Holistic Vocabulary Independent Spoken Term Detection

Dissertation

zur

Erlangung des Doktorgrads (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Daniel Schneider

aus

Saarlouis

Bonn 2011

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

1. Gutachter: Prof. Dr. Christian Bauckhage
2. Gutachter: Prof. Dr. Michael Clausen
Tag der Promotion: 22. Mai 2012
Erscheinungsjahr: 2012

Abstract

Within this thesis, we aim at designing a loosely coupled holistic system for Spoken Term Detection (STD) on heterogeneous German broadcast data in selected application scenarios. Starting from STD on the 1-best output of a word-based speech recognizer, we study the performance of several subword units for vocabulary independent STD on a linguistically and acoustically challenging German corpus. We explore the typical error sources in subword STD, and find that they differ from the error sources in word-based speech search. We select, extend and combine a set of state-of-the-art methods for error compensation in STD in order to explicitly merge the corresponding STD error spaces through anchor-based approximate lattice retrieval. Novel methods for STD result verification are proposed in order to increase retrieval precision by exploiting external knowledge at search time. Error-compensating methods for STD typically suffer from high response times on large scale databases, and we propose scalable approaches suitable for large corpora. Highest STD accuracy is obtained by combining anchor-based approximate retrieval from both syllable lattice ASR and syllabified word ASR into a hybrid STD system, and pruning the result list using external knowledge with hybrid contextual and anti-query verification.

Zusammenfassung

Die vorliegende Arbeit beschreibt ein lose gekoppeltes, ganzheitliches System zur Sprachsuche auf heterogenenen deutschen Sprachdaten in unterschiedlichen Anwendungsszenarien. Ausgehend von einer wortbasierten Sprachsuche auf dem Transkript eines aktuellen Wort-Erkenners werden zunächst unterschiedliche Subwort-Einheiten für die vokabularunabhängige Sprachsuche auf deutschen Daten untersucht. Auf dieser Basis werden die typischen Fehlerquellen in der Subwort-basierten Sprachsuche analysiert. Diese Fehlerquellen unterscheiden sich vom Fall der klassichen Suche im Worttranskript und müssen explizit adressiert werden. Die explizite Kompensation der unterschiedlichen Fehlerquellen erfolgt durch einen neuartigen hybriden Ansatz zur effizienten Anker-basierten unscharfen Wortgraph-Suche. Darüber hinaus werden neuartige Methoden zur Verifikation von Suchergebnissen vorgestellt, die zur Suchzeit verfügbares externes Wissen einbeziehen. Alle vorgestellten Verfahren werden auf einem umfangreichen Satz von deutschen Fernsehdaten mit Fokus auf ausgewählte, repräsentative Einsatzszenarien evaluiert. Da Methoden zur Fehlerkompensation in der Sprachsuchforschung typischerweise zu hohen Laufzeiten bei der Suche in großen Archiven führen, werden insbesondere auch Szenarien mit sehr großen Datenmengen betrachtet. Die höchste Suchleistung für Archive mittlerer Größe wird durch eine unscharfe und Anker-basierte Suche auf einem hybriden Index aus Silben-Wortgraphen und silbifizierter Wort-Erkennung erreicht, bei der die Suchergebnisse mit hybrider Verifikation bereinigt werden.

Acknowledgements

Many people have contributed to the thesis at hand in various forms. First, I would like to thank my advisor Prof. Dr. Christian Bauckhage for supporting the work on this thesis over the past years. His analytical skills helped me to concentrate on the core ideas and wipe away irrelevant aspects. Moreover, I thank Prof. Dr. Michael Clausen for taking over the second review, and for his feedback on our work on merging STD error spaces.

Collaboration is a key ingredient for successful scientific work. In the past years, I had the chance to work with many fascinating people from the speech community, and meet them at conferences and workshops all over the world. Two of them deserve special consideration. I would like to thank Dr. Martha Larson at Delft University of Technology for introducing me to the Spoken Term Detection community, and for her support while shaping the scope of this thesis. Thank you for your enthusiasm (and for proofreading this thesis!). I am also grateful to my colleague Timo Mertens from NTNU Trondheim. Thank you for endless phone calls before countless conference deadlines, for challenging my ideas and for sharing yours.

Since 2006, I have been with the Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS). Many colleagues at Fraunhofer have contributed to this thesis through discussions and valuable suggestions. In particular, I would like to thank our department head Dr. Joachim Köhler for his continuous support of my work and career. Unsurprisingly, working on a dissertation project takes up a lot of time, and many apsects of everyday project work need to stand back. Several colleagues at Fraunhofer, especially Jochen Schwenninger, Sebastian Tschöpel and Jochen Schon, have taken up some of my tasks when they didn't have to, and made my life much easier in these past years - thank you! I also wish to thank Dr. Rolf Bardeli for providing helpful insights into the dissertation process at the University of Bonn.

The work on this thesis has been carried out in various project contexts, both in cooperation with industry and academia. I am particularly grateful to the European Commission for funding the VITALAS project, where major parts of thesis have been prepared and published. I also wish to thank our partners from the broadcasting industry for providing requirements for real-world speech-driven applications, and for taking up some of the results from this thesis in innovative joint projects.

A long trail of education prequels the work on this thesis, and I am particularly grateful to my parents for making it possible. Thank you for believing in me from the very beginning, and for guiding my way until I could find my own.

I am deeply indebted to Annette for her never-ending belief and confidence in my abilities. Thank you for encouraging me during the most challenging moments, and for being the constant in my life. Thank you Alva for making sense out of everything. I would not have succeeded without the two of you.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Holistic Spoken Term Detection

Today, more new data is uploaded to YouTube in a minute than a single user can watch in two whole days [84]. A German household can choose from over 20,000 hours per day via digital satellite TV [100]. By 2015, it will take five years to watch all video data that crosses global networks *in a single second* [18]. Only a fraction of this data deluge that floods through our digital age is preserved and stored in audiovisual archives, nevertheless their size is exploding. For example, the French national audiovisual archive adds over 500,000 hours of TV and radio recordings to their archives every single year [49].

Considering the sheer scale of available data and the fact that it's largely unannotated, there is little need for further motivation of research in multimedia search, and it is clear that *speech* is a major source of information when searching in such large corpora [24]. Consider the scenario of a large broadcast archive, where the archivist wants to find statements on regulations of the financial system. A politician might comment on the topic in a discussion show on the financial crisis in general. Even if manual resources would be committed for annotating the discussion show with manually selected keywords, it is uncertain whether this particular statement would be reflected in the annotation. However, searching the speech track will *unlock* the archive and enable the archivist to find the item he is looking for.

Ad-hoc search in the speech track of audiovisual data for the occurrence of a written query is typically referred to as *Spoken Term Detection (STD)* in the literature. A key aspect of STD systems is their vocabulary independence, i.e., the set of possible queries is not known in advance. Therefore, typical STD approaches go beyond the application of classic word-based speech recognition, and exploit a wide range of techniques for increasing the STD accuracy on arbitrary queries.

Applications scenarios for STD include many interesting use cases, from searching large media archives to monitoring of radio and TV streams. At Fraunhofer IAIS, we are particularly interested in the task of STD on heterogeneous German broadcast data. We found that many interesting applications can be built on top of the core STD

technology in this data domain, and that interest from the owners of large media archives in unlocking their content is high [30, 28, 29].

End user expectations for such STD systems on large data sets are challenging, since they have been 'trained' by the daily use of Internet search engines. They expect that a result from a search engine actually contains an occurrence of the query, and that they will be able to find all documents where a query occurs. Furthermore, users will expect the same behavior across application domains on all types of data, e.g., from searching Wikipedia to spontaneous chat messages. And finally, every day we experience that web search engines deliver results for a search on the whole Internet within milliseconds. Why should searching a video archive take longer?

Considering the state of the art in STD research as summarized in chapter 2.3, we observe that current systems are far from fulfilling the expectations of the users: no approach yields highly precise and complete results across application domains, with reasonable efficiency on large data sets. Moreover, while STD research has received much attention recently, only little work has been published on the specific requirements for STD on German data [60].

Looking at the variety of application scenarios, we found the following challenges that prevented us from successfully deploying German STD in our projects:

- The actual requirements for STD systems in the various scenarios are not well defined, hence it is difficult to asses whether a certain approach is suitable for a given task.
- There is a lack of evaluation resources for German STD, so even if we had the perfect system, we could not measure its performance.
- It is unclear how recent approaches to STD perform on German, and how specific German language characteristics (such as inflections or compounding) impact the STD performance.
- There is only little interest in efficient approaches which scale beyond the research laboratory.
- There is a lack of flexibility in the current state of the art, since approaches are typically tailored towards a specific application scenario.

Within the scope of this thesis, we will approach these challenges with the aim of building an accurate and efficient system for German Spoken Term Detection that can be flexibly tailored towards a specific scenario. We envisage a *holistic* approach to

Spoken Term Detection, with loosely coupled components that can be flexibly assembled to meet the accuracy and efficiency requirements of a given application scenario. We aim at integrating these individual components into a *holistic* STD system suitable for heterogeneous German broadcast data. Figure 1.1 illustrates the architecture of the holistic STD system described within this thesis, and indicates the major design decisions that can be taken when targeting a new scenario.

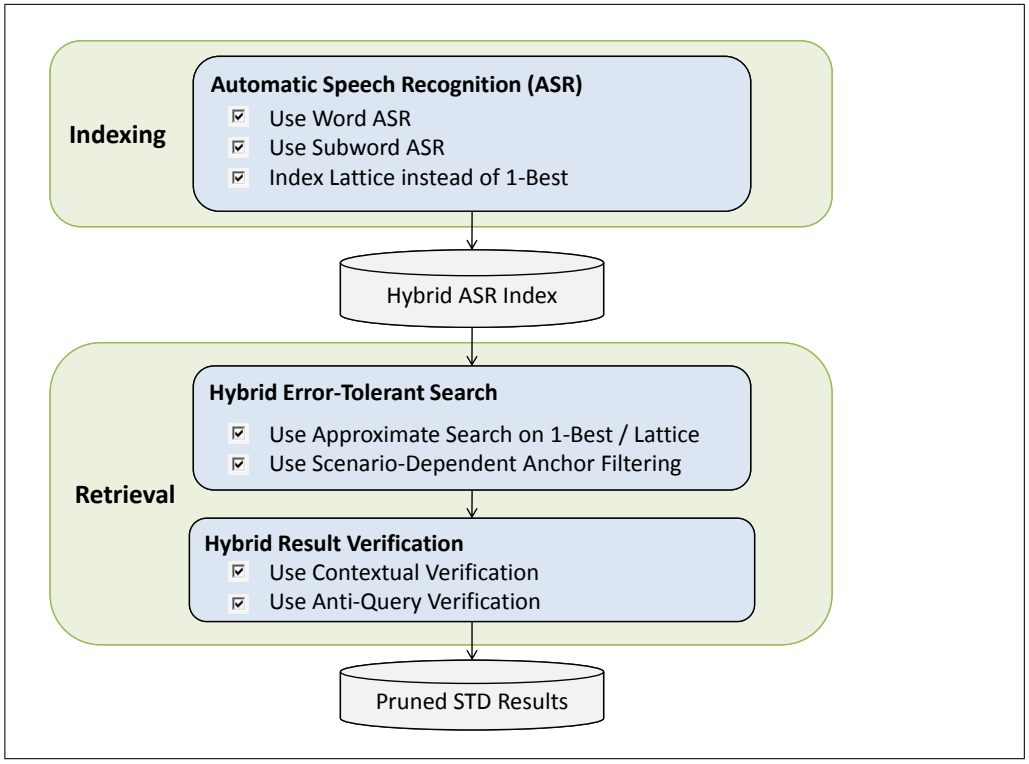

Figure 1.1.: Holistic Spoken Term Detection - system overview.

The individual components of our holistic system will be described and evaluated in the remainder of this thesis. In the following, we will specify the related scientific goals in more detail.

1.2. Scientific Goals

Our vision is to design and implement a holistic system for German Spoken Term Detection, which allows for error-tolerant and vocabulary independent speech search on large archives with heterogeneous data. This broad vision can be subdivided into a set

of individual scientific goals as follows:

- **Specification of user requirements for Spoken Term Detection.** Oftentimes, techniques proposed in STD research provide only punctual solutions aiming at specific application scenarios. However, indexing and retrieval requirements vary substantially between STD use cases. Based on well-defined use cases, we specify a set of requirements for Spoken Term Detection systems for selected scenarios. The requirements are based on an extensive collaboration with actual broadcast archives, which we have established over a range of joint projects [28, 30, 29, 2].

- **Design of an evaluation corpus for STD on heterogeneous German broadcast data.** To date, no standard evaluation corpus exists for evaluating German STD on broadcast data. This is a major obstacle which effectively hinders progress on German STD research. Hence, our aim is to design and develop a large scale evaluation corpus for German STD. The design will follow the characteristics of the successful and widely accepted NIST STD Evaluation corpus [82], which is used for STD research on English, Mandarin Chinese and Arabic throughout the STD research community. The corpus will not only contain the acoustic data and the reference transcriptions, but also a set of evaluation queries and the corresponding metric that can be used for assessing the quality of an STD run. Instead of purely automatic query selection as in the NIST corpus, we would like to enhance the query evaluation set with queries generated by humans, such that the evaluation data becomes even more realistic. Details on our novel corpus can be found in [6], and we have published a cross-site comparative study on the corresponding evaluation metrics in [80].

- **Design and implementation of a state-of-the-art word-based STD baseline for heterogeneous German broadcast data.** We will build a large vocabulary continuous speech recognizer as the baseline for subsequent STD experiments. The system architecture will be selected according to state-of-the art approaches to ASR, and language resources such as acoustic models, language models and pronunciation lexicon will be built to match with the characteristics of the selected application scenarios. The output of the system will be used for word-based STD, hence, we aim at a low OOV rate in the selected use cases and configure the system accordingly. A description of the system was recently published in [94].

- **Investigation of subword units suitable for German subword indexing.** Subword ASR has been proven to be a promising approach to vocabulary inde-

pendent STD in many scenarios and languages, however, only little work has been carried out on vocabulary independent STD on German data. Within this thesis, we will investigate the baseline STD performance of selected subword units particularly suited for the German language. Moreover, we will explicitly address the difference between recognition and retrieval unit, and investigate possible gains and drawbacks obtained from breaking down decoding units to smaller retrieval units. Our investigations in this chapter are based on our contribution in [96].

- **Investigation of new approaches for error-tolerant subword speech retrieval.** Error compensation plays an important role for effective subword retrieval, especially in complex acoustic and linguistic conditions where exact subword match yields only low search recall. Yet to date, there is no explicit analysis of the sources that lead to errors in subword STD. Within the scope of this theses, we will analyze the subword STD error space, and identify the major STD error sources. Based on this analysis, we will select and enhance state-of-the-art techniques for error compensation, and propose a novel hybrid approximate lattice retrieval approach, which effectively merges the error spaces, thereby increasing STD accuracy. Our results on error compensation in German STD have been published in [77] and [78].

- **Exploiting external knowledge for result verification at search time.** Applying methods for error compensation typically results in lower search precision. We investigate new approaches for verifying approximate STD results, where we aim to increase precision while preserving the recall gains obtained from the error compensation. Therefore, we introduce the concept of exploiting external knowledge about a specific query at search time in order to verify a putative STD result. We propose two methods which implement this paradigm: contextual verification and anti-query verification. We have first published preliminary results in [95], and provide a comprehensive investigation within the scope of this thesis.

- **Design of scalable algorithms for error-tolerant speech search on large corpora.** Search efficiency has not been in the focus of the STD research community so far, and query response time is not an issue for exact word search based on LVCSR. However, many of the proposed techniques for error compensation of subword errors suffer from high time complexity. We aim at designing retrieval approaches which allow for error compensation in a large scale subword search task, and which can be flexibly adjusted to the considered application scenarios.

- **Best practices for selected STD scenarios.** Based on our investigations into flexible and scalable STD, we will propose best practices for selected representative STD application scenarios. We will study the practical impact of merging actual word- and subword-based systems in to a hybrid STD approach, and investigate whether the additional burden of a second decoding subsystem pays off in terms of STD accuracy. Optimal search configurations will be given, which yield the highest STD accuracy while staying within the efficiency constraints of a specific scenario.

1.3. Structure of the Thesis

Following the scientific goals described above, the thesis at hand is structured as follows. In chapter 2, we first give an exact definition of the STD task. We describe the related state of the art in Spoken Term Detection, and identify limitations that will be investigated in the remainder of the thesis. We describe major STD scenarios in the field of media archive search and media monitoring, and derive requirements for actual deployed STD systems. Finally, the chapter presents our evaluation methodology for assessing the performance of a specific STD approach, including a novel STD evaluation corpus which we have presented in [6].

Each of the following chapters on vocabulary independent STD, error compensation and verification has the same structure: First, we point out our main own contributions. Then, the considered approaches are described in detail, followed by a comprehensive evaluation. Each chapter concludes with a summary that contains the main results and implications. Chapter 3 describes our baseline system for vocabulary independent Spoken Term Detection, which we have first presented in [96]. The following chapter 4 investigates error compensation for Spoken Term Detection, and introduces our hybrid to approximate lattice search based on our contributions in [77] and [78]. In chapter 5, we propose a novel approach for exploiting external knowledge for verification at query time, which was first published in [95].

Then, chapter 6 investigates the scalability of the approaches which we have studied above. We analyze the baseline efficiency of the most promising configurations, and propose optimizations that enable STD for a range of interesting scenarios. Finally, chapter 7.2 discusses the possible gain of hybrid word-subword STD systems in actual deployed systems, and provides best practices for STD in selected application scenarios. We conclude the thesis with a summary of our main contributions in the field of Spoken Term Detection, and close with a set of possible directions for future research.

2. Spoken Term Detection

We start with a formal description of the Spoken Term Detection task, and illustrate the goals and characteristics of the corresponding NIST evaluation. Current state-of-the-art approaches to STD are presented, and related to the scope of the thesis at hand. We identify a set of limitations in the current state of STD research, and describe gaps that will be bridged within the scope of this thesis.

Next, we present a set of STD application scenarios which are motivated by actual project contexts at Fraunhofer IAIS in the field of large scale media analysis [28, 30, 29], and derive a set of system requirements.

In section 2.4.2, we introduce the experimental setup for evaluating the proposed Spoken Term Detection approaches. A new corpus for ASR and STD evaluation on German data is presented, which we have published in [6] (and with a focus on STD in [94]). Finally, we describe a set of metrics for quantitative evaluation which are commonly used in the STD community. These metrics will then be used on the presented corpus in the following chapters for quantitative evaluation of STD vocabulary independence (chapter 3), STD error compensation (chapter 4), STD result verification (chapter 5) and STD scalability (chapter 6).

2.1. The Spoken Term Detection Task

The notion of *Spoken Term Detection (STD)* was coined by NIST in 2006 in the scope of the NIST STD evaluation campaign. According to NIST, STD focuses on "technologies that search vast, heterogeneous audio archives for occurrences of spoken terms"[1]. It contrasts to classic keyword spotting techniques like [116], as it is by definition an open-vocabulary task, i.e., the query terms are not known at indexing time. Vocabulary independence is a major requirement for many interesting applications, and it is especially useful in large corpora with heterogeneous content.

The STD task is formally defined as follows. Assume that a corpus C and a query set Q is given. The corpus consists of n audiovisual documents d

[1] http://www.itl.nist.gov/iad/mig//tests/std/

$$C = \{d_1, \ldots, d_n\} \tag{2.1}$$

and the query set Q contains r queries q:

$$Q = \{q_1, \ldots, q_r\} \tag{2.2}$$

where a query q_i consists of a sequence of i_m words $w_{i_1} \ldots w_{i_m}$. The goal of the STD task is to identify all occurrences $o(q_i)$ of each query q_i in the corpus. Such an occurrence hypothesis detected by the system is a tuple of the form

$$o(q_i) = (s, t_s, t_e, c) \tag{2.3}$$

where s is the document which contains the hypothesized hit at starting time t_s and end time t_e with a confidence of c. The STD confidence is the final confidence score produced by the system, and it is required that $c \in [0, 1]$. Typically, retrieval behavior is evaluated at different levels of confidence (e.g., using *receiver operating characteristics (ROC)* curves). It is left to the implementing STD system which mechanism is actually used for estimating the confidences, i.e., they must not necessarily come exclusively from the ASR decoder.

A full STD result is then a tuple (R, T), where

$$R = \bigcup_{i=1}^{r} \{o(q_i) | o \text{ is a hit hypothesis for } q_i\} \tag{2.4}$$

is the set of result hypotheses and T is the runtime of the retrieval run for all r queries. Efficiency is an important aspect for the practical applicability of STD approaches: in some use cases, retrieval is required to produce useful results on very large corpora with only small response times (see section 2.2 on STD requirements and use cases).

We note that STD is a technology situated on top of the ASR process. It searches for occurrences of spoken words regardless of the document or corpus context in which the words are spoken. The technical characteristics of the STD task description contrasts to what is typically understood by the term *Spoken Document Retrieval (SDR)*, which aims at producing a relevant spoken document for a given information need expressed by a user. One could rather think of STD as an auxiliary process, which provides input for the actual SDR system. An example run including STD and SDR could be illustrated as follows:

- The user expresses his information need, e.g., he wants to find videos about a

certain event.

- The SDR component expands the information need to a set of keywords which are typically spoken in videos about the event.
- The STD subsystem generates a set of hypotheses for the generated keywords.
- The SDR system integrates the information obtained from the STD system, e.g., by obtaining a relevance score for a document from the set of keywords hypothesized within this document.

This is a simple example for the possible interaction between STD and classic information retrieval techniques, which should clarify the typical auxiliary role of STD in a larger retrieval system. In this thesis, we limit our investigations to the actual STD task.

2.2. Application Scenarios and Requirements

Possible application scenarios for STD are numerous, and range from document retrieval in large audiovisual archives to continuous media monitoring of complex TV and radio data. In this section, we shortly review two representative use cases and describe possible general requirements for an STD system. Finally, we match these requirements to the described scenarios.

Scenario: Media archive search. Professional media archives can be quite large in terms of the amount of audiovisual data that is stored. For example, every day, the French national audiovisual archive (INA) stores over 1350 hours of data from a range of TV and radio stations [49]. It is obvious that such large amounts of audiovisual material cannot be annotated manually. In a joint experiment with archivists from a large broadcaster, the authors in [61] have found that Spoken Term Detection is a viable means for retrieving documents from a radio archive, and that it can successfully be embedded in the everyday workflow of the archivists.

There is a wide range of different archives, both in terms of type of content and size of content that is stored in archive. The type can range from professionally recorded content in archives of professional broadcasters to arbitrary user generated content in Internet video portals. The same is true for the size of the archive, which can range from a few hundred hours in a program-specific archive to millions of hours in large-scale professional TV and radio archives as in the case of INA. However, in most cases, the actual STD run will not be carried out on the complete large archive, as existing

formal metadata can be used to restrict the search to a reasonable sub-corpus (e.g., search only within a relevant time period or on relevant broadcasting stations).

Since 2007, we have worked with a wide range of different institutions that have access to small, medium and large-scale media archives. From 2007-2010, we contributed STD technology to the VITALAS project (Video & image Indexing and reTrievAl in the LArge Scale), a FP6-EU Project with the participation of l'Institut National de l'Audiovisuel (INA) and Institut für Rundfunktechnik (IRT) as end user partners. Within this project, we obtained requirements for large scale Spoken Term Detection on a large scale evaluation corpus of 10,000 hours of video data [2], and received direct feedback on our work from actual archivists.

Our approaches were partly deployed in several commercial contexts, where we could further refine the set of typical requirements given below. In 2009, our Spoken Term Detection system was used by the first German broadcaster ARD to search political speeches from the ARD archive during the German national election campaign ([28], see figure 2.1). In 2010, we deployed a speech search system for the archive of popular science show Galileo broadcasted by a commercial German TV station ([29], see figure 2.2). From 2010 to 2011, we built a Spoken Term Detection system at the ARD Mediathek, which enables end users to search in the transcripts of clips and allows for cross-linking of video citations with social networks ([30], see figure 2.3). This system was also selected for demonstration at the IEEE ASRU workshop on Automatic Speech Recognition and Understanding [97].

Scenario: Media monitoring. Another very interesting application of STD is continuous monitoring of TV and radio channels. For example, this could be used by companies to analyze the media coverage of their products. Here, the STD system must detect a set of specified keywords in the speech track of selected TV and radio programs. Such a monitoring system has the following key characteristics: First, we know that the set of keywords is known at indexing time, and can be automatically searched in the ASR output. However, the set of keywords can change at any time during the lifetime of the system (e.g., if the name of a new product must be detected). The retrieval latency must be low, i.e., if a spoken word was broadcasted at time t, the system must detect this occurrence within a small time period δ of a few minutes, such that it is reported no later than $t+\delta$. Hence, the automatic speech recognition must continuously produce word-by-

Figure 2.1.: Screenshot from ARD Web-Duell (2009), using Spoken Term Detection from Fraunhofer IAIS.

word output, and more accurate two-pass decoding approaches cannot be applied. As an alternative, one could segment the video stream into small yet acoustically homogeneous chunks (e.g., following the approach in [17]), and perform the multipass decoding on each chunk individually. Hence, the amount of data that needs to be searched in a single STD run is relatively small. Even with a relatively large chunk size of 15 minutes and a request to monitor 50 TV stations in parallel, only 12.5 hours of data would need to be searched for each keyword.

Next, we will describe a set requirements for STD systems, which will then be matched to the scenarios described above.

Requirement: High STD recall. The STD system should be able to locate as many query occurrences as possible. This includes approaches which enable the user to search for any word he can think of at the time of the query, not only for a fixed set of words determined at the time of building the models. At search time, the system should be able to cope to a certain extent with errors from the automatic speech recognition, such that incorrectly transcribed words can still be found.

Requirement: High STD precision. In some scenarios, a high precision of the STD results provided by the system is important. This is essential if the result set is large

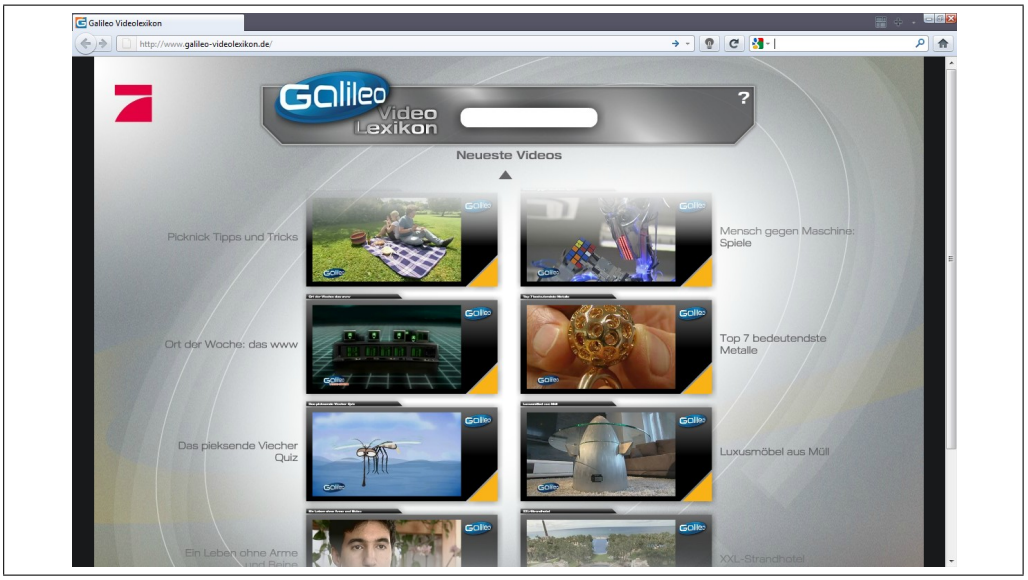

Figure 2.2.: Screenshot from Galileo Videolexicon (2010), using Spoken Term Detection from Fraunhofer IAIS.

(e.g., when searching for frequent terms in very large media archives). Of course, this requirement competes with high STD recall, and often only one of the two requirements can be fulfilled.

Requirement: Time and space efficiency. Despite the ubiquitous availability of CPU power and storage, time and space efficiency are major requirements for STD systems that handle large scale archives. A decrease in efficiency directly increases the hardware costs required for setting up the actual STD system, and approaches with very low efficiency will not be applicable to large-scale data sets.

Requirement: Flexibility The ultimate target of STD is to deliver both complete and precise result sets, thus aiming for both high precision and recall at the same time. This cannot be achieved in many scenarios, especially if the data is acoustically and linguistically complex (see section 3.3). However, the system should allow for easy adaptation to new scenarios, domains and information needs. System adaptation should be possible at two different stages:

1. Adjusting the *indexing system*, for example towards the characteristics of the data that will be indexed. Data that is acoustically or linguistically more challenging might need more complex STD methods than professional planned speech in a

Figure 2.3.: Screenshot from ARD Mediathek (2011), using Spoken Term Detection from Fraunhofer IAIS.

studio background.

2. Adjusting the *search system*. For example, consider a user who is using an STD system for his media archive. He wants to locate a particular document (known-item search), and cannot find it with the baseline configuration of the system. In this case, the system should enable the user to increase recall at query time with as little STD precision loss as possible.

Within the scope of this thesis, we will investigate several approaches to meet the described requirements. Table 2.1 indicates which chapters are most relevant for each aspect.

Table 2.1.: Spoken Term Detection system requirements.

Requirement	**Relevant chapters in this Thesis**
High STD recall	3, 4
High STD precision	5
Efficiency	6
Flexibility	7

Based on the description of the use cases and the proposal for different requirement aspects, we summarize the actual requirements for the individual scenarios in table 2.2. The symbol + indicates that the corresponding requirement has a low importance in the given scenario, while +++ indicates a major focus.

In the media monitoring use case, we can assume that for user satisfaction, result completeness is much more important than result precision. Response time per hour of data and required storage do not plan an important role, as the total amount of data to be searched is relatively low in this scenario. When searching media archives, retrieval efficiency becomes more important, especially for large archives of up to 100,000 hours of data. Here, compact indices with fast access to the STD results are mandatory. Moreover, STD systems for large archives should be rather configured towards precision, However, flexibility at search time is needed, especially for known-item search. If a user needs to find a specific document of which he knows that it is in the archive, he will tolerate larger response times, hence he must be able to tune the STD system on-the-fly towards recall at the cost of precision.

In the experimental evaluation, we will investigate to which extent our proposed methods are able to cope with the given requirements, and optimal system configurations for the individual scenarios will be provided in chapter 7.

Table 2.2.: Spoken Term Detection requirements for selected application scenarios. The number of + symbols indicates the importance of a specific requirement for a given scenario.

Scenario	**Precision**	**Recall**	**Efficiency**	**Flexibility**
Media monitoring	+	+++	+	+
Small media archive (up to 1000 hours)	++	++	++	+++
Large media archive (up to 100,000 hours)	+++	+	+++	+++

2.3. Related Work in Spoken Term Detection

This section describes the current state of the art for ASR-based Spoken Term Detection. First, we describe the evolution and current state of the STD research community, which manifested itself after the initial NIST STD evaluation in 2006. Then, we give a structured overview of current approaches to STD, covering all aspects that are relevant

for the scope of this thesis. Finally, we identify a set of gaps in the current state of the art, and motivate our approach to holistic STD.

2.3.1. The Spoken Term Detection Research Community

Historically, the task of detecting keywords in spoken utterances was referred to as *keyword spotting*. A classic example based on the standard ASR architecture is [116]. Here, the authors build Hidden Markov Models for each keyword that needs to be detected, and construct a garbage model (or *filler* model) that is used to model all other spoken words. In [98] we have shown that this approach yields reasonable keyword spotting results in challenging acoustic environments, even if resources for training the phoneme models are limited. Despite recent improvements in discriminative keyword spotting [56], a major drawback of this approach remains: the keyword spotting system needs prior knowledge about the query set during indexing.

For tasks which are more oriented towards ad-hoc Google-like searches in already indexed data, this approach was overcome by the success of word-based speech recognition. Back in 2000, after the successful TREC evaluation [34], Spoken Document Retrieval was declared "a solved problem": a large scale retrieval experiment showed that the output from word-based ASR on the speech track could be successfully used to retrieve relevant documents from a large broadcast corpus. As this was even possible for high ASR rates around 50%, there was no obvious need for further research in this direction.

However, the TREC evaluation ignored a key drawback when using word-based LVCSR as the only source for SDR: typically, state-of-the-art word speech recognizers depend on a finite decoding lexicon, i.e., all words that can be decoded (and retrieved) must be known a priori at indexing time. This opened up the way for a large variety of new approaches to overcome the so-called open vocabulary challenge: how can SDR systems retrieve spoken utterances, where the most important key words that influence retrieval and ranking are not part of the word decoding lexicon?

It is natural to decouple this challenge from the actual SDR task, and focus only an open vocabulary speech search as an auxiliary technology for SDR (see section 2.1. In 2006, NIST first used the notion of *Spoken Term Detection* for the large-scale vocabulary- and topic-independent localization of written queries in spoken content [82], and initiated a corresponding evaluation. The evaluation was carried out on English, Arabic and Mandarin Chinese corpora, including broadcast news data and conversational telephone speech recordings. It revealed that LVCSR is indeed suited for in-vocabulary speech search, however, efficient, accurate and flexible large-scale open-vocabulary speech retrieval is still in its infancy [26].

In 2007, the first Workshop on *Searching Spontaneous Conversational Speech* was held in conjunction with ACM SIGIR, with the goal of bringing together researchers from different communities such as speech processing or information retrieval [23]. Since then, the workshop has become a major event for the STD community, and it was held again at SIGIR 2008 [57], ACM Multimedia 2009 [62] and ACM Multimedia 2010 [63]. In 2011, a Special Interest Group on *Speech and Language Indexing for Multimedia (SLIM)* was founded within the International Speech Communication Association (ISCA), with a focus on "Spoken content retrieval and spoken term detection for multimedia collections"[2].

Compared to speech recognition research, the entry cost for new research teams is relatively low, as new STD approaches can be built on top of existing ASR systems. Hence, since 2006, more and more research groups have become involved in STD research. Spoken Term Detection research is crossing the borders of disciplines like speech recognition, linguistics and and information retrieval, and new results are published at all corresponding major conferences such as IEEE ICASSP, ISCA Interspeech, ACL HLT or ACM SIGIR.

2.3.2. Approaches to Spoken Term Detection

Searching for written keywords in spoken content has a long tradition in the speech community, and a wide range of approaches has been studied to cope with this problem. Based on their key characteristics, the field can be divided into two different directions of research:

- Template-based approaches such as [45] or [108], where an acoustic template of the query is obtained and matched with the audio signal of the corpus. Such approaches are typically language-independent.
- Language-dependent approaches which use the output from automatic speech recognition to locate written queries.

The choice of approach clearly depends on the actual STD scenario. For example, consider a multi-lingual environment, where STD is required for different under-resourced languages where no resources are available to built complex ASR systems (for instance, when searching in the archive of the Max-Planck-Institute for Psycholinguistics, which currently contains over 50 Terabyte of recordings from all over the world [118]). Here, template based methods can be applied without additional adaptation cost in the same

[2] http://www.searchingspeech.org/

manner across all occurring languages. However, such pure acoustic methods naturally suffer from lower retrieval rates compared to the more informed ASR-based systems, which can exploit language information by means of language models and decoding dictionaries.

Within the scope of this thesis, we focus on STD for media monitoring and media archive search, and can assume prior knowledge about the language of the spoken content. Hence, in the following, we can narrow down the description of the state of the art to those STD approaches which exploit ASR output, and our proposed view on holistic Spoken Term Detection will be built on top of these techniques. In this area, the STD community investigates several related topics:

- Using subword models to overcome the vocabulary dependence of classic speech recognizers.
- Applying error compensation at indexing and query-time to cope with high subword ASR error rates.
- Investigating the applicability of STD in selected scenarios, e.g., with respect to time and storage requirements.

Vocabulary Independent Spoken Term Detection

Baseline STD systems employ large vocabulary continuous speech recognition (LVCSR) for generating a word transcript, where the query can be searched on the word level. While LVCSR has reached a high level of accuracy in many domains, it is obviously not the most suitable solution for STD due to its inherent dependency on a fixed recognition lexicon. This is a major source for search errors, as the system can never detect queries which contain an out-of-vocabulary (OOV) word. A popular approach to overcome this challenge is to apply subwords instead of words as the decoding unit, where the set of subword units is finite and known a priori [81]. Queries are then broken into subword sequences, which are searched in the subword output of the ASR decoder.

In recent years, various units have been investigated, including phonemes [113], syllables [60] and data-driven subword units [10, 47]. Due to less constraining language models, subword systems typically suffer from lower ASR accuracy compared to word-based systems. Moreover, the subword representation of a query contains more (smaller) tokens that must be matched in the subword transcript. If only one of these tokens is incorrect, the matching will fail. Combining the results from word and subword decoding into hybrid STD systems can further increase the overall retrieval performance [1].

Various languages have been in the focus of vocabulary independent STD. English, Mandarin Chinese and Arabic have been used within the NIST STD Evaluation in 2006. The corresponding evaluation corpus is available via LDC, hence many research groups evaluate their approaches in one of these languages. However, there has been limited work on STD in other languages, including Turkish [88], Japanese [50] or Spanish [103]. In 2011, the MediaEval Benchmark Initiative proposed the *Spoken Web challenge*[3], which particularly targets the Spoken Term Detection community. The corresponding data set contains spontaneous speech from English, but also from Hindi, Gujarati and Telugu, and is hence rather suited for language-independent techniques. For German, only little work has been published on STD. While [40] and [115] provided first insights into open vocabulary spoken document retrieval on German data, [60] were the first to investigate the principle use of syllables for German STD. Still, most approaches have not been investigated on other languages than English.

Error Compensation in Spoken Term Detection

With an increasing number of word and subword ASR errors, more and more occurrences of user queries will not be found by the search system. As a remedy, error-tolerant indexing and retrieval approaches can be applied, which increase STD recall while not sacrificing too much precision.

Many systems do not only store the 1-best output of the recognizer, but also competing hypotheses in the form of lattices [93, 99]. Instead of retrieving from unconstrained lattices, more compact representations such as word confusion networks [44] or Position-Specific Posterior Lattices [15, 86] have been proposed, which achieve comparable high STD accuracy [87], where PSPL performs slightly better than word confusion networks. However, the authors in [55] note that the benefits of PSPL might be "coupled to [...] low-frequency search queries and low-WER environments". Recently, lattice extensions have been proposed which integrate lexical adaptation and subword decoding [4].

At retrieval time, the word query is typically broken down to a canonical subword sequence. For example, the phoneme sequence for a word query could represent the standard pronunciation of the query. Then, error-compensating algorithms can be applied to allow for deviations between the query and the the subword transcript [60] to cope with subword ASR errors and pronunciation variations. As an alternative, the authors is [72] have successfully expanded the subword query with likely deviations, which are then also searched in the subword transcript. A combination of both ideas further

[3] http://www.multimediaeval.org/

improves the results [114].

Only little work has been published on efficient approximate retrieval from lattices. Several groups rely on the extraction of subword multigrams [102, 114], which are all matched with the subword query. In [104], the authors describe a method that allows for fast approximate matching on lattices, however only on unconstrained output from phoneme ASR and without explicit modeling of the different search spaces covered by the two approaches as described in section 4.1.

For German, error compensation has been studied in an early work in the field of Spoken Document Retrieval on 1-best phoneme sequences by [115]. Moreover, syllable-based approximate search on 1-best syllable ASR output was investigated in a small-scale evaluation in [60], which forms the baseline of section 4.3.

Applicability and Scalability of Spoken Term Detection

An important issue in all mentioned topics is the efficiency of the respective STD approach: as STD retrieval is supposed to be executed on demand by actual end users, it must operate in reasonable time even on very large corpora. Depending on the application scenario, different STD approaches can be suitable. While monitoring applications in the security domain might focus on recall, media archive search systems for end users would require more precision-oriented systems.

Despite the importance for practical use of STD systems, dedicated scalability investigations and evaluations on large corpora are rare. In [52], the authors use exact match of phoneme-n-gram models to scale up to 2,000 hours with response times below one second. However, reasonable STD accuracy could only be obtained when applying a more expensive multistage approach, which resulted in higher response times.

Another approach for efficient approximate subword STD based on metric subspace indexing was studied in [53] on a set of hundred Japanese lecture recordings taken from [69]. Compared to continuous approximate phoneme matching, the authors achieve a search time reduction of over 30% absolute at equal STD accuracy. However, the proposed approach still requires about 200ms on a relatively small corpus.

In [54], suffix arrays were used for fast approximate search on phoneme ASR output. Similar to text retrieval, suffix arrays and suffix trees are particularly suited for fast approximate substring matching on a large data set. In the given implementation, approximate search with high similarity thresholds on a simulated corpus of 10,000 hours of data yielded very low response times. However, reasonably high STD accuracies could only be achieved at low similarity thresholds, which in turn causes high response times, especially for longer queries (over 16 seconds for a query with 18 phonemes).

An interesting idea similar to our proposal [78] was published in [121]: the authors use a filter approach to efficiently pre-select the most promising utterances that most likely contain the keyword, and then apply a more expensive retrieval technique on the remaining set of utterances. However, the idea is only used to filter word and phoneme lattices for *exact* lattice matching. In [101], the authors use a similar two-stage approach on syllable confusion networks. They retrieve all networks that contain one of the query syllables, and apply an error-tolerant matching between all filtered networks and the query sequence. A drawback of this approach is the relatively large number of initial networks that are selected for approximate matching, which in turn can cause high response times on large corpora.

2.3.3. Limitations of Current Approaches

In the following, we describe limitations of the current approaches to STD, and identify the gaps that need to be bridged for enabling holistic, scalable and flexible STD.

First, we observe that there is no systematic and exhaustive investigation of the different units for German STD. The optimal unit size differs from language to language. For example, agglutinative languages such as Turkish will benefit from relatively large units such as morphs [88], while for other languages like English, phonemes perform well [113]. Moreover, typically there is no distinction between recognition and retrieval unit. However, it could be interesting to investigate the effect of breaking down decoding units to smaller retrieval units. Hence, we will study different combinations for recognition and retrieval unit in section 3.2.

For error compensation, there is no explicit analysis of the different error sources that occur in subword STD, which in turn require dedicated methods for error compensation. An analysis of the STD error spaces, and methods for explicit handling of the corresponding errors will be given in chapter 4. Again, only little work on state-of-the-art error compensation has been published on a German STD task. Based on prior work from [60], we have investigated and published on German lattice STD [77] and approximate search on German subword transcripts ([96]). In [78], we have proposed a novel efficient and effective approach to hybrid approximate lattice search, which explicitly merges the STD error search spaces.

Current state-of-the-art approaches ignore a major source of information: at search time, the STD system has access to more information about the query than at indexing time. In chapter 5, we propose a novel approach to exploit this knowledge in order to verify whether a putative STD result is correct or not, which we have first published in [95].

Scalability has received the attention of the STD community only recently, but the proposed techniques provide punctual solutions, and where provided independent from actual application scenarios. However, different STD application scenarios can have different scalability requirements. Yet, an analysis of STD requirements in major scenarios and a relation of requirements to system configurations is still missing. A holistic view using different search strategies for different scenarios is not covered at all. Based on our scalability investigations in chapter 6, we will propose best practices and system configurations for selected STD scenarios in chapter 7.2.

In summary, we would like to close three gaps in current STD research:

- Investigate the exploitation of external query knowledge that is only available at search time, and study its interplay with state-of-the-art methods for error compensation.
- Derive scalable and flexible variants of the proposed hybrid approaches for large-scale STD, and provide best practices for selected STD application scenarios. Within this thesis, we will provide these search policies for two selected STD application scenarios: media monitoring and speech search in large media archives.
- Provide comprehensive STD investigations on German data using state-of-the-art approaches to the STD research community. To reach this goal, we have built a German evaluation corpus [6, 94], described a vocabulary independent STD baseline [96], and approached error compensation [77, 78] as well as a novel paradigm for STD result verification [95] on the German data.

2.4. Evaluation Methodology

Our evaluation methodology follows the evaluation plan developed by NIST for the STD evaluation in 2006. We decided to adhere to the standards provided by NIST for two reasons:

- The STD evaluation procedures are well-defined, and they have are capable of evaluating whether an STD system can be used in the specified scenarios.
- The NIST speech group has a long experience and outstanding reputation in evaluating results from state-of-the-art research in speech technology. The first NIST evaluations in the field of ASR date back to 1996. Currently NIST carries out a wide range of large scale benchmarks, including diverse topics such as speaker recognition, ASR or machine translation.

This section introduces a related set of metrics widely used in the STD community, and describes modifications which were necessary due to characteristics of the German language. No German STD corpus exists to date which could be used to evaluate speech search on heterogeneous and complex TV data. We describe the design and creation of *DiSCo*, a new German broadcast speech corpus, which is used for the evaluation in the thesis at hand.

2.4.1. Evaluation Metrics

A wide range of metrics exist which could be used to evaluate STD systems. The approaches presented here rely on automatic speech recognition, hence it is natural to asses the quality of the ASR output. Here, the standard quantitative measure is the *Word Error Rate (WER)*. It is estimated from the alignment between the reference transcription and the hypothesized output from the ASR decoder:

$$WER = \frac{S + D + I}{n} \tag{2.5}$$

where S is the number of substitutions, D the number of deletions and I the number of insertions in the alignment, and n is the number of reference words. An optimal alignment with minimal number of edit operations can be obtained using dynamic programming, a reference implementation is available from NIST[4].

The quality of a subword transcript is assessed in a similar fashion. First, the reference transcript is broken down into subwords, such as syllables or phonemes. Then, the subword ASR output is aligned with the subword reference. Similar formulae are then used for estimating the *Syllable Error Rate (SER)* and the *Phoneme Error Rate (WER)*. Note that the reference subword transcript is not necessarily correct with respect to the actual spoken subwords, as it contains the canonical transcription obtained from the grapheme-to-phoneme conversion (section 4.1 describes this phenomenon in more detail). Hence, the actual SER or PER values can be lower than those estimated on the canonical subword transcriptions. Many German speakers delete the final *t* [123], which for example occurs in the German conjunction *und - and*. A syllable decoder would tend to output *U_n_* instead of the canonical transcription *U_n_*, which is correct from an acoustic point of view, but incorrect when using the canonical syllable transcription as a reference. This can be substantially increase SER, as the conjunction typically has a high frequency.

The quality of the lattice output is often assessed with the *Lattice Word Error Rate*

[4] http://nist.gov/itl/iad/mig/tools.cfm

(LWER), e.g., in [4]. The LWER is equal to the lowest WER that can be obtained from any path through the lattice. Similar metrics can be used for subword lattices (*Lattice Syllable Error Rate (LSER)* and *Lattice Phoneme Error Rate (LPER)*).

The actual effectiveness of a retrieval algorithm is typically evaluated with two aspects:

1. Completeness: given a set of queries, does the system retrieve all occurrences of all queries in a given corpus?

2. Correctness: given the result set for a set of queries, how many results are correct?

Completeness can be measured with the *Recall* metric R:

$$R = \frac{TP}{N_{Ref}} \tag{2.6}$$

where TP is the number of true positives (i.e., correct results) and N_{Ref} is the number of correct occurrences in the reference transcription. In a similar fashion, we measure correctness with the *Precision* metric P:

$$P = \frac{TP}{TP + FP} \tag{2.7}$$

where FP is the number of false positives (i.e., incorrect results). Typically, recall decreases if a system is tuned towards higher precision and vice versa. For illustrating the performance of a system while varying the system configuration during tuning, Receiver-Operating-Characteristic (ROC) curves can be used, which is a 2-D plot of Precision versus Recall. An alternative to ROC are *Detection Error Tradeoff (DET)* curves, where the axes are scaled by their normal deviates [75].

The use of ASR word error rate is an obvious indicator for the overall performance of a speech search system, and unsurprisingly, it was found to strongly correlate with retrieval performance [107]. However, the word error rate is not necessarily an optimal target for system optimization. In [85], the authors investigated whether optimizing the ASR decoder towards low word error rate also leads to an increase in retrieval performance. They observed that tuning the decoding parameters[5] towards lower WER can also lead to a decrease in mean average precision. This might be caused by the fact that important content words often have a low frequency, and their impact on the word error rate is low. On the other hand, the parameter tuning privileges highly frequent filler words, which have no impact on retrieval performance.

[5] language model scaling and word insertion penalty

For the 2006 STD evaluation, NIST proposed the *Term-Weighted Value* as an evaluation measure for STD [82], which aims at un-biasing the evaluation by removing the influence of individual query frequencies . Let Q be the set of actually occurring queries that shall be detected by the STD system. From the result set of a given STD system, we can estimate the probability of missing a certain query $q \in Q$ with

$$p_{\text{miss}}(q) = 1 - \frac{TP(q)}{N_{Ref}(q)} \tag{2.8}$$

where $TP(q)$ is the number of correct hits produced by the system for q, and $N_{Ref}(q)$ is the number of reference occurrences of q. As defined by NIST in the STD evaluation plan, the probability that a given system produces a false alarm for a certain query q can be estimated with

$$p_{\text{FA}}(q) = \frac{FA(q)}{\text{possible number of trials}} \tag{2.9}$$

where $FA(q)$ is the number of false alarms produced by the STD system for q. The number of possible trials can be approximated with the total length of the corpus in seconds. Averaging over all terms we obtain two adjusted indicators for the two aspects *completeness* and *correctness*:

$$p_{\text{miss}} = \frac{1}{|Q|} \sum_{q \in Q} p_{\text{miss}}(q) \tag{2.10}$$

$$p_{\text{FA}} = \frac{1}{|Q|} \sum_{q \in Q} p_{\text{FA}}(q) \tag{2.11}$$

In order to obtain a single estimate for measuring the overall system performance at a given system configuration, NIST proposed to use the *actual term-weighted value* $ATWV$, which is estimated using

$$ATWV = 1 - \frac{1}{|Q|} \sum_{q \in Q} p_{\text{miss}}(q) + \beta \cdot p_{\text{FA}(q)} \tag{2.12}$$

where the false alarm probability of a certain term is weighted with the constant cost β:

$$\beta = CV \cdot (\frac{1}{p_{\text{prior}}} - 1) \tag{2.13}$$

The cost-value ratio CV indicates to which extend the user is willing to accept false

alarms in order to obtain more true positive hits. The value should be chosen according to the given use case - for example, monitoring applications in the security domain are likely to be more interested in minimizing the miss probability. In the original NIST evaluation, the cost-value ratio is set to 0.1 (i.e., the cost for producing a false alarm is assumed to be a tenth of the cost when missing a term). It is multiplied with the number of possible terms that could cause a false alarm. This value can be obtained from the inverse of the prior probability p_{prior} of a term, which is set to 10^{-4} in the original evaluation plan. The same configuration is used within the evaluation of this thesis in order to ensure comparability with other publications.

The ATWV measures the system performance for a particular configuration, i.e., at a certain actual confidence threshold. This is useful for comparing systems that share components, which can then use the same configuration (e.g., equal thresholds). On the other hand, NIST also proposed the *maximum term-weighted value MTWV*, which is the maximum ATWV that can be achieved while varying the system parameters.

Other single-point metrics for evaluating STD systems include the F_1 score (i.e., the harmonic mean between recall and precision) [96] or a Figure of Merit (FOM) focusing on false alarms per hour of data [111]. In [80], we found that the choice of STD evaluation metric has a direct impact on the characteristics of the system: MTWV using the default NIST configuration is biased towards more exact search approaches which are rather required in end-user search scenarios, while FOM is biased towards approximate, recall-oriented search approaches needed in the surveillance domain. Hence, for the scenarios targeted within the thesis at hand, we will use the standard TWV measures developed by NIST, keeping in mind that the resulting systems might need to be re-configured for other more recall-oriented domains such as surveillance applications.

The official STD evaluation plan defined that only exact orthographic matches are considered to be correct hits, i.e., partial substring matches are considered to be false alarms. This is reasonable for many cases, such as the hit *cat* for the actually spoken word *catalogue*, which is obviously a retrieval error. However, in some cases this strategy is debatable, e.g., when considering singular and plural forms such as *fence* and *fences*, which could fulfill the same information need. Hence, optimizing a system towards *not* finding the plural when searching for the singular might hurt the overall system performance. The impact of this evaluation requirement for STD on German data is even higher due to complex ending variations caused by flexions of verbs, adjectives and nouns. For example, four variations of the word *Zaun (fence)* exist (Zaun, Zauns, Zäune, Zäunen), which all are reasonable results for a corresponding search for *Zaun*. Moreover, German compounding causes additional false alarms during evaluation within

this restriction. An example is a search for *Wirtschaft (economy)*, which could easily return results such as *Marktwirtschaft (market economy)* or *Binnenwirtschaft (national economy)*. Within the limits of the NIST evaluation plan, these would be considered to be false alarms, but we prefer to optimize our German system towards detecting such terms rather than ignoring them. Following this rationale, we will accept compound words containing the query word and flexions of the query word as true positives. Partial matches are verified manually such that correctly labeled false positives (cat - catalogue) remain false positive.

An additional metric is required in order to assess the efficiency of an indexing and retrieval approach. The NIST STD evaluation plan requires to report the *Term Search Speed (TSS)*, which measures the time it takes the system to respond to a certain query on a given STD evaluation corpus. However, with only eight hours of data for English, the NIST STD corpus is too small for efficiency evaluation. Others have shifted the focus of the metric, such that it gives the search speed per hour of data [89]. Within this thesis, we will estimate the efficiency of a search approach by averaging the search speed for a single query over a large query set, and give the results per hour of data estimated on a large artificial data set.

For the evaluation in the thesis at hand, we will use the following metrics to assess the quality of an STD approach:

- The speech recognition error rate of the 1-best transcript in order to assess the quality of the ASR process.
- Recall R and precision P of the retrieval approach in order to individually assess completeness and correctness of the method.
- *ROC* curves for analyzing the tradeoff between recall and precision.
- $MTWV$ as a single-point metric for assessing the overall system performance.
- $ATWV$ for comparing systems at a fixed system configuration.
- The average search speed per query per hour of data as an indicator for the search efficiency.

Using recall, precision and ROC curves will reflect the actual frequency distribution of the queries, while MTWV and ATWV provide adjusted, single-point metrics for assessing the performance of a particular system configuration.

2.4.2. Evaluation Corpora

In this section, we present the design and preparation and of *DiSCo (Difficult Speech Corpus)*, a new German corpus for evaluating various speech technologies on challenging broadcast material [6].

The motivation for building the corpus is the lack of available data for in-depth evaluation of the proposed approaches. No publicly available corpus for evaluating Spoken Term Detection on heterogeneous German broadcast data exists at the time of writing this thesis. Moreover, no resources are available to the public which could be used for evaluating German ASR on heterogeneous Broadcast News data. Past German ASR evaluations such as [46, 76] used rather small and homogeneous data sets.

We considered using an available evaluation set stemming from an internal project with two public German broadcasters [59], which has been already used in the past to assess the quality of a prototype speech search system [96]. The corresponding STD evaluation queries have been chosen by professional archivists, hence they represent a relevant query set for the media archive scenario. However, the evaluation set consists only of radio data recorded in 2004 and 2005, and it has been collected from only a few different radio shows. It contains a substantial amount of clean planned and spontaneous speech, but additional acoustic artefacts (such as background music or background speech) are rare. No information about the dialect of the speaker is available. Hence we decided to build a new evaluation corpus, which should be representative of a variety of interesting TV and radio broadcast formats.

Selection of programs. We selected a range of programs from both public and private German broadcast stations, such that the corpus contains a balanced mix of both planned and spontaneous speech under various acoustic conditions. Our selection includes the following genres:

- **News.** The composition of broadcast news shows is rather formal, i.e., each show has a similar structure. A major part of the speech is uttered by the anchor person, who reads a prepared text and is a professional speaker. Most of the speech is recorded without any background noise in a high quality studio environment. This can be used as a baseline, where the speech search system should yield the best results. However, news shows also contain many sound bites such as interviews in more complex acoustic environments, which are particularly interesting for Spoken Term Detection. We collect both news from public and commercial broadcasting stations.

- **Political talk shows.** This type of program contains numerous interesting quotations from politicians or other important celebrities. Often, the shows have just one topic, with an up-to-date but rather limited vocabulary. The sound quality is high, as the participants often use close-talk microphones, but there is frequent background noise (other speakers interrupting or commenting on the main speaker, background applause). Speech is mostly from professional speakers, but often highly spontaneous.

- **Popular science shows:** Here, most of speech is planned and recorded in a professional studio environment. Often, background music is added to the speech signal for setting the atmosphere of the show. The vocabulary is highly specialized on the current topic of the recording.

- **Regional reports:** The documents in this set are recordings from a local Bavarian magazine, which is dedicated to regional stories. It contains reports and interviews, and often the speech has a strong Bavarian dialect.

- **Foreign affairs reports:** The documents in this collection contain reports about foreign countries all around the world. The speech parts contain numerous *dubbed* utterances, which is typical for the German TV program: the original voice of an utterance in a foreign language is audible relatively low in the background, and a louder, time-synchronous translation in German is added on top of the original signal. An additional complexity in this type of program is the vast vocabulary, comprising names of rather small geographical entities (cities, rivers, or other landmarks) which occur in the reports.

