Automating the Fact-Checking Task: Challenges and Directions

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

von
Diego Nascimento Esteves da Silva
aus
Rio de Janeiro, Brasilien

Bonn, März 2019

Dieser Forschungsbericht wurde als Dissertation von der Mathematisch-Naturwissenschaftlichen Fakultät der Universität Bonn angenommen und ist auf dem Hochschulschriftenserver der ULB Bonn http://hss.ulb.uni-bonn.de/diss_online elektronisch publiziert.

1. Gutachter:	Prof. Dr. Jens Lehmann
2. Gutachterin:	Prof. Dr. Sören Auer
Tag der Promotion:	29.05.2019
Erscheinungsjahr:	2019

Abstract

In recent years, misinformation has caused widespread alarm and has become a global concern, given the negative impact placed on society, democratic institutions and even computing systems whose the primary objective is to serve as a reliable information channel, e.g., Knowledge Bases (KBs). The proliferation of fake news has a wide range of characteristics and different motivations. For instance, it can be produced unintentionally (e.g., the creation process of KBs which is mostly based on automated information extraction methods, thus naturally error-prone) or intentionally (e.g., the spread of misinformation through social media to persuade). Thus, they differ considerably in complexity, structure and number of arguments and propositions. To further exacerbate this problem, an ever-increasing amount of fake news on the Web has created another challenge to drawing correct information. This huge sea of data makes it very difficult for human fact checkers and journalists to assess all the information manually. Therefore, addressing this problem is of utmost importance to minimize real-world circumstances which may provoke a negative impact on the society, in general. Presently *Fact-Checking* has emerged as a branch of natural language processing devoted to achieving this feat. Under this umbrella, *Automated Fact-Checking* frameworks have been proposed to perform claim verification. However, given the nature of the problem, different tasks need to be performed, from natural language understanding to source trustworthiness analysis and credibility scoring. In this thesis, we tackle the problem of fake news and underlying challenges related to the process of estimating the veracity of a given claim, discussing challenges and proposing novel models to improve the current state of the art on different sub-tasks. Thus, besides the principal task (i.e., performing automated fact-checking) we also investigate: the recognition of entities on noisy data and the computation of web site credibility. Ultimately, due to the challenging nature of the automated fact-checking task - which requires a complex analysis over several perspectives - we also contribute towards reproducibility of scientific experiments. First, we tackle the named entity recognition problem. We propose a novel multi-level approach named HORUS which - given an input token - generates heuristics based on computer vision and text mining techniques. These heuristics are then used to detect and classify named entities on noisy data (e.g., The Web). Second, we propose WebCred, a novel model to compute the credibility score of a given website, regardless of dependency on search engine results, which is a limiting factor when dealing with real scenarios. WebCred does not require any third-party service and is 100% open-source. Third, we conduct several empirical evaluations and extend DeFacto, a fact-checking framework initially designed to verify English claims in RDF format. DeFacto supports both structured claims (e.g., triple-like) as well as complex claims (i.e., natural language sentences). Last, but not least, we consistently contributed towards better reproducibility research tools, methods, and methodologies. We proposed ontologies (MEX, ML-Schema) and tools (LOG4MEX, MEX-Interfaces, Web4MEX, WASOTA) which turned into state of the art for better reproducibility of machine learning experiments, becoming part of a global W3C community.

Acknowledgements

First of all, I thank God for giving me the necessary strength and encouragement during all the challenging and stressful moments in completing this thesis. The thesis was written within the Smart Data Analytics (SDA) research group, led by Prof. Dr. Jens Lehmann. I was fortunate to have Prof. Lehmann as an adviser during the development of this thesis, which I thank for granting me the freedom to develop and pursue my research ideas that have lead to this work. The realization of this work would not have been possible without the indirect support of Prof. Dr. Elmar Langetepe. I will always be grateful to him. I also appreciate the great opportunity offered to me by Prof. Dr. Maria Claudia Reis Cavalcanti (Yoko), who encouraged me to pursue a Ph.D. abroad. I also want to thank Dr. Sebastian Hellmann for inviting me to become a member of the Agile Knowledge Engineering and Semantic Web (AKSW) group at the University of Leipzig, where SDA began to take shape. My sincere gratitude to Prof. Dr. Julio Cesar Duarte for his continuing partnership in research, which impacted in a successful project developed within the scope of this thesis. I am delighted to be part of SDA and have had the opportunity to work with so many smart people. Thanks to Dr. Giulio Napolitano and Dr. Anisa Rula for working together in challenging research publications. Special thanks go to Mehdi Ali, Monish Dubey, Gëzim Sejdiu, Hamid Zafar, Debanjan Chaudhuri, Harsh Thakkar, Denis Lukovnikov, Nilesh Chakraborty, Gaurav Maheshwari, Priyansh Trivedi and Debayan Banerjee. I would also like to extend my deepest gratitude to my colleagues from Leipzig, most notably the Brazilian mates and Tommaso Soru. It was an honor to meet you on this journey. In especial, I would like to sincerely thank Diego Moussallem, who contributed to various research milestones within the framework of reproducible research and for having become a true friend. I am grateful to have had the opportunity to discuss research topics with Dr. Amrapali Zaveri and Dr. Pablo N. Mendes. I extend my gratitude to Prof. Dr. Henrique Lopes and Gil Rocha for the invitation to collaborate with the University of Porto. I would also like to extend my deepest gratitude to my family. To my wife for supporting and motivating me to complete this thesis and for enriching my life beyond my scientific endeavors. Thank you for understanding stressful times and for fully supporting me through the journey. To my parents, Nelson and Lourdes to always encourage me to chase my dreams. They have always sacrificed their lives to raise me to be a better person. To my lovely and blessed sister Luane (*in memoriam*) who taught me life values which can not be learned in courses or through books. To my mother-in-law, Liliane, for your great support and love. I want to thank Fundação CAPES for the scholarship which supported great part of this research. Finally, my sincere gratitude to Microsoft and BabelNet to have offered me infrastructure grants. Further funding for travel expenses was granted by the DAAD, SDA Research and Amazon.

This Ph.D. thesis is dedicated to my wife, Bianca Esteves and my daughter, Bella Esteves.

Contents

CHAPTER 1

Introduction

With billions of individual pages on the web providing information on almost every conceivable topic, we should have the ability to collect factual information (fact extraction) to confirm a broad range of claims. However, only a small fraction of this information is contained in structured sources (e.g., Knowledge Bases), which makes the verification process time-consuming as the experts have to carry out several search processes and must often read several documents. The fact-checking process of collecting evidence to validating input claims also becomes relevant to knowledge-base population (KBP), since it mostly relies on automatic or semi-automatic information extraction methods. The process of creating and managing large-scale knowledge bases (KBs) has been the key to success of many applications [1–3]. However, if the quality of such KBs is insufficient, this poses a significant obstacle to the uptake of data consumption applications at large-scale [4].

Yet, there is another problem that has become the focus of a lot of recent research: *misinformation* coming from unreliable sources [5, 6]. The problem of veracity estimation is also recognized as one of the key challenges in building and maintaining large KBs [7]. Therefore, fact extraction and fact validation algorithms are also very important to the knowledge base construction process. Fact checking is studied in different communities (e.g., journalism, machine learning, semantic web) and historically has being referred under various names such as *truth finding*, *fact-checking*, *claim verification* and *trustworthiness* [5], for instance. Still, the research community still lacks a more granular taxonomy of the related tasks. Broadly speaking, *fact-checking* is a complex multi-disciplinary topic. It encompasses the intersection of different research fields, with the emphasis being placed on Natural Language Processing (NLP), machine learning (ML) and semantic web (SW). For instance, the following sub-tasks in NLP (i.e., mostly related to machine reading and comprehension) can be performed in a fact-checking framework: (1) information extraction methods (e.g. Named Entity Recognition and Dependency Parsing), (2) Natural Language Generation, (3) Natural Language Understanding, (4) Argumentation Mining and Textual Entailment and (5) Trustworthiness of Information Sources. A fact-checking system typically takes a statement as input and then tries to find evidence to support of refute this given *claim* by searching for textual information (*proofs*) in different data sources. The task can be performed over free text (e.g., a claim existing in a web site) or structured information (e.g., RDF triples from a Knowledge Base). Typically the output of a fact-checking framework is a *confidence score* that represents the level of trustworthiness w.r.t. the input claim. The task can

be sub-divided as follows:

1. **Plausibility Estimation**: indicates whether the *claim* is plausible (e.g., *"Barack Obama", "birth place", "NYC"*) or not (e.g., *"Barack Obama", "birth place", "Blue"*). However, the correctness of the input claim is not relevant.

2. **Validation**: indicates whether the *claim* is correct (*true*) or not (*false*).

3. **Ranking**: considering a number of input *claims* sharing the same *subject* and *predicate*, the task is to ordering them w.r.t. their relevance relying on common-sense rules (e.g., *"Barack Obama", "profession", "politician"* should be higher ranked then *"Barack Obama", "profession", "lawyer"*, since he is mostly known as a politician.). In general, all claims existing in the set are considered *true*.

In this thesis, we target to address the fact-validation task along with its challenges.

1.1 Motivation

With the increase in false fact circulation across different social media platforms, it has become pertinent to validate the claims and statements released online [8, 9]. False statements are mostly spread through the Web, but they can also exist in Knowledge Bases due to automatic database population procedures [10, 11].

Over news on the web, false claims expressed in natural language are often referred as *fake-news* or *junk-news*, for instance. These have gained importance in the last few years, mainly in the context of electoral activities in North America and Western Europe. Celebrity death hoaxes, misinformation about political leaders, political statements etc. are some of the most common types of fake-news types[1]. The deceptive information can in general can be categorized as *misinformation* (false or misleading information) and *disinformation* (false information that is purposely spread to deceive people) [12]. These false facts (or rumors), in the past, have led to situations like stock price drops and large scale investments [13]. Though social media platforms are the most common breeding grounds for fake-news, it sometimes finds its way into the mainstream media too [14]. The most effective methodology to consume news found on unconventional news media is to check multiple resources. But most news-media consumers cannot effectively validate the accuracy or may enjoy partisan views of the news [14]. In such cases, these media consumers start an uncontrolled reaction of consuming and sharing of false facts that spreads faster an epidemic in a demographic and sometimes throughout the world.

False claims stated in structured information sources (e.g., Knowledge Bases (KBs)) also are of utmost importance for many state of the art applications both in scientific research and in-use industry projects. They are mainly designed to store complex structured information, which depicts facts about the world. For instance, DBpedia, YAGO, Freebase, Wikidata and Google Knowledge Graph are examples of successful KBs projects. However, only a small fraction of the world information is contained in these structured sources. The creation process is mostly

[1] https://www.statista.com/topics/3251/fake-news/

based on automated information extraction (IE) methods which are error-prone. Therefore, the quality of knowledge is of high importance. To this end, fact extraction and fact verification algorithms are of utmost importance to the process of knowledge base population. Moreover, in this respect, ranking triples (fact ranking) based on their relevance is another important task. For instance, a person could have more than one profession or nationality, but among different possibilities, one of them often is more relevant.

To deal with the above-introduced challenges, fact-checking algorithms for KBs have been proposed and are classified in three different groups: *validation*, *ranking* and *plausibility estimation*. In this thesis, we aim at exploring different techniques to automatize the validation task, studying current limitations and developing novel solutions to overcome current state of the art results.

1.2 Problem Specification and Challenges

In general, the task of automated fact-checking is considered as one of the most challenging tasks in the natural language processing (NLP) field [15] requiring a multidisciplinary effort [16]. In the scope of automated fact-checking frameworks, Table 2.1 presents an overview of the features of some state-of-the-art approaches for fact-checking over KBs. The decentralized and autonomous nature of the Web along with the complexity of human languages allow for multiple representations of the same information, which brings multiple challenges to solve the task. At the conceptual level, we face (1) a knowledge retrieval problem, i.e., “search and integrate pieces of knowledge about a given claim spread on several documents”. Also, (2) a natural language understanding problem, i.e., "to understand and reason about diverse excerpts of texts obtained from different sources". Furthermore, (3) a credibility challenge, i.e., "distinguish trustworthy from doubtful information sources".

In the following section, we discuss the challenges that need to be addressed to produce an automated fact-checking framework.

1.2.1 Challenge 1: Detecting Entities on Short-Text

The first challenge to overcome is the process of recognizing entities in a free-format text, which is very challenging [17]. Named Entity Recognition (NER) is an important step in most of the natural language processing (NLP) pipelines, including automated fact-checking frameworks. However, most of the state-of-the-art solutions to detect objects in *short and informal text* (e.g., microblogs) still are not able to perform similarly to frameworks designed explicitly to *formal text* (e.g., newswire) [18], which naturally imposes a barrier on developing high-performance models. This is due to the lack of implicit linguistic formalism (e.g. punctuation, spelling, spacing, formatting, unorthodox capitalisation, emoticons, abbreviations and hashtags) [17, 19–21]. Furthermore, the lack of external knowledge resources is an important gap in the process regardless of writing style [22]. Thus, to perform language understanding we need an algorithm able to perform the task in a noisy environment such as the Web.

1.2.2 Challenge 2: Computing Trustworthiness of Web pages

Apart from detecting entities, deciding which information provider should have more relevance is yet another challenging task. Computing the trustworthiness of sources of information is a crucial step to enhance the quality of fact-checking algorithms. However, existing trustworthiness indicators are not freely available anymore[2], including Google Pagerank. The solution proposed by Nakamura [23] appears as an *open-source* alternative to compute web source trust indicator values. They developed a prototype for enhancing the search results provided by a search engine based on trustworthiness analysis. However, it is a graph-based model which calculates the relevance among retrieved sources, and not a global trustworthiness indicator. For instance, given a certain domain (e.g., `bbc.com`), how can we define its credibility measure. Moreover, how can we automatically infer that an information coming from `bbc.com` should be more credible than a claim obtained a random web blog?

1.2.3 Challenge 3: Automating the Fact-Checking Task

Fact-checking itself is especially hard due to the complexity inherent in creating and connecting logical arguments that are used to either support or refute a given *claims* [24]. This is a basis to communicate and defend opinions (or *claims* within this context), to understand new problems and to perform scientific reasoning [25]. Thus, more powerful methods are required (argumentation mining) other than standard information extraction methods (e.g., part-of-speech tagging, named entity recognition and dependency parsing) for better understanding text structures and relations among entities. Argumentation mining methods pose as next generation of algorithms to processing free-format text in natural language to recover inferential structure [26]. However, most of the proposed works are of a theoretical nature, lacking more real-world applications.

Finally, the dependency on relation extraction methods tends to restrict the comprehensiveness of fact-checking algorithms [27], since the verbalization of predicates is a crucial step to the information extraction process. This process allows to generate distinct verbalizations which have semantically the same meaning as the input claim. For instance, the relation "marriage" can be represented by the following verbalizations: "*X-wife-Y*", "*X-husband-Y*" as well as "*X-spouse-Y*". Traditional approaches predict relations within some fixed and finite target schema [28]. The strategies are described as follows: (1) hard-coded verbalization and rules (fixed or ontology-based), which naturally restricts scalability; (2) standard machine learning approaches which require manual annotation; (3) use distant supervision methods which very often have a sub-optimal performance, since the method relies on the availability of a large database that has the desired schema; or (4) use external linguistic corpora (e.g. lexical databases) to obtain similar words (e.g. synonyms) to a given *predicate*. It can be observed that these methods are rather self-limited and hence the verbalizations generated are either of a low quality or have low precision and/or recall.

[2] e.g., `https://www.alexa.com/siteinfo`

1.2.4 Challenge 4: Reproducibility Issues

Every scientific experiment should be replicable, specially experiments performed in an attempt to find answers to complex problems, such as fact-checking. The need for reproducibility is increasing dramatically as data analyses become more complex, involving more sophisticated and complex experimental configurations. Reproducibility allows for people to focus on the actual content of a data analysis, instead of on superficial details reported in a written summary. Overall, scientific experiments are still hard to be reproduced, because of these complex configurations, because of lack of time to properly report scientific experiments.

1.3 Research Questions

Based on the main problem and associated challenges described in Section 1.2, we formulate four research questions in the scope of this thesis. Each research question provides a solution for a correspondent challenge.

> **RQ1**: Can images along with news improve the performance of the named entity recognition models on noisy text?

To answer **RQ1** we fully explore existing techniques to detect and extract named entities from texts, in the context of the Web. The challenges to perform the task in such context are manifolds, from the lack of grammatical rules to the noisy existing in the Web. Our hypothesis is that images and related documents can help at boosting the task. In the scope of this thesis we propose a novel methodology to extract heuristics for potential named entities using computer vision and text mining techniques. The results of the research question **RQ1** allow us to address challenge three **C1**.

> **RQ2**: How to calculate a credibility score for a given information source?

To address **RQ2**, we evaluate state-of-the-art web credibility approaches that can be used to compute the trustworthiness of a given information source. We further propose a novel model that automatically extracts source reputation cues and computes a credibility factor based on metadata extracted from the source-code. The results of the research question **RQ2** allow us to address challenge two **C2**.

> **RQ3**: How to determine the veracity of a given claim?

To answer **RQ3** we investigate the state-of-the-art approaches for fact-checking, and detail steps of the automation process. Particularly we analyze and evaluate the complex fact-checking task

within three perspectives: (a) fact-checking over RDF triples (b) fact-checking over complex claims and (c) evidence extraction methods for fact-checking. The results of the research question **RQ3** allow us to address challenge three **C3**.

RQ4: Are existing reproducible research methods sufficient to enable reproducibility?

Reproducibility is a serious problem in scientific experiments. To address **RQ4** we investigate several approaches to support the representation of machine learning experiments, from ontologies and libraries to support the experimental process to repositories to storage the metadata. We highlight the challenges and trade-offs to offer the ideal scenario and define new standards which are more suitable to bridge this gap. As consequence, we have funded a new W3C group to guide the community towards better representation of scientific experiments. The results of the research question **RQ4** allow us to address challenge four **C4**.

1.4 Thesis Overview

To prepare the reader for the rest of the document, in this section we present an overview of our main contributions, the research areas investigated by this thesis, the references to scientific publications covering this work, and an overview of the thesis structure.

1.4.1 Contributions

The contributions of this thesis are cross disciplinary around the *fact-checking* topic involving Automated Fact-Checking, Named Entity Recognition, Web Credibility and Reproducible Research fields. Firstly, as a focus of this thesis, we fully exploited the domain of fact-checking in both simple as well as complex claims. Secondly, we advanced methods in necessary areas of domain to improve the fact-checking task, namely: Information Retrieval, Trustworthiness and Named Entity Recognition. Finally, we extensively contributed towards better reproducible experiments in the machine learning domain. Figure 1.1 shows the overall contributions across several levels of a scientific experiment.

Figure 1.2 shows the four main contributions of this thesis.

- ***Contrib. 1:*** *Named Entity Framework designed for Noisy Data*; to address the problem of detecting and classifying named entities on noisy data, we develop HORUS, a novel methodology that integrated computer vision and text mining techniques to perform NER. HORUS implements a three-fold approach for both detecting and classifying named entities. First it extracts image-based features and then in parallel performs a set of text classifications over extracted documents. Using the produced heuristics, it then performs a final classification using both decision trees methods as well as concatenating the vectors and using more powerful architectures such as B-LSTM. Furthermore, we demonstrate that this simple idea is able to outperforms state of the art without encoded rules. An empirical

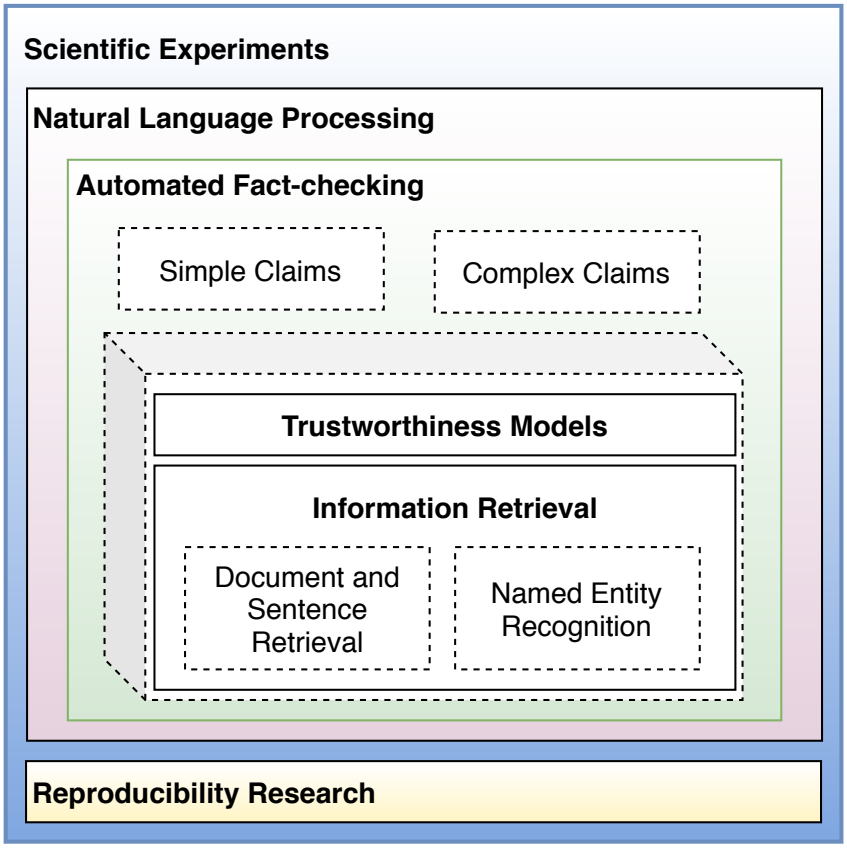


Figure 1.1: **Academic Contributions:** Natural Language Processing and Reproducible Research were the principal areas of research

evaluation assesses the effectiveness of HORUS for the detection of named entities on noisy data. The experiments are executed on *Ritter*, *WNUT-15*, *WNUT-16* and *WNUT-17* answering research question **RQ1**.

- ***Contrib. 2:*** *A Trustworthiness Framework for the Web*; defining the level of trustworthiness of a given information source is crucial to perform fact-checking. We propose a novel web credibility model based on a concept we call HTML2Seq. HTML2Seq the transformation of HTML tags into sequences of integers. We thus evaluate websites w.r.t. website metadata. Thus, we present WebCred, a web credibility model that performs trustworthiness analisys based on metadata inspection through the transformation of HTML tags into sequential vectors. We empirically study the performance of WebCred with respect to state-of-the-art methods. The observed results show that WebCred is very competitive in terms of precision and recall with respect to existing methods, with the advantage that it works for any given website as well as it is completely open-source. Experimental results answer our research question **RQ2**.

- ***Contrib. 3:*** *An Automated Fact-Checking Framework*; In order to tackle the fact-checking problem, an automated fact-checking framework, dubbed DeFacto, has been designed and extended to also perform claim verification over natural language. An empirical evaluation of the quality of the proposed framework in comparison with triple-based approaches

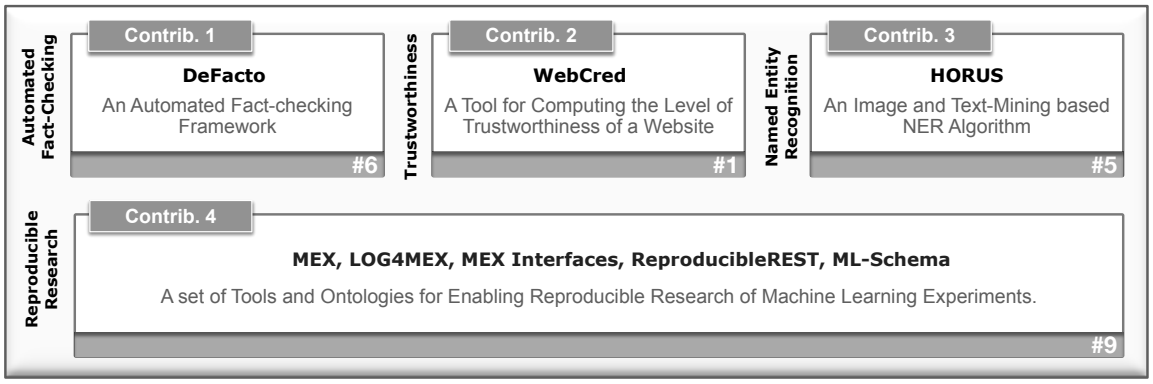


Figure 1.2: **Contributions:** Four are the main contributions of this thesis including: (1) an automated fact-checking framework; (2) a web credibility model; (3) a named entity recognition algorithm for noisy data; and (4) a set of tools and ontologies for enabling reproducible research on machine learning experiments. In the bottom box, the number of related publications to a specific area of research (#) during the development of this PhD thesis.

indicates that DeFacto approach accurately perform fact-validation for triple-like claims. For complex claims (unstructured claims), DeFacto ranks among state-of-the-art solutions when evaluated in a very selective fact-checking challenge, giving an answer to our research question **RQ3**.

- ***Contrib. 4:*** *Enabling Reproducible Research* During the development of this thesis, we faced a very common challenge when designing and executing scientific experiments: the ability to manage and extract outcomes as well as turning them open to the scientific community. The lack of patterns, standards and tools are the keystone to enable reproducible research. We joint efforts to existing attempts and created a W3C working group (ML-Schema) which aims to define standards to achieve this goal. Through the development of this thesis we proposed several tools and vocabularies towards better interpretability of machine learning experiments, answering our research question **RQ4**.

1.4.2 List of Publications

The development of this thesis has led to multiple scientific publications and three different open-source projects: DeFacto, HORUS and MEX (enclosing 5 related projects). Appendix A contains the complete list of 30 publications. However, the content of this thesis is based on the aforementioned open-source projects and the following 23 related scientific publications:

- *Journal Articles*:

  1. Gustavo Publio, Agnieska Ławrynowicz, Larisa Soldatova, Panče Panov, **Diego Esteves**, Joaquin Vanschoren and Tommaso Soru. *ML-Schema: An interchangeable format for description of machine learning experiments* In Journal of Web Semantics (JWS), 2019 - submitted;.

2. **Diego Esteves**, Anisa Rula, Aniketh Reddy, Jens Lehmann. *Toward Veracity Assessment in RDF Knowledge Bases: An Exploratory Analysis.* In Journal of Data and Information Quality (JDIQ), 2018.

3. Anisa Rula, Matteo Palmonari, Simone Rubbinaci, Axel Ngonga, Jens Lehmann, **Diego Esteves**. *TISCO: Temporal Scoping of Facts.* In Journal of Web Semantics (JWS), 2018.

4. Daniel Gerber, **Diego Esteves**, Jens Lehmann, Lorenz Bühmann, Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, René Speck. *DeFacto - Temporal and Multilingual Deep Fact Validation.* In Web Semantics: Science, Services and Agents on the World Wide Web (SWJ), 2015.

- *Conference and Workshop Papers*:

5. **Diego Esteves**, Asja Fischer, Piyush Chwala, Saad Khan, Jens Lehmann and Rafael Peres. *Beyond Lexical Features: Named Entity Recognition on Noisy Data through the Web.* In Proceedings of the 2019 Conference of the Association for Computational Linguistics (ACL'19) - submitted;

6. Piyush Chwala and **Diego Esteves** *Automating Fact-Checking: Why is it so difficult?* In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2019) - submitted;

7. **Diego Esteves**, Aniketh Janardhan Reddy, Piyush Chawla and Jens Lehmann. *Belittling the Source: Trustworthiness Indicators to Obfuscate Fake News on the Web.* In Proceedings of Fact Extraction and VERification (FEVER) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

8. Aniketh Janardhan Reddy, Gil Rocha and **Diego Esteves**. *DeFactoNLP: Fact Verification using Entity Recognition, TFIDF Vector Comparison and Decomposable Attention.* In Proceedings of Fact Extraction and VERification (FEVER) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

9. Gustavo Correa Publio, **Diego Esteves**, Agnieszka Ławrynowicz, Panče Panov, Larisa Soldatova, Tommaso Soru, Joaquin Vanschoren and Hamid Zafar. *ML-Schema: Exposing the Semantics of Machine Learning with Schemas and Ontologies.* In Proceedings of the 2nd Reproducibility in Machine Learning Workshop (MLTRAIN@RML) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Stockholm Sweden, 2018.

10. Rafael Peres, **Diego Esteves** and Gaurav Maheshwari. *Bidirectional LSTM with a Context Input Window for Named Entity Recognition in Tweets.* In Proceedings of the 9th International Conference on Knowledge Capture (K-CAP 2017)

11. Julio Cesar Duarte, Maria Claudia Reis Cavalcanti, Igor de Souza Costa and **Diego**

Esteves. *An Interoperable Service for the Provenance of Machine Learning Experiments.* In Proceedings of the International Conference on Web Intelligence (WI2017)

12. Agnieszka Lawrynowicz, **Diego Esteves**, Pance Panov, Tommaso Soru, Sašo Dzeroski and Joaquin Vanschoren. *An Algorithm, Implementation and Execution Ontology Design Pattern.* In Proceedings of Advances in Ontology Design and Patterns 32 (2017) - co-located with the 15th International Semantic Web Conference (ISWC 2016) in Kobe, Japan.

13. **Diego Esteves**, Pablo N. Mendes, Diego Moussallem, Julio Cesar Duarte, Amrapali Zaveri, Jens Lehmann and Ciro Baron Neto, Igor Costa and Maria Claudia Cavalcanti. *MEX Interfaces: Automating Machine Learning Metadata Generation.* In Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016).

14. **Diego Esteves**, Diego Moussallem, Ciro Baron Neto, Tommaso Soru, Ricardo Usbeck, Markus Ackermann and Jens Lehmann. *MEX Vocabulary: A Lightweight Interchange Format for Machine Learning Experiments.* In Proceedings of the 11th International Conference on Semantic Systems (SEMANTiCS 2015), 15-17 September 2015, Vienna, Austria.

15. **Diego Esteves**, Diego Moussallem, Tommaso Soru, Ciro Baron Neto, Jens Lehmann, Axel-Cyrille Ngonga Ngomo and Julio Cesar Duarte. *LOG4MEX: A Library to Export Machine Learning Experiments.* In Proceedings of the International Conference on Web Intelligence (WI'17), Leipzig, Germany.

16. **Diego Esteves**, Rafael Peres, Jens Lehmann and Giulio Napolitano. *Named Entity Recognition in Twitter using Images and Text.* In Proceedings of the 3rd International Workshop on Natural Language Processing for Informal Text (NLPIT 2017), Rome, Italy.

17. Ciro Baron Neto, Dimitris Kontokostas, Gustavo Publio, **Diego Esteves**, Amit Kirschenbaum and Sebastian Hellmann. *IDOL: Comprehensive & Complete LOD Insights.* In Proceedings of the 13th International Conference on Semantic Systems (SEMANTiCS 2017), 11-14 September 2017, Amsterdam, Holland.

- *Posters and Demos*:

18. **Diego Esteves**. *Named Entity Recognition on Noisy Data using Images and Text.* In Proceedings of The 4th Workshop on Noisy User-generated Text (W-NUT) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

19. René Speck, **Diego Esteves**, Jens Lehmann, Axel-Cyrille Ngonga Ngomo. *DeFacto - A Multilingual Fact Validation Interface.* In Proceedings of the 14th International Semantic Web Conference (ISWC 2015, Semantic Web Challenge)

20. **Diego Esteves**, Diego Moussallem, Ciro Baron Neto, Jens Lehmann, Maria Claudia Cavalcanti, Julio Cesar Duarte. *Interoperable Machine Learning Metadata using*

MEX. In Proceedings of the 14th International Semantic Web Conference (ISWC 2015, Posters & Demos)

21. Sandro A. Coelho, Diego Moussallem, Gustavo C. Publio, **Diego Esteves**. *TANKER: Distributed Architecture for Named Entity Recognition and Disambiguation.* In Proceedings of the Posters and Demos Track of the 13th International Conference on Semantic Systems - co-located with the 13th International Conference on Semantic Systems (SEMANTiCS 2017), Amsterdam, The Netherlands, September 11-14, 2017.

22. Ciro Baron Neto, **Diego Esteves**, Tommaso Soru, Diego Moussallem, Andre Valdestilhas and Edgard Marx. *WASOTA: What Are the States of the Art?* In Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016), 12-15 September 2016, Leipzig, Germany (Posters & Demos)

The following is the complete list of preprints submitted during the development of this Ph.D. thesis.

- *Technical Reports*:

23. **Diego Esteves**, Agnieszka Lawrynowicz, Pance Panov, Larisa Soldatova, Tommaso Soru and Joaquin Vanschoren. *ML-Schema Core Specification.* In Technical report, World Wide Web Consortium (W3C), 2016.

The final Ph.D. contributions are summarized as follows:

Ph.D. Statistics	From 07.2014 to 12.2018 (4.5 Years)			
	Accepted	**In Rev.**	**Total**	**Thesis Related**
Journal	3	1	4	4 (100%)
Conference/Workshop	14	1	15	12 (80%)
Poster/Demo	7	0	7	5 (71%)
Arxiv	3	0	3	1 (33%)
Technical Report	1	0	1	1 (100%)
Nr. of Publications	**28**	**02**	**30**	**23 (76%)**

1.5 Thesis Structure

The remainder of this thesis is comprised by seven chapters organized as follows:

- *Chapter 2 - Related Work;* provides the state-of-the-art approaches related to this thesis. Firstly, We give a complete view of the state-of-the-art approaches to fact-checking. Secondly, web credibility models are introduced and discussed. Thirdly, we progress to discuss the challenging task of named entity recognition on noisy data. Finally, we

introduce challenges on enabling reproducible research.

- *Chapter 3 - Named Entity Recognition on Noisy Data;* presents a novel named entity recognition for microblogs and informal text dubbed HORUS, which is crucial information extraction task to perform fact-checking.

- *Chapter 4 - Web Trustworthiness;* presents our novel credibility model (WebCred) for detecting trust and non-trust websites. We show limitations of existing approaches and how this model can enhance the fact-checking task.

- *Chapter 5 - Automated Fact-Validation;* presents DeFacto, an automated fact-checking framework for fact-validation. It is part of the final pipeline of the automation process. DeFacto accepts both structured as well as non-structured claims as input and can be customized according to the use case and domain.

- *Chapter 6 - Reproducible Research;* contextualizes reproducible issues in scientific experiments, shedding light on current limitations, difficulties and proposing novel solutions to make experiment more accessible and reproducible.

- *Chapter 7 - Conclusion and Future Direction*; finalizes this thesis with a summary of our results and main contributions to the complex problem of automating fact-checking, and defines the future directions of the research work.

CHAPTER 2

Related Work

2.1 Automated Fact-checking

Fact-checking is a relatively new research area which focuses on classifying (and/or assigning scores) a given statement [29]. In general, the task of automated fact-checking can be considered as one of the most challenging tasks in natural language processing (NLP). Apart from designing *trustworthiness* indicators associated with sources of information, the task is especially hard due to the complexity inherent in creating and connecting logical arguments. This is a basis to communicate and defend opinions (or *claims* within this context), to understand new problems and to perform scientific reasoning [30]. Thus, *argumentation mining* methods pose as state of the art solution for better understanding text structures and relations among entities, i.e., processing raw text in natural language to recover inferential structure [26]. However, it is still a challenging task and most of the proposed works are of a theoretical nature, lacking useful real-world applications.

We can divide the task in: fact-checking over structured (e.g., triples) and unstructured claims (i.e., news). With respect to structured fact-checking, we can perform three different tasks: *validation*, *plausibility* or *ranking*. Table 2.1 presents an overview of the features of some state of the art approaches. Verification of unstructured claims generally involves the design of neural architectures.

