torre, and Jessica K Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):582–596, 2012. 37, 54

Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018b. xv, 9, 19, 20, 21, 24, 40, 47, 82, 84, 88, 90, 150, 151, 153

Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018c. xv, 43, 46, 47, 84, 90, 100, 104, 105

Yipin Zhou and Tamara L Berg. Temporal perception and prediction in ego-centric video. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4498–4506, 2015. 28, 82

Bin Zhu and Chong-Wah Ngo. Cookgan: Causality based text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5519–5527, 2020. 44

Linchao Zhu and Yi Yang. Actbert: Learning global-local video-text representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8746–8755, 2020. 46

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015. 48

Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3537–3545, 2019. xv, 15, 21, 24, 36, 38, 41, 150

Mohammadreza Zolfaghari, Özgün Çiçek, Syed Mohsin Ali, Farzaneh Mahdisoltani, Can Zhang, and Thomas Brox. Learning representations for predicting future activities. *arXiv preprint arXiv:1905.03578*, 2019. 28