- **Sports shows:** The material from the sports domain is particularly challenging for speech recognition, even with well-adapted state-of-the-art systems [27]. The vocabulary is complex and ever-changing due to the various sports disciplines and the changing active athletes. Moreover, speech is often uttered in difficult acoustic conditions, e.g., with heavy background noise from the audience in a stadium. Interviews with athletes are frequent and pose additional challenges such as non-native non-professional German speech.

Table 2.3 gives an overview of the raw material that was collected for annotation. All documents were recorded from a high quality DVB-S signal.

Corpus annotation. Each broadcast recording was manually annotated by a single annotator. We asked the annotators to adhere to the following standards:

Table 2.3.: Raw recordings used for the DiSCo corpus.

Program	**Duration (hh:mm)**	**Percentage**
News (public)	02:42	17%
News (private)	01:11	8%
Political talk shows	04:44	30%
Popular science shows	00:29	3%
Regional reports	01:29	9%
Foreign affairs reports	03:04	20%
Sports commentaries	02:00	13%
All	15:39	100%

- Segment boundaries should be inserted at speech pauses and at speaker changes.
- Segments without speech are labeled as non-speech with a special marker. Non-speech includes speaker noise such as laughing or coughing. Telephone speech in professional broadcast recordings is rare (except for call-in interviews in the radio domain), hence we asked the annotators to mark telephone speech as non-speech. Moreover, suitable German telephone corpora are already commercially available via ELRA[6]).
- Segments that contained foreign speech or speech that was otherwise indiscernible or unintelligible to the transcriber are not transcribed, but labeled with a special marker.
- Segments with cross-talk by two or more speakers are not transcribed, but labeled with a special marker.
- Compound words should be transcribed using a longest-possible-match instead of individual nouns (e.g., Gammelfleischskandal instead of Gammelfleisch Skandal).
- Words should always be transcribed using the correct orthography, even if the speaker is using a popular mispronunciation (e.g., *haben* instead of *ham* or *wichtig* instead of *wichtich*).
- Hesitations should be transcribed with a special marker #ä#.

[6]http://www.elra.info/

- Stutter and slip of the tongue should be marked with an asterisk at the beginning of the correct word. For example, if the speaker said *Trankstelle* instead of *Tankstelle*, the word should be transcribed with **Tankstelle*.
- In case of doubt, the annotator should discard the whole utterance and mark it as unintelligible.

As a result, we obtained almost 12 hours of transcribed speech utterances that can be used for evaluation. Segments that contained stutter or slip of the tongue were rare (only 351 occurrences), and we removed the corresponding utterances from the evaluation set. Table 2.4 gives an overview on the corpus statistics by looking at the individual programs. Both public and private newscasts have a very high speech portion of around 90%, and are well represented in the corpus. In contrast, only 60% of the discussion shows have been considered to be transcribable speech by the annotators, caused by long non-speech applause sequences between answers, cross-talk, or false starts and stutters. Only 532 utterances from the popular science show are available for evaluation, nevertheless, over 90% of these segments contain interesting acoustic challenges with professional speech over various backgrounds. Both sports shows and foreign affairs magazines are fairly well represented. The complete set of transcriptions sums up to about 120,000 running words, with a vocabulary of 15438 unique words. The manual utterance segmentation yielded an average segment length of 2.5 seconds.

Table 2.4.: DiSCo corpus by program.

Program	**Speech utterances**	**Duration (hh:mm)**	**Percentage transcribed**	**Transcribed words**
News (public)	3,306	02:22	88%	23,146
News (private)	1,286	01:04	90%	11,024
Political talk shows	4,065	02:51	60%	31,259
Popular science shows	532	00:24	83%	3,401
Regional reports	1,833	01:16	85%	12,281
Foreign affairs reports	3,636	02:15	73%	22,065
Sports commentaries	2,494	01:35	79%	16,372
All	17,152	11:47	75%	119,548

Compared to other evaluation corpora for German ASR, the material contained in the corpus is diverse, and its size is rather large. For example, [76] uses only recordings from a single public news program, summing up to 1.5 hours of data. The authors in [46] only

use news broadcasts (3.5 hours in total). In [83], a more heterogeneous collection of web videos, broadcast news and conversational telephone speech data is used for evaluation, yet the corpus is relatively small (3 hours).

For each of the 17152 speech segments, we asked the annotators to add labels for characterizing the speech and the acoustic conditions of the recording. Figure 2.4 illustrates the label hierarchy that was used during the annotation process. All speech segments contain a label whether the speech is planned, spontaneous or whether the annotator was undecided. Moreover, the annotators assessed whether the speech of a certain utterance contains strong dialect or not. We also asked to label segments with frequent background noises, such as music, background speech or applause. Other noises (such as stadium noise) were subsumed in a common noise class, which enables us to further detail the noise annotations at a later stage.

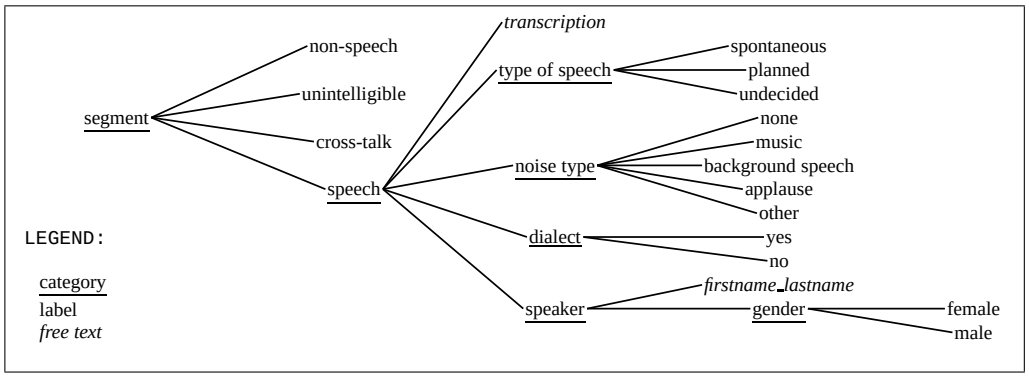

Figure 2.4.: Hierarchy of the DiSCo speech annotations, taken from [6].

DiSCo is not only designed for evaluation of ASR and STD, but also for Speaker Recognition experiments. For each utterance, the full DiSCo annotation set contains an additional label with the name of the corresponding speaker. Details about the speaker annotations can be found [6].

Based on the extensive annotation, we can decompose the corpus into individual subsets focusing on a particular challenge. In particular, we are interested in the following comparisons:

- Comparing planned to spontaneous speech. All other (such as dialect or background noises) should be eliminated.
- Comparing the influence of different background noises. While varying the acoustic background, we would like to keep the speech type as simple as possible (i.e., no

spontaneity and no dialect).

- Comparing non-dialect to dialect speech. German has a wide range of dialects, and their influence on ASR performance is well known [46]. We look at the influence on Spoken Term Detection when comparing planned non-dialect speech to planned dialect speech, both without any background noise.

The subsets which are needed for the corresponding experiments are listed in table 2.5. We obtained a sufficient amount of data for most of the major challenges. However, due to the restriction that only a single challenge per subset is allowed, some sets received only little data. The smallest subset is speech over applause with no background noise, which contains only 115 utterances. However data for this class is indeed hard to obtain, as speakers typically stop talking right after the applause begins. On the other hand, other classes contain large amounts of data that should be more representative of the corresponding challenge. A large part of the corpus (almost seven hours) is excluded from this detailed decomposition, as it contains various simultaneous challenges such as *spontaneous dialect speech over music*, but it can still be used for evaluations on the complete corpus. Experiments on the *all* data set should therefore give a good estimate of how well a certain approach performs on a real-life mix of German TV data.

Table 2.5.: DiSCo corpus by acoustic and linguistic challenge.

Subset	**Speech utterances**	**Duration (hh:mm)**	**Transcribed words**
Planned, clean	1,364	00:56	9,184
Spontaneous, clean	2,861	01:56	20,740
Planned, background speech	727	00:29	5,054
Planned, music	1,789	01:12	10,354
Planned, dialect	318	00:13	2,179
Planned, applause	115	00:06	994
Other (including mixes)	9,978	06:55	71,043
All	17,152	11:47	119,548

STD evaluation queries. A fixed set of queries is required in order to evaluate retrieval approaches for Spoken Term Detection on the DiSCo corpus. We base our selection of queries on two different sources of information: the query selection of the NIST

STD evaluation [82], and the query sets used in [96], as they were provided by actual professional broadcast archivists.

We used a semi-automatic approach for selecting a representative query set for the DiSCo corpus, consisting of two steps:

1. First, we applied the Term Selection Tool provided by NIST[7], which automatically extracted a set of queries from the training corpus.

2. In addition, five individuals were asked to manually select queries from the transcription text. These queries were merged with the automatically generated list, yielding a total set of 501 unique queries, which occur 2748 times in the complete corpus.

Table 2.6 shows the distribution of the queries among the individual DiSCo subsets.

Table 2.6.: Evaluation queries.

Data set	**Query Occurrences**
Planned, clean	268
Spontaneous, clean	427
Planned, music	319
Planned, background speech	89
Planned, dialect	54
Planned, applause	42
Other (including mixes)	1549
All	2,748

In the STD evaluation plan, NIST states that the search terms "will include single-word and multi-word terms, common and rare terms". Looking at the DiSCo query statistics, we find that our query set fulfills these requirements. Out of the 501 queries, 36% are single-word queries, and the remaining queries are composed of up to 5 words. Very rare queries that occur only once in the complete corpus make up 10% of the query set (i.e., 50 unique rare queries). On the other hand, the 50 most frequent queries already cover 60% of all query occurrences, so there is a good balance between frequent and infrequent terms.

Regarding the query lengths, we compare our selection with the queries from the German query set [96] instead of the official English NIST queries, as average English

[7] http://www.itl.nist.gov/iad/mig/tests/std/tools/

and German word lengths differ greatly due to the German compounding. Table 2.7 shows that the average length per query is almost equal across the two data sets, even when comparing the queries at the syllable or phoneme level. We also compared the distribution of the query lengths in both sets. Figure 2.5 shows that both sets have similar length distributions, and that in both selections, most of the queries that were selected consist of 6 to 15 phonemes.

Table 2.7.: Comparing DiSCo queries with [96].

Unit	**Queries in [96]**	**DiSCo queries**
Avg. words per query	1.5	1.8
Avg. syllables per query	4.8	4.8
Avg. phonemes per query	12.8	13.0
Named entities (people)	16.4%	17.4%
Named entities (places)	10.4%	13.6%

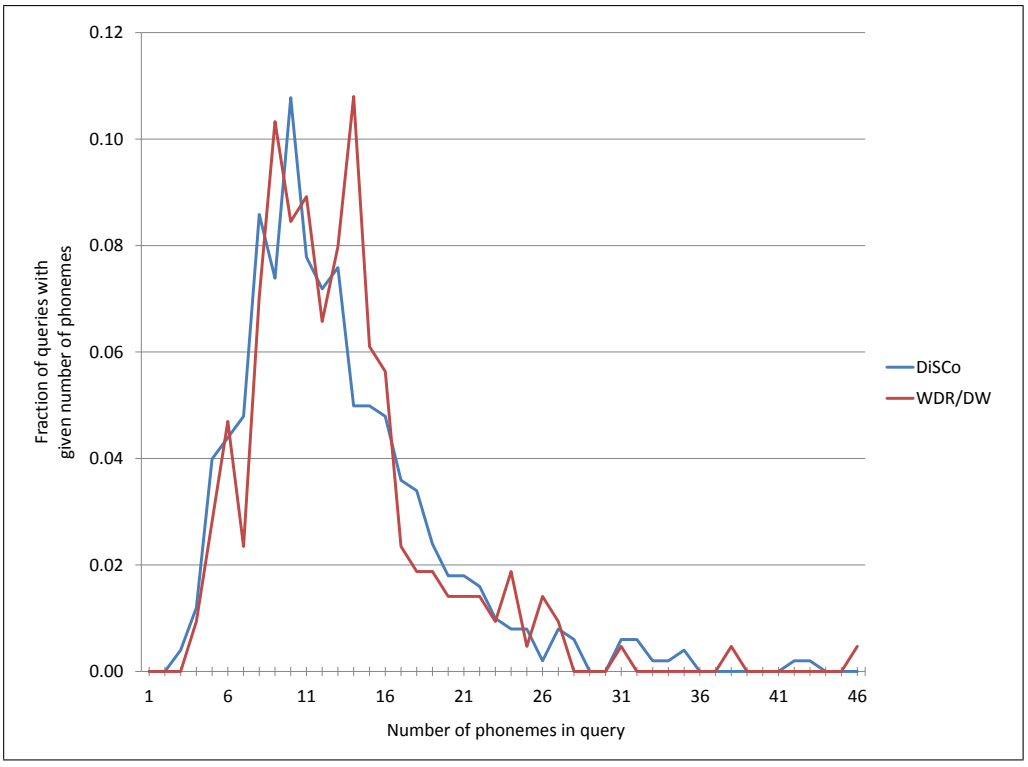

Figure 2.5.: Comparison of query lengths between the DiSCo query set and queries used in [96].

Table 2.7 also gives some interesting insights into the composition of the query set selected by the professional archivists. About 30% of the queries are proper names of people and places, which indicates the importance of this type of queries. A similar amount of such named entities are found in the DiSCo set.

2.5. Summary

Many interesting application scenarios exist for Spoken Term Detection. As they are inherently different in nature, it is obvious that the optimal approach for a specific application needs to be selected based on the specific requirements of the application scenario.

We have described two representative STD application scenarios (media monitoring and media archive search), and derived a set of required system characteristics. Several cooperations with the broadcasting industry revealed helpful insights into the actual requirements of end users.

While STD research has produced a range of interesting results in recent years, there are still many gaps that need to be bridged. Very little work has been published on German STD, and no comprehensive investigation of state-of-the-art approaches on German data exists. Moreover, no STD evaluation corpus comparable to the NIST evaluation set existed, which is a major obstacle since the construction of such a corpus is time-consuming and expensive. As a remedy, we have prepared DiSCo [6], a new corpus comparable to the NIST evaluation data, which allows for STD evaluation on German data.

Beyond the language aspect, we have identified several interesting limitations of the current state of the art that will be covered within the scope of this thesis. This includes explicit discrimination between decoding and retrieval unit (chapter 3) and explicit handling of STD error spaces (chapter 4). Moreover, we observed that current approaches do not exploit external query knowledge that is only available at search time (chapter 5). Finally, search scalability and scenario-dependent configuration of STD systems have not been in the focus of research so far (chapters 6 and 7).

3. Vocabulary Independent Spoken Term Detection

This chapter will present our system for vocabulary independent Spoken Term Detection on heterogeneous German broadcast data, which we have first published in [96], and recently described in more detail in [94]. We describe the baseline STD system using word-level automatic speech recognition (LVCSR), and present its evolution into a state-of-the-art system for speech search. Then, we investigate the use of different subword units to overcome the out-of-vocabulary problem.

Starting from a short description of the necessary theoretic background in automatic speech recognition, we describe our system for large vocabulary continuous speech recognition on German broadcast data that was developed within the scope of this thesis. The architecture of the system is presented, and a more detailed description of the parts which are relevant for STD is given, which includes the evolution of both acoustic and language model into the current state-of-the-art LVCSR system. The resulting 200,000 word ASR decoder was used for large scale automatic speech recognition in a range of STD-related projects, including several cooperations with the broadcasting industry (ARD Mediathek [30], Galileo Videolexikon [29], ARD Web Duell [28]), national research projects (including THESEUS[1] and TAT[2]), as well as large European research initiatives in the field of audiovisual library research (AXES[3], VITALAS[4]).

Next, we motivate the use of subword retrieval for vocabulary independent Spoken Term Detection, and describe the architecture of our subword-based STD system. We analyze the potential advantages and drawbacks of different subword units. Unlike other contributions in the field of subword STD, we explicitly distinguish between ASR decoding unit and STD retrieval unit.

Finally, section 3.3 presents an extensive evaluation of all described approaches, where we study in particular effects that are specific to the German language. We conclude

[1] http://www.theseus-programm.de/
[2] http://www.targeted-advertising.net/
[3] http://www.axes-project.eu/
[4] http://vitalas.ercim.org/

the chapter with best practices for subword-based vocabulary independent Spoken Term Detection on heterogeneous German broadcast data.

3.1. Baseline System for Word-Based Spoken Term Detection

We start our investigations into Spoken Term Detection from what many would consider the straightforward way of searching speech: converting the spoken content into a searchable word transcript. Typically, this transcription is obtained by applying techniques for automatic speech recognition.

Many approaches to word-based ASR have been investigated over the past decades, with state-of-the-art solutions which yield high accuracies for many interesting scenarios. Grammar-based techniques are typically used in highly constrained applications, such as voice portals in contact centers, where calling customers use their voice to navigate through the menu options and enter constrained information such as product numbers. By exploiting prior knowledge about the possible choices uttered by a client, a voice portal system can anticipate typical sentences that the client will use, and only allow for a small fraction of the possible word combinations. In a similar fashion, command-and-control applications make heavy use of prior knowledge to enable speech-driven application control even in adverse conditions, e.g., on motorcycles [117]. At the other end of the spectrum, flexible systems are geared towards optimal performance on surprise data, where the system has no prior knowledge about the next decoding task.

The speech that can be observed in the use cases illustrated in section 2.2 is typically not constrained at all. It can hardly be modeled by a fixed grammar, and in many applications the size of the used vocabulary is exceedingly high. A flexible system for *large vocabulary continuous speech recognition (LVCSR)* is required to transcribe the speech utterances.

Research on LVCSR systems has a long tradition in the speech community. For many years, progress of the core technologies has been monitored and fostered by the corresponding NIST evaluations, including the Broadcast News Recognition evaluations (1996-1999) and the NIST Rich Transcription evaluation (2003-present). While numerous different approaches to LVCSR exist, many systems that have been successful in the evaluations follow the same holistic statistical paradigm: Given an observed spoken utterance, the task is to find the word sequence w that has the highest probability of having generated the observation. First, a sequence of features Y is extracted from the speech signal using the *acoustic frontend*. The goal of the feature extraction is to provide a set of features that have a lower dimension than the original input samples, focusing

on the signal characteristics that allow for discriminating between spoken words. Then, for a given feature sequence Y, the maximization task can be formally defined as:

$$\hat{w} = argmax_w\{p(w|Y)\} \tag{3.1}$$

Rewriting the objective function in the right side of equation 3.1 we obtain

$$p(w|Y) = \frac{p(Y,w)}{p(Y)} \tag{3.2}$$

$$= \frac{p(w) \cdot p(Y|w)}{p(Y)} \tag{3.3}$$

We observe that the denominator of equation 3.3 does not depend on the maximizing argument w, hence we can omit the term $p(Y)$, resulting in:

$$\hat{w} = argmax_w\{p(w) \cdot p(Y|w)\} \tag{3.4}$$

Equation 3.4 allows us to split the probability for a hypothesized word sequence w into two individual parts: the *language model probability* $p(w)$ and the *acoustic model probability* $p(Y|w)$. The language model gives the prior probability for a certain word sequence, for example, *US president Barack Obama* should be much more likely than *US president Nathan Obama*. The acoustic model gives the probability that the feature sequence Y is observed if the word sequence w is spoken, e.g., it is a model for the acoustic realization of w. Most state-of-the-art systems for large vocabulary speech recognition use Hidden Markov Models for the acoustic modeling of words [32], which can be tuned towards a specific transmission channel or a certain speaker [119]. The pronunciation of a word is represented by a sequence of phonemes given by a pronunciation lexicon, which is typically generated automatically using grapheme-to-phoneme conversion [11].

In our baseline system, we use a typical holistic system setup which estimates $\hat{w}$ for a given observation Y using the aforementioned paradigm. The architecture is illustrated in figure 3.1. Assuming that a document of interest is already segmented into speech and non-speech parts, a single speech utterance is sent to the LVCSR system for transcription. The *decoder* than searches for the optimal word sequence $\hat{w}$ which maximizes equation 3.4, given the input feature vector sequence, a language model for $p(w)$ and an acoustic model for $p(Y|w)$. A detailed description of standard approaches to the individual parts illustrated in figure 3.1 can be found in [32].

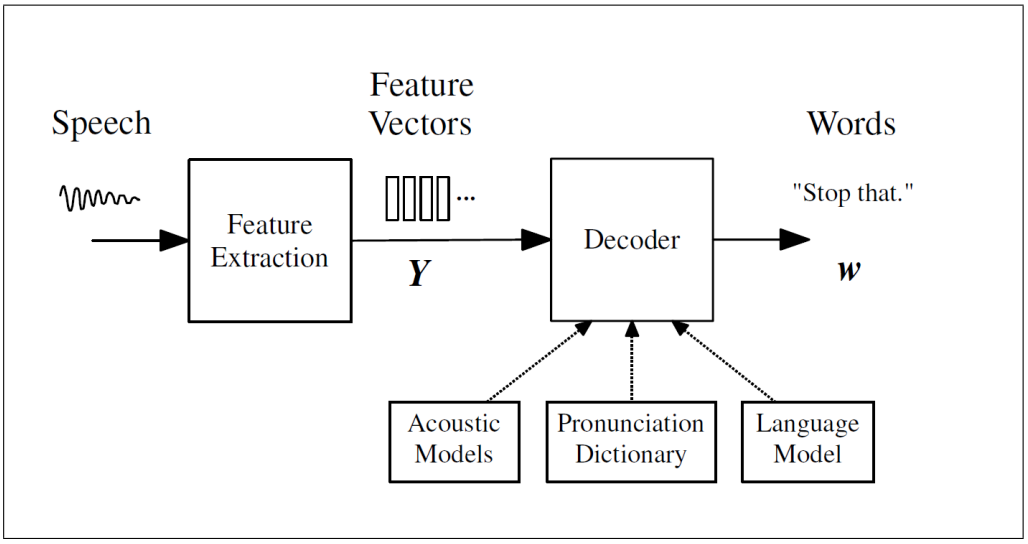

Figure 3.1.: Architecture of a HMM-based system for automatic speech recognition, taken from [32].

In this section, we describe the setup of our ASR system used for creating the 1-best word transcriptions. As illustrated above, the ASR system setup is determined by the actual decoding algorithm, the acoustic frontend, language and acoustic model, and the pronunciation lexicon. For the baseline setup, we use the following components.

Decoder. We use a state-of-the-art off-the-shelf component for speech decoding which is readily available: the *Julius Open-Source Large Vocabulary CSR Engine*[5], a flexible yet efficient decoding engine that has been developed over two decades at the Kawahara Lab at Kyoto University[6]. Julius has been successfully used for large vocabulary tasks in Japanese [66, 65]. It is known to be very efficient while showing similar recognition accuracies compared to other large vocabulary decoders [92].

Acoustic model. Large training sets are critical when it comes to building the acoustic model [31]. A baseline acoustic model which was available at Fraunhofer IAIS prior to starting the work on this thesis was trained on only 14 hours of transcribed broadcast speech. It was obvious that the set was too small to be representative for the heterogeneous characteristics of the targeted use cases. Hence, for the baseline of the thesis at hand, we decided to collect a new training corpus which is (a) substantially larger than the original set and (b) representative for many TV and radio formats. We col-

[5]http://julius.sourceforge.jp/en
[6]http://www.ar.media.kyoto-u.ac.jp/

lected a large set of data from both public German broadcasters ARD and ZDF, but also from German radio stations that have a lot of spoken content (Deutsche Welle, Deutschlandfunk, WDR). To a large extent, the data was taken from the professional podcast channels of the respective broadcaster, but we also recorded additional high quality material via DVB-S. All recordings were of a professional studio recording quality with no notable compression artifacts. After extracting the audio track from the media assets, each file was manually segmented into single speech utterances, which were then manually transcribed on the word level using Transcriber [5]. We were particularly interested in having a baseline training corpus which reflects *standard* German speech and is optimized for representing professional speakers in a controlled environment. Hence, we excluded utterances with strong dialects or heavy spontaneity from the manual transcription. For the same reason, only wideband speech was transcribed, i.e., all utterances transmitted via telephones were not included. Telephone speech is very rarely observed in TV data, hence we decided to focus on optimizing the wideband performance. Telephone speech is more prevalent in the radio domain, where interviewees are often not professionally recorded and transmitted, but rather call-in with their phone into a studio situation. For such scenarios, a dedicated telephone model should be used. The following summary recapitulates our rules for the manual segmentation and transcription process, which are inspired by existing broadcast corpora in other languages, such as the French ESTER [33] or the English HUB4 corpus [41]. The focus of training data transcription is rather on quantity than on annotation completeness. Typically, evaluation corpora are annotated with much more detail such as speaker names or speech and background types [6], whereas training transcriptions are often not even verified by a second annotator [35]. We specified a limited set of easy rules, similar but reduced compared to the DiSCo transcription rules described in section 2.4.2. Ideally, a segment resembles a single sentence with a duration between 5 and 30 seconds. A segment boundary should be inserted in the following cases:

- At a speaker change.
- Between two sentences.
- Between speech and non-speech segments.
- At the boundaries of long speech pauses ($>$ 1 second).

The annotator should skip utterances which cannot be exploited well in acoustic training. If in doubt, we asked the annotators to skip a segment.

- Segments without understandable speech.
- Foreign language (i.e., non-German).
- Strong dialect.
- Simultaneous speech from several speakers.
- Telephone speech.
- Very short segments ($<$ 1 second).
- Utterances containing stutter.
- Utterances containing slip of the tongue.

All utterances must be transcribed just as they are spoken, e.g., *fünf bis sieben Prozent* instead of *5-7%*.

Table 3.1 summarizes the composition of the final wideband acoustic training corpus. All in all, we collected and transcribed about 110 hours of German speech utterances, split into radio and TV sub corpora of roughly the same size. The transcription sums up to over one million running words, with a vocabulary of about 60000 unique word types.

Table 3.1.: Corpus for training the acoustic models.

Sub corpus	**Utterances**	**Hours of Speech**	**Transcribed Words**
Radio	50,057	51	465,750
TV	71,222	57	558,159
All	121,279	108	1,023,909

It is clear that further data collection will further increase the performance of the acoustic model. However, it has been observed in several contributions that the decrease in WER saturates, and the additional gain from adding more data becomes smaller. For example, in [31], the authors report a 0.6% absolute WER decrease while more than doubling the training data set from 144 hours to 375 hours. Note that the manual segmentation and transcription of a one hour file can take up to eight hours depending on the complexity of the audio track, hence a further increase of the training data set should be well-considered.

As in the case of decoding, we decided to select a well-established model structure and training approach for the acoustic model. The phoneme set is based on the SAMPA-D-Vmlex set[7], augmented with dedicated phonemes for frequent diphthongs[8], yielding a set of 49 phonemes. The system uses crossword triphone models to incorporate the left and right acoustic context of a phoneme. We use a phonetic decision tree to cluster the triphone models, such that models for similar triphones can share the same parameters and models which have not been observed in training can be synthesized [120]. After clustering all possible triphones, about 20000 physical model clusters remain. Each triphone is modeled by a three-state Hidden Markov Model with a 0-1-2 topology (i.e., only loop, forward and skip transitions are allowed). The emission probability of each HMM state is represented by a 16-component Gaussian Mixture Model.

Our speech recognition system uses *Mel-Frequency Cepstral Coefficients (MFCCs)* as features [22]. MFCCs are well-studied and have been successfully used in speech technology for many years. We use a standard configuration with 12 MFCCs and signal energy, augmented with both first and second order derivatives to capture temporal context. This yields a 39-dimensional feature vector per frame, which is calculated over a 25ms window at a rate of 100 vectors per second. All feature vectors are normalized using Cepstral Mean Normalization [32] in order to reduce the influence of different recording channels.

The parameters of the acoustic triphone models were estimated using the Hidden Markov Toolkit[9]. We use the Maximum-Likelihood paradigm to estimate the parameter set, which results in models that give the best explanation of the training utterances.

Pronunciation lexicon and language model. The vocabulary for the pronunciation lexicon was selected by taking the most frequent words from a large corpus of German newswire data obtained from DPA (Deutsche Presse Agentur), covering the years 2000-2006. The corpus contains about 10 million sentences, summing up to over 150 million running words and a vocabulary of 913041 unique word types. The data was assembled from DPA articles from the categories *politics* and *miscellaneous news*, yielding a good textual baseline for many recognition tasks in the broadcast domain. An important design decision for a word-based ASR system is the size of the decoding vocabulary. Many state-of-the-art broadcast news systems for English use a 60,000 word dictionary [31, 37], which yields a relatively low OOV rate on broadcast news data. For example, the authors in [37] report an OOV rate of 0.3% on the HUB4 evaluation set [35]. We decided

[7] http://coral.lili.uni-bielefeld.de/Documents/sampa-d-vmlex.html
[8] http://www.phon.ucl.ac.uk/home/sampa/german.htm
[9] http://htk.eng.cam.ac.uk/

to use a larger decoding lexicon for two different reasons:

- The number of valid German words is much higher than the number of valid English words. On the one hand, this is due to the rich morphology in German due to flexions. This includes dedicated endings for nominative, genitive, dative and accusative cases of nouns and adjectives, but also individual word forms for verb conjugation. Moreover, German makes heavy use of compounding, i.e., multiple words (typically nouns) can be combined into a new word. In [76], the authors report a OOV rate of 6.1% for a German ASR system with a 60,000 word dictionary. Similar OOV rates were reported in [68] and more recently in [83].

- The vocabulary for the decoding lexicon is typically built by selecting the most frequent words from a large corpus as the vocabulary for the decoding dictionary. By using a very large decoding lexicon, many words will only occur rather infrequently in the large corpus, and hence occur also infrequently in evaluation tasks. Therefore, the effect of increasing the vocabulary on the word error rate might be low. However, our focus is Spoken Term Detection rather than perfect transcription, and our primary target metric is not WER but ATWV (see section 2.4.1). It is obvious that word-based Spoken Term Detection benefits if more possible search terms are in the dictionary. This can either be achieved by specialized, task- and domain-dependent dictionaries, or by increasing the dictionary such that the decoding process fits just within the time and space constraints of the scenario. Hence, before deployment, the STD system should be adapted on in-domain data that is as close to the target data as possible. However, this is not always possible, either because in-domain training data is not available at the time of model training or if the target domain is just too diverse to be represented by a single model (e.g., a lexicon for all TV data from 1950 to 2010 will exceed the technical capabilities of a standard recognizer).

We can conclude that (i) German decoding lexica should be larger than English decoding lexica and that (ii) non-specialized STD lexica should be as large as possible. We have experimented with different corpus sizes (including a 65,000 word baseline [96]) and found that a vocabulary size of 200,000 words is a good compromise between vocabulary coverage and model complexity (this size was also reported to have a reasonable OOV rate in [76]). If the vocabulary is too small, many spoken words cannot be transcribed correctly as they are not in the vocabulary. On the other hand, due to the complexity of the decoding process, decoding time will increase substantially if the vocabulary exceeds a certain size. Note that there is no temporal overlap with the evaluation corpus, which

is from a later period (2008-2009). Hence, words that came up only after 2006 are most likely not found in the decoding dictionary.

The manual phonetization of words is time-consuming, and not at all error-free and consistent [21]. Only small pronunciation lexica are typically hand-crafted, and large lexica for LVCSR are generated by automatic grapheme-to-phoneme conversion tools. We used a data-driven approach based on the Bonn Open Synthesis System (BOSS) [13] for generating the phoneme transcriptions of the decoding lexicon. A trigram language model was trained, again using the same large newswire corpus. The resulting language model contains 200,000 unigrams, 7,149,558 bigrams and 17,962,254 trigrams. We applied Katz smoothing to allow for unseen bi- and trigrams [16].

Spoken Term Detection on 1-best word transcriptions boils down to a simple text search problem, hence we can use standard methods for efficiently storing and accessing the ASR output. We use an *inverted file*, i.e., for each word in the decoding dictionary, we store all document positions where the word was spoken. Hence only a lookup in the corresponding bin of the word is needed to decide which assets contain the requested spoken word, together with the exact location of the utterance. The number of bins is limited by the size of the decoding lexicon (200,000 in our word baseline). For a phrase query with n words, we collect all documents that contain one of the query words as above, and then intersect the result sets to decide which documents contain all query words. In the remaining set of documents, we verify that the sequence of the matched words is equal to the sequence of query words. We note that other retrieval approaches such as suffix arrays [73] are more efficient for searching phrase queries on large textual corpora. However, in the case of word-based Spoken Term Detection, the size of the transcribed corpora is relatively small compared to classic text search tasks, such as searching a large collection of books or web pages. We will cover the scalability aspect in more detail in chapter 6.

3.2. Subword-Based Spoken Term Detection

For many scenarios, word-based speech recognition transcripts are far from perfect, and word-based ASR systems show dramatically high error rates even for rather controlled scenarios such as meeting recognition [43]. Obviously, high word error rates will render an STD system based solely on word transcriptions unusable for many application scenarios. Large vocabulary continuous speech recognition is a complex process, which interprets an acoustic signal by interweaving prior knowledge from acoustic and language models and simultaneously applies a wide range of pruning techniques to keep the decoding

search space at a tractable size.

Errors can occur at many stages of the decoding process, however, it is obvious that spoken words which are missing in the decoding dictionary are a major source for ASR errors. Such out-of-vocabulary (OOV) words cannot be transcribed correctly by a word recognizer. It has been observed that on average, each OOV word leads to 1.6 errors in the transcription [39], hence the WER is greatly influenced by the OOV rate of the ASR system.

Several solutions have been investigated to approach the OOV problem. At first glance, adapting the decoding lexicon and the corresponding language model to the actual decoding task is the most promising and obvious solution [8]. However, complete prior knowledge about the required decoding vocabulary is rare. In many of the described scenarios the vocabulary changes drastically from document to document, and a complete coverage of the spoken word types is not possible. For this particular decoding situation, i.e., where prior knowledge about the decoding task is not available or not sufficient for adapting the lexicon, *subword-based* approaches to Spoken Term Detection have been investigated. The typical approach to subword Spoken Term Detection is as follows:

1. The spoken utterances are transcribed on a subword level (such as phonemes or syllables) instead of using a classic word-based LVCSR approach. For each utterance u, this yields a subword transcription $u = s_{u,1} \cdots s_{u,n}$.

2. At search time, the query q is broken down into a sequence of r subword units $s_{q,1} \cdots s_{q,r}$. The system searches for a match between the subword representation of the query and the subword transcript, and matches are returned as STD hits.

Approaches to generating a subword transcription typically follow the same holistic statistical paradigm as depicted for word-based speech recognition in section 3.1. We are particularly interested in evaluating different recognition units, and explicitly aim at reducing the overhead when exchanging the recognition unit. In our system setup, we treat subword units just as words in the case of the LVCSR system. We use a fixed decoding lexicon which contains all subwords that can be decoded. For each subword, we generate a corresponding phonetization for the pronunciation lexicon of the decoder. Then, a subword language model is trained using a subword training text and the aforementioned subword dictionary, using the same process that is used for training a word language model. Apart from different parametrizations of the training language model training toolkit, the main difference is the textual input for the training process: the large textual corpus used for training the word language model is broken down into

subwords. This yields a large amount of typical subword sequences that can be used for generating the subword language model.

We note that there is a clear tradeoff between overall ASR accuracy and ability to cope with OOVs when using a statistical subword language model. A strong language model will typically yield lower subword error rates, but will also prevent unseen or unlikely subword sequences from being correctly decoded [110]. Coping with infrequent query words is the main target when applying subword STD approaches, hence using a low language model scaling factor is often mandatory. All other parts of the system such as the acoustic frontend, the acoustic model or the decoding algorithm remain untouched. Figure 3.2 illustrates the main components of an exemplary system for word and subword decoding, which outputs word, syllable and phoneme transcripts. Only language model and pronunciation lexicon need to be defined if a new subword unit is introduced into the system.

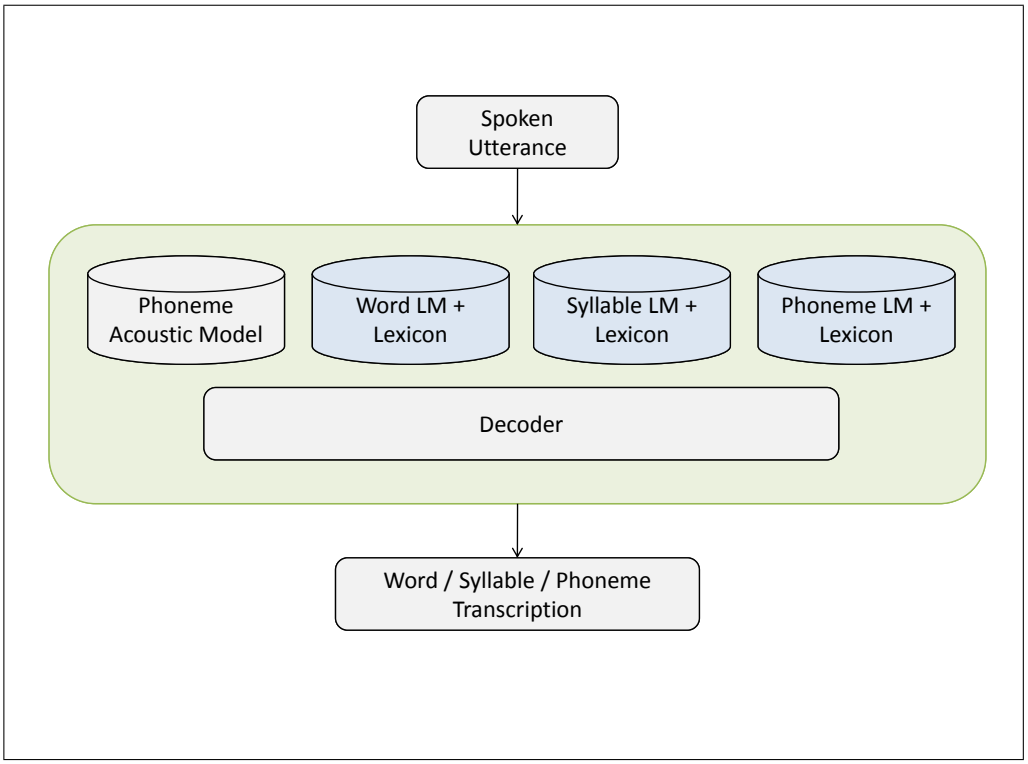

Figure 3.2.: Components required for word and subword ASR.

Several aspects of the subword STD process have an impact on the overall system performance. In particular, the following design decisions need to be investigated:

- **Choice of recognition unit:** The performance of the system will drastically depend on the choice of recognition unit. While longer units tend to produce more stable ASR results, smaller units might be more suitable for coping with the OOV challenge. In the following, we will introduce representatives for three popular units which are evaluated within the scope of this thesis.

- **Choice of retrieval unit:** It is not mandatory to use the same units during decoding and retrieval. At retrieval time, the search component can not only break down the query string, but also further break down the ASR transcript. Hence, it is possible to use larger units for decoding, and then retrieve from an automatically generated subword representation of the transcript.

Subword-based approaches to STD as described above are inherently language dependent: both subword lexicon and language model cannot be used across different languages. Even for very close languages (such as German and Austrian) the subword inventory and typical subword sequences will differ. However, for named entities with a low acoustic variance (such as *Putin*), retrieval from a subword transcript generated with the mismatched language model and a suboptimal subword inventory might still show reasonable performance. Within the scope of this thesis, we focus on optimizing the STD system components for the characteristics of the German language.

As already stated in section 2.3, many different units have been investigated for subword decoding. We can classify the different units into three broad categories, depending on the size of the unit: words, phonemes and intermediate units.

Words can be considered as an upper bound with respect to the unit length and are used in our LVCSR STD baseline. Using words as the building blocks results in powerful statistical language models, where short m-gram histories often already cover the most import word contexts. The main drawback is the OOV challenge, i.e., each word that shall be recognized must be part of the decoding dictionary.

Phonemes represent the smallest possible decoding unit for the aforementioned LVCSR system setup, as they are the core acoustic building blocks of the ASR system. Zerogram phoneme language models impose minimal linguistic constraints on the decoding process, and hence allow for maximum flexibility with respect to correctly decoding every phoneme sequence. On the other hand, such purely acoustic decoding configurations cannot benefit from statistical language modeling, and they are more sensitive, e.g., to

background noise or acoustic channel mismatch. This typically results in higher phoneme error rates on heterogeneous data sets.

Intermediate units range between words and phonemes. Using such intermediate units, one would expect a more stable decoding process under challenging conditions. Each subword unit provides additional structural information on the unit level by encompassing a fixed sequence of phonemes. This differs from purely statistical m-gram phoneme language modeling, as only phoneme sequences validated by prior constraints are taken into account during decoding. At the same time, the OOV challenge can still be approached: if the set of intermediate units is complete, every word can be built by concatenating the corresponding intermediate units. One can distinguish between different approaches for obtaining the subword units. Smaller units can be obtained from words by segmenting the orthographic transcription, or by segmenting the phoneme sequence corresponding to the word transcription. The latter approach has the advantage that the pronunciation of such a subword unit is immediately available. There is a range of alternatives with similar characteristics (e.g., [20, 60, 10]), from which we select *syllables* as a representative for the intermediate unit category. The syllabic representation is the natural phonologic segmentation of a word, and it is particularly suited for segmenting words from inflecting languages such as German, as the number of possible syllables is much higher than in agglutinating languages like Japanese or Turkish. Syllables have been successfully used for subword decoding in other inflecting languages such as Polish [71]. In [3], the authors note that syllable-based ASR cannot be used for generating word transcriptions as the system does not have knowledge about word boundaries. This disadvantage is less important in the case of STD. Still, the absence of word boundaries in the subword transcript or in the subword representation of a multi word query might yield additional false alarms during retrieval, especially in languages which make extensive use of compounding.

We also consider using different units during decoding and retrieval. By using larger units during decoding we can exploit the stability of decoding large units under more challenging acoustic conditions. Then, we break down the resulting large-unit transcript to smaller units, and search the small-unit representation of the query on the small-unit representation of the transcript. For example, when breaking down word transcripts to syllables or phonemes, this effectively enables us to find compound words if they are transcribed by their individual parts (such as *Gammelfleisch Skandal*) instead of *Gammelfleischskandal*).

3.3. Experiments

In the following, we will evaluate both the word-based baseline and the described vocabulary independent STD approaches on the complex DiSCo corpus introduced in section 2.4.2.

3.3.1. Word-Based Spoken Term Detection

First we study the lexical coverage of the evaluation data using our 200,000 word dictionary. Table 3.2 gives the corresponding OOV rates for the programs contained in the DiSCo corpus. The *OOV token rate* is the fraction of running words in the reference transcription that were not in the decoding vocabulary, while the *OOV type rate* is the fraction of unique word types that were out of vocabulary during decoding. Naturally, the OOV token rate is lower than the OOV type rate, as OOVs are typically low frequent (otherwise they would have been chosen for the decoding lexicon). Looking at the individual subsets divided by program, we observe that the OOV rate varies greatly between the data sources. The vocabulary of news and news magazines is covered quite well, however, non-political programs exhibit much higher OOV rates. For example, the OOV rate of the sports data is more than three times higher than the OOV rate of the political discussion show. This is not necessarily caused by a more complex vocabulary of a particular program, but rather by the fact that the corpus used for vocabulary selection matches better with the political evaluation data: while the OOV rate of the sports show is almost twice as high as the OOV rate of the foreign affairs magazine, each unique word in both shows is used about five times on average. We conclude that there is need for using a vocabulary baseline which is as close to the decoding scenario as possible. On the complete corpus, our system has an OOV rate on TV data which is comparable to other state-of-the-art systems for German LVCSR [76, 46]. The impact of the remaining out of vocabulary words on Spoken Term Detection performance is high. An OOV type rate of 11.5% on a representative set of TV and radio assets drastically limits the possible ATWV that can be obtained with any word-based Spoken Term Detection system.

We looked in detail at the OOV characteristics of the most frequent OOVs ordered by frequency rank which covered 25% of the whole OOV occurrences. We found that

- 97% of the most frequent OOVs were *nouns*,
- 60% of the most frequent OOVs were *proper names* and
- 32% of the most frequent OOVs were *compound nouns*.

Table 3.2.: OOV rates using the 200,000 word dictionary (by program).

Subset	**OOV token rate (%)**	**OOV type rate (%)**
News (public broadcaster)	1.3	5.3
News (private broadcaster)	2.1	5.7
Political discussion show	1.2	6.7
Foreign affairs	1.9	6.3
Regional magazine	2.2	7.6
Popular science	2.9	6.4
Sports show	3.7	12.0
All	1.9	11.5

Looking at some OOV examples, the impact of missing words on search performance becomes obvious. For example, in the sports domain, the most frequent OOV was *Hoffenheim*, the name of a soccer team that has not played in the first division of the German soccer league before 2008. Since 2008, the team is quite successful and popular, so a speech search system on sports shows would definitely need to support it. The same is true for the next most frequent OOVs, which stem from the surnames of three football players that were not active in the first league before 2007 (*Petric, Ibisevic, Ribery*). This *temporal* challenge in vocabulary design is accompanied by a time-independent *topical* aspect, i.e., words which are only used in the sports domain, but which are not only occurring in a certain period. Frequent examples from the sports evaluation data include *Torwartfehler - goalkeeping error, Dorfverein - small football club, KO - boxing knock out*. In other domains, the distinction between temporal and topical aspects is less obvious. For example, in the foreign affairs magazine, OOVs include numerous city and town names (e.g., *Diabakir* in Turkey) or other geographic landmarks such as rivers (e.g., *Volturno* in Italy), which can come up in the news and cease to be mentioned again at any point in time.

Hence we also give the OOV rates on the individual subsets introduced in 2.4.2 in order to interpret the word error rates of the LVCSR system on specific acoustic and linguistic challenges. From table 3.3 we observe the notable fact that spontaneous speech has a lower overall OOV rate than planned speech.

First we give the results of the ASR on the word level for each of these subsets in table 3.4. Our system produces similar results as other state-of-the-art research systems for German ASR on broadcast news data [76, 46]. The system produces an absolute

Table 3.3.: OOV rates using the 200,000 word dictionary (by acoustic and linguistic challenge).

Subset	**OOV token rate (%)**	**OOV type rate (%)**
Planned, clean	1.9	5.1
Spontaneous, clean	1.2	5.6
Planned, background speech	1.3	3.2
Planned, music	2.7	7.3
Planned, dialect	1.7	4.4
Planned, applause	3.1	5.2
Other (including mixes)	2.1	10.3
All	1.9	11.5

difference of 12.3% in WER between the simplest subset and the average on the complete evaluation corpus. The results are in line with existing evaluations of the different challenges on English data. In [38] the authors report an absolute difference of 14.8% in WER between planned clean data and an unconstrained evaluation set, while [109] reports a difference of 8.9% on a more recent system. The higher WER baseline is again due to the complex compounding and morphology of the German language [76].

Table 3.4.: Word error rates using the 200,000 word dictionary (by acoustic and linguistic challenge).

Subset	**Word error rate (%)**
Planned, clean	26.1
Spontaneous, clean	34.7
Planned, background speech	32.0
Planned, music	32.1
Planned, dialect	52.1
Planned, applause	64.0
Other (including mixes)	41.7
All	38.4

Looking at the differences between the recognition results for single challenges, we see that each individual challenge increases the WER compared to the planned speech baseline. Music and background speech increase the WER by only about 6% absolute.

In many cases, music serves as a filler in the background of the speech, and the dominant voice remains clearly understandable. The same is true for the data with background speech, which stems mostly from dubbed segments in a foreign language, where the original voice is still audible softly in the background. In contrast to this, the accuracy drops and the WER reaches over 63% when adding applause as the background noise, which typically has a high signal level and the dominant speaker becomes harder to understand.

Speech with a distinct dialect poses a major problem to the recognizer. As the acoustic training corpus is made up mostly of speech from High German speakers, it is likely that the observed dialects have not been observed during acoustic training. No acoustic adaptation is applied in the current baseline system configuration, hence the large mismatch between training and testing conditions leads to a poor performance.

For spontaneous speech, the WER is 8.6% absolute higher compared to planned speech under the same clean conditions. The language models are trained on a corpus consisting of written (and thus mostly planned) speech. This does not match the characteristics of the spontaneous test set, which contains numerous hesitations, repetitions or false starts. Moreover, spontaneous speech is typically uttered much faster than planned speech (180 vs. 167 words per minute in the two evaluation sets, estimated from the reference transcriptions). This characteristic is not well covered by the acoustic training set, which consists mostly of planned speech.

We retrieve the results for the 501 specified DiSCo queries from the inverted word index as described above. For this evaluation, we did not apply any further post-processing methods to the output of the word recognizer (such as stemming or compounding/decompounding of the ASR transcripts), and the query must be matched exactly with the ASR transcript in order to be found correctly. Multi-word queries are treated as phrase queries, i.e., the exact sequence of words needs to be matched. Figure 3.3 shows the results for all data subsets. Precision is always above 80% when retrieving from the word transcripts, even in the hard cases containing applause or dialect speech. Recall exceeds 70%, except for the two mentioned classes. In the case of applause, 62% of the existing references could not be found in the word transcript.

Next, we consider the influence of OOV queries on the evaluation results. Evaluating a word-based STD system on a fixed set of queries clearly depends on the number of queries that contain OOV words, i.e., queries that cannot be detected by the word system. This number depends both on the vocabulary design and the query composition, and can vary greatly between evaluation sets. First, we analyze the composition of the query set

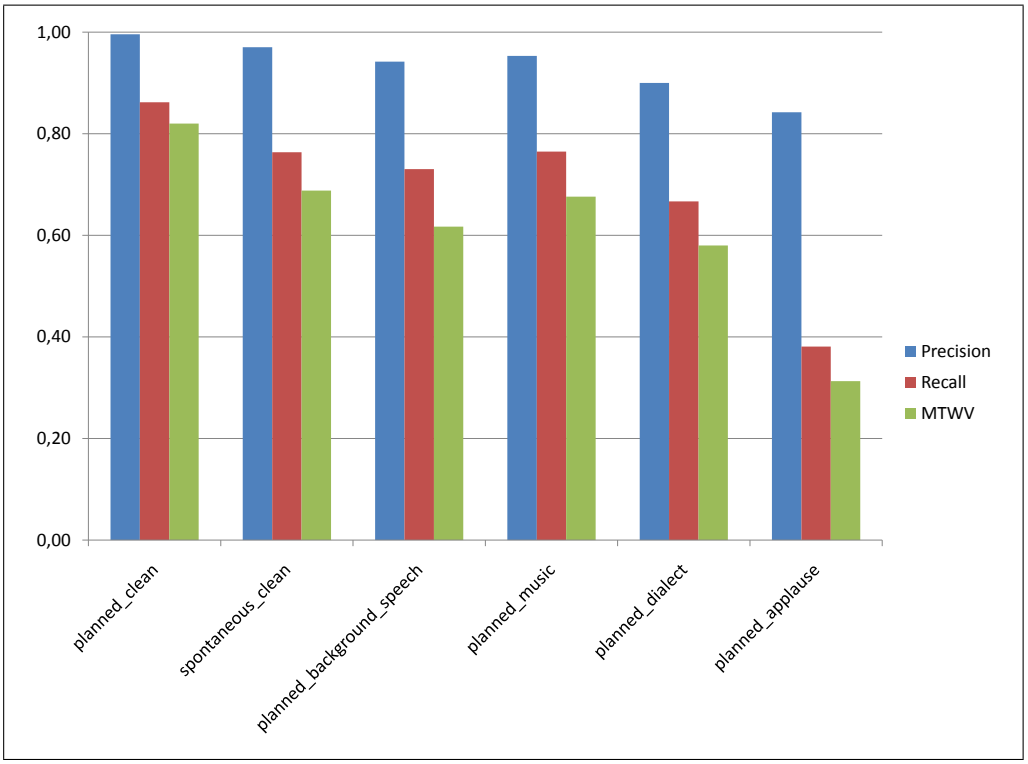

Figure 3.3.: STD performance on the individual subsets with exact retrieval from 1-best word transcriptions.

into in- and out-of-vocabulary queries with respect to the decoding lexicon of the word baseline. From table 3.5, we see that the number of OOV queries and corresponding OOV query occurrences on the complete corpus is relatively low, hence we expect that the STD performance on in-vocabulary terms is similar to the result on all queries. The large ASR lexicon used in the evaluation is optimized for the political domain (see below), which is dominating the corpus. Therefore the number of queries containing an OOV is rather small in this particular scenario. While this might not be true for new and unseen domains, it is a useful property in this evaluation: it allows us to compare the performance of word and subword retrieval approaches on the same STD task, without taking into account high OOV rates of the word recognizer.

In table 3.6, we compare the retrieval performance from the word 1-best baseline on the whole DiSCo query set to a restricted set, excluding all queries which contain an OOV term. By evaluating just on the in-vocabulary terms, recall is 4% absolute higher at equal precision compared to the complete query set, reflecting the impact of the OOV queries on the evaluation. The baseline ATWV of 0.67 is a suitable indicator for the

Table 3.5.: Composition of the query set with respect to the word decoding lexicon.

Query set	Unique queries	Query occurrences
All queries	501	2,748
IV queries	460	2,601
OOV queries	41	147

overall system performance of a word-based LVCSR system on the STD task, regardless of the relation between query set and decoding vocabulary composition.

Table 3.6.: STD performance using word 1-best retrieval.

Query set	Precision	Recall	MTWV
All queries	0.95	0.72	0.62
IV queries	0.95	0.76	0.67

STD error analysis. From the 2748 query occurrences, the word baseline misses 147 OOV query occurrences and 624 IV query occurrences. Looking at the IV misses, we observe that the corresponding query terms fall into three major categories:

- Proper names, which were infrequently mentioned in media at the time when the language model training data was collected, but gained popularity when the evaluation data was recorded. This includes the name of US president *Barack Obama*, who made it into the decoding dictionary as he was already a senator in 2004, but became widely popular in Germany not before 2007 when he started the campaign for the presidential election. The poor performance on such query terms (which also include *Andrea Ypsilanti* or *Lewis Hamilton*) indicates that augmenting the dictionary with new terms alone is not sufficient, and that either a continuous adaptation of the language model or a vocabulary independent approach is obligatory for such terms.

- Foreign words, such as *Champions League*, *New York*, or *Hypo Real Estate*. This can be caused by the larger pronunciation variability that can be observed in these words due to nativization [70], but also the fact that monolingual pronunciation inventories like the the SAMPA-D phoneme set are not designed to cope with the challenge of foreign words [19].

Moreover, the system produces 113 false alarms. We found that over 80% of these false positives were caused by queries with six phonemes or less, and only 2% were caused by queries longer than 10 phonemes. Examples for such short queries which cause frequent false alarms are *Wahlen - elections*, *Kinder - children* or *Markt - market*. The corresponding unigrams in the language model have a high probability, hence they are likely to be used as back-off in smoothing. These errors are occuring especially in more challenging acoustic situations. We conclude that word-based 1-best STD is highly precise on longer query terms, and that retrieval of shorter queries might benefit from additional result verification (see chapter 5).

In summary, the word-based 1-best STD system is especially well suited for search scenarios where the following requirements are met:

- The set of relevant search terms that can occur is limited and known prior to the indexing process.
- The acoustic and linguistic conditions are not too challenging, i.e., the word error rate is reasonably low.
- The focus of the search is on precision rather than on recall, and search flexibility with respect to true and false positives is not required.

3.3.2. Subword-Based Spoken Term Detection

In the following evaluation, we will look in more detail into the STD performance when using the different recognition and retrieval units. Before the actual STD evaluation, we compare the required vocabulary sizes for the different units, and describe appropriate language model configurations. Within the scope of this thesis, we limit ourselves to three different fixed configurations: a word-based system tuned for high stability, a phoneme-based system optimized towards lexical flexibility, and a syllable system in between. We measure the flexibility of a system configuration by the average number of phonemes covered by a single n-gram (e.g., a unigram phoneme-based system with maximum flexibility would cover one phoneme per unigram).

For the word STD experiments, we use the same setup as described in section 3.1, i.e., we applied a trigram language model and a pronunciation dictionary with 200,000 words. On average, a word trigram in our language model covers a sequence of 33 phonemes. As described above, we only exchanged the language model and the lexicon for building the phoneme and the syllable system.

The phoneme dictionary contains all 49 phonemes which are used as central monophones in the triphone acoustic model, and an additional entry for the silence model.

In order to limit the influence of the statistical language model and allow for increased flexibility during decoding, we trained a weak 4-gram phoneme language model on a phonetized version of the same training data that was used for training the word language models.

For the syllable language model, we first break down the language model training corpus into its syllable equivalent, and collect all occurring syllables. As words, syllable frequencies follow a Zipfian distribution, i.e., very few unique syllable types cause most of the absolute syllable occurrences (see section 6.1). But unlike in the word case, we can reach a large coverage of possible words with a relatively small amount of syllables. In our case, we collect only about 10,000 unique German syllables from the large training corpus, while over one million unique words can be observed in the data. We assume that the set of syllables which is needed to model all possible words is finite, and that it is covered by the set of syllables observed in the language model training corpus. In particular, this assumption holds for the query set which is used in the evaluation at hand: none of the IV or OOV queries from the set of 501 queries contains a syllable which is not part of the 10,000 syllable vocabulary. We were particularly interested in generating a system which is *in between* the word and phoneme systems with respect to decoding stability and flexibility towards transcribing infrequent words. Hence we decided to train a 4-gram language model on the syllabified version of the training data, in order to obtain a sequence of 15 phonemes per syllable-4-gram on average. Table 3.7 summarizes the key aspects of the three different language models.