In the following, we present recent work in each area:

2.1.1 Triple Plausibility Estimation

Plausibility assessment of triples is another related problem. It deals with the measurement of the plausibility of a certain *subject type* being linked to a certain *object type* through a given *predicate*. It could be seen as a prior task to *Triple Veracity Assessment*. Hong et al [38] propose `PAUST`, a three-phase system which determines the plausibility of a triple using both DBpedia and Wikipedia as sources of information. Given a set of test triples, `PAUST` first generates unlabelled training triples by changing the subject, object, predicate or a combination of the attributes

Fact-Checking Frameworks							
System	**Counter Evidences**	**Real-Time Source**	**Source Trust.**	**Reliance on SEs**	**Nr. Supported Predicates**	**Predicate Expansion**	**Open Source**
Triple Validation							
[31] DeFacto	Yes	No	Yes	Yes	10	Library	Yes
[32] OpenEval	No	No	No	Yes	Any (upon training)	Keywords	No
[33] KnowItAll	No	No	No	Yes	Many (ontology)	Patterns	No
Triple Ranking							
[34] Bast et al.	No	No	No	No	2	No	No
[35] Bokchoy	No	No	No	No	2	No	Yes
[36] Cress	No	No	No	No	2	No	Yes
[37] Goosefoot	No	No	No	No	2	Synonym-based	Yes
Triple Plausibility							
[38] PAUST	No	No	No	No	Any (ext. resources)	Lexical DBs	No

Table 2.1: Features of triple assessment frameworks

above of the test triples. These changes are made in such a way that subjects, objects or predicates are replaced with similar subjects, objects and predicates respectively. The similarity between various entities is determined using features extracted from DBpedia. WordNet [39], NOMLEX [40] and PreDic [41] are used to determine similarity between predicates. In the second phase, the unlabelled training triples are labeled as plausible or not plausible using the Wikipedia sentence corpus. Using statistical hypothesis testing, `PAUST` assigns a value between 0 and 1 to a triple denoting the distance between the test triple and the training triple. In the final phase, `PAUST` determines the plausibility of the test triple by examining the k-nearest neighbors of the test triple. If a majority of the nearest neighbors are plausible, then the test triple is also determined to be plausible. Otherwise it is labeled not plausible. When the subjects, objects and predicates of many triples belong to the same subject, object and predicate concepts respectively, all such triples are given the label which is the majority among such triples.

2.1.2 Triple Validation

Triple Validation is a task in which an input triple is classified as positive or negative, i.e. *true* or *false*. Thus, the process is often performed using supervised classification techniques. There are many approaches and strategies for validating the facts represented by the triples. First, one can search for the input triple on the web, and then apply some method to decide if the triple is true based on the features extracted from the search results [42]. The keywords used while querying the search engines are derived from the *subject* and the *object* of the triple. The web pages retrieved are then ranked based on the calculated values of different features. After determining the features values and ranks of the search results, the system finally outputs its classification, saying whether the input triple is true or false. These approaches often apply some method to obtain natural language representations (NLRs) of *predicates* (e.g., hard-coded NLRs, string similarity measures or distant supervision techniques). A different solution is to apply supervised knowledge extraction on the web, and consider a triple as verified if it can be extracted. The approach described in [43] also searches for the triple on the Web where it identifies relevant sources, extracts evidence from them, estimates source trustworthiness and uses those trustworthiness scores for improving triple evaluation. The difference with the

previous works is that it is completely machine learning based. A first a set of data is provided for training the classifiers for each category of triples, instead of using just one classifier for all the categories. This work classifies the set of unlabeled triples to either true or false and provides a confidence value attached to the label. It considers *IS-A* relationships. Furthermore, a third approach may first leverage both search-based and extraction-based techniques to find supporting evidence for each triple, and subsequently, predict the correctness of each triple based on the evidence. The extracted evidence may be from other knowledge bases, and further enriched with evidence from the web and finally from query logs. In addition to the other approaches data fusion techniques are applied for distinguishing correct triples from incorrect ones [44].

One of the earliest systems which leveraged the World Wide Web to validate facts was `KnowItAll` [33]. Soderland et al [45] describe how `KnowItAll` uses generic patterns and bootstrapping to gauge the confidence of a certain fact. It uses search engine hits to approximate the probability that a certain pattern is correct for detecting a fact which pertains to a given class or relation (a *predicate*). This probability is estimated as the number of web pages returned by a search engine which contain both the pattern and the given fact divided by the total number of pages returned which contain the fact. Each class or relation has multiple patterns and each pattern has an associated probability for a given fact. These probabilities are then fed as features to a naive Bayes classifier which finally outputs the confidence score of the fact. The main advantage of this method is that the system requires very little supervision because of its bootstrapping capabilities, and can be applied to any generic relation. But a major shortcoming of this approach is its sole reliance on search engines. Many search engines such as Google have now stopped providing APIs which facilitate automatic querying, thereby debilitating this approach. Another major shortcoming of this approach is that it is incapable of measuring the trustworthiness of the source of the information.

DeFacto [31] is a system which scores RDF triples based on evidence found in web pages. Though DeFacto uses search engines to find evidence, it overcomes the second shortcoming by using a two-pronged approach which takes into account both the trustworthiness of the source and the evidence which supports or contradicts the given fact, thereby improving the quality of its predictions.

`OpenEval` [32] is another fact validation system which leverages Google results to determine the confidence values of a given fact. The system is unique because of its ability to train classifiers within a given time limit (online algorithm). The performance of the classifiers gets better as more time is given for training. `OpenEval` takes as input a set of predicates, a set of seed instances for each of the predicates and the set of mutually exclusive relationships between the given predicates. For each seed instance, a Google query consisting of the subjects, objects and the automatically inferred keywords for the predicate is generated. After querying Google, the set of words which occur around the query in the top results is extracted. These sets, called Context-Based Instances (CBIs), are used for training the Support Vector Machines (SVMs) [46]. For each predicate, the set of CBIs generated using the seed instances of that predicate are used as positive examples, and the set of CBIs generated using seed instances of predicates which are mutually exclusive to the given predicate are used as negative examples while training the SVM for that predicate. After training all the SVMs, if there is some time remaining, the SVMs with the maximum entropy are re-trained by extracting new CBIs, so as to improve their

performance. Another unique aspect is that the keywords used while querying are generated automatically by selecting those words that have the highest weights in the SVMs as the set of possible keywords which represent that predicate. The newly generated keywords are used for generating new CBIs while re-training. While testing, the most important keywords are first used to generate the CBIs. These CBIs are then fed to the appropriate SVM to determine the confidence score of a test instance. If time remains, the keywords with lesser weights are also used while determining the final score. Though this approach has many unique features, it also suffers from its reliance on Google and it is incapable of measuring the trustworthiness of the source of the information. An SVM needs to be trained for each predicate, making it inefficient and time consuming because a set of seed instances and the set of relationships need to be supplied to train each such SVM.

2.1.3 Triple Ranking (Relevance Scoring)

Triple Ranking is the task of ordering triples with the same *subjects* and similar *objects* according to the relevance of the *objects* to the *subjects*. Bast et al [34] recently explored this problem in detail. Their dataset consisted of manually scored triples whose predicates were either "*profession*" or "*nationality*" and the triples were derived from Freebase. Each triple was scored on a scale from 0 to 7 with 0 indicating least relevance and 7 indicating most relevance. They assumed all triples to be true and built three different triple scoring mechanisms. All of the systems used a related text corpus which was used to extract features for the classifiers. The first system was based on logistic regression. They trained multiple binary classifiers, one for each profession and one for each nation, which classified triples as primary or secondary. For example, a classifier for the object "Actor" when the predicate is profession would classify the triple having "Tom Hanks" as the subject as primary and would classify the triple having "Barack Obama" as secondary. The second system computed a weighted sum which indicated the degree of relevancy of that object to that subject, predicate pair. This sum was computed by gathering the list of all words which indicated that a given *profession/nationality* was the primary *profession/nationality* for that person and then computing a weighted sum of the number of the occurrences of such words with the weights being the TF-IDF values of those words in the related text corpus. The third system used a generative model to assign the probabilities of the triple being relevant based on the related text corpus. The main advantage of the approaches proposed by the authors is that most of the learning happens in an unsupervised manner which lends the approaches to automation. An important observation is that all of these classifiers require the range of the predicate to be known. Moreover, a classifier needs to be trained for each predicate, object pair. This is not only time and resource intensive but also unfeasible if the range of a predicate is not known or subject to variation. DeFacto, on the other hand, does not need to know *a priori* the range of predicates. It uses a single classifier for all triples, leveraging the more generic features mapped by its architecture, thus making it more efficient. However, it is still dependent on a natural language library [47] to obtain the verbalizations for each possible predicate, potentially limiting the approach. Moreover, all of the approaches described in [34] require a related text corpus while training and also for evaluation. This means that the systems cannot handle real-time queries which may need information which is not contained in the related text corpus. DeFacto overcomes this major shortcoming by using search engines which provide real-time results and their results are then used to score triples.

The WSDM Cup 2017 had a challenge which required competitors to build triple ranking models similar to the ones proposed by Bast et al [34]. Zmiycharov et al [37] (`The Goosefoot Triple Scorer`) approached this problem by first downloading related Wikipedia, DeletionPedia and DBpedia data regarding the persons mentioned in the dataset. They then obtain more training data using distant supervision on the person files. The person files and training data were then normalized using synonym lists for the professions and nationalities and other basic transformations. Word2Vec embeddings, TF-IDF features and Type-like Occurrence Order features were then extracted from the person files for each of the training instances and a linear regression classifier was then trained using these features. This model was ranked the best according to the Kendall Tau metric (*tau*). Though this approach is good for the given task, it is incapable of scoring generic triples because it requires external person files which may not be available.

Hasibi et al [36] (`The Cress Triple Scorer`) uses handcrafted features to train a Random Forest model used to predict the relevance score. For each of the relations, these features are extracted from the annotated Wikipedia sentences provided by the challenge. This simple approach performed the best with respect to *average score difference* (avd) and was ranked second concerning *tau*. Although this approach performs surprisingly well, it does not work for any generic triple since the approach requires handcrafted features for each relation which is not feasible to achieve given the huge number of possible relations.

`Bokchoy` [35] employed ensemble learning to combine the results of four base scorers, three which used Wikipedia data and one which used Freebase. The three Wikipedia based classifiers are those proposed in [34]. The main novelty of this scorer was the fourth classifier which employed Freebase. It was a classifier which predicted the relevance score of a triple based on the path between the subject and the object of the triple in the knowledge base. Positive examples were obtained directly from Freebase and negative samples were generated by randomly replacing real professions/nationalities[1] with other ones and taking care that these replaced professions/nationalities were not associated with the subject. A random forest binary classifier was then trained which output the score indicating the likelihood of the given predicate connecting the given subject and object. An ensemble was employed to obtain the combined score by computing a weighted sum of the scores output by the base classifiers. The final step involved detecting "trigger" words (manually defined) for a given profession/nationality in the related text for a given person. If trigger words are found in the first paragraph of the Wikipedia text related to that person, the score computed by the ensemble is refined. This approach was ranked the best with respect to *accuracy*, second with respect to *avd* and third with respect to *tau index*. Thus, `Bokchoy` was one of the best classifiers in the challenge. However, it also can not be applied to any generic triple because it requires a trigger word based score tuning which is not possible for all *relations*. It also suffers from the same shortcomings as those experienced by the systems proposed by [34] since both use the same Wikipedia-based classifiers.

[1] the two predicates supported/available in the challenge

2.1.4 Natural Language Claim Verification

Apart from structured claims, fact-checking has also a more realistic facete: the verification of claims written in natural language. Thorne et al. [48] give an overview of fact-checking automation, bridging the gaps between fact-checking and related research areas. Starting with the fact-checking in journalism, they define basic terminology and then go on drawing a parallel between Fake news research, fact-checking, textual entailment etc. Fact-checking as a branch of NLP is the main focus of this study. They also discuss input and output structures of well-known fact-checking and validation systems and publicly available datasets. An important contribution of their work is shedding light on the importance of evidence in fact-checking systems.

Vlachos and Riedel [9] define the problem of fact-checking as the truthfulness of claims, a binary classification problem. They provide two datasets constructed from political fact-checking websites *PolitiFact* and *Channel-4*. Their work tries to define the problem of fact-checking as a one-to-one automation mapping of the human fact-checking process. But the work does not give a concrete implementation of their pipeline hypothesis. Lee et. al. [49] propose a neural-ranker-based evidence extraction method, an important part of the fact-checking pipeline, extending the [50] baseline method. They propose a three-step (document retrieval, evidence selection, and textual entailment) pipeline for the task. Taniguchi et al. [51] give a three-component pipeline consisting of document retrieval, sentence selection and recognizing textual entailment (RTE). Popat et al. [52] design an end2end model for fake-news detection. They use pre-retrieved articles related to a single fact and aggregate the veracity score from each article to make the final decision about the claim. They experiment on four political fact-checking datasets and compare the performance of their model with baselines on *Snopes* and *Politifact*. Tosik et al. [53] talk about the limitation of deep-learning in yielding interpretable results and the challenges faced specifically in the case of fake-news detection. They model a feature based framework for *stance detection*, outperforming the fake-news-challenge (FNC-1)[2] baseline. Yang et al. [54] propose a convolution neural network-based approach for fake news detection. They combine the text in the articles with the image cues. Though this is an interesting variation of fake news detection, no improvements over state-of-the-art were reported.

Recently many studies have proposed deep learning solutions to the fact-checking problem [52, 55–57]. DeclarE [52] combines the evidence extraction and claim classification in a single end2end model. Sizhen et al [55] tackle the problem with a deep learning paradigm. They select relevant Wikipedia entities using an online available tool, *S-MART*, and use their model to perform both *evidence selection* and *claim classification* in a combined fashion using bi-directional attention. The model reaches baseline results but does not perform well on *evidence retrieval* task. Conforti et al. [56] propose a deep learning approach for fake-news detection by focusing on the stance of the claims. They use the dataset from Fake-News-Challenge (FNC-1) to test their model. The model yields better results than the top performers of FNC-1 but there is no comparison with other end-to-end deep learning approaches. However, the approach handles cross-level stance detection when the input corpus has a high variation in length. Yin and Roth [57], give a two-wing-optimization strategy and combine the last two steps of Fact-Checking pipeline. Their model beats the baseline [58] on evidence identification and claim verification by a good margin. Karadzhov1 et al. [12] proposes a fully automated fact-checking system. Their model follows a

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

three-component structure, named: *External support retrieval*, *Text representation* and *veracity prediction*[3].

Baly et al. [59] use the approach of bias detection in the news media and predict the "factuality" in news reported by the media source. They divide the task of veracity assessment of information into four classes: fact-checking (claim level), fake news detection (article level), troll detection (user level) and source reliability estimation (medium level). Their work explains the subtle difference in all these related areas. Popat et al. [60] add source trend and language to the credibility assessment approach. Their model also provides user interpretable explanations of the final decision. They use web for retrieving source documents a.k.a reporting articles from using a search engine. Due to this method of retrieving source documents, they call their model a content-aware model (an idea similar to the topical-evidence based approach).

Some previous works have focused on knowledge bases [9, 61] as a method for fact-checking. Ciampaglia et al. [61] formulate the fact-checking problem as a special case of link prediction in knowledge graphs. They use *DBpedia*[4] database, a knowledge base derived from Wikipedia. The fundamental limitation with this line of research is that the most commonly used knowledge bases are outdated and they cannot leverage from the status-quo of the world, thus lacking the external-evidences.

2.2 Named Entity Recognition on Noisy Text

Over the past few years, the problem of recognizing named entities in natural language texts has been addressed by several approaches [62–66], reporting decent to very good performance measures on *newswire* datasets [66, 67] (mostly CoNLL-2003[5]), but failing to recognize entities in *microblogs* and similar resources with little context or sentences which are not compliant with grammatical norms (e.g. isolated snippets of search engine results and *microblog* posts). Thus, the major disadvantage of most NER architectures is the domain-specific knowledge dependency, which imposes a natural barrier to generalize over different contexts and datasets. Designing hand-crafted features for each domain represents a major obstacle for generalization. To bridge this gap, Collobert et al. [68] proposed a neural network (NN) model which required little feature engineering using word embeddings. Al-Rfou et al. [69] also proposed a language agnostic model that learns distributed word representations (i.e., word embeddings) which encode semantic and syntactic features. Chiu and Nichols [70] minimized the NN known design gap regarding long-term dependency with a RNN. In a similar architecture Lample et al. [71] recently reported the performance for different languages (*English, German, Dutch* and *Spanish*), showing F_1 measures (on the CoNLL-2003 test set) similar to other state of the art NER models: $90.9, 78.7, 81.7$ and 85.7 (respectively). They, however, focused on *newswire* data sets to improve current state of the art systems and not on *microblogs* texts (e.g. `WNUT` datasets), which are naturally harder to tackle due to the issues previously introduced. Shifting to this context, approaches have emerged specifically designed to better perform on short and noisy texts, such as `T-NER` [19] and `TwiterIE` [72]. The first performs tokenization, POS tagging and noun-phrase

[3] although a different name convention, it is in line with the approach proposed by [50]
[4] https://wiki.dbpedia.org/
[5] http://www.cnts.ua.ac.be/conll2003/ner/

chunking before using topic models to find named entities whereas the second – an extension of GATE ANNIE [73] – implements an NLP pipeline customized to microblog texts at every stage (including Twitter-specific data import and metadata handling). Also, Liu et al. [20] propose a gradient-descent graph-based method for text normalization and recognition. Likewise, these approaches are highly dependent on hand-crafted rules. Recently, in order to overcome this issue, different models were proposed. Limsopatham and Collier [74] proposed a neural architecture for NER on *microblogs*, which combines a bidirectional LSTM with an CRF achieving a F_1 measure of 52.41 for *English* text. Models supporting other languages were proposed, however, similar performances (min-max F_1 measure) have also been observed across different languages other than English, such as French, Portuguese and Chinese, for instance ([75] - $21.28 - 58.59$, [76] - $24.40 - 52.78$ and [77] - $44.29 - 54.50$, respectively). [78] proposed a methodology to encode image and news features into NER architectures, showing promising preliminary results. [79] followed the same idea to detect entities in Twitter, but just analyzing existing images associated to a given tweet, which drastically restricts the approach. Furthermore, in a period of three years of a very famous workshop for NER in social media[6], modest results have been reported by a vast number of different NER architectures: $16.47 - 56.41$, $19.26 - 52.41$ and $39.98 - 41.86$ (min-max F_1 in WNUT 2015, 2016, 2017, respectively) [80–82]. Therefore, although neural architectures pose a good choice to outperform standard architectures (e.g. CRFs), the task is still far from being solved for *microblogs*.

2.3 Web Credibility

Credibility is an important research subject in several different communities and has been the subject of study over the past decades. Most of the research, however, focuses on theoretical aspects of credibility and its persuasive effect on different fundamental problems, such as economic theories [83]. Due to its fuzzy nature, the definition of credibility may be subject of distinct interpretations [84].

2.3.1 Fundamental Research

A thorough examination of psychological aspects in evaluating documents credibility has been studied [84–86], which reports numerous challenges. Apart from sociological experiments, *Web Credibility* has a more practical perspective. Figure 2.1 depicts the main sides of credibility assessment.

While our work falls into the category *Automated Models - Website Scoring*, we briefly introduce related research (which, in turn, are relevant to validate important credibility factors in a practical scenario) as follows:

Rating Systems, Simulations are mostly platform-based solutions to conduct experiments (mostly using private data) in order to detect credibility factors. Nakamura et al. [87] surveyed internet users from all age groups to understand how they identified trustworthy websites. Based

[6] http://noisy-text.github.io

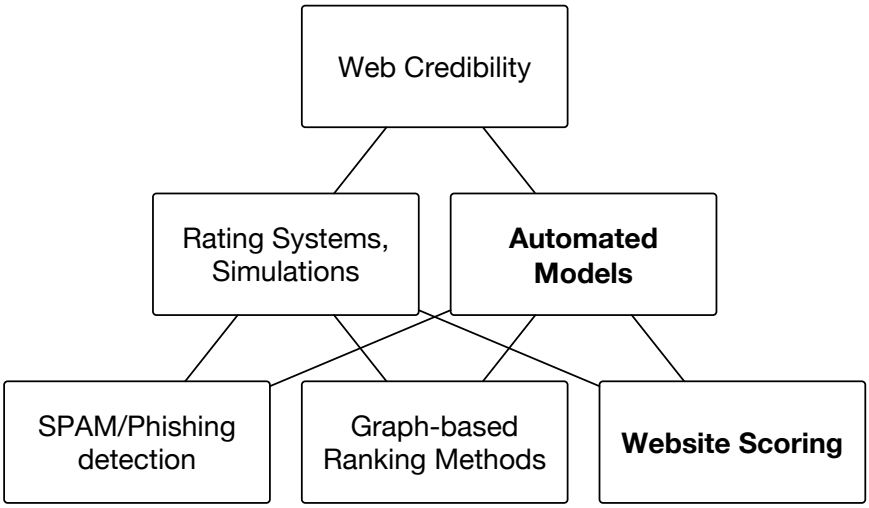


Figure 2.1: Credibility models

on the results of this survey, they built a graph-based ranking method which helped users in gauging the trustworthiness of search results retrieved by a search engine when issued a query $\mathcal{Q}$. A study by Stanford University revealed important factors that people notice when assessing website credibility [86], mostly visual aspects (*web site design*, *look* and *information design*). The *writing style* and *bias of information* play a small role as defining the level of credibility (selected by approximately 10% of the comments). However, this process of evaluating the credibility of web pages by users is impacted only by the number of heuristics they are aware of [88], biasing the human evaluation w.r.t. a limited and specific set features. An important factor considered by humans to judge credibility relies on the search engine results page (SERP). The higher ranked a website is when compared to other retrieved websites the more credible people judge a website to be [89]. Popularity is yet another major credibility factor [90]. Liu et al. [91] proposed to integrate recommendation functionality into a Web Credibility Evaluation System (WCES), focusing on the user's feedback. Shah et al. [92] propose a full list of important features for credibility aspects, such as 1) the quality of the design of the website and 2) how well the information is structured. In particular, the perceived accuracy of the information was ranked only in 6th place. Thus, superficial website characteristics as heuristics play a key role in credibility evaluation. Dong et al [93] propose a different method (KBT) to estimate the trustworthiness of a web source based on the information given by the source (i.e., applies fact-checking to infer credibility). This information is represented in the form of triples extracted from the web source. The trustworthiness of the source is determined by the correctness of the triples extracted. Thus, the score is computed based on *endogenous* (e.g., correctness of facts) signals rather then *exogenous* signals (e.g., links). Unfortunately, this research from Google does not provide open data. It is worth mentioning that - surprisingly - their hypothesis (content is more important than visual) contradicts previous research findings [86, 92]. While this might be due to the dynamic characteristic of the Web, this contradiction highlights the need for more research into the real use of web credibility factors w.r.t. automated web credibility models. Similar to [87], Singal and Kohli [94] proposes a tool (dubbed TNM) to re-rank URLs extracted from Google search engine according to the trust maintained by the actual users). Apart from the search engine API, their tool uses several other APIs to collect website usage information

(e.g., traffic and engagement info). [95] perform extensive crowdsourcing experiments that contain credibility evaluations, textual comments, and labels for these comments.

SPAM/phishing detection: Abbasi et al. [96] propose a set of design guidelines which advocated the development of SLT-based classification systems for fraudulent website detection, i.e., despite seeming credible - websites that try to obtain private information and defraud visitors. PhishZoo [97] is a phishing detection system which helps users in identifying phishing websites which look similar to a given set of protected websites through the creation of profiles. It first creates a set of profiles for the given set of protected web pages. These are web pages which the user logs in to and hence wants to be warned of similar looking phishing websites. A webpage profile is created by storing the SSL certificates, URL and content of the webpage. The profile also stores the SIFT features [98] of user-selected logo of the webpage. Upon browsing, when a webpage is being loaded, its SSL certificate and URL are matched with those of the protected websites. If they do not match with any protected website, the page may be a phishing website. To further examine the webpage, tokens from the website URL and body are extracted. These tokens are matched with the keywords present in the protected websites. The keywords are determined based on TF-IDF scores. The protected website with the most number of matching keywords is chosen and the SIFT features of the logo of the protected website are compared with the SIFT features of all the images present in the given webpage. If a high level of similarity is discovered, the website is likely to be a phishing website and the system alerts the user.

2.3.2 Automated Web Credibility

Automated Web Credibility models for website classification are not broadly explored, in practice. The aim is to produce a predictive model given training data (annotated website ranks) regardless of an input query $\mathcal{Q}$. Existing gold standard data is generated from surveys and simulations (see *Rating Systems, Simulations* related work). Currently, state of the art (SOTA) experiments rely on the Microsoft Credibility dataset[7] [89]. Recent research use the website label (Likert scale) released in the Microsoft dataset as a gold standard to train automated web credibility models, as follows:

Olteanu et al. [99] proposes a number of properties (37 linguistic and textual features) and applies machine learning methods to recognize trust levels, obtaining 22 relevant features for the task. Wawer et al. [100] improve this work using psychosocial and psycholinguistic features (through The General Inquirer (GI) Lexical Database [101]) achieving state of the art results.

Finally, another resource is the Content Credibility Corpus (C3) [95], the largest Web credibility Corpus publicity available so far. However, in this work authors did not perform experiments w.r.t. *automated credibility models* using a standard measure (i.e., Likert scale), such as in [99, 100]. Instead, they rather focused on evaluating the theories of web credibility in order to produce a much larger and richer corpus.

[7] It is worth mentioning that this survey is mostly based on confidential data and it is not available to the open community (e.g., overall popularity, popularity among domain experts, geo-location of users and number of awards)

2.4 Reproducible Research

When sharing experiment results, researchers often describe them in different language writing style (which is possible to become ambiguous) in their manuscripts making it difficult to directly compare results from different papers. Besides the interpretation issue, a major problem is the difficult to confirm scientific findings. A relatively recent *key term* to face this lack of metadata is *Reproducible Research*, which aims to make analytic data and code freely available so that others will be able to reproduce findings, i.e., an environment where "provenance metadata" is accessible and a "high interoperability" level is achievable, so anyone is able to reproduce scientific achievements. Therefore, *Reproducibility* is one of the main principles of the scientific methods (Figure 1.2). According to the IOM Report [102] the following rules should be applied: 1) data/metadata publicly available; 2) the computer code and all the computational procedures should be available; 3) ideally the computer code will encompass all of the steps of computational analysis. In the following we list different attempts to bridge this gap.

2.4.1 Workflow Systems

Managing the configurations, inputs and outputs of ML algorithms poses a huge challenge for developers. Several machine learning experiments are performed through general purpose programming languages, such as *Java*, *C#*, *Python*, *C++*, with or without use of libraries like *Weka*[8],*scikit-learn*[9] or *Shogun*[10]. In this context, one of the requirements is to implement some machine-readable way for interchanging results over distinct architectures. Examples of state-of-the-art formats and patterns for interchanging are: *Comma-Separated Values (CSV)*, *eXtensible Markup Language (XML)*, *Value-Object (VO)*, *Data-Transfer-Objects (DTO)*. However, the drawbacks here are threefold: (1) the technology dependence for the implementation of the certain design pattern; (2) the possible lack of schema information on the implementation of certain format; (3) the lack of semantic information in both cases.

In order to bridge this gap, several research groups have developed in house frameworks for managing their own workflows. Table 2.2 sums up the described related works on provenance meta-data for scientific experiments.

Platform	Description
MyExperiment [103]	It is a collaborative environment where scientists can publish their workflows and experiment plans
Wings [104]	A Semantic Approach to creating very large scientific workflows
OpenTOX [105]	An interoperable predictive toxicology framework
Open ML [106]	A frictionless, collaborative environment for exploring machine learning

Table 2.2: The state-of-the-art platforms for e-science workflows

Although posing as a specific-solution for tackling the problem, they are not generic enough to become a generic solution for representing any machine learning experiment. This results

[8] http://www.cs.waikato.ac.nz/ml/weka/
[9] http://scikit-learn.org/stable/
[10] http://www.shogun-toolbox.org

in redundancy and increased maintenance costs (often within the same institution). Instead, state-of-the-art workflow approaches have been successfully developed to describe and manage configurations and outcomes of specific problems which derived from the experimental process in specific research communities (e.g., Bioinformatics) [103, 105, 107].

2.4.2 Vocabularies and Ontologies

Therefore, in order to bridge the gap between description of experiments and its reproducibility, more rich and generic structures are required. Likewise, Table 2.3 shows the related works on data mining ontology.

Platform	Description
The Data Mining OPtimization Ontology (DMOP) [108]	It supports informed decision-making at various choice points of the data mining process
OntoDM-KDD [109]	Ontology for representing the knowledge discovery process
Exposé [110]	An ontology for data mining experiments used in conjunction with *experiment databases* (ExpDBs [111])

Table 2.3: The related (heavy-weight) ontologies for data mining and their respective conceptualizations

2.4.3 Repositories for Machine Learning Metadata Experiments

Besides, a more generic machine learning platform and very interesting approach for representing machine learning experiments is *OpenML* [106], which misses an ontology for representing the metadata[11] and runs over *XML* schema representation[12].

[11] https://github.com/openml/OpenML
[12] http://www.openml.org/r/454640/output/description

CHAPTER 3

Detecting Entities on Noisy Data

This chapter is dedicated to address one of the core challenges of this thesis, i.e., to identify named entities in noisy data. The content of this chapter is based on the publications [18, 78, 112–114]. Over the past decade, different Named Entity Recognition frameworks have been proposed, most of them based on handcrafted features and restricted to a particular dataset, which imposes a natural restriction w.r.t. generalization. To bridge this gap, recent work based on neural networks were proposed, achieving state-of-the-art performance in *newswire* datasets, but still failing at performing similarly in *microblogs*. Thus, designing and exploring new methods and architectures is highly necessary to overcome current challenges. In this Section, we shift the focus to an entirely different perspective. We investigate the potential of embedding word-level *global features* extracted from images and news.

The results of this chapter provide an answer to the following research question:

> **RQ1**: Can images along with news improve the performance of the named entity recognition models on noisy text?

We performed a comprehensive study in order to validate the hypothesis that images and news boost the task on noisy data. In the best configuration setting, we show that this approach outperforms strong baselines.

First, in Section 3.1, we present a motivating example illustrating the problem of detecting named entities in noisy text, i.e., informal domains. To address research question RQ1, we devise HORUS, a multi-level NER Framework based on computer vision and data mining techniques.

Next, Section 3.2 describes our approach, including a formal problem statement as well as the main steps HORUS performs. Overall, HORUS performs a two-steps approach to derive heuristics which are relevant to perform the task. Training phases are detailed in Sections 3.3.1 and 3.3.2, for computer vision and text mining modules, respectively. Afterwards, a comprehensive evaluation of the HORUS approach and analysis of the obtained results is presented in Section 3.4. Observed results suggest that HORUS is able to boost the NER task in noisy context. Finally, Section 3.5 presents the closing remarks of this chapter. We summarize

the contributions of this chapter as follows:

- A novel methodology to extract relevant information from tokens which is based on the concept of images and news.
- A novel NER Framework named HORUS, which implements the concepts behind this methodology, generating a set of heuristics to boost the task.
- An empirical evaluation to assess the effectiveness of HORUS for the NER task on noisy text. Experiments are executed over the most famous datasets for the task: Ritter, WNUT-15, WNUT-16 and WNUT-17.

3.1 Named Entity Recognition for Noisy Text

While named entity recognition (NER) on *newswire* datasets (e.g., CoNLL datasets) has been shown to be reasonably accurate – achieving average F_1 measure up to 90% [115] – most of the state-of-the-art approaches still heavily rely on carefully constructed orthographic features and language-specific resources, such as *gazetteers*. To bridge this gap, more recent work have proposed architectures based on LSTM networks. Although this not necessarily introduces state of the art (SOTA) performance [71, 116], the trained networks achieved very similar performance on a popular *newswire* corpora (respectively 88.83% and 90.94% on CoNLL-2003 test set). Besides supporting different languages with low effort, the great advantage of such (end-to-end) approaches lies in the fact that specific knowledge resources are not required (excepting for specific *embeddings*, which are language dependent), alleviating the dependency on manually annotated data and encoded rules.

However, unlike *newswire*, *microblogs* often deal with more informal languages, which do not have such implicit linguistic formalism. Thus, they have more unstructured properties, are shorter, they lack context and present more grammatical and spelling errors [19–21]. With respect to that – not surprisingly – the performance of SOTA degrades significantly in *microblogs*, evidencing the sensibility of the proposed models when dealing with noisy and out-of-domain text. In recent work [19, 74, 117, 118] F_1 ranging from 0.19 to 0.52 have been reported.

Hence, devising models to deal with linguistically complex contexts such as *twitter* remains an open and very challenging problem to tackle, regardless of the architecture's design.

3.2 The HORUS Approach

3.2.1 Problem Definition

Named-entity recognition (NER) is a subtask of information extraction that attempts to detect and label named entity mentions in unstructured text accord to pre-defined categories. These categories may vary from more classical entities, such as *person*, *location* and *organisation*

to more specific classes such as *species of plants*. The NER algorithms receive as input an unannotated excerpt of text (often a sentence) as input. For instance: "*paris hilton was once the toast of the town.*" should ideally be annotated as "*paris|PER hilton|PER was once the toast of the town|LOC*". However, most of NER architectures still fails at correctly recognizing when grammatical structure of some sentences may be incorrect. The difficult of the task is accentuated when more informal language is used, e.g., "*YOLO bro let s head to jacks!*" ("*You|PER only live once brother, lets head to the Jacks|LOC!*").

3.2.2 Proposed Solution

In this section, we introduce the main idea of the proposed approach. The main motivation behind our approach lies in the fact that linguistic features are proved to not be enough to perform the NER task reasonably well. We argue that multi-model comprehension has a great potential to boost the task, i.e., information extracted from related images and documents. Thus, we develop a novel architecture that learns latent features from images and textual information to detect named entities, without requiring further engineering effort.

Figure 3.1 exemplifies the approach. Given a sentence $\mathcal{S}$, we perform tokenization (`A`) and, for each (*noun*) token, query a corpus (in this example, the Web) in order to obtain documents (website) and images (`B`). Afterwards, we cache the resulting metadata (`C`). The next step involves the execution of a set of models to perform image classification (`D.1`) as well as text classification (`D.2`). Finally, a supervised learning classifier is trained to obtain the final NER class for a given token.

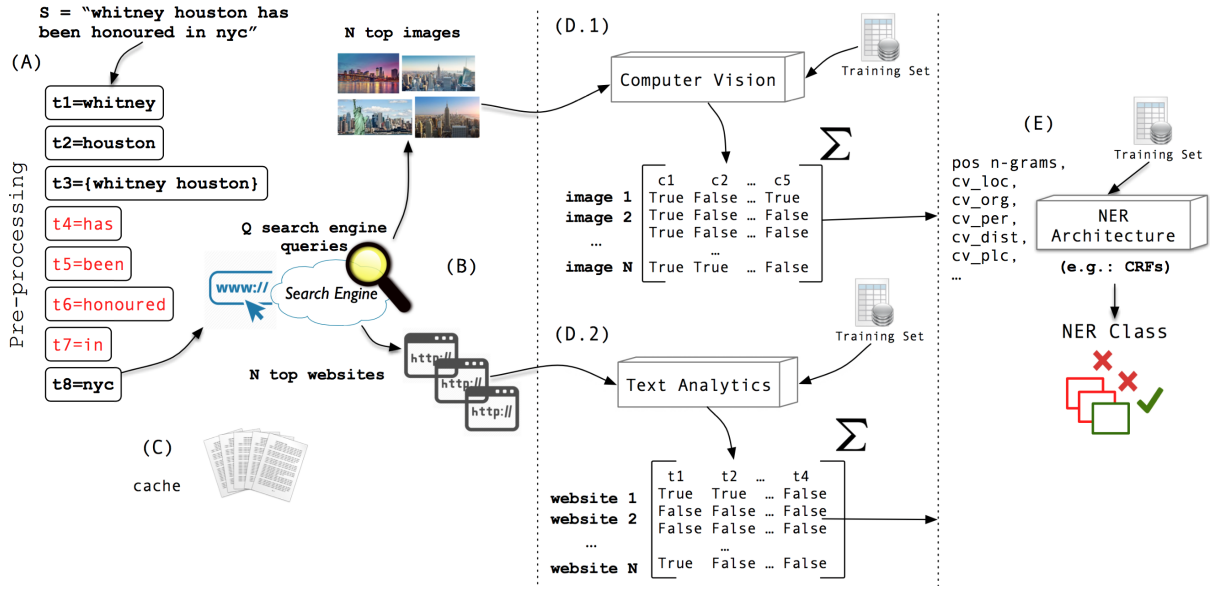


Figure 3.1: sadsa

In order to train the set of classifiers, first we define - for each category of interest - the most representative content which should be identified. For instance, in images we train models to detect *faces* if "PERSON" is the target class. Similarly, we train models to detect *logos* if

"ORGANISATION" is the class to be discovered. We define this associations by common-sense, given that a name of a person has a high correlation to images containing *faces* whereas a name of a company has a high correlation to images containing *logos*, for instance. Thus, named entities can be classified as belonging to a certain category by detecting these representative objects in the related images. Table 3.1 overviews the object categories to be discovered.

NER	Images Candidates (number of trained models)
LOC	Building, Suburb, Street, City, Country, Mountain, Highway, Forest, Coast and Map (10)
ORG	Company Logos (1)
PER	Human Faces (1)

Table 3.1: NER classes and respective objects defined by common sense rules.

Therefore, for each token $t \in \mathcal{T}$ we extract a set of image and text feature vectors $\mathcal{F} = (\mathcal{F}_1, \ldots, \mathcal{F}_n)$ that serve as input features to a NER classifier. In the following sections, we explain the two feature extraction modules we designed.

3.3 Framework Modules

3.3.1 Computer Vision Module

Given a set of images $\mathcal{I}$, the basic idea behind this component is to detect a specific object (denoted by a class c) in each image. Thus, we query the web for a given term t and then extract the features from each image and try to detect a specific object, as introduced before, for the top N images[1] retrieved as source candidates.

For training our classifiers, we used a technique called SIFT (Scale Invariant Feature Transform) features [119] for extracting image descriptors. In order to create the clusters of the extracted features, we apply BoF (Bag of Features) [120, 121]. The clustering is possible by constructing a large vocabulary of many visual words and representing each image as a histogram of the frequency words that are in the image. We use *k-means* [122] to cluster the set of descriptors to k clusters. The resulting clusters are compact and separated by similar characteristics. An empirical analysis shows that some image groups are often related to certain named entities (NE) classes when using search engines, as described in Table 3.1. For training purposes, we used the *Scene 13* dataset [123] to train our classifiers for *location* (LOC), "faces" from *Caltech 101 Object Categories* [124] to train our *person* (PER) and logos from *METU dataset* [125] for *organisation* ORG object detection. These datasets produces the training data for our set of supervised classifiers (1 for ORG, 1 for PER and 10 for LOC). We trained our classifiers using Support Vector Machines [126] once they generalize reasonably enough for the task[2].

[1] We set $N = 10$ in our experiments and used Microsoft Bing as the search engine.
[2] *scikit-learn*: svm.NuSVC(nu=0.5, kernel='rbf', gamma=0.1, probability=True).

3.3.2 Text Mining Module

analogously to previous section, we perform clustering to group texts together that are "distributively" similar. Thus, for each retrieved web page (title and excerpt of its content), we perform the classification based on the main NER classes. We extracted features using a classical sparse vectorizer (Term frequency-Inverse document frequency - TF-IDF. In experiments, we did not find a significant performance gain using *HashingVectorizer*) - `Training (D.2)`: with this objective in mind, we trained classifiers that rely on a bag-of-words technique. We collected data using *DBpedia* instances to create our training dataset ($N = 15000$) and annotated each instance with the respective MUC classes, i.e. *PER*, *ORG* and *LOC*. Listing 3.1 shows an example of a query to obtain documents of organizations (ORG class). Thereafter, we used this annotated dataset to train our model.

```
SELECT ?location, ?abstract FROM <http://dbpedia.org>
WHERE {?location rdf:type dbo:Location .
       ?location dbo:abstract ?abstract .
FILTER (lang(?abstract) = 'en')} LIMIT 15000
```

Listing 3.1: SPARQL: an example of querying DBpedia to obtain LOC data for training

3.3.3 Final Classifier

We use the outcomes of (Sections 3.3.1 and 3.3.2) as part of the input to our final model. The final set of indicators is defined as follows: let W_s be a set of *tokens* existing in a given sentence $s \in S$. We extract the POS tag (using Stanford POS Tagger) for each token w and filter out any token classified other than *PROP-NOUN* and existing *compounds* as entity candidates ($t \in S'$). The result is a simple structure:

$$M_i = \{j, t, ng_{pos}, C_{loc}, C_{per}, C_{org}, C_{dist}, C_{plc}, T_{loc}, T_{per}, T_{org}, T_{dist}\} \tag{3.1}$$

where i and j represent the i^{th} and j^{th} position of $s \in S$ and $w \in W_s$, respectively. ng_{pos} represents the n-gram[3] of POS tag. C_k and T_k ($k \in \{loc, per, org\}$) represent the total objects found by a classifier Φ for a given class k ($\sum_{n=1}^{N} \Phi(k, img_n)$)[4] (where N is the total of retrieved images $\mathcal{I}$). C_{dist} and T_{dist} represent the distance between the two higher predictions ($\mathcal{P} = \{C_k \forall K\}$), i.e. $\max(\mathcal{P}) - \max(\mathcal{P}')|\mathcal{P}' = \mathcal{P} - \{\max(\mathcal{P})\}$. Finally, C_{plc} represents the sum of all predictions made by all *LOC* classifiers $\mathcal{CL}$ ($\sum_{l=1}^{L} \sum_{n=1}^{N} \mathcal{CL}_l(loc, img_n)$)[5]. - `Training (E)`: the outcomes of `D.1` and `D.2` (M) are used as input features to our final classifier. We implemented a simple *Decision Tree*[6] (non-parametric supervised learning method) algorithm for learning simple decision rules inferred from the data features (since it does not require any assumptions of linearity in the data and also works well with outliers, which are expected to be found more often in a noisy environment, such as the Web of Documents).

[3] *bigram*, in our experiments.
[4] $pos = +1$, $neg =$ -1.
[5] $pos = +1$, $neg = 0$.
[6] *scikit-learn*: criterion='entropy', splitter='best'.

3.4 Experimental Setup

3.4.1 Datasets

In order to evaluate the models closer to a real use-case scenario, we performed training and test across all dataset combinations. (e.g., `[training = Ritter, test = WNUT-15]`, `[training = Ritter, test = WNUT-16]` and `etc.`). In case of Ritter, we followed the proposed split [19], since there is no specific `training`, `dev` and `test` splits. Finally, we also report performance measure for all `training` sets using 3-fold cross-validation as sampling method. In the following we briefly described the algorithms used in our experiments.

3.4.2 NER Algorithms

NER Architectures: We benchmark the methodology on top of different *weak* and *strong* baselines, as follows:

Random Forests (RF) [127], a non-parametric (decision trees) supervised learning method for classification and regression tasks. The goal is to create a model that classifies by learning simple decision rules inferred from the data features. They are simple to understand and to interpret. We chose Random Forests (RF) in order to gain further insight on the relation between the heuristics we propose and the target (named entities) to predict.

Conditional Random Fields (CRFs) [128], a more classical solution to *sequence labeling* problems. CRF is an undirected graphical models trained to achieve the maximization of a conditional probability. A linear-chain CRF with parameters w defines a conditional probability for a output sequence $y = y_1, y_2, \ldots, y_n$ given an input sequence $x = x_1, x_2, \ldots, x_n$, as $P_w(y|x) = \frac{1}{Z_w} \exp(w^n \phi(x, y))$, where Z_w is a normalization constant which ensures that the sum over all possible outputs equals one. The major advantage is that this class of algorithms can take *context* into account. We implemented the standard CRF (gradient descent with L-BFGS method) and the CRF with passive-aggressive (PA) updates to overcome some social media noise [129].

Recurrent Neural Networks (RNNs) which are state of the art architecture for NER in *formal domains* [71, 116] but have been shown to perform only reasonably in *noisy data* [130]. It is unclear how traditional NNs could reasoning about past events[7] to update later ones (i.e., what they perceived one step back in time). Long Short Term Memory networks (LSTMs) [131] are a special type of RNNs that are explicitly designed to avoid the long-term dependency problem, which is an issue in standard RNNs. We implement a series of neural network-based architectures, as follows: B-LSTM+CRF [116], Char+B-LSTM+CRF [71] and B-LSTM+CNN+CRF [132].

[7] tokens in our context

3.4.3 Experiments: Binary Classifiers Committee

In order to check the overall performance of the proposed technique, we ran our algorithm without any further rule or *apriori* knowledge using a gold standard for NER in microblogs (Ritter dataset [19]), achieving 0.59 F1. Table 3.2 details the performance measures per class. Table 3.3 presents current state-of-the-art results for the same dataset. The best model achieves 0.8 F1-measure, but uses encoded rules. Models which are not rule-based, achieve 0.49 and 0.56. We argue that in combination with existing techniques (such as linguistic patterns), we can potentially achieve even better results.

NER Class	Precision	Recall	F-measure
Person (PER)	0.86	0.53	0.66
Location (LOC)	0.70	0.40	0.51
Organisation (ORG)	0.90	0.46	0.61
None	0.99	1.0	0.99
Average (PLO)	0.82	0.46	**0.59**

Table 3.2: Performance measure for our approach in Ritter dataset: 4-fold cross validation

NER System	Description	Precision	Recall	F-measure
Ritter et al., 2011 [19]	LabeledLDA-Freebase	0.73	0.49	0.59
Bontcheva et al., 2013 [72]	Gazetteer/JAPE	0.77	0.83	0.80
Bontcheva et al., 2013 [72]	Stanford-twitter	0.54	0.45	0.49
Etter et al., 2013 [133]	SVM-HMM	0.65	**0.49**	0.54
our approach	Cluster (images and texts) + DT	**0.82**	0.46	**0.59**

Table 3.3: Performance measures (PER, ORG and LOC classes) of state-of-the-art NER for short texts (Ritter dataset). Approaches which do not rely on hand-crafted rules and *Gazetteers* are highlighted in gray. Etter et al., 2013 trained using 10 classes.

As an example, the sentence "*paris hilton was once the toast of the town*" can show the potential of the proposed approach. The token "*paris*" with a LOC bias (0.6) and "*hilton*" (global brand of hotels and resorts) with indicators leading to LOC (0.7) or ORG (0.1, less likely though). Furthermore, "*town*" being correctly biased to LOC (0.7). The algorithm also suggests that the *compound* "*paris hilton*" is more likely to be a PER instead (0.7) and updates (correctly) the previous predictions. As a downside in this example, the algorithm misclassified "*toast*" as LOC. However, in this same example, Stanford NER annotates (mistakenly) only "*paris*" as LOC. It is worth noting also the ability of the algorithm to take advantage of search engine capabilities. When searching for *"miCRs0ft"*, the returned values strongly indicate a bias for ORG, as expected (C_{loc} = 0.2, C_{org} = 0.8, C_{per} = 0.0, C_{dist} = 6, C_{plc} = -56, T_{loc} = 0.0, T_{org} = 0.5, T_{per} = 0.0, T_{dist} = 5). More local organisations are also recognized correctly, such as "*kaufland*" (German supermarket), which returns the following metadata: C_{loc} = 0.2, C_{org} = 0.4, C_{per} = 0.0, C_{dist} = 2, C_{plc} = -50, T_{loc} = 0.1, T_{org} = 0.4, T_{per} = 0.0, T_{dist} = 3.

We further investigate the impact of basic lexical ($\mathcal{S}$) features in the pipeline along with the image $\mathcal{CV}$ (SIFT+K-means+SVM) and textual features $\mathcal{TX}$ (TF-IDF+SVM). For such, we break the experiments into a set of distinct configurations. Table 3.4 depicts related features sets as well as originally proposed features (`cfg04`).

Cfg[8]	Features
cfg01	$\mathcal{S}$
cfg02	$\mathcal{S} + \mathcal{TX}$ (TF-IDF+SVM)
cfg03	$\mathcal{S} + \mathcal{CV}$ (SIFT+K-means+SVM)
cfg04	$\mathcal{S} + \mathcal{TX} + \mathcal{CV}$ [78]
cfg05	$\mathcal{S}$ + Lemma
cfg06	$\mathcal{S}$ + Lemma + $\mathcal{TX}$
cfg07	$\mathcal{S}$ + Lemma + $\mathcal{CV}$
cfg08	$\mathcal{S}$ + Lemma + $\mathcal{TX}$ + $\mathcal{CV}$

Table 3.4: Experiment Configurations (Exp): Previous work's features expanded into different set of features.

3.4.4 Experiments: Advanced Neural Network Techniques

In order to further investigate the potential of embedding visual and textual features in the NER extraction task, we extended and performed a full investigation of the impact of such features. First, we also included other relevant NER features, such as Brown Clusters [134]. Then we also adapted the proposed methodology implementing recent neural architectures (e.g., B-LSTM [116]). We detail modifications and configurations as follows:

Brown Clusters

The usefulness of Brown clusters ($\mathcal{B}$) in NER [135] and the sensitivity of the number of clusters in the NER task has been recently studied in [134]. We explore these findings altering the number of clusters (320, 640 and 1000). In these configuration settings, the brown clusters are then used as features along with Standard ($\mathcal{S}$) and [78] features. Table 3.5 details the configurations.

Cfg	Features
cfg09	$\mathcal{S}$ + Brown 64M c320
cfg10	$\mathcal{S}$ + Brown 64M c640 ($\mathcal{B}_{best}$)
cfg11	$\mathcal{S}$ + Brown 500M c1000
cfg12	$\mathcal{S}$ + Lemma + Brown 64M c320
cfg13	$\mathcal{S}$ + Lemma + Brown 64M c640
cfg14	$\mathcal{S}$ + Lemma + Brown 500M c1000
cfg15	$\mathcal{S} + \mathcal{B}_{best} + \mathcal{CV}$
cfg16	$\mathcal{S} + \mathcal{B}_{best} + \mathcal{TX}$
cfg17	$\mathcal{S} + \mathcal{B}_{best} + \mathcal{CV} + \mathcal{TX}$

Table 3.5: Exploring Brown Clusters along with [78] features. $\mathcal{B}_{best}$ stands for the best found number of clusters (640).

Neural Network-based Features

We finally explore and analyze the impact of several new proposed features considering state of the art methods for image recognition and text classification. In order to evaluate the impact of each new designed feature, we also split them into distinct experimental *configurations* (Table 3.6).

Cfg	Features	Cfg	Features
cfg18	$\mathcal{S}$+$\mathcal{CV}_{cnn}$	cfg30	=18+$\mathcal{B}_{best}$
cfg19	$\mathcal{S}$+$\mathcal{TX}_{cnn}$	cfg31	=19+$\mathcal{B}_{best}$
cfg20	$\mathcal{S}$+$\mathcal{TX}_{emb}$	cfg32	=20+$\mathcal{B}_{best}$
cfg21	$\mathcal{S}$+$\mathcal{TX}_{stats}$	cfg33	=21+$\mathcal{B}_{best}$
cfg22	$\mathcal{S}$+$\mathcal{TX}_{cnn}$+$\mathcal{TX}$	cfg34	=22+$\mathcal{B}_{best}$
cfg23	$\mathcal{S}$+$\mathcal{TX}_{cnn}$+$\mathcal{TX}$+$\mathcal{TX}_{e}$ +$\mathcal{TX}_{stats}$	cfg35	=23+$\mathcal{B}_{best}$
cfg24	$\mathcal{S}$+$\mathcal{TX}_{cnn}$+$\mathcal{CV}_{cnn}$	cfg36	=24+$\mathcal{B}_{best}$
cfg25	$\mathcal{S}$+$\mathcal{TX}_{cnn}$+$\mathcal{TX}$+$\mathcal{CV}$	cfg37	=25+$\mathcal{B}_{best}$
cfg26	$\mathcal{S}$+$\mathcal{CV}_{cnn}$+$\mathcal{CV}$	cfg38	=26+$\mathcal{B}_{best}$
cfg27	$\mathcal{S}$+$\mathcal{CV}_{cnn}$+$\mathcal{CV}$+$\mathcal{TX}$	cfg39	=27+$\mathcal{B}_{best}$
cfg28	$\mathcal{S}$+$\mathcal{CV}_{cnn}$+$\mathcal{CV}$+$\mathcal{TX}_{cnn}$ +$\mathcal{TX}$	cfg40	=28+$\mathcal{B}_{best}$
cfg29	$\mathcal{S}$+$\mathcal{CV}_{cnn}$+$\mathcal{CV}$+$\mathcal{TX}_{cnn}$ +$\mathcal{TX}$+$\mathcal{TX}_{emb}$+$\mathcal{TX}_{stats}$	cfg41	=29+$\mathcal{B}_{best}$

Table 3.6: The highlighted row depicts the best results w.r.t. Brown Clusters ("*Brown Best*") (cfg10)

(a) "$\mathcal{S}$" stands for Standard features; **(b)** "CNN CV" for state of the art neural network computer vision); **(c)** "Standard CV" for traditional computer vision algorithms and techniques; **(d)** "Brown" ($xMcL$) the Brown cluster configuration (x=vocabulary size and c=number of clusters); **(e)** "Standard" means the classic NER features; **(f)** "CNN TX" for state of the art text classification; **(g)** "TX Embeedings" calculates the intersection rate of a given word embeeding and a common-sense set for a given NE class; **(h)** "Standard TX" for traditional text classification algorithms and techniques; **(i)** "CNN TX Statistics" computes the basic statistics per NE class ("sum", "max", "min" and "average") from predictions of **(f)** and **(g)**.

Results

The complete benchmark configuration has the following dimensions: $cfg \times (\mathcal{DS}_{train} + (\mathcal{DS}_{train} \times \mathcal{DS}_{test})) \times \mathcal{A}$; where cfg is the total of feature sets (i.e., distinct configurations), $\mathcal{D}_{train}$ is the total of training sets, $\mathcal{D}_{test}$ is the total of test sets and finally $\mathcal{A}$ is the total number of algorithms. This leads to the following number of experiments: $41 \times (4 + (4 \times 3)) \times 9 = 5.904$[9]. **Demystifying the impact of images and news**: Figure 3.2 shows the performance of CRF in different

[9] 41 experiment configurations, 4 training sets (Ritter, WNUT-15, WNUT-16 and WNUT-17), 3 test sets (WNUT-15, WNUT-16 and WNUT-17) and 9 NER architectures (DT, RF, CRF, CRF-PA, LSTM, B-LSTM+CRF, Char+B-LSTM+CRF and B-LSTM+CNN+CRF)

datasets/feature sets. The x-axis represents the different feature sets (Tables 3.4, 3.5 and 3.6), while y-axis average of F1-measure[10]. To highlight the impact of the different groups of features, we categorize F1's in four ascending scales, from worse to the best: *red*, *yellow*, *gray* and *green*. Some patterns w.r.t. the addition of images and text as input features are clearly observable. First, standard textual features ($\mathcal{TX}$) have often a much worse performance when compared to standard image features ($\mathcal{CV}$) as well as in the combination of both, as observed in the following sets `cfg02`×`cfg03`×`cfg04`, `cfg06`×`cfg07`×`cfg08` and `cfg15`×`cfg16`×`cfg17`. This is at some extend expected since *one-vs-all* strategy to classify *news* data is not an easy task. In this sense, a better solution might be taking into account probabilities instead of binary values. Moreover, we notice our improvement in the $\mathcal{TX}$ component (`cfg19`, `cfg20` and `cfg21`) outperform the similar features proposed by [78]. Among those, it is worth noting that the *text correlation* ($\mathcal{TX}_{stats}$) has a greater impact than any other textual feature. This is due to the higher level of abstraction when computing word embedding distances across *seeds* in a distance supervision fashion. Regarding the image detection component, introducing state-of-the-art computer vision algorithms ($\mathcal{CV}_{cnn}$) has also been beneficial to beat previous strategy ($\mathcal{CV}$), although without bringing major improvements as in the $\mathcal{TX}$. This is due to the *common-sense* rules proposed by [78] in this layer. **Brown Clusters**: Finally, the inclusion of tuned Brown clusters ($\mathcal{B}_{best}$, `cfg30-41`) along with proposed features shows to be beneficial to the performance. Overall, the best results were obtained from the concatenation of the previous and proposed features along with Brown clusters (`cfg41`). **Architectures**: We benchmark distinct NER architectures comparing the following feature sets: `cfg10` (weak baseline), `cfg04` [78] as *images-and-news* baseline and `cfg41` (HORUS).Table 3.7 presents results. As expected, CRFs and state-of-the-art NNs architectures performed best. The comparison shed light on the impact of our proposed features (best configuration, `cfg41`) when compared to the broadly implemented NER features (`cfg10`) and the architecture proposed by [78] (`cfg04`). We can see that overall the additional features introduced in this work clearly improves the performance of the majority of the models (DT, RF, CRF, B-LSTM+CRF) in all data sets.

Dataset		Decision Trees			Random Forest			CRF			B-LSTM CRF [116]			B-LSTM C+CRF [71]			B-LSTM C+CRF+CNN [132]		
cfg ⟶		10	04	41	10	04	41	10	04	41	10	04	41	10	04	41	10	04	41
Ritter	P	0.48	+2%	+4%	0.51	+1%	+24%	0.73	+5%	+7%	0.77	+1%	−3%	0.81	−5%	−1%	0.81	−5%	−5%
	R	0.49	+1%	+3%	0.48	−1%	−2%	0.58	−8%	−2%	0.63	+5%	+5%	0.59	+5%	+4%	0.62	+3%	+5%
	F	0.49	+1%	+3%	0.49	+4%	+7%	0.58	+2%	+7%	0.68	+1%	+1%	0.67	+1%	+1%	0.69	−1%	+1%
WNUT-15	P	0.49	+2%	+5%	0.52	+7%	+25%	0.72	+7%	+9%	0.72	−4%	−2%	0.77	−3%	−4%	0.78	−4%	−5%
	R	0.50	+0%	+5%	0.49	+0%	+1%	0.48	−1%	+6%	0.69	+1%	+1%	0.65	+2%	+2%	0.66	+2%	+2%
	F	0.50	+0%	+5%	0.50	+5%	+9%	0.56	+2%	+8%	0.68	+0%	+0%	0.69	+0%	−1%	0.71	−1%	−2%
WNUT-16	P	0.49	+1%	+6%	0.52	+14%	+23%	0.72	+7%	+9%	0.72	−4%	−2%	0.77	−3%	−3%	0.78	−4%	−6%
	R	0.50	+1%	+6%	0.48	+0%	+2%	0.48	−1%	+6%	0.69	+0%	+1%	0.65	+2%	+2%	0.66	+2%	+2%
	F	0.49	+1%	+6%	0.50	+5%	+10%	0.56	+2%	+8%	0.69	−1%	+0%	0.69	+0%	+0%	0.71	−1%	−2%
WNUT-17	P	0.44	+3%	+7%	0.47	+13%	+24%	0.76	+2%	+1%	0.76	−2%	−2%	0.76	+0%	−2%	0.77	−3%	−3%
	R	0.45	+4%	+6%	0.44	+3%	+4%	0.50	+0%	+5%	0.63	+1%	+1%	0.64	+0%	+1%	0.62	+1%	+1%
	F	0.44	+4%	+6%	0.45	+6%	+12%	0.60	+0%	+4%	0.67	+0%	+0%	0.69	+0%	−1%	0.67	+0%	−1%

Table 3.7: The performance measure's improvements (*green*) and decreases (*red*) in different datasets, feature sets (`cfg`) and architectures are represented in a color gradient of 5 points interval. 0% represents a tiny improvement i ($0.1\% \leq i \leq 0.99\%$), which is not representative, although technically not zero.

Sampling sets: Figure 3.3 depicts the impact of images and news on F1-measure for the best

[10] 3-fold cross-validation.

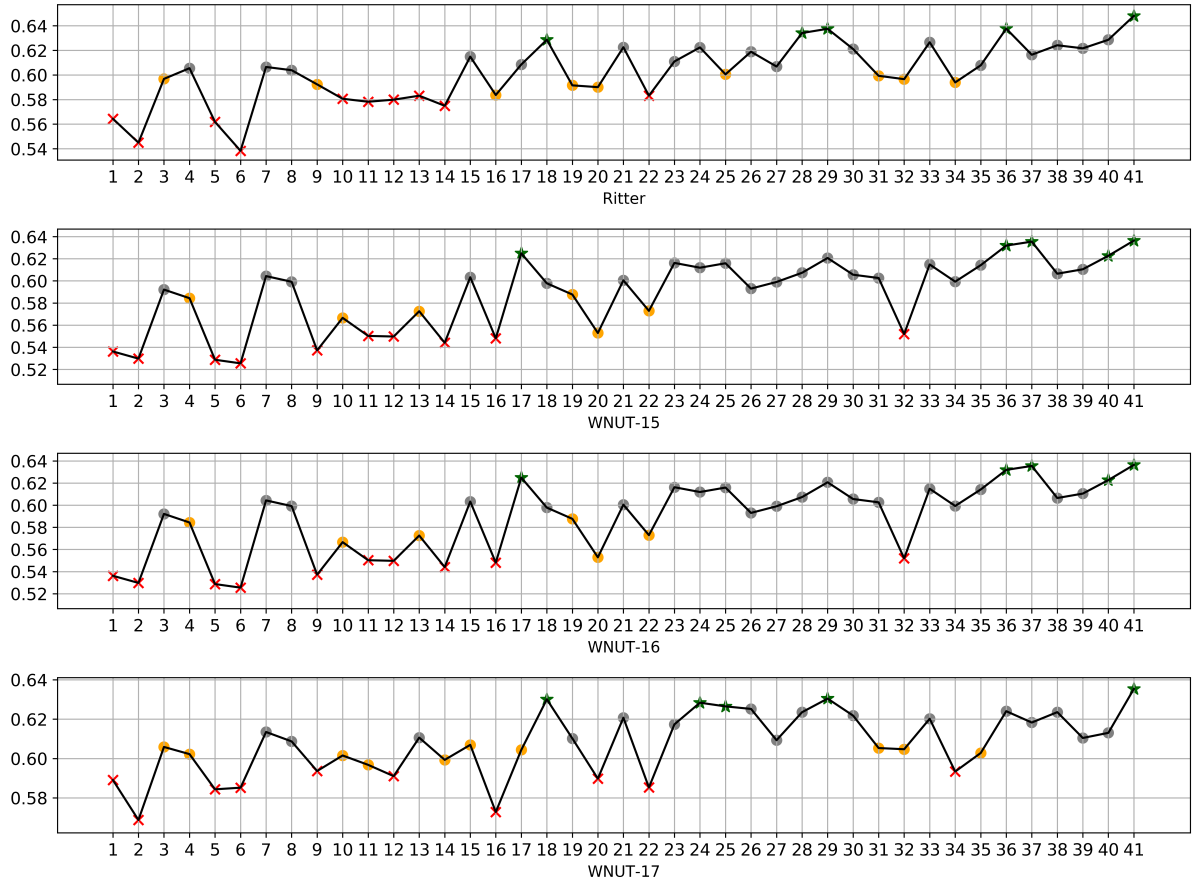


Figure 3.2: The CRF performance (`cfg` × F1) in different datasets/feature sets

architecture of a *weak* baseline (CRF) and a *strong* baseline (B-LSTM+CRF), defined according to previous experiments (Table 3.7). The results confirm that the proposed features consistently boost the performance of the models in the majority of the experiments. It is worth noting the substantial impact in the CRF-based model. Our proposed features (`cfg41`) improves *Lexical + Brown Cluster* and [78] in more than 90% of the cases (and at least similar in 100% of the cases).

The precision and recall trade-off: Moreover, we notice that a basic CRF architecture with the best feature configuration (`cfg41`) outperforms a state-of-the-art B-LSTM architecture w.r.t. *precision*. The same feature set also positively impacted *recall* of B-LSTM in all experiments.

Increasing training data: Finally we trained a B-LSTM+CRF architecture with an expanded set created merging all data sets. We removed duplication from the union of the respective *training*, *dev* and *test* sets, i.e., occurrences of overlap sentences. The results are presented in Table 3.8, which support the claim that we can go beyond lexical features and further investigate the use of images and news to benefit NER on noisy data.

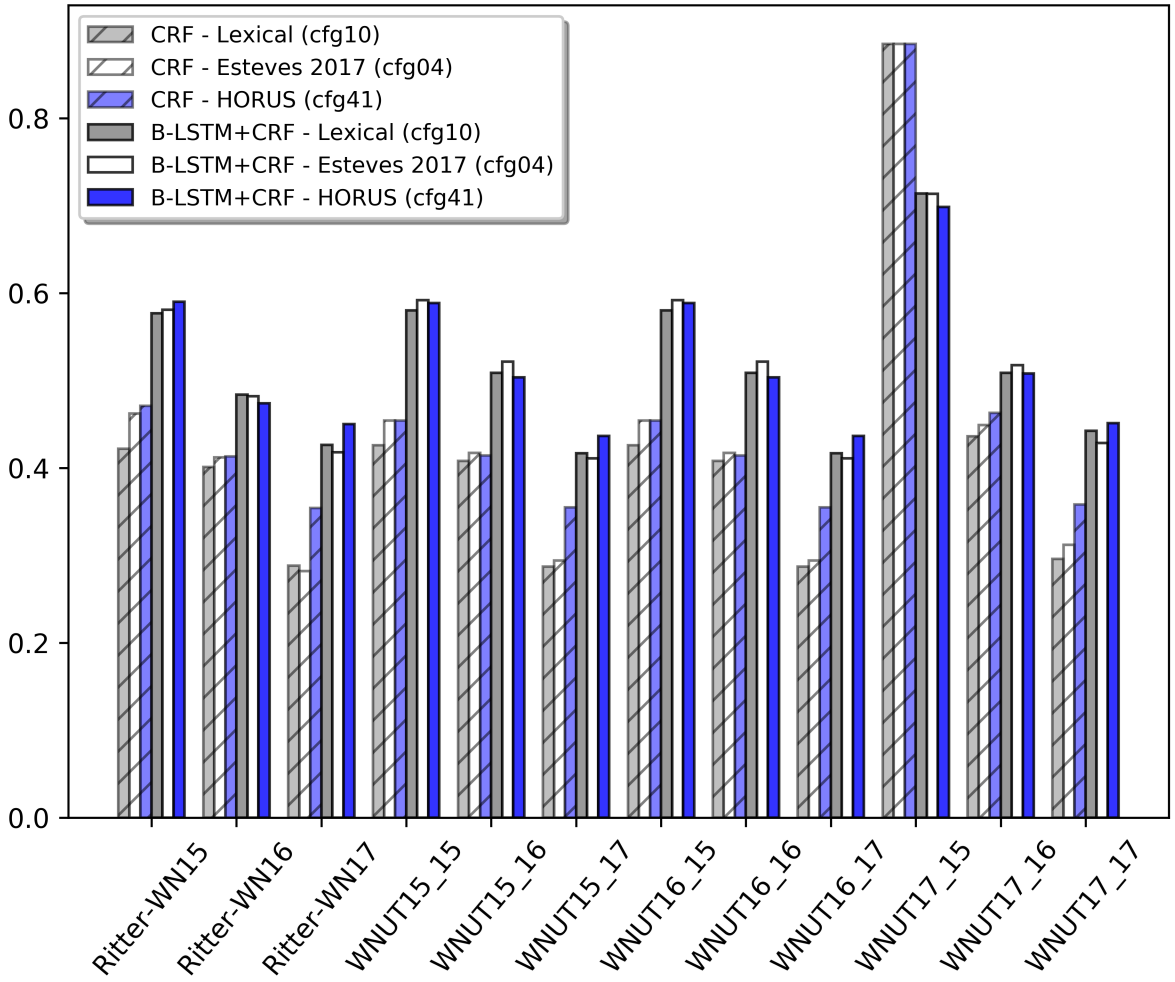


Figure 3.3: The positive impact of images and news through distinct training-test pairs sets. A comparison between the best *weak* (CRF) and the best *strong* (B-LSTM+CRF) baselines.

	+cfg10	+cfg04	+cfg41
B-LSTM+CRF	0.5217	0.5352 ↑	0.5376 ↑

Table 3.8: B-LSTM+CRF F1-measure with expanded training/dev/test data over different feature sets.

3.5 Summary

We proposed a novel multi-model architecture based on computer vision and text mining techniques to boost the NER task. This architecture, dubbed HORUS, has shown potential to overcome existing challenges at recognizing entities on noisy data. Therefore, given the nature of our work and its novelty, the outcomes of this work are of high relevance for NER on social media.

First, we designed a more straightforward architecture based on binary classifiers to detect

certain objects in images and classify a given textual input for a given text. This architecture led to motivating results, beating state-of-the-art in the Ritter dataset. After, we modified and extended the model to support neural network-based architectures. In a comprehensive study, we observed that features extracted from images are of high relevance, whereas textual features not necessary improve the task over regular NER features. However, the concatenation of the features help to maximize the performance of the model. We compared several NER algorithms along with the features we extract from HORUS, showing the benefit of the proposed method. Several gold-standard datasets have been tested. HORUS exhibited competitive results, even outperforming in some specific configurations. In traditional NER architectures (e.g. CRF), the proposed features have proved feasible to notably improve its overall model performance and, when compared to SOA, achieved a higher *precision*. SOA had improved in *recall*, but at expense of *precision*. However, when benchmarking the models across different training-test sets, the images and news also proved to be beneficial.

The advantages are summarized as follows:

1. A challenging (pre-processing) tasks, such as *text normalization* [80], is bypassed.
2. The proposed approach is language-agnostic.
3. It does not rely on *gazetteers*, *lookups* and *normalization* and also does not implement any encoded rules.
4. As a result of our experiments, we released to the community a word-level feature database for NER based on image and text. This database contains approx. 3 millions data features for more than 72.000 distinct tokens and has been explored over 5.904 experiments in different configurations.

As downside, we also shed light on existing problem which require further study, as follows:

1. The proposed architecture slows down the classification of named entities, performing considerably slower than existing solutions. This tends to decrease due to caching, but still might affects at production scale.
2. Using the Web as corpus implies in financial costs as well
3. Due to the existing noisy in the Web, performing text-classification from generic unseen web content is a challenge.

CHAPTER 4

Web Credibility

This chapter is dedicated to tackle yet one of the core challenges of this thesis, i.e., to assign credibility scores for information sources. The content of this chapter is based on the publications [31, 136].

With the growth of the internet, the number of *fake-news* online has been proliferating every year. The consequences of such phenomena are manifold, ranging from lousy decision-making process to bullying and violence episodes. Therefore, fact-checking algorithms became a valuable asset. To this aim, an important step to detect fake-news is to have access to a credibility score for a given information source. However, most of the widely used Web indicators have either been shut-down to the public (e.g., Google PageRank) or are not free for use (Alexa Rank). Further existing databases are short-manually curated lists of online sources, which do not scale. Finally, most of the research on the topic is theoretical-based or explore confidential data in a restricted simulation environment.