Table 3.7.: Characteristics of different decoding units.

	Word	**Syllable**	**Phoneme**
Vocabulary size	200,000	10,816	49
Language model history	trigram	4-gram	4-gram
Avg. phonemes per dictionary entry	11.1	3.8	1
Avg. phonemes per n-gram	33.3	15.2	4
Training data (sentences)	9,802,550	9,802,550	9,802,550
Training data (running tokens)	158 million	328 million	904 million

The actual decoding speed at indexing time is becoming less important in many scenarios with the growing and transparent availability of ubiquitous computing power, e.g., through services such as the Amazon Elastic Compute Cloud[10]. Nevertheless, required

[10]http://aws.amazon.com/ec2/

CPU time for indexing is still a notable cost factor in designing speech search systems. In order to achieve comparability at reasonable cost, we selected the pruning parameters of the decoder such that the overall decoding duration is less than two times the duration of the decoded utterance.

If we look at the systems in isolation, we can measure the performance of the three individual ASR systems with word, syllable and phoneme error rate. However, results for word, syllable and phoneme error rate are hard to compare. For the word system, we can also give syllable error rate by breaking down both reference and ASR transcript into their syllabic equivalents. Using the same evaluation metric for the word and the syllable system enables a direct comparison of the system performance, regardless of the chosen unit. The same procedure can be used to compare word, syllable and phoneme system with respect to the phoneme error rate. Table 3.8 compares the quality of the syllable transcript obtained from word and syllable decoding. The quality of word decoding systems is higher across all challenges despite inevitable errors in the word transcript stemming from OOV terms. This is mainly due to the more powerful language model and the larger decoding unit, and the relatively low OOV rate with respect to the large 200,000 word lexicon.

Table 3.8.: Syllable error rates obtained from word and syllable ASR (by acoustic and linguistic challenge).

	Syllable error rate (%)	
Subset	**Word ASR**	**Syllable ASR**
Planned, clean	17.2	20.6
Spontaneous, clean	24.9	28.7
Planned, background speech	22.9	27.9
Planned, music	22.3	28.2
Planned, dialect	41.4	48.1
Planned, applause	54.7	56.0
Other (including mixes)	31.3	36.0
All	28.3	32.9

Table 3.9 compares the phoneme error rate across all three systems. First we observe that the relation between the phoneme error rate of the word and the syllable transcript is very similar to the case of the syllable error rate comparison above. The phoneme error rate of the word ASR output is lower across all challenges, but on average by only

1.9% absolute compared to the direct syllable transcription. Compared to this result, the performance of the unconstrained phoneme decoding is very low. On average, the phoneme error rate of the phoneme decoder is 41.2% absolute higher compared to the output of the word decoder. This poor performance is caused by several aspects. Due to the small language model context and the absence of structural constraints as in the case of syllables, in many cases the decoder hypothesizes phonemes even at very short speech pauses, especially in the presence of background noise and despite the existence of a dedicated short pause silence model. Looking at the size of the ASR output, we observe that all syllable transcriptions contain a total of 240927 syllables, which is close to the size of the syllabified version of the word transcription (232003 syllables). In contrast to this, the phoneme transcriptions contain 822995 phonemes, opposed to only 578483 phonemes in the phonetized word transcriptions. In addition, the low influence of the language model shifts more influence to the core acoustic models which are naturally unable to match all the hetrogeneous acoustic conditions in the complex DiSCo corpus. We conclude that from the three systems, the most constrained word system produces the lowest phoneme error rate. We expect that the least constrained phoneme decoder cannot yield competitive STD results due to the high ASR error rates.

Table 3.9.: Phoneme error rates obtained from word, syllable and phoneme ASR (by acoustic and linguistic challenge).

	Phoneme error rate (%)		
Subset	**Word ASR**	**Syllable ASR**	**Phoneme ASR**
Planned, clean	11.1	12.6	43.8
Spontaneous, clean	17.4	18.0	48.7
Planned, background speech	16.0	18.3	61.0
Planned, music	15.1	18.0	75.3
Planned, dialect	30.1	33.6	75.9
Planned, applause	45.5	44.0	91.8
Other (including mixes)	22.7	22.1	64.4
All	20.2	22.1	61.4

We start our analysis of Spoken Term Detection using the different approaches by looking at the performance on in-vocabulary queries on the complete DiSCo corpus. This enables us to focus on the performance of the individual units irrespective of the OOV rate. Table 3.10 contains the results for all relevant combinations of recognition

and retrieval units on the in-vocabulary queries.

First, we observe that all approaches have an STD precision which is tolerable in many applications. Unsurprisingly, the pure decoding to and retrieving from phonemes yields the lowest overall performance. However, we note that despite an average phoneme error rate of 61.4%, the approach detects more than a fifth of the query occurrences at reasonable precision.

Both precision and recall drastically increase when using syllables as the recognition unit. Precision reaches 94% if syllables are used in retrieval, and it is decreased only slightly if the syllables are broken down to phonemes at search time. This is due to false alarms that are caused by ommitting implicit information about word boundaries, which is not available anymore during retrieval at the phoneme level. For example, consider *Ban Ki Moon*, the name of the Secretary-General of the United Nations as of 2011. A correct syllable transcription of a spoken occurrence of the name would result in *b_a_n_ k_i:_ m_u:_n_*. If this result is further broken down to phonemes, we obtain the phoneme sequence *b_a_n_k_i:_m_u:_n_*. Here, searching for the query *Bank* and its phonetic representation *b_a_n_k_* would lead to a false alarm, that would not occur at the syllable level.

On the other hand, breaking down syllables to phonemes increases recall and MTWV by 2% absolute. Looking at the results, we observe ambisyllabic movement of consonant clusters between two consecutive syllables. Consider the example *Ratte - rat*. Automatic syllabification of the query *Ratte* will yield a unique syllabification, e.g., *r_a_ t_@_*. However, from an acoustic point of view, the central *t* could belong both to the first and the second syllable, so with a relatively weak syllable language model, it is not clear whether the decoder should transcribe either *r_a_ t_@_* or *r_a_t_ @_*. In this case, breaking down both transcription and query to the phoneme representation copes with the amisyllabic movement of the consonant *t*.

As we can expect from the different ASR results, in-vocabulary Spoken Term Detection performs best if performed on the basis of word transcripts. Compared to the case of using syllables for recognition, recall is more than 10% higher at similar precision. For the same reasons as described above, precision decreases slightly when breaking down words to syllables for retrieval, and further if broken down to the smallest possible retrieval unit. A small gain in recall can be observed by breaking down words to syllables during retrieval for queries containing relatively infrequent compound words. In this case, the decoder might rather hypothesize the more frequent individual word parts, which cannot be found by exact word search for the full compound. However on the syllable level, word boundaries both in the query and the transcript are ommitted, and the syllable

representation of the compound can be found in the subword transcript. As described above, additional gain is possible by further breaking the transcript down to phonemes, yielding the best overall system for in-vocabulary terms.

We note that there is a scalability issue when using smaller units such as phonemes for retrieval. This effectively means that the amount of units to be stored and indexed increases drastically, e.g., in the case of the DiSCo corpus from 158 million words to almost 1 billion phonemes. This scalability challenge will be investigated in more detail in section 6.

Table 3.10.: Exact STD performance on IV queries.

Recognition unit	**Retrieval unit**	**Precision**	**Recall**	**MTWV**
Phoneme	Phoneme	0.73	0.21	0.07
Syllable	Syllable	0.94	0.66	0.52
	Phoneme	0.94	0.68	0.54
Word	Word	0.95	0.76	0.67
	Syllable	0.94	0.77	0.69
	Phoneme	0.93	0.78	**0.70**

Looking at the remaining errors for the best performing word-phoneme system, we observe that 85% of the false alarms are caused by queries with only one or two syllables. This reflects the fact that the ASR correct rate of longer words is typically higher than for short words [122]. In addition, most of the false alarms are caused by queries with a relatively high frequency such as *Wahlen - elections* or *Bayern - bavaria*. Following the statistical language modelling paradigm, high frequency words are prefered by the ASR decoder. Regarding the missed query occurrences, we observe that many misses are caused by queries containing rather infrequent proper names such as *Arthur Abraham* or *Andreas Kappler*. All parts of these multiphrase queries are part of the decoding lexicon and can in principle be decoded, however, the corresponding bigrams have a low frequency in the language model training corpus. Despite the lower ASR accuracy, some of these misses were detected by the less constrained syllable system.

Next, we investigate the performance of the three systems on very infrequent terms, namely queries containing a word which is not among the 200,000 most frequent words in the large language model training corpus. Words cannot be used as a unit for retrieval on OOV queries, hence Table 3.11 compares only subword-based approaches on the OOV query set. All in all, searching highly infrequent terms results in lower accuracy than searching for IV terms with a high frequency. First, we observe that again the

performance of retrieval from the phoneme transcript shows a very poor performance. Only 3% of the query occurrences can be obtained with an exact search on the phoneme transcript, although at a reasonable precision of 80%. Comparing retrieval from word and syllable decoding output, we observe that in the case of OOV queries, the syllable-based approaches outperform the search on the syllabified or phonetized word transcript. This indicates that the retrieval benefits from the greater flexibility with respect to the language model. As with IV queries, recall can be slightly increased by further breaking down words to phonemes instead of syllables.

Table 3.11.: Exact STD performance on OOV queries.

Recognition unit	**Retrieval unit**	**Precision**	**Recall**	**MTWV**
Phoneme	Phoneme	0.80	0.03	0.02
Syllable	Syllable	1.00	0.20	**0.24**
	Phoneme	1.00	0.20	0.24
Word	Syllable	1.00	0.07	0.15
	Phoneme	1.00	0.08	0.15

Despite the large improvement compared to the word decoder baseline, syllable STD on infrequent terms still suffers from low recall. A major drawback of the intuitive exact subword matching is the requirement that *all* syllables of the query must be transcribed correctly in order to yield a match. Especially for longer words as well as for short syllables this requirement is too strong. We will incorporate approaches to overcome this challenge in the next section.

For reference, table 3.12 summarizes the results on the complete query set. We observe that compared to the in-vocabulary results, the difference in recall between the word and the syllable-based systems is smaller due to the inclusion of the OOV queries.

Table 3.12.: Exact STD performance on complete query set.

Recognition unit	**Retrieval unit**	**Precision**	**Recall**	**MTWV**
Phoneme	Phoneme	0.73	0.20	0.07
Syllable	Syllable	0.94	0.64	0.50
	Phoneme	0.94	0.65	0.51
Word	Word	0.95	0.72	0.62
	Syllable	0.94	0.73	0.64
	Phoneme	0.93	0.75	**0.65**

3.4. Summary

In this chapter, we have investigated different setups for German Spoken Term Detection using exact search on 1-best ASR transcripts. The results on the complete DiSCo corpus show that for in-vocabulary queries, word-based approaches perform best (MTWV 0.70), while for rare OOV queries, the syllable-based systems outperform the other approaches (MTWV 0.20). The high error rates of phoneme decoding render pure phoneme systems unusable for heterogeneous corpora.

For the evaluation, we have selected words, syllables and phonemes as representative STD units, and made an explicit distinction between recognition and retrieval unit. A 200,000 word LVCSR system was set up, yielding error rates comparable with other state-of-the-art systems for German ASR. Language models for word, syllable and phoneme decoding were built with a focus on covering different language model histories (with 33, 15 and 4 phonemes per unit for word, syllable and phoneme m-gram, respectively). Word ASR yielded the lowest phoneme error rate due to the large size of the decoding unit and the most constrained language model. While syllable ASR showed comparable performance, phoneme accuracy drops when using a rather unconstrained phoneme language model.

The STD evaluation revealed that even with our large 200,000 in-domain word dictionary, many interesting queries cannot be found by the word-based STD system because they are not part of the vocabulary. We observed that almost all OOVs are nouns, and about 60% are proper names, which represent an important class of STD queries. Hence, the word-based 1-best STD system is especially well suited if the set of relevant search terms is known prior to the ASR and the lexicon can be adapted accordingly. Word STD produces highly precise results, but search flexibility with respect to true and false positives is not given.

For OOV queries, syllable STD performed best, however, still only 20% of the OOV occurrences could be retrieved. On the IV query set, subword STD based on syllables achieves almost 10% less recall compared to word-based ASR, which is mainly caused by the higher syllable error rate and the harder constraint that *all* syllables of the query sequence must be matched exactly (instead of only *one* match in the word case). In the following chapter, we will investigate methods for error compensation which cope with this drawback of syllable retrieval by allowing for partial mismatch between query and ASR output.

When distinguishing between recognition and retrieval unit, we observed the following on our German evaluation task:

- On IV queries, breaking down decoded words to phonemes yields the best performance due to better handling of compounding and ambisyllabic movement of phonemes.
- On OOV queries, STD on syllable ASR output outperforms retrieval from word or phoneme output. Accuracy is even higher if the recognition output is broken down to phonemes, which again copes with ambisyllabic movement.
- Retrieval on the output from phoneme ASR turned out to produce very poor results on DiSCo due to the high phoneme error rate on the complex data set. This is also true for OOV queries, where both word and syllable-based phoneme retrieval performed much better.
- Some OOVs can be retrieved when breaking down words to subwords, but these are mostly compound words, where the query was a compound part.

4. Compensation of Spoken Term Detection Errors

The preceding chapter has introduced subword-based STD as a viable means for coping with the OOV problem in speech retrieval. The evaluation clearly indicates that searching for infrequent words on transcripts based on syllable decoding outperforms exact search on the ASR output of other units such as words or phonemes. Nevertheless, the performance when searching such rare words is still rather poor, as only 20% of the ocurring queries could be found when perfoming an exact search on the syllable transcript. In this section, we will investigate reasons for this poor performance, and study approaches to further increase recall, especially for infrequent words. In particular, our approaches adress the challenge of exact subword match on imperfect ASR transcripts and cope with pronunciation variation in subword transcripts.

In section 4.1, we motivate our approaches to error compensation by analyzing the sources for STD errors in the subword domain. From our analysis we conclude that subword ASR errors and pronunciation variation are the major - and potentially disjoint - sources for STD errors. In the following sections 4.2 and 4.3, we present a set of new approaches to cope with the different error types, building upon existing work in this area.

First, we investigate the applicability of state-of-the-art lattice retrieval techniques for German lattice STD, which we have first published in [77]. A confidence measure based on the work published in [65] is used for on- and offline pruning of word and subword lattices. Based on existing work from [60], we study approximate search on German subword transcripts in section 4.3. We propose a novel syllable distance metric based on position-specific phoneme clusters (PSCs), which is focused on explicitly modeling syllable pronunciation variation in Spoken Term Detection. Our results on approximate matching have been first published in [96], while the PSC-based approach was first described in our contribution in [78]. The sub-syllabic units have application beyond STD, and we have successfully used them for ideolectal speaker recognition [7] and subword vocabulary adaptation [79]. Following the taxonomy from chapter 3.2, where

we distinguished between recognition and retrieval unit, we are again interested in the effect of approximate subword match on word transcripts broken down to subwords: does it pay off to invest in a dedicated subword decoding run, or can we simply retrieve from the broken down word transcripts with the same accuracy?

Finally, section 4.4 describes a novel approach to hybrid approximate lattice retrieval, which effectively adresses the two STD error spaces described above. We achieve this by loosely coupling lattice retrieval and approximate lattice path alignment in a two-step compensation cascade, which we have first proposed in [78], and then successfully applied on DiSCo in [94].

4.1. Error Sources in Spoken Term Detection

The exact subword matching approach requires that the subword transcription of a query occurrence is equal to the subword subword representation of the query in order to produce a match. Any deviation on either side will prevent the exact search from detecting query occurrences. In [78], we have shown that low retrieval recall in subword Spoken Term Detection stems from two different sources: it can be caused by *ASR errors* or by *pronunciation variation*, which are described in the following.

ASR errors. In many - if not most - scenarios that require transcribing natural speech, systems are far from reaching the ultimate goal of generating perfect transcriptions. The inevitable recognition errors are often caused by a mismatch between the conditions during training the system and using it for recognition. This mismatch can be subdivided into two categories:

- Language model mismatch: there is a mismatch between the textual corpora used for training the language model and the characteristics of the actual speech that shall be decoded. This includes typical word sequences, which can differ greatly between written newswire texts and spoken utterances from a TV discussion show. For subword decoding, the language model mismatch is less important as the influence of the language model is systematically reduced in order to allow for decoding of highly infrequent words.

- Acoustic model mismatch: the acoustic conditions of the evaluation data are not well covered by the data used for training the acoustic models. Acoustic conditions include the particular characteristics of a speaker's voice, but also characteristics

of the recording channel. The acoustic model mismatch is of particular importance in subword decoding due to the reduced influence of the language model.

In addition, subword decoders tend to have a lower ASR performance compared to word decoders, and ASR errors inevitably lead to STD errors if exact matching is applied. In the evaluation of the subword STD baseline, we have observed that the subword decoders typically suffer from higher syllable and phoneme error rates. The increased flexibility of the subword decoders and their ability to decode the actual acoustic realization of a subword sequence comes at the cost of less decoding stability under more challenging conditions.

Pronunciation variation. Breaking down word queries to subword sequences yields a *canonical* subword representation of a query, i.e., the subword transcription has an ideal reference pronunciation. However, different speakers might use different pronunciations for queries, hence there is a certain variability in the query pronunciation.

From a subword point of view, the decoder has to balance its hypotheses between two different targets:

- The decoder can aim at producing a subword transcript close to the canonical subword representation. This can be achieved by increasing the influence of the language model, e.g., via language model scaling factor and longer language model history.

- In contrast to the more constrained decoding optimized for generating the canonical representation, the decoder can be tuned towards producing a subword transcription close to the actual acoustic realization by reducing the influence of the language model.

From the evaluation in section 3.3 we have seen that the word baseline typically outperforms subword STD on highly frequent queries, hence we rather focus on tuning the subword approaches towards detecting very infrequent words. In this case, a more flexible language model configuration is much more appropriate, as the infrequent words are unlikely to be represented well by the training corpora. This approach has been used in the evaluation in section 3.3, where on average, each syllable m-gram covers 50% less phonemes than an m-gram in the LVCSR baseline.

Figure 4.1 illustrates the effect pronunciation variation on retrieval from perfect subword transcripts, if the ASR is tuned towards capturing the actual acoustic realization of the utterance.

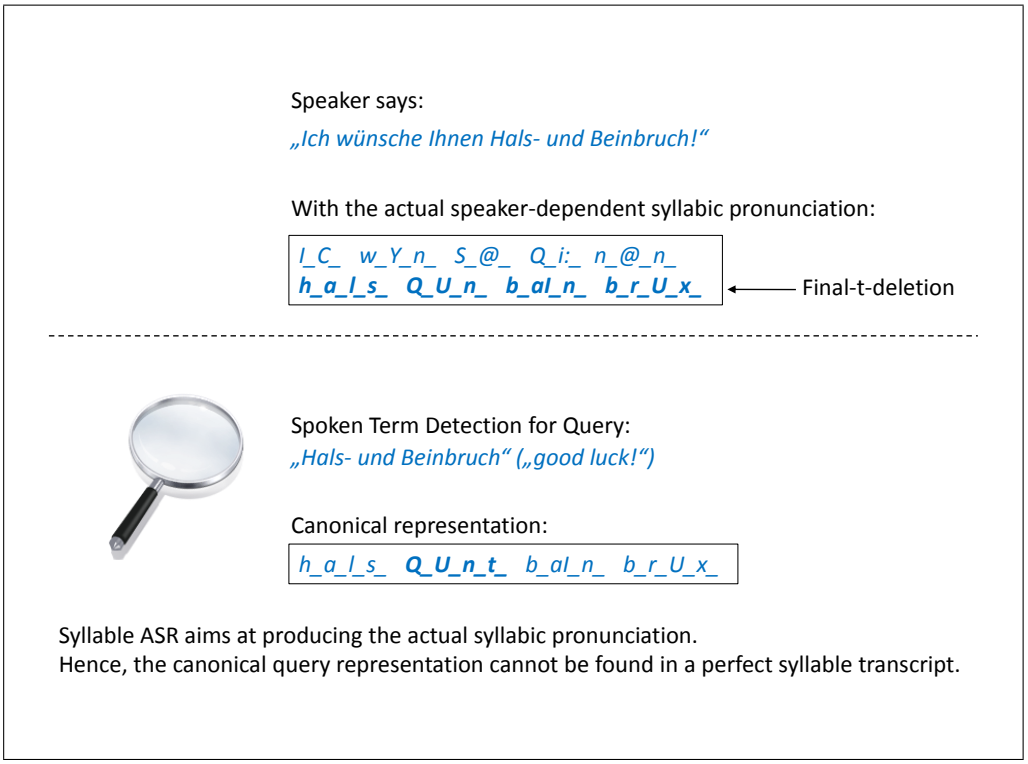

Figure 4.1.: Example for final-t-deletion, which causes STD misses on perfect syllable ASR transcripts.

In the following sections, we will investigate techniques to explicitly overcome the described challenges. First, we look into established methods for coping with ASR errors in word and subword STD by taking alternative recognition hypotheses into account. Then, we describe our approach to cope with deviations between a subword query and a subword transcript by means of approximate matching for subword sequences. Finally, we present a novel hybrid approach which applies approximate subword matching to competing ASR hypotheses. Here, we explicitly aim at overcoming ASR errors and pronunciation variation in an integrated approach.

4.2. Compensation by Alternative Recognition Hypotheses

Many systems for ASR follow the paradigm specified by equation 3.4. They aim at estimating the word sequence with the highest probability of having generated the observed feature sequence. This is also true for the decoder used within this evaluation [65]. However, ASR decoders can output more information about the decoding process than

just the 1-best transcription, for example they can provide competing recognition hypotheses. This additional output from the decoder can be exploited in retrieval. In the following, we describe lattices, which represent a standard approach for encoding alternative recognition hypotheses from the ASR decoder, and show how they can be used in Spoken Term Detection.

Instead of storing only the most probable *1-best* transcription of a single utterance, the speech recognizer can also produce a list of competing sentence hypotheses. This *N-best list* contains the N most probable sentence hypotheses, ordered by probability of the sentence hypothesis. This approach is particularly suited for simple and efficient retrieval on recognition alternatives. In principle, each sentence hypothesis can be indexed as a different transcription of the same utterance, and searching for the query boils down to simple text search. We note that the difference between two hypotheses is typically very small, and will often consist of only one or two words. Hence, in the worst case, large parts of the information encoded in the N-best-lists will consist of repetitions. This is a major disadvantage for longer utterances consisting of many words, where the number of sentences N that need to be stored to cover the most important alternatives is too large for practical applications. The amount of required sentence alternatives even increases if we consider using N-best lists for subword retrieval, as the number of tokens per utterance transcription is much higher than in the case of words (c.f. table 3.7).

More compact representations of the ASR hypothesis space can be used in order to overcome the storage drawbacks of the N-best list. In particular, *lattices* have been used extensively for this task, and they have been successfully applied across different retrieval units such as words [99], syllables [77] or phonemes [112].

Formally, a lattice is an acyclic directed graph

$$G = (V, E) \tag{4.1}$$

with a set of nodes V and a set of edges E. Each node $n \in V$ is a tuple

$$n = (i, l, t_s, t_e, c) \tag{4.2}$$

The tuple specifies a unique node id i, a node label l representing the transcribed token identity (i.e., the word, syllable or phoneme) and the start and end time t_s and t_e of the speech segment covered by this node. Moreover, for each node we store a confidence $c, 0 \leq c \leq 1$ which indicates the degree of uncertainty from the decoder. The node set contains two special nodes: an initial node n_{init} with no incoming and at least one outgoing edge, and a terminal node n_{end} with at least one incoming and no

outgoing edges. Figure 4.2 illustrates exemplary word and syllable lattices, generated on the same sample utterance from the DiSCo corpus with the reference transcription *über den ganz Deutschland staunt*. We used the ASR configuration described in section 3.3 to generate the lattices. Note that both ASR decoders use the same acoustic model, but differ in language model and lexicon. Hence, paths through the syllable lattice can be substantially different from syllabified paths through a word lattice. For example, alternatives for the correct word *staunt* includes the word *stammt*, but the corresponding syllable *S_t_a_m_t_* is not part of the syllable lattice.

The syllable lattice contains a good example for pronunciation variation and its impact on retrieval from subword ASR output. Here, the lattice contains the syllable sequence *d_OY_t_S_ l_a_n_* corresponding to the spoken word *Deutschland*, which is the correct syllabic representation if the speaker deleted the final *t* of the second syllable. In such cases, it is possible that the correct syllable sequence is not part of the lattice (because it has a relatively low acoustic likelihood), and retrieval would fail to recover from the pronunciation variation.

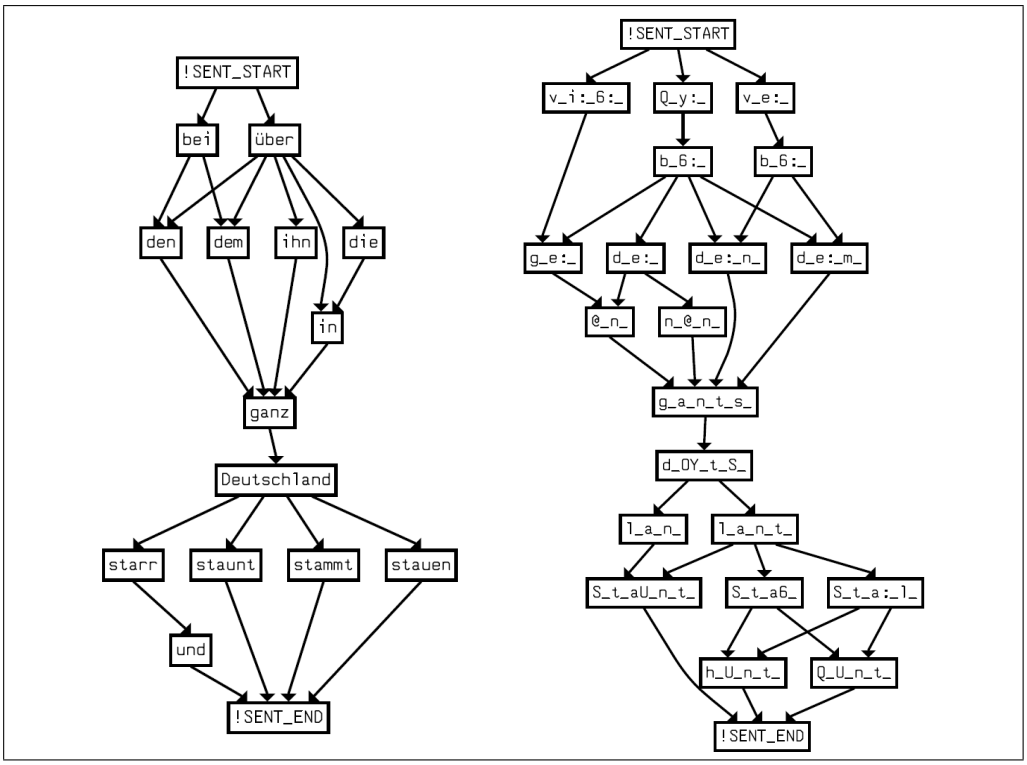

Figure 4.2.: Example for word and syllable lattices, generated on the same DiSCo sample utterance.

Lattice retrieval, i.e., detecting query occurrences in a lattice, can formulated as a search problem on the lattice graph. Consider a lattice G and a given word query $q = q_1 \cdots q_n$ with n words. Then, we search for all node paths $p = p_1 \cdots p_n$ with n nodes where the sequence of the corresponding node labels equals q. This description can be easily transformed for retrieval from subword graphs, such as syllable or phoneme lattices. In this case, the query is again broken down to subwords just as in the case of exact search on the 1-best transcript, and the retrieval is performed on the the subword graph just as in case of word lattices.

Several important design decisions have to be taken when using lattices for retrieval. In particular, one has to decide

1. on the size of the lattice graph, i.e., on the amount of competing hypotheses that shall be included,
2. how to control the number of nodes that will be included in the recognition lattice and
3. how to estimate the uncertainty of the decoder for a certain lattice node.

The optimal size of the graph that will be indexed depends on the requirements during retrieval. Basically, there are two alternatives. The graph can be pruned by removing unlikely nodes (i) until the graph contains only hypotheses that are assumed to yield correct STD results or (ii) by keeping as much hypotheses as possible within the given storage constraints. In the first case, only nodes with a high local recognition confidence are kept, and all others are removed during an offline process at indexing time. In the second case, we keep all hypotheses above a certain minimal confidence threshold and defer the actual STD decision to the retrieval process. The latter approach offers a greater flexibility for the user, who can adjust the balance between precision and recall of the search at query time. We refer to the first variant as *offline graph pruning* and to the second variant as *online graph pruning*.

Both approaches require a process for controlling the number of nodes in the lattice graph, which in turn requires a confidence measure for assessing the decoding quality at a certain node.

Our process for pruning nodes with low confidence is as follows. First, we obtain a large and unpruned lattice from the ASR decoder. Then, we estimate a confidence score for each lattice node based on the acoustic and language model likelihoods that were estimated by the ASR process. We follow the approach described in [14]. Given a node q, the lattice confidence C_q that the label of the node was spoken at the given time stamp is estimated by

$$C_q = \frac{L_\alpha(q) L_{AM}(q) L_{LM}(q) L_\beta(q)}{L_{max}} \tag{4.3}$$

Here, L_α and L_β are the forward and backward scores of the considered node q. The forward score of a node represents the likelihood that a lattice path leads to this particular node, while the backward score represents the likelihood for a path from this node to the end of the lattice. The acoustic likelihood of the node is denoted by $L_{AM}(q)$, and $L_{LM}(q)$ is the language model likelihood. The confidence score is normalized by the maximum likelihood L_{max} of the Viterbi path through the lattice, yielding a confidence score between 0 an 1. As usual in ASR decoders, we use log-likelihood scores instead of probabilities to cope with the problem of very small probability values. Hence, multiplication of probabilities in equation 4.3 becomes adding the corresponding scores.

In [14], the authors further distinguish between word and subword systems, as their subword phoneme system in not constrained by a language model, and is thus completely unconstrained from a linguistic point of view. In our case, this further distinction is not necessary and we can use the same confidence scoring approach for all considered units.

For the actual implementation, a standard forward-backward algorithm is used [90]. First, the list of nodes is sorted in decreasing order by the time stamps of the nodes. Starting with the rightmost node (which is now the first node in the list), we perform the following *forward procedure* on all nodes:

- If the node is the rightmost node, we initialize the forward score with a value of 0.0.
- The forward pass terminates if an initial node is encountered (the last node in the list).
- For all other nodes, we propagate the forward score of the current node to all its left neighbors by adding the acoustic model score of the current node and the language model score for the transition between neighbor and current node. For each left neighbor q' of the current node q, the forward score $L_\alpha(q')$ is given by

$$L_\alpha(q') = L_\alpha(q) L_{AM}(q) L_{LM}(q', q) \tag{4.4}$$

In a similar fashion, we carry out the backward procedure by sorting the nodes by time stamp in increasing order. Starting with the leftmost node, we estimate the backward scores from left to right. For the actual confidence score, we apply equation 4.3 for each

node q, using the estimated forward and backward scores. For the normalizing factor L_{max}, we take the maximal forward score (i.e., the forward score at the initial node).

For offline graph pruning, we directly use this confidence score for limiting the number of nodes per time frame by applying the following procedure which was used successfully to prune lattices in [65]:

- for each time frame t, we obtain a list of nodes that have a start time $t_s \leq t$ and an end time $t_e \geq t$
- we sort the list of obtained nodes in increasing order by their confidence value as estimated by the forward-backward procedure above
- for a *graph cut* of n, we remove all but the last n nodes from the list that represent the n nodes with the highest confidence at the current time frame

For online graph pruning, we have proposed a method in [77], which assesses the confidence of a path through the lattice at search time. The idea is to pre-calculate the confidence of each node at indexing time, and combine the confidences of a matching lattice path at runtime. We use the same approach as described above for pre-calculating the confidence for a single node at indexing time. Then, at runtime we can approximate the confidence score for the whole sequence in several ways. One approach presented in [77] is to calculate the product of the normalized node confidence scores as a lower bound for the query confidence score, i.e., for a query with n tokens we obtain:

$$C_q = \prod_{i=1}^{n} C_{q_i} \tag{4.5}$$

We note that this approach is particularly sensitive to outliers, i.e., the overall confidence score can become very small if only one of many query tokens has a low confidence. As an alternative, one can also consider using the average of the confidence scores:

$$C_q = \frac{1}{n} \sum_{i=1}^{n} C_{q_i} \tag{4.6}$$

For both approaches, we only need access to the node confidence scores at runtime and can ignore all other node-specific information (such as acoustic and language model likelihoods). This is especially useful in the case of subword decoding, where the graphs are typically substantially larger than word lattices. The size of the large subword ASR output graph is a major drawback when using subwords for lattice decoding and retrieval, and there is need for efficient approaches to scoring and accessing the information

contained in the lattice. In our experiments, the already pruned lattices from syllable decoding contained on average 13 times more nodes than the 1-best syllable transcription (see section 6). This requires attention in scenarios where users perform ad-hoc searches on large data sets, as exhaustive graph search for long query path matches (and optionally additional online pruning) can be computationally expensive. Moreover, storing such large lattices for thousands of hours of data requires compact representations for data persistence. Within the scope of this section, we focus on the baseline performance of lattice indexing and retrieval, i.e., we are interested in the performance without tight restrictions on the indexing and retrieval efficiency. Scalability aspects of the retrieval system - both in terms of retrieval efficiency and storage requirements - will be investigated in detail in section 6.

4.3. Compensation by Approximate Matching

The preceding section has introduced lattices as a means for coping with ASR errors in the context of Spoken Term Detection. Searching lattices instead of words allows us to increase the completeness of the search result, while we hope not to sacrifice too much STD precision by constraining the graph to the most promising hypotheses.

In this section, we introduce our approach for approximate phonetic matching between subword sequences, which can be used as a means for fuzzy STD on subword transcripts. As a baseline, our system builds upon the work proposed in [60]. We extend the approach with additional distance measures, investigate its applicability to other retrieval units and optimize its computational requirements. Finally, we validate our published results from [96] on the more challenging DiSCo corpus.

We define approximate phonetic STD on 1-best as follows:

Given a word query and a 1-best transcription, obtain all positions in the transcription where the local phonetic similarity between query and transcript is above a certain threshold.

Thus exact search on transcripts is a special case of approximate search, where the minimum required similarity is equal to 1. In the context of STD, phonetic approximation by minimum edit distance between the subword representations has been shown to be useful in several contributions [104, 91, 60]. We will use the idea as a baseline for our further investigations.

Figure 4.3 illustrates the generic workflow for approximate search on ASR output. First, we obtain the subword representation of the word query. As in the case of exact search on subword transcripts, we can apply grapheme-to-phoneme or grapheme-to-

syllable conversion, such that the unit type of the subword query matches with the unit type obtained from the speech recognizer. Then, we compare the subword sequence that represents the query with the subword transcripts which we obtain from the ASR. We estimate the local similarity between the subword query and the subword transcript at each position in the transcript. If the local similarity is above a decision threshold, we accept the transcript position as an STD hit. As in the case of lattices, low decision thresholds will increase recall at the inevitable cost of decreasing precision, while too high decision thresholds will not enable additional recall compared to exact search. In the following evaluation, we will particularly investigate the behavior of the different approximate search configurations while varying the decision threshold in order do find promising search configurations for the different search scenarios.

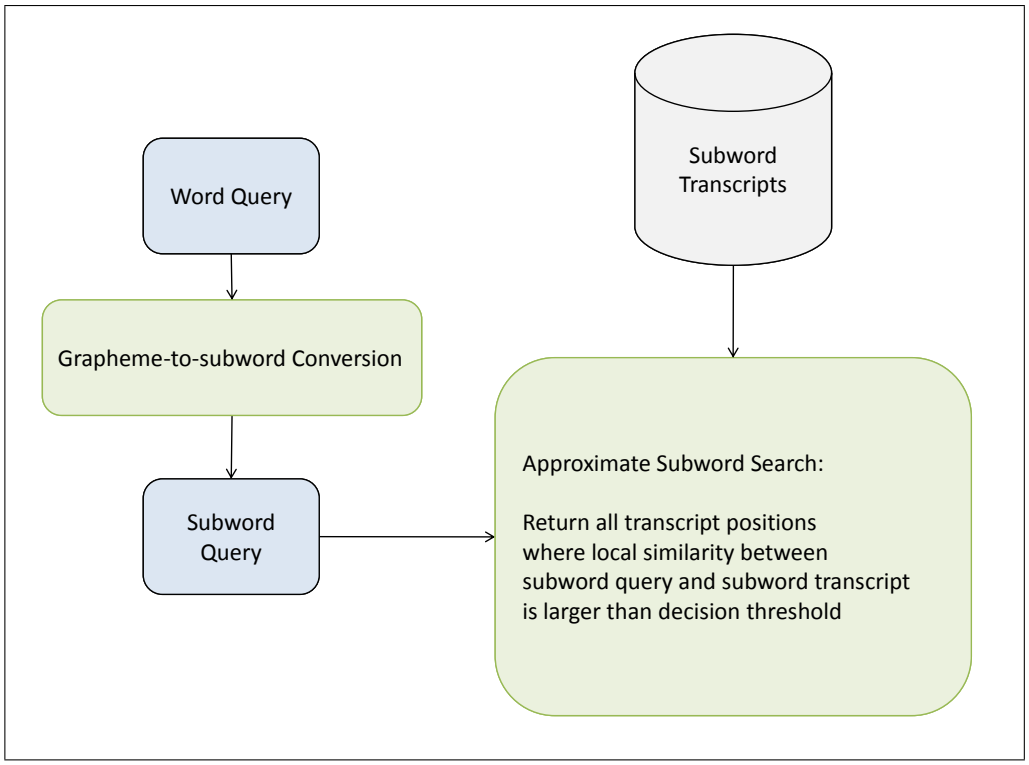

Figure 4.3.: Workflow for approximate subword search on ASR output.

4.3.1. Approximation using Minimum Edit Distance

In the following, we give a formal definition of the approximate subword search approach using minimum edit distance, regardless of the unit we use for subword transcription.

Recall the definition of a single occurrence hypotheses detected by the system for a particular query q as defined in section 2.1:

$$o(q) = \{s, t_s, t_e, c\} \tag{4.7}$$

where s is the document which contains the hypothesized hit at starting time t_s and end time t_e with a confidence of c. Then, an approximate subword match for the query q is defined as follows.

Assume that $t = s_1 \cdots s_n$ is the subword transcription of the document s, and $q = q_1 \cdots q_r$ is the subword transcription of the word query q_w. A sub sequence $m = s_v \cdots s_w$ of t is called an approximate match for q_w, if

$$d(q, m) <= \gamma \tag{4.8}$$

where $d(q, m)$ is the distance between the two subword sequences and γ is a threshold that indicates the degree of sequence variation that is tolerated by the system. The threshold γ is the decision boundary for the approximate matching and can be configured depending on the STD scenario: for example, recall-oriented applications such as media monitoring would rather require a low threshold, which enables additional true positive hits at the cost of lower precision.

The distance $d(q, m)$ can be estimated using a minimum edit distance, i.e., by calculating the minimal number of subword substitutions, deletions and insertions that are required to transform q into m. We estimate the minimum edit distance using Dynamic Programming recursion. and obtain the distance between q and m by

$$d(q, m) = D(r, (w - v)) \tag{4.9}$$

where D is the minimum accumulated distance at the last subword of the query and the last subword of the hypothesized matching subword sequence. Then, we can estimate D in a recursive manner. For a particular pair of subwords q_i, s_j, where q_i is the i-th subword in the subword query and s_j is the j-th subword in the hypothesized subword transcript match, the minimum accumulated distance at i, j is given by:

$$
\begin{aligned}
D(i,j) \;=\; & min\{ \\
& D(i-1,j) + TDP, \\
& D(i,j-1) + TDP, \\
& D(i-1,j-1) + sub(q_i, s_j) \\
& \}
\end{aligned}
$$

where TDP is the time distortion penalty for inserting or deleting a subword, and $sub(q_i, s_j)$ is the cost for substituting the subword q_i with the subword s_j. We will investigate possible definitions for the subword distance in the next sections, which are dedicated to the application of the minimum edit distance to phoneme and syllable sequences respectively.

Following the work presented in [60], we normalize the minimum edit distance by the maximum edit distance that can occur for the two given sequences and obtain a distance between 0 and 1.

In the following, we will estimate the confidence c for an approximate for a hit occurrence $o(q) = \{s, t_s, t_e, c\}$ by

$$c = 1 - d(q, m) \tag{4.10}$$

where $m = s_v \cdots s_w$ is a subword subsequence of the transcription of s, t_s is the start time of the subword s_v and t_e is the ending time of s_v. Focusing on *confidence* rather than *distance* allows us to use the same evaluation infrastructure as in the case of lattice matching, i.e., we can estimate the *actual* term-weighted value of a system given a confidence threshold $\delta = (1 - \gamma)$.

4.3.2. Approximation Scenarios for Selected Recognition and Retrieval Units

Next, we analyze the different approximation scenarios that are available for different word and subword transcription units.

- Approximation on phoneme transcripts: We could break down the word query to a phoneme sequence. Then, we would align the query phoneme sequence with the transcript phoneme sequence and search for positions with high similarity.
- Approximation on syllable transcripts: We could break down the word query to a syllable sequence and align it to the syllable transcript as described for phonemes

above. Moreover, we could further break down both query and transcript to phoneme sequences and perform a phoneme instead of a syllable alignment in order to overcome the ambisyllabic movement of consonant clusters between two consecutive syllables (see section 3.3.2). We note that breaking down from syllables to phonemes inevitably decreases the efficiency of the approximation, as the number of tokens that need to be aligned is increased by the average number of phonemes per syllable.

- Approximation on word transcripts: We could break down word queries and transcripts to both syllable or phoneme sequences and perform the alignment as described above.

Approximate search on phoneme transcripts Figure 4.4 illustrates the workflow for approximate search on phoneme transcripts. We break down the query into a phoneme sequence using grapheme-to-phoneme conversion, and find locations in the phoneme transcript that are phonetically similar.

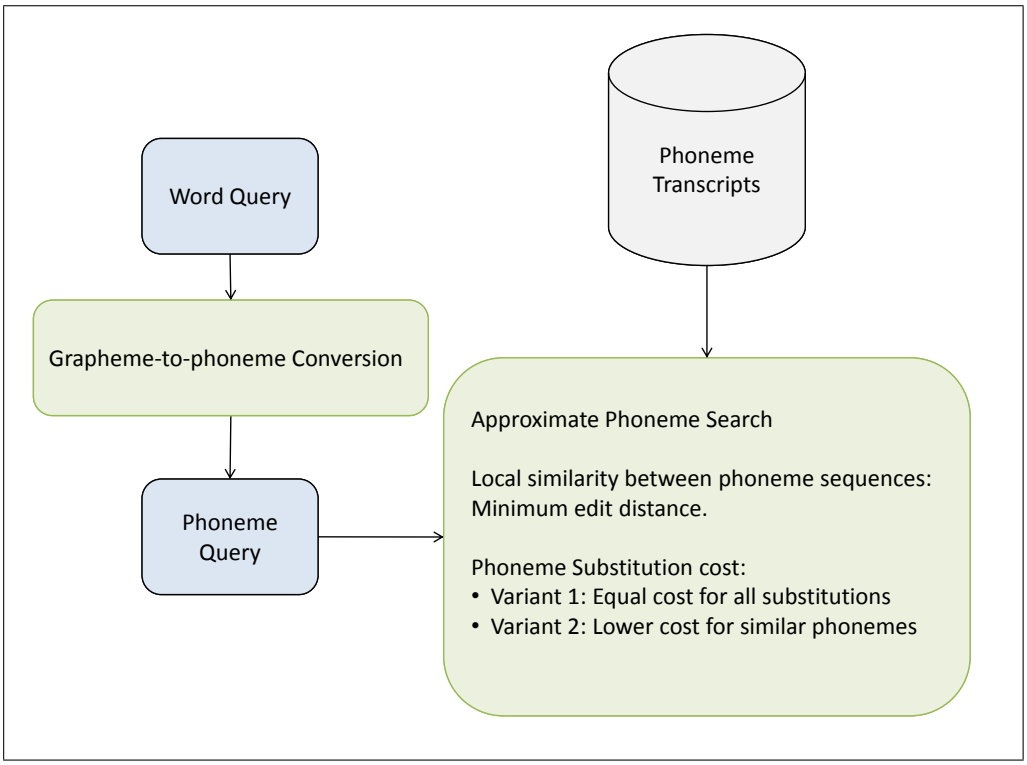

Figure 4.4.: Workflow for approximate phoneme search on phoneme ASR output.

The minimum edit distance according to equation 4.9 is used for the estimation of the similarity between a phoneme query and a subsequence of similar length from the phoneme transcript.

We expect that similar phonemes are falsely substituted more often by the ASR than dissimilar phonemes. Hence, it could be beneficial to decrease the substitution cost for pairs of similar phonemes when estimating the minimum edit distance. One can group the phonemes into different classes according to their similarity [104]. Typically, phonemes are either considered equal, completely dissimilar (such as a and S) or similar (such as b and p). Then, the subword distance $sub(p, q)$ between two phonemes p and q depends on the class they belong to:

$$sub(p,q) = \begin{cases} 0 & \text{if } p = q \\ c_{simil} & \text{if } simil(p,q) \\ 1 & \text{else} \end{cases} \tag{4.11}$$

where c_{simil} is a constant substitution cost for substituting similar phonemes and simil is a binary function which determines whether two phonemes are similar. As a baseline, we could disable the substitution cost model and assume equal substitution costs for all phonemes (i.e., $c_{simil} = 1$). The substitution cost for substituting similar phonemes should be lower than 1, and it should be lower than the time distortion penalty such that substituting similar phonemes is cheaper than deleting a dissimilar phoneme.

Similar phonemes could be selected by using prior linguistic knowledge about phoneme classes (e.g., a pair of vowels could be more similar than a fricative and a vowel) [104]. However, this approach is rather inflexible as it needs manual interaction when adapting the system, e.g., to a new dialect.

We apply a more flexible data-driven approach that can be easily adapted to a new decoding situation without deriving new rules. For each pair of phonemes (p, q) we estimate the confusion probability $p_c(p, q)$ that the ASR hypothesizes phoneme q while phoneme p is actually spoken. We obtain the phoneme confusion counts from the speech decoder as follows:

1. Perform phoneme decoding of development data set that has similar acoustic properties than the target set for STD (in our case: broadcast data).

2. Break down both automatic and reference transcription of the development data to the phoneme level using grapheme-to-phoneme conversion.

3. Perform alignment between reference and hypothesis phoneme sequences using

minimum edit distance. For each pair of phonemes (p, q), this yields $n(p, q)$ as the number of times where p was substituted with q in the minimum edit distance alignment.

We can assume that the frequency of substituting p with q is similar to substituting q with p. Hence we sum up the frequencies of both substitution directions and normalize the counts with the total number of occurrences of the two phonemes $n(p)$ and $n(q)$, respectively:

$$p_c(p, q) = \frac{n(p, q) + n(q, p)}{n(p) + n(q)} \tag{4.12}$$

For a list of k phonemes $q_1 \cdots q_k$, we obtain a *phoneme confusion matrix (PCM)* of size $k \times k$, where the element at position (i, j) is equal to $p_c(q_i, q_i)$.

Then, we use algorithm 1 to select a set of similar phoneme pairs that have a confusion probability above a predefined phoneme confusion threshold θ. By increasing the phoneme confusion threshold, more phoneme pairs will be labeled as similar. In section 4.5.2, we will investigate the effect when varying the threshold.

Algorithm 1 Generate set of similar phoneme pairs $S = \{(p, q)|simil(p, q)\}$ with phoneme confusion threshold θ.

```
S ← ∅
for all q is a phoneme from the finite set of phonemes do
  for all p is a phoneme from the finite set of phonemes do
    if (p_c(p, q) ≥ θ) then
      S = S ∪ (p, q)
    end if
  end for
end for
```

Approximate search on syllable transcripts The principle workflow for approximate search on syllable transcripts using syllables as the retrieval unit is illustrated in figure 4.5. We break down the query into a syllable sequence using grapheme-to-syllable conversion, and find locations in the syllable transcript that are phonetically similar.

As in the case of approximate phoneme search we estimate the minimum edit distance between two syllable sequences as a means for calculating the sequence similarity. In a similar fashion as above, we look at different possibilities for estimating syllable substitution costs by investigating different syllable similarity measures.

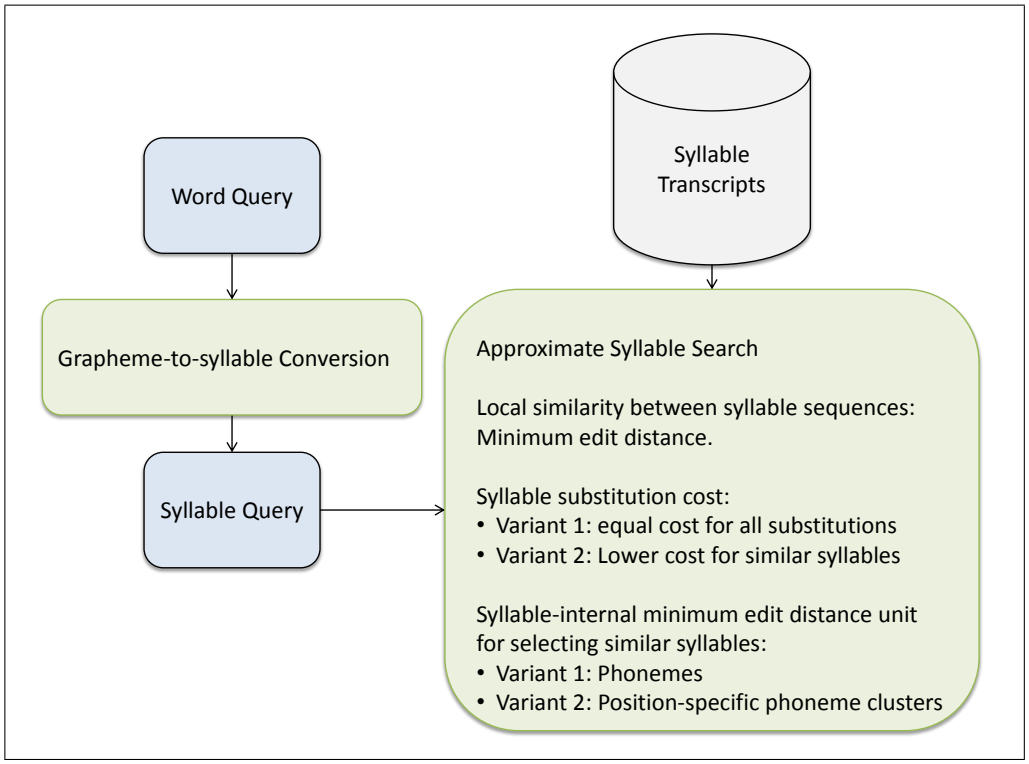

Figure 4.5.: Workflow for approximate syllable search on syllable ASR output.

The size of syllable inventory prohibits the direct estimation of the syllable similarity with the same procedure as in the case of phonemes (i.e., breaking reference down to syllables, aligning it to the syllable ASR output and counting the substitutions). Given a decoding lexicon of $10,000$ syllables, we can only observe a small fraction of the $10,000^2$ possible syllable pairs. Instead, we follow the idea presented in [60]. First, we break down both syllables that shall be compared into smaller subunits. Then, we use again the minimum edit-distance between the subunit sequences as an indicator for the similarity between the two syllables. In the following we will describe two different approaches for estimating the subword distance $sub(p, q)$ between two syllables p and q, namely by using phonemes and position-specific phoneme clusters as a subunit.

When using phonemes, the distance between two syllables is estimated as follows:

1. First, each syllable is broken down into the corresponding phoneme sequence.
2. Then, the distance between the two phoneme sequences is calculated according to equation 4.9.

Again, we can use different approaches for the phoneme substitution cost in this

syllable-internal minimum edit distance, i.e., equal cost or substitution cost derived directly from the phoneme confusion matrix as described in the preceding section.

We use the syllable distance to model the possible deviation between canonical reference transcription and ASR hypothesis. Phonemes are the smallest subunit that can be used to model the variation of a syllable. While this guarantees maximum flexibility, it also allows for almost arbitrary variations. Looking at the typical structure of a syllable, we realize that we can exploit additional structural characteristics when estimating the distance between two syllables and thereby constrain the set of possible variations.

A syllable is naturally structured into three adjacent phone clusters: a consonant cluster called *Onset*, a vowel cluster *(Nucleus)* and again a consonant cluster *(Coda)*. For example, the syllable *f_l_aI_S_* can be broken down into onset *fl*, nucleus *aI* and coda *S*. The clusters are realized differently depending on the phonotactics of the language. In the following, we are concentrating only on the structure of German syllables, where we observe the following properties of the three phone clusters that make up a syllable:

- The nucleus is a cluster of one or two vowels.
- Onset and coda are clusters consisting of zero to several consonants.
- The nucleus cluster is required, while Onset and Coda can be omitted (e.g., omitting the onset in the monosyllabic word *Eis - ice*, omitting the coda in the syllable *la*, and omitting both onset and coda in the syllable *a*).
- The variation of a phone cluster depends on the position. For example, a canonical *t* is very often dropped in the coda position as in the German conjunction *und - U_n_t*. However, dropping the phoneme *t* in the onset is very unlikely.

In the following, we aim at obtaining position-specific confusion evidence for phoneme clusters from held-out training data as in the case of phonemes, and then generalize pronunciation variation at the phone cluster level to the syllable level. We start with a parser given by algorithm 2 that breaks down syllables to position-specific clusters.

Again, we estimate the most probable PSC confusions in a data driven manner from some held-out data that has a similar characteristics as our evaluation data set. We estimate the PSC confusion probability for substituting the PSC p with the PSC q as follows.

1. Perform syllable decoding of development data set that has similar acoustic properties than the target set for STD (in our case: broadcast data).

Algorithm 2 Break down a phoneme string $s = p_1 \cdots p_k$ corresponding to syllable s to position-specific clusters (PSCs), namely onset (O), nucleus (N) and coda (C).

```
onset_done = false
for i = 1 → k do
  if p_k is a vowel then
    onset_done ← true
    add p_k to N
  else if onset_done then
    add p_k to C
  else
    add p_k to O
  end if
end for
```

2. Break down the transcription of the development data to the PSC level using algorithm 2.

3. Perform alignment between reference and hypothesis PSC sequences using minimum edit distance. For each pair of PSCs (p, q), this yields $n(p, q)$ as the number of times where p was substituted with q in the minimum edit distance alignment.

4. Again, we ignore the direction of the substitution by accumulating the counts for (p, q) and (q, p). We obtain the final PSC confusion probability by normalizing with the respective total counts of p and q as in the case of phonemes (see equation 4.12).

In [42], the authors note that variation within a syllable is often realized at the cluster level. Therefore, we exchange the subunits when estimating the distance between two single syllables, and use PSCs instead of phonemes. Then, the process for estimating the syllable distance is as follows:

1. First, each syllable is broken down into the corresponding PSC sequence using algorithm 2.

2. Then, the distance between the two PSC sequences is calculated according to equation 4.9, using the definition for PSC substitutions from above.

We have also investigated the usefulness of the PSC structure in other tasks that exploit pronunciation variation. In [7], we have successfully used PSCs to model inter-speaker pronunciation variability for large scale speaker recognition experiments. In [79], we predicted the most probable syllable variations using the PSC model and adapted the syllable decoding lexicon accordingly using multiple pronunciations.

In section 4.5.2, we will evaluate both described syllable distance subword units, namely phonemes and PSCs. We expect that PSCs will perform well if a system must be tuned for high precision, and that the phoneme-driven syllable distance metric will be more useful in recall-oriented scenarios, where flexible alignments are needed - even if they are not linguistically motivated.

In addition to searching the direct output from the syllable decoder, we can also break down the syllable transcripts into phoneme sequences and perform approximate phoneme search exactly as described in section 4.3.2. Our syllables consist of connected phoneme strings, hence generating a phoneme sequence from a syllable only requires removal of the syllable boundaries.

Approximate search on word transcripts For approximate search on word transcripts, we have two different options with individual advantages and drawbacks:

- We could break down both the query and the word transcripts to phoneme sequences and apply the workflow depicted in figure 4.4.
- We could generate syllable query and transcripts and use the workflow illustrated in figure 4.5.

We can expect higher STD performance when retrieving from the phoneme sequences, but at higher computational cost due to the larger amount of tokens that need to be aligned.

Applying subword approximation to word transcription search could result in higher STD performance compared to exact search for three reasons:

1. As German has a complex morphology, we obtain many transcription errors due to small letter variations. Approximation could serve as an implicit stemmer, e.g., by accepting deletions or insertions of word endings in different flexions.
2. Approximation can be used to cope with the decompounding challenge, as flexions at compound boundaries can be tolerated. Assume the query is the compound word *Wirtschaftskrise (economic crisis)* that consists of the two nouns *Wirtschaft (economy)* and *Krise (crisis)*, where the compound word has a variation at the end of the first compound part (*Wirtschafts*). The ASR decoder could prefer to transcribe the sequence *Wirtschafts Krise* instead of the full single word. This becomes more probable if the compound consists of many sub-nouns and becomes very long, which is typically penalized by the ASR. Moreover, the unigram *Wirtschaft* has a much higher language model probability than the unigram *Wirtschafts*, which is

usually not used in isolation, resulting in the transcription *Wirtschaft Krise*. When using approximate subword search, this transcription is broken down to subwords, where word boundaries do not exist, and the letter variation between compound query and subword sequence can be tolerated using approximate matching.

3. Consider the case where the word transcription is wrong because the spoken word was not in the dictionary, i.e., due to an OOV occurrence. In this case, the ASR decoder might hypothesize a phonetically similar word or word sequence from the decoding lexicon. For example, in the experiments in section 3.3.2 the name of the football club *Hoffenheim* is not in the dictionary, and the decoder sometimes output *hoffen heim*, which is a sequence with the same phonetic transcription but completely different meaning. However, a phonetically equal sequence for OOVs is only rarely available. Even if an OOV word can be re-written as a sequence of words with the same pronunciation, it is often unlikely that the ASR decoder hypothesizes this word sequence due to its low language model probability. Assume that the word *iPod* is not part of our word decoding dictionary. We might construct a valid word sequence *Ei Pott (egg pot)*, but it is unlikely that this word sequence will be transcribed. The ASR will rather select more probable competitor sequences such as *ein Pott (a pot)*. Here, approximate search on the subword level will help to overcome some of these errors, and we expect that we can find some OOV words using approximate search on the subword version of the word transcript.

Within the scope of this thesis, we focus on compensating errors that stem from the ASR and reduce the mismatch between query and ASR output. Additional techniques known from text information retrieval could be applied to compensate orthographic deviations caused for example by misspelled queries [58].

The approximate search acts as a general means for compensating deviations between subword query and subword transcript. Our expectations in terms of STD accuracy are twofold. In principle, the approximate search should be able to cope with both ASR errors and pronunciation variation, as both can be seen as deviations between query and transcript. Moreover, we expect that phonetizations and syllabifications from the grapheme-to-phoneme conversion can be compensated using the approximate search.

The approximation does not exploit knowledge about the actual decoding situation, it is *uninformed* with respect to the actual acoustic observation. Free parameters of the matching approach - such as the phoneme confusion matrix used for building the syllable distance matrix - are typically estimated based on statistics obtained from parallel development corpora. Hence it is important that the characteristics of the development

data are similar to the actual retrieval situation.

4.4. Hybrid Compensation

In this section, we propose a new integrated approach to STD error compensation, which targets both ASR errors and pronunciation variation explicitly. We motivate how the methods presented in section 4.2 and 4.3 can be merged for increasing the STD accuracy and present our hybrid retrieval approach.

4.4.1. Motivation for Hybrid Approach

First, we look into the question why merging the two already presented methods for error compensation might increase the STD performance compared to applying the individual methods on their own.

The approximate search presented in section 4.3 is *uninformed* with respect to the actual observation that is decoded by the ASR. Hence, adding knowledge about promising competing hypotheses from the ASR output graph can guide the approximation, and we expect that we can use higher approximation thresholds for approximate STD on the lattice.

On the other hand, if we consider exact subword lattice search as the baseline retrieval method, it is clear that additional approximation during search can find additional true positive hits. From the analysis of the error spaces in section 4.1, we know that compensation of pronunciation variation is not covered by subword lattice retrieval. However, it might be beneficial for some scenarios if the approximate search would be tuned towards compensating only pronunciation variations, as ASR errors are already covered by the lattice search.

4.4.2. Error Compensation Cascade

Intuitively, our hybrid approach assumes that if a query occurs in an utterance, then it can be found with approximate search on one of the paths through the lattice. In the following, we will focus on compensation for syllable STD, i.e., we perform approximate syllable search on a syllable lattice in order to find STD occurrences.

Combining the confidence scores from the two approaches at query time is not straightforward, as the scores stem from different sources (ASR node confidence score and minimum edit distance). Instead, we split the scoring into two stages: we exploit the lattice scores at runtime by applying offline pruning as described in section 4.2. This yields

lattices that contain only *valid* paths, i.e., paths which we would accept as hit during exact lattice search. Then, we use only the syllable distance as the primary metric to asses the quality of a hit on the already pruned lattice. This keeps knowledge from the ASR through offline pruning and preserves the flexibility from approximate search at query time.

As a baseline, we first consider an exhaustive approach to the problem using the following algorithm:

1. Break down query to syllable sequence $s_1 \cdots s_n$.
2. Obtain all paths of length n through the lattice.
3. For each path, we obtain the distance between query and path as described in section 4.3.1. Here, we are particularly interested in the effect of the different distance metrics described above, namely phoneme minimum edit distance and position-specific cluster minimum edit distance.
4. Accept path position as an STD hit if the syllable distance is below a predefined threshold γ.

Subword lattices can become quite large, even for relatively short utterances. This is especially true for spontaneous utterances or recordings with a complex acoustic background, where the ASR gets easily confused and many competing recognition hypotheses come up. For long queries (especially in the subword case), the number of possible subpaths with the length of the subword query can become intractable. Therefore, we add a simple yet effective restriction by requiring that at least one syllable of the query must occur correctly in the lattice. With this assumption, the amount of paths that need to be matched against the query is drastically reduced. Algorithm 3 specifies our approach for extracting paths that contain one of the query syllables, and figure 4.6 illustrates its application on a simple example. In section 6, we will investigate opportunities for further reducing the amount of paths that need to be aligned.

In the experiments below, we increase the length of extracted paths for a query of length n to $n + k$, where k is the maximum tolerated number of syllable insertions in the lattice (2 in the experiments below).

Within the scope of this thesis, we limit ourselves to hybrid compensation of STD errors obtained on *syllable ASR output*, and compare it to the respective individual baselines in the evaluation section below. We note that the same idea can be easily transferred to approximate search on phoneme lattices using exactly the same approach

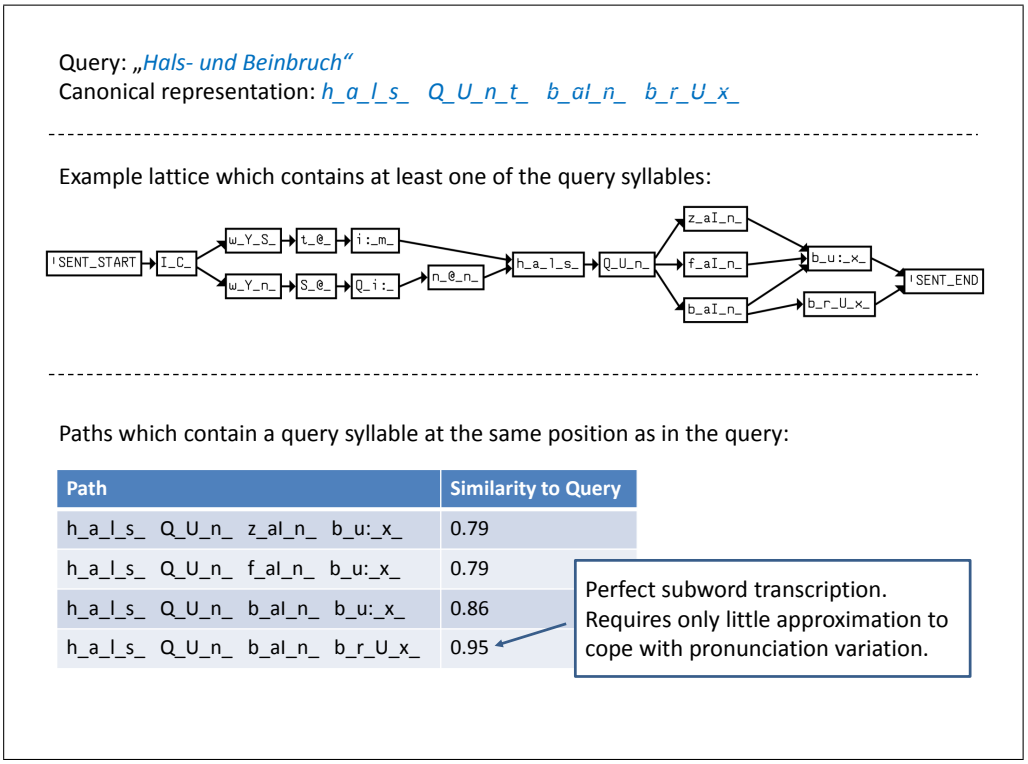

Figure 4.6.: Example for error compensation cascade, including path extraction and approximate path matching.

as in the case of syllables. We do not expect additional gain by applying approximate search in the same manner to word lattices (which would be possible by breaking down the word lattice paths to subwords followed by subsequent approximate matching on the subword level): STD errors due to pronunciation variation are less frequent in the case of word decoding due to the stronger language model and typically longer decoding units.

4.5. Experiments

In this section, we present an evaluation of the proposed set of methods for error compensation in our open vocabulary STD framework according to the following evaluation strategy.

1. We are particularly interested in the STD performance of the various approaches on the complete DiSCo query and data set. Where applicable, we will also investigate the performance of an approach on a set of rare queries, namely those which contain

Algorithm 3 Hybrid approximate lattice search for syllable sequence $q = s_1 \cdots s_n$ with at least one exact syllable match.

```
for i = 1 ··· n do
  for all Lattice l do
    if l contains a node t with label s_i then
      for all syllable path p of length n through t, where the ith syllable of p is equal
      to s_i do
        if d(p, q) > δ then
          Accept path position as STD hit.
        end if
      end for
    end if
  end for
end for
```

an OOV word with respect to our large 200,000 word decoding lexicon. This is possible for all approaches that use subword units for retrieval.

2. In many cases, the evaluated approaches can be optimized for a specific scenario, e.g., by specifying a minimum confidence for putative search results. In this case, we will consider MTWV as the main single-point metric, as it is the standard metric proposed by NIST. In some cases, we will also investigate the behavior of an approach while varying this decision boundary by looking at the corresponding Receiver-Operating-Characteristic curves.

We note that the direct output from the phoneme decoder will not be taken into account further. We cannot expect that the high phoneme error rate and the very poor STD performance both on frequent and rare queries can be overcome by the gain expected from lattices. An experimental comparison between phoneme and syllable lattice STD can be found in [77], where we have shown that for German data, using syllables for decoding and retrieval outperforms the phoneme approach both in terms of accuracy and efficiency.

4.5.1. Compensation by Alternative Recognition Hypotheses

We start our quantitative analysis by looking at lattice indexing as a means for coping with ASR errors. The lattices used in this evaluation have been generated with exactly the same ASR setup that was used for generating the 1-best transcripts in sections 3.3, i.e., using the same acoustic and language resources and the same pruning parameters during decoding.

In order to limit the amount of required storage, we prune all lattices with a graph cut of 20, which basically limits the amount of competing hypotheses per time frame to 20. In the following experiments, this set of lattices will be denoted as *unconstrained lattices*, as we do not expect further increase in STD recall by allowing for more competing nodes.

As a baseline for evaluating the lattice approach, we retrieve results for all 501 DiSCo queries from all 17152 unconstrained lattices without any further graph pruning. This means that if a query is found on one of the paths through the unconstrained lattice, it is accepted as hit, irrespective of the node confidences along the matching path. Table 4.1 compares the results of word and syllable lattice retrieval directly to the corresponding 1-best results. Neither 1-best STD nor unconstrained lattice search have a flexible decision boundary, hence we measure the performance by precision, recall and MTWV.

Table 4.1.: STD performance using unconstrained lattices.

System	**Precision**	**Recall**	**MTWV**
Word 1-best	0.95	0.72	0.62
Word lattice	0.61	0.76	0.61
Syllable 1-best	0.94	0.64	0.50
Syllable lattice	0.59	0.70	0.52

Comparing systems by recall and precision is not intuitive if we cannot assume the same value for one of the two variables due to lack of system configuration flexibility. Often, increase in recall comes at loss in precision and vice versa. Nevertheless, from table 4.1 we see that using lattices instead of 1-best enables a substantial increase in STD recall, both for words and syllables. We observe an increase in recall by 4% and 6% for words and subwords, respectively. However, the precision loss is dramatic: from almost perfectly precise word and syllable 1-best systems, STD precision drops by over 30% absolute when using lattices. Hence, even though MTWV is more tolerant to precision loss using the NIST evaluation defaults, the overall system performance of the unconstrained word lattice approach is even worse than the 1-best baseline, and the syllable MTWV is only increased by 1% absolute.

In the next experiment, we consider removing recognition hypotheses with a low confidence in order to overcome the drastic loss in precision when using lattices. As in the experiment on unconstrained lattices, we retrieve the STD results from the pre-pruned graphs with a graph cut of 20. However, in spite of accepting all matches between query sequence and lattice path as a hit, we accept only those matches that have a hit confidence above a certain threshold.

We compare two different methods for efficient confidence scoring of lattice path matches. First, we evaluate a method which we proposed in [77], where the confidence for a path is estimated by multiplying the confidences of the corresponding nodes along the path. As described in section 4.2, we expect lower recall for this method on longer queries with many query tokens. As an alternative, we evaluate the average node confidence as a metric for the confidence of the whole match. Figure 4.7 illustrates the results for the two methods while varying the decision boundary, both for word and syllable retrieval.

First, we observe that as in the case of 1-best retrieval, word lattice STD outperforms syllable lattice STD on the complete set in terms of recall when compared at different levels of precision. For both units, we can obtain substantial improvements in recall at still reasonable precision which is sufficient for many scenarios. With online graph pruning, the system becomes much more flexible, and its search behavior can be adapted to a particular usage scenario. Looking at the two methods for confidence scoring, we observe that the method based on average confidence yields higher STD performance at all levels of precision. It outperforms the baseline method both in the word and the syllable lattice system. The difference in STD performance between the two confidence scoring methods is larger for syllables than for words (3.2% absolute MTWV gain for syllables compared to 1.1% absolute gain for words), as syllable queries contain three times more query tokens than word queries. Confidence scoring based on average node confidence will be used as the default confidence scoring technique throughout the remainder of this thesis.

With the additional flexibility when using online graph pruning we are able to compare systems at certain interesting configurations. As motivated in section 2.4.1, we are particularly interested in the maximum term-weighted value using the NIST definition. Table 4.2 compares the static output of the 1-best systems to the respective lattice systems using online graph pruning. Unlike above, we can directly compare the systems as we are able to constrain one of the evaluation axes.

We carried out the following series of experiments to obtain the MTWV for online graph pruning:

- We obtain unconstrained word and syllable lattices from the ASR decoder.
- We perform a complete STD evaluation with different online pruning thresholds δ.
- The reported MTWV is the highest ATWV that could be obtained by varying δ between 0.0 and 1.0.

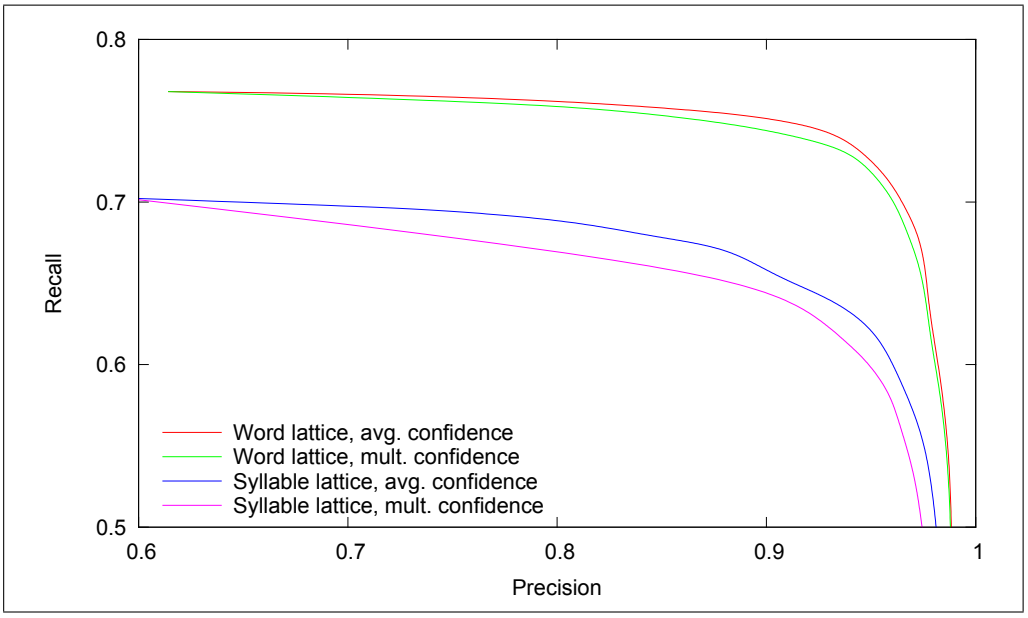

Figure 4.7.: Lattice retrieval with varying online graph pruning threshold.

For the word system, we observe a considerable gain in MTWV of 3% absolute. For the syllable system, MTWV is increased by 5% absolute. Hence, both word and subword lattice retrieval outperform the 1-best baseline in terms of MTWV and should be used whenever the corresponding efficiency requirements are met (see 6 for more details on the scalability aspects of lattice retrieval).

Still, retrieval from lattices suffers from a large drop in precision compared to the 1-best baseline. This is especially true for subword lattice retrieval. In section 5, we will investigate approaches which aim at reducing precision drop inherent to subword lattice retrieval by introducing second-pass STD result verification.

Table 4.2.: STD performance using online graph pruning on unconstrained lattices.

Unit	**Precision**	**Recall**	**MTWV**
Word 1-best	0.95	0.72	0.62
Word lattice	0.51	0.76	0.61
Word lattice with pruning	0.90	0.75	**0.65**
Syllable 1-best	0.94	0.64	0.50
Syllable lattice	0.59	0.70	0.52
Syllable lattice with pruning	0.81	0.68	**0.55**

The results when using online graph pruning on unconstrained lattices are promising.

However, large subword lattices ask for efficient access and search methods as well as for effective storage mechanisms. We will investigate these issues in detail in section 6. However, it is already interesting in the context of this section to which extent the large unpruned graphs contribute to the recall gain, and whether parts of the lattices can be removed at indexing time without affecting the search performance.

For evaluating the effect of offline graph pruning, we start again from the unconstrained lattice set pruned with a graph cut of 20, and retrieve results for the complete query set from both word and syllable lattices. Then, we apply a more intense pruning by decreasing the graph cut parameter towards 1. For each step, we again obtain the pruned lattices and retrieve all results without further online pruning. While varying the offline pruning parameter, we store the retrieval performance as well as the total number of nodes in the pruned lattices for the whole DiSCo set. Figure 4.8 and 4.9 illustrate the results for word and syllable lattice retrieval, respectively.

For both cases, we observe that most of the recall gain can already be obtained from heavily pruned graphs, where only the most promising recognition alternatives are stored in the lattice. Recall gain saturates at a graph cut of 8 for the case of words, and at a graph cut of 10 for syllables. Due to the drop in precision, MTWV is reached even earlier (graph cut of 3 for words and 5 for syllables). This means that we can retrieve from more compact lattices using offline graph pruning, still having access to most of the recall potential of the unconstrained recognition lattices. At the same time, the number of nodes in the system is drastically reduced (e.g., by more than 50% in the MTWV configuration of both word and syllable STD, see table 4.3).

Next, we compare the results of offline and online pruning. Table 4.3 gives the results for the MTWV system configurations. As expected, both perform similar in terms of STD accuracy, as they rely on the same core confidence measure, namely the node confidence. However, the online pruning approach results in a slightly higher MTWV both for word and syllable STD, as it estimates the confidence for the actual query instead of relying only on the local node confidence. On the other hand, offline pruning results in greatly reduced storage requirements.

Applying online pruning to the already pre-pruned graphs at runtime gives additional improvements in terms of MTWV. We observed an increase of MTWV of 1% absolute for both words and syllables when applying online pruning to the offline-pruned lattices used for the MTWV results given in table 4.3.

Next, we look at the results from lattice retrieval on the set of rare queries which

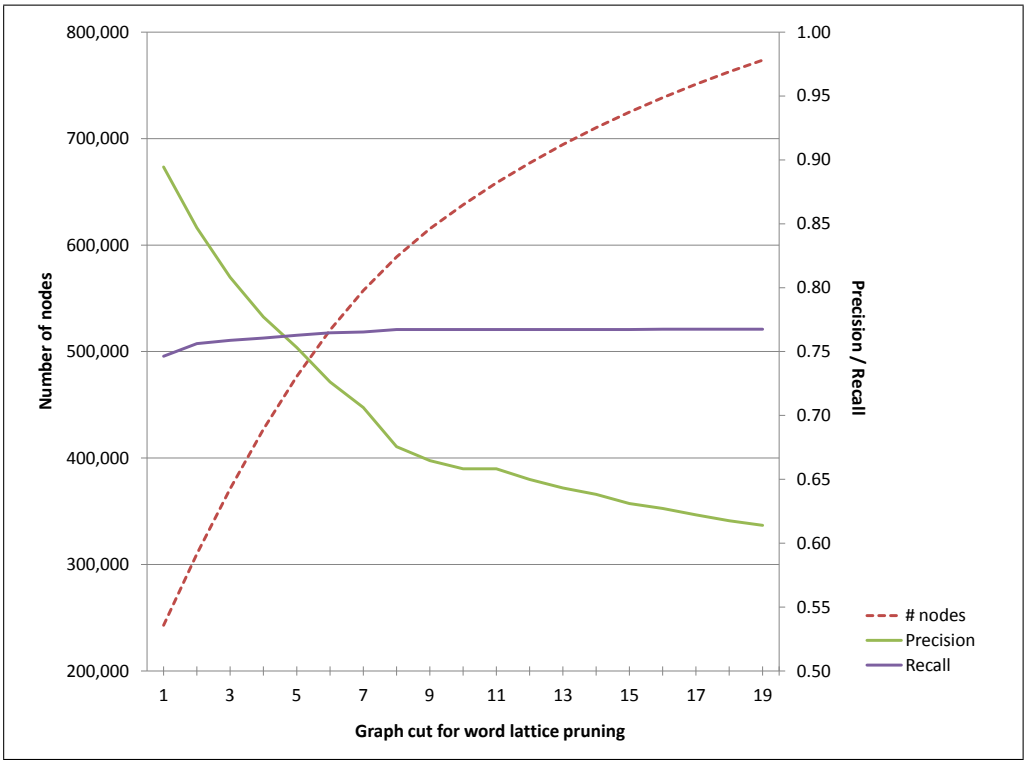

Figure 4.8.: Retrieval from word lattices with varying offline pruning threshold.

contain OOV words with respect to the 200,000 word decoding lexicon. Again, we observe that lattice retrieval outperforms 1-best in terms of MTWV.

Table 4.4 shows that the MTWV improvement when using lattice-based error compensation is more than twice as high as on the complete query set. Another major difference to the evaluation on the complete corpus is the different optimal offline pruning configuration. On the complete query set, MTWV using the syllable system was reached at a graph cut of 5. For the rare queries, MTWV was reached at a graph cut of 18, hence less pruning should be applied to subword lattices if they should be able to cope with rare queries. Unlike in the case of the complete query set, which contains many frequent queries, substantial recall improvements can be obtained on rare queries by investing in larger recognition graphs. This reflects the fact that rare words are less likely to be decoded correctly, which is caused by multiple factors. First, the syllable sequence corresponding to the rare query has not or only rarely been observed during language model training. Moreover, such rare queries often consist of named entities such as names of persons, places or organizations, and the corresponding query phonetization is more likely to consist of rare triphone combinations that have never been observed in

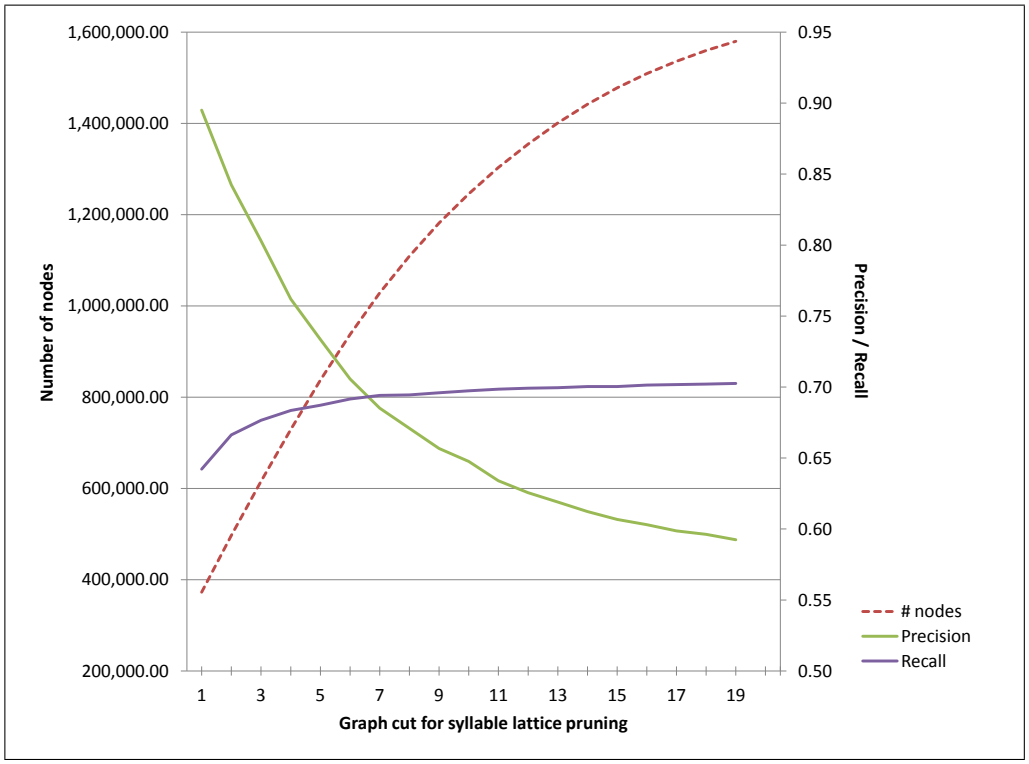

Figure 4.9.: Retrieval from syllable lattices with varying offline pruning threshold.

acoustic training, and which have to be synthesized by acoustic parameter tying. Hence, recognition alternatives help greatly when retrieving matches for such rare queries. As an alternative to storing and searching such large lattices, approximate lattice search on already pruned graphs will be investigated later in this section.

For the complete query set, we conclude that using lattices improves MTWV by 3% absolute for word and by 5% absolute for syllable lattice STD, compared to the respective 1-best baselines on the complete corpus. For rare queries, error compensation using syllable lattices increases MTWV by up to 13% absolute compared to the 1-best syllable baseline.

4.5.2. Compensation by Approximate Matching

Next, we evaluate the impact of approximate subword matching on the STD performance. In particular, we are interested in the following configurations for decoding and retrieval:

1. Using syllables as the decoding unit, and then performing approximate syllable

Table 4.3.: STD performance using online and offline graph pruning.

Unit	Pruning	MTWV	Number of nodes
Word	Online	0.65	773,701
	Offline	0.64	309,894
Syllable	Online	0.55	1,579,991
	Offline	0.53	837,534

Table 4.4.: Syllable STD performance using online and offline graph pruning on OOV queries.

Unit	MTWV	Number of nodes
1-best	0.24	240,927
Online pruning	0.37	1,579,991
Offline pruning	0.36	1,245,659

search on the 1-best syllable output. Alternatively, we can break down the decoded syllables into phonemes and perform an approximate phoneme search on the 1-best syllable-to-phoneme output.

2. We can also use the 1-best output from the word decoder and break it down to either syllables or phonemes, and perform the respective approximate search on the generated subword transcripts.

Again, we refrain from using the direct output from the phoneme decoder due to the low accuracy on the complex evaluation data.

As a baseline for approximate search, we carry out the following experiment. We use approximate syllable search and approximate phoneme search as defined in section 4.3.2, and use equal substitution costs for all phoneme pairs. Then, we perform a complete STD evaluation with different similarity thresholds δ. The reported MTWV is the highest ATWV that could be obtained by varying δ between 0.0 and 1.0.

Looking at the results shown in table 4.5, we observe a drastic MTWV improvement for approximate syllable search over the exact 1-best baseline of 10% absolute. This indicates that approximate search is a viable means for coping with the observed challenges in subword STD, namely compensation of ASR errors and pronunciation variation. For approximate syllable retrieval from word transcripts, we obtain smaller gains. Less ASR errors and pronunciation variation can be found in the subword transcript due to the longer decoding unit, and hence less approximation is required during retrieval.

Table 4.5.: STD performance using approximate match on 1-best transcripts.

		MTWV	
Recognition unit	**Retrieval unit**	**Exact**	**Approx.**
Syllable	Syllable	0.50	**0.60**
	Phoneme	0.51	0.60
Word	Word	0.62	-
	Syllable	0.64	0.69
	Phoneme	0.65	**0.70**

Next, we look in more detail at the behavior of the system if the retrieval units are further broken down to phonemes. As motivated in section 4.3.2, we expect only small additional gain over the syllable baseline. Figure 4.10 compares the results between approximate syllable and phoneme retrieval for both word and syllable ASR output. We note that in both cases, approximate phoneme retrieval is superior in terms of STD accuracy for all possible confidence values. However, the additional recall gain at equal precision is relatively small, and the computational cost for approximate phoneme search is higher due to the longer subword sequences (see section 6 for more details on search efficiency).

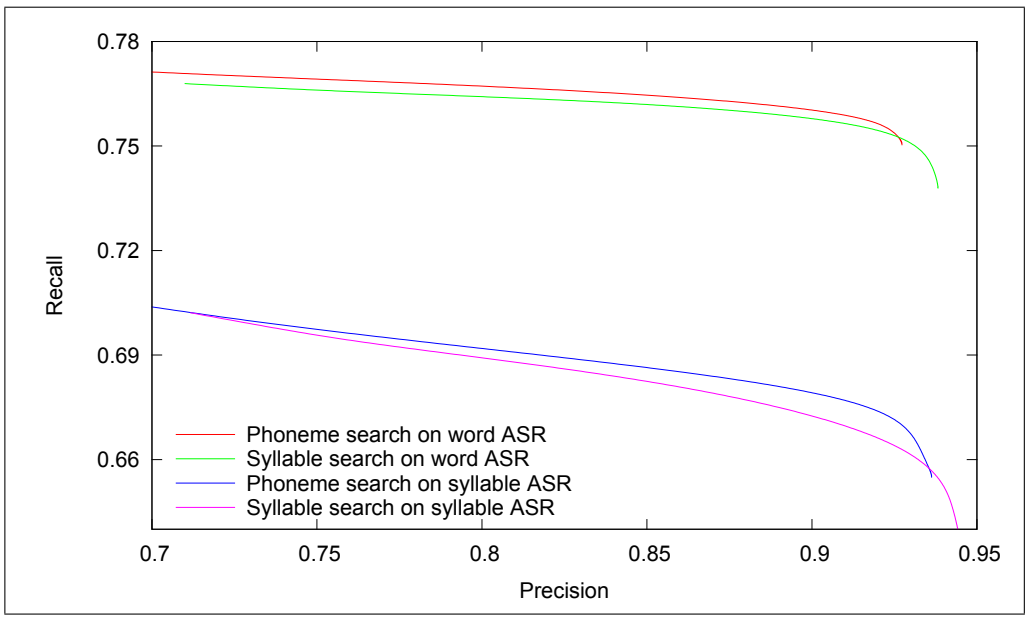

Figure 4.10.: Approximate subword search with varying approximate search threshold.

We further optimize the approximate search by replacing the substitution cost for a pair of subword tokens. As described in section 4.3, we assume that a substitution of phonetically close subwords should be cheaper than substitutions of completely different subwords. The cost for a single pair of phonemes is quantized to be either 0 (equal phonemes), 0.5 for phonemes which are often confused, and 1 for dissimilar phonemes. As illustrated above, we exploit information from a phoneme confusion matrix estimated on some held-out data to asses which phoneme pairs belong to which of the three classes. For estimating the matrix, we considered a parallel German corpus with Broadcast News and Broadcast Conversation shows. We used this corpus in earlier German STD evaluations. It is disjoint with the DiSCo data set used for evaluation within this thesis, but it has similar characteristics. The corpus contains about 3.5 hours of data, with about 50% spontaneous and planned speech, respectively. The utterances do not contain background noise. In order to obtain a phoneme confusion matrix for our approximate search algorithm, we performed a syllable recognition of the data using the same recognizer that is used within this thesis. Then, we broke down the syllable ASR output as well as the reference transcriptions to the phoneme level. By aligning the reference phoneme sequence of each utterance with its corresponding ASR output, we obtained the phoneme confusion matrix, which encodes likely phoneme confusions for 1-best syllable ASR.

In the following experiment, we evaluate different thresholds for the quantization. A threshold of 0 means that all non-equal phoneme pairs will be assumed to be easily confusable, and all pairs receive a reduced substitution cost. A larger threshold means that more confusions must be observed before a phoneme pair is assumed to be part of the *confusable* class. Above a certain threshold, all phoneme pairs will be assumed to be dissimilar, and we will obtain the same results as in the case without using the phoneme confusion matrix, where each substitution cost for non-equal phonemes was set to the maximum cost of 1. The experiment is carried out as follows:

- We focus on a specific approximate search approach, namely approximate syllable search on the output from the syllable decoder, which has already been proven useful for rare OOV queries.

- For each phoneme confusion threshold, we vary the approximate matching threshold and find the configuration with the highest STD performance in terms of MTWV for this particular phoneme substitution cost configuration.

- We look at the results for all queries, IV and OOV queries separately, as we expect different characteristics while varying the thresholds.

Figure 4.11 illustrates the STD performance while varying the threshold for confusable phoneme pairs. We obtain the performance of the baseline without reduced substitution costs for confusable phonemes at a threshold of 0.05 for both IV and OOV queries. Looking at the frequent IV queries, we observe that we can obtain a small gain in MTWV by decreasing the threshold and thereby reducing the substitution cost for a small number of phoneme pairs. However, for rare OOV queries, the potential gain is much larger. Here, MTWV is increased by 8% absolute when comparing the baseline with the best configuration at a phoneme confusion threshold of 0.01. As expected, further reducing the threshold (i.e., including more and more phoneme pairs in the confusable class) decreases overall system performance.

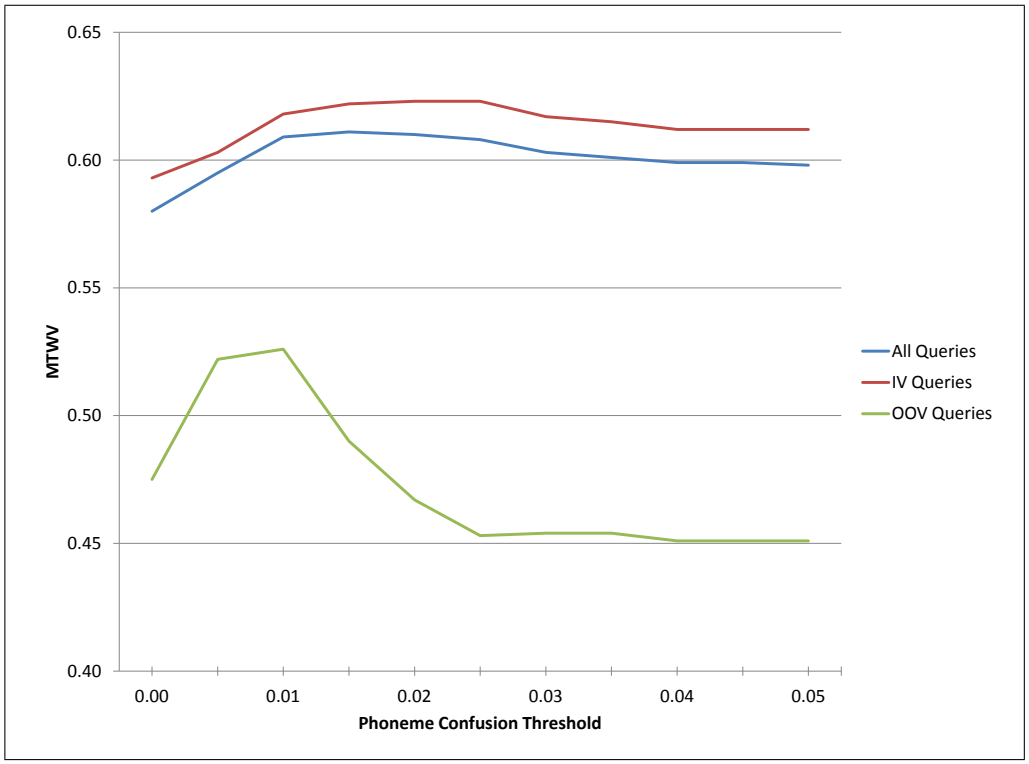

Figure 4.11.: Optimal approximate subword search performance with varying thresholds for phoneme confusion threshold.

Next, we compare the single-point system performance of different configurations for decoding/retrieval units, using both approximate match with and without phoneme confusion matrix. As we could already expect from figure 4.11, table 4.6 shows that for retrieval from syllable ASR output on the whole query set, we obtain only a small

overall improvement by including the phonetic confusion information. When retrieving from words broken down to phonemes, no gain can be observed.

Table 4.6.: Optimizing approximate match with phoneme confusion matrix.

		MTWV	
Recognition unit	**Retrieval unit**	**Equal cost**	**With PCM**
Syllable	Syllable	0.60	0.61
	Phoneme	0.60	0.61
Word	Syllable	0.69	0.70
	Phoneme	0.70	0.70

However, when looking only at the rare OOV queries, we observe a large increase in MTWV when exploiting prior knowledge about likely phoneme confusions as shown in table 4.7. For syllable retrieval from syllable transcripts, MTWV when using PCM is increased by 8% absolute, compared to 1% absolute on the complete corpus.

A similar gain when using PCM can be observed when retrieving from words broken down to syllables. Still, retrieval from actual subword ASR results outperforms the word-based subword retrieval by 6% absolute in terms of MTWV on OOV queries. These rare queries often deviate heavily from the words contained in the decoding lexicon (e.g., artificial proper names of products such as *iPod*). Hence more intense approximation is required to compensate this deviation compared to direct retrieval from subword decoding output, where the decoding output is closer to the actual acoustic realization of the utterance.

Table 4.7.: Optimizing approximate match with phoneme confusion matrix for rare queries.

		MTWV	
Recognition unit	**Retrieval unit**	**Equal cost**	**With PCM**
Syllable	Syllable	0.45	**0.53**
Word	Syllable	0.38	0.47

Table 4.8 summarizes the results for error compensation in vocabulary independent STD. For retrieval from syllable ASR output, we observe that the best syllable lattice retrieval configuration yields higher STD accuracy than the exact 1-best baseline for both query sets (complete and rare), and that the best approximate syllable search even outperforms the best syllable lattice search both on all and OOV-only queries. The

difference between lattice and approximate search is particularly large in the case of rare OOV queries, where the approximate search does not only cover pronunciation variation (which is by design not covered by the lattice search), but also strong deviations between the OOV query and the subword transcript due to the language model mismatch. Decoding long OOV syllable sequences (even as part of a rather unconstrained lattice) becomes improbable if the syllable sequence was never observed during language model training.

When retrieving from word ASR output, the MTWV gain by applying lattices over the 1-best baseline is comparable to the syllable case. Again, approximate search on words broken down to subwords produces the best overall MTWV on word ASR output.

All in all, retrieval from word ASR outperforms retrieval from syllable ASR on the complete corpus, while rare queries are best found on the output from subword ASR.

Table 4.8.: Comparing lattice indexing and approximate match for error compensation.

		Queries	
Unit	**System**	**All**	**OOV**
Syllable	Exact 1-best	0.50	0.24
	Lattice	0.55	0.37
	Approximate 1-best	0.61	**0.53**
Word	Exact 1-best	0.62	N/A
	Lattice	0.65	N/A
	Approximate 1-best (syllable)	**0.70**	0.47

4.5.3. Hybrid Compensation

In this section, we evaluate our proposed approach to hybrid compensation of ASR errors and pronunciation variation. We start by applying approximate search on lattices using different offline pruning thresholds. Then, we compare different approaches for approximation, and look in more detail at the performance on rare queries.

In [78], we have shown that approximate search on lattices yields very poor results on short queries. If we consider a short monosyllabic query, then the amount of available phonetic information is not sufficient for the twofold approximation via lattices and approximate search. Even heavily pruned lattices will very often contain syllables that have a small edit distance to the query, where only few edit operations are needed to transform the query into one of these candidates. Thus, in order to evaluate the hybrid compensation without the negative side effects of short queries, we restrict the

evaluation to queries with at least 10 phonemes (we have observed in [78] that the effect ceases to exist beyond this minimum length). A large subset of the queries in the DiSCo corpus meet the length requirement: 1083 out of 2748 query occurrences cover at least 10 phonemes. The performance on shorter queries will be evaluated in more detail in section 5, where we investigate new approaches to identify false positive hits at query time using additional external information.

We use the following baseline setup for the hybrid compensation cascade, given a query $q = s_1 \cdots s_n$:

1. For a given offline pruning threshold, we obtain all lattices that contain at least one of the query syllables, and retrieve the matching lattice nodes.

2. For each matched lattice node, we obtain all paths of length n through the lattice which contain the matched node.

3. We estimate the similarity between each extracted path and the query sequence, and keep only those results that have a similarity above the STD decision boundary δ. For the baseline approximation, we use the best configuration from the 1-best search, i.e., using the syllable distance with substitution costs based on a phoneme confusion matrix.

For each evaluated offline lattice pruning threshold, we obtain a series of results while varying the approximate search threshold δ. Then, we report the δ configuration with the highest ATWV as the MTWV setup for each offline lattice pruning threshold.

In table 4.9, we compare the results with the best exact lattice search and the best configuration for approximate search on 1-best.

First, we observe that each of hybrid compensation approaches outperforms the individual baselines. This is true for the complete query set, but also valid if we look only at the rare queries. Even when retrieving from very small lattices (GC=2), we can already obtain a small gain in terms of MTWV. With larger and larger lattices, the additional gain over the 1-best approximation becomes larger, but saturates fast (GC=4). Using the still relatively small lattices from this setting, we can already obtain large MTWV gains.

Exact lattice search for long queries with at least 10 phonemes shows a rather poor performance, as each of the syllables has to be matched exactly on the lattice, and small variations or ambisyllabic movement of phonemes cannot be compensated during retrieval. As most of the long queries cannot be matched exactly, we observe a large

increase in MTWV of 26% absolute when comparing exact lattice search to hybrid compensation. Moreover, we obtain an increase in MTWV of 7% absolute on the complete corpus when comparing approximate lattice search to approximate 1-best search, as we pre-select promising lattice paths for the approximate search and can thus apply higher approximation thresholds during search.

Looking at the rare OOV queries, the additional gain becomes even larger, when comparing hybrid compensation to exact lattice search (43% absolute MTWV improvement). Syllable sequences corresponding to the rare OOV queries are not well covered by the language model used for lattice decoding, hence it is rather improbable that long sequences are added to the lattice, even with low offline pruning thresholds. Approximation can overcome these ASR errors that are not covered by lattice compensation.

Table 4.9.: Comparing fuzzy lattice baseline performance to exact lattice and fuzzy 1-best.

		Queries	
System		**All**	**OOV**
Exact lattice		0.47	0.22
Approx. 1-best		0.66	0.57
Approx. lattice	GC2	0.67	0.59
Approx. lattice	GC3	0.71	0.65
Approx. lattice	GC4	**0.73**	**0.65**
Approx. lattice	GC5	0.73	0.65

Next, we compare the performance of different offline pruning configurations while varying the approximation threshold in the second phase of the hybrid compensation cascade. Figure 4.12 illustrates that increasing the amount of node hypotheses in the lattice increases STD performance for all evaluated approximation thresholds, i.e., at equal precision, lower offline pruning thresholds yield higher STD recall. Again, we see that even compact lattices offer additional gain, and that the STD performance improvement becomes smaller as we further increase the amount of hypotheses in the lattice. In the following, we will build upon the best configuration after saturation in terms of MTWV according to table 4.9 (GC=4).

In the experiments above, we have used the same approximation strategy for hybrid compensation which we have used before for 1-best syllable sequence approximation, namely using phoneme minimum edit distance for estimating the syllable substitution cost. As described in 4.5.2, we have estimated the phoneme confusion matrix on some

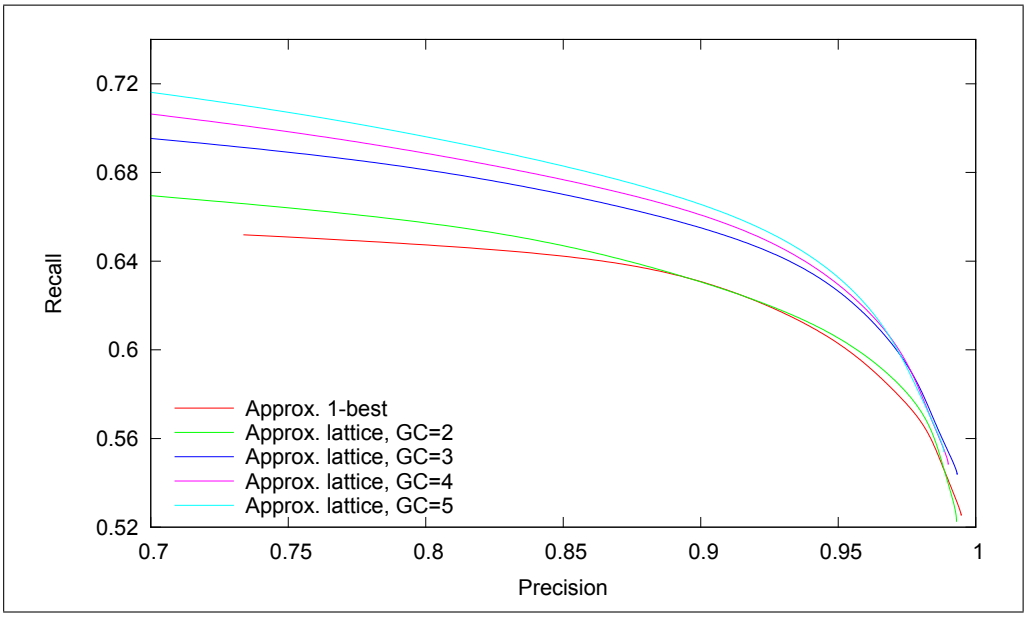

Figure 4.12.: Comparing approximate 1-best search to approximate lattice baseline at different offline pruning thresholds.

held-out data that closely resembles the acoustic characteristics of the DiSCo evaluation corpus. However, it would be interesting to investigate a more constrained means for approximation: as lattices already cope with ASR errors, we could envisage an approximation strategy that focuses primarily on compensating pronunciation variation.

In the following experiment, we will investigate the following alternative, which we expect to yield more precise STD configurations. We estimate the phoneme confusion matrix directly on the acoustic training data instead of using a parallel evaluation corpus. Decoding acoustic training data results in very low error rates due to the maximum likelihood training criterion. Here, the idea is that most of the remaining errors are caused by pronunciation variation, as there is virtually no remaining acoustic model mismatch (the language model mismatch is negligible when using subword transcripts for estimating probable phoneme confusions). In addition, we evaluate whether more constrained subunits for the intra-syllabic minimum edit distance can further increase the precision. We apply position specific phoneme clusters (PSCs) introduced in section 4.3.1, and compare the performance on the complete corpus and on rare OOV queries.

First, we look at the behavior of the baseline approximate search on the 1-best syllable transcript when exchanging the distance metrics. From figure 4.13 we observe that on the 1-best transcript, the proposed approach yields lower STD accuracy compared to the baseline syllable substitution cost for recall-oriented approximation thresholds. As

expected, the approach is less capable of coping with ASR errors, as its focus is rather on compensating pronunciation variation only.

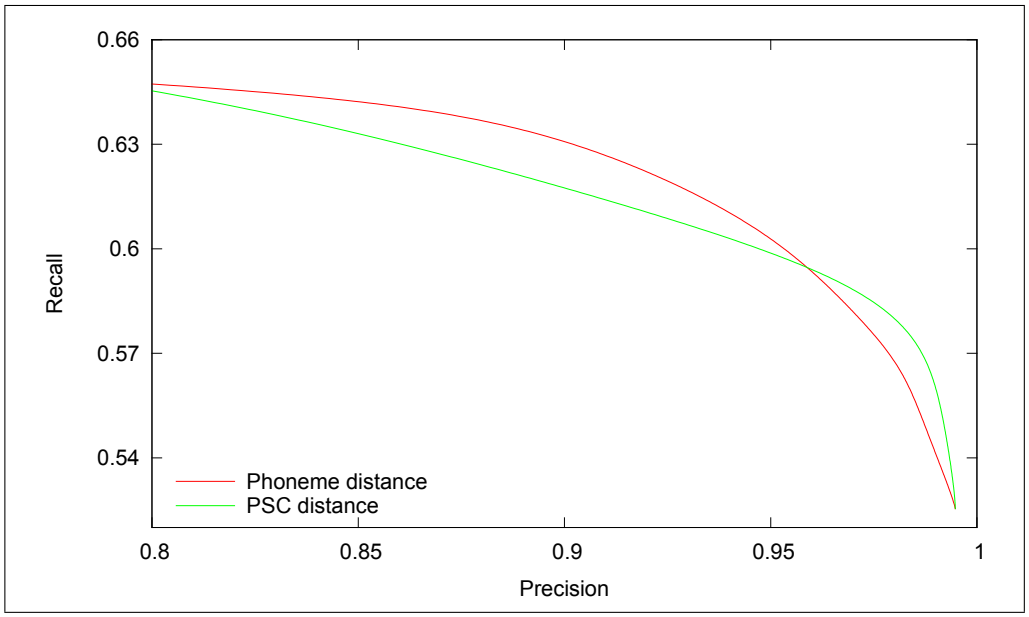

Figure 4.13.: Comparing different syllable distance metrics for approximate search on 1-best.

However, the situation is different when looking at hybrid approximate retrieval from the lattice. Figure 4.14 shows the results using the same approximation configurations that were used for generating the results in figure 4.13, only retrieving from lattices instead of 1-best transcripts. As expected, structural constraints using position specific clusters yields the best results. We observe consistent improvements of our proposed PSC-based approach over the baseline.

Next, we look at the performance on rare queries. Here, we know from the experiments above that the lattice still contains many ASR errors, even at low offline pruning thresholds. Figure 4.15 shows that additional compensation beyond tolerating pronunciation variation is needed in this case, as the baseline approximation strategy yields the best overall STD results and outperforms the more constrained PSC-based approach. Note that we could bridge the gap between the baseline and the more focused PSC search by using lower offline pruning thresholds when generating the lattices, which in turn contain less ASR errors. However, this comes at the cost of storing and retrieving from much larger lattices, which is prohibitive for many scenarios (see section 6).

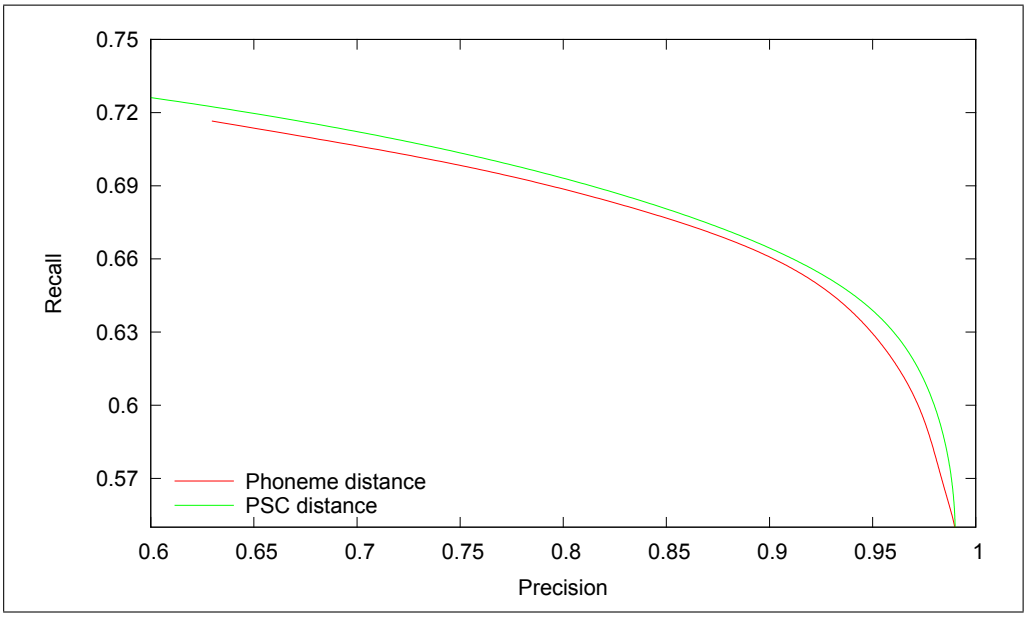

Figure 4.14.: Comparing different syllable distance metrics for approximate search on pruned lattice with GC=4.

As subword search is particularly important for rare queries, we will use the phoneme approach trained on comparable development data as the default strategy for estimating syllable similarities during approximate syllable search.

Finally, table 4.10 summarizes the results for hybrid compensation, where we observe large gains in MTWV compared to the individual baselines. For completeness, we also give the results on the whole corpus (i.e., including short queries with less than 10 phonemes).

Table 4.10.: Comparing Syllable STD performance of approximate lattice indexing with individual baselines.

	Queries	
System	**All**	**> 10 phonemes**
Exact 1-best	0.50	0.47
Lattice, online pruning	0.55	0.55
Approximate 1-best	0.61	0.66
Approximate lattice, offline pruning	0.63	**0.73**

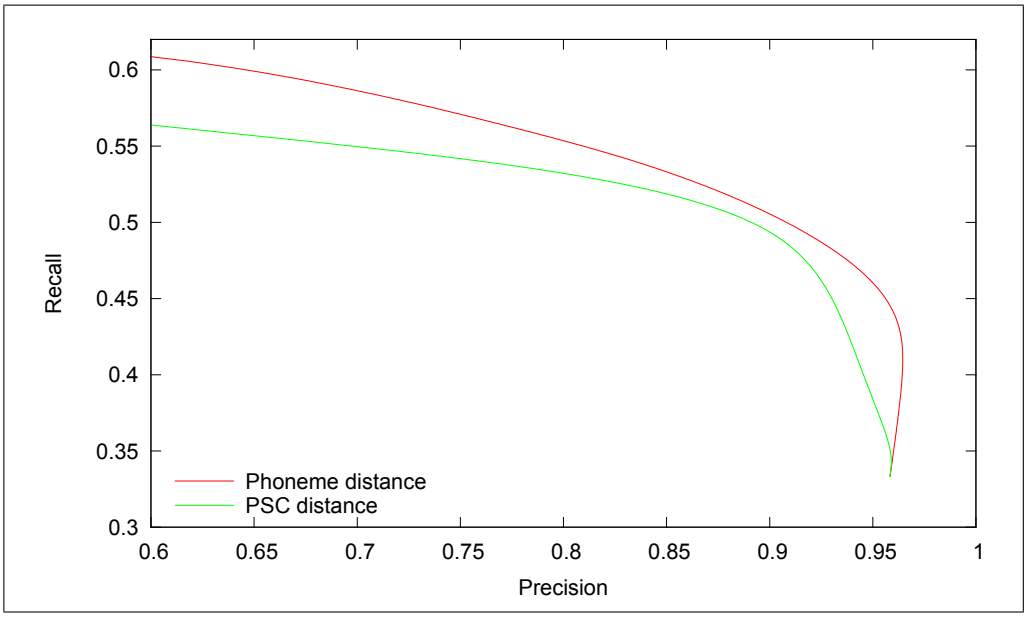

Figure 4.15.: Comparing different syllable distance metrics for approximate OOV search on pruned lattice with GC=4.

4.6. Summary

The preceding evaluation confirmed that error compensation is needed in subword STD in order to cope with inevitable deviations between subword transcript and subword query sequence. Within this chapter, we have investigated and extended a range of state-of-the-art techniques that allow for error-tolerant search, and which are especially suited for subword-based STD. Our hybrid approximate syllable lattice approach, which we have first presented in [78], improves MTWV on rare OOV queries by 40% absolute to 0.60 over the exact 1-best syllable baseline. The hybrid approach outperforms both lattice and approximate 1-best search, and thereby effectively merges the corresponding subword STD error spaces.

First, we have analyzed the error sources in STD such that the best compensation strategies could be derived accordingly. We found that errors in subword-based STD stem from two different sources: subword ASR errors and pronunciation variation. This is different from the case of word-based STD, and has not yet been addressed explicitly.