The results of this chapter provide an answer to the following research question:

> **RQ2**: How to calculate a credibility score for a given information source?

First, in Section 4.1, we present a motivating example illustrating the problem of assigning trustworthy scales for a given information source. To address research question RQ2, we devise WebCred, the first 100% open-source web-based credibility model. WebCred extracts source code metadata and computes scores of trustworthiness for a given website.

Next, Section 4.2 describes our approach. Overall, WebCred detects credibility patterns derived from metadata extracted from source code of websites. Afterwards, a comprehensive evaluation of the WebCred approach and analysis of the obtained results is presented in Section 4.3. Observed results suggest that WebCred is able to generalize well to unseen websites. Finally, Section 4.6 presents the closing remarks of this chapter. We summarize the contributions of this chapter as follows:

- A novel methodology to compute trustworthiness indicators for websites.

- A novel web credibility Framework named WebCred, which implements the concepts behind this methodology and is 100% open-source.
- An empirical evaluation to assess the effectiveness of WebCred for the web credibility task. Experiments are executed over the most famous datasets for the task: Microsoft and 3C Corpus.
- An updated release of the Microsoft Credibility dataset.

4.1 How Credible is a Website

With the enormous daily growth of the Web, the number of *fake-news* sources have also been increasing considerably [137]. This social network era has provoked a communication revolution that boosted the spread of misinformation, hoaxes, lies and questionable claims. The proliferation of unregulated sources of information allows any person to become an opinion provider with no restrictions. For instance, websites spreading manipulative political content or hoaxes can be persuasive. As introduced in previous sections, to tackle this problem, different *fact-checking* tools and frameworks have been proposed [138]. Yet an important underlying fact-checking step relies upon computing the credibility of sources of information, i.e. indicators that allow answering the question: *"How reliable is a given provider of information*?". Due to the obvious importance of the Web and the negative impact that misinformation can cause, methods to demote the importance of websites also become a valuable asset. In this sense the high number of new websites appearing at everyday [139], make straightforward approaches - such as *blacklists* and *whitelists* - impractical. Moreover, such approaches are not designed to compute credibility scores for a given website but rather to binary label them. Thus, they aim at detecting mostly "fake" (threatening) websites; e.g., *phishing detection*, which is out of scope of this work. Thus, open credibility models have a great importance, especially due to the increase of fake news being propagated. There is much research into credibility factors. However, they are mostly grouped as follows: (1) theoretical research on psychological aspects of credibility and (2) experiments performed over private and confidential users information, mostly from web browser activities (strongly supported by private companies). Therefore, while (1) lacks practical results (2) report *findings* which are not much appealing to the broad open-source community, given the non-open characteristic of the conducted experiments and data privacy. Finally, recent research on credibility has also pointed out important drawbacks, as follows:

1. Manual (human) annotation of credibility indicators for a set of websites is costly [140].
2. Search engine results page (SERP) do not provide more than few information cues (URL, title and snippet) and the dominant heuristic happens to be the search engine (SE) rank itself [140].
3. Only around 42.67% of the websites are covered by the credibility evaluation knowledge base, where most domains have a low credibility confidence [91]

Therefore, automated credibility models play an important role in the community - although not broadly explored yet, in practice. In this paper, we focus on designing computational models

to predict the credibility of a given website rather than performing sociological experiments or experiments with end users (simulations). In this scenario, we expect that a website from a domain such as `bbc.com` receives a higher trustworthiness score compared to one from `wordpress.com`, for instance.

4.2 The WebCred Approach

We propose a new supervised model to provide heuristics of a given website based on a sequence-to-sequence approach which encodes `URL` content. The task is to learn that official *news* websites should be more credible than opinions stated in web *blogs* and personal web sites which, in turn, should be more credible than web sites flagged by *blacklists*. In other words, heuristics that consequently could belittle or strengthen information in websites. Furthermore, we implement a set of grounded measures specific to infer the trustworthiness of a website.

We introduce a new paradigm similar to bag-of-words: "*bag-of-tags*" (BoT). The idea behind BoT is simple: to find patterns which may or may not be visible to an end-user. Thus, we expect to capture not only visual information, but also patterns hidden in the web page code which may or may not be related to the credibility of web sites. Apart from this method, we also explore different textual features (e.g., Vader Lexicon). Figure 4.1 summarizes the method in a nutshell.

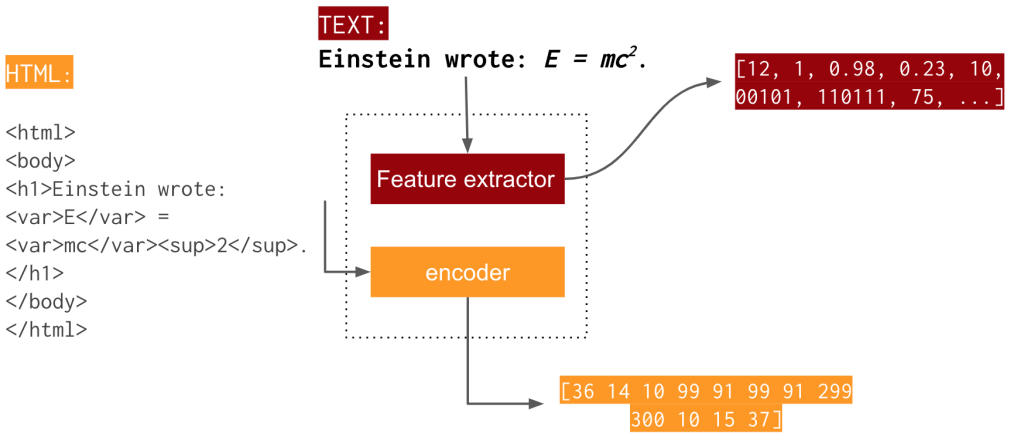


Figure 4.1: The concept of bag-of-tags: extracting information from HTML tags.

As result of our experiments and research, we released an *open-source* tool dubbed WebCred which calculates and provides heuristics to belittling or strengthening information in websites.

4.3 Experimental Setup

4.3.1 State-of-the-art (SOTA) Features

Recent research on credibility factors for web sites [99] have initially divided the features into the following logical groups:

1. **Content-based** (25 features): number of special characters in the text, spelling errors, web site category and etc..
   a) **Text** (20 features)
   b) **Appearance** (4 features)
   c) **Meta-information** (1 feature)
2. **Social-based** (12 features): Social Media Metadata (e.g., Facebook shares, Tweets pointing to a certain URL, etc.), Page Rank, Alexa Rank and similar.
   a) **Social Popularity** (9 features)
   b) **General Popularity** (1 feature)
   c) **Link structure** (2 features)

According to [99], a resultant number of 22 features (out of 37) were selected as most significant (10 for **content-based** and all **social-based** features). Surprisingly (but also following [93]), none from the sub-group **Appearance**, although studies have systematically shown the opposite, i.e., that visual aspects are one of the most important features [86, 92, 140].

In this picture, we claim the most negative aspect is the reliance on **Social-based** features. This dependency not only affects the final performance of the credibility model, but also implies in financial costs as well as presenting high discriminative capacity, adding a strong bias to the performance of the model[1]. The computation of these features relies heavily on external (e.g., Facebook API[2] and AdBlock[3]) and commercial libraries (Alchemy[4], PageRank[5], Alexa Rank[6]. Thus, engineering and financial costs are a must. Furthermore, popularity on Facebook or Twitter can be measured only by data owners. Additionally, vendors may change the underlying algorithms without further explanation. Therefore, also following Wawer et al. [100], in this paper we have excluded **Social-based** features from our experimental setup.

On top of that, Wawer et al. [100] incremented the model, adding features extracted from the General Inquirer (GI) Lexical Database, resulting in a vector of 183 extra categories, apart from

[1] authors applied ANOVA test confirming this finding
[2] https://developers.facebook.com/
[3] https://adblockplus.org/
[4] www.alchemyapi.com
[5] excepting for heuristic computations, calculation of PageRank requires crawling the whole Internet
[6] https://www.alexa.com/siteinfo

the selected 22 base features, i.e. total of 205 features (However, this is subject to contradictions. Please see Section 4.5 for more information).

4.3.2 Datasets

Website credibility evaluation

Microsoft Dataset [89] consists of thousands of URLs and their credibility ratings (five-point Likert Scale[7]), ranging from 1 ("very non-credible") to 5 ("very credible"). In this study, participants were asked to rate the websites as credible following the definition: "*A credible webpage is one whose information one can accept as the truth without needing to look elsewhere*". Studies by [99, 100] use this dataset for evaluation. **Content Credibility Corpus (C3)**[8] is the most recent and the largest credibility dataset currently publicly available for research [95]. It contains 15.750 evaluations of 5.543 URLs from 2.041 participants with some additional information about website characteristics and basic demographic features of users. Among many metadata information existing in the dataset, in this work we are only interested in the URLs and their respective five-point Likert scale, so that we obtain the same information available in the Microsoft dataset.

Fact-checking influence

In order to verify the impact of our web credibility model in a real use-case scenario, we ran a fact-checking framework to verify a set of input claims. Then we collected the sources (URLs) containing proofs to support a given claim. We used this as a dataset to evaluate our web credibility model.

The primary objective is to verify whether our model is able, on average, to assign *lower* scores to the websites that contain *proofs* supporting *claims* which are labeled as *false* in the FactBench dataset (i.e., the source is providing false information, thus should have a lower credibility score). Similarly, we expect that websites that support *positive* claims are assigned with higher scores (i.e., the source is supporting an accurate claim, thus should have a higher credibility score).

The (gold standard) input claims were obtained from the **FactBench** dataset[9], a multilingual benchmark for the evaluation of fact validation algorithms. It contains a set of RDF[10] models (10 different relations), where each model contains a singular fact expressed as a *subject-predicate-object* triple. The data was automatically extracted from DBpedia and Freebase KBs, and manually curated in order to generate true and false examples.

The website list extraction was carried out by DeFacto [31], a fact-checking framework designed for RDF KBs. DeFacto returns a set of websites as pieces of evidence to support its prediction

[7] https://en.wikipedia.org/wiki/Likert_scale
[8] also known as Reconcile Corpus
[9] https://github.com/DeFacto/FactBench
[10] https://www.w3.org/RDF/

(true or false) for a given input claim.

4.3.3 Final Features

We implemented a set of **Content-based** features (Section 4.3.1) adding more lexical and textual based features. **Social-based** features were not considered due to financial costs associated with paid APIs. The final set of features for each website w is defined as follows:

1. *Web Archive*: the temporal information w.r.t. cache and freshness. Δ_b and Δ_e correspond to the temporal differences of the first and last 2 updates, respectively. Δ_a represents the age of w and finally Δ_u represents the temporal difference for the last update to today. γ is a penalization factor when the information is obtained from the *domain* of w (w_d) instead w.

$$f_{arc}(w) = ([\frac{1}{\log(\Delta_b \times \Delta_e)} + \log(\Delta_a) + \frac{1}{\Delta_u}]) \times \gamma$$

2. *Domain*: refers to the (encoded) domain w (e.g. org)

3. *Authority*: searches for authoritative keywords within the page HTML content w_c (e.g., contact email, business address, etc..)

4. *Outbound Links*: searches the number of different outbound links in $w \wedge w_d \in d$, i.e., $\sum_{n=1}^{P} \phi(w_c)$ where P is the number of web-based protocols.

5. *Text Category*: returns a vector containing the probabilities P for each pre-trained category c of w w.r.t. the sentences of the website w_s and page title w_t: $\sum_{s=1}^{w_s} \gamma(s) \frown \gamma(w_t)$. We trained a set of binary multinomial Naive Bayes (NB) classifiers, one per class, as follows: *business*, *entertainment*, *politics*, *religion*, *sports* and *tech*.

6. *Text Category - LexRank*: reduces the noisy of w_b by classifying only top N sentences generated by applying LexRank [141] over w_b ($S\prime = \Gamma(w_b, N)$), which is a graph-based text summarizing technique: $\sum_{s\prime=1}^{S\prime} \gamma(s\prime) \frown \gamma(w_t)$.

7. *Text Category - LSA*: similarly, we apply Latent Semantic Analysis (LSA) [142] to detect semantically important sentences in w_b ($S\prime = \Omega(w_b, N)$): $\sum_{s\prime=1}^{S\prime} \gamma(s\prime) \frown \gamma(w_t)$.

8. *Readability Metrics*: returns a vector resulting of the concatenation of several R readability metrics [143]

9. *SPAM*: detects whether the w_b or w_t are classified as spam: $\psi(w_b) \frown \psi(w_t)$

10. *Social Tags*: returns the frequency of social tags in w_b: $\bigcup_{i=1}^{R} \varphi(i, w_b)$

11. *OpenSources*: returns the open-source classification (x) for a given website:

$$x = \begin{cases} 1, & \text{if } w \in \mathcal{O} \\ 0, & \text{if } w \notin \mathcal{O} \end{cases}$$

12. *PageRankCC*: PageRank information computed through the CommonCrawl[11] Corpus

13. *General Inquirer* [101]: a 182-lenght vector containing several lexicons

14. *Vader Lexicon*: lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments

15. *HTML2Seq*: we introduce the concept of *bag-of-tags*, where similarly to *bag-of-words*[12] we group the HTML tag occurrences in each web site. We additionally explore this concept along with a sequence problem, i.e. we encode the tags and evaluate this considering a window size (offset) from the header of the page.

4.4 Experiments

Previous research proposes two **application settings** w.r.t. the classification itself, as follows: (A.1) casting the credibility problem as a classification problem and (A.2) evaluating the credibility on a five-point Likert scale (regression). In the classification scenario, the models are evaluated both w.r.t. the 2-classes as well as 3-classes. In the 2-classes scenario, websites ranging from 1 to 3 are labeled as "low" whereas 4 and 5 are labeled as "high" (credibility). Analogously, in the 3-classes scenario, websites labeled as 1 and 2 are converted to "low", 3 remains as "medium" while 4 and 5 are grouped into the "high" class.

We first explore the impact of the *bag-of-tags* strategy. We encode and convert the tags into a sequence of tags, similar to a sequence of sentences (looking for opening and closing tags, e.g., <a>and). Therefore, we perform document classification over the resulting vectors. Figures 4.2(a) ,4.2(b), 4.2(c) and 4.2(d) show results of this strategy for both 2 and 3-classes scenarios. The x-axis is the log scale of the paddings (i.e., the offset of HTML tags we retrieved from w, ranging from 25 to 10.000). The charts reveal an interesting pattern in both gold-standard datasets (Microsoft Dataset and C3 Corpus): the first tags are the most relevant to predict the credibility class. Although this strategy does not achieve state of the art performance[13], it presents reasonable performance by just inspecting website metadata: F1-measures = 0.690 and 0.571 for the 2-classes and 3-classes settings, respectively. However, it is worth mentioning that the main advantage of this approach lies in the fact that it is language agnostic (while current research focuses on English) as well as less susceptible to overfitting.

We then evaluate the performance of the textual features (Section 4.3.3) isolated. Results for the 2-classes scenario are presented as follows: Figure 4.3(a) highlights the best models performance

[11] http://commoncrawl.org/
[12] https://en.wikipedia.org/wiki/Bag-of-words_model
[13] F1 measures = 0.745 (2-classes) and 0.652 (3-classes).

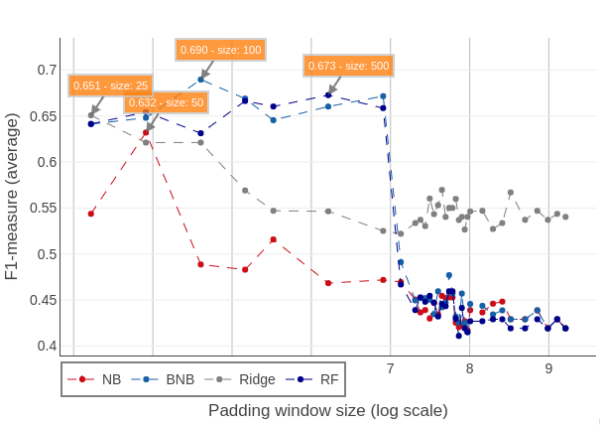
Microsoft Dataset (Gradient Boosting, $K = 25$)			
Class	Precision	Recall	F1
low	0.851	0.588	0.695
high	0.752	0.924	0.829
weighted	0.794	0.781	0.772
micro	0.781	0.781	0.781
macro	0.801	0.756	0.762
C3 Corpus (AdaBoost, $K = 75$)			
Class	Precision	Recall	F1
low	0.558	0.355	0.434
high	0.732	0.862	0.792
weighted	0.675	0.695	0.674
micro	0.695	0.695	0.695
macro	0.645	0.609	0.613

Table 4.1: Text+HTML2Seq features (2-class): best classifier performance

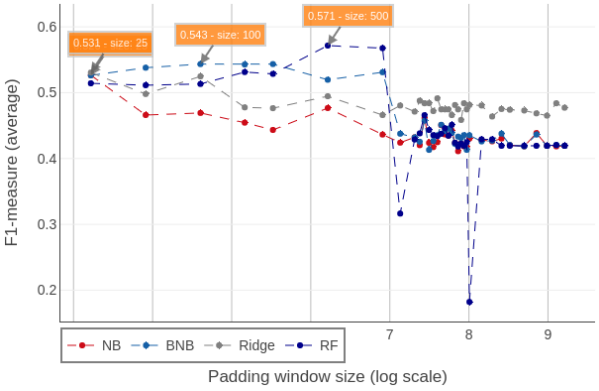
using textual features only. While this as a single feature does not outperform the lexical features, when we combine the *bag-of-tags* approach (predictions of probabilities for each class) we boost the performance (F1 from 0.738 to 0.772) and outperform state of the art (0.745), as shown in Figure 4.3(b). Tables 4.1, 4.2 and 4.3 show detailed results for both datasets (2-classes, 3-classes and 5-classes configurations, respectively). For 5-class regression, we found that the *best pad = 100* for the Microsoft dataset and *best pad = 175* for the C3 Corpus. We preceded the computing of both classification and regression models with feature selection according to a percentile of the highest scoring features (*SelectKBest*). We tested the choice of 3, 5, 10, 25, 50 75 and K=100 percentiles (thus, no selection) of features and did not find a unique K value for every case. It is worth noticing that in general it is easy to detect high credible sources (F1 for "high" class around 0.80 in all experiments and both datasets) but recall of "low" credible sources is still an issue.

Table 4.4 shows statistics on the data generated by the fact-checking algorithm. For 1500 claims, it collected pieces of evidence for over 27.000 websites. Table 4.5 depicts the impact of the credibility model in the fact-checking context. We collected a small subset of 186 URLs from the FactBench dataset and manually annotated[14] the credibility for each URL (following the Likert scale). The model corrected labeled around 80% of the URLs associated with a positive claim and, more importantly, 70% of non-credible websites linked to false claims were correctly identified. This helps to minimize the number of non-credible information providers that contain information that supports a *false* claim.

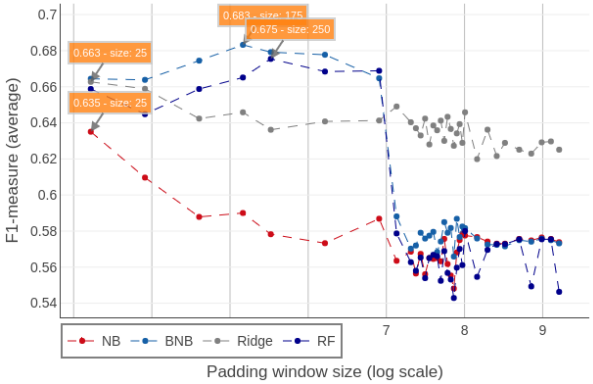
[14] By four human annotators. In the event of a tie we exclude the URL from the final dataset.



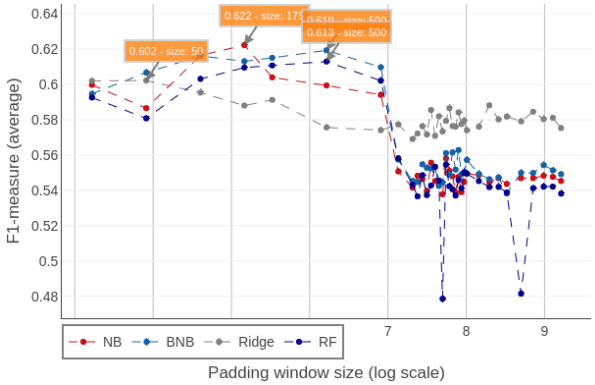
(a) HTML2Seq (F1): Microsoft dataset 2-classes



(b) HTML2Seq (F1-measure): Microsoft dataset 3-classes



(c) HTML2Seq (F1): C3 Corpus 2-classes



(d) HTML2Seq (F1): C3 Corpus 3-classes

Figure 4.2: HTML2Seq (F1-measure) over different padding sizes.

4.5 Discussion

Although the relevance of the research topic, very few work have proposed open work prototypes and/or models to perform credibility evaluation. The available datasets and metadata are scarce resources. In fact, reproducibility is still one of the cornerstones of science and scientific projects [144]. In the following, we list some relevant issues encountered while performing our experiments:

Experimental results: this gap is also observed w.r.t. results reported by [99], which is acknowledged by [100], despite numerous attempts to replicate experiments. Authors [100] believe this is due to the lack of parameters and hyperparameters explicitly cited in the previous research [99].

Microsoft dataset: presents inconsistencies. Although all the web pages are cached (in theory) in order to guarantee a deterministic environment, the dataset - in its original form[15] - has a

[15] The original dataset can be downloaded from http://research.microsoft.com/en-us/projects/

Microsoft Dataset (Gradient Boosting, $K = 75$)			
Class	Precision	Recall	F1
low	0.567	0.447	0.500
medium	0.467	0.237	0.315
high	0.714	0.916	0.803
weighted	0.626	0.662	0.626
micro	0.662	0.662	0.662
macro	0.583	0.534	0.539
C3 Corpus (AdaBoost, $K = 100$)			
Class	Precision	Recall	F1
low	0.143	0.031	0.051
medium	0.410	0.177	0.247
high	0.701	0.916	0.794
weighted	0.583	0.660	0.598
micro	0.660	0.660	0.660
macro	0.418	0.375	0.364

Table 4.2: Text+HTML2Seq features (3-class): best classifier performance

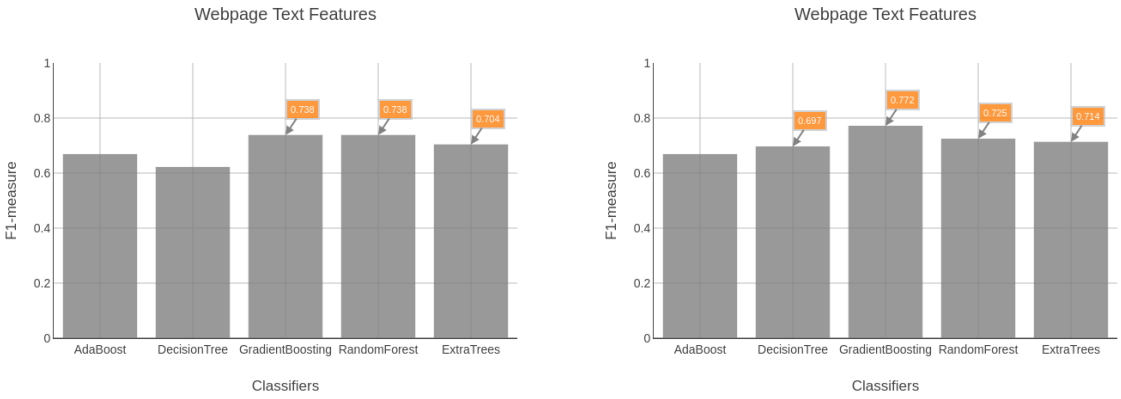
Microsoft Dataset					
model	K	R^2	RMSE	MAE	EVar
SVR	3	0.232	0.861	0.691	0.238
Ridge	3	0.268	0.841	0.683	0.269
C3 Corpus					
model	K	R^2	RMSE	MAE	EVar
SVR	25	0.096	0.939	0.739	0.102
Ridge	25	0.133	0.920	0.750	0.134

Table 4.3: Text+HTML2Seq: regression measures (5-class). Selecting top K lexical features.

FactBench (Credibility Model)		
label	**claims**	**sites**
true	750	14.638
false	750	13.186
-	1500	27.824

Table 4.4: FactBench: Web sites collected from claims.

`credibility/`



(a) Textual Features.

(b) Textual+HTML2Seq (best padding) Features.

Figure 4.3: Evaluating distinct classifiers in the 2-classes setting (Microsoft dataset): increasing almost +3% (from 0.745 to 0.772) on average F1 (Gradient Boosting). Feature selection performed with ANOVA *SelectKBest* method, K=0.25.

FactBench (Sample - Human Annotation)				
label	**claims**	**sites**	**non-cred**	**cred**
true	5	96	57	39
false	5	80	48	32
-	10	186	105	71
FactBench (Sample - Credibility Model)				
label	**non-cred**	**%**	**cred**	**%**
true	40	0.81	31	0.79
false	34	0.70	24	0.75

Table 4.5: FactBench Dataset: analyzing the performance of the credibility model in the fact-checking task.

number of problems, as follows: (a) web pages not physically cached (b) URL not matching (dataset links *versus* cached files) (c) Invalid file format (e.g., PDF). Even though these issues have also been previously identified by related research [99] it is not clear what the URLs for the final dataset (i.e., the support) are nor where this new version is available.

Contradictions: w.r.t. the divergence of the importance of visual features have drawn our attention [93] and [88, 92] which corroborate to the need of more methods to solve the web credibility problem, in practice. The main hypothesis that supports this contradiction relies on the fact that feature-based credibility evaluation eventually ignites cat-and-mouse play between scientists and people interested in manipulating the models. In this case, *reinforcement learning* methods pose as a good alternative for adaptation.

Proposed features: The acknowledgement made by authors in [100] that "*solutions based purely on external APIs are difficult to use beyond scientific application and are prone for manipulation*" confirming the need to exclude **social features** from research of [99] contradicts itself. In the course of experiments, authors mention the usage of all features proposed by [99]: "*Table 1 presents regression results for the dataset described in [13] in its original version (37 features) and extended with 183 variables from the General Inquirer (to 221 features)*".

Therefore, due to the number of relevant issues presented w.r.t. reproducibility and contradiction of arguments, the comparison to recent research becomes more difficult. In this work, we solved the technical issues in the Microsoft dataset and released a new fixed version[16]. Also, since we need to perform evaluations in a deterministic environment, we cached and released the websites for the C3 corpus. After scraping, 2.977 URLs were used (out of 5.543). Others were left due to processing errors (e.g., 404). The algorithms and its hyperparameters and further relevant metadata are available through the MEX Interchange Format [145]. By doing this, we provide a computational environment to perform safer comparisons, being engaged in recent discussions about mechanisms to measure and enhance the reproducibility of scientific projects [146].

4.6 Summary

In this work, we discuss existing alternatives, gaps and current challenges to tackle the problem of web credibility. More specifically, we focused on automated models to compute a credibility factor for a given website. This research follows the former studies presented by [99, 100] and presents several contributions. First, we propose different features to avoid the financial cost imposed by external APIs in order to access website credibility indicators. This issue has become even more relevant in the light of the challenges that have emerged after the shutdown of Google PageRank, for instance. To bridge this gap, we have proposed the concept of bag-of-tags. Similar to [100], we conduct experiments in a highly-dimensional feature space, but also considering web page metadata, which outperforms state of the art results in the 2-classes and 5-classes settings. Second, we identified and fixed several problems on a gold standard dataset for web credibility (Microsoft), as well as indexed several web pages for the C3 Corpus. Finally, we evaluate the impact of the model in a real fact-checking use-case. We show that the proposed model can help in belittling and supporting different websites that contain evidence of true and false claims, which helps the very challenging fact verification task. As future work, we plan to explore deep learning methods over the HTML2Seq module.

[16] more information at the project website: https://github.com/DeFacto/WebCredibility

CHAPTER 5

DeFacto: An Automated Fact-Checking Framework

This chapter is dedicated to solve the core challenge of this thesis, i.e., to compute the level of veracity for a given claim. The content of this chapter is based on the publications [31, 147–150].

The results of this chapter provide an answer to the following research question:

> **RQ1**: How to determine the veracity of a given claim?

First, in Section 5.1, we present a motivating example illustrating the problem of fact-checking. To address research question RQ1, we consistently developed DeFacto, a fact-checking framework firstly proposed by Lehmann et al [151].

Next, Section 5.3 highlights the problem and describes the solution: DeFacto, a fact-checking framework able to perform claim validation over both structured and unstructured claims. Afterwards, a comprehensive evaluation of the DeFacto approach and analysis of the obtained results is presented in Section 5.3.3. Next, we extend the framework to perform verification over natural language claims (5.4). Last, but not least, in Section 5.5 we dissect methods to improve performance of the most critical component of an automated fact-checking approach: the evidence retrieval component. Finally, Section 5.6 presents the closing remarks of this chapter. We summarize the contributions of this chapter as follows:

- The development and extension of DeFacto, a RDF fact-checking framework .
- The extension of DeFacto to increase its coverage on the validation task w.r.t. the allowed predicates.
- The extension of DeFacto to support triple ranking.
- The improvement of its architecture w.r.t. new evidence extraction methods.
- An empirical evaluation to assess the effectiveness of DeFacto for the fact-checking task in

the most important fact-checking challenge. Experiments performed validate the viability of the framework on real-use cases scenarios.

5.1 Automating The Fact-Checking Task

Given a *claim*, the task of fact-checking consists at defining whether what is being said is close to *false* or *true*, based on external *evidence* which supports the decision. The automation of this process involves a pipeline which performs the task by (often) assigning a credibility score. Most recent ideas in fact-checking revolve around automation of the human (or journalist) fact-checking process. Currently, this is broadly translated into a 3-step process which involves 1) collecting articles about the claim, 2) selecting prospective evidence and finally 3) performing a final judgment.

For the structured fact-validation task, we consider a claim as follows: a claim c existing in a Knowledge Base K (denoted as $c \in K$) is represented by a triple (s, p, o), where s is the subject *uri*, p is the predicate (or *relation*) *uri* and o is the object *uri*.

Claim: "dbr:Albert_Einstein","dbo:award","dbr:Nobel_Prize_in_Physics"
Evidence: "*The Nobel Prize in Physics 1921 was awarded to Albert Einstein for his services to Theoretical Physics, and especially for his discovery of the law of the...*"

For the unstructured fact-validation task, we consider a claim as a sentence represented in natural language.

Claim : "That '70s Show is a sitcom"
Evidence: "*That '70s Show is an American television period sitcom that originally aired on Fox from August 23, 1998, to May 18, 2006. The series focused on the lives of a group of teenage friends living in the fictional suburban town of Point Place, Wisconsin, from May 17, 1976, to December 31, 1979.*"

Thorne et al. in their baseline on FEVER 2018 [152] give a good summary of a standard automated fact-checking pipeline. In general, this automation is achieved by training the machine learning models which benefit from information retrieval method and existing annotated datasets. This pipeline is broken down into the following steps:

1. Document Retrieval: a first step focuses on obtaining relevant documents [50, 59]. A *document retrieval* module selects documents from a large corpus that are related to a given claim. The relatedness is determined by selecting a matching metric. Throne et al. [58] uses DrQA [153] for document retrieval that selects documents from Wikipedia[1] corpus based on TF-IDF. An alternative approach could be using web search APIs [12, 31, 60, 154, 155] (e.g., Bing API[2]) for collecting related webpages from the internet. This approach is a better approximation of

[1] https://www.wikipedia.org/
[2] https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/

the human fact-checking process, since human fact-checkers do not restrict their research to a single corpus (like Wikipedia).

2. Evidence Selection: The next step of the pipeline is to select potential evidences from the documents or collection of sentences that we retrieved in the first step. This step is called *evidence selection* or *proof extraction*. This component does not differentiate a piece of evidence that refutes the claim, from one that supports it. The main goal at this phase is to collect sentences that could potentially be used to run inference on the veracity of the claim.

3. Source Classification: This step, although not strictly necessary in order to perform the task, has major importance in order to weight all of the extracted claims (proofs) according to the trustworthiness of hubs and authorities [60, 156].

4. Claim Classification: The last step is called *claim classification*. As the name suggests, in this step the model makes final decision on the claim by taking all the collected information into account [52] and produces scores for each evidence, and then makes the final decision on aggregation of all the scores, [58] do textual entailment on the claim and a concatenation of all pieces of evidence.

In the following we introduce our modules to perform fact-checking over both structured as well as unstructured claims.

5.2 Automation Challenges

On the computational side, there are different fundamental challenges w.r.t. the execution of the underlying tasks in this pipeline. We extend the definition of [15] (items 1 and 4) by highlighting two more challenge (items 2 and 3) we argue that are crucial to bridge the gap between *automated fact-checking* approaches and human fact-checkers, as follows:

1. to understand what one says [15] (NLU)
2. to have the ability to generate equivalent arguments and counter-arguments (NLG)
3. to have the ability to distinguish credible and non-credible information sources (Credibility)
4. to have the ability to obtain plausible evidence [15] (Argumentation Mining)

First, algorithms need to have the ability to understand what is being said. This refers to a specific research area called Natural Language Understanding (NLU), which comprehends several NLP sub-tasks, such as Named Entity Recognition (NER), and Part-of-speech (POS), for instance. Although significant leaps and bounds have been made in this context, these technologies are far from human performance, especially in more noisy contexts, such as microblogs [157].

Second, algorithms need to process similar content accordingly, i.e., to collect equivalent and related content which are potentially useful in the fact-checking process. This is part of a branch in NLP called Natural Language Generation (NLG) [158]. This is of utmost importance in

order to have a broader coverage when checking claims. In structured fact-checking [31], this is a crucial step towards interpreting and transforming the input *claim* into natural language. Moreover, in free text, for instance, the sentences "*he was born in USA.*" and "*he is a Yankee.*" share the same meaning. Given a claim "*he is American.*", algorithms should be capable to perceive that, in this context, "USA" and "Yankee" are synonyms for "American", thus the information extraction phase should generate similar content automatically in order to increase *recall*.

Third, the level of trustworthiness of authorities and sources must be checked and taken into account. This has been studied in a topic known as *(Web) Credibility* [156]. This step is important both in assessing the credibility of sources isolated, as well as when confronting opposite claims made by two or more authorities [159]. For instance, consider a scenario of researching information about dietary which potentially helps in certain disease treatment. One may find websites from reputable agencies (e.g., NCI USA) alongside sites from private organizations which sell dietary supplements (which may serve as advice hub whilst pointing to their own products). Discerning which sources are trustworthy and which are not is a crucial step forward automated fact-checking systems [156].

Finally, besides collecting sufficient evidence for asserting a given claim, explicit and implicit relations among extracted arguments (as well as possible counter-arguments) should (ideally) be labeled and linked. This is studied in another branch of NLP called Argumentation Mining [160]. The generated graph allows achieving a richer level of metadata in order to better perform the final fact-checking task [161]. In the following section, we categorize and introduce existing fact-checking approaches.

5.3 The DeFacto Approach: Validating RDF Triples

DeFacto is the only *open-source* approach which supports simple counter-evidence searching[3] and also implements metrics to compute the trustworthiness of web sources. The major shortcoming of the system is its dependence on search engines, leading to a higher cost of deployment. However, this disadvantage is common to all *triple validation* architectures. Another major disadvantage is its dependence on predicate expansion methods. Current approaches implement either 1) hard-coded verbalization and rules (fixed or ontology-based), which naturally restricts scalability; 2) use distant supervision methods which very often have a sub-optimal *precision*; or 3) use external linguistic corpora (e.g., lexical databases) to obtain similar words (e.g., synonyms) to a given *predicate*. It can be observed that these methods are rather crude and hence the verbalizations generated are of a low quality, making verbalization an unsolved task. In the following sections, we describe each task.