Then, we introduced lattices a means for explicit compensation of ASR errors, which has been successfully applied in several languages and subword units. We have proposed on- and offline pruning techniques that allow for flexible lattice configuration depending on the requirements of the STD scenario. Next, we have described a two-stage approxi-

mate search based on Minimum Edit Distance. Starting from the baseline from [60], we explore a distance measure focused on pronunciation variations based on position-specific syllable clusters, which we have first introduced in [78], and successfully applied in [79] and [7]. Combining both methods into a hybrid approximate lattice cascade effectively merges the identified search spaces. From the experimental evaluation of the proposed hybrid combination we can draw the following conclusions:

- On the complete query set, hybrid approximate matching increases syllable STD performance from 0.50 to 0.63, and it is particularly suited for longer queries, where MTWV is increased by 26% absolute from 0.47 to 0.73.
- The position-specific cluster approach based on cluster confusion from acoustic training data outperforms the baseline distance metric on IV queries. Here, the deviation between lattice and canonical query sequence can be better compensated by the predicted pronunciation variations.
- However, for rare OOV queries, fewer ASR errors can be compensated by the pruned lattice. Hence, the less constrained phoneme-based distance metric based on phoneme confusion matrix from actual development data performs better.
- For long OOV queries of at least ten phonemes, our proposed hybrid approach increases MTWV by 43% absolute over the baseline (from 0.22 to 0.65).
- For IV queries, the best results could be obtained with approximate search on word transcripts broken down to subwords, although the improvement is moderate compared to the case of OOVs (MTWV of 0.74 compared to the exact baseline of 0.70).

5. Verification of Spoken Term Detection Results

The preceding section on error compensation has shown that the approximate compensation cascade consisting of lattice and approximate search is able find new true positive hits in our heterogeneous evaluation data set. However, the additional recall gain comes with a drastic loss in precision, especially for short queries. Figure 5.1 illustrates the effect of the different approaches on retrieval precision and recall. Adding more and more retrieval flexibility via error compensation inevitably decreases retrieval precision. In this section, we propose a novel approach for verifying STD results using external knowledge in order to increase retrieval precision without loss of recall, which we have first published in [95].

We exploit the fact that the STD system has access to more information about the query at search time than at indexing time. This information advantage at search time is twofold: first, the query itself was not available during indexing, hence the indexing process was not optimized towards detecting the query. Second, the additional information about the query might not have even existed at the time of indexing. In this chapter, we investigate a novel approach to exploit this knowledge in order to verify whether a putative STD result is correct or not. Section 5.1 describes our generic process for result verification using external knowledge.

Next, we describe two approaches which represent actual implementations of the verification process: *contextual verification* and *anti-query verification*. In contextual verification, we use local contextual query information as knowledge, i.e., typical neighboring words or subwords that occur as contexts of query terms, and verify whether the ASR output agrees with our hypothesized query context. This contrasts to SDR approaches such as [51], which aim at expanding queries with related terms from the same topic, whereas STD verification is considered to be topic- and domain-independent. The work proposed in [64] can be considered as a progenitor, where the authors evaluated a method for 1-best phoneme expansions for known country-name queries, whose contexts are highly regularized. In contrast to contextual verification, anti-query verification ex-

ploits external knowledge to a different end: here, we look for competing phonetically similar queries that are likely to produce false alarms for a given query, the so-called *anti-queries*. Based upon our investigations in [95], both novel approaches are evaluated in detail in section 5.5.

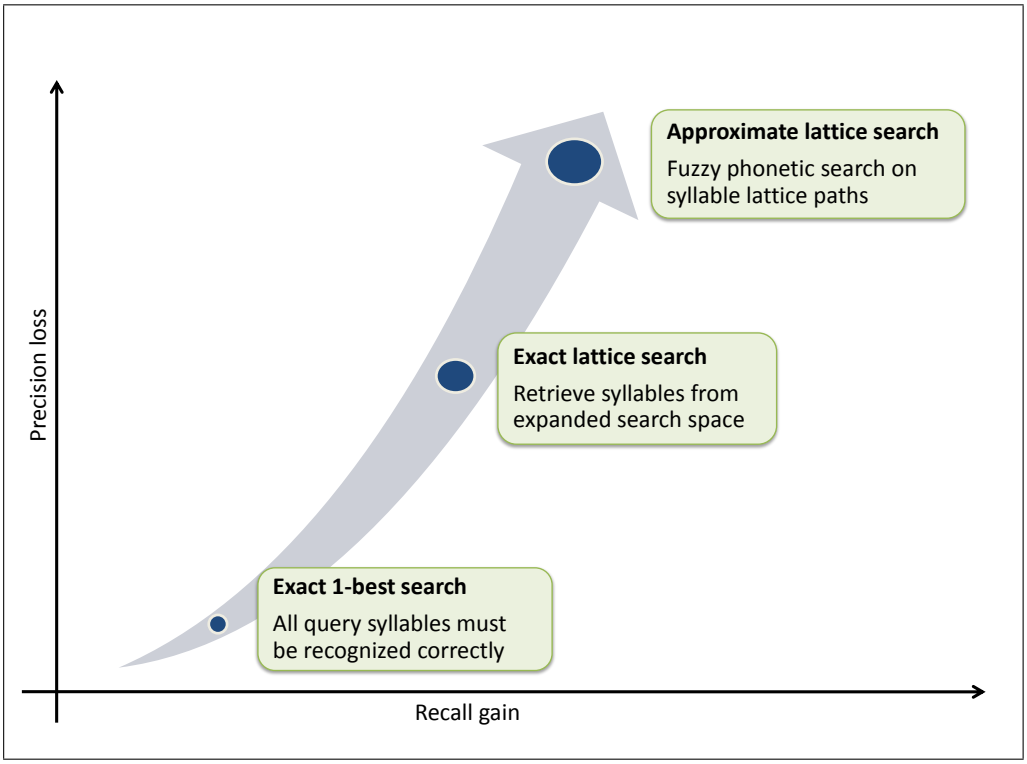

Figure 5.1.: Recall gain and precision loss in STD error compensation.

5.1. Generic Verification Approach

Our new approach is based on a two-pass strategy as illustrated by figure 5.2. In the first pass, an error-compensating search is carried out on the ASR output. This search is tuned towards recall in order to obtain as many true positive hits from the ASR output as possible. Then, in the second pass, we remove unlikely false alarms with one of the proposed verification strategies.

This process has two core advantages:

1. The verification step can exploit *external query knowledge* which is only available

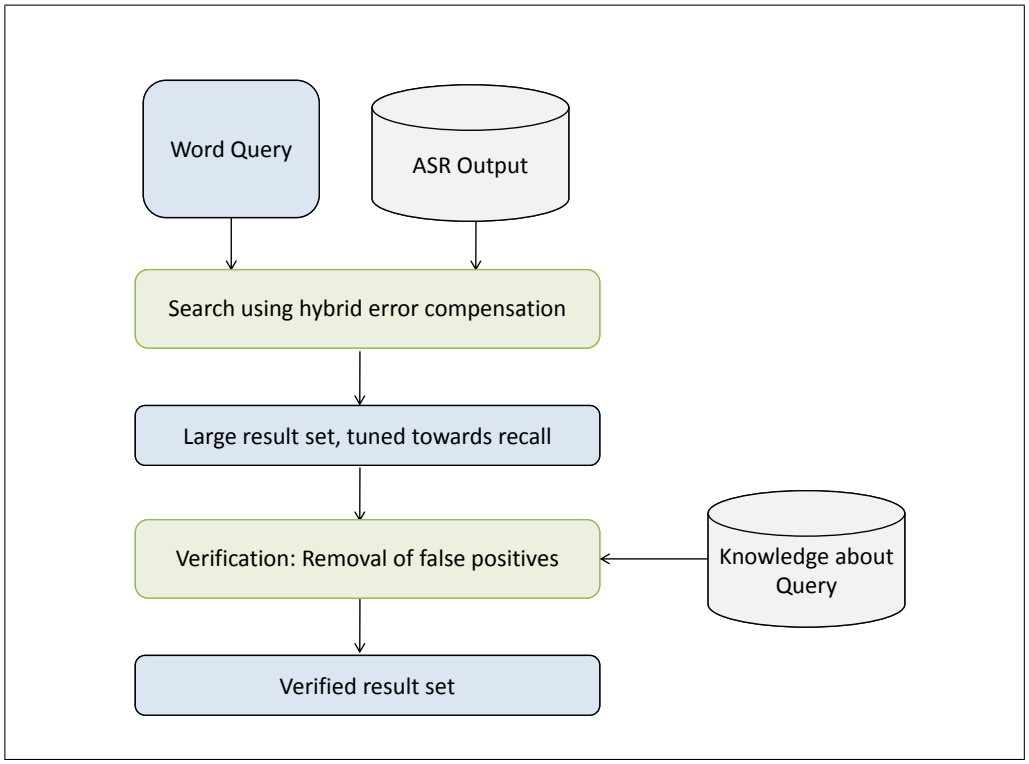

Figure 5.2.: Generic process for STD result verification.

at query time. For example, we can collect word contexts that are typically spoken around a query, or we can identify other queries that typically lead to false alarms for the query in question. External knowledge can be obtained even if the query is rare and not part of a typical word decoding lexicon, or if the corresponding subword sequence has not been observed in language model training text. The notion of *out-of-vocabulary* is not applicable anymore, as we can obtain such external knowledge for virtually any relevant query via available Internet resources.

2. The system is equipped with a new parameter (namely the degree of verification) that can be used to adapt a search result to user needs: a user could first review the verified set of search results, which is rather compact. Then, if his information need is not fulfilled, he could add more and more results from the set of results that did not pass the verification step, knowing that this also increases the probability that a result is not correct.

One might ask why splitting up the decision into two individual steps should increase STD accuracy, and why the verification cannot be integrated in the first step. The reason

for this is twofold. Our verification scheme targets primarily hybrid error compensation for subword decoding, as we observe the largest precision loss when applying the error compensation. In this case, the syllable language model used during ASR decoding is trained on the syllabified version of a large language model training corpus. Hence, the ASR decoder does not have access to word level information, such as word boundaries, as these are lost when converting the training text to the syllable level. At query time, we can use the original word query to obtain information such as typical contexts on the word level, which we can then exploit to verify the result. This adds new information to the process, even if we use exactly the same training text that was used to train the syllable language model. In addition, we can use new textual resources that were not available or that were not thought to be relevant at indexing time.

As an example, consider the name of a small town that is struck by an earthquake. The town name might be rare and not part of the decoding lexicon, and we need to search the subword transcript. At query time, we can obtain a large amount of typical word contexts that typically surround the town name, which we can break down to syllable sequences and exploit these in the verification of the result. We might use an Internet news feed as a source for the word contexts.

The actual verification step has to be carried out *online*, i.e., at query time. The collection of the external knowledge can be carried out both at indexing and query time, depending on the scenario. For example, in the media observation case, a regular update of the verification models is mandatory, as new terms can come up every day.

Formally, we define the STD verification step as follows. Given an STD result $o(q) = (s, t_s, t_e, c)$ for a query q, we generate a verification hypothesis $h = s_1 \cdots s_n$ at query time. The verification hypothesis is a sequence of subwords $s_1 \cdots s_n$ which we assume to be spoken at the position of the hit.

Then, the STD verification step decides whether the subword hypothesis h was spoken at the hit position or not. The idea is to perform another subword STD search for h on s, and compare the result with the hit $o(q)$. Searching for the subword verification sequence h, we obtain a set of STD hits. Each hit can be described as $o(h) = (s, t_{s_h}, t_{e_h}, c_h)$, where t_{s_h} and t_{e_h} are the start end end times of the hit alignment between h and the ASR output, and c_h is the confidence of the alignment. This idea can be apploed on 1-best and lattice output in the same way.

A putative hit $o(q)$ is then verified by a verification hypothesis $o(h)$ if the following conditions are met.

1. $o(q)$ and $o(h)$ occur at similar timestamps, i.e., they represent the same hit region in the ASR output. In the following, we will use the idea of the NIST STD evaluation

plan [82], where the authors assume that two hits stem from an equal ASR region if the timestamps at the respective centers differ only by a small timespan ϵ.

2. The STD confidence of $o(h)$ is at least as high as the confidence of the putative hit $o(q)$.

Hence, we can define the verification of $o(q)$ using h as a binary function V, where

$$V(o(q), h) = \begin{cases} 1 & \text{if } \exists o(h) : c_h \geq c \text{ and} \\ & \tau(o(q), o(h)) < \epsilon \\ 0 & \text{else} \end{cases} \tag{5.1}$$

We use the following definition for the temporal distance between two aligned hits as given by [82]:

$$\tau(o(q), o(h)) = |mid_{o(q)} - mid_{o(h)}| \tag{5.2}$$

where the center of an aligned hit $o(q)$ is given by

$$mid_{o(q)} = t_s + \frac{t_e - t_s}{2} \tag{5.3}$$

In the following sections, we investigate two different techniques for verification based on external knowledge, which both use the described verification scheme:

1. **Contextual verification:** The verification system rejects the putative hit if the subword context around the result in the ASR output is not predicted by the context verification model (section 5.2). This is a *positive* verification, i.e., the actual query is extended with likely contexts and then used in the described verification process.

2. **Anti-query verification:** The putative hit $o(q)$ will be rejected if a phonetically similar anti-query exists (section 5.3), which is known to cause false alarms for the given query. This is a *negative* verification, i.e., the system tries to verify the putative hit with competing queries. If a competing query yields a better match, the putative hit will be removed from the result set.

Contextual query verification focuses on rejecting putative hits whose context is unlikely. Anti-query verification removes results which are closer to a phonetically similar query than to the actual search term. Both approaches cover different aspects, hence

we expect a performance increase from combining the methods in a hybrid verification system.

5.2. Contextual Verification

In the following, we describe an approach for contextual verification which we have proposed in [95]. The intuitive idea is to remove those STD results from the result set where the local context in the ASR output is highly unlikely for the given query. We exploit the fact that we have additional knowledge about the query at query time, that was not available when the subword ASR was carried out.

As a motivation, consider the following example for a spoken utterance *Hoffenheim spielte in München - Hoffenheim was playing in Munich.*

- We assume a perfect subword transcription from the syllable ASR, i.e., *h_O_ f_@_n_ h_aI_m_ S_p_i:_l_ t_@_*. Note that we do not have access to the word boundaries at query time.
- A user queries the system for the term *Hockenheim*, the name of a German race course.
- Using approximate matching, only one consonant needs to be substituted to align the two syllable sequences *h_O_ f_@_n_ h_aI_m_* and *h_O_ k_@_n_ h_aI_m_*. Hence, error compensating STD will most likely produce a false alarm for the query *Hockenheim*, even using a high approximation threshold.
- However, at query time, we can obtain additional contextual knowledge about the query. For instance, we know that *Ring* is a highly probable right context for the word *Hockenheim*.
- We expand the query with this probable context, and verify whether the resulting syllable sequence *h_O_ k_@_n_ h_aI_m_ r_I_N_* is also found with a reasonable confidence by the STD system. If not, we remove the hit from the result set.

We note that query expansion and verification is only carried out locally with respect to the query occurrence. Hence, we can assume that this approach stays within the topic-independent boundaries of STD. Starting from the example above, we will investigate the following questions in the sections below:

- How can we select a sufficiently large and appropriate set of contexts for a query?

- How to perform the actual context matching, such that only few true positives are removed due to missing contexts?
- In some scenarios, response and storage efficiency are important success criteria for STD. How can we remove contexts and thus reduce the computational burden during verification, without loosing too much recall?

5.2.1. Collecting Query Contexts

First we describe an approach to obtain a set of probable contexts for a given query. It is desirable to cover as many valid query context solutions as possible, as only results with a valid context will survive. For the same reason, we verify with left and right context separately. Our system collects a set of contextual queries for a query q using the following idea:

1. We mimic a search resulting in a true positive hit by locating exact occurrences of q in a parallel textual corpus on the word level.
2. Then, for each occurrence, we store the left and right subword contexts of the match as candidates for contextual verification.

The textual corpus $c = w_1 \cdots w_r$ does not contain time stamps, hence we adapt the definition of an STD search hit occurrence from equation 2.3 as follows:

$$o(q) = \{w_s, w_e, c\} \tag{5.4}$$

where s and e are the indices of the first and last word that are covered by the approximate STD alignment. If available, we perform the aforementioned exact search on a textual corpus that closely resembles the actual decoding situation, such that we can expect to observe the query and its most probable contexts. Let $C(q)$ be the set of contextual queries that will be used for verification of a putative hit $o(q)$ from the first STD pass for a query q. Then, we construct $C(q)$ according to algorithm 4.

The described process produces a set of contexts that are likely to be observed for a given query. However, the set does not necessarily contain the most appropriate contexts for all decoding situations:

- *Insufficient contexts* occur if a query has existed while collecting the corpus, but has become more important at query time. For example, the football club *Hoffenheim* was not in the first division of the German football league until 2008, and it was

Algorithm 4 Construct set of contextual queries $C(q)$ for a query q, with context length k.

Let $c = w_1 \cdots w_r$ be the external parallel corpus.
Break down c to syllables $c_s = s_1 \cdots s_n$.
Perform exact word search for q on c and obtain set of textual STD results $o(q) = \{w_s, w_e, c\}$.
for all $o(q)$ is a hit from the result set **do**
  Obtain syllable sequence $s_i \cdots s_{i+t}$ of t subwords that is covered by query q
  Obtain left subword context $c_l = s_{i-k} \cdots s_{i-1}$
  Obtain right subword context $c_r = s_{i+t+1} \cdots s_{i+t+k}$
  Store contextual query $c_l(q) = c_l s_i \cdots s_{i+t}$ in $C(q)$.
  Store contextual query $c_r(q) = s_i \cdots s_{i+t} c_r$ in $C(q)$.
end for

only mentioned once in the complete DPA corpus, yielding only a single contextual query for this term. However, it occurs several times in the DiSCo evaluation set as an OOV query. Using the best fuzzy lattice search approach in section 4.5, 24 occurrences of the term *Hoffenheim* were correctly found. From these, 8 correct hits would be falsely removed by contextual verification, because the single available context did not even produce a fuzzy match. From the remaining 16 correct hits, only 8 could be verified exactly, i.e., without approximate matching. The high amount of fuzzy matches is characteristic for infrequent queries. The corresponding n-grams have not been observed often during subword language model training, and the corresponding query triphones have not been trained well in acoustic training. In the Hoffenheim example above, only 33% of the correct hits could be contextually verified with a 1-word context exactly. Looking at all queries that have the same length as the query *Hoffenheim*, over 90% of the corresponding true positive hits could be verified using contextual queries.

- *Inappropriate contexts* are collected from the corpus if the typical meaning or typical usage of a query word has changed over time between collecting the training data and issuing the query. As an example, consider the query *Obama*. The contextual verification *Senator Obama* that could be collected from the DPA corpus was widely used in 2006, when Barack Obama was a senator in Illinois. However, in 2008, the verification *President Obama* has become much more important, but is not available at all from the DPA corpus. Ideally, we would augment the training corpus with up-to-date material, while preserving the original contexts.

In the evaluation below, we will investigate the effect of using different parallel corpora for obtaining the query contexts. We expect that the overall performance is higher if we use up-to-date expansion corpora matching well with the query, and that we can obtain additional gain by using additional data which differs from the original language model training corpus.

5.2.2. Detecting Non-Contextual Matches

We only validate non-exact matches where we expect that verification can improve the low baseline precision. Consider a putative hit occurrence $o(q)$. Intuitively, our approach assumes that $o(q)$ is a correct hit if the system also detects q expanded with a local context, at the same position and with a similar confidence. This is a *positive* verification, i.e., we only keep results where we already have external evidence that the local context is valid. Hence, we design the matching procedure such that only very unlikely contexts cause a hit removal, and that matching of contextual queries is facilitated. In addition to adding as many valid contexts to the verification set for a query as possible, we expand the query with left and right context individually in order to increase the possibilities for matching a given contextual query.

Algorithm 5 describes the process for detecting hits without proper context. It can be applied in the same manner to both approximate 1-best search and approximate lattice search.

A possible drawback of this matching approach is the fact that for all non-exact matches of the first STD pass, the spoken context of a query needs to be actually observed in the parallel corpus. However, we can assume that virtually all valid word contexts for a given query are available through web resources.

For highly spontaneous and non-professionally spoken utterances, the word context might not be *valid* in terms of grammatical correctness, and it will become unlikely that these spoken contexts can be observed in written text. For this special case, additional means for smoothing similar to language model smoothing [16] could be helpful. However, this kind of data rarely exists in our evaluation scenario as defined in 2.4.2.

5.2.3. Contextual Query Optimization

Contextual pruning is a *positive* verification, where we would like to add all possible contexts to the set of verification hypotheses. However, at query time, some scenarios will require further possibilities for increasing precision, especially if the baseline precision from the first STD pass is as low as shown in the hybrid approximate lattice experiments

Algorithm 5 Verify an STD result o(q) with a set of contextual queries C(q).

$o(q) = \{s, t_s, t_e, c\}$
if $c < 1.0$ **then**
  contextual_match = false
  Let s_{syll} be the syllable ASR output for document s
  for all contextual query $d = d_1 \cdots d_r \in C(q)$ **do**
    Perform approximate search for d on s_{syll}
    for all contextual hit occurrence $o(d) = \{s, t_{s_d}, t_{e_d}, c_d\}$ **do**
      $mid_{hit} = t_s + \frac{t_e - t_s}{2}$
      $mid_{ctx} = t_{s_d} + \frac{t_{e_d} - t_{s_d}}{2}$
      if $|mid_{hit} - mid_{ctx}| > \epsilon$ **then**
        Continue
      end if
      if $c_d \geq c$ **then**
        contextual_match = true
        break
      end if
    end for
  end for
  if contextual_match = false **then**
    Remove o(q) from result set
  end if
end if

above.

Due to the Zipfian distribution of syllable frequencies (see section 6.1), we can assume that many syllable contexts only occur rather infrequently. However, if the set of unique contextual queries is sufficiently large, there will always be a contextual query that can be found at the putative hit position using approximate STD, leading to an incorrect verification of a false positive hit. Our idea is to remove the most infrequent contexts for a given query to reduce this effect, and we expect to further increase STD precision at small recall loss.

Hence, if additional precision is required, it can be beneficial to remove the most unlikely contexts from the result set, and verify the putative hit only with a reduced verification set. Similar to standard approaches in language modeling, we can estimate the probability that a context occurs by obtaining its relative frequency on a parallel textual corpus, i.e., the prior probability $p(h)$ for the contextual query h obtained for a syllabified query q_s is given by:

$$p(h, q_s) = \frac{N(h)}{N(q_s)} \tag{5.5}$$

where $N(h)$ is the number of times h occurs in the syllabified parallel corpus, and $N(q_s)$ is the number of times q_s occurs in the same data set.

Then, at verification time, the user can specify a verification threshold κ, and we remove all contexts where $p(h, q_s) < \kappa$.

Another possible direction for the optimization of contexts is the context width, i.e., the number of syllables that are added to the original query. Our expectations when increasing the number of syllables are twofold:

1. STD verification using a longer context hypotheses will be more reliable, as STD results for longer queries are more accurate. Hence, we expect only few false positive verifications caused by long contextual queries.

2. Increasing the context width will drastically increase the amount of contexts that are available for verification. There might be need for combining this idea with contextual query pruning as described above in order to limit the size of the verification query set.

We note that the number of syllables in the context should not be exceedingly high, as we might loose topic independence, which is an important requirement for STD systems.

5.3. Anti-Query Verification

In [95] we have observed that false alarms for a particular query in approximate subword retrieval are often caused by phonetically similar subword sequences in the reference transcript. Consider the following example:

- The system searches for the query *Bayern - Bavaria* in the subword transcript with the corresponding syllable sequence *b_aI_ 6_n_*.
- Consider the term *Arbeitern - employees*, which is syllabified *Q_a_6_ b_aI_ t_6_n_*. If an occurrence of this term is transcribed correctly by the syllable ASR, than the system will generate a false alarm, even at high confidence levels for the approximation. This is due to the high similarity between *b_aI_ 6_n_* and *b_aI_ t_6_n_*.
- The effect is increased because of the rich morphology of the German language. In the example above, false alarms will also be caused by other flexions of the term (such as *Q_a_6_ b_aI_ t_6_*).
- Even more false alarms are caused by compounding, if the word that causes the false alarm is combined with another word yielding a new meaning (such as *Arbeiterklasse - working class - Q_a_6_ b_aI_ t_6_ k_l_a_ s_@_*).

Inspired by the work in [12], we call these word sequences that are likely to cause a false alarm for a query q an *anti-query* for q. In the following, we formally describe our approach to collecting such anti-queries from an external text corpus, and propose a method for detecting false alarms caused by anti-query matches.

5.3.1. Collecting Anti-Queries

We collect a set of anti-queries for a query q using the following idea:

1. We mimic the search behavior of approximate search on a parallel corpus with ground truth, i.e., on a corpus where we know whether a search result is correct or not.
2. Then, for a query q, we inspect all positions in the corpus that caused a false alarm, and construct an anti-query using the words that actually occur at the false hit position.

However, we refrain from searching lattice or 1-best ASR output for collecting the anti-queries. Audio data and corresponding aligned reference transcriptions would be

required in order to generate anti-queries from ASR output. While we could closely imitate the error behavior with this approach, it would not be feasible to collect enough data for all possible queries, as the cost for producing manual transcriptions is to high (see section 2.4.2).

Instead of searching ASR output directly, we collect anti-queries for a query q by searching existing large textual corpora. Ideally, the set of anti-queries is collected from a corpus with similar characteristics as the transcriptions that are typically generated by the subword ASR, such that the differences between the predicted behavior and the actual decoding output is small. Hence, we consider using the same text that was used for training the subword language model in the evaluation (section 5.5).

We break down the external textual corpus to syllables. This resembles an ASR output with 0% syllable error rate. Then, an approximate search for q is carried out on the syllabified corpus. If q is found by the search and the hit is a false positive, then we collect an anti-query from the hit position.

The syllabified textual corpus $c_s = s_1 \cdots s_r$ does not contain time stamps, hence we adapt the definition of an STD search hit occurrence from equation 2.3 as follows:

$$o(q) = \{s_s, s_e, c\} \tag{5.6}$$

where s and e are the indices of the first and last syllable that are covered by the approximate STD alignment. Let $A(q)$ be the set of anti-queries that will be used for detecting false alarms for a query q. We construct A according to algorithm 6.

Algorithm 6 Construct set of anti-queries $A(q)$ for a query q.

Let $c = w_1 \cdots w_r$ be the external parallel corpus.
Break down c to syllables $c_s = s_1 \cdots s_n$.
Perform approximate search for syllabified q on c_s and obtain set of textual STD results $o(q)$.
for all $o(q) = \{s_s, s_e, c\}$ is a hit with confidence below 1.0 **do**
  Obtain word sequence $w_i \cdots w_{i+t}$ of t words that is covered by the subword match:
  $w_i = s_{i,1} \cdots s_s \cdots s_{i,p}$ and
  $w_{i+t} = s_{i+t,1} \cdots s_e \cdots s_{i+t,k}$ and
  Store $a(q) = s_{i,1} \cdots s_{i+t,k}$ as an anti-query for q.
end for

Consider again the query *Bayern - bavaria* and the corresponding syllable query *b_aI_ 6_n_*. Figure 5.3 illustrates the selection of the anti-query *Arbeitern - employees* using the algorithm described above (only one word is covered by the match in this example).

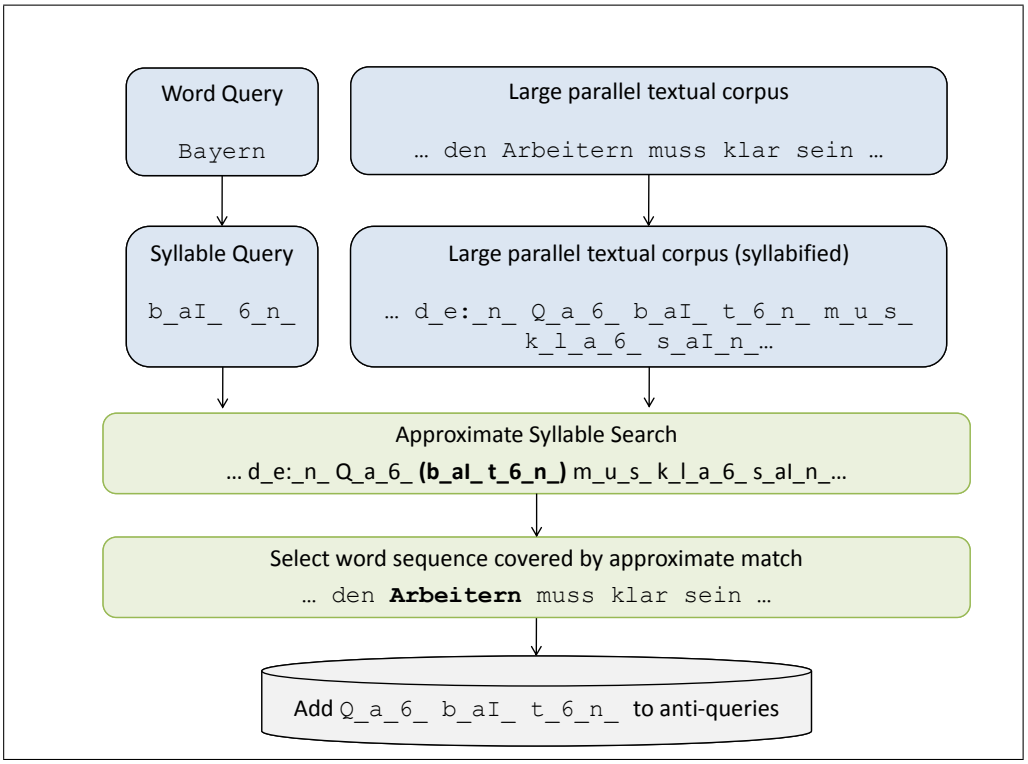

Figure 5.3.: Example: selection of anti-query.

As a baseline, we collect all anti-queries that cause a false alarm above the anti-query approximation threshold. In section 5.3.3, we will describe possibilities for further improving the anti-query set using additional anti-query context from the external corpus, and by removing anti-queries that are likely to remove true positive hits.

In the next section, we describe how putative hits for a query q are verified against the anti-query set $A(q)$.

5.3.2. Detecting Anti-Query Matches

As motivated earlier, we will only verify non-exact matches, as the precision of exact hits is already very high. For each query q, the system first obtains STD results using error compensation as described in section 4, either from the 1-best or from the lattice results. Then, for each each hit occurrence $o(q)$, the verification decides whether the hit is correct or not. Intuitively, we assume that a putative hit is a false positive result if one of the queries in the anti-query set $A(q)$ matches better than the original query q at the same hit position. We assume two hits relate to the same temporal region if the centers of the two approximate alignments differ only by a small time period ϵ (similar

to the approach in section 5.2).

Anti-query verification is a *negative* verification where we verify that a result is *incorrect* using a better-matching counter example. In order to further constrain the verification and reduce the amount of true positive removals, we perform the anti-query match on the 1-best transcript instead of the lattice output, even if the original approximate search was carried on the lattice. With this approach, it is more likely that anti-queries will only match with a higher score than the original search if the anti-query was actually spoken.

Let $o(q) = \{s, t_s, t_e, c\}$ be the hit occurrence as defined in section 2.1. Then, algorithm 7 describes the process for verifying the result using the already collected anti-queries $A(q)$.

Algorithm 7 Verify an STD result o(q) with a set of anti-queries A(q).

$o(q) = \{s, t_s, t_e, c\}$
Let $s_{syll} = s_1 \cdots s_n$ be the 1-best syllable transcript of s
if $c < 1.0$ **then**
  for all anti-query $a = a_1 \cdots a_r \in A(q)$ **do**
    Perform approximate search for a on s_{syll}
    for all anti-query hit occurrence $o(a) = \{s, t_{s_a}, t_{s_a}, c_a\}$ **do**
      $mid_{hit} = t_s + \frac{t_e - t_s}{2}$
      $mid_{anti} = ta_s + \frac{t_{se} - t_{s_a}}{2}$
      if $|mid_{hit} - mid_{anti}| > \epsilon$ **then**
        Continue
      end if
      if $c_a > c$ **then**
        Remove o(q) from result set
      end if
    end for
  end for
end if

Note that the same algorithm is used for both 1-best and lattice ASR output.

5.3.3. Anti-Query Optimization

The baseline approach to anti-query verification can also remove true positives from the result set. In the following, we aim at (i) explicitly removing anti-queries from the anti-query verification set that are likely to remove true positives from the STD result set and (ii) extend queries such that they cover a larger amount of phonetic context. This is particularly important for short queries.

For the first goal, we remove those anti-queries that are a substring of the actual query, and which are hence much easier to match. For example consider the query *Wirtschaftskrise*, and its anti query *Krise*. We can decide whether the anti-query is a substring on the word level, but also on the syllable or phoneme level. An example for the subword case would be the anti-query *Haus* for the English query *White House*.

For the second goal, we consider the short query *Wald - forest*, and the anti-query *bald - soon*. If *bald* is part of the syllable lattice but *Wald* is not, then the verification will most likely remove the STD hypothesis (because the correct *Wald* could only be matched with additional approximation). However, we could extend each anti-query with context as follows: assume the subword anti-query a was collected from the training text. Then, for each occurrence of a we obtain all left subword contexts c_l and all right subword contexts c_r from the training text. We only want to include anti-queries if the corresponding match is a strong indicator for a false alarm, and hence require both sides to be present at the same time. We construct a new contextual anti-query $c_l a c_r$ and add it to the set of anti-queries. The original anti-queries without context which have caused the recall decrease are removed from the verification set.

5.4. Verification Queries from Web Resources

In some cases knowledge about the verification queries time must be updated continuously, if not immediately before the actual query is issued by the user. For example, in the media monitoring scenario, a new company might be founded, and the company name did not exist when the baseline contextual verification set was built. Another example is context variation, where a word might be used in a different textual context, possibly with a different meaning (*an apple a day keeps the doctor away*, *my new apple iPhone*). As a remedy, we use the following process to in order to cope with the verification variability.

- Let K be the set of queries which the system can verify. Results for queries $q \notin K$ are assumed to be correct, and are presented to the user.
- Let C be the set of contextual queries and A be the set of anti-queries, which are both empty at system start.
- The system then continuously crawls a large set of relevant textual news feeds from the Internet. The feeds must cover the topics of TV programs that will be monitored, such as politics, sports and culture.

- From each new text t that is crawled, all major keywords are extracted, e.g., with the keyword extraction algorithm which we have proposed in [105].
- All new keywords from t are added to the set of queries K.
- Then, the system obtains verification queries for all queries $q \in K$ from the new text t. For contextual verification, contexts of true positive occurrences of q in t are collected. For anti-query verification, an approximate search for each q is carried out on t, and all false positive occurrences are collected as anti-query candidates. All contextual queries and anti-queries that were collected from t are added to C and A, respectively.

This process continuously updates the verification set with external knowledge from the Internet, and thereby ensures that the query verification set is as complete and up-to-date as possible. It is motivated by our work on continuous language model adaptation using web resources, where we have successfully exploited news feeds for continuous language model adaptation in German word and subword ASR. An experimental evaluation of this work can be found in [36].

Obtaining the verification queries using web resources is completely decoupled from the STD verification step, which just uses the most recent verification set produced by the described process. Hence, the continuous update does not affect the runtime of the actual query verification.

5.5. Experiments

In the following, we evaluate our proposed approaches to STD result verification. We refrain from verifying matches with a confidence of 1.0 due to the inherent high precision of these matches.

We focus on the following aspects throughout the evaluation:

1. We are particularly interested in the performance gain on *short* queries, i.e., those queries that were not in the focus of the error compensation evaluation above. We restrict the detailed evaluation below to queries with less than 10 phonemes, where we expect the largest impact when applying verification. Results on the complete query set are presented at the end of this section.
2. The error compensation approaches have been evaluated with varying levels of confidence. Hence we could report MTWV as the configuration that yields the

highest ATWV, and include ROC curves where appropriate. In the context of verification, we are rather interested in increasing precision for high-recall configurations. Hence, for the evaluation below, we choose a fixed error compensation setting at a low level of confidence, and report the ATWV for this particular setting.

3. We will apply verification on both approximate 1-best and hybrid approximate lattice search, both using low confidence thresholds as motivated above. This will enable us to study the effect of verification on two levels of compensation. We expect that the more intense compensation by hybrid approximate lattice search will benefit most from the verification.

5.5.1. Contextual Verification

We start by investigating the effect of the contextual verification approach and measure the performance on STD results with a low fuzzy threshold, i.e., with high recall and low precision. We aim at removing false positives, if possible with no change in recall.

As a baseline experiment, we obtain all possible left and right contexts for each query from the syllabified version of the DPA language model training text corpus. We start with the smallest possible context constraint of one syllable, and verify each of the STD results from approximate 1-best search with each of the available contextual expansions of the query. Each result that cannot be verified is removed from the result set. The bresults in table 5.1 show that despite the large number of running words in the DPA corpus, about 1% of the correct hits in the STD result set cannot be verified using the obtained contexts and are removed. The baseline language model training corpus was collected between 2000 and 2006, and some of the queries in the DiSCo query set are not well or not at all covered by the data. As defined above, the corpus contains *insufficient, inappropriate or even no contexts* for these queries.

In order to reduce the amount of insufficient and inappropriate contexts, we extend the corpus with additional text data collected from the German weekly newspaper *Die ZEIT*. The additional corpus contains 18 million running words, summing up to about 180 million running words. Table 5.1 illustrates the characteristics of the different corpora. Using only the DPA corpus, 1% of the correct STD hits cannot be verified due to missing contexts from the corpus, despite the fact that on average, 363 contextual verfications are available per query. Looking at the verifications from the ZEIT corpus, we observe that despite its small size, we observe more required contexts in the data. As motivated above, this is due to the fact that the ZEIT corpus was collected in a similar time span

than the DiSCo corpus, and less inappropriate and insufficient contexts are observed. There is an additional notable gain in context coverage by merging the two corpora, as *outdated* contexts - such as *Senator Obama* - can still be used in a retrospective manner. On the other hand, by adding more and more contexts to the contextual verification, less and less false positives will be removed. This effect is increased as the contexts are approximately matched with the ASR result. Hence, we observe the smallest precision gain by verification when using the largest corpus.

Table 5.1.: Influence of different external knowledge sources on contextual verification of approximate 1-best syllable STD results.

Corpus	**Falsely removed true positives (%)**	**Correctly removed false positives (%)**
DPA	1.0	13.0
ZEIT	0.4	14.4
DPA+ZEIT	**0.2**	9.6

We will use the complete DPA+ZEIT corpus in the remainder of this chapter, as it preserves almost all true positive hits from the original STD result. Here, only 0.2% of the true positive hits are falsely removed due to missing contextual verification.

In order to limit the amount of contextual expansions we apply the contextual expansion threshold, and remove infrequent contexts from the expansion set. Figure 5.4 illustrates the behavior of the verification while varying the expansion threshold on the baseline result shown in table 5.1. More and more expansions are removed from the verification set as the context expansion threshold is increased, resulting in higher precision as more and more false positives cannot be verified anymore. At the same time, an increasing amount of true positives is also removed from the result set, although the ROC graph shows that recall decrease is slow compared to precision increase. Hence, the contextual expansion threshold is an effective means for configuring a system towards higher precision and thus tailoring the search configuration towards a specific user need.

Furthermore, we investigate the effect of different amounts of context by increasing the context width. Extending the query only with a single context syllable on one side of the query already increases overall precision while preserving the high recall from the baseline as shown in table 5.1. More possible contexts become available for verification when adding expansions with a context length of two syllables to the verification set. On average, we observe over four times more bigram contexts per query compared to

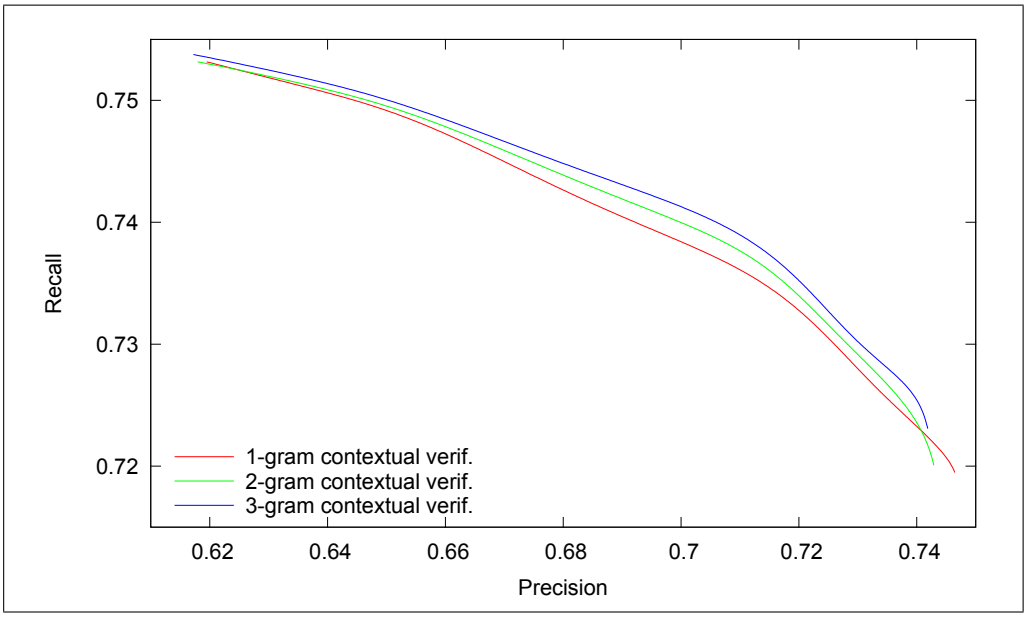

Figure 5.4.: Varying the amount of context for contextual verification of approximate 1-best syllable STD results.

the unigram expansion. Figure 5.4 also shows the results for bigram verification while varying the expansion threshold. For all thresholds, precision is increased at equal recall compared to unigram-only verification. Further extending the context width to three syllables results in additional gain, however many typical contexts are already well predicted by the syllable bigrams. Moreover, the number of queries that need to be verified is almost twice as high compared to the bigram context verification.

Table 5.2 summarizes the results for contextual verification of the 1-best results. We observe that contextual verification with 3-gram syllable contextual expansions and applied expansion threshold increases the precision by 7% absolute at equal recall compared to the unverified baseline.

Table 5.2.: Contextual verification of approximate 1-best syllable STD results. At most three syllables of left or right context used for verification.

Approach	**Precision**	**Recall**	**ATWV**
Unverified	0.60	0.75	0.48
With contextual verification	0.67	0.75	**0.51**

In the next experiment, we apply the contextual verification on the best approximate

lattice search result obtained in section 4.5. Again, we use the contextual query set built from the merged DPA+ZEIT corpus. From table 5.3 we see that using the same configuration as above, we obtain a precision gain of 6% absolute at equal recall compared to the unverified approximate lattice search baseline. Hence, the gain through verification is twice as high as in the case of approximate 1-best STD, which is caused by the inherently less precise result set of approximate lattice search.

Table 5.3.: Contextual verification of approximate syllable lattice STD results. At most three syllables of left or right context used for verification.

Approach	**Precision**	**Recall**	**ATWV**
Unverified	0.38	0.81	0.35
With contextual verification	0.44	0.81	**0.41**

5.5.2. Anti-Query Verification

First we collect the anti-query set using the algorithm given in section 5.3.1. As motivated, we collect the anti-queries from the syllabified version of the language model training corpus.

For the 152 queries with less than 10 phonemes, we obtain a total of 42571 anti queries. Note that efficiency is a minor issue for the anti-query verification step, as on average, a putative hit has to be verified against only 280 anti-queries.

The baseline results for anti-query verification of approximate 1-best syllable STD results using all anti-queries are given by table 5.4. We observe that by verifying the results on short queries using the anti-query approach, we obtain a drastic precision increase of 31% absolute while recall decreases only by 3%. As a result of this, ATWV is also increased by 8% absolute.

Table 5.4.: Anti-query verification of approximate 1-best syllable STD results.

System	**Precision**	**Recall**	**ATWV**
Unverified	0.60	0.75	0.48
All anti-queries	0.91	0.72	**0.56**

Next, we look into the errors produced by anti-query verification in order to further improve the results. A verification error occurs if the verification step removes a true positive hit from the result set, thereby decreasing recall. First, we observe that some

of the true positive removals are caused by exact substring matches of the anti-query as described in section 5.3.3. We apply the proposed pruning and remove those anti-queries where the anti-query is an exact substring of the query. Table 5.5 shows the results for different configurations. Here, we focus on the decrease of true positive removal that we can achieve by the pruning, while keeping precision gain as high as possible. First we remove queries if the anti-query is an exact substring of the query on the word level. We already obtain a decrease in TP removal of 19% absolute, while precision is only slightly decreased by 1% absolute. In the next experiment, the match is based on the phonetic representation, i.e., we would not apply the anti-query *Haus* for the query *White House*. We can further decrease the TP removal by 5% absolute without notable loss in precision. Further TP removals can be prevented by removing anti-queries also in the case of a reverse match, i.e., where the phonetic representation of the query is a subsequence of the phonetic representation of the anti-query. The resulting TP removal is 27% absolute lower than the unpruned baseline, while STD precision remains at 90% such that only few false positives are not removed anymore by the anti-query approach. Examples include the removal of the anti-query *Mark* for the query *Markt* caused by word level substring match or the pruning of the anti-query *Wahlen* for the query *Wale*, caused by reverse phoneme level substring match. We note only few anti-queries cause already many TP removals, as the average amount of anti-queries per query is only reduced from 280 to 250 by anti-query pruning.

Table 5.5.: Anti-query match pruning of approximate 1-best syllable STD results.

System	**True positive removal (%)**	**Precision**	**Recall**
All anti-queries	100	0.91	0.72
Word match pruning	81	0.90	0.72
Phoneme match pruning	76	0.90	0.73
+ reverse match	73	0.90	0.73
Unverified	0	0.60	0.75

In the next experiment, we evaluate whether adding context to the anti-queries can further reduce the amount of recall loss while keeping precision as high as possible. With only a single syllable of context on both sides of the query, table 5.6 shows that the removal of true positive hits is reduced by another 40% absolute, with only 3% loss in precision. Next, we remove all contextual anti-queries that are detected only once in the parallel corpus. This singleton-cutoff is often used in language model training,

where very rare bi- and trigrams are removed from the language model, as they often encode only noise. When removing singleton anti-queries, only 22% of the original true positive removals persist, while precision remains high at 85%.

Using words instead of syllables as the unit for determining the anti-query syllable context further decreases the amount of true positives that are removed. Words are often longer than one or two syllables. The corresponding contextual anti-query becomes longer, and the matching constraint becomes harder compared to using only one syllable of context. Precision is also further decreased, hence the selection of the context unit should depend on the scenario, i.e., recall-oriented applications should consider using word-based context expansion.

Looking at the results after pruning, we observe that pruning can cope with the fact that rare queries that are not well covered by the language model training data. For example, consider again the football team *Hoffenheim*. The competing term *Hockenheim* occurs much more frequently in LM training data, and hence the corresponding syllable trigram *h_O_ k_@_n_ h_aI_m_* is more likely to be decoded in challenging decoding situations. When collecting anti-queries for *Hoffenheim*, phonetically similar words such as *Hockenheim* will be detected as anti-queries, causing a true positive removal in the decoding example above. Adding context to the anti-query helps: *Hockenheim* is often followed by the word *Ring*, as *Hockenheim Ring* is a well-known race course in Germany. However, it is highly unlikely that (i) Hoffenheim gets decoded incorrectly by *h_O_ k_@_n_ h_aI_m_* (ii) and at the same time, the decoder outputs *Ring - r_I_N_* after Hoffenheim was spoken.

Only very few true positive hits are still removed after applying the most intense pruning. These include short words that phonetically very close, and are also used within exactly the same local context, such as the true positive *Irak - Q_i:_ r_a:_k_* which is still removed by the anti-query *Irak - Q_i:_ r_a:_n_*. These special cases can only be removed exploiting higher level contextual knowledge, which is beyond the scope of Spoken Term Detection.

Figure 5.5 summarizes the main results for anti-query verification of 1-best STD results. We observe that applying all proposed pruning techniques, we can drastically increase precision with a negligible loss in recall.

Next, we validate the results for 1-best verification on approximate syllable lattice search. From the results shown in table 5.7, we can observe that recall is decreased by 2% absolute, which is similar to 1-best anti-query verification (3% decrease). At the same time, precision is increased by 11% absolute. While this performance gain already

Table 5.6.: Anti-query context pruning of approximate 1-best syllable STD results.

System	True positive removal (%)	Precision	Recall
Reverse phoneme match pruning	73	0.90	0.73
+ syllable context	33	0.87	0.74
+ no singletons	22	0.85	0.75
+ word context	19	0.86	0.75
+ no singletons	13	**0.83**	**0.75**
Unverified	0	0.60	0.75

provides a more usable system configuration for approximate lattice search, we note that the precision increase using anti-query verification on approximate 1-best search is almost three times higher. Anti-queries are designed for overcoming systematic errors from the minimum edit distance alignment, and additional false positive errors that stem from the lattice search itself are not taken care of explicitly.

As in the case of 1-best verification, pruning based on the phoneme match approach recovers about one quarter of the true positives that were removed by the unrestricted anti-query verification, while precision remains unchanged at 49%.

When extending the anti-queries with additional context, we observed that a single context syllable at each side of the query is not sufficient for anti-query pruning. In this case, pruning does not recover much additional recall, as the slightly extended queries are still too easy to find on the lattice. When extending the query with whole single words on the left and right side, we can recover almost all original true positives, while precision is still increased by 6% absolute over the unverified baseline.

Table 5.7.: Anti-query verification of approximate syllable lattice STD results.

System	True positive removal (%)	Precision	Recall
Unverified	0	0.38	0.81
All anti-queries	100	0.49	0.79
Reverse phoneme match pruning	77	0.49	0.79
Word context pruning	6	**0.44**	**0.81**

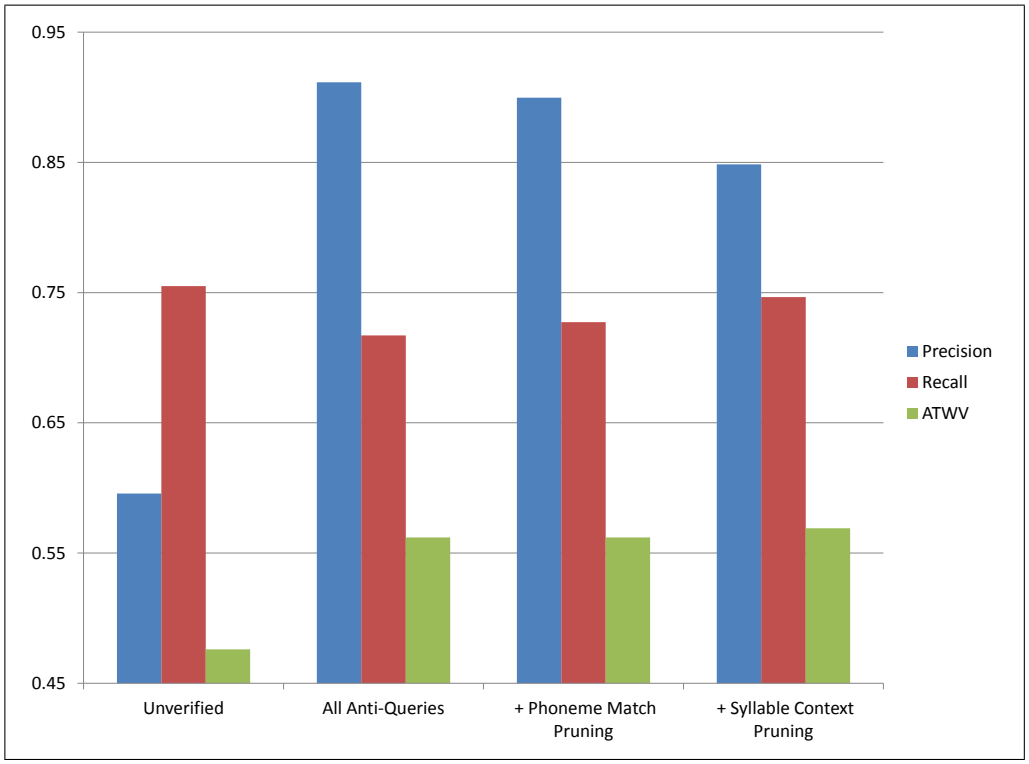

Figure 5.5.: Anti-query verification of approximate 1-best syllable STD results.

5.5.3. Hybrid Verification

In this section, we combine the two different variants for verification and evaluate whether precision can be further increased by applying the methods in sequence, i.e., we remove results using contextual verification, and then remove putative hits from the resulting reduced set using anti-query verification. Note that the order is irrelevant. Again, we verify the results when using a low approximation threshold during approximate search, which enables high recall values of the unverified baseline. We use the best configurations derived in the previous sections.

First, we look at the results obtained from approximate 1-best syllable retrieval. Combining the two approaches only yields little additional precision at equal recall over the anti-query approach. This indicates that most of the precision loss observed in the unverified baseline stems from systematic errors caused by the approximate search.

The situation is different for lattice-based approximate retrieval. In contrast to the 1-best baseline, additional gain is possible through the combination of the two approaches. Here, we obtain a precision gain of 3% absolute over each individual baseline, and ATWV is increased by 4% absolute. We conclude that verification does not only compensate for

Table 5.8.: Hybrid verification of approximate 1-best syllable STD results.

Approach	Precision	Recall	ATWV
Unverified	0.60	0.75	0.48
Contextual	0.67	0.75	0.51
Anti-query	0.85	0.75	0.57
Hybrid	**0.86**	0.75	**0.57**

systematic errors from approximate search, but also for errors caused by lattice retrieval. All in all, we obtain an absolute precision increase of 9% over the unverified baseline, while recall remains unchanged and high at 81%.

Table 5.9.: Hybrid verification of approximate syllable lattice STD results.

Approach	Precision	Recall	ATWV
Unverified	0.38	0.81	0.35
Contextual	0.44	0.81	0.41
Anti-query	0.44	0.81	0.41
Hybrid	**0.47**	0.81	**0.45**

So far, we have used the contextual verification on an approximate lattice result set with very low approximation threshold. This enables high recall of 81% at the cost of still relatively low precision. Even with hybrid verification, precision is still below 50%. While this might be tolerated in recall-oriented applications, it is a prohibitive characteristic in many scenarios, especially involving end-users.

Obviously, verification helps most if the threshold for the STD confidence is low, such that a large phonetic distance between query and actually decoding output will be tolerated by the STD approach. However, we also observe gain through verification at higher levels of baseline STD confidence. Figure 5.6 compares the performance of the unverified baseline to the three verification variants described in table 5.9 (Contextual, Anti-Query and hybrid verification), while varying the approximation threshold in the first STD pass. We observe the following characteristics:

1. Both Contextual and Anti-Query verification always outperform the baseline.
2. The performance of hybrid verification always exceeds the performance of the individual approaches.
3. Contextual and Anti-Query verification show similar performance.

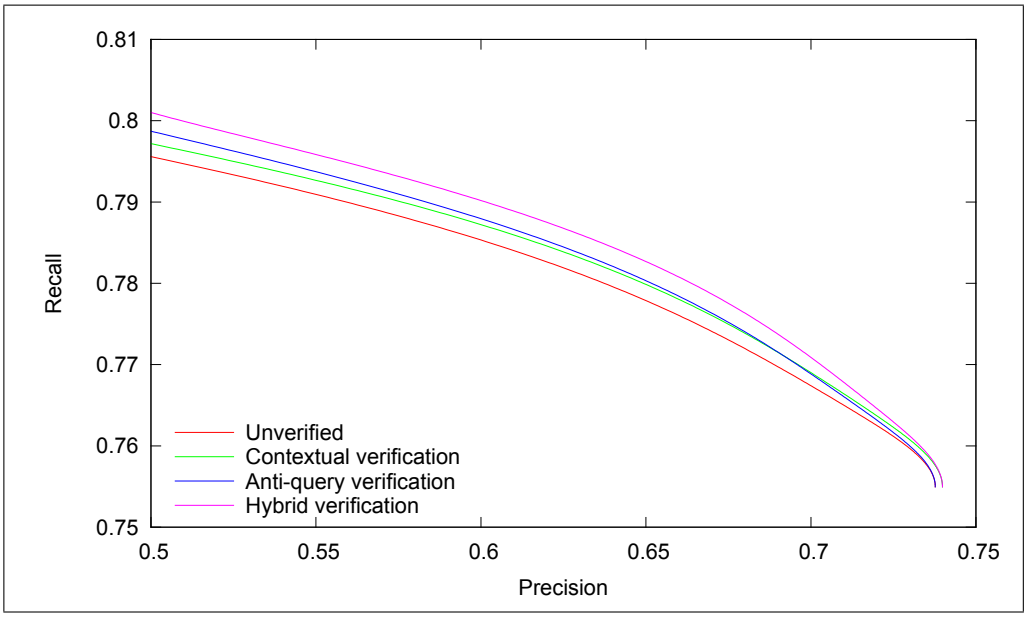

Figure 5.6.: Hybrid Verification of approximate syllable lattice STD results with varying approximate search threshold.

For comparison, we also evaluate the impact of verification on the complete set of queries, i.e., also including the second half of the query set consisting of longer queries. Table 5.10 compares the results for the hybrid approach to the unverified baseline on the complete query set. As expected, we observe only little increase in ATWV, but still obtain a precision improvement of 6% for high recall scenarios.

Table 5.10.: Hybrid verification on complete query set.

Approach	**Precision**	**Recall**	**ATWV**
Unverified	0.45	0.77	0.61
Hybrid verification	**0.51**	0.77	**0.63**

5.6. Summary

Current approaches to STD do not exploit external query knowledge that is only available at search time. In the preceding chapter, we have described a generic process for STD result verification based on this idea. In [95], we have proposed two verification methods that implement this process: contextual verification and anti-query verification. Applying both verifications in sequence on a subset of short queries improves

recall-oriented ATWV on approximate lattice search from 0.35 to 0.45, and increases precision by 9% absolute at constant high recall of 81%.

Contextual verification of a putative STD result is a *positive* verification, where we attempt to find evidence in external knowledge that the local ASR context of the putative hit is valid. If such evidence is not found, then the hit is rejected. In contrast to this, anti-query verification is a *negative* verification: here, we reject the hit if we find evidence that a phonetically similar query fits better at the putative hit position. From the evaluation of both approaches, we can draw the following conclusions:

- Contexts for contextual verification should be collected from an up-to-date parallel corpus, such that the number of inappropriate and insufficient contexts per query is minimized. Larger contexts help, but gain saturates at three syllables of context.
- For anti-query verification, competing anti-queries and corresponding contexts should be collected from the actual language model training data, which best resembles the possible output from the subword ASR.
- The anti-query verification set should contain those anti-queries that most likely cause a false alarm for a given query, and anti-query pruning should be applied to further increase the STD accuracy.
- Applying both approaches in sequence yields the best overall results, since they cover different verification aspects.
- Verification increases STD accuracy for both approximate 1-best and approximate lattice retrieval. However, due to the lower precision of the unverified baseline, impact is higher on the lattice result set.

6. Scalability Investigations

Retrieval efficiency is only rarely in the focus of STD research, and is often not reported at all. This is a little surprising, since long response times during retrieval will render an STD system unusable for many scenarios, even if it shows high performance in terms of MTWV. The important role of efficiency is also reflected by the NIST STD evaluation plan [82], which requires that "search time is to be reported" by each participating group.

In our contributions in [96] and [77], we have looked in detail at the performance characteristics of selected promising subword STD approaches, and proposed fast retrieval methods for German subword STD. Within the scope of this chapter, we will extend these results and investigate the scalability of all major aspects that have been studied within this thesis: vocabulary independent STD, error compensation in STD and STD result verification. Our main goals for this section are to describe the actual implementation of the selected approach, and study its efficiency. Where required, we propose optimizations and pruning strategies that reduce runtime while keeping accuracy at a high level.

Scalability does not only mean that a system can *scale up* to large data sets with reasonable response times. In the context of some scenarios, it might also be required to move in the opposite direction, and increase STD accuracy at the cost of retrieval efficiency. Hence, an additional goal is to get in control over the tradeoff between STD response time and STD accuracy, which is a yet unexplored trail in STD research. This will enable us to configure an STD approach for a new scenario with specific efficiency and accuracy requirements. We are particularly interested in the following aspects:

- What is the largest possible archive that can be searched with a reasonable retrieval time (e.g., response below one second)?
- What is the highest accuracy in terms of MTWV that can be achieved for medium-sized audiovisual archives of up to 1,000 hours?
- What is the highest accuracy in terms of MTWV that can be achieved for the media monitoring scenario, where about 10 hours of data need to be searched at

once?[1]

Obviously, the runtime of an individual approach heavily depends on the hardware and software platform that was used to run the experiment. For the retrieval experiments below, a standard state-of-the-art desktop PC was used to generate the runtime numbers. Table 6.1 contains the exact specification of the system. We note that exactly the same platform was used for all reported experiments.

Table 6.1.: Specification of system used for scalability experiments.

CPU type	Intel Core 2 Quad CPU Q9650
CPU clock	3.00GHz
RAM	8 GB
Operating system	SuSE Linux 11.3 64Bit, Kernel 2.6.34

Only a single CPU core was used for the experiments below. However, we note that the proposed implementations can be easily parallelized by the following simple procedure:

1. Segmenting the corpus into subcorpora of approximately equal size.
2. Perform search as described below on each sub-corpus, where each search is carried out on a different CPU core.
3. Merge the verified STD results by a simple and fast union operation.

The decrease in computation time for a single query will then be proportional to the number of used cores, i.e., the 4-core machine specified in table 6.1 will only use a quarter of the efficiency values given in the experiments below for a single core.

6.1. Scalable Vocabulary Independent Spoken Term Detection

Searching 1-best word transcripts can be solved exactly as searching textual data, which is a mature and well studied domain. Fast approaches exist and enable word search even for extremely large corpora such as the English Wikipedia, which currently contains over 2.5 billion running words[2]. From the DiSCo statistics, we can expect that the size of 1-best ASR output for very large corpora will be in the same range. If we consider a media archive of 100,000 hours of pure speech data, then we could expect more than a billion running words based on the DiSCo estimate of 10,000 words per hour of speech

[1] Cf. the example in section 2.2, where the last 15 minutes from 50 TV stations were monitored
[2] Estimated from a Wikipedia dump taken at 2011-08-12

data. However, in many scenarios, this number will be much lower, as there are many non-speech fragments (for example, DiSCo contains only about 75% speech utterances), and many interesting applications need to search far less data.

In [96], we have first described a large scale experiment for exact German syllable retrieval on a simulated corpus of 10,000 hours, where we indexed and searched 5000 copies of a small STD evaluation set consisting of two hours. In this section, we will describe our key findings, and extend this experiment as follows. First, the simulated corpus will be synthesized from a larger and more complex evaluation corpus (12 hours from DiSCo) instead of the only two hours used in [96]. This will yield more realistic word frequency distributions due to the larger sample. Moreover, other than in [96], we will obtain average runtimes on the complete DiSCo query set with its large variety of realistic queries. Finally, we will look in more detail at the response time behavior while increasing the amount of data.

We base our retrieval system on the idea of an inverted index data structure: For each occurring term, the system stores all document indices where the term occurs, thereby enabling fast retrieval without increasing the index size beyond the number of running words.

As an example, consider the following three sentences produced by the ASR:

1. *I would like to know how you feel.*

2. *You mean, how could I like this car?*

3. *Do you like this car?*

For each utterance $i \in \{1, 2, 3\}$, we collect all occurring words. For each word w_i that occurs in utterance i, we store the index i in a set with label w. In the end, this set contains all references to w across all utterances. If the utterances contain N running words and U unique terms, then this will result in U term sets with all in all N entries, hence the required storage is limited by the number of terms that occur in the text. For the example above, table 6.2 illustrates the contents of the resulting inverted index. Typically, in actually deployed systems, very frequent words (such as articles or pronouns) are not stored in the index. Removing these so-called *stop-words* from the index drastically reduces the index size, as only very few words make up most of the set of running words.

Then, retrieving the utterances that contain a single term is a simple operation with constant time, as we just have to obtain the corresponding utterance index set. The

Table 6.2.: Inverted index example.

Term	Utterance index set
car	{2,3}
could	{2}
do	{3}
feel	{1}
how	{1,2}
I	{1,2}
know	{1}
like	{1,2,3}
mean	{2}
this	{2,3}
to	{1}
would	{1}
you	{1,2,3}

inverted index concept is also used in the popular open source search engine Lucene[3], which we use as the basis for our further investigations in the following.

For exact subword STD, phrase queries play an important role. If we consider the case of syllable STD, then all queries with more than one syllable become phrase queries on the subword level, hence the indexing and retrieval system must not only be efficient for single terms, but also for multi-word queries where the order of the query terms must be correctly found.

The current implementation for phrase queries in Lucene relies on the following strategy, which is specified by algorithm 8. First, a Boolean AND-Query is carried out on the complete set of documents, such that the resulting set of documents already contains all terms. Boolean operations can be executed in a very efficient manner on inverted indices. For example, a Boolean AND over n terms is equal to intersecting the n corresponding index sets, while a Boolean OR can be easily solved by constructing the union of the n index sets. In order to further speed up the Boolean AND query, the terms are first sorted by their respective inverse frequency in the corpus, i.e., the term with the smallest number of occurrences is considered first. Its index set is intersected with the index set of the second most infrequent term. Hence, the two smallest index sets are intersected, reducing the overall cost for intersection.

Then, an exact search for the complete phrase on each document in the result set is carried out. Only documents where the phrase terms are found in the correct order are added to the final result set. We can expect that the amount of candidates for this exact

[3]http://lucene.apache.org/

matching is already drastically reduced by the AND query described above.

Algorithm 8 Collect set D of all documents that contain phrase query $s = s_1 \cdots s_n$.

```
D ← ∅
Sort s by ascending size of corresponding index set:
obtain o_1 ··· o_n, where o_i = s_j, and s_j is the term with the i − th smallest index set
for all i = 1 ··· n do
  D_i = {d|o_i occurs in document d}
  if i == 1 then
    D = D_1
  else
    D = D ∩ D_i
  end if
  if D == ∅ then
    break
  end if
end for
```

There exist other, even more efficient approaches to exact phrase search such as suffix arrays [73] or suffix trees [106], yet they typically require more complex operations in order to build and update the index. These alternatives could be considered if the index is only rarely updated, and the corpus size exceeds the maximum size that can be handled by our inverted index approach (see below).

In order to generate the full STD result as required by the NIST evaluation plan, we also store the timestamps for each 1-best ASR transcription. Then, for each document that contains the exact query phrase, we obtain the corresponding start timestamp and duration from the stored array of timestamps, such that we can asses whether a putative hit is within the tolerance boundaries of a reference occurrence.

We use the described system to index and search the 1-best output from *word ASR* and *syllable ASR*. For comparison, we also index the results which we obtain from breaking down the syllable results to phoneme sequences. We have shown earlier that this step results in slightly higher accuracy, however, we expect drastically higher search times for the phoneme-based index.

We note that text produced by ASR has a key characteristic which differs significantly from text produced by humans: the ASR decoding dictionary is fixed and relatively small. The number of unique words that occur in the transcripts can never exceed the size of our decoding dictionary, even if we would index billions and billions of hours of video. This has negative impact on the expected performance of retrieval from the inverted file, as the number of terms that can be indexed by the inverted file is also limited. As the

corpus increases, the slots for each term are more and more filled, because no new terms can be observed due to the fixed decoding lexicon. The situation is different for indexing of arbitrary texts generated by humans: here, the number of unique words is not fixed, and for large corpora much larger than in the case of word-based ASR. For example, our language model training corpus contains about 900,000 unique words, compared to about 200,000 unique words that can be produced by our word-based ASR. Hence, one could expect a more linear increase in response time with respect to the data size when indexing and searching ASR output. This has even greater impact when indexing subwords such as syllables or phonemes, where the size of the decoding dictionary is further reduced (in our case to about 10,000 syllables and 50 phonemes).

As motivated above, we generate an artificially large corpus by duplicating the original DiSCo ASR output. Note that we cannot simply index and search a large textual source. We could only evaluate 1-best approaches on such data, and it would not be realistic as it would resemble a perfect transcription. We are particularly interested in the response time on 1,000 hours of data, as many interesting media archive scenarios fall below this boundary. Table 6.3 contains some statistics for (i) the baseline DiSCo corpus consisting of 12 hours of data and (ii) a large corpus that was built by concatenating 85 copies of the ASR output on DiSCo, resembling a corpus of 1,000 hours of data. The large corpus contains about 1.5 million utterances, and over 12 million decoded words. In a similar fashion, we multiplied the syllable and phoneme outputs and created artificial syllable and phoneme transcripts for 1,000 hours of data, containing 20 million syllables and about 70 million phonemes. In the following, the large corpus will be denoted by DiSCo_{1k}.

Table 6.3.: Size of 1-best ASR results used for scalability experiments.

	DiSCo	**DiSCo_{1k}**
Hours	12	1,000
Utterances	17,152	1,455,615
ASR words	145,085	12,312,730
ASR syllables	240,927	20,446,421
ASR phonemes	822,995	69,843,989

In the following experiments, we obtain the average query response time t_{avg} over all $N = 501$ DiSCo queries. Here, query response time for a single query includes all steps which are necessary to produce the final set of STD results $o(q)$ for a query q. This

includes performing the actual search as well as collecting timestamp, hit duration and hit confidence. Time for setting up the system (e.g., index loading time) is not included.

The average time per query is then defined as follows:

$$t_{DiSCo} = \frac{1}{N} \sum_{i=1}^{N} t(q_i) \tag{6.1}$$

where $t(q_i)$ is the query response time for query q_i on the complete DiSCo corpus.

From the inverted file structure used within Lucene we can expect linear search time increase as we increase the size of the corpus by copying the original data: the number of indexed terms remains stable, but the size of the indexed documents per term is linearly increased. We note that we expect better sub-linear performance if we would increase the corpus with new unseen data. Here, more and more unseen word types are indexed, and the size of the document index per term increases slower than in our artificial experiment. However, this effect is less dramatic for syllables, and even less for phonemes, where the finite vocabulary is covered rather fast. Based on this observations, we expect a linear correlation between corpus size and query response time when artificially increasing the corpus, and lower response times for random unseen data.

The efficiency measurements given below are always averaged over a series of 10 experiments with identical setup in order to remove the influence of outliers. For a single measurement below, all 501 queries are searched 10 times in all indexed utterances in order to obtain the average value. For example, the result for the 1,000 hour syllable STD result was estimated by searching each of the $10 * 501 = 5,010$ queries in $85 * 17,152 = 1,457,920$ utterances.

Table 6.4 contains the average response times for word, syllable and phoneme-based exact STD. All experiments were carried out on both DiSCo and DiSCo_{1k}. First, we observe that retrieval is highly efficient on the baseline DiSCo corpus. With all units, we can obtain very low response times, and even phoneme retrieval can be carried out with a search time below 10ms. However, on the large corpus, word STD is almost twice as fast as syllable retrieval. Compared to the word decoding lexicon, the finite syllable is about 20 times smaller, causing the index sets per term to be larger. Moreover, most queries consist of more than one syllable, which requires an expensive exact phrase search for the syllable sequence, whereas many queries contain only one word, where a lookup in constant time is possible. However, syllable retrieval is still very fast on the large corpus, with an average retrieval time below 10ms. Even more interestingly, both word and syllable retrieval outperform phoneme search. The latter is almost 40 times slower

than syllable retrieval. This relatively slow performance on 1,000 hours of data is caused by the very small alphabet over which the inverted index is constructed: only 50 index sets for the 50 phonemes take up all 70 million running phonemes of DiSCo_{1k}. Moreover, a phoneme phrase query for a certain query word contains more query terms than the corresponding syllable phrase query.

Another cause for the different runtimes can be found in the different frequency distributions. Word frequencies in large collections are known to follow a Zipfian distribution, i.e., very few common words (such as articles or pronouns) make up most of the set of running words. More formally, Zipf's law states that frequencies are inversely proportional to the corresponding frequency rank. This has a major impact on the assumption that the least frequent part of a query term has a small index set, which in turn enables fast AND operations during Lucene's phrase search.

We carried out the following experiment in order to compare the frequency distributions of words, syllables and phonemes *on ASR output*. First, we collect the absolute frequency $c(w)$ for each of occurring word type w. Next, we sort all n word types by frequency into a list $w_1 \cdots w_n$, such w_1 has the highest frequency and w_n has the lowest frequency. Then, for the $i-th$ word w_i in the ordered list, we obtain the accumulated relative frequency $r(w_i)$ over the complete word ASR output as follows:

$$r(w_i) = \frac{\sum_{k=1}^{i} c(w_k)}{\sum_{k=1}^{n} c(w_k)} \tag{6.2}$$

We obtain the same values for the syllable and phoneme ASR outputs, respectively. Using the accumulated relative frequencies, we can observe which fraction of the running words can be covered with a given percentage of the unit vocabulary. This allows us to compare the frequency distributions across units with very different vocabulary sizes.

Figure 6.1 illustrates the different frequency distributions, using the accumulated relative frequency as a function of the number of covered unique terms ordered by frequency. We observe that similar to words, syllables follow a Zipfian distribution, and a large fraction of running syllables is already covered after accumulating a few very frequent unique syllable types. Hence, many syllables have very small index sets, and can contribute to speeding up phrase search. However, unlike words and syllables, phoneme frequencies decrease much faster if ordered by rank, hence the AND operation in the phrase query search becomes more expensive. Combined with the typically longer phrase queries for phoneme sequences and a large phoneme index, it is clear that phoneme search is outperformed by both word and syllable STD.

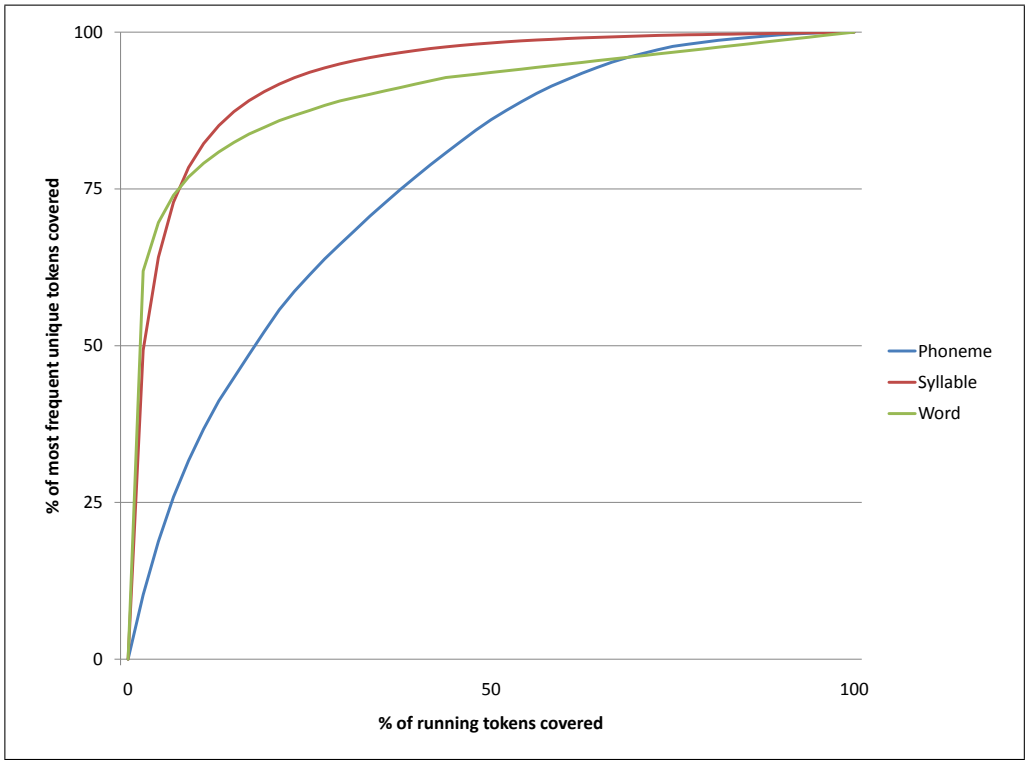

Figure 6.1.: Distribution of word, syllable and phoneme frequencies in ASR output on DiSCo.

So far, we have tested our hypothesis regarding word, syllable and phoneme distributions only on the ASR output of the relatively small DiSCo corpus. In the next experiment, we would like to compare the frequency distributions for the three units on a very large ASR output corpus. We simulate this corpus with the language model training text specified in 3.3, consisting of over 150 million words. Based on the DiSCo statistics of about 10,000 hours per word of speech, this corpus can be used as an example for perfect 1-best ASR output on about 15,000 hours of data. For the phoneme and syllable frequency distributions, we break down the text into phoneme and syllable sequences, respectively, and count the frequencies. For the word frequency distribution, we omit all words that are not in the 200,000 word lexicon of the word decoder used for generating figure 6.1 in order to simulate ASR output from the same speech recognizer. For comparison, we also calculate the word frequency distribution using the full word vocabulary (913,041 unique words).

First, we observe that the relative behavior between phoneme, syllable and word distributions remains unchanged compared to the results on DiSCo: few syllables and words

already cover most of the running tokens, while the number of accumulated running phonemes increases much slower. Comparing the results to figure 6.1, we can observe that the phoneme frequency distributions are almost equal, because all phonemes have already been observed in a similar distribution in the DiSCo results. On the other hand, relative syllable and word frequencies of the most frequent tokens are even higher in the case of the DPA corpus. Here, the amount of observed unique terms is much greater than in the case of DiSCo, where only 3,793 out of 10,816 possible syllables and 14,267 out of 200,000 possible words were actually decoded by the ASR. Terms which were not decoded in DiSCo despite being part of the decoding dictionary are likely to be of low frequency in the DPA corpus. Adding such low-frequency terms to the frequency distribution increases the impact on accumulated relative frequency for the high frequent terms, hence the DPA curves for words and syllables are much steeper than the DiSCo curves. This is also confirmed by looking at the difference between using ASR decoding vocabulary and full vocabulary when calculating the DPA word frequency distribution. In the latter case, many more low-frequency words are added to the distribution, further increasing the relative impact of the highly frequent words. We can conclude that words and syllables follow a Zipfian distribution on ASR output, which can be exploited for efficient indexing and retrieval.

Table 6.4.: Response times of vocabulary independent STD.

	Response time (ms)	
Retrieval unit	**DiSCo**	**DiSCo$_{1k}$**
Word	0.1	5.2
Syllable	0.2	9.2
Phoneme	4.2	362.6

Table 6.5 shows that the storage requirements of all three approaches are moderate, and the differences between the units reflect the relation between words, syllables and phonemes as described in table 6.3. Storing the index for the output from word ASR together with the corresponding time stamps requires 264 MB of space per 1,000 hours of data. For syllables, only about 50% more storage is required per thousand hours, while the phoneme index is almost 2.5 times larger than the word index.

Next, we verify whether our assumption about the linear correlation between corpus size and response time can be observed in the actual implementation. Figure 6.3 illustrates the response time behavior of word and syllable STD while varying the amount

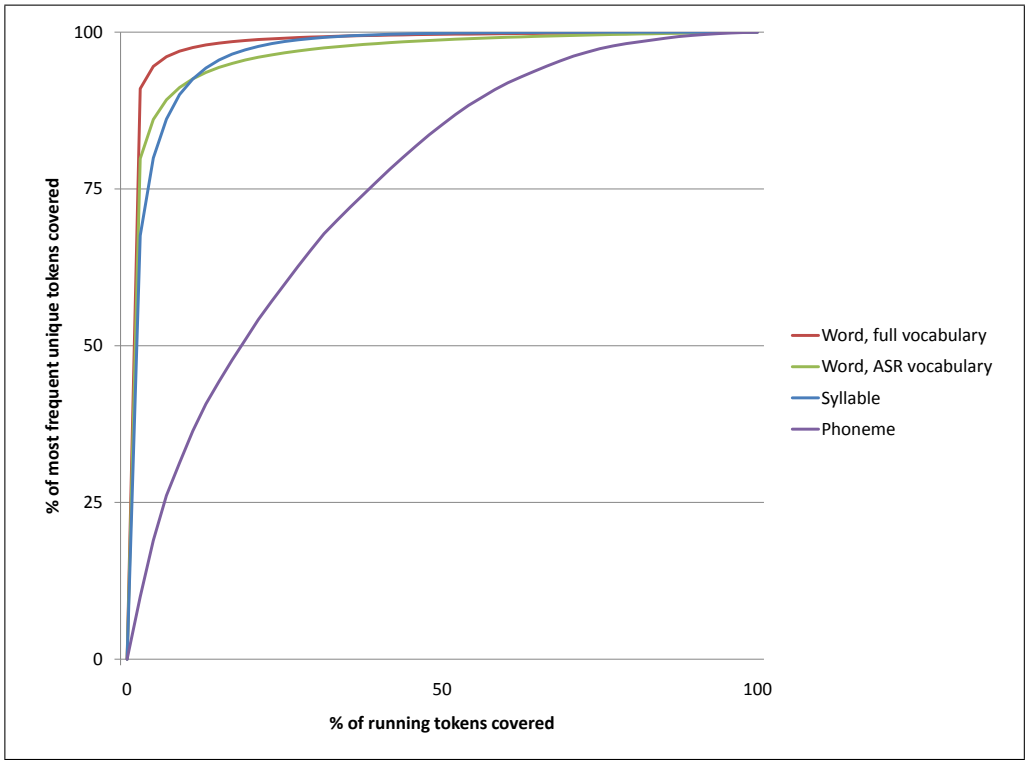

Figure 6.2.: Distribution of word, syllable and phoneme frequencies in ASR output on DPA.

of indexed data. In both cases, amount of data and response time are clearly linearly correlated (word: r=0.997, syllable: r=0.999), however the increase per 1,000 hours of data of the corresponding linear function is higher in the syllable case (word: 5 ms per 1,000 hours, syllable: 10 ms per 1,000 hours). However, syllable STD is still far more scalable than phoneme-based retrieval, where the response time is increased by 367 ms with each 1,000 hours of data.

In the experiment above, we assume that the 120,000 running words of the DiSCo corpus already represent the expected word and syllable distributions well. As mentioned before, we expect even lower query response times per 1,000 hours of data when extending the corpus with unseen data, as there will be more indexed terms in the inverted file structure, and the existing term indices will be less filled due to more vocabulary variability. Even with the upper boundary given by the linear extrapolation of the presented results, a large archive of up to 100,000 hours of data could be searched using either word or syllable-based STD, with a low response time below one second. The

Table 6.5.: Storage requirements of vocabulary independent STD, for DiSCo_{1k}.

Retrieval Unit	**Required storage (MB)**
Word	264
Syllable	378
Phoneme	645

storage requirements for the large archive scenario would still be relatively small (about 25GB for the word and about 31 GB for the syllable case, linearly extrapolated from the experiments above).

Phoneme-based STD offers only little accuracy increase over syllable-based STD, and only if the phonemes were obtained by breaking down decoded syllables instead of direct phoneme decoding. On the other hand, phoneme indices are substantially larger than syllable indices, and the retrieval time increases drastically when increasing the amount of data. Hence, from the experiments above we can conclude that phoneme-based STD should only be used if the amount of data is small and limited, and maximum STD accuracy has high priority.

6.2. Scalable Error-Tolerant Spoken Term Detection

In this section, we investigate selected aspects of the presented techniques for error compensation. We will concentrate on the scalability of syllable STD approaches. As shown in section 6.1, phoneme STD is inherently less efficient than word or syllable STD due to the large amount of tokens that need to be stored and retrieved, and already exact 1-best phoneme STD yielded relatively high response times. Compared to word STD, syllable retrieval allows for more complex error compensation approaches, and has hence been selected as representative for this section.

In the first part, we will investigate the CPU and memory requirements of the approximate search on 1-best and propose several optimizations and pruning ideas in order to increase the efficiency of the process without sacrificing too much STD accuracy. Then, we will describe the implementation of our anchor-based filter approach for fast approximate subword STD. The second part will transfer the ideas from approximate 1-best to the more complex approximate lattice search.

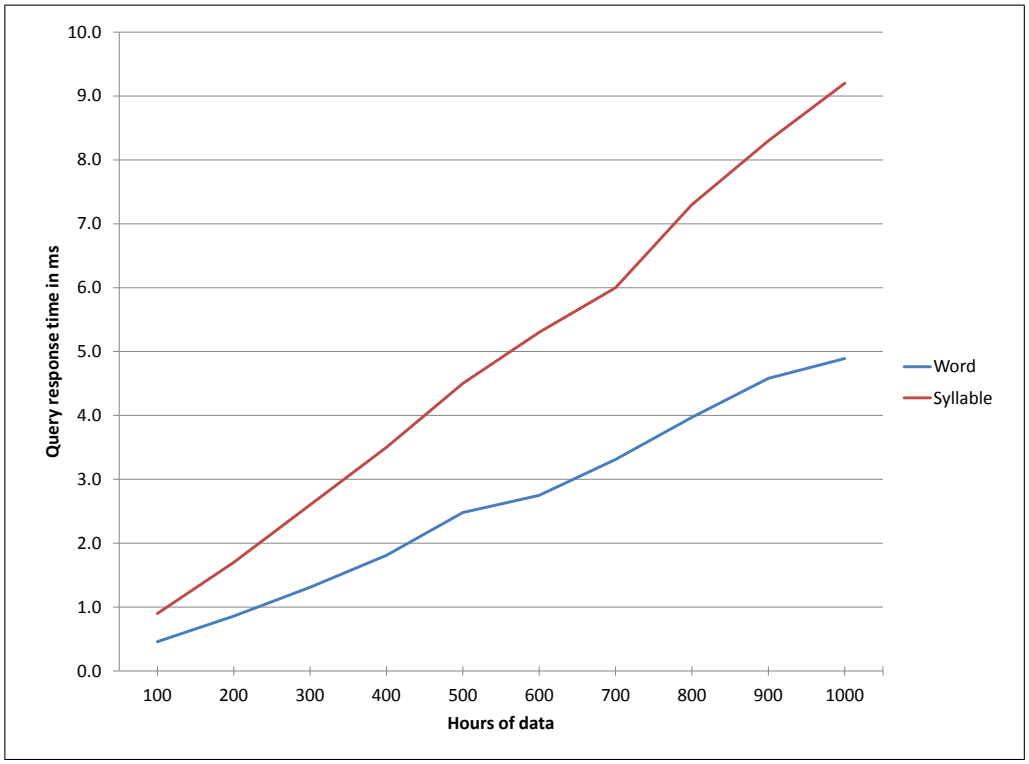

Figure 6.3.: Correlation between corpus size and query response time for vocabulary independent STD.

6.2.1. Approximate Search on 1-best Syllable Transcripts

As shown in section 4.5.2, approximate search using minimum edit distance (MED) is a powerful approach to overcome errors in STD. This is especially the case if we explicitly exploit phonetic knowledge through the use of a syllable distance metric for the syllable substitution cost, in our case again based on MED.

In principle, the time complexity for estimating the minimum edit distance between two strings of the same size is quadratic, and the time complexity for approximate syllable search for a query with m syllables on a window with n syllables taken from a 1-best transcript is in $O(n * m)$. Typically, both m and n are small, but depending on the size of the corpus, the calculation needs to be executed very often.

In a baseline experiment, we estimate the full two-stage minimum edit distance as defined in section 4.3.1 for each window position in the transcript of each utterance. A window position is assumed to be an STD hit if the local alignment between query and windowed transcription is above the confidence threshold. Table 6.6 contains the runtime per query on the DiSCo corpus (11.6 hours), averaged over all 501 DiSCo queries. On

average, the system requires almost two seconds to calculate the alignments between the query and all transcription windows for the small DiSCo corpus. The high runtime is caused by two different aspects:

1. The cost for a single MED alignment is local, but very high due to the two-stage approach.

2. The linear scan through the complete transcription yields a high lower bound for the response time.

We will address both issues in the following. First, we look more closely at the second stage of the MED distance calculation, where the phonetic distance between two syllables is estimated using the Levenshtein distance. This estimation does neither depend on the query syllable sequence nor on the windowed transcript sequence, and is thus independent of the input. Hence, we can pre-calculate the distances between all possible syllable pairs *offline*, i.e., before the retrieval system is deployed.

With about 10,000 syllables in the system, we estimate the pairwise distance for all 100 million possible syllable pairs, and store it in a 10,000 $\times$ 10,000 matrix. The syllable distance matrix is kept in main memory for fast access during alignment of the syllable sequences, such that the syllable substitution cost can be obtained in $O(1)$. This approach substantially reduces the amount of calculations that is required at query time, and thereby drastically decreases the average response time on DiSCo to about 160 ms per query. With the proposed optimization, approximate search on 1-best syllable transcripts becomes a feasible option for small-scale recall-oriented scenarios such as media monitoring. However, the linear extrapolation to 1,000 hours shown in table 6.6 indicates that this approach is still not applicable to larger corpora.

Table 6.6.: Average response time of approximate search on 1-best syllable transcripts.

	Response time (ms)	
Retrieval unit	**DiSCo**	**DiSCo$_{1k}$**
Two-stage online MED	1,846	156,658
+ syllable distance matrix	164	13,918

As indicated above, the query response time depends directly on the length of the query. In order to verify this behavior on our corpus, we estimated the individual runtimes for each occurring query length in DiSCo. For query length i, we estimated $t_{avg}(i)$ as the average runtime for all DiSCo queries of length i. Figure 6.4 illustrates

the linear increase in response time while increasing the query length. Each data point was estimated as the average over 5 independent runs on the queries of that particular length. In section 2.4.2 we already observed that only few queries have more than 20 phonemes, hence we can expect runtimes of 250 ms or less for most queries on corpora that have comparable size to DiSCo (such as in the media monitoring use case).

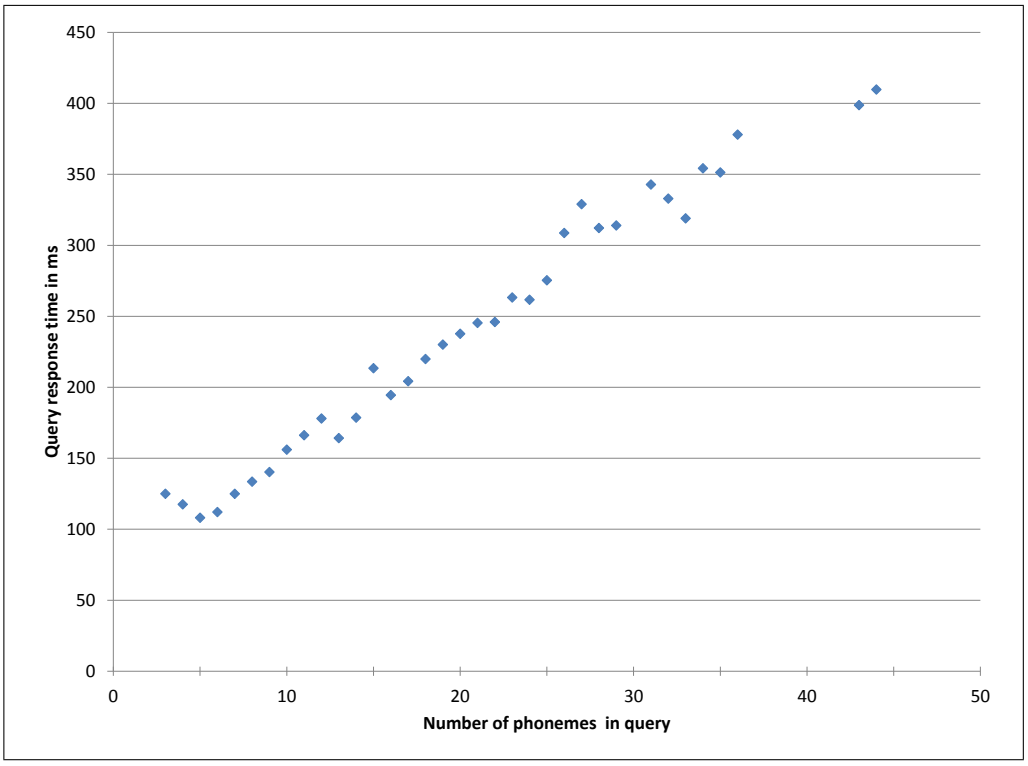

Figure 6.4.: Correlation between query length and query response time for approximate search on 1-best syllable transcripts.

The decrease in response time when using the offline syllable distance estimation comes at the cost of increased memory requirements for keeping the distance matrix in main memory. If all the pair-wise distances for all possible syllable pairs are stored as floating point values, then about 0.5GB of main memory is required for storing the matrix. This can be further reduced without negative effect on accuracy and response time by two ideas: assuming symmetric distance and quantization.

First, we assume that the pair-wise syllable distance is symmetric. We estimate the distance between syllables i and j as the average over the two pair-wise distances, and store the same value in the both the upper and lower triangle of the syllable distance

matrix. While this assumption is reasonable, it could have a negative impact on STD accuracy as it ignores evidence obtained from the parallel corpus through the (non-symmetric) phoneme confusion matrix.

If the distance is symmetric for all syllable pairs, then the syllable distance matrix will also be symmetric. Then, it is sufficient to store only the upper triangle of the matrix, thereby reducing the required amount of storage by 50%. In order to store the matrix in a flat list, we first note that the number of elements in the upper triangle (including the diagonal elements) of a $n \times n$ matrix is equal to

$$\sum_{i=1}^{n} i = \frac{(n \cdot (n+1))}{2} \tag{6.3}$$

Then, we can create a flat syllable distance list of this size, and populate it using the original symmetric distance matrix. During alignment of two syllable sequences, one needs to calculate the list index for a given syllable pair (i, j), which could have a negative effect on the search runtime.

In the next step, we reduce the amount of memory required for storing a single syllable distance. As we already quantized the distance between phonemes in the phoneme confusion matrix to three classes (equal - similar - dissimilar), it is natural to also quantize the distance between short phoneme sequences, namely syllables. Hence, instead of using 4-byte floats for the original distance $d \in [0, \ldots, 1]$, we quantize d to the range $[-127, \ldots, 127]$. Similar to the case above, this changes the actual syllable distance, which might have an impact on the STD accuracy, while runtime is not be affected here.

Table 6.7 compares the resulting storage requirements. As expected, storing only the upper triangular matrix requires only half of the original storage. Quantization from 4-byte floats to bytes further reduces the size of the data structure to about 13% of the original matrix. We found that neither MTWV nor response time were affected by the symmetry assumption or the quantization.

Table 6.7.: Storage requirements of syllable distance matrix for fast approximate search on 1-best syllable transcripts.

Approach	**Required storage (MB)**
Full float matrix	448
+ symmetry assumption	224
+ distance quantization	57

Next, we would like to overcome the linear scan through the complete syllable tran-

scription in order to further reduce the response time. Some solutions exist to fast approximate MED, e.g., using suffix arrays [48], but they suffer from the same challenges described in the exact case above (such as relatively large index size and complex update operations). In [54], the authors successfully used suffix arrays for large scale approximate phoneme retrieval on a Japanese STD task, however configurations tuned towards higher recall values still lead to relatively high response times (especially for longer queries).

For the scenarios described in section 2.2, we are rather interested in robust indexing strategies with compact indices and fast update operations, such as the inverted index data structure used in section 6.1.

We propose the following idea in order to make use of the inverted index in approximate syllable STD. The core idea is to first filter the whole set of transcripts by assuming that the query was at least partially decoded correctly, and then perform the expensive approximate matching only on the filtered set.

- We assume that at least one of the syllables in the query was correctly decoded for a particular reference occurrence. Syllables that are candidates for exact match are called *anchor syllables*. Hence, in this case, all query syllables are anchor syllables.
- The system retrieves all utterance transcripts that contain one of the query syllables, i.e., we execute a Boolean OR query over all query syllables on the complete index.
- Then, the system detects whether there is an approximate match for the whole query in each of the filtered transcripts.
- A hit is added to the final result set if the phonetic similarity between the query and a sub-sequence of the transcript is above the confidence threshold. This copes with errors that occur on non-anchor syllables.

In order to speedup the search, we exploit the assumption that at least one query syllable was correctly decoded, and can thus efficiently retrieve exact matches of this anchor syllable from an inverted index. Assuming that at least one query syllable is reasonable, since almost 70% of the syllables in the ranscript are correctly decoded according to the syllable error rate obtained in section 3.3.2. In order to understand the impact of the assumption, we looked in detail at the STD results for a fixed confidence threshold, and compared the unrestricted baseline to the proposed approach. We observed that exactly the same true positives were found despite the restriction, which indicates that our assumption is reasonable. Moreover, the proposed approach even produces over 9%

absolute less false alarms. The reduction is mainly caused by preventing false alarms for monosyllabic queries, which must be matched exactly with the proposed approach. For longer monosyllabic queries, an approximate match often leads to completely different meanings (i.e., from *S_p_r_I_t_* - *gasoline* to *S_p_r_I_C_t_* - *speaking*), which in turn can cause many false alarms.

However in principle, true positives can also be omitted when using this approach for the following reasons:

- With this approach, monosyllabic queries can only be matched exactly, although errors can occur here as well (especially for longer monosyllabic queries).
- Moreover, ambisyllabic movement can cause reduction in recall. Consider a bi-syllabic query, where the final consonant of the first syllable moves to the coda position of the second syllable (e.g., from *p_a6_t_ n_6_* to *p_a6_ t_n_6_*). Here, both syllables are corrupted by the consonant movement, and the canonical form *p_a6_t_ n_6_* cannot be found on the transcript using our anchor-based filter approach.

With the proposed approach, we can reduce the runtime by over 75% absolute at equal MTWV, yielding an average response time of 35 ms per query on DiSCo. As described above, the overall high system performance does not decrease, because all true positive hits are still found with the proposed approach. Although the approach is substantially faster than the linear baseline, it is still not applicable to medium-sized corpora such as DiSCo_{1k}, where the average response time exceeds 3 seconds per query. Next, we reduce the time required for approximate search by reducing the region for the approximate match. In the baseline above, the system searches for the position with minimum alignment cost in each complete utterance which contains at least one of the query syllables. This can be improved by the following regional approach:

- As above, the system retrieves all utterance transcripts that contain one of the query syllables.
- For each matching syllable, we store the hit environment around the matching syllable as a putative hit region.
- Then, the system aligns the query sequence with each putative hit region.
- A putative hit is added to the final result set if the similarity between the query sequence and the putative hit region is above the confidence thresholds.

Table 6.8 indicates that using the regional approach, we can further reduce the runtime by about 17% absolute at equal system accuracy. However, this approach requires that information about term positions is stored in the index, such that we can obtain the hit position for each utterance directly from the index. This requires about 35% more storage compared to the values given in table 6.7, hence it depends on the actual application requirements whether this is tolerable in order to decrease the runtime. In the following experiments, we will use the regional approach as a baseline.

Table 6.8.: Fast approximate syllable search using index filter.

		Response time per query (ms)	
Approach	**MTWV**	**on DiSCo**	**on DiSCo$_{1k}$**
All correct (exact search)	0.50	0.2	9.2
No restriction, linear scan	0.61	164.0	13,918.0
At least one correct syllable	0.61	35.1	3,036.9
+ region alignment	0.61	29.0	2,431.3

When using all query syllables as anchors, about 1700 documents survive the filtering for each query on average, which is about 10% of the original size. In the next series of experiments, we investiage whether we can remove some of the anchor syllables when filtering the initital set of lattices in order to further reduce the size of this set. Removing syllables from the anchor set will speed-up the search, as the number of OR clauses is reduced and the resulting filtered set of utterances will be smaller. However, if the only query syllable that was decoded correctly is not part of the anchor set, we will not be able to retrieve the corresponding utterance in the filter step, thus decreasing STD recall. Hence, our goal for anchor selection is twofold:

1. The system should keep those anchor syllables that are unlikely to be substituted, deleted or inserted by the ASR, and rather remove those syllables that are likely to cause ASR errors. Therefore, we need to obtain a list for all syllables sorted by *ASR instability A*. We define the ASR instability $A(s)$ for a syllable s as the number of times s was deleted, inserted, or substituted with another syllable while aligning representative ASR output to the canonical syllable reference. We can obtain $A(s)$ for each syllable from a parallel corpus such as the WDR/DW corpus described section 2.4.2.

2. Moreover, we should aim at removing frequent syllables, as they contribute most

to the set of documents found in the inverted index. The syllable frequency distribution of a given corpus can be obtained directly from the inverted index.

For the following experiment, we only keep the most infrequent and the most stable syllable in the respective anchor set and compare the results to the baseline using all query syllables as anchors. The results for this experiment are given in table 6.9. First, we remove all but the least frequent query from the query set, and obtain all utterances that contain the anchor syllable. Obviously, this drastically reduces the amount of utterance hits we obtain from the inverted index. When using all query syllables as anchors as in the experiment above, we obtain 874,914 hits from the DiSCo index for all 501 queries. When using only the least frequent syllable as an anchor, this number is reduced to only 16846 hits that need to be processed by the approximate alignment search. Naturally, the overall response time of this approach is substantially lower, and requires only 54 ms on average on a corpus of 1,000 hours compared to over three seconds for the baseline. The efficiency increase comes at the cost of accuracy decrease, as ATWV is reduced by 5% absolute over approximate baseline. However, it exceeds exact syllable search by 6% absolute. As we add more frequent syllables to the query, we approach the performance of the baseline with only three anchor syllables per query, while reducing retrieval time on 1,000 hours by 98% absolute.

If we keep the most stable syllable in terms of ASR stability, the number of hits that need to be processed is also greatly reduced (23882 hits for the most stable syllable), which indicates that syllables that tend to be correctly decoded also tend to be infrequent. This is not obvious from an ASR point of view, since infrequent syllables typically also occur infrequently in AM and LM training data, and hence are more likely to produce errors. However, our observation could be explained from a linguistic angle: It has been shown that compared to rare syllables, high-frequency syllables have a "tendency towards stronger coarticulation and greater coarticulatory variability" [9], which are a major source of ASR errors. As the anchor syllables have been selected based on their ASR stability, we can expect that an anchor is more likely to be found in the index than an anchor that was selected only based on frequency. As expected, STD performance is slightly higher in this case (0.57 vs. 0.56). Again, the STD performance reaches the accuracy of the full query set when adding three stable syllables to the anchor set.

Next, we compare the behavior of the different approaches while varying the decision boundary of the approximate search that is carried out on the filtered putative hit regions. We find that the relative behavior between the different approaches is the same as observed on the single-point MTWV metric above. If efficiency is a major requirement, a single anchor based on the ASR stability criterion yields the most promising result.

Table 6.9.: Comparing different approaches for anchor selection, using regional approximate matching on the filtered set of utterances. LFS = least frequent syllable, MSS = most stable syllable.

Anchor		**Response time per query (ms)**	
Selection	**MTWV**	**on DiSCo**	**on DiSCo$_{1k}$**
All correct (exact search)	0.50	0.2	9.2
LFS = 1	0.56	0.6	54.0
LFS = 2	0.60	2.5	217.2
LFS = 3	0.61	6.4	548.1
MSS = 1	0.57	0.8	73.5
MSS = 2	0.60	3.0	254.7
MSS = 3	0.61	6.8	582.8
All query syllables	0.61	35.1	3,036.9

With only two anchor syllables, the baseline STD accuracy can be approached with both anchor selection techniques.

We can conclude that ASR result filtering by anchor selection is a viable means for fast approximate syllable STD on medium sized corpora. When using only the least frequent query syllable as a retrieval anchor, the system can search almost 20,000 hours in the given response time requirement of one second, while MTWV is increased by 6% absolute over exact syllable search. If the most stable query syllable is used instead, MTWV is increased by another 1% absolute, still enabling search below one second on over 10,000 hours of data. With three anchor syllables, medium-sized archives of 1,000 hours can be searched yielding the same STD accuracy as the exhaustive approximative linear scan, while the response time is substantially reduced from 13 to 0.6 seconds.

6.2.2. Approximate Search on Syllable Lattices

Next, we investigate the efficiency of the most complex approach for error compensation that was presented in section 4.2: hybrid error compensation by approximate search on lattice paths. As shown in section 4.5, this technique is particularly suited for recall-oriented applications such as media monitoring, hence we aim at providing a suitable configuration for this scenario.

For implementation, we apply the same idea as in the case of approximate 1-best search above, where the system first filters promising utterances from the complete

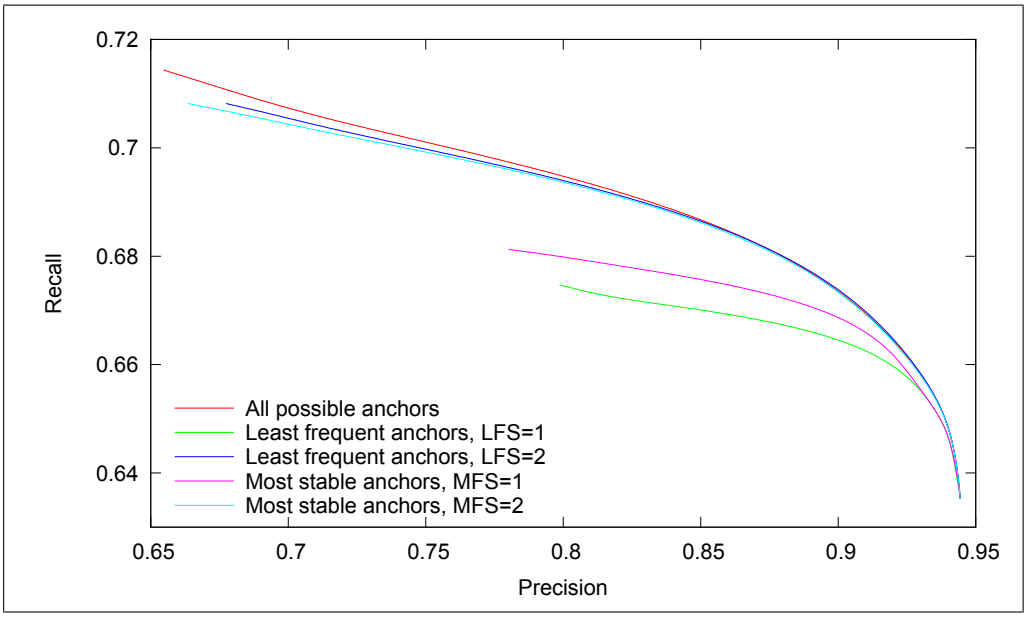

Figure 6.5.: Different approaches to anchor selection, while varying the approximate search threshold.

corpus by some criterion, and then performs the expensive approximate search only on a relatively small subset. In the following, we intuitively describe our filter approach for fast approximate lattice retrieval for a single query. The exact specification of the process is given by algorithm 9.

- First, we filter the complete set of lattices by keeping only promising candidates that contain at least one correctly decoded syllable. As a result, we obtain a set of lattices G where at least one query syllable occurs as a node label.
- For each matching lattice, we extract paths that contain one of the query syllables. These paths will be candidates for approximate alignment. We use the same idea as in the case of 1-best transcripts, hence we extract only regions that have the same size as the query sequence. Moreover, we require that the matching syllable is at about the same position as in the query sequence.
- Each path is then aligned with the query sequence, exactly as in the 1-best case above. Again, a putative hit path is added as a result, if the similarity between path and query sequence is above the decision threshold.

The actual implementation consists of three parts, namely *lattice filter*, *path extraction* and *approximate path alignment*, which are described in the following.

Algorithm 9 Filter approach for fast approximate lattice retrieval for a syllable query $s = s_1 \cdots s_n$ corresponding to a word query q.

```
G = {lattice g|g contains a node with label l ∈ {s_1, ..., s_n}}
for all g_u = (E, V) ∈ G is a lattice for utterance u do
  for all nodes r = (i, l, t_{s_r}, t_{e_r}, c) ∈ V where l = s_i ∈ {s_1, ..., s_n} do
    Obtain all node label paths p = p_1 ··· p_n through r of length n
    where p_i = s_i.
    for all paths p = p_1 ··· p_n that meet this criterion do
      if c = (1 − d(s, p)) > δ then
        Add o(q) = (u, t_{s_{p_1}}, t_{e_{p_n}}, c) to the list of results.
      end if
    end for
  end for
end for
```

Lattice filter. For each utterance in the corpus, we store the corresponding lattice such that the system can efficiently

1. decide whether one of the nodes has a certain syllable label, and
2. obtain the identity of the node, such that paths through this node can be obtained during path extraction in the next step.

Again, we are applying an inverted index in order to enable efficient retrieval of both aspects. First, all k lattice nodes of a given lattice are assigned with an ID $i \in \{1 \cdots k\}$. Then, the corresponding node labels (i.e., the corresponding syllable IDs) are ordered by node ID, yielding a sequence $s_1 \cdots s_k$ of syllable IDs, where s_i is the syllable ID corresponding to the lattice node with ID i. Then, this sequence of syllable IDs is indexed by Lucene, including the token positions and the document ID of the utterance to which the lattice corresponds. This approach results in a relatively small index of size $O(N)$ where N is the total number of nodes in all lattices.

During retrieval, the system can then retrieve all documents that contain a given syllable ID, and for each hit obtain all positions of that syllable within the document. These positions are equal to the actual node ID within the lattice corresponding to the retrieved document.

Path extraction. Next, the system needs to extract all paths through each identified node, following the path specifications given in algorithm 9. Extracting and storing all possible paths prior to retrieval can be very expensive in terms of required storage, especially if we consider the typically long subword queries. As an alternative, we propose

to keep an optimized variant of the actual lattices in memory, and retrieve the required paths on the fly during retrieval.

We use the following compact lattice definition, which still allows for extracting paths through a node with a given ID. A compact lattice is defined by an array of compact lattice nodes, which contain the following information:

1. Node start time (Float).
2. Node end time (Float).
3. ID of the corresponding syllable (Integer)
4. Set of incoming node IDs (Shorts), each pointing to an index in the array of compact lattice nodes.
5. Set of outgoing node IDs (Shorts), each pointing to an index in the array of compact lattice nodes.

For the lattices that were used for approximate lattice search in section 4.5, approximately 50,000 compact lattice nodes need to be stored per hour. Together with an average number of 1.5 incoming edges and 1.5 outgoing edges for each node, all compact lattices for 1,000 hours of data require less than 1 GB of main memory. With the ever-increasing availability of large RAM capacities, even larger corpora can be kept in memory using the given specification of compact lattices.

We require that the order in the array of compact lattice nodes is the same that was used during Lucene indexing above. Then, we can obtain the compact lattice node n_i for a node ID i retrieved from the index by a simple array lookup at position i. The actual construction of the path set for alignment is then implemented as follows:

- First, a depth-first search starting in n_i towards the final node is carried out. The system collects all syllable sequences that can be generated while traversing the lattice starting in n_i. If n_i is the $p-th$ query syllable of a query with t syllables, we restrict the search to terminate after $t-p$ steps, and obtain a set of right partial paths R where all initial nodes have incoming edges from n_i, and are at most of length $t-p$.
- Then, a reverse depth-first search starting in n_i towards the initial node is carried out, again collecting the corresponding syllable sequences that are generated. If n_i is the $p-th$ query syllable, we restrict the search to terminate after $p-1$ steps. We obtain a set of left partial paths L where the final node of each path has an edge pointing to n_i, and each path has at most length $p-1$

- Finally, all path combinations are merged into the final set of paths through n_i. For each path $l \in L$ and $r \in R$ we store the concatenation ln_ir as a path through n_i of at most length t in the final list of paths P. In the experiments below, we extend each path by at most one syllable on each side, such that the approximate match can also allow for syllable insertions.

Approximate path alignment. Finally, each path is aligned with the query syllable sequence, exactly as in the case of approximate 1-best retrieval above. Each path that has a similarity to the query above the decision threshold is added to the final result set.

However, the number of paths that are generated can be huge, and many paths will yield very low similarities during approximation, as they where only retrieved based on a single anchor syllable.

As each path will be aligned with the query syllable sequence using the relatively expensive minimum edit distance alignment, we propose an inexpensive pruning technique which removes those paths from P which are very unlikely to yield a high similarity during approximate alignment.

Our idea is that for each query syllable, a promising path should contain at least one syllable that is phonetically close. We define the *average lowest distance* $d_a(q, h)$ for measuring whether this is the case for a given query syllable sequence $q = q_1 \cdots q_n$ and a putative hit path $h = h_1 \cdots h_r$ as follows:

$$d_a(q, h) = \frac{1}{n} \sum_{i=1}^{n} \min_{j \in 1 \cdots r} \{d(q_i, h_j)\} \tag{6.4}$$

If q and a are equal, then $d_a(q, h)$ will be 0. Moreover, $d_a(q, h)$ can never reach 1, as at least the anchor syllable is found in both sequences. The system removes a path from the list if $d_a(q, h)$ is above a given threshold. Our assumption is that most of the paths will produce values for $d_a(q, h)$ above the threshold, and that the more expensive full approximate alignment is carried out only for the most promising candidates. On DiSCo, even low pruning thresholds (< 0.4) did not remove any true positives, hence we can safely use this technique to reduce the amount of paths that need to be aligned.

Using only the least frequent query syllable as an anchor, we already obtain an MTWV of 0.59, exceeding the same 1-best configuration by 2% absolute. Looking at the response time, the system delivers the result on the 1,000 hours of DiSCo_1k in 566 ms, compared to the 54 ms that were needed for approximate 1-best retrieval above. Profiling the retrieval implementation revealed that most of the time is spent on the actual approximate path alignment. Figure 6.6 illustrates that index retrieval and path extraction only require

about 15% of the total time each, while the alignment of all extracted paths with the query syllable sequence requires about 400 ms on average for each query. When using path pruning with a conservative pruning threshold of 0.5, we observe the following:

- The path pruning is carried out on each extracted path, but due to its low local cost, it requires only about 10% of the time that is required for the full approximate matching.
- The additional invest caused by path pruning pays off, as the time required by approximate matching is reduced from 395 to 276 ms, yielding an overall gain in response time of about 15% absolute over the unpruned baseline.
- As indicated above, the efficiency improvement does not affect the STD accuracy.

One might think that we could just use to the inexpensive pruning score as a confidence measure, and completely skip the expensive approximate alignment. However, this would drastically reduce STD performance in terms of MTWV (from 0.59 to 0.49). We note that the proposed path pruning could also be applied to fast approximate 1-best retrieval as described above, however, the gain will be lower as the amount of putative hit regions for a single query syllable is substantially lower than the amount of putative lattice hit paths for the same syllable on the same corpus.