[3] we do not rely on complex arguments, but rather in simple evidence that can potentially negate an input claim. For instance, "*James was born in Seattle*" as a counter-evidence to the input claim "`James, born, Paris`

5.3.1 Proposed Solution

Given an input claim c, we first perform language verbalisation to generate similar claims, as follows:

1. Natural Language Generation: the function $\gamma(s, p, o, \mathcal{L})$ takes as input a triple (s, p, o) and the set of languages $\mathcal{L}$ and returns a matrix V containing a set of triples. The function γ is calculated as follows:

$\gamma(s, p, o, \mathcal{L}) = [\phi(s, l_1) \times \Gamma(p, l_1) \times \phi(o, l_1)] \cup [\phi(s, l_2) \times \Gamma(p, l_2) \times \phi(o, l_2)], \cdots, \cup [\phi(s, l_n) \times \Gamma(p, l_n) \times \phi(o, l_n)]$ where:

1. $\phi(x, l_i)$ returns a set of m labels $(x_1, x_2, ..., x_m)$ that are similar to the label of the resource x ($s \in \mathcal{S}$ and $o \in \mathcal{O}$) which is extracted from the `rdfs:labels` predicate for a given language $l_i \in \mathcal{L}$.

2. $\Gamma(p, l_i)$ returns a set of verbalized patterns $\mathcal{P}$ for a given predicate p and a language $l_i \in \mathcal{L}$.

The function $\gamma(s, p, o, \mathcal{L})$ returns a matrix V with number of elements $(\mathcal{S} \times \mathcal{P} \times \mathcal{O} \times \mathcal{L})$.

2. Information Retrieval: Afterwards a set of search engine queries (we call them *metaqueries*) are formalized by concatenating each i^{th}term $(v_i^1, v_i^2, v_i^3) \in V$ without specific search engine parameters (i.e., excepting from the option *market* which defines the location of the retrieved websites and is defined by l, no further parameter is set). The complete retrieval process is carried out by issuing these several queries (the total number of elements of V) to a regular search engine. In the next step, the highest ranked web pages associated to each *metaquery* are retrieved (*evidence sources candidates*).

3. Web Page Evaluation: Once all the web pages have been retrieved, they are processed further, as follows: a) HTML content is extracted; b) *fact confirmation* methods are applied to the content extracted (in essence, the algorithm decides whether the web page contains a natural language formulation of the input fact). In addition to *fact confirmation*, the system computes different indicators for the trustworthiness of a web page[4].

Figure 5.1 shows the DeFacto pipeline in a nutshell, from the verbalisation stage to the proof extraction phase.

4. Final score: In addition to finding and displaying sources and their indicators, DeFacto also outputs a general discrete confidence value for the input fact that ranges between 0 and 1.0. DeFacto uses features from textual evidence combined with trustworthiness measures to compute the score [31]. It indicates the confidence level of the model for a given input claim. The higher the value, the more likely the input claim is *true*.

[4] *Topic Terms*, *Topic Majority in the Web*, *Topic Majority in Search Results* and *Topic Coverage*

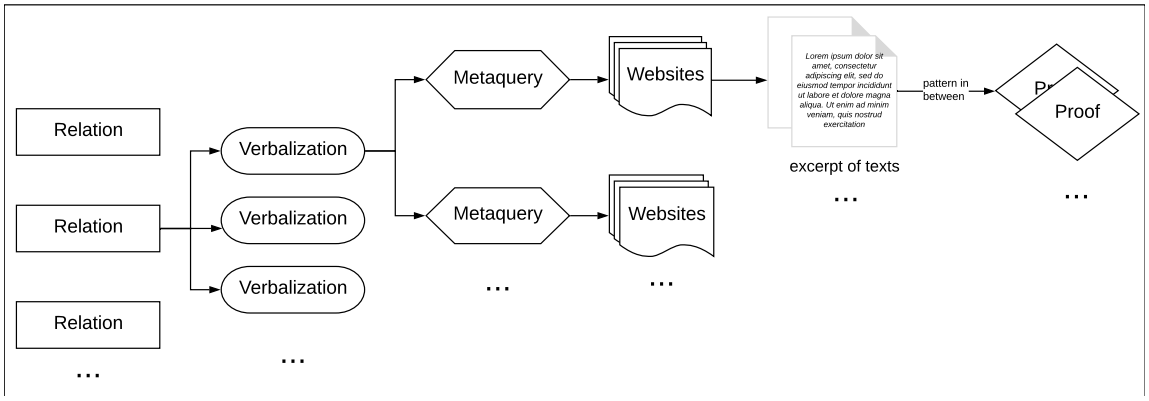


Figure 5.1: The *Proof Extraction* pipeline (*Pattern Verbalization* and *Proof Searching*): extracting excerpt of texts (*proof*) which represent a verbalization for a given *triple*.

5.3.2 Features

In order to obtain an estimate of the confidence that there is sufficient evidence to consider the input triple to be true, we chose to train a supervised machine learning algorithm. Similar to the above presented classifier for fact confirmation, this classifier also requires computing a set of relevant features for the given task. In the following, we describe those features and why we selected them.

First, we extend the score of single proofs to a score of web pages as follows: When interpreting the score of a proof as the probability that a proof actually confirms the input fact, then we can compute the probability that at least one of the proofs confirms the fact. This leads to the following stochastic formula[5], which allows us to obtain an overall score for proofs scw on a web page w:

$$scw(w) = 1 - \prod_{pr \in prw(w)} (1 - fc(pr)) \,. \tag{5.1}$$

In this formula, fc (fact confirmation) is the classifier which takes a proof pr as input and returns a value between 0 and 1. prw is a function taking a web page as input and returning all possible proofs contained in it.

Combination of Trustworthiness and Textual Evidence In general, we assume that the trustworthiness of a web page and the textual evidence found in it are orthogonal features. Naturally, web pages with high trustworthiness and a high score for its proofs should increase our confidence in the input fact. We thus combine trustworthiness and textual evidence as features for the underlying machine learning algorithm. This is achieved by multiplying both criteria and then

[5] To be exact, it is the complementary even to the case that none of the proofs do actually confirm a fact.

using their sum and maximum as two different features:

$$F_{fsum}(t) = \sum_{w \in s(t)} (f(w) \cdot scw(w)) \tag{5.2}$$

$$F_{fmax}(t) = \max_{w \in s(t)} (f(w) \cdot scw(w)) \tag{5.3}$$

In this formula, f can be instantiated by all three trustworthiness measures: topic majority on the the Web (tm_{web}), topic majority in search results (tm_{sr}) and topic coverage (tc). s is a function taking a triple t as argument, executing the search queries and returning a set of web pages. Using the formula, we obtain 6 different features for our classifier, which combine textual evidence and different trustworthiness measures.

Other Features In addition to the above described combinations of trustworthiness and fact confirmation, we also defined other features:

1. The **total number of proofs** found.
2. The **total number of proofs found above a relevance threshold** of 0.5. In some cases, a high number of proofs with low scores is generated, so the number of high scoring proofs may be a relevant feature for learning algorithms. The thresholds mimics a simple classifier.
3. The **total evidence score**, i.e., the probability that at least one of the proofs is correct, which is defined analogously to *scw* above:

$$1 - \prod_{pr \in prt(t)} (1 - fc(pr)) \,. \tag{5.4}$$

   where $prt(t)$ is a function returning all proofs found for t from all web pages.
4. The **total evidence score above a relevance threshold** of 0.5. This is an adaption of the above formula, which considers only proofs with a confidence higher than 0.5.
5. The **total hit count**, i.e., search engine's estimate of the number of search results for an input query. The total hit count is the sum of the estimated number of search results for each query send by DeFacto for a given input triple.
6. A **domain and range verification**: If the subject of the input triple is not an instance of the domain of the property of the input triple, this violates the underlying schema, which should result in a lower confidence in the correctness of the triple. This feature is 0 if both domain and range are violated, 0.5 if exactly one of them is violated and 1 if there is no domain or range violation. At the moment, we are only checking whether the instance is asserted to be an instance of a class (or one of its subclasses) and do not use reasoning for performance reasons.
7. **Statistical triple evidence**: Usually certain classes have a higher probability to cooccur as type of subject and object in a given triple, e.g., there might be a higher probability that

instances of `dbo:Person` and `dbo:Film` are related via triples than for instance `dbo:Insect` and `dbo:Film`. This observation also holds for the cooccurence of classes and properties, both for the types in subject and object position. This kind of semantic relatedness allows for computing a score for the statistical evidence STE of a triple $t = (s, p, o)$ by

$$STE(t) = \max_{\substack{c_s \in cls(s) \\ o_s \in cls(o)}} (PMI(c_s, c_o) + PMI(c_s, p) + PMI(p, c_o)) \quad (5.5)$$

where cls denotes the types of the resource and PMI denotes the Pointwise Mutual Information, which is a measure of association and defined by

$$PMI(a, b) = log\left(N \cdot \frac{occ(a, b)}{occ(a) \cdot occ(b)}\right) \quad (5.6)$$

using $occ(e)$ as number of occurrences of a given entity e in a specific position of a triple and N as the total number of triples in the knowledge base.

5.3.3 Experimental Setup

The dataset used in our experiments is named FactBench, a multilingual dataset for the evaluation of fact validation algorithms. All facts in FactBench are scoped with a timespan in which they were true, enabling the validation of temporal relation extraction algorithms. FactBench currently supports English, German and French. The current release V1 is freely available (MIT License) at `http://github.com/AKSW/FactBench`. FactBench consists of a set of RDF models. Each one of the 1500 models contains a singular fact and the time period in which it holds true. Each fact was checked manually by three independent human quality raters. In addition, the FactBench suite contains the SPARQL and MQL queries used to query Freebase[6] and DBpedia, a list of surface forms for English, French and German as well as the number of incoming and outgoing links for the English wikipedia pages. FactBench provides data for 10 well-known relations. The data was automatically extracted from DBpedia and Freebase. A detailed description on what facts the benchmark contains is shown in Figure 5.2. The granularity of FactBench time information is year. This means that a timespan is an interval of two years, e.g., 2008 - 2012. A time point is considered as a timespan with the same start and end year, e.g., 2008 - 2008.

FactBench is divided in a training and a testing set (of facts). This strict separation avoids the overfitting of machine learning algorithms to the training set, by providing unseen test instances.

The aim of our evaluation was three-fold. We wanted to quantify how well/much *a*) DeFacto can distinguish between correct and wrong facts; *b*) DeFacto is able to find correct time points or time periods for a given fact and if the year frequency distribution (1900 vs. 2013) does influence the accuracy; and *c*) the use of multi-lingual patterns boost the results of DeFacto with respect to fact validation and date detection. In the following, we describe how we set up our evaluation system, present the experiments we devised and discuss our findings.

[6] Since there are no incremental releases from Freebase we include the crawled training data

Figure 5.2: FactBench provides data for 10 relations. The data was automatically extracted from Wikipedia (DBpedia respectivly) and Freebase

In a first step, we computed all feature vectors for the training and test sets. DeFacto relies heavily on web requests, which are not deterministic (i.e., the same search engine query does not always return the same result). To achieve deterministic behavior and to increase the performance as well as reduce load on the servers, all web requests were cached. The DeFacto runtime for an input triple was on average slightly below four seconds per input triple[7] when using caches.

We stored the features in the ARFF file format and employed the WEKA machine learning toolkit[8] for training different classifiers. In particular, we were interested in classifiers which can handle numeric values and output confidence values. Naturally, confidence values for facts such as, e.g., 95%, are more useful for end users than just a binary response on whether DeFacto considers the input triple to be true, since they allow a more fine-grained assessment. We selected popular machine-learning algorithms satisfying those requirements. We focused our experiments on the 10 relations from FactBench. The system can be extended easily to cover more properties by extending the training set of BOA to those properties. Note that DeFacto itself is also not limited to DBpedia or Freebase, i.e., while all of its components are trained on these datasets, the algorithms can be applied to arbitrary URIs and knowledge bases.

5.3.4 Fact Scoring

For this evaluation task, we used each FactBench training set to build an independent classifier. We then used the classifier on the corresponding test set to evaluate the built model on unseen data. The results on this task can be seen in Table 5.1. The J48 algorithm, an implementation of the C4.5 decision tree – shows the most promising results. Given the challenging tasks, F-measures up to 84.9% for the *mix* test set appear to be very positive indicators that DeFacto can be used to effectively distinguish between true and false statements, which was our primary

[7] The performance is roughly equal on server machines and notebooks, since the web requests dominate.
[8] http://www.cs.waikato.ac.nz/ml/weka/

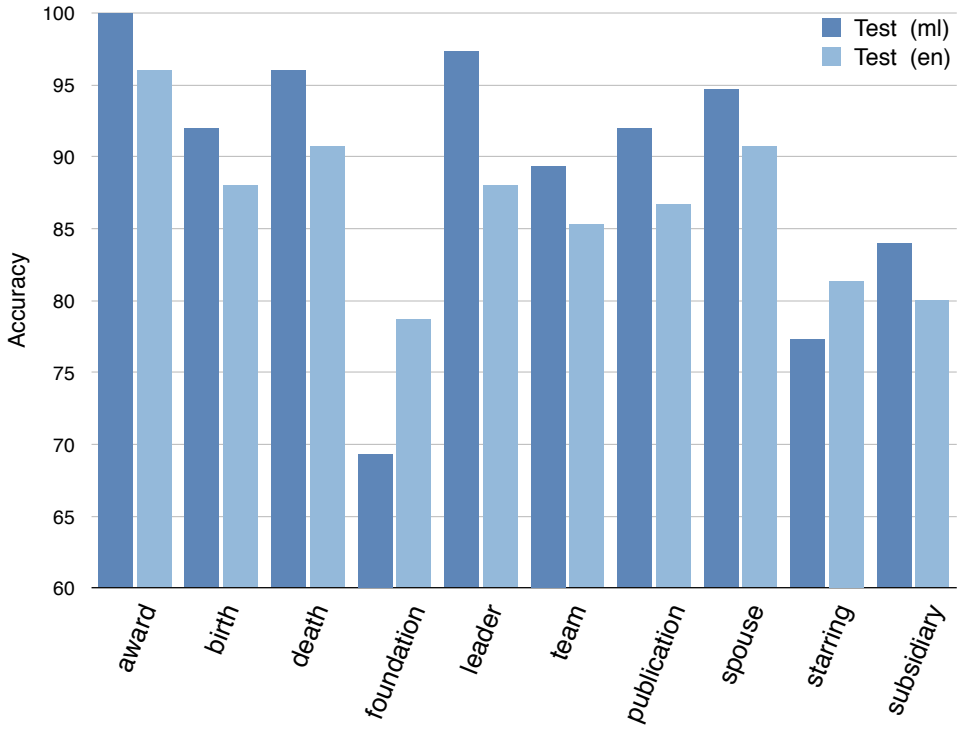


Figure 5.3: Accuracy results for learned J48 *mix* classifier on correct subset of the test set. The abbreviation *ml* indicates that multi-lingual (English, French, German) search results and surface forms were used, *en* is limited to English only.

	Domain						Range					
	C	P	R	F_1	AUC	RMSE	C	P	R	F_1	AUC	RMSE
J48	89.7%	0.898	0.897	0.897	0.904	0.295	90.9%	0.909	0.909	0.909	0.954	0.271
SimpleLogistic	89.0%	0.890	0.890	0.890	0.949	0.298	88.0%	0.880	0.880	0.880	0.946	0.301
NaiveBayes	81.2%	0.837	0.812	0.808	0.930	0.415	83.3%	0.852	0.833	0.830	0.933	0.387
SMO	85.4%	0.861	0.854	0.853	0.854	0.382	83.3%	0.852	0.833	0.830	0.833	0.409
	DomainRange						Property					
	C	P	R	F_1	AUC	RMSE	C	P	R	F_1	AUC	RMSE
J48	91.0%	0.910	0.910	0.910	0.953	0.270	70.8%	0.786	0.708	0.687	0.742	0.427
SimpleLogistic	88.9%	0.889	0.889	0.889	0.950	0.296	64.9%	0.653	0.649	0.646	0.726	0.460
NaiveBayes	84.5%	0.861	0.845	0.843	0.935	0.380	61.3%	0.620	0.613	0.608	0.698	0.488
SMO	83.6%	0.853	0.836	0.834	0.836	0.405	64.6%	0.673	0.646	0.632	0.646	0.595
	Random						Mix					
	C	P	R	F_1	AUC	RMSE	C	P	R	F_1	AUC	RMSE
J48	90.9%	0.910	0.909	0.909	0.933	0.283	84.9%	0.850	0.849	0.849	0.868	0.358
SimpleLogistic	87.8%	0.879	0.878	0.878	0.954	0.293	80.2%	0.810	0.802	0.799	0.880	0.371
NaiveBayes	84.1%	0.851	0.841	0.839	0.942	0.375	78.7%	0.789	0.787	0.787	0.867	0.411
SMO	84.3%	0.864	0.843	0.841	0.843	0.396	76.9%	0.817	0.769	0.756	0.754	0.480

Table 5.1: Classification results for FactBench test sets (C = correctness, P = precision, R = recall, F_1 = F_1 Score, AUC = area under the curve, RMSE = root mean squared error).

evaluation objective. In general, DeFacto also appears to be stable against the various negative test sets given the F_1 values ranging from 89.7% to 91% for the *domain*, *range*, *domainrange*

and *random* test set. In particular, the algorithms with overall positive results also seem less affected by the different variations. On the *property* test set, in our opinion the hardest task, we achieved an F_1 score of 68.7%. Due to the results achieved, we use J48 as the main classifier in DeFacto and, more specifically, its results on the mix sets as this covers a wide range of scenarios. We observe that the learned classifier has an error rate of 3% for correct facts, but fails to classify 55.3% of the false test instances as incorrect.

We also performed an evaluation to measure the performance of the classifier for each of the relations in FactBench. The results of the evaluation are shown in Figure 5.3. We used the precision of the main classifier (J48 on the *mix* models) on the correct subset for this figure.[9] The average precision for all relations is 89.2%. The worst precision for an individual relation, i.e., 69%, is achieved on the *foundation* relation, which is by far the least frequent relation on the Web with respect to search engine results.

5.3.5 Date Scoring

To estimate time scopes, we first needed to determine appropriate parameters for this challenging task. To this end, we varied the context size from 25, 50, 100 and 150 characters to the left and right of the proofs subject and object occurrence. Additionally, we also varied the used languages which is discussed in more detail in Section 5.3.6. The final parameter in this evaluation was the normalization approach. We used the occurrence (number of occurrences of years in the context for all proofs of a fact), the domain and range approach. We performed a grid search for the given parameters on the *correct* train set. As performance measures we choose precision[10] P (shown in Equation 5.7), recall R (shown in Equation 5.8) and F-measure, defined as $F_1 = 2 * \frac{P*R}{P+R}$.

$$P = \frac{|relevant\ \ years \cap retrieved\ \ years|}{|retrieved\ \ years|} \tag{5.7}$$

$$R = \frac{|relevant\ \ years \cap retrieved\ \ years|}{|relevant\ \ years|} \tag{5.8}$$

If for example, for a single fact the correct time period is 2008 (a time point), the F_1 score is either 0 or 1. However, if the correct time period is 2011 – 2013 and the retrieved results are 2010 – 2013, we would achieve a precision $P = \frac{3}{4}$ (three of the four retrieved years are correct) and a recall $R = 1$ (all of the relevant years were found), resulting in an F_1 score of $\frac{6}{7}$.

The final results for the train set are shown in Table 5.2. Please note that it is out of scope of this paper to decide whether a given property requires a time period or a time point. As expected, facts with time point show a higher F_1 measure as facts with time period. Calculating the average F_1 score for the individual relations leads to $F_1 = 70.2\%$ for time points and $F_1 = 65.8\% F_1$ for relations associated with time periods. The relations performing well on fact scoring also appear to be better suited for year scoping, e.g., the *award* relation. In general, the training results show that the *domain* normalization performs best and the optimal context size varies for each relation. We now applied the learned parameters for each relation on the FactBench *correct* test

[9] We are using the correct subset, since some negative examples are generated by replacing properties. For those, it would not be clear, which property they refer to.

[10] Finding no year candidate for a given fact only influences the recall.

subset. The results are shown in Table 5.3. The average F_1 score decreases by 2.5% to 67.7% for time points and 4.6% to 61.2% for time period relations compared to the train set. Since it is not possible to determine a correct time point or time period for all facts (the context does not always include the correct year(s)) we also calculated DeFacto's accuracy. We define the accuracy *acc* for a time period *tp* as follows:

$$acc(tp) = \begin{cases} 1 & \text{if } tp_{from} \textit{ is correct} \wedge tp_{to} \textit{ is correct} \\ 0 & \text{otherwise.} \end{cases} \tag{5.9}$$

The average accuracy for time point (from and to are equal) relations is 76%. Since for time periods we have to match both start and end year, which aggravates this task significantly, we achieved an accuracy of 44% on this dataset. Finally, we wanted to see if DeFacto's performance is influenced by how recent a fact is. We grouped the time intervals in buckets of 10 years and plotted the proportion of correctly classified facts within this interval. We did this for the multilingual as well as the English-only setting of DeFacto. The results are shown in Figure 5.4. In general, all values are between 80% and 100% for the English version and between 93% and 100% for the mulit-lingual version. While there is some variation, no obvious correlation can be observed, i.e., DeFacto appears to be able to handle recent and older facts. In this figure, it is interesting to note that the multilingual setting appears to be more stable and perform better. We performed a paired t-test using all 750 facts and obtained that the improvement of the multilingual setting is statistically very significant.

	occurrence							global							domain						
Set	C	P	R	F	MRR	P_{75}	A	C	P	R	F	MRR	P_{75}	A	C	P	R	F	MRR	P_{75}	A
award$_{en}$	25	100	98.7	**99.3**	100	74	100	25	98.6	97.3	**98**	100	74	98.6	100	100	98.7	**99.3**	100	74	100
award$_{ml}$	25	100	98.7	**99.3**	100	74	100	25	100	98.7	**99.3**	100	74	100	25	100	98.7	**99.3**	100	74	100
birth$_{en}$	25	83.3	80	81.6	92.9	69	87	50	91.7	88	**89.8**	91.8	70	94.3	50	76.4	73.3	74.8	91.8	70	78.6
birth$_{ml}$	50	93.2	92	92.6	96.6	73	94.5	25	94.6	93.3	**94**	96.2	73	95.9	25	89.2	88	88.6	96.2	73	90.4
death$_{en}$	50	74.3	73.3	73.8	85.7	69	79.7	25	61.1	58.7	59.9	86.6	68	64.7	25	80.6	77.3	**78.9**	86.6	68	85.3
death$_{ml}$	25	77.3	77.3	77.3	86.8	75	77.3	25	66.7	66.7	66.7	86.8	75	66.7	25	84	84	**84**	86.8	75	84
foundation$_{en}$	150	14.1	12	12.9	56.6	28	32.1	150	17.2	14.7	15.8	56.6	28	39.3	150	25	21.3	**23**	56.6	28	57.1
foundation$_{ml}$	25	16.4	13.3	14.7	57.3	23	43.5	150	21.7	20	20.8	46.7	41	36.6	150	26.1	24	**25**	46.7	41	43.9
publication$_{en}$	100	58.3	56	57.1	77.1	63	66.7	150	60.3	58.7	**59.5**	72.8	67	65.7	100	51.4	49.3	50.3	77.1	63	58.7
publication$_{ml}$	25	70.8	68	69.4	86	68	75	150	74.7	74.7	**74.7**	79.4	72	77.8	50	60	60	60	81.8	70	64.3
starring$_{en}$	25	64.4	38.7	48.3	90.4	35	82.9	50	67.9	48	**56.3**	83.7	40	90	100	59.3	46.7	52.2	75.9	46	76.1
starring$_{ml}$	25	59.6	45.3	51.5	87.5	44	77.3	50	58.1	48	52.6	80.4	48	75	100	62.7	56	**59.2**	75.5	57	73.7
subsidiary$_{en}$	100	63.5	44	52	81.9	45	73.3	50	63	38.7	47.9	86.5	39	74.4	150	64.8	46.7	**54.3**	80.7	46	76.1
subsidiary$_{ml}$	25	70.8	45.3	**55.3**	87.6	43	79.1	25	68.8	44	53.7	87.6	43	76.7	25	70.8	45.3	**55.3**	87.6	43	79.1
spouse$_{en}$	100	67.5	68	67.7	75	53	50.9	25	75.5	64.4	69.5	55.6	37	78.4	25	77.1	65.2	**70.6**	55.6	37	78.4
spouse$_{ml}$	25	69.6	66.5	68	70.4	49	59.2	25	70.8	65.6	68.1	70.4	49	55.1	25	75.2	67.2	**71**	70.4	49	61.2
nbateam$_{en}$	100	54.2	47.4	50.6	68.6	44	34.1	100	57.8	47	51.9	68.6	44	34.1	150	59.1	48.4	**53.2**	61.1	53	28.3
nbateam$_{ml}$	50	60.2	58.1	59.1	69.1	58	25.9	100	62.1	55.4	58.6	58.4	63	23.8	25	65.2	58.7	**61.8**	67.8	53	32.1
leader$_{en}$	100	42.6	65.1	51.5	70.7	55	41.8	100	42.6	63.1	50.9	70.7	55	41.8	100	46.7	64.4	**54.1**	70.7	55	43.6
leader$_{ml}$	100	53.6	75.4	62.6	72.6	72	44.4	100	53.3	75.6	62.5	72.6	72	44.4	100	55.9	76.7	**64.7**	72.6	72	45.8
timepoint$_{en}$	25	61	48	**53.7**	86.7	277	78	25	60.2	47.3	53	86.7	277	76.9	100	57.8	50	53.6	78.6	317	71
timepoint$_{ml}$	25	65.9	56.7	**60.9**	86.8	326	78.2	25	64.1	55.1	59.3	86.8	326	76.1	150	61.6	58.2	59.9	74.7	373	70.2
timeperiod$_{en}$	100	54.7	60.2	57.3	70	152	42.8	100	54.9	60.3	57.4	70	152	42.8	100	58.7	60.6	**59.7**	70	152	44.7
timeperiod$_{ml}$	100	59	67.2	62.8	63.4	198	38.9	100	59.4	67.5	63.2	63.4	198	39.4	100	63	69	**65.9**	63.4	198	40.9
all$_{en}$	50	61.3	56.2	58.6	86.8	496	72	25	64	54	58.6	88.5	460	75.4	100	62.7	58.1	**60.3**	82	543	67.6
all$_{ml}$	25	67.1	63.2	65.1	87.8	568	70.1	25	66.3	62.4	64.3	87.8	568	68.7	100	66.1	65.2	**65.7**	80.4	635	65.4

Table 5.2: Overview of the time-period detection task for the FactBench training set with respect to the different normalization methods. *ml* (multi-lingual) indicates the use of all three languages (en,de,fr).

$\text{Set}^{context}_{language}$	P	R	F	MRR	C_S	C_E	P_{75}	Acc
award^{100}_{en}	93.3	93.3	93.3	100	70	-	75	93.3
award^{25}_{ml}	93.3	93.3	93.3	100	70	-	75	93.3
birth^{50}_{en}	77.8	74.7	76.2	81.6	56	-	69	81.2
birth^{25}_{ml}	93.2	92	92.6	93.3	69	-	73	94.5
death^{25}_{en}	72	72	72	84.5	54	-	69	78.3
death^{25}_{ml}	81.3	81.3	81.3	87.1	61	-	74	82.4
$\text{foundation}^{150}_{en}$	22.2	18.7	20.3	66.1	14	-	20	70
$\text{foundation}^{150}_{ml}$	20.3	18.7	19.4	48.1	14	-	33	42.4
$\text{publication}^{150}_{en}$	62	58.7	60.3	77.8	44	-	68	64.7
$\text{publication}^{150}_{ml}$	67.6	66.7	67.1	75.5	50	-	74	67.6
$\text{starring}^{50}_{en}$	57.1	48	52.2	87.1	36	-	44	81.8
$\text{starring}^{100}_{ml}$	61.4	57.3	59.3	73.6	43	-	60	71.7
$\text{subsidiary}^{150}_{en}$	60.7	49.3	54.4	79.3	37	-	53	69.8
$\text{subsidiary}^{25}_{ml}$	70.2	53.3	60.6	87.5	40	-	50	80
spouse^{25}_{en}	69.2	59	63.6	-	34	35	34	76.5
spouse^{25}_{ml}	73.7	61.4	67	-	36	36	42	59.5
team^{150}_{en}	52.7	42.7	47.2	-	25	16	51	23.5
team^{25}_{ml}	59.9	49.6	54.3	-	28	16	45	26.7
leader^{100}_{en}	46.3	60.8	52.5	-	29	29	56	44.6
leader^{100}_{ml}	55	71.5	62.2	-	38	37	72	45.8
$\text{timepoint}^{25}_{en}$	62	52	56.6	85.8	273	273	356	76.7
$\text{timepoint}^{25}_{ml}$	66.9	60.6	63.6	87.1	318	318	404	78.7
$\text{timeperiod}^{100}_{en}$	55.7	55.6	55.7	-	92	82	159	41.5
$\text{timeperiod}^{100}_{ml}$	59.6	60.1	59.8	-	102	91	195	38.5
all^{100}_{en}	58.2	54.4	56.2	-	375	365	563	62
all^{100}_{ml}	61.5	59.6	60.5	-	414	403	634	61

Table 5.3: Overview of the *domain*-normalization on the FactBench test set. *ml* (multi-lingual) indicates the use of all three languages (en,de,fr). $\text{C}_{(S|E)}$ shows the number of correct start and end years, P_{75} is the number of time-periods possible to detect correctly and A is the accuracy on P_{75}.

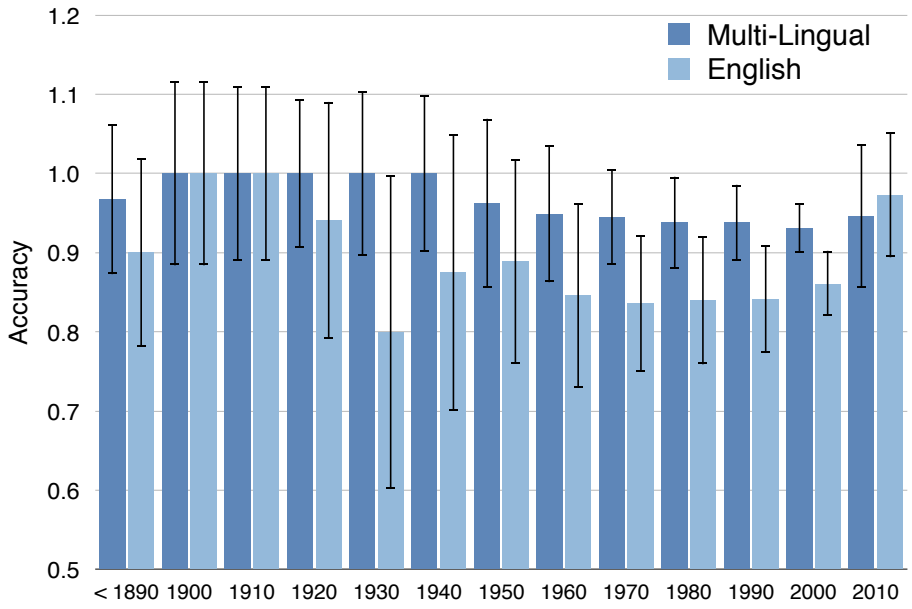


Figure 5.4: A plot showing the proportion of correctly classified facts (y-axis) for the FactBench *mix-correct*-test-set using the J48 classifier. The time intervals (x-axis) are buckets of ten years, e.g., 1910 stands for all years from 1910 to 1919. Results for the multilingual and English-only setting of DeFacto are shown.

	C	P	R	F_1	AUC	RMSE
J48	83.4%	0.834	0.834	**0.834**	0.877	0.361
SimpleLogistic	80.6%	0.811	0.806	0.804	0.884	0.368
NaiveBayes	78.1%	0.788	0.781	0.782	0.872	0.428
SMO	78.6%	0.816	0.786	0.777	0.773	0.463

Table 5.4: Classification results for FactBench mix test set on English language only.

5.3.6 Effect of Multi-lingual Patterns

The last question we wanted to answer in this evaluation is how much the use of the multi-lingual patterns boosts the evidence scoring as well as the date scoping. For the fact scoring we trained different classifiers on the *mix* training set. We only used English patterns and surface forms to extract the feature vectors. As the results in Table 5.4 on the test set show, J48 is again the highest scoring classifier, but is outperformed by the multi-lingual version shown in Table 5.1 by 1.5% F_1 score.

The detailed analysis for the different relations in Figure 5.3 indicates a superiority of the multi-lingual approach. We also performed the grid search as presented in Section 5.3.5 for English patterns and surface forms only. As shown in Table 5.2 the multi-lingual date scoping approach outperforms the English one significantly on the training set. The multi-lingual version achieved an average 4.3% on the time point and a 6.5% better F_1 measure on time period relations. The difference is similar on the test set, where the difference is 6.5% for time points

and 6.9% for time period relations. Finally, as shown in Figure 5.4, the English version performs equally well on recent data, but performs worse for less recent dates, which is another indicator that the use of a multilingual approach is preferable to an English-only setting.

5.3.7 Discussion

In this paper, we presented DeFacto, a multilingual and temporal approach for checking the validity of RDF triples using the Web as corpus. In more detail, we explicated how multi-lingual natural-language patterns for formal relations can be used for fact validation. In addition, we presented an extension for detecting the temporal scope of RDF triples with the help of pattern and frequency analysis. We support the endeavour of creating better fact validation algorithms (and to that end also better relation extraction and named entity disambiguation systems) by providing the full-fledged benchmark FactBench. This benchmarks consists of one training and several test sets for fact validation as well as temporal scope detection. We showed that our approach achieves an F_1 measure of 84.9% on the most realistic fact validation test set (FactBench *mix*) on DBpedia as well as Freebase data. The temporal extension shows a promising average F_1 measure of 70.2% for time point and 65.8% for time period relations. The use of multi-lingual patterns increased the fact validation F_1 by 1.5%. Moreover, it raised the F_1 for the date scoping task of up to 6.9%. Of importance is also that our approach is now fit to be used on non-English knowledge bases.

5.4 The DeFactoNLP Approach: Validating Unstructured Claims

After designing a framework to validate RDF triples we then extended it and conceived a system that can not only automatically assess the veracity of simple and structured but also complex and unstructured claims. Instead of the Web, which is costly to query and collect documents, we adapted the framework to retrieve evidence supporting the decision from Wikipedia. The Wikipedia documents whose Term Frequency-Inverse Document Frequency (TFIDF) vectors are most similar to the vector of the claim and those documents whose names are similar to those of the named entities (NEs) mentioned in the claim are identified as the documents which might contain evidence. One of the major limitations of our first framework is the dependency on natural language generation tools (BOA), which rely on supervised learning methods. In practice, a few number of predicates (in a sentence) are supported, which is a bottleneck.

To bridge this gap, we feed the sentences from the retrieved documents into textual entailment recognition module. This module calculates the probability of each sentence supporting the claim, contradicting the claim or not providing any relevant information to assess the veracity of the claim. In the next steps we detail the implementation.