Recently, several improvements of the lattice structure have been investigated in the STD community, including more compact representations such as word confusion networks [74] or position-specific posterior lattices (PSPL) for words [15] and subwords [86], which aim at further reducing storage requirements and increasing retrieval efficiency without loss in retrieval accuracy [87]. We note that our filter approach to hybrid error compensation could be further improved in terms of retrieval efficiency by exchanging the baseline lattice structure with one of these approaches. However, as described above, lattice filtering only plays a minor role in the overall retrieval costs for approximate lattice search.

Next, we look in more detail at the actual STD accuracy while using only the least frequent syllable as a query anchor. Table 6.10 shows that using two anchors substantially increases the STD accuracy, exceeding the best approximate 1-best configuration by 2% absolute. However, retrieval time increases considerably from 6.2 ms to 25.9 ms on DiSCo when using two anchor syllables, which is caused by the large amounts of paths that are added when increasing the maximal length during path extraction. Using more anchors only increases runtime, while MTWV remains stable, as all relevant paths have

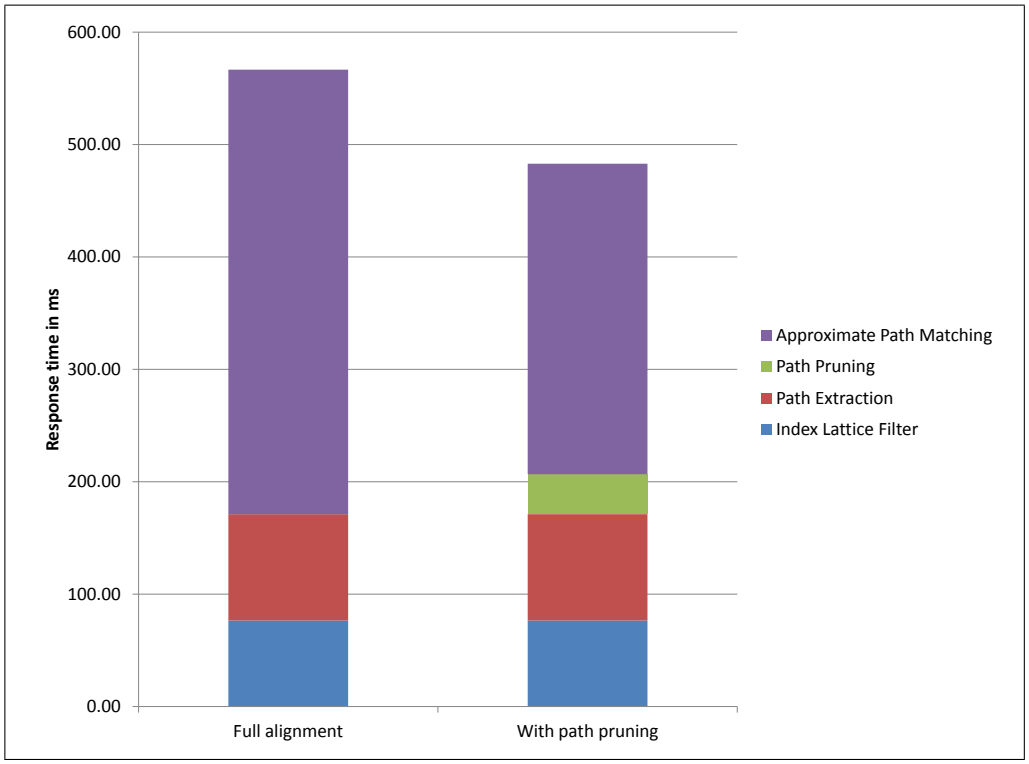

Figure 6.6.: Response time analysis of approximate lattice retrieval, with and without path pruning.

been already extracted. We note that the more expensive lattice configuration might not be suitable for very large corpora, however it is still applicable to smaller archives and delivers the best results in terms of STD accuracy for the media monitoring use case. The relatively low MTWV increase of LFS-based approximate lattice retrieval compared to LFS-based 1-best search is again caused by the low precision of the approach on short queries (see section 5).

We close the scalability investigations for error compensation in STD with the following conclusions:

- For medium-sized scenarios, anchor-based fast approximate 1-best retrieval outperforms exact search in terms of STD accuracy and full approximate alignment in terms of retrieval efficiency.

- Anchor-based fast approximate lattice retrieval enables efficient STD in recall-oriented scenarios such as media monitoring, where its accuracy exceeds the performance of the 1-best approach.

Table 6.10.: Comparing different LFS anchor sets, using path pruning on DiSCo.

Anchor selection	**MTWV**	**Response time per query (ms)**
All correct (exact search)	0.50	0.2
Lattice LFS = 1	0.59	6.2
Lattice LFS = 2	0.63	25.9
Lattice LFS = 3	0.63	64.0
Best 1-best (LFS = 2)	0.61	6.4

We note that the anchor-based two-stage retrieval idea is particularly suited for flexible known-item search in large archives, where users tolerate interaction and longer response times. A first very fast run could present the results using only the most stable syllable as an anchor. The user could then request more results if the known item is not in the result set, and new anchors would be added to the anchor set, while already searched anchors would be removed. Subsequent runs will take longer, as anchor syllables become more and more frequent.

6.3. Scalable Result Verification

Scalability is not an issue for STD result verification, neither in terms of time nor space complexity. Even if we obtain as many contexts per query as possible without any pruning and scale up to three syllables of context, only about 2,300 contextual queries need to be stored for contextual verification per query in the experimental setup used in section 5.5. On average, storing all required contextual information including contexts and relative frequencies requires only about 20 kilobyte per query. Here, we assume that syllable contexts are stored as sequences of maximum three two-byte short values, and relative frequencies are stored as four-byte floats. With this configuration, the contextual verification set for over 350,000 queries could be stored in the 8GB main memory of the standard desktop PC specified in above. With a very small expansion threshold (0.01) which yields almost equal recall at further increased precision, we could store the contexts for over 10 million queries using the same amount of memory. The storage requirements for anti-query are even lower, as only the most promising anti-queries are kept in the verification set.

When performing the actual verification, the putative hit needs to be verified against all queries in the verification set, which is achieved by searching for the verification query

in the putative hit document. In the unpruned example above, this requires 2,300 STD runs, one with each verification query. However, each run is only carried out on a single document, namely the one which contains the putative hit.

6.4. Summary

Starting from our contributions in [96] and [77], we investigated efficient implementations and scalable variants for the STD approaches presented within this thesis. We have proposed a two step filter approach, which exploits the assumption that the subword query sequence is at least partially decoded correctly. Furthermore, we have found restricting this assumption to the least frequent syllable is particularly helpful: it maximizes the reduction of the search space without sacrifycing too much STD accuracy. Based on this assumption, we could reduce the response time for approximate search on a corpus of 1,000 hours from 3 to 0.6 seconds, without any loss in STD accuracy. The same idea was then successfully applied to approximate lattice retrieval, where we assume that at least one path through the lattice contains at least one of the query syllables. Regarding the target scenarios, we can draw the following conclusions from our investigations on STD scalability, which will be discussed in more detail in the following chapter:

- Exact retrieval from inverted subword indices enables efficient vocabulary independent STD in very large media archives of up to 100,000 hours of data.
- For medium-sized scenarios of several thousand hours of video, anchor-based fast approximate 1-best retrieval outperforms exact search in terms of STD accuracy.
- Anchor-based fast approximate lattice retrieval enables efficient STD in recall-oriented scenarios such as media monitoring, where its accuracy exceeds the performance of the approximate 1-best approach.

We found that scalability is not an issue for result verification, as only relatively few putative hit documents need to be verified against the verification set.

7. Applied Spoken Term Detection: Best Practices

Applying Spoken Term Detection in real-world applications can bring up new questions, which are often neglected in the STD research community. From the experimental results above, we have learned that approximate STD on the output from subword ASR outperforms any word-based STD approach on rare OOV queries. However, depending on the application, an STD system can reach very low OOV rates: for example, in media monitoring, a continuous update of the word decoding lexicon using Internet news feeds can lead to high lexical coverage when transcribing daily news broadcasts [25], and the overall impact of OOV-STD will become smaller. Hence, an interesting question is whether augmenting word-based STD with subword-based STD could still increase STD performance - not only on OOV queries, but also on IV queries. In that case, we could recommend the hybrid use of both approaches even in low-OOV scenarios.

Another interesting question is the selection of the best search strategy for a particular scenario. While a new approach might perform well in a particular application scenario with specific precision and recall requirements, it might be inappropriate for a scenario which has tight constraints on low response times. Within this chapter, we will focus on two different aspects that need to be considered when deploying STD systems:

1. Given a word-based STD baseline, is it reasonable to augment it with syllable-based STD in an actual STD application? This question is especially interesting if we can expect a low OOV rate, either through lexical adaptation or the usage of large vocabularies for word decoding.

2. Given an STD application scenario with specific data and response time requirements, what is the best search strategy that maximizes STD accuracy within the given constraints?

While word and subword STD have been studied in isolation in the preceding chapters, it is natural to combine the two approaches into a hybrid system for real-life speech search applications that need to handle both frequent and rare queries, and where recall needs

to be maximized at tolerable precision. In section 7.1, we describe and evaluate our approach for merging results from two individual word and syllable-based German STD systems.

Then, based on our investigations in accurate and scalable STD so far, we provide best practices for the representative scenarios introduced in section 2.2, namely search in media archives and media monitoring. For each scenario, we select the most accurate approach which still delivers the result set within the given response time constraint on the expected amount of data. Again, we will consider the impact of hybrid STD by merging syllable retrieval with our LVCSR system, and observe whether the additional burden of having two parallel decoding systems pays off in terms of STD accuracy.

7.1. Hybrid Spoken Term Detection

In chapter 3.2, we have shown that both word- and subword-based retrieval have individual strengths and weaknesses. Hence, it is natural to combine the two approaches into a hybrid variant, which can be deployed in actual systems that need to handle both IV and OOV queries. We expect additional STD performance gain from combining the two result sets for two reasons:

1. Word STD typically outperforms syllable-based STD on IV queries. However, word-based STD is unable to detect OOV queries, and shows only limited search performance even if the words are broken down to subwords for retrieval. Hence, using word STD for IV terms and subword STD for OOV terms is a straightforward idea for increasing the STD performance of the overall system.

2. In addition, using both word and syllable STD for IV terms can further increase the system performance. If an IV query is not detected by the word system due to inevitable ASR errors, it might still be found by subword STD using error compensation.

Two different families of merging approaches have been studied in the literature on STD systems with comparable characteristics: *hybrid decoding* using mixed word and subword decoding units and *hybrid retrieval* from the output of two *parallel* word and subword decoders.

Hybrid decoding is an integrated hybrid retrieval approach, where the decoding vocabulary typically consists of word and subword units as in [10] or [102]. In [102], the authors evaluate a decoding system with word and phoneme units, which outputs mixed

unit transcriptions that can then be used for retrieval. While the authors observe reduced runtime and storage requirements, they also conclude that this approach leads to lower STD accuracy compared to parallel decoding and hybrid retrieval.

In [93], the authors investigated hybrid retrieval from parallel word and phoneme decoders on an English STD task. Three different types of combination were evaluated:

- *Combined search*, where both word and subword STD are carried out for each query, and the corresponding result sets are merged with a union operation into a single hybrid result set.
- *OOV search*, where subword STD search is only carried out if the query contains an OOV word.
- *No result search*, where subword STD search is only carried out if word search returns an empty result set.

The authors conclude that in all three cases hybrid retrieval outperforms the individual STD results. The three different merging methods showed similar performance. In [77], we evaluated both combined and OOV search and could confirm the findings for hybrid German syllable lattice STD. We found that the *Combined* method yields the highest overall recall at tolerable precision.

Moreover, combined search allows for evaluating the impact of hybrid retrieval on both IV and OOV queries. While the performance gain of hybrid retrieval over pure word search is not surprising on OOV queries, we are particularly interested in the question whether approximate subword search also adds new aspects to the search on IV terms. Earlier sections of this thesis have already shown that word STD outperforms subword STD on IV terms, however, approximate subword STD might perform better on different queries than word STD (such as hard-to-recognize in-vocabulary proper names).

The same evaluation is obviously not possible for OOV search, where subword STD is not even activated for IV terms. It is also hard to asses the effects for on-no-result-search.

In the following, we describe our approach for *combined* hybrid STD in detail. This method will then be evaluated in the following section, merging the results from two most promising word and subword STD runs.

Intuitively, a combined search result is obtained by (i) constructing the union of word and subword result for a query and (ii) removing duplicate results from the individual systems that correspond to the same hit.

In [93], the authors normalize the individual scores from the two individual sub systems, such that the joint result list can be reasonably ranked for retrieval above the STD

decision threshold. However, this is only possible if the two scores stem from a similar retrieval approach. In the mentioned publication, the authors use exact lattice search for both sub systems, and hence are able to use the node posteriors as the basis for the normalized score calculation.

Within the scope of this thesis, we have developed a wider range of retrieval approaches, where the resulting confidence scores are hard to compare and normalize. However, a meaningful joint ranking of word and subword results is only important if only a single decision boundary is used within the retrieval system. This is not a definite requirement in an actual deployed STD application, where we can both parts of the retrieval system can be tuned and configured individually. Hence, as an alternative to the approach in [93], we evaluate the use of individual decision boundaries per subsystem. We first obtain two optimal thresholds that yield the maximum term-weighted value on word and subword STD, respectively. Then, we merge the results, and obtain the MTWV for the joint system, which we expect to exceed the MTWV when using only a single decision boundary. Algorithm 10 gives the exact specification of combined search using individual decision boundaries as described above.

Algorithm 10 Perform combined hybrid STD for a given query q, word STD decision boundary c_w and subword STD decision boundary c_s.

$O_w(q) = \{o(q) | o(q) = \{s, t_s, t_e, c\}$ has been found by word STD with $c \geq c_w\}$
$O_s(q) = \{o(q) | o(q) = \{s, t_s, t_e, c\}$ has been found by subword STD with $c \geq c_s\}$
$O(q) = O_w(q) \cup O_w(q)$
for all $o(q) = \{s, t_s, t_e, c\} \in O(q)$ **do**
  if $\exists \hat{o}(q) = \{\hat{s}, \hat{t}_s, \hat{t}_e, \hat{c}\} \in O(q)$ at similar time where $\hat{c} \geq c$ **then**
    Remove $o(q)$ from $O(q)$
  end if
end for

When evaluating hybrid word and subword STD, we might take up two different positions. First, we can mimic and evaluate the performance of an actually deployed STD system. Here, the resulting performance will inevitably depend heavily on the configuration of the word ASR, including many factors such as the size of the lexicon, and how well it matches the vocabulary of actual decoding scenario. Even though we can relate the results to the OOV rate of the system, it is hard to predict the behavior on unseen data.

On the other hand, we can look at the performance on queries that can be handled by both word and subword STD, namely the set of IV queries. This will enable us (i) to study the difference between the best word and syllable systems on the same task and

then (ii) observe whether additional gain is available through combined search on the same query set.

Table 7.1 shows the hybrid STD results for both aspects on the DiSCo query set. For word retrieval, we use a lattice retrieval system with online pruning. For syllable STD, we apply hybrid approximate lattice STD with hybrid verification and phonetic result pruning. In both cases, the decision boundary is selected such that ATWV is maximized (hence MTWV is slightly higher than ATWV of the best recall-oriented result in chapter 5).

Table 7.1.: Hybrid Spoken Term Detection.

	Queries		
Approach	**IV**	**OOV**	**All**
Word lattice	0.71	-	0.65
Approx. syllable lattice with verification	0.65	0.59	0.64
Hybrid	0.72	0.59	0.71
+ indiv. thresholds per subsystem	**0.76**	**0.59**	**0.74**

First, we look at the performance on IV queries only. As expected, the word system outperforms syllable STD, although the difference between the most advanced systems is drastically smaller compared to the difference between exact 1-best word and syllable search. Combining the two systems with the same decision boundary only yields a small improvement of 1% absolute MTWV increase over the word baseline on IV queries, as the two systems are controlled with two different confidence metrics (lattice posteriors vs. approximate lattice path matching). However, MTWV is increased by 5% absolute on in-vocabulary queries when using individual thresholds per sub-system.

Looking in detail at the IV results, we observe the following detection capabilities.

- The word system misses 538 out of 2601 IV occurrences.
- The syllable system misses 613 out of 2601 IV occurrences.
- When merging the results of the best performing configurations, we miss only 388 out of 2601 IV occurrences.

Hence, the syllable system detects 150 additional query occurrences which were not found by the word system, even though the queries did not contain OOV terms. All but one of these 150 occurrences are nouns or noun phrases. Over 50% of the additional true positives are named entities such as rare or phonetically complex people names (*Andrea*

Ypsilanti, General Nkunda, Murad Kurnaz, Tarik al Wazir). Such named entities are often of very high interest in STD applications, and the additional recall gain through hybrid search on IV queries increases the overall value of the search system.

The system produces 773 additional false alarms on the IV queries compared to the word system. However we observe that

1. Over 50% of the additional false alarm occurrences are caused by only 10 queries. All but one consist of only one or two syllables, and only 2 out of 10 are named entities (*Irak* and *Bayern*).

2. The most frequent false alarm (*Wahlen - elections*) causes 21% of the false alarms. The query is phonetically very close to the highly frequent verb *waren - have been*.

3. Over 58% of the queries that did not cause a false alarm in word STD only produce a single false alarm in hybrid STD.

We can conclude that augmenting a word-based STD system with recall-oriented syllable STD results can further increase recall, especially for interesting named entities. This increase comes at the cost of precision loss for some search terms, especially for short non-named-entity queries, which are often not in the focus of search.

As expected, hybrid retrieval yields further improvements on the complete query set as OOVs cannot be detected by the word lattice baseline. The best configuration with individual MTWV thresholds for word and syllable STD as well as MTWV thresholds for IV and OOV queries respectively yields an absolute improvement of 9% absolute over the word-only baseline. Again, the possible gain depends directly on the word decoding lexicon and whether it matches the domain of the evaluation data. We expect even larger gains if there is a more severe mismatch between decoding lexicon and vocabulary occurring in the evaluation data.

Finally, we investigate whether restricting the hybrid approach to longer queries on the IV set might be beneficial for the overall system performance. Figure 7.1 confirms the assumption that very short queries do not benefit from hybrid STD, and that the overall MTWV can be increased if the augmentation with approximate syllable STD is restricted to longer queries. However, available recall gain through hybrid STD is sacrificed if the minimum length constraint for syllable STD becomes too tight (MTWV drops below baseline above 12 phonemes).

In summary, we can conclude that augmenting word-based STD with full-fledged approximate syllable STD yields large improvements in terms of MTWV. Table 7.2

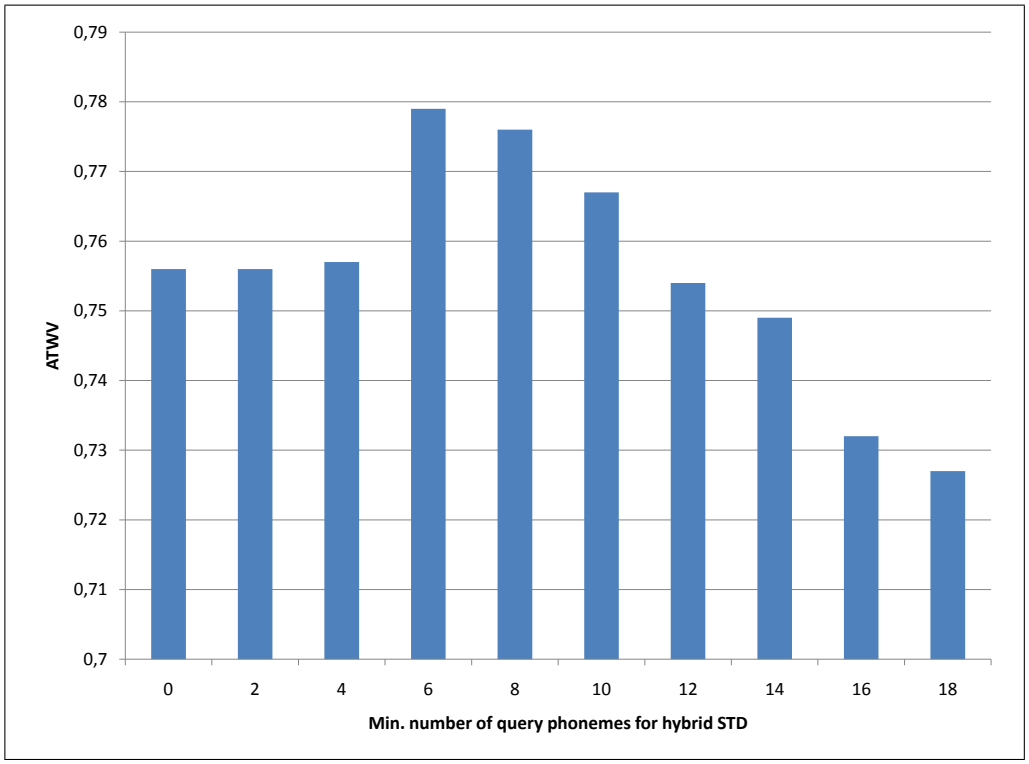

Figure 7.1.: Varying the minimal number of required query phonemes before word STD is augmented with syllable STD, IV queries only.

indicates the possible performance gains for IV queries, and also illustrates the superior performance on the complete query set for a particular word ASR lexicon configuration, where the hybrid exceeds the MTWV baseline by 11% absolute.

Table 7.2.: Hybrid Spoken Term Detection, augmentation with syllable STD only for queries with at least six phonemes.

	Queries		
Approach	**IV**	**OOV**	**All**
Word lattice	0.71	-	0.65
Best hybrid	0.78	0.59	0.76

7.2. Search Strategies for Selected Scenarios

In the following, we summarize our findings from the preceding chapters, and match the results to the requirements for the representative scenarios presented in section 2.2, namely large and medium media archive search as well as media monitoring.

The summary follows the workflow of a holistic STD system as depicted at the beginning of this thesis (see figure 1.1 for a schematic overview). First, automatic speech recognition is performed to extract a textual representation of the spoken words found in the acoustic signal. The best ASR configuration already depends on the scenario:

- If there is no restriction regarding storage size and retrieval efficiency, both word and subword output should be stored, since this increases accuracy on both IV and OOV terms.
- If efficiency matters, than 1-best syllable transcripts should be stored instead of lattices. This will reduce accuracy, but enable error-tolerant vocabulary independent STD on media archives of medium size.
- If resources are limited (and especially if the set of search terms is finite and known a priori), then a well-adapted word-based LVCSR system should be used as the baseline. This system can then be used for approximate subword retrieval from syllabified word output.

From our findings, we can derive the following best practices for the retrieval step:

- Approximate search on the ASR output will increase the STD accuracy in all configurations (e.g., on syllable 1-best and lattice or on syllabified word output).
- A linear approximate scan through the archive is prohibitive in most use cases, hence one could apply the proposed anchor filtering to prune the search space with little STD accuracy loss.
- More anchors will increase accuracy at the cost of efficiency. Hence, the size of the anchor set should be chosen depending on the size of the data set.
- For recall-oriented scenarios which apply one of the approximate search approaches, STD result verification should be applied. This effectively removes false alarms with virtually no loss in STD recall. However, for large-scale exact search on 1-best, verification is not needed since baseline precision is already high.
- The benefits of contextual and anti-query verification add up, and they should always be applied in sequence.

For each scenario, we select the strategy that yields the highest STD performance in terms of MTWV while still fulfilling the efficiency requirements of the given use case. Table 7.3 summarizes the best strategies for subword-based STD. As found in section 3.3.2, syllables represent the most powerful recognition unit for rare queries which are likely to be OOV with respect to standard word dictionaries. Moreover, syllables are ideally suited for retrieval. Similar to phonemes, syllables allow for approximate matching and effective retrieval of rare query words, however the size of the syllable dictionary allows for much more efficient storage and retrieval using inverted indices.

For very large archives of up to 100,000 hours, we have shown that exact retrieval on the 1-best output can already yield relatively high STD performance. However, MTWV on rare query tends to be low due to inevitable subword recognition errors on such words, which are caused by a lack of acoustic and language training data for infrequent syllable sequences.

For small and medium-sized media archives, approximate subword search can be integrated in order to overcome this drawback. Using a combination of fast 1-best filter through anchor selection by most stable syllables (proposed in chapter 6), followed by a hybrid verification with anti-queries and contextual verification (proposed in chapter 5), we can enable precise approximate search on 1-best transcripts for archives of up to 10,000 hours of speech data. Verification at query time enables further control over the tradeoff between recall and precision without influence on search time.

Next, we consider media monitoring, a recall-oriented scenario with relatively small amounts of data that need to be searched. Here, we can further increase recall at equal precision by applying the full-fledged approximate lattice search. As in the case of 1-best above, we approach the scalability challenge using our proposed lattice filter through anchor syllable selection. As efficiency requirements are less strict, we increase the anchor set until the MTWV gain saturates (at LFS=4). Together with our proposed hybrid verification scheme, this enables search on 100 hours of data within the given response time constraints. We note that in the case of media monitoring, response time is less critical: here, searching for a list of specified queries of interest is carried out automatically, and search response time does not add up notably to the latency of the tolerated speech recognition system.

Considering the output from word-based ASR, we found that first breaking down the word transcripts to syllable sequences and then retrieving from the subword output outperforms the 1-best word baseline. This is true both for exact and approximate subword search, while the approximate variant further increases STD performance as in the case of STD on syllable ASR output. As above, retrieving from phonemes is

Table 7.3.: Summary: Efficiency of best strategies for each scenario (syllable ASR).

Scenario	**Best strategy**	$max_h < 1s$
Large media archives	Exact 1-best search	100,000
Medium media archives	Approximate 1-best search with verification and anchor selection (MSS = 1)	10,000
Media monitoring	Approximate lattice search with verification and anchor selection (LFS = 4)	100

prohibitive, hence we consider *words broken down to syllables* to represent the best retrieval unit for word ASR output in terms of STD accuracy and retrieval efficiency. For the large scale media archive case, we again consider exact retrieval from the syllable output obtained from word ASR, which of course has the same efficiency characteristics than in the case of syllable ASR above (see table 7.4). The same holds for medium-sized archives of up to 10,000 hours of data, where we apply our proposed approximate 1-best search using anchor selection on the syllable level in the same manner as above, again using hybrid verification on the result set. However, in the case of media monitoring, we refrain from syllabifying the word lattice, as the overall gain through word lattices was relatively low compared to the approximate 1-best approach (see chapter 4). Hence, we simply allow for more anchor syllables on 1-best (LFS =4), yielding the most accurate STD results on word ASR output on up to 1,000 hours of data.

Table 7.4.: Summary: Efficiency of best strategies for each scenario (word ASR).

Scenario	**Best strategy**	$max_h < 1s$
Large media Archives	Exact 1-best search on word-to-syllable	100,000
Medium media Archives	Approximate 1-best search with verification and anchor selection (MSS = 1) on word-to-syllable	10,000
Media monitoring	Approximate 1-best search with verification and anchor selection (LFS = 4) on word-to-syllable	1000

Finally, table 7.5 summarizes the STD accuracy for each scenario using the selected STD strategies on IV, OOV and all queries. From the results obtained in the considered scenarios, we can draw the following conclusions:

- The search strategy should be selected depending on the amount of data that can

be expected in a given scenario, as more complex search approaches always yield more accurate STD results.

- All proposed methods outperform the STD accuracy of the word LVCSR baseline (MTWV = 0.62 on the complete query set).
- On IV queries, retrieval from syllabified word ASR output is more accurate than direct retrieval from syllable ASR output.
- On OOV queries, direct retrieval from syllable ASR output is more accurate than retrieval from syllabified word ASR output.
- Combining the output from word and syllable ASR into a hybrid system outperforms the individual systems in all considered scenarios on the complete query set. Note that the response time will double if the two subsystems for word and subword STD are called in sequence. However, if efficiency really matters, the processes could as well be carried out in parallel.

Table 7.5.: Summary: Accuracy of best strategies for each scenario on IV, OOV and all queries.

		MTWV		
Query set	**Scenario**	**Syllable ASR**	**Word ASR**	**Hybrid**
Large media archives	IV	0.52	0.68	0.71
	OOV	0.24	0.15	0.31
	All	0.50	0.64	0.68
Medium media archives	IV	0.60	0.72	0.74
	OOV	0.36	0.28	0.48
	All	0.58	0.68	0.72
Media monitoring	IV	0.63	0.73	0.75
	OOV	0.56	0.43	0.66
	All	0.62	0.71	0.74

We can conclude that applying our proposed methods substantially increases STD accuracy in all considered scenarios. With the best hybrid configuration for media monitoring, we obtain a recall of 0.82 on the complete DiSCo query set, which exceeds the performance of the LVCSR baseline by over 10% absolute despite the low OOV rate of the word decoder.

8. Conclusions

In the following, we will summarize the main conclusions of our work on holistic vocabulary independent Spoken Term Detection, and point out the main contributions in relation of the thesis at hand. Then, we will highlight possible directions for further research in the field.

8.1. Contributions

Specification of system requirements for Spoken Term Detection. Spoken Term Detection is an appealing research topic with approaches to speech search beyond searching the word transcript. However, there is no single STD approach that is equally well suited for all kinds of application scenarios, which is often neglected when new results are published. Hence, we have started our investigations on STD by analyzing representative STD use cases, namely media archive search and media monitoring. A set of system requirements was collected, including STD accuracy, time and space efficiency as well as system flexibility. This enables us to select the best STD strategy for a given scenario based on the importance of a particular requirement, resulting in a more holistic, scenario-dependent view on Spoken Term Detection.

Design of an evaluation corpus for STD on heterogeneous German broadcast data. The NIST STD Evaluation in 2006 has kick-started many research activities in the area of Spoken Term Detection, with a vital and growing community of researchers. However, the official NIST evaluation data contains only English, Mandarin Chinese and Arabic data. As a remedy, we prepared DiSCo, a new German corpus comparable to the NIST evaluation set, which allows for STD evaluation on German data. Over 17,000 speech utterances, summing up to about 12 hours of pure speech were selected from German television recordings and manually transcribed. A set of 501 queries was chosen semi-automatically, and we have shown that it reflects the characteristics of queries typically chosen by human archivists. The corpus specifications and the corresponding STD evaluation plan were published in [6].

Design and implementation of a word-based STD baseline for heterogeneous German broadcast data. We have set up a state-of-the-art LVCSR decoder for heterogeneous broadcast data, which was used to generate word-level 1-best transcripts from the evaluation data. The system produced word error rates close to published results on comparable corpora. Details about our baseline system can be found in [94].

Investigation of subword units suitable for German subword indexing. Based on our investigations in [96], we have studied different setups for German Spoken Term Detection using exact search on 1-best ASR transcripts. Three differen unit types were selected, depending on the amount of phonetic context covered by a single unit: very large units (words), very small units (phonemes), and intermediate units (syllables). Syllables represent a powerful unit for STD on German data, since they are the natural phonologic segmentation of a word, and are particularly suited for segmenting words from inflecting languages such as German. A distinction was made between decoding and retrieval unit, i.e., we also investigated the impact of decoding a larger unit and breaking it down for retrieval.

For in-vocabulary queries, word-based approaches perform best. Breaking down words to syllables and further to phonemes for retrieval increases STD accuracy due to implicit handling of compounds and ambisyllabic phoneme movement. However, even with our large 200,000 in-domain word dictionary, many interesting queries are outside the decoding dictionary and cannot be retrieved by the word-based system. In some cases, OOVs could also be detected by breaking down words to subwords, but these are mostly compound words, where the query was a compound part. However, we found that about 60% of the OOVs are proper names. STD based on syllable ASR output performs best on OOV retrieval, and outperforms the poor results based on unconstrained phoneme decoding.

All exact searches are highly precise ($> 90\%$ precision), however, especially the exact subword-based approaches suffer from relatively low recall, as the complete subword sequence must be decoded correctly for an exact hit in the transcript.

Investigation of new approaches for error-tolerant subword speech retrieval. Based on our preliminary investigations in [78], we have analyzed the error spaces in subword Spoken Term Detection, and found that errors stem from two different sources, namely (i) ASR errors caused by model mismatch or decoding errors, and (ii) pronunciation variation caused by deviation between canonical query sequence and the actual acoustic realization of a query. We have exploited this finding by adapting and combining

two methods for error compensation, namely lattice search for compensation of ASR errors [77], and approximate syllable search using minimum edit distance for compensation of pronunciation variation [96].

We obtain the best overall results in terms of MTWV when merging both methods into a hybrid recall-oriented variant. On in-vocabulary queries, most ASR errors are already compensated by the lattice, and additional restriction of the approximate search using position-specific clusters (PSCs) instead of phonemes as the confusion unit further increases STD accuracy. On rare OOV queries, the phoneme-based syllable distance performs best, since it can cope better with variations that do not stem from pronunciation variation. Here, MTWV for syllable retrieval is increased by 40% absolute over the exact syllable baseline.

Exploiting external knowledge for result verification at search time. We have derived a novel paradigm for exploiting external knowledge about the query at search time. In conventional STD, external knowledge is only incorporated into the system during indexing, e.g., by means of training resources for ASR. The advantage from exploiting external knowledge at search time is twofold: first, we can exploit *query-dependent* knowledge. Second, we can exploit up-to-date knowledge resources that might not have been available or in the focus of interest during the time of indexing.

Based on the results of our contribution in [95], we have proposed two actual implementations of this paradigm, namely contextual verification and anti-query verification. Contextual verification exploits typical contexts for a given query from a parallel textual corpus. Then, we can verify whether a putative STD result is actually correct by checking whether the decoded contexts in the ASR output are valid. Anti-query verification detects systematic false alarms that are caused if a phonetically similar competitor of the query was actually spoken. Here, we exploit the external knowledge to identify those competing terms that most likely generate false alarms for a given query.

Both verification approaches increase precision without notable effect on recall by removing a substantial amount of false alarms from the putative results set. Since both ideas cover different verification aspects we obtain the best results from a hybrid combination. If applied in sequence, verification increases STD precision on approximate lattice search by 9% absolute at equal recall.

Design of scalable algorithms for error-tolerant speech search on large corpora. We have found that search approaches can differ drastically in terms of retrieval efficiency, and that small accuracy degradations through pruning can already lead to large effi-

ciency gains. Motivated by our finding in [77], we have derived a flexible efficient filter approach for fast two-stage approximate lattice and 1-best retrieval. It is based on the assumption that the canonical representation of a query was at least partly decoded correctly. We exploit this assumption by applying the expensive approximate alignment only on documents where the ASR output contains at least parts of the query. With this generic approach, we obtain a flexible means for adjusting the search behavior between search time and accuracy: adding more and more anchor syllables to the filter set will increase retrieval accuracy, but reduce efficiency at the same time.

We have found that the most stable syllables in terms of ASR decoding errors are typically also relatively infrequent. This property allows for drastic search space reduction while preserving most of the STD accuracy gain obtained by approximate search. With three anchor syllables, search time on the syllable ASR output from 1,000 hours of data is reduced from 13 to 0.6 seconds compared to the exhaustive approximate baseline.

The approach can be applied to both approximate 1-best and lattice search, and is particularly suited for scenarios requiring flexibility at search time such as known-item search in large media archives, where users tolerate interaction and longer response times (see section 6.2).

Best practices for selected STD scenarios. Based on our findings, we have selected the best STD strategies for each of the considered scenarios, which return the final STD result set with a response time below one second on the given corpus size. The selected configurations are summarized below:

- For recall-oriented applications with relatively small data sets that need to be searched (such as media monitoring), we recommend to use the full-fledged variant including all approaches discussed within this thesis. First, we apply the hybrid approximate lattice cascade, using anchor-based lattice filtering with a large anchor set such that all but the most frequent query syllables are used as path anchors for approximate alignment. All results should be verified with hybrid verification, using contextual verification and anti-query verification in sequence. Then, the syllable result set is merged for both IV and OOV queries with anchor-based approximate retrieval from syllabified word-ASR output using a relatively large least-frequent anchor set. Again, we apply hybrid verification on the putative result set. With this configuration, we can obtain an MTWV of 0.74 within the response time limit on 100 hours of data.

- For medium-scale media archive search, a hybrid approximate 1-best approach

should be used. For the syllable sub-system, we apply the anchor-based transcript filtering, using only the most stable syllable as an anchor such that the search space for approximate alignment is already substantially reduced. Again, verification is applied as above. For the word-subsystem, we suggest again to use the syllabified version, and perform the same approximate alignment as in the case of the syllable sub-system (MSS-based anchor retrieval followed by hybrid verification). With this hybrid configuration, we obtain an MTWV of 0.72 within the time constraints on up to 10,000 hours of data.

- For large-scale media archive search, we resort to exact search on the 1-best ASR output. Again, a combination of word and syllable output yields the best results given the time constraints. Words should be broken down to syllables to cope with misses due to German compounding. Result pruning through verification is not needed since the results from exact search are typically highly precise. With this setup, we obtain an MTWV of 0.68 within the time constraints on up to 100,000 hours of data.

8.2. Opportunities for Future Research

Taking up a holistic position on Spoken Term Detection has enabled us to derive STD systems for selected scenarios with very different requirements, from large scale media archive search to recall-oriented media monitoring. In the following, we sketch possible directions for further investigations.

Exploiting external knowledge at search time. Within the scope of this thesis, we have proposed the paradigm for exploiting external knowledge at search time, and have described two successful approaches to query verification which implement this idea. So far, we have only exploited parallel textual corpora for building up our query and anti-query models. Future endeavors could also build upon other external knowledge sources at search time. As an example, we could consider implicit or explicit relevance feedback, which has been recently successfully exploited for Spoken Term Detection [67]. While the authors used the manual feedback examples for re-training acoustic ASR models, we could also exploit the data for direct verification of putative STD results. For example, we could use ASR lattice paths from the positive feedback examples for a query for positive verification of STD results corresponding to the same query. In a similar fashion, we could build the anti-queries from negative feedback examples.

From cascades to integrated approach. Throughout this thesis, we have been in favor of loosely coupled components for the individual steps of our full-fledged holistic STD system. Individual approaches come to a decision, and than pass on that decision in a cascading manner to the next level. For instance, we accept the output from approximate lattice search for a given query as a definite decision from that step in the cascade, and then prune the corresponding putative result set with the verification step. While this allows for flexible configuration of STD systems depending on the actual requirements of an STD scenario, it could be interesting to investigate the possibility of an integrated approach, i.e., aiming at a single global decision which takes into account all signals provided to the system at the same time.

Cross-site, cross-language evaluation of query verification. In [80], we have collaborated with two other STD research sites (NTNU Trondheim and QUT Brisbane) with the aim of evaluating selected STD approaches on several data sets, including different languages, data challenges and evaluation metrics. Such cross-site and cross-language STD evaluations are very helpful in understanding the impact of new approaches across data sets, languages and evaluation metrics, and we suggest to perform a similar comparative study on our novel approach to STD result verification.

A. List of Evaluation Queries

For reference, this appendix contains a list of all 501 DiSCo queries used in the evaluations presented within this thesis. For each query, table A.1 gives the corresponding syllabification that was used by the subword retrieval approaches. This subword representation was obtained automatically using a data-driven approach based on the Bonn Open Synthesis System (BOSS) [13]. Note that the syllabifications have not been manually corrected, and some syllable sequences are not correct. Incorrect syllabification occur particularly for foreign proper names, such as *YouTube - j_u:_ t_u:_ b_@_*. In this case, approximation is already required to cope with the incorrect syllabification of the query.

Table A.1.: List of evaluation queries. OOV queries are marked with an asterisk (*)

Query	**Syllabification**
Abenteuer Forschung	Q_a:_ b_@_n_ t_OY_ 6:_ f_O6_ S_U_N_
Abgeordnete Homann	Q_a_p_ g_@_ Q_O6_t_ n_@_ t_@_ h_o:_ m_a_n_
Abwrackprämie (*)	Q_a_p_ v_r_a_k_ p_r_E:_m_ j_@_
Adolf Merkle	Q_a:_ d_O_l_f_ m_E6_k_ l_@_
Afghanistan	Q_a_f_ g_a:_ n_I_s_ t_a:_n_
Afro Amerikaner	Q_a_f_ r_o:_ Q_a_ m_e:_ r_i:_ k_a:_ n_6:_
Ahmadinedschad	Q_a_x_ m_a:_ d_i:_ n_@_ d_Z_a_t_
Ahmadinedschads	Q_a_x_ m_a:_ d_i:_ n_@_ d_Z_a_t_s_ Q_a_
Atombombenträumen (*)	t_o:_m_ b_O_m_ b_@_n_ t_r_OY_ m_@_n_
Aktien	Q_a_k_ t_s_j_@_n_
Aktion Mensch	Q_a_k_ t_s_j_o:_n_ m_E_n_S_
Aktuelle Sportstudio	Q_a_k_ t_u:_ E_ l_@_ S_p_O6_t_ S_t_u:_ d_i:_ o:_
Altenberg	Q_a_l_ t_@_n_ b_E6_k_
Amerikaner	Q_a_ m_e:_ r_i:_ k_a:_ n_6:_
Amerikas Wirtschaft	Q_a_ m_e:_ r_i:_ k_a:_s_ v_I6_t_ S_a_f_t_

Table A.1 – continued from previous page

Query	Syllabification
Amnesty International	Q_E_m_ n_@_s_ t_i:_ Q_I_n_ t_6:_ n_E_ S_@_ n_@_l_
Andrea Nahles	Q_a_n_ d_r_e:_ a:_ n_a:_ l_@_s_
Andreas Huppert	Q_a_n_ d_r_e:_ a_s_ h_U_ p_E6_t_
Andreas Kappler	Q_a_n_ d_r_e:_ a_s_ k_a_ p_l_6:_
Andrea Ypsilanti	Q_a_n_ d_r_e:_ a:_ Q_Y_p_ s_i:_ l_a_n_ t_i:_
Angela Merkel	Q_a_N_ g_e:_ l_a:_ m_E6_ k_@_l_
anglo amerikanischen	Q_a_N_ l_o:_ Q_a_ m_e:_ r_i:_ k_a:_ n_I_ S_@_n_
Anne Will	Q_a_ n_@_ v_I_l_
Ansprüche	Q_a_n_ S_p_r_Y_ C_@_
Arbeitnehmer	Q_a6_ b_aI_t_ n_e:_ m_6:_
Arbeitslosigkeit	Q_a6_ b_aI_t_s_ l_o:_ z_I_C_ k_aI_t_
Arbeitsplatz	Q_a6_ b_aI_t_s_ p_l_a_t_s_
Arbeitsplätze	Q_a6_ b_aI_t_s_ p_l_E_ t_s_@_
ARD Morgenmagazins	Q_a:_ Q_E6_ d_e:_ m_O6_ g_@_n_ m_a_ g_a_ t_s_i:_n_s_
Armenhaus Europas	Q_a6_ m_@_n_ h_aU_s_ Q_OY_ r_o:_ p_a:_s_
Arminia Bielefeld	Q_a6_ m_i:_ n_i:_ a:_ b_i:_ l_@_ f_E_l_t_
Arsenal London	Q_a6_ z_e:_ n_a:_l_ l_O_n_ d_O_n_
Arsene Wenger (*)	Q_a6_ s_e:_ n_@_ v_E_ N_6:_
Arthur Abraham	Q_a:_6:_ t_u:_6:_ Q_a:_ b_r_a_ h_a_m_
Attac Deutschland	Q_a_ t_a_k_ d_OY_t_S_ l_a_n_t_
Aufbau Ost	Q_aU_f_ b_aU_ Q_O_s_t_
Auflösung Guantanamos	Q_aU_f_ l_2:_ z_U_N_ g_u:_ a_n_ t_a:_ n_a_ m_o:_s_
Aufschwung Chinas	Q_aU_f_ S_v_U_N_ C_i:_ n_a:_s_
Auschwitz	Q_aU_ S_v_I_t_s_
Ausländische Marken	Q_aU_s_ l_E_n_ d_I_ S_@_ m_a6_ k_@_n_
Außenminister Colin Powell	Q_aU_ s_@_n_ m_i:_ n_I_s_ t_6:_ k_O_ l_I_n_ p_aU_ @_l_
Außenminister Hans Dietrich Genscher	Q_aU_ s_@_n_ m_i:_ n_I_s_ t_6:_ h_a_n_s_ d_i:_ t_r_I_C_ g_E_n_ S_6:_
Außenminister Steinmeier	Q_aU_ s_@_n_ m_i:_ n_I_s_ t_6:_ S_t_aI_n_ m_aI_ 6:_
Außenpolitik	Q_aU_ s_@_n_ p_o:_ l_i:_ t_I_k_

Table A.1 – continued from previous page

Query	Syllabification
Automobilindustrie	Q_aU_ t_o:_ m_o:_ b_i:_l_ Q_I_n_ d_U_s_ t_r_i:_
Baden Württemberg	b_a:_ d_@_n_ v_Y6_ t_@_m_ b_E6_k_
Bad Tölz	b_a:_t_ t_9:_l_t_s_
Bahnchef Mehdorn	b_a:_n_ S_E_f_ m_e:_ d_O6_n_
Bakterielle Erreger	b_a_k_ t_e:_6:_ j_E_ l_@_ Q_E6_ r_e:_ g_6:_
Banken	b_a_N_ k_@_n_
Barack Obama	b_a_ r_a_k_ Q_o:_ b_a:_ m_a:_
Bastian Schulz	b_a_s_ t_j_a:_n_ S_U_l_t_s_
Bayer Leverkusen	b_aI_ 6:_ l_e:_ v_6:_ k_u:_ z_@_n_
Bayern	b_aI_ 6:_n_
Bayern München	b_aI_ 6:_n_ m_Y_n_ C_@_n_
Bela Rethy (*)	b_e:_ l_a:_ r_e:_ t_i:_
Berichterstattung aus China	b_@_ r_I_C_t_ Q_E6_ S_t_a_ t_U_N_ Q_aU_s_ C_i:_ n_a:_
Berlin	b_E6_ l_i:_n_
Berlin Mitte	b_E6_ l_i:_n_ m_I_ t_@_
bester Mann auf dem Platz	b_E_s_ t_6:_ m_a_n_ Q_aU_f_ d_e:_m_ p_l_a_t_s_
Beziehungen nach Russland	b_@_ t_s_i:_ U_ N_@_n_ n_a:_x_ r_U_s_ l_a_n_t_
Biathlon Weltcup	b_i:_ Q_a_t_ l_O_n_ v_E_l_t_ k_a_p_
Big Apple	b_I_k_ Q_E_ p_@_l_
Bildungspolitik	b_I_l_ d_U_N_s_ p_o:_ l_i:_ t_I_k_
Bildungsrepublik Deutschland (*)	b_I_l_ d_U_N_s_ r_e:_ p_u:_ b_l_I_k_ d_OY_t_S_ l_a_n_t_
Bill Clinton	b_I_l_ k_l_I_n_ t_@_n_
Bischof Richard Williamson	b_I_ S_o:_f_ r_I_ C_a:_6:_t_ v_I_l_ j_@_m_ z_@_n_
Blutbad	b_l_u:_t_ b_a:_t_
Bochum	b_o:_ x_U_m_
Bochumer Kindertafel (*)	b_o:_ x_U_ m_6:_ k_I_n_ d_6:_ t_a:_ f_@_l_
Boris Jelzin	b_o:_ r_I_s_ j_E_l_ t_s_I_n_
Börse	b_9:_6:_ z_@_
Borussia Dortmund	b_o:_ r_U_s_ j_a:_ d_O6_t_ m_U_n_t_
Borussia Mönchengladbach	b_o:_ r_U_s_ j_a:_ m_9:_n_ C_@_n_ g_l_a_t_ b_a_x_
Boxen	b_O_k_ s_@_n_

Table A.1 – continued from previous page

Query	Syllabification
Boxlegende Schwergewichtschampion Nikolai Walujew (*)	b_O_k_s_ l_e:_ g_E_n_ d_@_ S_v_e:_6:_ g_@_ v_I_C_t_s_ t_S_E_m_ p_j_@_n_ n_i:_ k_o:_ l_aI_ v_a_ l_u:_ j_E_f_
Brandenburger Tor	b_r_a_n_ d_@_n_ b_U6_ g_6:_ t_o:_6:_
Bremer Trainer Thomas Schaaf	b_r_e:_ m_6:_ t_r_E:_ n_6:_ t_o:_ m_a_s_ S_a:_f_
Buenos Aires	b_u:_ E:_ n_O_s_ Q_aI_ r_@_s_
Bundesagentur	b_U_n_ d_@_s_ Q_a_ g_E_n_ t_u:_6:_
Bundesdrogenbeauftragte Sabine Bätzing	b_U_n_ d_@_s_ d_r_o:_ g_@_n_ b_@_ Q_aU_f_ t_r_a:_k_ t_@_ z_a_ b_i:_ n_@_ b_E_ t_s_I_N_
Bundesinnenminister Schäuble	b_U_n_ d_@_s_ Q_I_ n_@_n_ m_i:_ n_I_s_ t_6:_ S_OY_ b_l_@_
Bundespräsident Köhler	b_U_n_ d_@_s_ p_r_E_ z_i:_ d_E_n_t_ k_2:_ l_6:_
Bundesregierung	b_U_n_ d_@_s_ r_e:_ g_i:_ r_U_N_
Bundesrepublik Deutschland	b_U_n_ d_@_s_ r_e:_ p_u:_ b_l_I_k_ d_OY_t_S_ l_a_n_t_
Bundesumweltminister Jürgen Trittin	b_U_n_ d_@_s_ Q_U_m_ v_E_l_t_ m_i:_ n_I_s_ t_6:_ j_Y6_ g_@_n_ t_r_I_ t_i:_n_
Bürgerrechtler Jesse Jackson	b_Y6_ g_6:_ r_E_C_t_ l_6:_ d_Z_E_ s_i:_ d_Z_E_k_ s_@_n_
Burkina Faso	b_U6_ k_i:_ n_a:_ f_a:_ z_o:_
Bush	b_U_S_
Candle Light Diner	k_E_n_ d_@_l_ l_aI_t_ d_i:_ n_e:_
Cash Flow (*)	k_E_S_ f_l_O_U_
CDU	t_s_e:_ d_e:_ u:_
CDU Chefin	t_s_e:_ d_e:_ u:_ S_E_ f_I_n_
CDU CSU	t_s_e:_ d_e:_ u:_ t_s_e:_ Q_E_s_ Q_u:_
CDU Fraktion	t_s_e:_ d_e:_ u:_ f_r_a_k_ t_s_j_o:_n_
CDU Generalsekretär Pofalla	t_s_e:_ d_e:_ u:_ g_e:_ n_@_ r_a:_l_ z_e:_ k_r_e:_ t_E:_6:_ p_o:_ f_a_ l_a:_
CDU Ministerpräsident Roland Koch	t_s_e:_ d_e:_ u:_ m_i:_ n_I_s_ t_6:_ p_r_E_ z_i:_ d_E_n_t_ r_o:_ l_a_n_t_ k_O_x_
CDU Vorsitzende	t_s_e:_ d_e:_ u:_ f_o:_6:_ z_I_ t_s_@_n_ d_@_
Cem Özdemir	t_S_E_m_ Q_9:_t_s_ d_e:_ m_I6_

Table A.1 – continued from previous page

Query	Syllabification
Champions League	t_S_E_m_ p_j_@_n_s_ l_i:_k_
Chancen	S_O_ s_@_n_
Charlotte Knobloch	S_a6_ l_O_ t_@_ k_n_o:_ b_l_O_x_
Claudia Roth	k_l_aU_ d_i:_ a:_ r_o:_t_
Commerzbank	k_o:_ m_E6_t_s_ b_a_N_k_
Computer	k_O_m_ p_j_u:_ t_6:_
CSU	t_s_e:_ Q_E_s_ Q_u:_
CSU Politiker	t_s_e:_ Q_E_s_ Q_u:_ p_o:_ l_i:_ t_I_ k_6:_
CSU Positionen	t_s_e:_ Q_E_s_ Q_u:_ p_o:_ z_i:_ t_s_j_o:_ n_@_n_
Dänische Telekommunikations-unternehmen	d_E:_ n_I_ S_@_ t_e:_ l_@_ k_O_ m_U_ n_I_ k_a_ t_s_j_o:_n_s_ Q_U_n_ t_6:_ n_e:_ m_@_n_
das Weiße Haus	d_a_s_ v_aI_ s_@_ h_aU_s_
Datenbank	d_a:_ t_@_n_ b_a_N_k_
Demokratie	d_e:_ m_o:_ k_r_a_ t_i:_
deutsche Bahn	d_OY_ t_S_@_ b_a:_n_
deutsche Bank	d_OY_ t_S_@_ b_a_N_k_
deutsche Bank Chef Ackermann	d_OY_ t_S_@_ b_a_N_k_ S_E_f_ Q_a_ k_6:_ m_a_n_
deutschen Badminton Verbandes	d_OY_ t_S_@_n_ b_E:_t_ m_I_n_ t_@_n_ f_E6_ b_a_n_ d_@_s_
Deutschland	d_OY_t_S_ l_a_n_t_
DFB Pokal	d_e:_ Q_E_f_ b_e:_ p_o:_ k_a:_l_
Diabakir (*)	d_i:_ a_ b_a_ k_i:_6:_
die Ärztliche Schweigepflicht	d_i:_ Q_E:_6:_t_s_t_ l_I_ C_@_ S_v_aI_ g_@_ p_f_l_I_C_t_
Dieter Kronzucker	d_i:_ t_6:_ k_r_o:_n_ t_s_U_ k_6:_
Dietmar Hopp	d_i:_t_ m_a:_6:_ h_O_p_
Dimitri Medwedew	d_i:_ m_i:_ t_r_i:_ m_E_t_ v_e:_ d_E_f_
diplomatischen Beziehungen	d_i:_ p_l_o:_ m_a:_ t_I_ S_@_n_ b_@_ t_s_i:_ U_ N_@_n_
Discounter	d_I_s_ k_aU_n_ t_6:_
Disney World	d_I_s_ n_E_I_ v_9:_6:_l_t_
Duell	d_u:_ E_l_
DVD Shops	d_e:_ f_aU_ d_e:_ S_O_p_s_

Table A.1 – continued from previous page

Query	Syllabification
Ecuador	Q_e:_ k_U_ a_ d_O6_
Edeka	Q_e:_ d_e:_ k_a:_
Ehegatten Splitting	Q_e:_ @_ g_a_ t_@_n_ s_p_l_I_ t_I_N_
eines Schwarzen Loches	Q_aI_ n_@_s_ S_v_a6_ t_s_@_n_ l_O_ x_@_s_
Eintracht Frankfurt	Q_aI_n_ t_r_a_x_t_ f_r_a_N_k_ f_U6_t_
Eisenbahnorchester Sankt Wendel (*)	Q_aI_ z_@_n_ b_a:_n_ Q_O6_ k_E_s_ t_6:_ z_a_N_k_t_ v_E_n_ d_@_l_
Election Party (*)	Q_i:_ l_E_k_ t_S_@_n_ p_a:_6:_ t_i:_
Energiewende	Q_e:_ n_E6_ g_i:_ v_E_n_ d_@_
Epizentrum Deutschland	Q_e:_ p_i:_ t_s_E_n_ t_r_U_m_ d_OY_t_S_ l_a_n_t_
Eric Holder	Q_e:_ r_I_k_ h_O_l_ d_6:_
erneuerbarer Energien	Q_E6_ n_OY_ 6:_ b_a:_ r_6:_ Q_e:_ n_E6_ g_i:_ @_n_
Erwin Huber	Q_E6_ v_i:_n_ h_u:_ b_6:_
EU Agrarminister	Q_e:_ Q_u:_ Q_a_ g_r_a:_6:_ m_i:_ n_I_s_ t_6:_
EU Beschluss	Q_e:_ Q_u:_ b_@_ S_l_U_s_
EU Mitgliedsstaat	Q_e:_ Q_u:_ m_I_t_ g_l_i:_t_s_ S_t_a:_t_
Europa	Q_OY_ r_o:_ p_a:_
europäsche Union	Q_OY_ r_o:_ p_E:_ I_ S_@_ Q_U_n_ j_o:_n_
europäische Zentralbank	Q_OY_ r_o:_ p_E:_ I_ S_@_ t_s_E_n_ t_r_a:_l_ b_a_N_k_
Europa Parlament	Q_OY_ r_o:_ p_a:_ p_a6_ l_a_ m_E_n_t_
EU Staaten	Q_e:_ Q_u:_ S_t_a:_ t_@_n_
Ex Präsident	Q_E_k_s_ p_r_E_ z_i:_ d_E_n_t_
Fach Ethik	f_a_x_ Q_e:_ t_I_k_
Familie Deichmann	f_a_ m_i:_l_ j_@_ d_aI_C_ m_a_n_
FC Bayern	Q_E_f_ t_s_e:_ b_aI_ 6:_n_
FC Köln	Q_E_f_ t_s_e:_ k_9:_l_n_
FDP	Q_E_f_ d_e:_ p_e:_
Felix Sturm	f_e:_ l_I_k_s_ S_t_U6_m_
Fidel Castro	f_i:_ d_@_l_ k_a_s_ t_r_o:_
Finanzkrise	f_i:_ n_a_n_t_s_ k_r_i:_ z_@_

Table A.1 – continued from previous page

Query	Syllabification
Finanzminister Steinbrück	f_i:_ n_a_n_t_s_ m_i:_ n_I_s_ t_6:_ S_t_aI_n_ b_r_Y_k_
First Lady	f_I6_s_t_ l_E_I_ d_i:_
Forschungsprogramm	f_O6_ S_U_N_s_ p_r_o:_ g_r_a_m_
Frank Plasberg	f_r_a_N_k_ p_l_a_s_ b_E6_k_
Frank Walter Steinmeier	f_r_a_N_k_ v_a_l_ t_6:_ S_t_aI_n_ m_aI_ 6:_
Franz Josef Strauß	f_r_a_n_t_s_ j_o:_ z_E_f_ S_t_r_aU_s_
Franz Maget	f_r_a_n_t_s_ m_a:_ g_E_t_
Franz Müntefering	f_r_a_n_t_s_ m_Y_n_ t_@_ f_e:_ r_I_N_
Freenet Mobilcom	f_r_i:_ n_E_t_ m_o:_ b_i:_l_ k_O_m_
freie Demokraten	f_r_aI_ @_ d_e:_ m_o:_ k_r_a:_ t_@_n_
freien Wählern	f_r_aI_ @_n_ v_E:_ l_6:_n_
Gabor Halazs (*)	g_a:_ b_o:_6:_ h_a_ l_a_S_
Gefangenenlagers	g_@_ f_a_ N_@_ n_@_n_ l_a:_ g_6:_s_
Gegenspieler	g_e:_ g_@_n_ S_p_i:_ l_6:_
Gegner	g_e:_g_ n_6:_
Generals Nkunda	g_e:_ n_@_ r_a:_l_s_ Q_E_n_ k_U_n_ d_a:_
Generation	g_e:_ n_@_ r_a_ t_s_j_o:_n_
Genossin Merkel	g_@_ n_O_ s_I_n_ m_E6_ k_@_l_
George Bush	d_Z_O_d_Z_ b_U_S_
Gerhard Schröder	g_e:_6:_ h_a6_t_ S_r_2:_ d_6:_
Gesundheitsministerin Schmidt	g_@_ z_U_n_t_ h_aI_t_s_ m_i:_ n_I_s_ t_@_ r_I_n_ S_m_I_t_
Gewissensentscheidung	g_@_ v_I_ s_@_n_s_ Q_E_n_t_ S_aI_ d_U_N_
Gott	g_O_t_
Grundgesetz	g_r_U_n_t_ g_@_ z_E_t_s_
Grünen	g_r_y:_ n_@_n_
Guantanamo	g_u:_ a_n_ t_a:_ n_a_ m_o:_
Guido Westerwelle	g_i:_ d_o:_ v_E_s_ t_6:_ v_E_ l_@_
Guitar Hero	g_i:_ t_a6_ h_e:_ r_o:_
Hamas	h_a_ m_a_s_
Hamburger Parteitag	h_a_m_ b_U6_ g_6:_ p_a6_ t_aI_ t_a:_k_
Hamburger SV	h_a_m_ b_U6_ g_6:_ Q_E_s_ f_aU_

Table A.1 – continued from previous page

Query	Syllabification
Handwerker	h_a_n_t_ v_E6_ k_6:_
Hans Tietmeyer	h_a_n_s_ t_i:_t_ m_aI_ 6:_
Hartz vier	h_a:_6:_t_s_ f_i:_6:_
Harvard University	h_a6_ v_6:_t_ j_u:_ n_i:_ v_9:_6:_ s_I_ t_i:_
Hasardeuren	h_a_ z_a6_ d_2:_ r_@_n_
Haschisch Konsumenten	h_a_ S_I_S_ k_O_n_ z_u:_ m_E_n_ t_@_n_
Hauptschule Vilseck	h_aU_p_t_ S_u:_ l_@_ v_I_l_ z_E_k_
Haushalte	h_aU_s_ h_a_l_ t_@_
Heavy Metal	h_E_ v_i:_ m_E_ t_@_l_
Heide Simonis	h_aI_ d_@_ z_i:_ m_o:_ n_I_s_
Heiner Brand	h_aI_ n_6:_ b_r_a_n_t_
Helmut Kohl	h_E_l_ m_u:_t_ k_o:_l_
Herzlich Willkommen	h_E6_t_s_ l_I_C_ v_I_l_ k_O_ m_@_n_
Hessen CDU	h_E_ s_@_n_ t_s_e:_ d_e:_ u:_
Himalaya	h_i:_ m_a:_ l_a_ j_a:_
Hip Hop	h_I_p_ h_O_p_
Hisbollah	h_I_s_ b_O_ l_a:_
Hoffenheim (*)	h_O_ f_@_n_ h_aI_m_
Hoffnungsträger John F Kennedy	h_O_f_ n_U_N_s_ t_r_E:_ g_6:_ d_Z_O_n_ Q_E_f_ k_E_ n_@_ d_i:_
Hollywood Film	h_O_ l_i:_ v_U_t_ f_I_l_m_
Holocaust	h_o:_ l_o:_ k_aU_s_t_
Holocaust Leugner	h_o:_ l_o:_ k_aU_s_t_ l_OY_g_ n_6:_
Hot Spots	h_O_t_ S_p_O_t_s_
Houston	j_u:_s_ t_@_n_
Humboldt Universität	h_U_m_ b_O_l_t_ Q_U_ n_i:_ v_E6_ z_i:_ t_E:_t_
Hutu (*)	h_u:_ t_u:_
Hyde Park	h_aI_t_ p_a6_k_
Hypo Bank	h_y:_ p_o:_ b_a_N_k_
Hypo Real	h_y:_ p_o:_ r_e:_ a:_l_
Hypo Real Estate	h_y:_ p_o:_ r_e:_ a:_l_ Q_E_s_ t_E_I_t_
IBF Gürtel (*)	Q_i:_ b_e:_ Q_E_f_ g_Y6_ t_@_l_
ICE Trassen	Q_i:_ t_s_e:_ Q_e:_ t_r_a_ s_@_n_

Table A.1 – continued from previous page

Query	Syllabification
IG Metall	Q_i:_ g_e:_ m_e:_ t_a_l_
Ilse Aigner	Q_I_l_ z_@_ Q_aI_ g_n_6:_
Immanuel Kant	Q_I_ m_a:_ n_u:_ E_l_ k_a_n_t_
Immobilienboom Amerikas (*)	Q_I_ m_o:_ b_i:_l_ j_@_n_ b_u:_m_ Q_a_ m_e:_ r_i:_ k_a:_s_
Industrienationen	Q_I_n_ d_U_s_ t_r_i:_ n_a_ t_s_j_o:_ n_@_n_
injiziert	Q_I_n_ j_i:_ t_s_i:_6:_t_
ins Heilige Land	Q_I_n_s_ h_aI_ l_I_ g_@_ l_a_n_t_
Investitionspaket (*)	Q_I_n_ v_E_s_ t_i:_ t_s_j_o:_n_s_ p_a_ k_e:_t_
Investitionsprogramm	Q_I_n_ v_E_s_ t_i:_ t_s_j_o:_n_s_ p_r_o:_ g_r_a_m_
Irak	Q_i:_ r_a:_k_
Irlands Wirtschaft	Q_I6_ l_a_n_t_s_ v_I6_t_ S_a_f_t_
Island	Q_i:_s_ l_a_n_t_
Jamaika Koalition	d_Z_a_ m_aI_ k_a:_ k_o:_ a_ l_i:_ t_s_j_o:_n_
Joachim Löw	j_o:_ Q_a_ x_i:_m_ l_2:_f_
Johannes B Kerner	j_o:_ h_a_ n_@_s_ b_e:_ k_E6_ n_6:_
Johnny Depp	d_Z_O_ n_i:_ d_E_p_
Juden	j_u:_ d_@_n_
jüdisches Leben	j_y:_ d_I_ S_@_s_ l_e:_ b_@_n_
Justizminister	j_U_s_ t_i:_t_s_ m_i:_ n_I_s_ t_6:_
Kanalinsel Jersey	k_a_ n_a:_l_ Q_I_n_ z_@_l_ d_Z_2:_6:_ z_i:_
Kannegiesser	k_a_ n_@_ g_i:_ s_6:_
Kanzlerkandidat	k_a_n_t_s_ l_6:_ k_a_n_ d_i:_ d_a:_t_
Kanzlerkandidat Frank Walter Steinmeier	k_a_n_t_s_ l_6:_ k_a_n_ d_i:_ d_a:_t_ f_r_a_N_k_ v_a_l_ t_6:_ S_t_aI_n_ m_aI_ 6:_
Kanzlerkandidat Steinmeier	k_a_n_t_s_ l_6:_ k_a_n_ d_i:_ d_a:_t_ S_t_aI_n_ m_aI_ 6:_
Kanzlerschaft Angela Merkels	k_a_n_t_s_ l_6:_ S_a_f_t_ Q_a_N_ g_e:_ l_a:_ m_E6_ k_@_l_s_
Karl Liebknecht	k_a:_6:_l_ l_i:_p_ k_n_E_C_t_
Kasinokapitalismus (*)	k_a_ z_i:_ n_o:_ k_a_ p_i:_ t_a_ l_I_s_ m_U_s_
Kenias Hauptstadt Nairobi	k_e:_n_ j_a_s_ h_aU_p_t_ S_t_a_t_ n_aI_ r_o:_ b_i:_
KFZ Steuer	k_a:_ Q_E_f_ t_s_E_t_ S_t_OY_ 6:_

Table A.1 – continued from previous page

Query	Syllabification
KFZ Steuerersparnis	k_a:_ Q_E_f_ t_s_E_t_ S_t_OY_ 6:_ Q_E6_ S_p_a:_6:_ n_I_s_
Kigali	k_i:_ g_a:_ l_i:_
Kinder	k_I_n_ d_6:_
Klagemauer in Jerusalem	k_l_a:_ g_@_ m_aU_ 6:_ Q_I_n_ j_e:_ r_u:_ z_a_ l_E_m_
Klaus Wowereit	k_l_aU_s_ v_o:_ v_@_ r_aI_t_
Klitschko Brüder	k_l_I_t_S_ k_o:_ b_r_y:_ d_6:_
Knecht Ruprecht	k_n_E_C_t_ r_u:_p_ r_E_C_t_
KO (*)	k_a:_ Q_o:_
Koalitionspartner SPD	k_o:_ a_ l_i:_ t_s_j_o:_n_s_ p_a6_t_ n_6:_ Q_E_s_ p_e:_ d_e:_
Kochs Wahlschlappe	k_O_x_s_ v_a:_l_ S_l_a_ p_@_
Kölns Oberbürgermeister	k_9:_l_n_s_ Q_o:_ b_6:_ b_Y6_ g_6:_ m_aI_s_ t_6:_
Kölns Oberbürgermeister Fritz Schrammer (*)	k_9:_l_n_s_ Q_o:_ b_6:_ b_Y6_ g_6:_ m_aI_s_ t_6:_ f_r_I_t_s_ S_r_a_ m_6:_
Kölsche Mentalität	k_9:_l_ S_@_ m_E_n_ t_a_ l_i:_ t_E:_t_
Kommando Spezialkräfte KSK	k_O_ m_a_n_ d_o:_ S_p_e:_ t_s_j_a:_l_ k_r_E_f_ t_@_ k_a:_ Q_E_s_ k_a:_
Kompetenz	k_O_m_ p_e:_ t_E_n_t_s_
König Artur	k_2:_ n_I_C_ Q_a:_6:_ t_u:_6:_
König Herodes	k_2:_ n_I_C_ h_e:_ r_o:_ d_E_s_
Konjunktur	k_O_n_ j_U_N_k_ t_u:_6:_
Konjunktur ankurbeln	k_O_n_ j_U_N_k_ t_u:_6:_ Q_a_n_ k_U6_ b_@_l_n_
Konjunkturpaket	k_O_n_ j_U_N_k_ t_u:_6:_ p_a_ k_e:_t_
Konjunkturprogramm	k_O_n_ j_U_N_k_ t_u:_6:_ p_r_o:_ g_r_a_m_
Konjunkturspritzen (*)	k_O_n_ j_U_N_k_ t_u:_6:_ S_p_r_I_ t_s_@_n_
konkrete Zusagen	k_O_N_ k_r_e:_ t_@_ t_s_u:_ z_a:_ g_@_n_
Konstrukteurs WM	k_O_n_s_ t_r_U_k_ t_2:_6:_s_ v_e:_ Q_E_m_
KO Sieg (*)	k_a:_ Q_o:_ z_i:_k_
Krötensonderkommando (*)	k_r_2:_ t_@_n_ z_O_n_ d_6:_ k_O_ m_a_n_ d_o:_
Kurt Beck	k_U6_t_ b_E_k_

Table A.1 – continued from previous page

Query	Syllabification
Landeskriminalamt Saarbrücken	l_a_n_ d_@_s_ k_r_I_ m_i:_ n_a:_l_ Q_a_m_t_ z_a:_6:_ b_r_Y_ k_@_n_
Landtagsabgeordneten	l_a_n_t_ t_a:_k_s_ Q_a_p_ g_@_ Q_O6_t_ n_@_ t_@_n_
Las Vegas	l_a:_s_ v_e:_ g_a_s_
Lebewesen	l_e:_ b_@_ v_E_ s_@_n_
Lehman Brothers	l_e:_ m_a_n_ b_r_O_ z_a_s_
Lewis Hamilton	l_u:_ I_s_ h_E_ m_I_l_ t_@_n_
Libanon	l_i:_ b_a_ n_O_n_
Linkskurs	l_I_N_k_s_ k_U6_s_
Linkspartei	l_I_N_k_s_ p_a6_ t_aI_
LKW Maut	Q_E_l_ k_a:_ v_e:_ m_aU_t_
Londoner Schmuddelwetter	l_O_n_ d_o:_ n_6:_ S_m_U_ d_@_l_ v_E_ t_6:_
Long Island	l_O_N_ Q_i:_s_ l_a_n_t_
Lothar Bisky	l_o:_ t_a:_6:_ b_I_s_ k_i:_
Lothar Matthäus	l_o:_ t_a:_6:_ m_a_ t_E:_ U_s_
Luca Toni	l_u:_ k_a:_ t_o:_ n_i:_
Ludwig Erhard	l_u:_t_ v_I_C_ Q_e:_6:_ h_a6_t_
Lukas Podolski	l_u:_ k_a_s_ p_o:_ d_O_l_s_ k_i:_
Mailänder Dom	m_aI_ l_E_n_ d_6:_ d_o:_m_
Manila	m_a_ n_i:_ l_a:_
Margarete Teiner (*)	m_a:_6:_ g_a_ r_e:_ t_@_ t_aI_ n_6:_
Marins erstes Bundesligator (*)	m_a_ r_i:_n_s_ Q_E6_s_ t_@_s_ b_U_n_ d_@_s_ l_i:_ g_a:_ t_o:_6:_
Markt	m_a:_6:_k_t_
Markus Söder	m_a:_6:_ k_U_s_ s_2:_ d_6:_
Martin Luther King	m_a:_6:_ t_i:_n_ l_U_ t_6:_ k_I_N_
McLaren Mercedes	m_E_k_ l_a:_ r_@_n_ m_E6_ t_s_e:_ d_@_s_
Mecklenburg Vorpommern	m_E_k_ l_@_n_ b_U6_k_ f_o:_6:_ p_O_ m_6:_n_
Mehrwegflasche	m_e:_6:_ v_e:_k_ f_l_a_ S_@_
Merkels Außenminister	m_E6_ k_@_l_s_ Q_aU_ s_@_n_ m_i:_ n_I_s_ t_6:_
Merkels Widersacher	m_E6_ k_@_l_s_ v_i:_ d_6:_ z_a_ x_6:_
Michael Glos	m_I_ C_a_ e:_l_ g_l_o:_s_

Table A.1 – continued from previous page

Query	Syllabification
Michael Schumacher	m_I_ C_a_ e:_l_ S_u:_ m_a_ x_6:_
Michael Steinbrecher	m_I_ C_a_ e:_l_ S_t_aI_n_ b_r_E_ C_6:_
Michelle Obama	m_I_ C_E_l_ Q_o:_ b_a:_ m_a:_
Milchbauern	m_I_l_C_ b_aU_ 6:_n_
Milliarden Euro Konjunkturpaket	m_I_l_ j_a6_ d_@_n_ Q_OY_ r_o:_ k_O_n_ j_U_N_k_ t_u:_6:_ p_a_ k_e:_t_
Millionen Wanderarbeiter	m_I_l_ j_o:_ n_@_n_ v_a_n_ d_6:_ Q_a6_ b_aI_ t_6:_
Ministerpräsident	m_i:_ n_I_s_ t_6:_ p_r_E_ z_i:_ d_E_n_t_
Ministerpräsident Oettinger	m_i:_ n_I_s_ t_6:_ p_r_E_ z_i:_ d_E_n_t_ Q_9:_ t_I_ N_6:_
Monte Carlo	m_O_n_ t_@_ k_a:_6:_ l_o:_
Moralische Empörung	m_o:_ r_a:_ l_I_ S_@_ Q_E_m_ p_2:_ r_U_N_
MTV	Q_E_m_ t_i:_ v_i:_
MTV Awards	Q_E_m_ t_i:_ v_i:_ Q_@_ v_O_t_s_
Murad Kurnaz	m_u:_ r_a_t_ k_U6_ n_a_t_s_
Nachbarschaft	n_a_x_ b_a6_ S_a_f_t_
Nachtraser (*)	n_a_x_t_ r_a:_ z_6:_
NATO Gipfel	n_a:_ t_o:_ g_I_ p_f_@_l_
NATO Mitglied	n_a:_ t_o:_ m_I_t_ g_l_i:_t_
Nazis	n_a:_ t_s_i:_s_
Neckar	n_E_ k_a6_
Nettoeffekt (*)	n_E_ t_o:_ Q_E_ f_E_k_t_
Neuankömmlinge ohne gültige Aufenthaltserlaubnis	n_OY_ Q_a_n_ k_9:_m_ l_I_ N_@_ Q_o:_ n_@_ g_Y_l_ t_I_ g_@_ Q_aU_f_ Q_E_n_t_ h_a_l_t_s_ Q_E6_ l_aU_p_ n_I_s_
New Deal	n_j_u:_ d_i:_l_
New York	n_j_u:_ j_O6_k_
New York Times	n_j_u:_ j_O6_k_ t_aI_m_s_
Nordkoreas Militär	n_O6_t_ k_o:_ r_e:_ a_s_ m_i:_ l_i:_ t_E:_6:_
NPD Verbot	Q_E_n_ p_e:_ d_e:_ f_E6_ b_o:_t_
NS Zeit	Q_E_n_ Q_E_s_ t_s_aI_t_
Obama	Q_o:_ b_a:_ m_a:_
Obama Feeling	Q_o:_ b_a:_ m_a:_ f_i:_ l_I_N_

Table A.1 – continued from previous page

Query	Syllabification
ohne Antwort	Q_o:_ n_@_ Q_a_n_t_ v_O6_t_
Olaf Scholz	Q_o:_ l_a_f_ S_O_l_t_s_
Oliver Kahn	Q_O_ l_i:_ v_6:_ k_a:_n_
olympischen Spiele	Q_o:_ l_Y_m_ p_I_ S_@_n_ S_p_i:_ l_@_
Online Durchsuchung	Q_O_n_ l_aI_n_ d_U6_C_ z_u:_ x_U_N_
Opel	Q_o:_ p_@_l_
Opel Autohaus	Q_o:_ p_@_l_ Q_aU_ t_o:_ h_aU_s_
Opfer	Q_O_ p_f_6:_
Oprah Winfrey	Q_o:_ p_r_a:_ v_I_n_ f_r_E_I_
Oskar Lafontaine	Q_O_s_ k_a:_6:_ l_a_ f_O_n_ t_E:_n_
Ostalgie	Q_O_s_ t_a_l_ g_i:_
OSZE	Q_o:_ Q_E_s_ t_s_E_t_ Q_e:_
Panathinaikos Athen	p_a_ n_a_ t_I_ n_a:_ i:_ k_o:_s_ Q_a_ t_e:_n_
Papst	p_a:_p_s_t_
Papst Johannes Paul	p_a:_p_s_t_ j_o:_ h_a_ n_@_s_ p_aU_l_
Paralympics	p_a_ r_a_ l_Y_m_ p_I_k_s_
Parteivorstand	p_a6_ t_aI_ f_o:_6:_ S_t_a_n_t_
Party	p_a:_6:_ t_i:_
Patchwork Familie	p_E_t_S_ v_9:_6:_k_ f_a_ m_i:_l_ j_@_
Peer Steinbrück	p_e:_6:_ S_t_aI_n_ b_r_Y_k_
per Schiff transportiert	p_E6_ S_I_f_ t_r_a_n_s_ p_O6_ t_i:_6:_t_
Peter Frey	p_e:_ t_6:_ f_r_aI_
Peter Hahne	p_e:_ t_6:_ h_a:_ n_@_
Peter Kloeppel	p_e:_ t_6:_ k_l_9:_ p_@_l_
Peter Ramsauer	p_e:_ t_6:_ r_a_m_ z_aU_ 6:_
Peter Struck	p_e:_ t_6:_ S_t_r_U_k_
Petra Gerster	p_e:_ t_r_a:_ g_E6_s_ t_6:_
Pharaonen	f_a_ r_a_ Q_o:_ n_@_n_
Pisa Studie	p_i:_ s_a:_ S_t_u:_ d_j_@_
Pius Bruderschaft	p_i:_ U_s_ b_r_u:_ d_6:_ S_a_f_t_
Podolski	p_o:_ d_O_l_s_ k_i:_
Pogrom	p_o:_ g_r_o:_m_
Politik	p_o:_ l_i:_ t_I_k_

Table A.1 – continued from previous page

Query	Syllabification
Politikwissenschaftler	p_o:_ l_i:_ t_I_k_ v_I_ s_@_n_ S_a_f_t_ l_6:_
Politische Probleme	p_o:_ l_i:_ t_I_ S_@_ p_r_o:_ b_l_e:_ m_@_
Postchef Klaus Zumwinkel	p_O_s_t_ S_E_f_ k_l_aU_s_ t_s_U_m_ v_I_N_ k_@_l_
Präsident	p_r_E_ z_i:_ d_E_n_t_
Präsident Clinton	p_r_E_ z_i:_ d_E_n_t_ k_l_I_n_ t_@_n_
Präsidenten des Deutschen Bauernverbandes	p_r_E_ z_i:_ d_E_n_ t_@_n_ d_E_s_ d_OY_ t_S_@_n_ b_aU_ 6:_n_ f_E6_ b_a_n_ d_@_s_
Präsident Sarkozy	p_r_E_ z_i:_ d_E_n_t_ z_a6_ k_o:_ z_i:_
PR Berater Huntzinger (*)	p_e:_ Q_E6_ b_@_ r_a:_ t_6:_ h_U_n_ t_s_I_ N_6:_
Prinz Poldi (*)	p_r_I_n_t_s_ p_O_l_ d_i:_
Probleme weg geschoben	p_r_o:_ b_l_e:_ m_@_ v_e:_k_ g_@_ S_o:_ b_@_n_
Profil schärfen	p_r_o:_ f_i:_l_ S_E6_ f_@_n_
Programme	p_r_o:_ g_r_a_ m_@_
Prozent	p_r_o:_ t_s_E_n_t_
PR Termin	p_e:_ Q_E6_ t_E6_ m_i:_n_
Puerto Rico	p_u:_ E6_ t_o:_ r_i:_ k_o:_
Putin	p_u:_ t_i:_n_
Ralf Rangnick	r_a_l_f_ r_a_N_ n_I_k_
Rassismus in Amerika	r_a_ s_I_s_ m_U_s_ Q_I_n_ Q_a_ m_e:_ r_i:_ k_a:_
Rauchmelder	r_aU_x_ m_E_l_ d_6:_
Rebellengeneral Nkunda (*)	r_e:_ b_E_ l_@_n_ g_e:_ n_@_ r_a:_l_ Q_E_n_ k_U_n_ d_a:_
Regentschaft	r_e:_ g_E_n_t_ S_a_f_t_
Regierungsära (*)	r_e:_ g_i:_ r_U_N_s_ Q_E:_ r_a:_
reizvollen	r_aI_t_s_ f_O_ l_@_n_
Rekordolympiasiegerin Claudia Pechstein (*)	r_e:_ k_O6_t_ Q_o:_ l_Y_m_ p_j_a:_ z_i:_ g_@_ r_I_n_ k_l_aU_ d_i:_ a:_ p_E_C_ S_t_aI_n_
Rendite	r_E_n_ d_i:_ t_@_
Reykjavik	r_aI_k_ j_a_ v_I_k_
Rezept	r_e:_ t_s_E_p_t_
Rezession	r_e:_ t_s_E_ s_j_o:_n_
Rheinland Pfalz	r_aI_n_ l_a_n_t_ p_f_a_l_t_s_
Ribery (*)	r_i:_ b_@_ r_i:_

Table A.1 – continued from previous page

Query	Syllabification
Richard Williamson	r_I_ C_a:_6:_t_ v_I_l_ j_@_m_ z_@_n_
Riesenautos (*)	r_i:_ z_@_n_ Q_aU_ t_o:_s_
Rodelherren (*)	r_o:_ d_@_l_ h_E_ r_@_n_
Rohölpreise	r_o:_ Q_2:_l_ p_r_aI_ z_@_
Roland Berger	r_o:_ l_a_n_t_ b_E6_ g_6:_
Roland Koch	r_o:_ l_a_n_t_ k_O_x_
Romantik	r_o:_ m_a_n_ t_I_k_
Rosa Luxemburg	r_o:_ z_a:_ l_U_k_ s_@_m_ b_U6_k_
russische Führung	r_U_ s_I_ S_@_ f_y:_ r_U_N_
Russlands Vordenker	r_U_s_ l_a_n_t_s_ f_o:_6:_ d_E_N_ k_6:_
Sachsen Anhalt	z_a_k_ s_@_n_ Q_a_n_ h_a_l_t_
Sachverständigenrat	z_a_x_ f_E6_ S_t_E_n_ d_I_ g_@_n_ r_a:_t_
Saddam Hussein	z_a_ d_a_m_ h_U_ s_e:_ i:_n_
Sahra Wagenknecht	z_a:_ r_a:_ v_a:_ g_@_n_ k_n_E_C_t_
SAP	Q_E_s_ Q_a:_ p_e:_
Sarah Palin (*)	z_a:_ r_a:_ p_E_I_ l_I_n_
Schäfer Gümbel (*)	S_E:_ f_6:_ g_Y_m_ b_@_l_
Schrotthändler	S_r_O_t_ h_E_n_t_ l_6:_
Schulsanierung (*)	S_u:_l_ z_a_ n_i:_ r_U_N_
Schwesterpartei	S_v_E_s_ t_6:_ p_a6_ t_aI_
Schwesterpartei CSU fordert Entlastungen	S_v_E_s_ t_6:_ p_a6_ t_aI_ t_s_e:_ Q_E_s_ Q_u:_ f_O6_ d_6:_t_ Q_E_n_t_ l_a_s_ t_U_ N_@_n_
sechs Prozent weniger	z_E_k_s_ p_r_o:_ t_s_E_n_t_ v_e:_ n_I_ g_6:_
SED Erbe	Q_E_s_ Q_e:_ d_e:_ Q_E6_ b_@_
Senator Obama	z_e:_ n_a:_ t_o:_6:_ Q_o:_ b_a:_ m_a:_
Shooting Star	S_u:_ t_I_N_ s_t_a:_6:_
Sichtbarkeit	z_I_C_t_ b_a:_6:_ k_aI_t_
Sommerinterview	z_O_ m_6:_ Q_I_n_ t_6:_ v_j_u:_
soziales Profil	z_o:_ t_s_j_a:_ l_@_s_ p_r_o:_ f_i:_l_
spanische Wirtschaft	S_p_a:_ n_I_ S_@_ v_I6_t_ S_a_f_t_
SPD Finanzminister	Q_E_s_ p_e:_ d_e:_ f_i:_ n_a_n_t_s_ m_i:_ n_I_s_ t_6:_
SPD Führung	Q_E_s_ p_e:_ d_e:_ f_y:_ r_U_N_

Table A.1 – continued from previous page

Query	Syllabification
SPD Haushaltsexperten	Q_E_s_ p_e:_ d_e:_ h_aU_s_ h_a_l_t_s_ Q_E_k_s_ p_E6_ t_@_n_
SPD Innenminister	Q_E_s_ p_e:_ d_e:_ Q_I_ n_@_n_ m_i:_ n_I_s_ t_6:_
SPD Politik	Q_E_s_ p_e:_ d_e:_ p_o:_ l_i:_ t_I_k_
SPD Spitze	Q_E_s_ p_e:_ d_e:_ S_p_I_ t_s_@_
Spielwarenmesse	S_p_i:_l_ v_a:_ r_@_n_ m_E_ s_@_
Spitzenkandidatin	S_p_I_ t_s_@_n_ k_a_n_ d_i:_ d_a:_ t_I_n_
Sprit	S_p_r_I_t_
Staatsanwaltschaft	S_t_a:_t_s_ Q_a_n_ v_a_l_t_ S_a_f_t_
Steuerreform	S_t_OY_ 6:_ r_e:_ f_O6_m_
Steuerstreit	S_t_OY_ 6:_ S_t_r_aI_t_
Südkorea	z_y:_t_ k_o:_ r_e:_ a:_
Superlative	z_u:_ p_6:_ l_a_ t_i:_ v_@_
Synagoge	z_y:_ n_a_ g_o:_ g_@_
Tabellenende (*)	t_a_ b_E_ l_@_n_ Q_E_n_ d_@_
Tagesthemen	t_a:_ g_@_s_ t_e:_ m_@_n_
Taliban	t_a_ l_i:_ b_a:_n_
Tango auf Türkisch	t_a_N_ g_o:_ Q_aU_f_ t_Y6_ k_I_S_
Tarifstreit	t_a_ r_i:_f_ S_t_r_aI_t_
Tarik Al Wazir	t_a:_ r_I_k_ Q_a_l_ v_a_ z_i:_6:_
Tengelmann	t_E_ N_@_l_ m_a_n_
Terroristen	t_E_ r_o:_ r_I_s_ t_@_n_
Thema Steuerentlastungen	t_e:_ m_a:_ S_t_OY_ 6:_ Q_E_n_t_ l_a_s_ t_U_ N_@_n_
Thorsten Schäfer Gümbel (*)	t_O6_s_ t_@_n_ S_E:_ f_6:_ g_Y_m_ b_@_l_
Tibet	t_i:_ b_E_t_
Tim Borowski	t_I_m_ b_o:_ r_O_f_s_ k_i:_
Trainer	t_r_E:_ n_6:_
Tribüne	t_r_i:_ b_y:_ n_@_
TV Sender	t_e:_ f_aU_ z_E_n_ d_6:_
Umweltminister Jürgen Trittin	Q_U_m_ v_E_l_t_ m_i:_ n_I_s_ t_6:_ j_Y6_ g_@_n_ t_r_I_ t_i:_n_
UN Friedenstruppen	Q_u:_ Q_E_n_ f_r_i:_ d_@_n_s_ t_r_U_ p_@_n_

Table A.1 – continued from previous page

Query	Syllabification
Union	Q_U_n_ j_o:_n_
unsere Sprit fressenden Monsterautos (*)	Q_U_n_ z_@_ r_@_ S_p_r_I_t_ f_r_E_ s_@_n_ d_@_n_ m_O_n_s_ t_6:_ Q_aU_ t_o:_s_
US Filme	Q_u:_ Q_E_s_ f_I_l_ m_@_
US Präsident	Q_u:_ Q_E_s_ p_r_E_ z_i:_ d_E_n_t_
Valentinstag	v_a_ l_E_n_ t_I_n_s_ t_a:_k_
Venedig	v_e:_ n_e:_ d_I_C_
Vereinigten Arabischen Emiraten	f_E6_ Q_aI_ n_I_C_ t_@_n_ Q_a_ r_a:_ b_I_ S_@_n_ Q_e:_ m_i:_ r_a:_ t_@_n_
Vereinigten Staaten	f_E6_ Q_aI_ n_I_C_ t_@_n_ S_t_a:_ t_@_n_
Vereinten Nationen	f_E6_ Q_aI_n_ t_@_n_ n_a_ t_s_j_o:_ n_@_n_
Verhältnissen	f_E6_ h_E_l_t_ n_I_ s_@_n_
Vertrauen	f_E6_ t_r_aU_ @_n_
Viertel ihres Werts	f_I6_ t_@_l_ Q_i:_ r_@_s_ v_e:_6:_t_s_
Völkermord	f_9:_l_ k_6:_ m_O6_t_
Vorbild	f_o:_6:_ b_I_l_t_
Vulkanausbrüche	v_U_l_ k_a:_n_ Q_aU_s_ b_r_Y_ C_@_
VW	f_aU_ v_e:_
VW Aktien	f_aU_ v_e:_ Q_a_k_ t_s_j_@_n_
Waffen	v_a_ f_@_n_
Waffenstillstand	v_a_ f_@_n_ S_t_I_l_ S_t_a_n_t_
Wahlen	v_a:_ l_@_n_
Wahlkampfthema Bildung	v_a:_l_ k_a_m_p_f_ t_e:_ m_a:_ b_I_l_ d_U_N_
Wahlparty	v_a:_l_ p_a:_6:_ t_i:_
Wahlschlappe	v_a:_l_ S_l_a_ p_@_
Wahlsieg	v_a:_l_ z_i:_k_
Wall Street	v_a_l_ s_t_r_i:_t_
Washington	v_O_ S_I_N_ t_@_n_
Weltmeister	v_E_l_t_ m_aI_s_ t_6:_
wichtige Personalentscheidungen	v_I_C_ t_I_ g_@_ p_E6_ z_o:_ n_a:_l_ Q_E_n_t_ S_aI_ d_U_ N_@_n_
Wirren der Wirtschaftskrise	v_I_ r_@_n_ d_e:_6:_ v_I6_t_ S_a_f_t_s_ k_r_i:_ z_@_
Wirtschaft	v_I6_t_ S_a_f_t_

Table A.1 – continued from previous page

Query	Syllabification
Wirtschaftliche Leistung der EU	v_I6_t_ S_a_f_t_ l_I_ C_@_ l_aI_s_ t_U_N_ d_e:_6:_ Q_e:_ Q_u:_
Wirtschaftskrise	v_I6_t_ S_a_f_t_s_ k_r_i:_ z_@_
Wirtschaftsmacht	v_I6_t_ S_a_f_t_s_ m_a_x_t_
Wirtschaftspolitik	v_I6_t_ S_a_f_t_s_ p_o:_ l_i:_ t_I_k_
Wirtschaftswunderland	v_I6_t_ S_a_f_t_s_ v_U_n_ d_6:_ l_a_n_t_
Wladimir Klitschko	v_l_a_ d_i:_ m_i:_6:_ k_l_I_t_S_ k_o:_
WWW	v_e:_ v_e:_ v_e:_
You Tube	j_u:_ t_u:_ b_@_
Ypsilanti	Q_Y_p_ s_i:_ l_a_n_ t_i:_
Zeitschrift	t_s_aI_t_ S_r_I_f_t_
Zentralrats der Juden	t_s_E_n_ t_r_a:_l_ r_a:_t_s_ d_e:_6:_ j_u:_ d_@_n_
Zick Zack Laufen	t_s_I_k_ t_s_a_k_ l_aU_ f_@_n_
zusätzliche Steuer	t_s_u:_ z_E_t_s_ l_I_ C_@_ S_t_OY_ 6:_

Bibliography

[1] M. Akbacak, D. Vergyri, and A. Stolcke. Open-vocabulary Spoken Term Detection using graphone-based hybrid recognition systems. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 5240–5243, 2008.

[2] P. Altendorf, H. Rode, D. Schneider, and J. Schon. Techniken für die Suche in audiovisuellen Medien: Vitalas - ein neues System zur Inhaltssuche. *FKT, Fachzeitschrift für Fernsehen, Film und elektronische Medien*, 12:701–704, 2009.

[3] E. Arisoy, H. Dutagaci, and L. M. Arslan. A unified language model for large vocabulary continuous speech recognition of Turkish. *Signal Processing*, 86(10):2844–2862, 2000.

[4] E. Arisoy and M. Saraclar. Lattice extension and vocabulary adaptation for Turkish LVCSR. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(1):163–173, 2009.

[5] C. Barras, E. Geoffrois, Z. Wu, and M. Liberman. Transcriber: development and use of a tool for assisting speech corpora production. *Speech Communication*, 33(1–2):5–22, 2001.

[6] D. Baum, D. Schneider, R. Bardeli, J. Schwenninger, B. Samlowski, T. Winkler, and J. Köhler. DiSCo - a German evaluation corpus for challenging problems in the broadcast domain. In *Proc. International Conference on Language Resources and Evaluation, LREC*, pages 1695–1699, 2010.

[7] D. Baum, D. Schneider, T. Mertens, and J. Köhler. Constrained subword units for speaker recognition. In *Proc. ISCA Odyssey Speaker and Language Recognition Workshop*, 2010.

[8] J. R. Bellegarda. Statistical language model adaptation: review and perspectives. *Speech Communication*, 42(1):93–108, 2004.

[9] U. Benner, I. Flechsig, G. Dogil, and B. Möbius. Coarticulatory resistance in a mental syllabary. In *Proc. International Congress of Phonetic Sciences, ICPhS*, pages 485–488, 2007.

[10] M. Bisani and H. Ney. Open vocabulary speech recognition with flat hybrid models. In *Proc. Interspeech 2005*, pages 725–728, 2005.

[11] M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434 – 451, 2008.

[12] O. L. Blouch and P. Collen. Improving phoneme-based spoken document retrieval with phonetic context expansion. In *Proc. IEEE International Conference on Multimedia and Expo, ICME*, pages 1217–1220, 2008.

[13] S. Breuer. *Multifunktionale und multilinguale Unit-Selection-Sprachsynthese*. PhD thesis, University of Bonn, 2009.

[14] J. Cernocky, I. Szoke, M. Fapso, M. Karafiat, L. Burget, J. Kopecky, F. Grezl, P. Schwarz, O. Glembek, I. Oparin, P. Smrz, and P. Matejka. Search in speech for public security and defense. In *Proc. IEEE Workshop on Signal Processing Applications for Public Security and Forensics, SAFE*, pages 1–7, 2007.

[15] C. Chelba, J. Silva, and A. Acero. Soft indexing of speech content for search in spoken documents. *Computer Speech & Language*, 21(3):458–478, July 2007.

[16] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*, pages 310–318, 1996.

[17] S. S. Chen and P. S. Gopalakrishnan. Clustering via the bayesian information criterion with applications in speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, volume 2, pages 645–648, 1998.

[18] Cisco. Visual networking index: Forecast and methodology 2010-2015, 2010.

[19] N. Cremelie and L. ten Bosch. Improving the recognition of foreign names and non-native speech by combining multiple grapheme-to-phoneme converters. In *Proc. ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition*, pages 151–154, 2001.

[20] M. Creutz, T. Hirsimäki, M. Kurimo, A. Puurula, J. Pylkkönen, V. Siivola, M. Varjokallio, E. Arisoy, M. Saraçlar, and A. Stolcke. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing*, 5(1):1–29, 2007.

[21] M. H. Davel and F. de Wet. Verifying pronunciation dictionaries using conflict analysis. In *Proc. Interspeech*, pages 1898–1901, 2010.

[22] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

[23] F. de Jong, D. Oard, R. Ordelman, and S. Raaijmakers, editors. *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, 2007.

[24] F. de Jong, T. Westerveld, and A. P. D. Vries. Multimedia search without visual analysis: The value of linguistic and contextual information. *IEEE IEEE Transactions on Circuits and Systems for Video Technology*, 17(3):365 – 371, 2007.

[25] M. Federico and N. Bertoldi. Broadcast news LM adaptation over time. *Computer Speech & Language*, 18(4):417–435, Oct. 2004.

[26] J. G. Fiscus, J. Ajot, and J. S. Garofolo. Results of the 2006 Spoken Term Detection evaluation. In *Proc. Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, 2007.

[27] M. Fleischman and D. Roy. Grounded language modeling for automatic speech recognition of sports video. In *Proceedings of HLT-NAACL*, 2008.

[28] Fraunhofer IAIS. *ARD Webduell*, 2009 (accessed August 23, 2011). `http://www.iais.fraunhofer.de/ard-webduell.html`.

[29] Fraunhofer IAIS. *Galileo Videolexikon*, 2010 (accessed August 23, 2011). `http://www.iais.fraunhofer.de/galileo-videolexikon.html`.

[30] Fraunhofer IAIS. *ARD-Mediathek Search*, 2011 (accessed August 23, 2011). `http://www.iais.fraunhofer.de/ard-videozitat.html`.

[31] M. Gales, D. Y. Kim, P. Woodland, H. Y. Chan, D. Mrva, R. Sinha, and S. Tranter. Progress in the CU-HTK broadcast news transcription system. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1513–1525, 2006.

[32] M. Gales and S. Young. The application of hidden markov models in speech recognition. *Found. Trends Signal Process.*, 1:195–304, January 2007.

[33] S. Galliano, E. Geoffrois, G. Gravier, J.-F. Bonastre, D. Mostefa, and K. Choukri. Corpus description of the ESTER evaluation campaign for the rich transcription of french broadcast news. In *Proc. International Conference on Language Resources and Evaluation, LREC*, pages 139–142, 2006.

[34] J. Garofolo, G. Auzanne, and E. Voorhees. The TREC spoken document retrieval track: A success story. In *Proc. Content Based Multimedia Information Access Conference*, 2000.

[35] J. Garofolo, J. G. Fiscus, and W. M. Fisher. Design and preparation of the 1996 HUB-4 broadcast news benchmark test corpora. In *Proc. DARPA Speech Recognition Workshop*, pages 15–21, 1997.

[36] J. Gaspers. Lexikon- und Sprachmodelladaption zur Verbesserung der automatischen Indizierung deutschsprachiger Nachrichtensendungen. Magisterarbeit, University of Bonn, 2011.

[37] J.-L. Gauvain, L. Lamel, and G. Adda. The LIMSI broadcast news transcription system. *Speech Commun.*, 37:89–108, May 2002.

[38] J.-L. Gauvain, L. Lamel, G. Adda, and M. Adda-Decker. Transcription of broadcast news. In *Proc. Eurospeech*, pages 907–910, 1997.

[39] J. L. Gauvain, L. Lamel, and M. Adda-Decker. Developments in continuous speech dictation using the ARPA WSJ task. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 65–68, 1995.

[40] U. Glavitsch and P. Schäuble. A system for retrieving speech documents. In *Proc. ACM SIGIR*, pages 168–176, 1992.

[41] D. Graff. The 1996 broadcast news speech and language-model corpus. In *Proc. DARPA Speech Recognition Workshop*, pages 11–14, 1997.

[42] S. Greenberg. Speaking in shorthand - a syllable-centric perspective for understanding pronunciation variation. *Speech Communication*, 29(2-4):159–176, 1999.

[43] T. Hain, L. Burget, J. Dines, P. N. Garner, A. E. Hannani, M. Huijbregts, M. Karafiat, M. Lincoln, and V. Wan. The AMIDA 2009 meeting transcription system. In *Proc. Interspeech*, 2010.

[44] D. Hakkani-Tür, F. Béchet, G. Riccardi, and G. Tur. Beyond ASR 1-best: Using word confusion networks in spoken language understanding. *Computer Speech & Language*, 20(4):495–514, Oct. 2006.

[45] T. Hazen, W. Shen, and C. White. Query-by-example Spoken Term Detection using phonetic posteriogram templates. In *Proc. IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, pages 421–426, 2009.

[46] R. Hecht, J. Riedler, and G. Backfried. German broadcast news transcription. In *Proc. Interspeech*, pages 1753–1756, 2002.

[47] T. Hirsimäki, M. Creutz, V. Siivola, M. Kurimo, S. Virpioja, and J. Pylkkönen. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech & Language*, 20(4):515–541, 2006.

[48] T. N. D. Huynh, W.-K. Hon, T.-W. Lam, and W.-K. Sung. Approximate string matching using compressed suffix arrays. In *Proc. Symposium on Combinatorial Pattern Matching*, pages 434–444, 2004.

[49] Ithaka Strategic Consulting. L'institut national de l'audiovisuel: Free content and rights licensing as complementary strategies. *Ithaka Case Studies in Sustainability*, 2011.

[50] Y. Itoh, H. Nishizaki, X. Hu, H. Nanjo, T. Akiba, T. Kawahara, S. Nakagawa, T. Matsui, Y. Yamashita, and K. Aikawa. Constructing Japanese test collections for Spoken Term Detection. In *Proc. Interspeech*, pages 677–680, 2010.

[51] P. Jourlin, S. E. Johnson, K. S. Jones, and P. C. Woodland. Spoken document representations for probabilistic retrieval. *Speech Commun.*, 32(1-2):21–36, 2000.

[52] N. Kanda, H. Sagawa, T. Sumiyoshi, and Y. Obuchi. Open-vocabulary keyword detection from super-large scale speech database. In *Proc. IEEE Workshop on Multimedia Signal Processing*, pages 939–944, 2008.

[53] T. Kaneko and T. Akiba. Metric subspace indexing for fast Spoken Term Detection. In *Proc. Interspeech*, pages 689–692, 2010.

[54] K. Katsurada, S. Teshima, and T. Nitta. Fast keyword detection using suffix array. In *Proc. Interspeech*, 2009.

[55] S. Kazemian, F. Rudzicz, G. Penn, and C. Munteanu. A critical assessment of spoken utterance retrieval through approximate lattice representations. In *Proc. ACM International Conference on Multimedia Information Retrieval, MIR*, pages 83–88, 2008.

[56] J. Keshet, D. Grangier, and S. Bengio. Discriminative keyword spotting. *Speech Communication*, 51(4):317–329, Apr. 2009.

[57] J. Köhler, M. Larson, F. de Jong, W. Kraaij, and R. Ordelman, editors. *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, 2008.

[58] K. Kukich. Techniques for automatically correcting words in text. *ACM Comput. Surv.*, 24:377–439, December 1992.

[59] M. Larson, T. Beckers, and V. Schlögell. Structuring and indexing digital archives of radio broadcasters. In *Proc. Jahrestagung der Gesellschaft für Informatik, Workshop Integration heterogener, interaktiver Systeme*, pages 83–87, 2005.

[60] M. Larson and S. Eickeler. Using syllable-based indexing features and language models to improve German spoken document retrieval. In *Proc. Eurospeech*, pages 1217–1220, 2003.

[61] M. Larson, S. Eickeler, and K. Joachim. Supporting radio archive workflows with vocabulary independent spoken keyword search. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, 2007.

[62] M. Larson, J. Köhler, F. de Jong, W. Kraaij, and R. Ordelman, editors. *Searching Spontaneous Conversational Speech Workshop at ACM Multimedia*, 2009.

[63] M. Larson, R. Ordelman, F. Metze, F. de Jong, and W. Kraaij, editors. *Searching Spontaneous Conversational Speech Workshop at ACM Multimedia*, 2010.

[64] O. Le Blouch and P. Collen. Improving phoneme-based spoken document retrieval with phonetic context expansion. In *2008 IEEE International Conference on Multimedia and Expo*, pages 1217–1220. IEEE, June 2008.

[65] A. Lee and T. Kawahara. Recent development of open-source speech recognition engine julius. In *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2009.

[66] A. Lee, T. Kawahara, and K. Shikano. Julius - an open source real-time large vocabulary recognition engine. In *Proc. Eurospeech*, pages 1691–1694, 2001.

[67] H.-Y. Lee and L.-S. Lee. Integrating recognition and retrieval with user feedback: A new framework for Spoken Term Detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2010.

[68] W. Macherey and H. Ney. Towards automatic corpus preparation for a German broadcast news transcription system. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 733–736, 2002.

[69] K. Maekawa, H. Koiso, S. Furui, and H. Isahara. Spontaneous speech corpus of Japanese. In *Proc. International Conference on Language Resources and Evaluation, LREC*, pages 947–952, 2000.

[70] S. F. Mail, S. Fitt, and S. Bridge. The pronunciation of unfamiliar native and non-native town names, 1995.

[71] P. Majewski. Syllable based language model for large vocabulary continuous speech recognition of polish. In *Proc. Text, Speech and Dialogue, TSD*, pages 397–401, 2008.

[72] J. Mamou and B. Ramabhadran. Phonetic query expansion for spoken document retrieval. In *Proc. Interspeech*, 2008.

[73] U. Manber and G. Myers. Suffix arrays: a new method for on-line string searches. *SIAM Journal on Computing*, 22:935–948, October 1993.

[74] L. Mangu, E. Brill, and A. Stolcke. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech & Language*, 14(4):373–400, 2000.

[75] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proc. Eurospeech*, pages 1895–1898, 1997.

[76] K. McTait and M. Adda-Decker. The 300k LIMSI German broadcast news transcription system. In *Proc. Eurospeech*, 2003.

[77] T. Mertens and D. Schneider. Efficient subword lattice retrieval for German Spoken Term Detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 4885–4888, 2009.

[78] T. Mertens, D. Schneider, and J. Köhler. Merging search spaces for subword Spoken Term Detection. In *Proc. Interspeech*, pages 2127–2130, 2009.

[79] T. Mertens, D. Schneider, A. B. Naess, and T. Svendsen. Lexicon adaptation for subword speech recognition. In *Proc. IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, pages 562–567, 2009.

[80] T. Mertens, R. Wallace, and D. Schneider. Cross-site combination and evaluation of subword Spoken Term Detection systems. In *Proc. International Workshop on Content-Based Multimedia Indexing, CBMI*, pages 61–66, 2011.

[81] K. Ng. *Subword-based approaches for spoken document retrieval.* PhD thesis, Massachusetts Institute of Technology, 2000.

[82] NIST. The Spoken Term Detection (STD) 2006 evaluation plan.

[83] M. Nußbaum-Thom, S. Wiesler, M. Sundermeyer, C. Plahl, S. Hahn, R. Schlüter, and H. Ney. The RWTH 2009 Quaero ASR evaluation system for English and German. In *Proc. Interspeech*, pages 1517–1520, 2010.

[84] Official Google Blog. *Thanks, YouTube community, for two BIG gifts on our sixth birthday!*, 2011 (accessed September 08, 2011). `http://googleblog.blogspot.com/2011/05/thanks-youtube-community-for-two-big.html`.

[85] S. Olsson. Improved measures for predicting the usefulness of recognition lattices in ranked utterance retrieval. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, pages 7–11, 2007.

[86] Y.-C. Pan, H.-L. Chang, B. Chen, and L.-S. Lee. Subword-based position specific posterior lattices (S-PSPL) for indexing speech information. In *Proc. Interspeech*, pages 318–321, 2007.

[87] Y.-C. Pan, H.-L. Chang, and L.-S. Lee. Analytical comparison between position specific posterior lattices and confusion networks based on words and subword units for spoken document indexing. In *Proc. of IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, pages 677–682, 2007.

[88] S. Parlak and M. Saraclar. Spoken Term Detection for Turkish broadcast news. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 5244–5247, 2008.

[89] J. Pinto, I. Szoke, S. Prasanna, and H. Hermansky. Fast approximate Spoken Term Detection from sequence of phonemes. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, pages 28–33, 2008.

[90] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[91] B. Ramabhadran, A. Sethy, J. Mamou, B. Kingsbury, and U. Chaudhari. Fast decoding for open vocabulary Spoken Term Detection. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies, HLT-NAACL*, pages 277–280, 2009.

[92] T. Rotovnik, M. S. Maucec, B. Horvat, and Z. Kacic. A comparison of HTK, ISIP and Julius in Slovenian large vocabulary continuous speech recognition. In *Proc. Interspeech*, pages 681–684, 2002.

[93] M. Saraclar and R. Sproat. Lattice-based search for spoken utterance retrieval. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies, HLT-NAACL*, 2004.

[94] D. Schneider and J. Köhler. Spoken Term Detection on German speech data. In *Proc. ITG Symposium on Speech Communications*, 2010.

[95] D. Schneider, T. Mertens, M. Larson, and J. Köhler. Contextual verification for open vocabulary Spoken Term Detection. In *Proc. Interspeech*, pages 697–700, 2010.

[96] D. Schneider, J. Schon, and S. Eickeler. Towards large scale vocabulary independent Spoken Term Detection: Advances in the Fraunhofer IAIS Audiomining system. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, pages 34–41, 2008.

[97] D. Schneider, S. Tschöpel, J. Schwenninger, T. Kissels, J. Schon, and J. Köhler. Speech-based citation of tv scenes in social networks. In *Demonstration at IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, 2011.

[98] D. Schneider, T. Winkler, J. Löffler, and J. Schon. Robust audio indexing and keyword retrieval optimized for the rescue operation domain. In *Proc. Mobile Response*, pages 135–142. Springer, 2007.

[99] F. Seide, P. Yu, and Y. Shi. Towards spoken-document retrieval for the enterprise: Approximate word-lattice indexing with text indexers. In *Proc. IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU*, pages 629–634, 2007.

[100] SES ASTRA. *List of TV and radio programmes available in Germany via ASTRA digital satellite.*, 2011 (accessed September 08, 2011). `http://www.astra.de`.

[101] J. Shao, Q. Zhao, P. Zhang, Z. Liu, and Y. Yan. A fast fuzzy keyword spotting algorithm based on syllable confusion network. In *Proc. Interspeech*, pages 2405–2408, 2007.

[102] I. Szoke, M. Fapso, L. Burget, and J. Cernocky. Hybrid word-subword decoding for Spoken Term Detection. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, pages 42–48, 2008.

[103] J. Tejedor, D. Wang, J. Frankel, S. King, and J. Colás. A comparison of grapheme and phoneme-based units for Spanish Spoken Term Detection. *Speech Communication*, 50(11-12):980–991, 2008.

[104] K. Thambiratnam and S. Sridharan. Rapid yet accurate speech indexing using dynamic match lattice spotting. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):346–357, 2007.

[105] S. Tschöpel and D. Schneider. A lightweight keyword and tag-cloud retrieval algorithm for automatic speech recognition transcripts. In *Proc. Interspeech*, 2010.

[106] E. Ukkonen. On-Line Construction of Suffix Trees. *Algorithmica*, 14(3):249–260, 1995.

[107] L. van der Werff and W. Heeren. Evaluating ASR output for information retrieval. In *Searching Spontaneous Conversational Speech Workshop at ACM SIGIR*, pages 13–20, 2007.

[108] D. von Zeddelmann, F. Kurth, and M. Müller. Perceptual audio features for unsupervised key-phrase detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2010.

[109] A. Waibel, H. Yu, M. Westphal, H. Soltau, T. Schultz, T. Schaaf, Y. Pan, F. Metze, and M. Bett. Advances in meeting recognition. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies, HLT-NAACL*, pages 11–13, 2001.

[110] R. Wallace, B. Baker, R. Vogt, and S. Sridharan. The effect of language models on phonetic decoding for Spoken Term Detection. In *Proc. Searching Spontaneous Conversational Speech Workshop at ACM Multimedia*, pages 31–36, 2009.

[111] R. Wallace, R. Vogt, B. Baker, and S. Sridharan. Optimising figure of merit for phonetic Spoken Term Detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2010.

[112] R. Wallace, R. Vogt, and S. Sridharan. A phonetic approach to the 2006 NIST Spoken Term Detection evaluation. In *Proc. Interspeech*, pages 2385–2388, 2007.

[113] R. Wallace, R. Vogt, and S. Sridharan. Spoken term detection using fast phonetic decoding. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 4881–4884, 2009.

[114] D. Wang, S. King, J. Frankel, and P. Bell. Stochastic pronunciation modelling and soft match for out-of-vocabulary Spoken Term Detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 5294–5297, 2010.

[115] M. Wechsler. *Spoken Document Retrieval based on Phoneme Recognition.* PhD thesis, ETH Zürich, 1998.

[116] J. G. Wilpon, L. R. Rabiner, C. H. Lee, and E. R. Goldman. Automatic recognition of keywords in unconstrained speech using hidden markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(11):1870–1878, 1990.

[117] T. Winkler and R. Bardeli. An integrated approach for a robust command and control application on the motorcycle. In *Proc. International Conference Speech and Computer, SPECOM*, pages 464–469, 2009.

[118] P. Wittenburg, P. Trilsbeek, and P. Lenkiewicz. Large multimedia archive for world languages. In *Searching Spontaneous Conversational Speech Workshop at ACM Multimedia*, pages 53–56, 2010.

[119] P. Woodland. Speaker adaptation for continuous density HMMs: A review. In *Proc. ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition*, pages 11–19, 2001.

[120] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book.* Cambridge University Engineering Department, 2006.

[121] P. Yu and F. Seide. Fast two-stage vocabulary independent search in spontaneous speech. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 481–484, 2005.

[122] P. Yu, D. Zhang, and F. Seide. Maximum entropy based normalization of word posteriors for phonetic and LVCSR lattice search. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2006.

[123] F. Zimmerer, M. Scharinger, and H. Reetz. When beat becomes house: Factors of word final /t/-deletion in german. *Speech Commun.*, 53:941–954, July 2011.