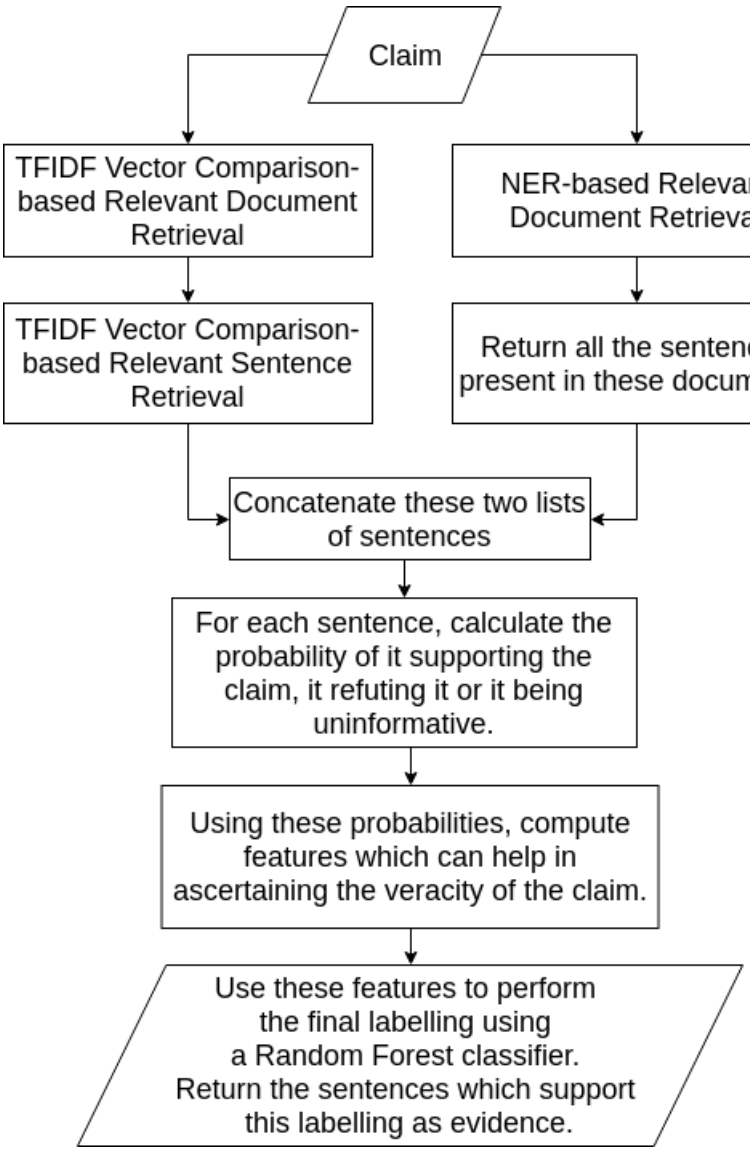


Figure 5.5: The main steps of our extended approach

5.4.1 Proposed Solution

Our approach has four main steps: Relevant Document Retrieval, Relevant Sentence Retrieval, Textual Entailment Recognition and Final Scoring and Classification. Given a claim, Named Entity Recognition (NER) and TFIDF vector comparison are first used to retrieve the relevant documents and sentences. The relevant sentences are then supplied to the textual entailment recognition module that returns a set of probabilities. Finally, a Random Forest classifier [162] is employed to assign a label to the claim using certain features derived from the probabilities returned by the entailment model. The proposed architecture is depicted in Figure 5.5.

5.4.2 Retrieval of Relevant Documents and Sentences

We used two methods to identify which Wikipedia documents may contain relevant evidences. Information about the NEs mentioned in a claim can be helpful in determining the claim's veracity. In order to get the Wikipedia documents which describe them, the first method initially uses the Conditional Random Fields-based Stanford NER software [163] to recognize the NEs mentioned in the claim. Then, for every NE which is recognized, it finds the document whose name has the least Levenshtein distance [164] to that of the NE. Hence, we obtain a set of documents which contain information about the NEs mentioned in a claim. Since all of the sentences in such documents might aid the verification, they are all returned as possible evidences.

The second method used to retrieve candidate evidences is identical to that used in the baseline system [165] and is based on the rationale that sentences which contain terms similar to those present in the claim are likely to help the verification process. Directly evaluating all of the sentences in the dump is computationally expensive. Hence, the system first retrieves the five most similar documents based on the cosine similarity between binned unigram and bigram TFIDF vectors of the documents and the claim using the DrQA system [153]. Of all the sentences present in these documents, the five most similar sentences based on the cosine similarity between the binned bigram TFIDF vectors of the sentences and the claim are finally chosen as possible sources of evidence. The number of documents and sentences chosen is based on the analysis presented in the aforementioned work by [165].

The sets of sentences returned by the two methods are combined and fed to the textual entailment recognition module described in Section 5.4.3.

5.4.3 Textual Entailment Recognition Module

Recognizing Textual Entailment (RTE) is the process of determining whether a text fragment (Hypothesis H) can be inferred from another fragment (Text T) [166]. The RTE module receives the claim and the set of possible evidential sentences from the previous step. Let there be n possible sources of evidence for verifying a claim. For the i^{th} possible evidence, let s_i denote the probability of it entailing the claim, let r_i denote the probability of it contradicting the claim, and let u_i be the probability of it being uninformative. The RTE module calculates each of these probabilities.

The SNLI corpus [167] is used for training the RTE model. This corpus is composed of sentence pairs $\langle T, H \rangle$, where T corresponds to the literal description of an image and H is a manually created sentence. If H can be inferred from T, the "Entailment" label is assigned to the pair. If H contradicts the information in T, the pair is labelled as "Contradiction". Otherwise, the label "Neutral" is assigned.

We chose to employ the state-of-the-art RTE model proposed by [168] which is a re-implementation of the widely used decomposable attention model developed by [169]. The model achieves an accuracy of 86.4% on the SNLI test set. We selected it because at the time of development of this work, it was one of the best performing systems on the task with publicly available code.

Split	Entail.	Contradiction	Neutral
Training	122,892	48,825	147,588
Dev	4,685	4,921	8,184
Test	4,694	4,930	8,432

Table 5.5: FEVER SNLI-style Dataset split sizes for Entailment, Contradiction and Neutral classes

Model	Macro	Entail.	Contra.	Neutral
Vanilla	0.45	0.54	0.44	0.37
Fine-tuned	0.70	0.70	0.64	0.77

Table 5.6: Macro and class-specific F1 scores achieved on the FEVER SNLI-style test set

Additionally, the usage of preprocessing parsing tools is not required and the model is faster to train when compared to the other approaches we tried.

Although the model achieved good scores on the SNLI dataset, we noticed that it does not generalize well when employed to predict the relationships between the candidate claim-evidence pairs present in the FEVER data. In order to improve the generalization capabilities of the RTE model, we decided to fine-tune it using a newly synthesized FEVER SNLI-style dataset [170]. This was accomplished in two steps: the RTE model was initially trained using the SNLI dataset and then re-trained using the FEVER SNLI-style dataset.

The FEVER SNLI-style dataset was created using the information present in the FEVER dataset while retaining the format of the SNLI dataset. Let us consider each learning instance in the FEVER dataset of the form $\langle c, l, E \rangle$, where c is the claim, $l \in$ {SUPPORTS, REFUTES, NOT ENOUGH INFO} is the label and E is the set of evidences. While constructing the FEVER SNLI-style dataset, we only considered the learning instances labeled as "SUPPORTS" or "REFUTES" because these were the instances that provided us with evidences. Given such an instance, we proceeded as follows: for each evidence $e \in E$, we created an SNLI-style example $\langle c, e \rangle$ labeled as "Entailment" if $l =$ "SUPPORTS" or "Contradiction" if $l =$ "REFUTES". If e contained more than one sentence, we made a simplifying assumption and only considered the first sentence of e. For each "Entailment" or "Contradiction" which was added to this dataset, a "Neutral" learning instance of the form $\langle c, n \rangle$ was also created. n is a randomly selected sentence present the same document from which e was retrieved. We also ensured that n was not included in any of the other evidences in E. Following this procedure, we obtain examples that are similar (retrieved from the same document) but should be labeled differently. Thus, we obtained a dataset with the characteristics depicted in Table 5.5. To correct the unbalanced nature of the dataset, we performed random undersampling [171]. The fine-tuning had a huge positive impact on the generalization capabilities of the model as shown in Table 5.6. Using the fine-tuned model, the aforementioned set of probabilities are finally computed.

5.4.4 Final Classification

Twelve features were derived using the probabilities computed by the RTE module. We define the following variables for notational convenience:

$$cs_i = \begin{cases} 1 & \text{if } s_i \geq r_i \text{ and } s_i \geq u_i \\ 0 & \text{otherwise} \end{cases}$$

$$cr_i = \begin{cases} 1 & \text{if } r_i \geq s_i \text{ and } r_i \geq u_i \\ 0 & \text{otherwise} \end{cases}$$

$$cu_i = \begin{cases} 1 & \text{if } u_i \geq s_i \text{ and } u_i \geq r_i \\ 0 & \text{otherwise} \end{cases}$$

The twelve features which were computed are:

$$f_1 = \sum_{i=1}^{n} cs_i \qquad f_8 = max(r_i)\ \forall i$$

$$f_2 = \sum_{i=1}^{n} cr_i \qquad f_9 = max(u_i)\ \forall i$$

$$f_3 = \sum_{i=1}^{n} cu_i \qquad f_{10} = \begin{cases} \frac{f_4}{f_1} \text{ if } f_1 \neq 0 \\ 0 \text{ otherwise} \end{cases}$$

$$f_4 = \sum_{i=1}^{n} (s_i \times cs_i) \qquad f_{11} = \begin{cases} \frac{f_5}{f_2} \text{ if } f_2 \neq 0 \\ 0 \text{ otherwise} \end{cases}$$

$$f_5 = \sum_{i=1}^{n} (r_i \times cr_i) \qquad f_{12} = \begin{cases} \frac{f_6}{f_3} \text{ if } f_3 \neq 0 \\ 0 \text{ otherwise} \end{cases}$$

$$f_6 = \sum_{i=1}^{n} (u_i \times cu_i)$$

$$f_7 = max(s_i)\ \forall i$$

Each of the possible evidential sentences supports a certain label more than the other labels (this can be determined by looking at the computed probabilities). The variables cs_i, cr_i and cu_i are used to capture this fact. The most obvious way to label a claim would be to assign the label with the highest support to the claim. Hence, we chose to use the features f_1, f_2 and f_3 which represent the number of possible evidential sentences which support each label. The amount of support lent to a certain label by supporting sentences could also be useful in performing the labelling. This motivated us to use the features f_4, f_5 and f_6 which quantify the amount of support for each label. If a certain sentence can strongly support a label, it might be prudent to assign that label to the claim. Hence, we use the features f_7, f_8 and f_9 which capture how strongly a single sentence can support the claim. Finally, we used the features f_{10}, f_{11} and f_{12} because the average strength of the support lent by supporting sentences to a given label could also help the classifier.

These features were used by a Random Forest classifier [162] to determine the label to be assigned to the claim. The classifier was composed of 50 decision trees and the maximum depth of each tree was limited to 3. Information gain was used to measure the quality of a split. 3000

claims labelled as "SUPPORTS", 3000 claims labelled as "REFUTES" and 4000 claims labelled as "NOT ENOUGH INFO" were randomly sampled from the training set. Relevant sentences were then retrieved as detailed in Section 5.4.2 and supplied to the RTE module (Section 5.4.3). The probabilities calculated by this module were used to generate the aforementioned features. The classifier was then trained using these features and the actual labels of the claims.

We used the trained classifier to label the claims in the test set. If the "SUPPORTS" label was assigned to the claim, the five documents with the highest $(s_i \times cs_i)$ products were returned as evidences. However, if $cs_i = 0 \, \forall i$, then the label was changed to "NOT ENOUGH INFO" and a null set was returned as evidence. A similar process was employed when the "REFUTES" label was assigned to a claim. If the "NOT ENOUGH INFO" label was assigned, a null set was returned as evidence.

5.4.5 Experimental Setup

Our system was evaluated using a blind test set which contained 19,998 claims. Table 5.7 compares the performance of our system with that of the baseline system. It also lists the best performance for each metric. The evidence precision of our system was 0.5191 and its evidence recall was 0.3636. All of these results were obtained upon submitting our predictions to an online evaluator. DeFactoNLP had the 5^{th} best evidence F1 score, the 11^{th} best label accuracy and the 12^{th} best FEVER score out of the 24 participating systems.

Metric	*DeFactoNLP*	*Baseline*	*Best*
Label Accuracy	0.5136	0.4884	0.6821
Evidence F1	0.4277	0.1826	0.6485
FEVER Score	0.3833	0.2745	0.6421

Table 5.7: System Performance

The results show that the evidence F1 score of our system is much better than that of the baseline system. However, the label accuracy of our system is only marginally better than that of the baseline, suggesting that our final classifier is not very reliable. The low label accuracy may have negatively affected the other scores. Our system's low evidence recall can be attributed to the primitive methods employed to retrieve the candidate documents and sentences. Additionally, the RTE module can only detect entailment between two pairs of sentences. Hence, claims which require more than one sentence to verify them cannot be easily labelled by our system. This is another reason behind our low evidence recall, FEVER score and label accuracy. We aim to study more sophisticated ways to combine the information obtained from the RTE module in the near future.

To better assess the performance of the system, we performed a manual analysis of the predictions made by the system. We observed that for some simple claims (*ex.*"Tilda Swinton is a vegan") which were labeled as "NOT ENOUGH INFO" in the gold-standard, the sentence retrieval module found many sentences related to the NEs in the claim but none of them had any useful information regarding the claim object (*ex.*"vegan"). In some of these cases, the RTE module

would label certain sentences as either supporting or refuting the claim, even if they were not relevant to the claim. In the future, we aim to address this shortcoming by exploring triple extraction-based methods to weed out certain sentences [31].

We also noticed that the usage of coreference in the Wikipedia articles was responsible for the system missing some evidences as the RTE module could not accurately assess the sentences which used coreference. Employing a coreference resolution system at the article level is a promising direction to address this problem.

The incorporation of named entity disambiguation into the sentence and document retrieval modules could also boost performance. This is because we noticed that in some cases, the system used information from unrelated Wikipedia pages whose names were similar to those of the NEs mentioned in a claim to incorrectly label it (*ex.* a claim was related to the movie "Soul Food" but some of the retrieved evidences were from the Wikipedia page related to the soundtrack "Soul Food").

5.4.6 Discussion

After analyzing our results, we have identified many ways of improving the system in the future. For instance, triple extraction-based methods can be used to improve the sentence retrieval component as well as to improve the identification of evidential sentences. We also wish to explore more sophisticated methods to combine the information obtained from the RTE module. Overall, the main challenge remains at correctly extracting the evidence to a given claim.

5.5 Boosting the Evidence Extraction

In this section we further explore methods to improve the fact-extraction module, i.e., the evidence selection. Figure 5.6 exemplifies this in a nutshell, delineating these three components in a computer science perspective: 1) *document retrieval*, 2) *evidence selection* and 3) *claim classification*.

5.5.1 Features

We generate a set of eleven features that incorporate the morphological, syntactic and semantic information of the claim and evidence pair. The features are generated by extracting claim specific information from the evidence. We utilize subject, predicate and object (spo) triples (pre-extracted from the claim) in our datasets. A claim like "That '70s show is a sitcom" can be broken down into a spo triple as shown below.

Given a *spo* triple for each claim, a simple method to find similarity between *claim* and *evidence* is to find whether subject, predicate, and object (or their synonyms) appear in the sentence. We generate eleven such features. The first eight features utilize the triples to extract morphological information from evidence sentence. We later added semantic similarity, cosine similarity

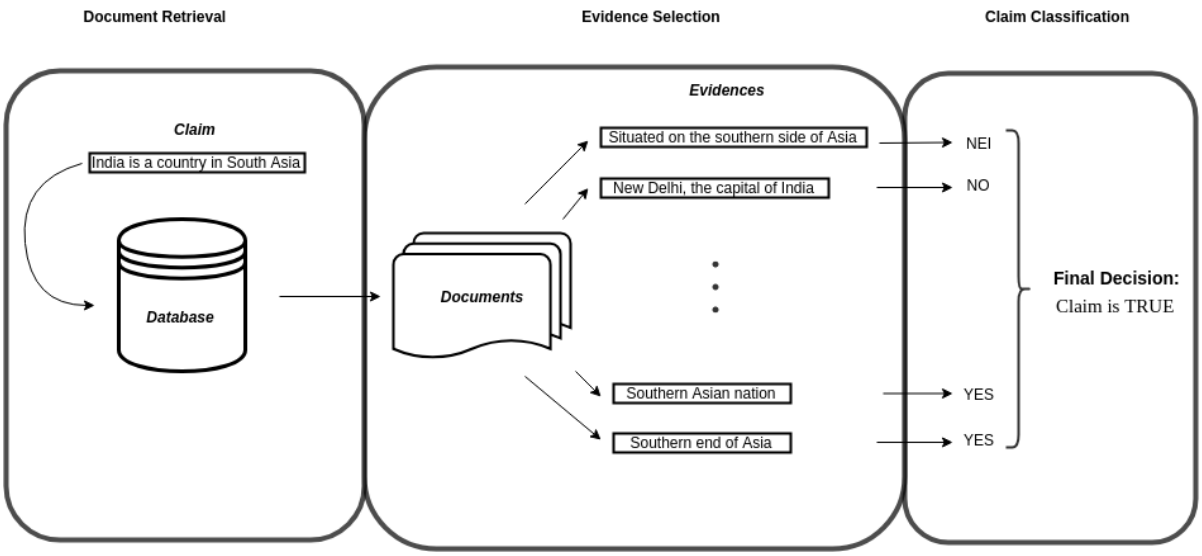


Figure 5.6: The figure shows three-component fact-checking pipeline: 1: document retrieval, 2: evidence selection and 3: claim classification. A database is queried using information extracted from the claim and top n documents are retrieved. In the second step, evidences (sentences or set of sentences similar to the claim) are retrieved from these documents. Lastly, the selected evidences are used to make a collective decision on the veracity of the claim.

and Jaccard similarity between the claim and evidence sentence as three features. These features represent the similarity measure between the *claim* and the most similar statement in the evidence. For Cosine Similarity, Jaccard Similarity and Semantic Similarity evidence S is divided into sentences $S_1, S_2, S_3, ..., S_n$ and maximum similarity scores with the claim are selected as the features.

Cosine distance: The claims and sentences are transformed into vectors as a function of token counts. Let C and S_i be the count vector for the claim and a statement in evidence respectively. The cosine distance is calculated as:

$$CosineDistance = max\left(\frac{C.S_i}{|C||S_i|}\right) \quad \forall\, S_i \in S$$

Jaccard Distance: If C and S_i are the sets of tokens of claim and a sentence in the evidence respectively, Jaccard distance is given by:

$$JaccardDistance = max\left(\frac{C \cap S_i}{C \cup S_i}\right) \quad \forall\, S_i \in S$$

Pre-trained word embeddings were used to compute the similarity based metrics using spaCy[11] python library. Table 5.8 gives details of all the features with an example illustrating all the feature values extracted from evidence using the claim.

[11] github.com/explosion/spaCy

Num	Feature	Definition	Feature Value
1	*is sub*	Checks if the document contains subject	1
2	*is obj*	Checks if the document contains object	1
3	*is pred*	Checks if the document contains predicate	1
4	*dist sub obj* Text follows	Distance between subject and object	48
5	*pred between*	Does predicate occur between subject and object	1
6	*sub relax*	Checks whether subject is present in partial form	1
7	*obj relax*	Checks whether object is present in partial form	1
8	*pred relax*	Checks whether predicate is present in partial form	1
9	*Jaccard distance*	Maximum Jaccard coefficient	0.08
10	*Cosine similarity*	Maximum cosine similarity	0.43
11	*Semantic similarity*	Similarity score of most semantically similar sentence	0.69

Table 5.8: The table gives an example of a claim-evidence pair and corresponding feature values.

The extracted features are trained on three (weak) baseline classifier models: Support Vector Machine (SVM), Random Forest (RF) and Multi-Layer Perceptron (MLP), further detailed in Section 5.5.3.

5.5.2 SimpleLSTM

`SimpleLSTM` is a Long-Short-Term-Memory (LSTM) based model that extracts semantic information from claims and evidences and then combines these representations. The combined layer is then fed to a fully-connected neural network, where the final decision making (classification) is done. We use stacked-LSTM layers for both, claim and evidence.

The last output at the end of LSTM stack gives the semantic encoding vector for both, the claim (*claimvec*) and evidence (*sentvec*). These two vectors are merged and fed to a fully connected layer. Since we use pre-trained word2vec embeddings, our first approach was to compute a cosine distance between the *sentvec* and *claimvec* as a merging criterion. however, in practice any binary function `MERGE(setvec,claimvec)` can be used to produce a merged representation. We experimented with the following merging criteria:

1. Cosine distance: Computes the cosine distance between *sentvec* and *claimvec* vectors.

$$CosineDistance = \frac{C.D}{|C||D|}$$

2. Concatenation: Concatenation of the *sentvec* and *claimvec* vectors.

$$Concatenation = [CD]$$

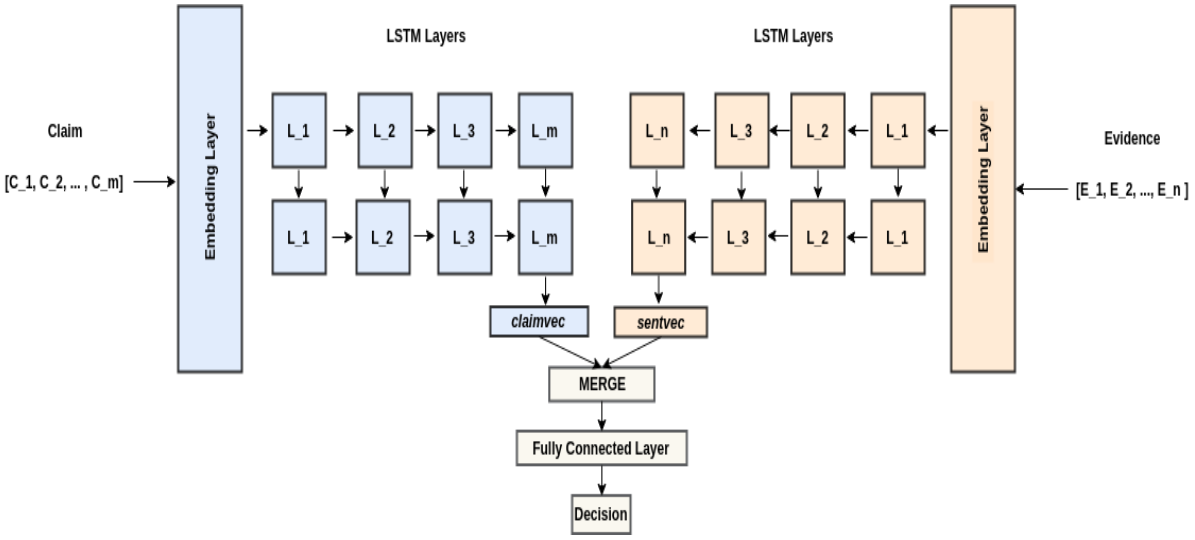


Figure 5.7: `SimpleLSTM` model. The diagram gives a schematic representation of the `SimpleLSTM` model. The input are claim and evidence. Both, the evidence and the claim are fed to an embedding layer (common for both) that outputs embedding representation for each word. These embeddings are then passed through LSTM layers (Our implementation uses 2 stacked-LSTMs). The final output of LSTM, sentvec and claimvec, are merged and fed to the fully connected layer.

3. Multiplication: Elementwise multiplication of the *sentvec* and *claimvec* vectors.

$$Multiplication = [C * D]$$

We found that out of cosine distance, element-wise multiplication, and plain concatenation, element-wise multiplication works the best, so we decided to go with it. In light of the simplistic design of our model, we call it `SimpleLSTM`.

The inputs to the model are a claim, evidence and a target label. The claim is represented as $[C_1, C_2, \cdots, C_m]$ where m ranges from 10-20 words, and the evidence is represented as $[E_1, E_2, \cdots, E_n]$ where n ranges from 100-200 words. The claim and evidence vectors are first passed through a pre-trained embedding layer to get a corresponding $d \times m$ and $d \times n$ matrices for claim and evidence respectively, where d is the size of each word embedding. It should be noted that both the evidence and claim share the embedding layer which facilitates the merging of *claimvec* and *sentvec* vectors. The embedding matrices are then fed to two parallel stacked-LSTM layers. The last LSTM outputs for the both the LSTM modules give feature representations for claim and evidence. We call them *claimvec* and *sentvec*. The feature vectors are passed through a merge function that then feeds the combined vector to the fully connected layer. Figure 5.7 gives a schematic representation of `SimpleLSTM`.

A common approach for training RNN models on textual data is to let the model train its own word embeddings. This is done by adding an embedding layer at the start of the deep-learning pipeline. Many studies have showed improved performance by using pre-trained word embeddings [172–175]. Our experiments found that using pre-trained word embeddings from models like Word2Vec [176], Glove [177], FastText [178] helps reducing the over-fitting and gives

better overall performance. We use pre-trained word2vec embeddings to train our models.

5.5.3 Experimental Setup

For the experiments, we used the most relevant fact-checking dataset public available: FEVER [152]. In this work, we mainly focused on simple claims, i.e., claims which do not exceed one sentence. Therefore, we extracted from FEVER only claims which are represented by a *subject-predicate-object* triple. We refer to this extracted subset as FEVER-Simple. Given the FEVER-Simple dataset, we further divide the problem into three: FEVER-Support, FEVER-Reject, and 3-Class. The first two datasets transform the examples into a binary problem (respectively), i.e., sentences either *support/reject* (Positive) or are *not related* (Negative) to the claim. The third dataset (3-Class) is a multi-label dataset, containing support sentences (Positive), counter-argumentative sentences (Negative) and finally sentences not related to the claim (NEI). Table 5.9 gives details about the number of instances for each dataset. For training the models, we divide the datasets into the train (80%), validation (10%) and test (10%) split. The next sub-section gives an overview of the data processing steps and heuristics we used for obtaining the datasets.

Dataset	Label	Count
FEVER-Support	Support	2761
	NEI	2761
FEVER-Reject	Reject	2955
	NEI	2955
3-Class	Support	2761
	Reject	2847
	NEI	2804

Table 5.9: FEVER-Simple Datasets

As introduced in above, we defined three weak and two strong baselines, as follows: *XGBoost* model is trained on FeverSimple datasets the best hyperparameter settings for each model is selected using a grid search with cross-validation on the training set. A max_depth of 8 and 1000 estimators provide the best performance overall on the three FeverSimple datasets. *TE* represents the textual entailment model with decomposable attention [179] by AllenNLP [180].

Weak Baselines

1. Random Forest Classifier: As decision trees often suffer from the problems of over-fitting, we decided to use the random forest classifier. It is an ensemble classifier in its own that trains on a set of decision trees and provide an inference by aggregating the results yielded on a randomly chosen subset of trained decision trees. The prediction accuracy mostly depends on the number of decision trees along with pruning strategy for each tree. We cross-validate the results on a random sample of data and tune the hyperparameters to obtain the best results.

2. Support Vector Machine: SVMs are known to perform well even with non linearly related data and hence are often used for classification tasks in NLP. We learned from our initial experiments that a linear SVM wouldn't yield any better results than a random forest classifier. Hence the choice for a kernel method is obvious. In order to find the most suitable kernel function among Linear, RBF, and polynomial kernel, we used the grid search cross-validation technique on validation data. We found that RBF kernel yields the highest accuracy among the three and hence we used this to further find the hyperparameters.

3. Multi-Layer Perceptron: MLP was also used to define the architecture of the FEVER Baseline in the last Shared Task [152]. We use a simple neural network with two hidden layers. Our experiments show that test accuracy gradually reduces with an increase in the number of layers. We found two hidden layers to be optimal. We further apply cross-validation technique to fine-tune our model.

Strong Baselines

4. Gradient Boosting Classifier (*XGBoost*)

Tosik et al. [53] propose a feature based model for *stance detection*. The model uses a gradient boost method for classification which combines a number of weak learners into a single learner on an iterative basis in the form of decision trees. The feature set contains twenty features based on various distance measures such as *cosine distance* and *hamming distance*, *relative entropy* between topic model probability distributions, *sentiment scores* of the claim and sentence (or evidence) and etc.. We use this model as one of the baselines for the claim classification task.

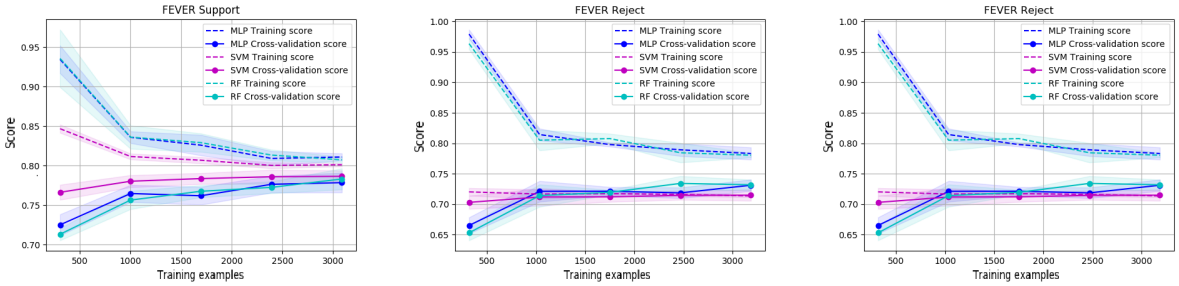
5. Textual Entailment

Textual Entailment (TE) is under the umbrella of Argumentation Mining in NLP and is a natural language processing task to find directional relation between texts [159]. Given a text fragment, the task is to determine if this text is a consequence of another text. The first text fragment is called a *hypothesis* and the second reference text *entailing text*, where the *entailing text* and *hypothesis* can be seen as the evidence and the claim, respectively. We use the TE model implementation by AllenNLP [180] as the second baseline for the claim classification task. The model is an implementation of the decomposable attention model given by Parikh et. la [179].

5.5.4 Results

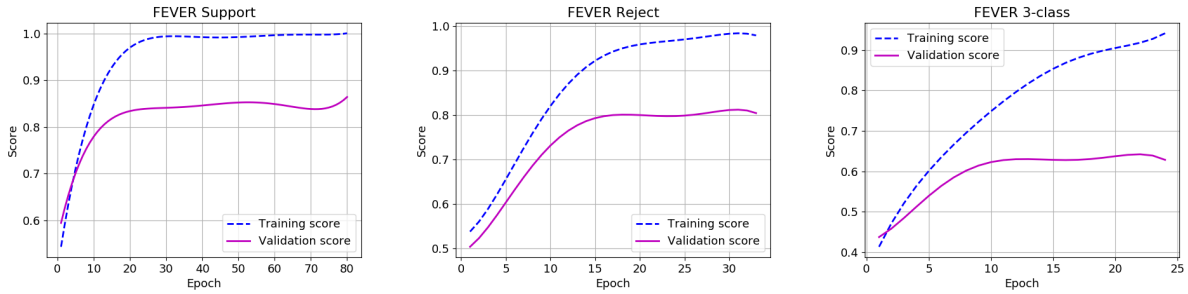
The `FeatureModel` experiments involve tuning the hyper-parameters for all the three classifiers: MLP, RF and SVM. We used cross-validation as sampling method with grid search for hyper-parameter optimization[12]. According to our experiments, Random Forest yield the best results with a maximum depth of 10 and *entropy* as splitting criterion. For SVM penalty of 100, *gamma* 0.001 and *rbf* kernel gave the best performance. The neural network has two hidden layers

[12] GridSearchCV from scikit-learn to obtain the best hyper-parameters



(a) FEVER-Support (b) FEVER-Reject (c) FEVER-3class

Figure 5.8: `FeatureModel` training graphs on different datasets



(a) FEVER-Support (b) FEVER-Reject (c) FEVER-3class

Figure 5.9: `SimpleLSTM` training graphs on different datasets

with 44 perceptrons in each. The model provides best results with *Adam* optimizer and *ReLU* activation function.

The tables 5.10,5.11,5.12 depict the accuracy, precision, recall and F1 scores for all the models. It can be observed that the our feature models outperform the gradient-boost (XGBoost) [53] and TE [180]. The MLP models gives better performance than RF and SVM on the FEVER-Simple datasets. The learning curves for three feature-based classifiers shown in figure 5.8. It shows a plot of training accuracy and cross-validation accuracy versus sample number.

We observe a high variance from the training graphs for in Figure 5.8 (a), (b) and (c) on FEVER dataset. We also observe a lower training accuracy implying a high bias situation. This indicates that a larger dataset and a higher number of features could yield better results. This is one of the factors that motivated us to come up with a deep learning model.

Classifier	Accuracy	Precision	Recall	f1 Score
XGBoost [53]	0.766	0.766	0.766	0.762
TE [180]	0.691	0.835	0.655	0.734
`FeatureModel` RF	0.79	0.76	0.83	0.79
`FeatureModel` SVM	0.79	0.71	0.85	0.77
`FeatureModel` MLP	0.79	0.76	0.81	0.78
`SimpleLSTM`	0.850	0.834	0.856	0.845

Table 5.10: Performance comparison of different models on FEVER support Dataset

Classifier	Accuracy	Precision	Recall	f1 Score
XGBoost [53]	0.74	0.738	0.736	0.73
TE [180]	0.548	0.759	0.533	0.626
`FeatureModel` RF	0.73	0.73	0.81	0.76
`FeatureModel` SVM	0.642	0.73	0.78	0.75
`FeatureModel` MLP	0.74	0.69	0.78	0.73
`SimpleLSTM`	0.816	0.836	0.811	0.824

Table 5.11: Performance comparison of different models on FEVER reject Dataset

Classifier	Accuracy	Precision	Recall	f1 Score
XGBoost [53]	0.535	0.54	0.534	0.539
TE [180]	0.418	0.372	0.622	0.465
`FeatureModel` RF	0.55	0.60	0.61	0.60
`FeatureModel` SVM	0.55	0.54	0.56	0.53
`FeatureModel` MLP	0.59	0.61	0.62	0.61
`SimpleLSTM`	0.635	0.643	0.620	0.642

Table 5.12: Performance comparison of different models on FEVER 3-class Dataset

For SimpleLSTM, we chose GoogleNews vectors*GoogleNews vectors*[13] [181] for pre-trained word embeddings. This is a word2vec [181] model has been trained on Google News dataset that has about 100 billion words. It contains word embeddings of dimension 300 for 3 million words and phrases. We fixed batch size of 64 and Adam Optimizer, with learning rate of 0.001, for all the datasets. The loss is *binary cross-entropy* loss for binary classification tasks, and *categorical cross-entropy* loss for the 3-class problem. The input size for LSTM stack is 300, and the output size is 150 for FEVER-Support and FEVER-Reject, and 100 for 3-Class.

Figure 5.9 gives the graphs showing training and validation accuracy versus epochs for each dataset. Tables 5.10,5.11,5.12 show that `SimpleLSTM` outperforms the baselines on all datasets.

[13] https://code.google.com/archive/p/word2vec/

5.6 Summary

In this section we presented our solution to tackle the fact-checking problem. More specifically, we discussed the process of automation of the task, as well as detailed the three contributions we achieved. First, we presented DeFacto, a multilingual and temporal approach for checking the validity of RDF triples using the Web as corpus (Section 5.3). In more detail, we explicated how multi-lingual natural-language patterns for formal relations can be used for fact validation. In addition, we presented an extension for detecting the temporal scope of RDF triples with the help of pattern and frequency analysis. We support the endeavour of creating better fact validation algorithms (and to that end also better relation extraction and named entity disambiguation systems) by providing the full-fledged benchmark FactBench. This benchmarks consists of one training and several test sets for fact validation as well as temporal scope detection. We showed that our approach achieves an F1 measure of 84.9% on the most realistic fact validation test set (FactBench mix) on DBpedia as well as Freebase data. The temporal extension shows a promising average F1 measure of 70.2% for time point and 65.8% for time period relations. However, the main limitation of this first release of the framework was the dependency on external libraries to verbalize predicates. Moreover, it was firstly designed to perform fact-validation on RDF-like claims.

Then, as a second contribution we extended the framework to verify also complex claims, which are expressed in natural language (Section 5.4). DeFactoNLP, an extension of DeFacto, was designed and tested in the FEVER 2018 Shared Task. When supplied a claim, it makes use of NER and TFIDF vector comparison to retrieve candidate Wikipedia sentences which might help in the verification process. An RTE module and a Random Forest classifier are then used to determine the veracity of the claim based on the information present in these sentences. The proposed system achieved a 0.4277 evidence F1-score, a 0.5136 label accuracy and a 0.3833 FEVER score. During our experiments, we noticed that the main challenge remains at correctly extracting evidence to support or refute a given claim.

Therefore, as a third contribution we investigated the impact of neural architectures in the evidence extraction module (Section 5.5). We give an overview of the problem and its automation in the context of natural language processing. In order to solve this task, we propose two new models: `SimpleLSTM` and `FeatureModel`, comparing the results with two strong baselines. Our experiments show that `SimpleLSTM` outperforms all the baselines. Compared to the best baseline (XGBoost [53]), it outperforms it by 11%, 10.2% and 18.7% on the FEVER-Support, FEVER-Reject and 3-Class tasks respectively. The advantage of such architecture, besides the increased performance, is the ability to easily deal with an unlimited number of predicates and languages, requiring only to update the pre-trained embeddings. Unlike, feature-based architectures (`FeatureModel`) that tend to not scale in more complex scenarios.

CHAPTER 6

The Quest for Reproducibility in the Context of Machine Learning Experiments

The aim of this chapter is to shed light on reproducible challenges in scientific experiments encountered throughout the development of this thesis. The content of this chapter is based on the publications [182–185]

RQ4: Are existing reproducible research methods sufficient to enable reproducibility?

Machine learning (ML) experiments - as most of scientific experiments - are complex studies involving many steps and iterations which require expert knowledge.

A relatively recent *key term* to face this lack of metadata is *Reproducible Research*, which aims to make analytic data and code freely available so that others will be able to reproduce findings, i.e., an environment where "provenance metadata" is accessible and a "high interoperability" level is achievable, so anyone is able to reproduce scientific achievements. Therefore, *Reproducibility* is one of the main principles of the scientific methods. According to the *IOM Report* [102] the following rules should be applied:

1. data/metadata publicly available;
2. the computer code and all the computational procedures should be available
3. ideally the computer code will encompass all of the steps of computational analysis

Ensuring that outcomes of scientific experiment are properly comparable, understandable, interpretable, reusable and reproducible is a challenge [186]. Although most of the machine learning libraries already provide some sort of machine readable meta-data, there is still no consensus on data formats and interchange rules. Yet another challenge lies in the data manipulation, i.e., the ability to manipulate and extract information out of a broad range of distinct experiments which are designed to solve a specific problem. Therefore, the problem in this case is the lack of common format, rules and standards to rapidly interchange data

among different systems and frameworks as well as manipulating metadata. In order to do so, scientist and developers are obliged to implement different wrappers, which is time-consuming and implies in financial costs.

Just to shed light on the challenge, our named entity recognition architecture (Section 3) needed more than 5.900 different experiments, producing an outcome of more than 3 million of data features. The manual management of this amount of metadata is infeasible without proper tools and schemata.

In order study the impact of different algorithm hyper-parameters and framework configurations over the same population, a set of controlled experiments are executed. Each producing a set of outcomes, which are further compared, usually under a certain metric. Finding the answer in a proper environment which enables reproducibility can take as much as a query requires to process, which usually is a matter of milliseconds (Section 6.2.2).

The proposed work introduced in this section aims to minimize the existing gap in this field by defining standards and providing a methodology to automatically represent machine learning experiments. Therefore, our motivation is to provide a decoupled and lightweight language-independent format for achieving the higher level of interoperability as well as supporting provenance.

6.1 Reproducing Machine Learning Experiments: An Open Problem

So far, we have seen a variety of publications involving Machine Learning (ML) topics, many of them contributing to the state of the art in their respective fields. However, in the last years we experienced a *knowledge gap* in the standardization of experiment results for scientific publications. In particular, experimental results are often not delivered in a common machine-readable way, causing the information extraction and processing to be tricky and burdensome. Moreover, recurring issues regarding the experiment could benefit from the existence of a public vocabulary. Reviewers of ML publications often need to investigate basic information on experiments conducted, e.g., which implementation of an algorithm was used, its configuration or choices for related hyper-parameters. Several questions may arise during the reviews, such as *"Which kernel method did the authors use?", "What is the regularization constant value?", "How many folds were used for the cross-validation section?", "Did they normalize the data?", "Which data distribution was used?", "Have any hypothesis test been applied?"*. This interpretation and look-up process is time-consuming and introduces misinterpretations which affect both comparability and reproducibility of scientific contributions.

6.1.1 Enabling Reproducibility: Challenges

Over the past years, different approaches have tried to address the reproducibility problem. Wings [104], OpenTox [105] and MyExperiment [187] are examples of *e-Science* tools to bridge this gap. Nevertheless, each of them dealing with a specific context (e.g., The toxicology domain [105]), failing to address a generic scientific scenario, i.e., existing approaches were

tailored as workflow systems for specific domains which do not offer a generic environment for the management of ML experiments. Thus, despite their noted achievements, we still do not have a consensus for a public format to achieve the interoperability for machine-learning experiments over any system implementation in a lightweight and simple format.

To bridge this gap, in recent years ontologies - as formal machine readable knowledge representations - have been introduced. An ontology formally defines essential *concepts*, their *properties*, and relevant *axioms* pertinent to a particular area of interest [188].

In the following we list the most notable contributions to propose generic ML schemata: *OntoDM* (an Ontology of Data Mining) provides generic representations of principle entities in the area of data mining [189]. *DMOP* (Data Mining OPtimization ontology) has been developed to support meta-mining, i.e. meta-learning from complete ML processes [190]. *Exposé* has been designed to describe and reason about ML experiments [110]. It underpins *OpenML* [106], a collaborative meta-learning platform for machine learning [106]. Table 6.1 overviews current workflow systems whereas Table 6.2 depicts existing ontologies.

Platform	**Description**
MyExperiment [187]	It is a collaborative environment where scientists can publish their workflows and experiment plans
Wings [104]	A Semantic Approach to creating very large scientific workflows
OpenTOX [105]	An interoperable predictive toxicology framework
Open ML [106]	A frictionless, collaborative environment for exploring machine learning

Table 6.1: The state-of-the-art platforms for e-science workflows

Figure 6.1 shows the overview of MEX classes and its relations. The diagram represents the classical iterations for an execution of a machine learning problem (*Classification, Regression or Clustering*). The white rounded rectangles representing a complete path for a *Classification* problem as well as its input (*Model*, *Corpus*, *Phase* and *Algorithm*) and outputs variables (*Example Performance* and *Overall Performance*).

Platform	**Description**
DMOP [108]	It supports informed decision-making at various choice points of the data mining process
OntoDM-KDD [109]	Ontology for representing the knowledge discovery process
Exposé [110]	An ontology for data mining experiments used in conjunction with *experiment databases* (ExpDBs [111])

Table 6.2: The related (heavy-weight) ontologies for data mining and their respective conceptualizations

While workflow systems were tailored to attack specific domains, proposed ontologies become too complex to represent ML experiments in a generic format. Despite their noted achievements, we still did not have a consensus for a public format to achieve the interoperability for machine-learning experiments over any system implementation in a *lightweight* and simple format.

6.2 Proposed Solutions

In the following we introduce the proposed solutions to tackle different problems.

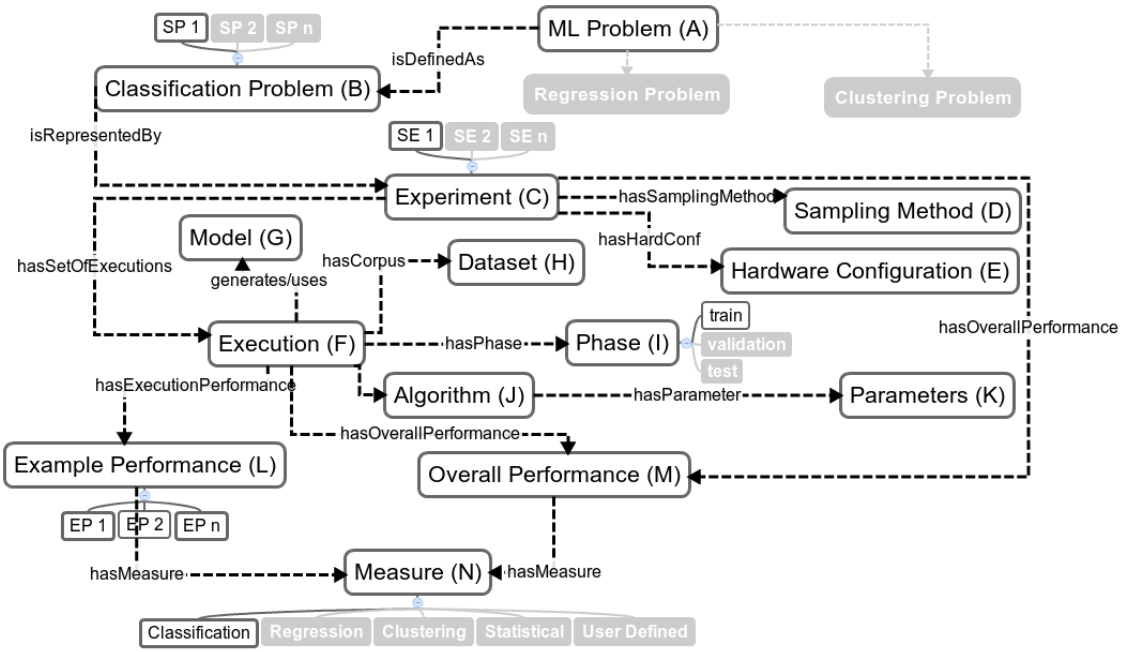


Figure 6.1: The MEX Vocabulary at a glance.

1. First, we design the **MEX Vocabulary**, a lightweight schema to represent ML experiments [185, 191];

2. Then we introduce **LOG4MEX**, a library to export ML experiment configurations based on the proposed vocabulary in a transparent manner [184, 192];

3. Thereafter we propose a set of interfaces to automatically generate machine learning metadata through **MEX-Interfaces** [193].

4. In order to improve scalability to different programming languages and tools, we propose **WEB4MEX**, a REST Interface which connects to LOG4MEX in the backlog and generates the metadata [194].

5. To allow metadata discovery and management, we further designed a metadata repository dubbed **WASOTA**. WASOTA supports RDF files based on the MEX Vocabulary to be stored and queried [195];

6. Last, but not least, we achieved the highest interoperability level in an international community effort to create an upper-level ontology to connect all pieces of metadata within the scope of the **MLS Project** [196]

6.2.1 The MEX Vocabulary

The MEX vocabulary has been designed to tackle the problem of sharing provenance information particularly on the basic machine learning iterations in a lightweight format. We extended the

W3C PROV Ontology (PROV-O) since it provides an excellent model for representing, capturing and sharing provenance information on the Web. The *PROV-O* provides three main classes, *Entity*, *Agent* and *Activity*, as well as other classes and properties enriching the provenance representation. Also, is endorsed by W3C [197]. The MEX vocabulary is composed as three sub-vocabularies:

1. **MEX-Core**: formalizes the key entities for representing the basic steps on machine learning executions, as well as the provenance information for linking between the published paper and the produced meta-data.

2. **MEX-Algorithm**: representing the context of machine learning algorithms and their associated characteristics, such as learning methods, learning problem and class of the algorithm.

3. **MEX-Performance**: provides the basic entities for representing the experimental results of executions of machine learning algorithms.

The first release of MEX vocabulary (Figure 6.1) aims to provide an embracing formalization to define the basics of a generic machine learning configuration (*"the algorithm and its parameters, the input features of given dataset, the sampling method and the hardware environment"*) as well as the representation of experiment outcomes (*"measures"*). We argue the standardization of a vocabulary is not sufficient to enable reproducibility. The development of proper tools are required in order to ensure that the process is transparent to data analysts and scientists. In the next sections we progress to discuss related frameworks to manage ML metadata through the vocabulary.

6.2.2 LOG4MEX

LOG4MEX [192] is a library based on MEX vocabulary [198] which aims at reducing ML gaps by exporting ML outputs directly from source code independently of which ML library is used. The conceptual ML entities are mapped to its structure, making the metadata generation process easier to the end-user (once the process occurs in a transparent manner). The library complies with the software engineering best practices, thus producing an enriched meta-data file to share configurations of ML executions. LOG4MEX stands as a flexible and lightweight library to represent executions of algorithms and the related variables. Hence, LOG4MEX covers an important existing gap in standardization of ML approaches. Diverse areas which implement the flow $input(parameters) \Rightarrow algorithm(mode\text{-}\ ls) \Rightarrow outputs(measures)$ can benefit from the proposed library, such as experiments in *natural language processing* or *stock market predictions*. A description of the architecture components depicted by the figure 6.2 is available in table 6.3.

The library enables the generation of the experimental metadata without extra coding effort. Figure 6.3 shows an example of the plotted metadata. The most relevant impact is the possibility to efficiently perform queries over thousand of experiments in a fraction of second, boosting the ability to explore and manage the data better. As a downside, this solution implies in extra code to perform logging. In the next, we compare existing methods and develop a solution which generates the metadata automatically through the definition of system interfaces [193].

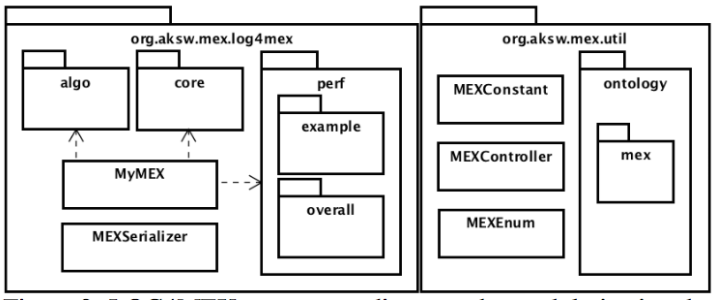


Figure 6.2: LOG4MEX component diagram: the modularization designed to keep the abstract concepts of machine learning. Furthermore, the package `ontology` has been designed to allow further Data Mining/Machine Learning schemata integrations.

Package (org.aksw.mex.log4mex*)	**Description**
*.algo	Mappings to mex-algo vocabulary
*.core	Mappings to mex-core vocabulary
*.perf	Mappings to mex-perf vocabulary
*.perf.example	Classes to represent performance of executions at *example* level [mexcore:SingleExecution]
*.perf.overall	Classes to represent performance of executions at *subset* level [mexcore:OverallExecution]
*.util	Static variables to map the vocabulary and control variables
*.util.ontology	Representation of diverse useful existing ontologies
*.util.ontology	Basic MEX classes types

Table 6.3: LOG4MEX Architecture Components: MEX implementation.

6.2.3 Metadata Generation Frameworks

Despite recent efforts to achieve a high level of interoperability of Machine Learning (ML) experiments, positively collaborating with the Reproducible Research context, we still run into problems created due to the lack of automatic methods to generate the metadata properly. This scenario leads to an extra coding-effort to achieve both the desired interoperability and a better provenance level as well as a more automatized environment for obtaining the generated results. Hence, when using ML libraries or platform, it is a common task to re-design specific data models (schemata) and develop wrappers to manage the produced outputs. In this section, we discuss this gap focusing on the solution for the question: "What is the cleanest and lowest-impact solution, i.e., the minimal effort to achieve both higher interoperability and provenance metadata levels?". We introduce a novel and low-impact methodology specifically designed for code built in that context, combining Semantic Web concepts and reflection in order to minimize the gap for exporting ML metadata in a structured manner, allowing embedded code annotations that are, in run-time, converted in one of the state-of-the-art ML schemas for the Semantic Web: MEX Vocabulary.

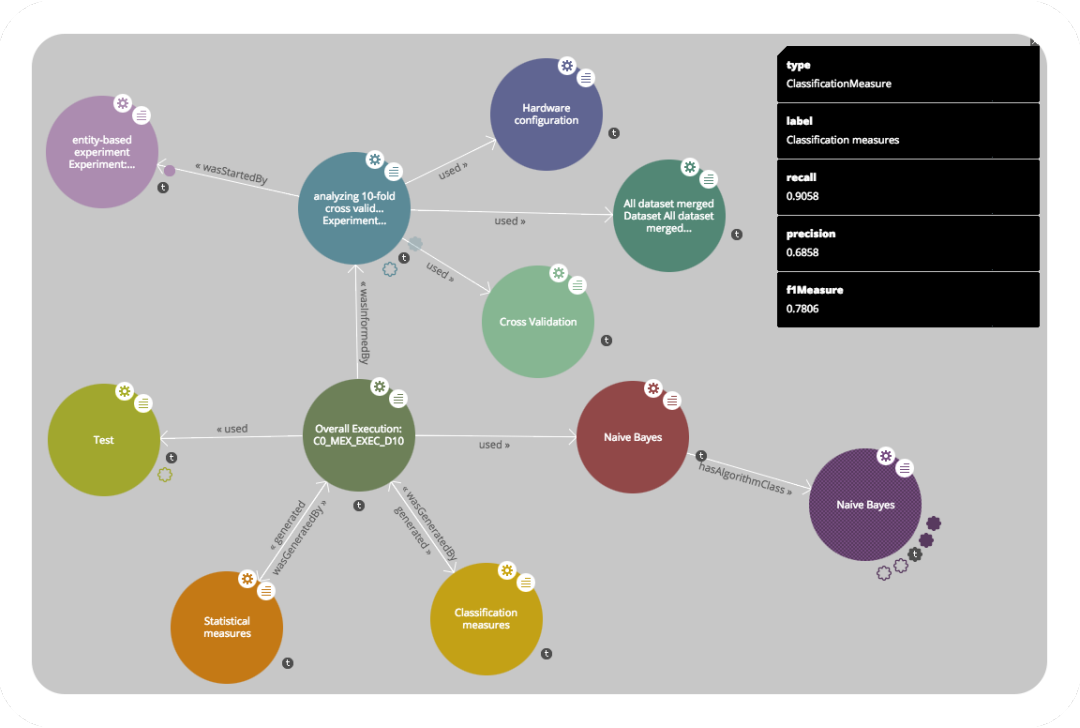


Figure 6.3: ML Metadata Exported through LOG4MEX

SWFS, MLF or MLL: A trade-off problem

Machine Learning has become an important tool for data scientists in research and business contexts. Plenty of workbenches/environments (*MLF*), libraries (*MLL*)[1] and workflow systems (*SWFS*) have emerged to serve as platform for creating ML models and executing experiments. Each of these provide a different level of implementation.

SWFS provides a good level of provenance metadata, data management, control of execution and allows the interchange of experiment configurations between researchers that use the same tool. However, they lead to a high level of dependency and the configurations are not portable among other *SWFS* implementations. Moreover, they imply a high level of algorithm's implementation dependency only once available algorithms can be used. Primarily, they are not commonly designed for specifically dealing with ML problems, but have either general scientific workflow proposed [199] or too specific scientific workflows [105] as focus of their implementation. Therefore, *SWFS* have the drawback which stands in the obligation of developing the solution specifically following its rules and natural limitations.

Another alternative, *MLF* are specifically designed to deal with ML problems and commonly provide a broad range of ML algorithm implementations. Some of them allow experiment configurations [200] and do not require refined programming skills with its user interface. As drawbacks, we can mention the lack of provenance and interoperability among implementations.

[1] from this point on we are going to refer to *MLL* as the situation where a developer works with an API by importing it into an IDE, instead of just referring to the library itself.

Also, this kind of platform does not allow programming flexibility to the user, which is a reason why *APIs* (*MLL*) are often released in order to be loaded into IDEs for developing specific and flexible applications. Table 6.4 lists the characteristics and the main differences among each platform. Also, Figure 6.4 depicts examples for the three different platforms discussed.

Platform	Advantages	Drawbacks
SWFS	High Provenance Interoperability Workflow Management	No (High) Interoperability updates are dependent of tool
MLF	Front-end No updates delay (Low) Workflow Management	No (High) Interoperability No much code flexibility
MLL	High code-flexibility	Low Provenance Low Interoperability

Table 6.4: Comparison of Machine Learning Platforms: Drawbacks and Advantages

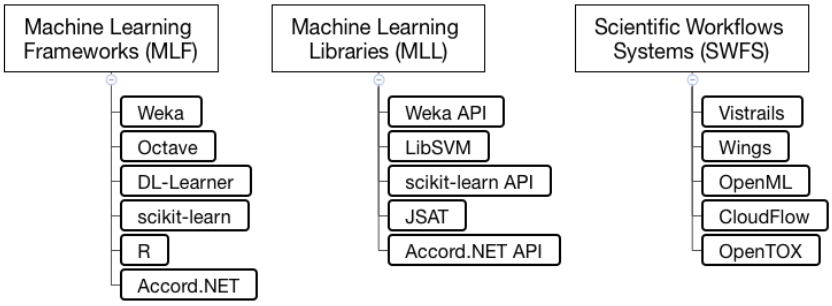


Figure 6.4: Examples of common machine learning platforms: *frameworks* that often implement a *front-end* interface (*MLF*), libraries to be imported into IDEs (*MLL*) and *workflow systems* which commonly have ML components as features (*SWFS*).

MLL: The Current Gap and Recurrent Solutions

As introduced, the major problem in the *MLL* context refers to the lack of *interoperability* and *provenance* metadata. Disregarding the possible lack of schema problem, the *MLL* context also does not provide *data management* features, i.e., without a proper *management system* becomes tricky to get and analyze different dimensions of the generated data.

As a result, the lack of automatized and straightforward solutions for *data management* requires to develop *wrappers* and implement *connectors* for any *Database Management Systems (DBMS)*, for instance. This extra step brings the focus out of the main problem being investigated, i.e., an extra code-effort is required to set up the desired environment. On the other hand,

avoiding this stage means to deal with pure *text files* or *stdout* outputs, which are not the best machine-readable solution and require a high level of effort to process and extract data, in addition to the discussed lack of *provenance* and *interoperability* (Figure 6.5). In other words, both situations are not welcome in terms of the implementation effort.

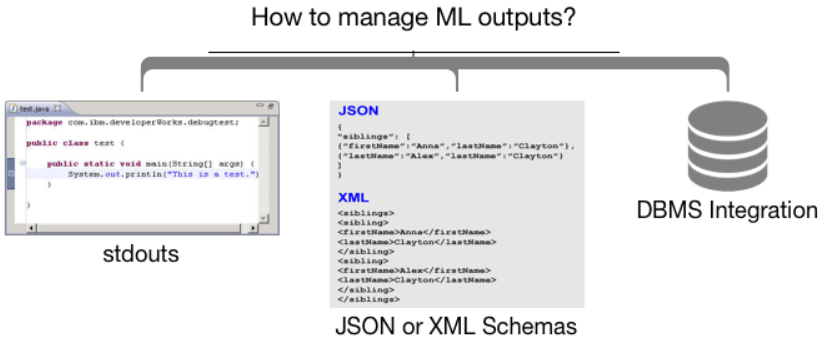


Figure 6.5: Managing output of machine learning executions in *MLL*: pure text (*stdouts*), self-schema definitions (e.g.:JSON or XML) or data base integrations (DBMS)

The *provenance* normally limits itself to an excerpt of text written in natural language linked to the produced data. *Interoperability* issues are commonly treated with *self-schema* definitions, which are then shared among developers, e.g., by designing a particular simple structure using an existing standard (e.g.: JSON[2] or XML[3]) or just logging using an API (e.g.: LOG4J[4]). However, this scenario has 1) the inconvenience to present a poor level of metadata 2) an inability to represent the data semantically, abstracting specific implementation issues (e.g.: "logit function" and "logistic regression", which points out to the same concept) 3) the extra code-effort needed. Here, the SW comes into play, offering a much more sophisticated approach to achieve a higher level of provenance, but still allowing to achieve a decent level of interoperability. Endorsed by W3C, RDF "*has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed*"[5]. In this paper we have developed a new methodology combining SW tools, *annotations* and *reflection* in order to reduce the effort to generate good and inter-operable metadata as well as to provide query features. Table 6.5 summarizes the different strategies discussed to bridge the gap.

MEX-Interfaces

One step forward to simplify the annotation process is proposed in this chapter. The major contribution is to allow metadata generation regardless of the *IDE*, machine-learning *library* and context of the experiment. We argue developers dealing with machine learning problems can directly benefit of the interfaces, automatizing the process of generating metadata of machine learning experiments. Furthermore, the proposed *interfaces* provide guidance on the

[2] http://www.json.org
[3] http://www.w3schools.com/XML/xml_whatis.asp
[4] http://logging.apache.org/log4j/2.x/
[5] http://www.w3.org/RDF/

Method	Advantages	Drawbacks
stdout	No Extra Coding Effort Required	Lack of Provenance
		Lack of Interoperability
		Lack of Data Query Feature
DBMS	Data Query Feature	Extra Coding Effort (Integration)
		Lack of Provenance
		Lack of Interoperability
Self-schema Definition	Straightforward Solution	Extra Coding Effort
		Extra Analysis Effort (modeling)
		Lack of Provenance
		Lack of Interoperability
Annotations + SW	Provenance	Extra Processing Time
	Interoperability	Security Issues
	Data Query Feature	
	Automatic Metadata Generation	

Table 6.5: Comparison of strategies for representing machine learning metadata in *MLL* contexts

standardization of the generated metadata, once they are based on a state of the art vocabulary for ML [6]. Figure 6.6 depicts the general process of generating the metadata. In this example, two annotated Java classes following the MEX annotation's descriptions are passed by parameter to the *MetaGeneration* class. The entire process occurs in a transparent manner and no further step is required. By doing so, developers reach a clean solution to narrow down the issues discussed before (Section 6.2.3)

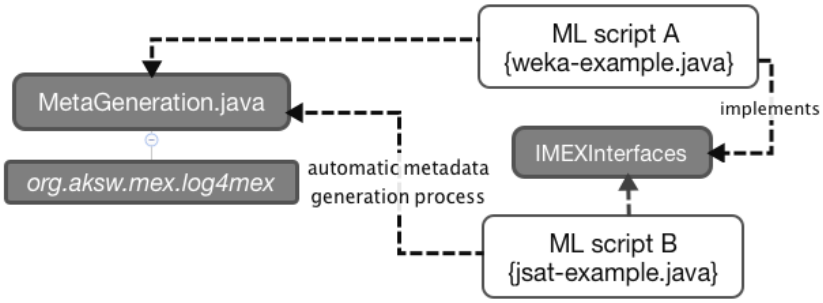


Figure 6.6: MEX Interfaces at a glance: a low impact solution for generating machine learning metadata from annotated classes

The produced metadata is based on MEX [198], a vocabulary specifically designed to deal with inputs and outputs of machine learning executions and relies on three main layers: *mexcore*[7]

[6] https://github.com/ML-Schema/core/wiki/Vocabulary
[7] http://mex.aksw.org/mex-core

for execution's controlling, *mexalgo*[8] for ML algorithms representations and *mexperf*[9] for performance indicators. It is a lightweight format built upon W3C PROV-O[10] - categorized as a vocabulary - which abstracts the core machine learning concepts regarding the execution of an algorithm. Further schemata - more focused on data mining flows - including OntoDM [109], Exposé [110] and DMOP [201] are classified as Ontologies. The Predictive Model Markup Language (PMML) [202] is a XML based schema and was conceived to represent (predictive and descriptive) data models as well as pre and post-processing. In this scenario, MEX stands as a flexible and lightweight solution for representing the basic triple - *inputs*, *run* and *outputs* - for any machine learning algorithm. Figure 6.7 depicts current technologies and schemas for representing machine learning metadata.

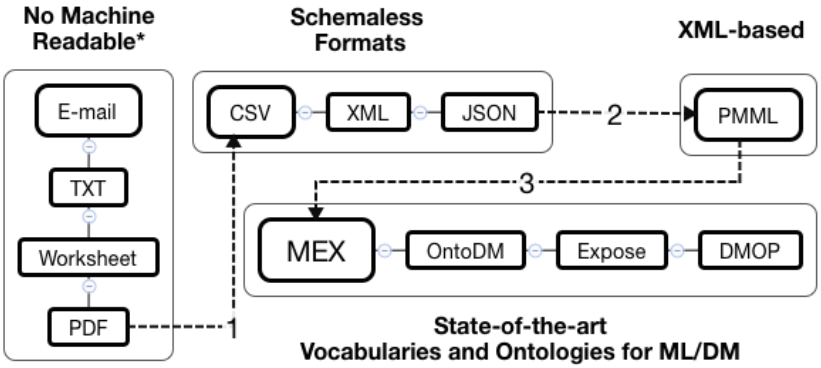


Figure 6.7: Open-source formats for representing ML metadata: from straightforward representations (1) formats until more refined schema representations (2)(3). Note: (*) Although it can be - technically - considered machine-readable, we assume that the effort to make it happen does not pays off.

Drawbacks and Limitations

Despite a more clean and less coupled solution (once a vocabulary provides a context-less list of common terms), the proposed methodology faces some limitations, as follows:

Reflection and Annotations: programming-language must allow reflection and annotations. As a use case, we have implemented *Java* examples, although other programming languages could be used (as long as it implements reflection). In case reflection is not allowed, LOG4MEX can be used for logging [184].

Performance Overhead and Security Restrictions: the use of *Reflection* directly impacts on the execution-time, decreasing the overall performance as well as expose the code impacting in security restrictions[11]. An impact analysis of performance is planned, although we argue that the most costly steps in ML scripts are *I/O* operations and mathematical calculations and not object creations.

[8] http://mex.aksw.org/mex-algo
[9] http://mex.aksw.org/mex-perf
[10] http://www.w3.org/TR/prov-o/
[11] https://docs.oracle.com/javase/tutorial/reflect/

Methodology Coverage: The MEX Vocabulary covers just pure machine learning metadata (an algorithm, its inputs and outputs for given execution). Pre-processing steps or data mining tasks are not covered due to the complexity of the task.

Local Variables: *reflection* in *Java* does not allow to capture local variables, i.e., variables that are not explicitly declared as *class variables* cannot be obtained via *annotations* and *reflection*.

Advantages

The biggest benefit of the proposed methodology is to use a standard model which abstracts the particular concepts existing into each ML environment/implementation and to create an upper layer that is able to inter connect knowledge as easy as possible with the produced metadata. The following list details the key advantages:

In-line Annotations: a *Java* class can be simply annotated and the metadata will be generated in *run-time*.

More Abstraction: by using a vocabulary, developers can benefit of the high level of abstraction provided. A *Support Vector Machines* algorithm for a classification problem can be represented with a single reference: `http://mex.aksw.org/ mex-algo#C-SVM`, there is no need to re-define a vocabulary.

Less Coding Effort and More Agreement Rate: there is no need to create and share the structure of the schema for representing the output data.

Better Interoperability and Provenance Levels: a common schema allows higher levels of data interchanging and *RDF* encourages better metadata descriptions.

Querying Capabilities: Once the vocabulary is RDF-based, developers can benefit from SPARQL queries[12].

Reproducible Research: the methodology collaborates with reproducible research rules, following best practices for data publishing and code management.

6.2.4 WEB4MEX

WEB4MEX [194] is a concise, but important contribution to boost interoperability. WEB4MEX is simply a HTTP service that is designed to work as an intermediate layer that receives and manages the client‘s calls with experiment‘s data, so it is also responsible to realize the interactions with the LOG4MEX API (that provides the MEX vocabulary). In order to accomplish that, at least theoretically, any language that is capable of HTTP communication and JSON data manipulation can use this service. This enables experiments in different languages to generate, by the end of the execution, data that is formatted by the MEX vocabulary.

[12] http://www.w3.org/TR/rdf-sparql-query/

6.2.5 WASOTA

Recently, a few web repositories have been released to share general experiment configurations and scientific workflows (e.g., RunMyCode, CodaLab, myExperiment and OpenML). None of the above projects, however, provide a straightforward way to gather information about the states of the art through an organized taxonomy of domains. Instead, they aim at being platforms for sharing complex meta-information about an experiment. Although some of them still allow users to get informed on how well different approaches perform on a given task, none comes as a semantic, light-weight aggregator of such performance values. Moreover, the domain scope of WASOTA [195] is wider than just Machine Learning or Natural Language Processing. For instance, OpenML algorithms are evaluated on datasets where features have already been engineered; whilst WASOTA considers an algorithm as a black box, which optionally contains the feature engineering process and can process raw data, such as text, images, or RDF graphs.

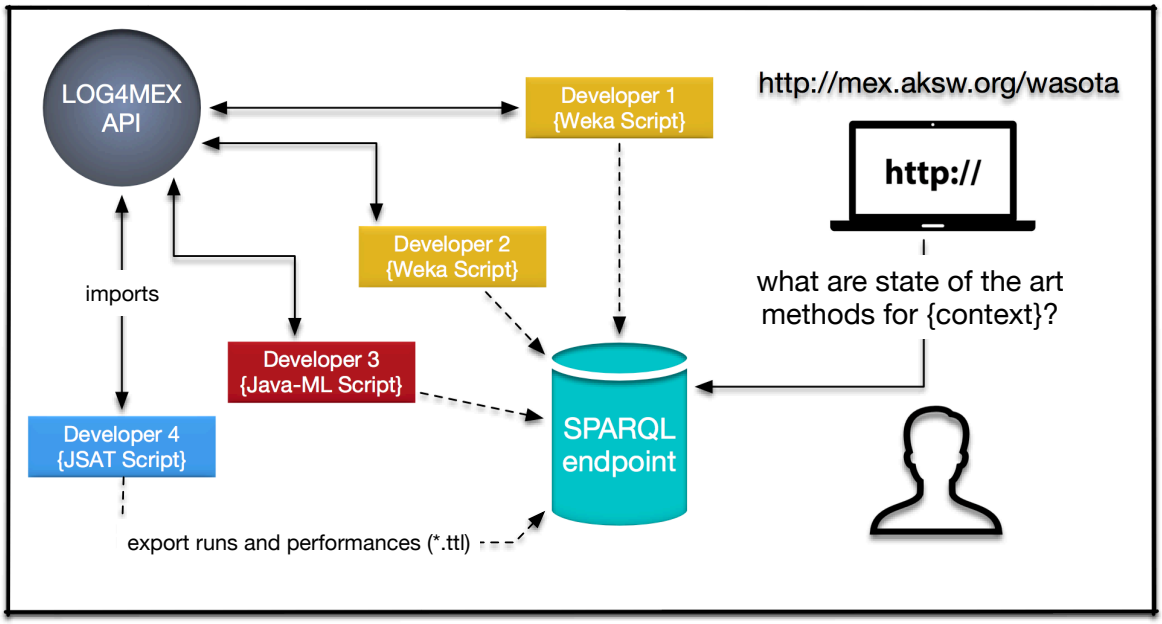


Figure 6.8: WASOTA: A blueprint of the WASOTA architecture. A simple solution to reduce the searching time for state of the art methods and a central repository of metadata for ranking

Figure 6.8 depicts the overall system's architecture where different researchers export metadata[13] from their experiments to WASOTA, regardless programming-language or framework. The system consolidates and groups the information automatically, providing a platform to readily present best existing methods (based on a performance measure) for a specific domain. Further indicators are also possible to be applied, such as "dataset", for instance. Due to the metadata be semantic enriched with linked data, more detailed information can also be discovered, e.g.: the hardware configuration of given environment, if it is provided.

[13] http://mex.aksw.org/

6.2.6 ML-Schema

The development of ML ontologies (Section 6.1.1) is a significant step towards ensuring unambiguous interpretability and reproducibility of ML experiments. However, none of the existing ontologies fully covers the area of machine learning and supports all the needs for the representation and encoding ML experiments. Instead of the development of a comprehensive general purpose ML ontology, here we propose a more practical and flexible approach that involves the development of ML-Schema – Machine Learning Schema (MLS) – for mapping of the existing ML ontologies and to support a variety of useful extensions. To achieve this ambitious goal, in September 2015 developers of several ML ontologies formed a W3C Community Group[14]. The development of MLS has been initiated as an attempt to prevent a proliferation of incompatible ML ontologies and to increase interoperability among existing ones. The *MLS Community Group* (MLS-CG) is an open-source community comprehending over 50 international researchers.

The main challenge in the development of MLS is to align existing ML ontologies and other relevant representations designed for a range of particular purposes following sometimes incompatible design principles, resulting in different not easily interoperable structures. Moreover, ML experiments are run on different ML platforms; each of those having specific conceptualization or schema for representing data and metadata.

To address the challenge, the members of the MLS-CG identified and aligned a set of principle ML entities – a core ML vocabulary. The core vocabulary of MLS deals with ML algorithms. The schema is focusing on the representation of the algorithms, the machine learning tasks they address, their implementations and executions, as well as inputs (e.g., data), outputs (e.g., models), and performances. The schema also defines a relationship between machine learning algorithms and their single executions (runs), experiments and studies encompassing them.

The terms in the core vocabulary were defined and manually mapped to the ML ontologies participating in this endeavor through several rounds of consultations. In 2016, the MLS-CG published an online proposal for MLS, and welcomed comments and suggestions from the research community and wider [196].

Given the importance of the work to the Semantic Web community, next, we dedicate a chapter to detail the work under the W3C Machine Learning Schema Group, presenting the results of three years of MLS-CG efforts in standardization of the encoding of ML experiments.

6.3 The ML-Schema

In this section, we introduce the MLS ontology w.r.t. its aims and design principles. We also describe the properties defined within the MLS ontology namespace.

[14] See www.w3.org/community/ml-schema/

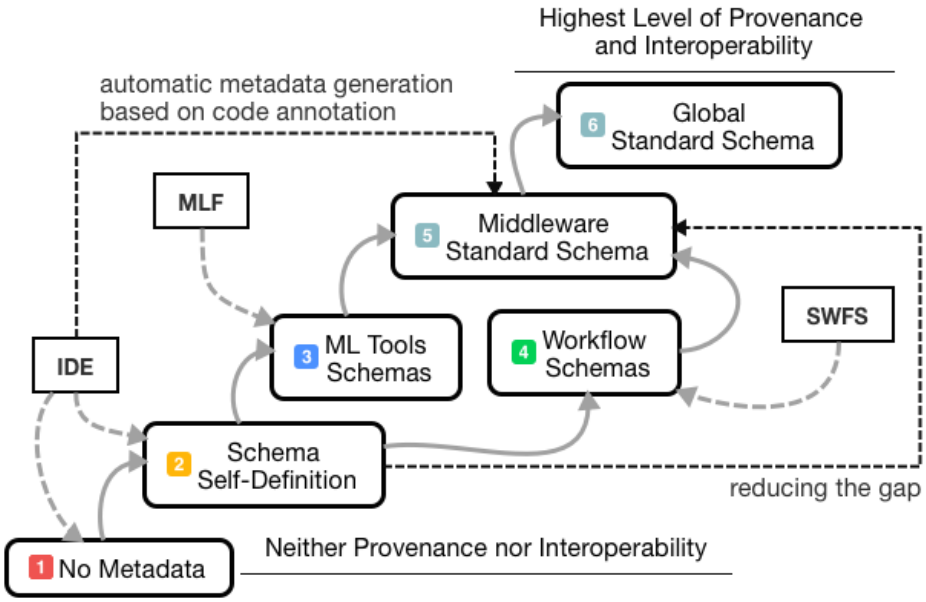


Figure 6.9: Vertical and Horizontal Interoperability across ML Environments.

6.3.1 The MLS ontology

The main aim of MLS (or the MLS ontology) is to provide a high-level standard to represent ML experiments in a concise, unambiguous and computationally processable manner. In particular, it aims to align existing ML ontologies and to support development of more specific ontologies for particular purposes and applications.

To serve its purposes, MLS ontology has to be compact but sufficiently comprehensive and easily extendable. To achieve such an aim, we chose to design MLS ontology as a light-weight ontology that can be used as a basis for ontology development projects, markup languages and data exchange standards. We then show how the MLS ontology is open for further extensions and mappings to other resources.

For example, MLS ontology can support vertical and horizontal interoperability across various ML environments. Different ML platforms have different underlying schemes for representing data and metadata (see Figure 6.9: items 3 and 4: vertical interoperability). This may lead to an extra coding-effort (see Figure 6.9: item 2) if to achieve both the desired interoperability and a better provenance level as well as a more automatized environment for obtaining the generated results. To reduce the gap, ML vocabularies and ontologies have been proposed (see Figure 6.9: item 5).

The gap can be further significantly reduced by achieving interoperability among state-of-the-art (SOTA) schemata of those resources (see Figure 6.9: item 5) i.e. achieving the horizontal interoperability (Figure 6.9: item 6). Therefore, different groups of researchers could exchange SOTA metadata files in a transparent manner, e.g.: from OntoDM and MEX (`MLS.Schemadata =MLS.convert('myfile.ttl',MLS.Ontology.OntoDM,MLS.Ontology.MEX)`).

6.3.2 MLS Ontology properties

In the following we list and briefly describe the properties modelled in MLS:

achieves: A relation between a run and a task, where the run achieves specifications formulated by the task.

definedOn: A relation between a task and either the data or an evaluation specification pertinent to this task.

defines: The inverse relation of ***definedOn***

executes: A relation between a run and an implemantation that is being executed during the run.

hasHyperParameters: A relation between an implementation of a machine learning algorithm and its hyperparameter..

hasInput: A relation between a run and data that is taken as input to the run.

hasOutput: A relation between a run and either a model or model evaluation that is produced on it's output.

hasPart: A relation which represents a part-whole relationship holding between an entity and its part.

hasQuality: A relation between entities and their various characteristics.

implements: A relation between an information entity and a specification that it conforms to.

realizes: A relation between a run and an algorithm, where the run realizes specifications formulated by the algorithm.

specifiedBy: A relation between an entity and the information content entity that specifies it.

6.3.3 Related ontologies

The following related ML ontologies are those that MLS is aligned to the moment. These alignments will be further described in the Section 6.4.

The OntoDM-core ontology

For the domain of data mining there are several developed ontologies, with the aim of providing formal descriptions of domain entities. One of the proposed ontologies is the OntoDM-core

ontology. In one of the preliminary versions of the ontology, the authors decided to align the proposed ontology with the Ontology of Biomedical Investigations (OBI) [203] and consequently with the Basic Formal Ontology (BFO) at the top level[15], in terms of top-level classes and the set of relations. That was beneficial for structuring the domain in a more elegant way and the basic differentiation of information entities, implementation entities and processual entities. In this context, the authors proposed a horizontal description structure that includes three layers: a specification layer, an implementation layer, and an application layer. The specification layer in general contains information entities. In the domain of data mining, example classes are data mining task and data mining algorithm. The implementation layer in general contains qualities and entities that are realized in a process, such as parameters and implementations of algorithms. The application layer contains processual classes, such as the execution of the data mining algorithm.

The Exposé ontology

The main goal of Exposé is to describe (and reason about) machine learning experiments. It is built on top of OntoDM, as well as top-level ontologies from bio-informatics. It is currently used in OpenML, as a way to structure data (e.g. database design) and share data (APIs). MLS will be used to export all OpenML data as linked open data (in RDF).

For the sake of simplicity and comprehension, we further refer to the Exposé ontology as the OpenML vocabulary, or simply OpenML.

The DMOP ontology

The DMOP ontology has been developed with a primary use case in meta-mining, that is meta-learning extended to an analysis of full DM processes. At the level of both single algorithms and more complex workflows, it follows a very similar modeling pattern as described in the MLS. To support meta-mining, DMOP contains a taxonomy of algorithms used in DM processes which are described in detail in terms of their underlying assumptions, cost functions, optimization strategies, generated models or pattern sets, and other properties. Such a "glass box" approach which makes explicit internal algorithm characteristics allows meta-learners using DMOP to generalize over algorithms and their properties, including those algorithms which were not used for training meta-learners.

The MEX vocabulary

MEX has been designed to reuse existing ontologies (i.e., PROV-O, Dublin-Core[16], and DOAP[17]) for representing basic machine learning information. The aim is not to describe a complete data-mining process, which can be modeled by more complex and semantically refined structures.

[15] http://basic-formal-ontology.org/
[16] http://dublincore.org
[17] http://usefulinc.com/doap/

Instead, MEX is designed to provide a simple and lightweight vocabulary for exchanging machine learning metadata in order to achieve a high level of interoperability as well as supporting data management for ML outcomes.

6.4 MLS core and alignments

MLS provides a model for expressing data mining and machine learning algorithms, datasets, and experiments. This section introduces the related ontologies, the core of the MLS model – namely the classes (types) that are used to represent the majority of the cases – and the mappings with the existing ML ontologies. This mapping highlights how MLS is compatible with prior ontologies and how resources currently described in other ontologies can be described uniformly using MLS, hence allowing us to link currently detached machine learning resources.

6.4.1 Task

In MLS, the Task class represents a formal description of a process that needs to be completed (e.g.based on inputs and outputs). A Task is any piece of work that needs to be addressed in the data mining process. Table 6.6 depicts a synthesis of the alignments detailed below.

OpenML

OpenML differentiates a TaskType (e.g. classification, regression, clustering,...) and Task instances. The TaskType defines which types of inputs are given (e.g. a dataset, train-test splits, optimization measures) and which outputs are expected (e.g. a model, predictions,...). On the other hand, a Task contains specific dataset, splits, etc. It can be seen as an individual of the class.

DMOP

In DMOP, a task is any piece of work that is undertaken or attempted. A DM-Task is any task that needs to be addressed in the data mining process. DMOP's DM-Task hierarchy models all the major task classes: CoreDM-Task, DataProcessingTask, HypothesisApplicationTask, HypothesisEvaluationTask, HypothesisProcessingTask, InductionTask, ModelingTask, DescriptiveModelingTask, PredictiveModelingTask, and PatternDiscoveryTask.

OntoDM

OntoDM defines a data mining task as an objective specification that specifies the objective that a data mining algorithm needs to achieve when executed on a dataset to produce as output a generalization. It is represented as a subclass of the IAO: objective specification class, where

objective specification is a directive information entity that describes and intended process endpoint. The data mining task is directly dependent of the datatypes of the data examples on which the task is defined, and is included directly in the task representations. This allows us to represent tasks defined on arbitrarily complex datatypes. The definition of data mining algorithm and generalizations is strongly dependent on the task definition.

OntoDM contains a taxonomy of data mining tasks. At the first level, we differentiate between four major task classes: predictive modelling task, pattern discovery task, clustering task, and probability distribution estimation task. Predictive modelling task is worked out in more detail. Since, a predictive modeling task is defined on a pair of datatypes (one describing the part of the data example on the descriptive side and the other describing the part of the data example on the target/output side), we differentiate between primitive output prediction tasks (that include among others the traditional ML tasks such as classification and regression) and structured output prediction tasks (that include among others tasks such as multi-label classification, multi-target prediction, hierarchical multi-label classification).

MEX

MEX has a higher level of abstraction, designed for representing ML executions and related metadata and not DM tasks. There are specific classes for representing specific ML standards. This information could be obtained from Learning Problem + Learning Method + Algorithm Class in a more concise level.

> Learning Problem: Association, Classification, Clustering, Metaheuristic, Regression, Summarization, ...
>
> Learning Method: Supervised Learning, Unsupervised Learning, Semi-supervised Learning, Reinforcement Learning, ...
>
> Algorithm Class: ANN, ILP, Bagging, Bayes Theory, Boosting, Clustering, Decision Trees, Genetic Algorithms, Logical Representations, Regression Functions, Rules, Support Vector Networks, ...

As an `:ExperimentConfiguration` may have many `:Executions` and an `:Experiment` may have many `:ExperimentConfigurations`, these can be aligned to a `mls:Task`.

6.4.2 Algorithm

In MLS, the Algorithm class represents an algorithm regardless of its software implementation. Table 6.7 summarizes the alignments described below.

Property	Value
Example Classes	Classification, Regression, Clustering, Feature Selection, Missing value imputation,...
Example Individuals	Classification on Dataset Iris
OpenML	TaskType
DMOP	DM-Task
OntoDM	"Data Mining Task"
MEX	The closest concept is `mexcore:ExperimentConfiguration`

Table 6.6: The syntheses of the MLS Task class and its relation with aligned ontologies.

OpenML

OpenML currently does not abstract over algorithms anymore, it simply has 'implementations'. The underlying reasoning is that algorithms can come in endless variations, including hybrids that combine multiple pre-existing algorithms. Classifying every implementation as a specific type of algorithm is therefore not trivial and hard to maintain. Instead, to organize implementations, OpenML has 'tags', so that anybody can tag algorithms with certain keywords, including the type of algorithm that is implemented. Hence, hybrid algorithm can have multiple tags.

DMOP

An DM-Algorithm is a well-defined sequence of steps that specifies how to solve a problem or perform a task. It typically accepts an input and produces an output. A DM-Algorithm is an algorithm that has been designed to perform any of the DM tasks, such as feature selection, missing value imputation, modeling, and induction. The higher-level classes of the DM-Algorithm hierarchy correspond to DM-Task types. Immediately below are broad algorithm families or what data miners more commonly call paradigms or approaches. The Algorithm hierarchy bottoms out in individual algorithms such as CART, Lasso or ReliefF. A particular case of a DM-Algorithm is a Modeling (or Learning) algorithm, which is a well-defined procedure that takes data as input and produces output in the form of models or patterns.

OntoDM

In OntoDM, authors differentiate between three aspects of algorithms: algorithm as a specification, algorithm as an implementation, and the process of executing an algorithm. Data mining algorithm (as a specification) is represented as a subclass of IAO: algorithm. In this sense, a data mining algorithm is defined as an algorithm that solves a data mining task and as a results outputs a generalization and is usually published/described in some document (journal/conference/workshop publication or a technical report).

In OntoDM, it is given a higher level taxonomy of algorithms. At the first level, it is differentiated

between single generalization algorithms (algorithms that produces a single generalization as a result) and ensemble algorithms (algorithms that produce an ensemble of generalizations as a result). At the second level, the taxonomy follows the taxonomy of tasks. This modular and generic approach allows easy extensions to characterize each algorithm class with its own distinctive set of characteristics that can be represented as qualities.

MEX

Sharing the problem stated by OpenML, MEX labels high levels of ML algorithms in Algorithm class instead of specific algorithm characterisations. As much as more precise information is needed, related classes could be instantiated, such as Learning Problem + Learning Method + Algorithm Class + Implementation.

Property	Value
Example Classes	Algorithm
Example Individuals	Linear Regression, Random Forest, AdaBoost...
OpenML	None
DMOP	DM-Algorithm
OntoDM	"Data Mining Algorithm"
MEX	mexalgo:Algorithm

Table 6.7: MLS Algorithm class alignments

6.4.3 Implementation

In MLS, the Implementation class represents an executable implementation of a machine learning algorithm, script, or workflow. It is versioned, and sometimes belongs to a library (e.g. WEKA). The alignments of this class against related ML ontologies are described following, and Table 6.8 summarizes them.

OpenML

OpenML does not distinguish between 'operators' and 'workflows', because the line is often very blurry. Many algorithms have complex internal workflows to preprocess the input data and make them more robust. Also, many environments (e.g. R, Matlab, etc.) do not have the concept of operator; they just have function calls, which are part of scripts. Hence, in OpenML, every implementation is called a Flow, which can be either atomic or composite.

OntoDM

In OntoDM, authors represent a data mining algorithm implementation as a subclass of OBI: plan is a concretization of a data mining algorithm. Data mining algorithms have as qualities parameters that are described by a parameter specification. A parameter is a quality of an algorithm implementation, and it refers the data provided as input to the algorithm implementation that influences the flow of the execution of algorithm realized by a data mining operator that has information about the specific parameter setting used in the execution process.

MEX

Implementation in MEX is meant to represent the *Software* Implementation and has no link to the algorithm itself. Examples are Weka, SPSS, Octave, DL-Learner.

Property	Value
Example Classes	LearnerImplementation,DataProcessingImplementation,EvaluationProcedureImplementation
Example Individuals	`SVMlib,weka.J48,rapidminer.RandomForest,weka.evaluation.CrossValidation,weka.attributeSelection.GainRatioAttributeEval`
OpenML	Flow / Implementation
DMOP	DM-Operator / DM-Workflow
OntoDM	"Data mining algorithm implementation"
MEX	`mexalgo:Implementation`

Table 6.8: A syntheses of the MLS Implementation class and its alignments.

6.4.4 HyperParameter

The MLS HyperParameter class represents a a prior parameter of an implementation, i.e., a parameter which is set before its execution (e.g. C, the complexity parameter, in `weka.SMO`). As shown in Table 6.9, this is directly aligned with OpenML's Parameter, MEX's AlgorithmParameter, and DMOP's OperatorParameter.

In OntoDM, however, a data mining algorithm execution is a subclass of SWO:information processing, which is an OBI:planned process. Planned processes realize a plan which is a concretization of a plan specification. A data mining algorithm execution realizes (executes) a data mining operator, has as input a dataset, has as output a generalization, has as agent a computer, and achieves as a planned objective a data mining task.

Data mining operator is a role of a data mining algorithm implementation that is realized (executed) by a data mining algorithm execution process. The data mining operator has

information about the specific parameter setting of the algorithm, in the context of the realization of the operator in the process of execution. The parameter setting is an information entity which is a quality specification of a parameter.

Property	Value
Example Classes	HyperParameter
Example Individuals	`weka.SMO_C,weka.J48_M,rapidminer.RandomForest_number_of_trees`
OpenML	Parameter
DMOP	OperatorParameter
OntoDM	Parameter
MEX	`mexalgo:AlgorithmParameter(mexalgo:HyperParameter` under proposal)

Table 6.9: MLS HyperParameter class and its alignments

6.4.5 Data

In MLS, the Data class represents a data item composed of data examples and it may be of a various level of granularity and complexity. With regard to granularity, it can be a whole dataset (for instance, one main table and possibly other tables), or only a single table, or only a feature (e.g., a column of a table), or only an instance (e.g., row of a table), or a single feature-value pair. With regards to complexity, data examples are characterized by their datatype, which may be arbitrarily complex (e.g., instead of a table it can be an arbitrary graph). OpenML describes data at this level of granularity, while the alignment is more complex in other. Table 6.10 summarizes the alignments.

DMOP

DM-Data: In SUMO, Data is defined as an item of factual information derived from measurement or research. In IAO, Data is an alternative term for 'data item': 'an information content entity that is intended to be a truthful statement about something (modulo, e.g., measurement precision or other systematic errors) and is constructed/acquired by a method which reliably tends to produce (approximately) truthful statements'. In the context of DMOP, DM-Data is the generic term that encloses different levels of granularity: data can be a whole dataset (one main table and possibly other tables), or only a table, or only a feature (column of a table), or only an instance (row of a table), or even a single feature-value pair.

OntoDM

OntoDM imports the IAO class dataset (defined as 'a data item that is an aggregate of other data items of the same type that have something in common') and extends it by further specifying that a DM dataset has part data examples. OntoDM-core also defines the class dataset specification to enable characterization of different dataset classes. It specifies the type of the dataset based on the type of data it contains. In OntoDM, we model the data characteristics with a data specification entity that describes the datatype of the underlying data examples. For this purpose, we import the mechanism for representing arbitrarily complex datatypes from the OntoDT ontology. Using data specifications and the taxonomy of datatypes from the OntoDT ontology, in OntoDM-core have a taxonomy of datasets.

MEX

In MEX, it is possible to represent even each instance (mexcore:Example) and each feature (mexcore:Feature) of the dataset.

Property	Value
Example Classes	Dataset, Train-test splits, Predictions
Example Individuals	Iris, FaceScrub, IMDB-WIKI
OpenML	Data
DMOP	DM-Data
OntoDM	Dataset specification, DM-dataset
MEX	`mexcore:Dataset` (as metadata)

Table 6.10: MLS Data class and its alignments.

6.4.6 Model

We define Model as a generalization of a set of training data able to predict values for unseen instances. It is an output from an execution of a data mining algorithm implementation. Models have a dual nature: they can be treated as data structures and as such represented, stored and manipulated; on the other hand, they act as functions and are executed, taking as input data examples and giving as output the result of applying the function to a data example. Models can also be divided into global or local ones. A global model has global coverage of a data set, i.e., it generalizes the whole data set. A local model, such as a pattern set, is a set of local hypotheses, i.e. each applies to a limited region of the data set.

Table ?? demonstrates the alignments of MLS Model class that are described in the following.

Property	Value
Example Classes	Decision tree, Rule set, Clusterings, Pattern set, Bayesian Network, Neural Net, Probability Distribution,...
Example Individuals	Decision tree built on Iris
OpenML	None
DMOP	DM-Hypothesis (with main subclasses: DM-Model, DM-PatternSet)
OntoDM	Generalization
MEX	None

Table 6.11: MLS Model class and its alignments

DMOP

By Hypothesis, DMOP actually meant roughly ML models. They introduced the concept of a 'hypothesis' to differentiate ML models from pattern sets. On the other hand, the DM-PatternSet represents a pattern set, as opposed to a model which by definition has global coverage, is a set of local hypotheses, i.e. each applies to a limited region of the sample space.

OntoDM

In OntoDM, authors take generalization to denote the outcome of a data mining task. They consider and model three different aspects of generalizations: the specification of a generalization, a generalization as a realizable entity, and the process of executing a generalization.

Generalizations have a dual nature. They can be treated as data structures and as such represented, stored and manipulated. On the other hand, they act as functions and are executed, taking as input data examples and giving as output the result of applying the function to a data example. In OntoDM, a generalization is defined as a sub-class of the BFO class realizable entity. It is an output from a data mining algorithm execution.

The dual nature of generalizations in OntoDM is represented with two classes that belong to two different description layers: generalization representation, which is a sub-class of information content entity and belongs to the specification layer, and generalization execution, which is a subclass of planned process and belongs to the application layer.

In addition, a MLS ModelCharicteristic is a Generalization quality, while a MLS ModelEvaluation is mapped to a Generalization evaluation.

6.4.7 Run

An MLS run is an execution of an implementation on a machine (computer). It is limited in time (has a start and end point), can be successful or failed. If successful, it often has a specific result, such as a model and evaluations of that model's performance. Although runs are called very differently in the different existing ontologies, the semantics are the same. Table 6.12 shows the alignments with them.

Property	Value
Example Classes	SimpleProcess, Execution
Example Individuals	Process running SVMlib on Iris on Machine m on timestamp t
OpenML	Run
DMOP	DM-Process (i.e., execution)
OntoDM	Data mining algorithm execution
MEX	`mexcore:Execution` (singly `mexcore:SingleExecution`, collectively `mexcore:OverallExecution`)

Table 6.12: MLS Run class and its alignments

6.4.8 EvaluationMeasure

An MLS evaluation measure unique defines how to evaluate the performance of a model after it has been trained in a specific run. As shown in table 6.13, this is directly aligned across the different existing ontologies. In DMOP, however, there exist subclasses, such as Computational-ComplexityMeasure, HypothesisEvaluationMeasure, and ModelComplexityMeasure.

6.4.9 Study

An MLS study is a collection of runs that belong together to perform some kind of analysis on its results. This analysis can be general or very specific (e.g. an hypothesis test). It can also be linked to files, data, that belong to it. Studies are often the most natural product of a scientific investigation, and can be directly linked to certain claims and other products, such as research papers. As shown in Table 6.14, existing ontologies call this either a study or an experiment, although the semantics are the same.

Property	Value
Example Classes	`ClassificationMeasure,RegressionMeasure,ClusteringMeasure,RuntimeMeasure...`
Example Individuals	`Predictive_accuracy,root_mean_squared_error,inter_cluster_variance,cputime_training_milliseconds`
OpenML	`EvaluationMeasure`
DMOP	Measure
OntoDM	None
MEX	`mexperf:PerformanceMeasure`

Table 6.13: MLS EvaluationMeasure class and its alignments

Property	Value
Example Classes	`BenchmarkStudy`
Example Individuals	Specific collections of runs
OpenML	Study
DMOP	DM-Experiment (i.e., something that resembles a bundle in PROV, e.g. `prov:Bundle`)
OntoDM	None
MEX	`mexcore:Experiment`

Table 6.14: MLS Study class and its alignments

ML-Schema	OntoDM-core	DMOP	OpenML/Exposé	MEX Vocabulary
Task	Data mining task	DM-Task	Task	`mexcore:ExperimentConfiguration`
Algorithm	Data mining algorithm	DM-Algorithm	Algorithm	`mexalgo:Algorithm`
Software	Data mining software	DM-Software	N/A	`mexalgo:Tool`
Implementation	Data mining algorithm implementation	DM-Operator Algorithm implementation	N/A	`mexalgo:Implementation`
HyperParameter	Parameter	Parameter	Parameter	`mexalgo:HyperParameter`
HyperParameterSetting	Parameter setting	OpParameterSetting	Parameter setting	N/A
Study	Investigation	N/A	N/A	`mexcore:Experiment`
Experiment	N/A	DM-Experiment	Experiment	N/A
Run	Data mining algorithm execution	DM-Operation	Algorithm execution	`mexcore:Execution`
Data	Data item	DM-Data	N/A	`mexcore:Example`
Dataset	DM dataset	DataSet	Dataset	`mexcore:Dataset`
Feature	N/A	Feature	N/A	`mexcore:Feature`
DataCharacteristic	Data specification	DataCharacteristic	Dataset specification	N/A
DatasetCharacteristic	Dataset specification	DataSetCharacteristic	Data quality	N/A
FeatureCharacteristic	Feature specification	FeatureCharacteristic	N/A	N/A
Model	Generalization	DM-Hypothesis (DM-Model / DM-PatternSet)	Model	`mexcore:Model`
ModelCharacteristic	Generalization quality	HypothesisCharacteristic	Model Structure, Parameter, ...	N/A
ModelEvaluation	Generalization evaluation	ModelPerformance	Evaluation	N/A
EvaluationMeasure	Evaluation datum	ModelEvaluationMeasure	Evaluation measure	`mexperf:PerformanceMeasure`
EvaluationProcedure	Evaluation algorithm	ModelEvaluationAlgorithm	Performance Estimation	N/A

Table 6.15: Final full comparison between the terms of ML-Schema and aligned vocabularies

6.5 Use cases

To elucidate the benefits of MLS, we present three use cases where MLS can be utilized to foster the reproducibility of experiments. In particular, we show how previous research can benefit from the existence of an upper ontology which interlinks several vocabularies used for the exchange of experiment data and metadata.

6.5.1 Open Provenance Model for Workflows and Research Objects

It is often crucial to know exactly which data was used to train a machine learning model, where this data came from, and how it was processed before modelling. MLS is compatible with the *Open Provenance Model for Workflows (OPMW)* [204] and *Research Objects* [205]. This allows machine learning experiments to be described in a uniform way that preserves the provenance of data and models.

The term *provenance*, in computer science and scientific research, means metadata about the origin, derivation or history of data or thing. For instance, in biology or chemistry, we track steps of experimental processes to enable their reproduction. In computer science, we track the creation, editing and publication of data, including their reuse in further processes. The PROV data model for provenance was created, founded on previous efforts such as Open Provenance Model (OPM) [206], and later became recommended by W3C [207]. The PROV Ontology (PROV-O), also recommended by W3C [208], expresses the PROV Data Model using the OWL language. PROV-O provides a set of classes, properties, and restrictions that can be used to represent and exchange provenance information generated in various systems. The Open Provenance Model for Workflows (OPMW) is an ontology for describing workflow traces and their templates which extends PROV-O and the ontology P-plan designed to represent plans that guided the execution of processes [204]. Figure 6.10 presents the mapping of the MLS directly to OPMW and indirectly to PROV-O and P-plan.

Belhajjame et al. [205] proposed a suite of ontologies for preserving workflow-centric *Research Objects*. The ontologies use and extend existing widely used ontologies, including PROV-O. Especially, the two ontologies from the suite, the Workflow Description Ontology (wfdesc), used to describe the workflow specifications, and the Workflow Provenance Ontology (wfprov), used to describe the provenance traces obtained by executing workflows, follow a very similar conceptualization of workflows to that of OPMW and map to MLS.

6.5.2 OpenML

The OpenML platform contains millions of machine learning experiments, which were run using thousands of machine learning workflows on thousands of datasets. However, in themselves, these experiments form another island of data disconnected to the rest of the world. To remedy this, we have used MLS to describe all of these experiments as linked open data, so that scientists can connect their machine learning experiments to other knowledge sources, or build novel knowledge bases for machine learning research.

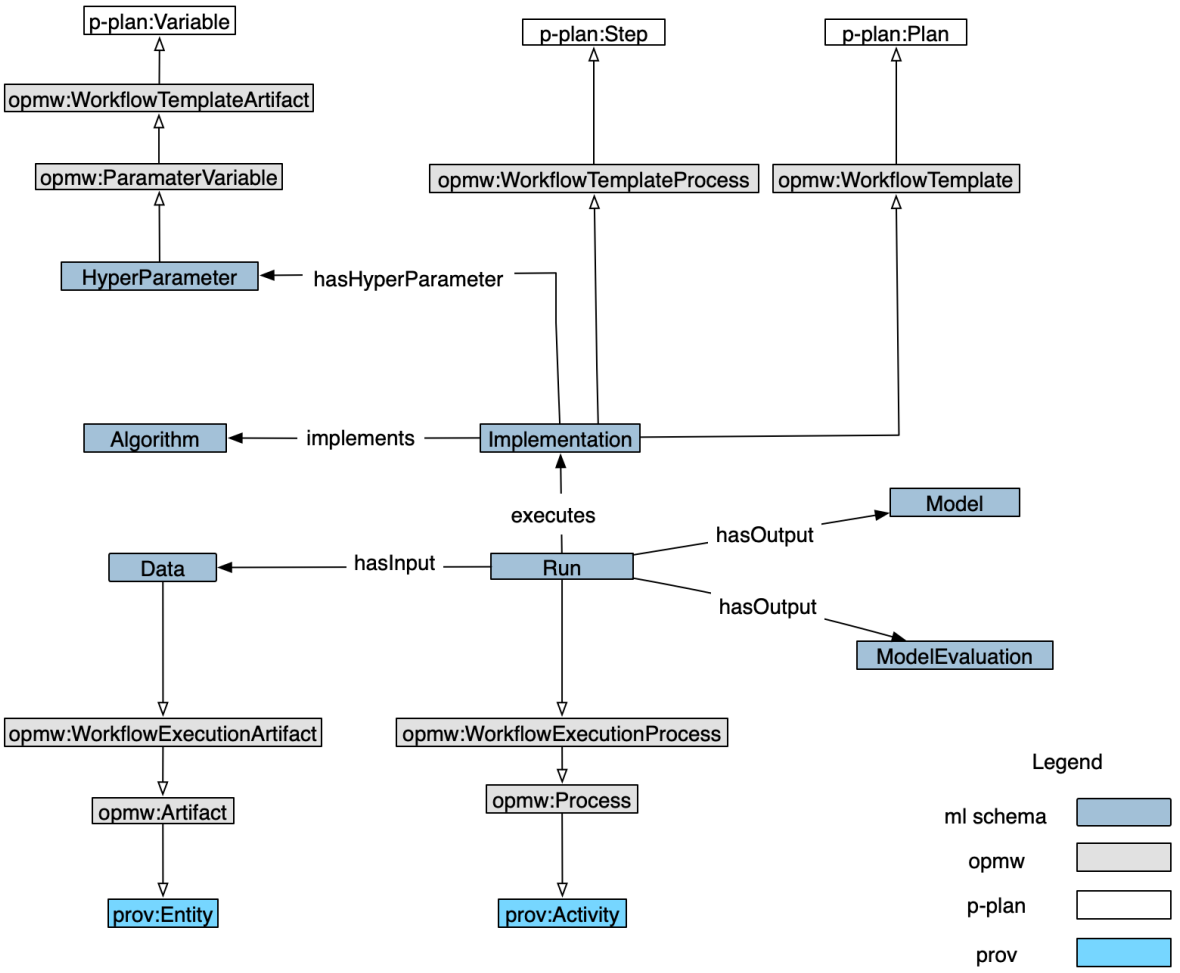


Figure 6.10: The mapping of MLS to OPMW, PROV-O and P-plan.

This is achieved through an export function that reads in OpenML's current JSON descriptions of datasets, tasks, workflows, and runs, and emits an RDF description using the MLS schema. This functionality is available as an open source Java library[18]. OpenML also supports this export functionality on the platform itself. In the web interface (`openml.org`) every dataset, task, workflow (flow), and run page has an RDF export button that returns the RDF description of that object, linked to other objects by their OpenML IDs. This functionality is also available via predictable URLs in the format `https://www.openml.org/{type}/{id}/rdf`, where *type* is either d (dataset), t (task), f (flow), or r (run), and id the OpenML ID of that object. Hence, the RDF description of dataset 2 can be obtained via `https://www.openml.org/d/2/rdf`.

As such, OpenML data becomes part of the semantic web, which allows scientists to link it to other data and reuse it in innovate new ways.

[18] The library is available on https://github.com/ML-Schema/openml-rdf

6.5.3 Deep Learning

This use case can also be described as a possible future work of MLS, where it is extended to support Deep Learning (DL) models.

By initiative of Microsoft and Facebook, a recently created community group called Open Neural Network Exchange (ONXX)[19] aims to allow users to share their Neural Network models and transfer them between frameworks. At the moment, it covers import/export to 3 different frameworks, while libraries for other 5 frameworks are under development or have partial support.

DL models have some requirements that MLS cannot describe at the moment – information such as number of layers and neurons, weights, and pre-trained models – as it only contains the HyperParameter class that is not able to store this additional information.

Unfortunately, the ONNX initiative does not provide an ontology; instead, their operators are described in the project GitHub documentation, while their terms are hardly defined in C code. On the other hand, the extension of the MLS ontology by adding new properties based on those terms would benefit not only the MLS, but all the aligned ontologies described in this work, that would instantly be able to use those properties to extend their models and support the description of DL models and experiments.

6.6 Summary

Scientific experiments are often neither replicable nor reproducible, which go against beliefs of "the scientific method" [209]. Among different reasons for that, the lack of standard ways to represent scientific experiments is one of the key challenges to instigate reproducibility research. In this chapter we presented ontologies and tools to enable reproducibility in the machine learning context. We demonstrated the expressiveness in the work of ML-Schema and how the MLS ontology was designed to be aligned with several ML ontologies, such as DMOP [190], OntoDM [189], MEX [182], and Exposé [110]. It was also possible to elucidate through use cases the capabilities of our work, such as the usage of MLS format for exporting ML experiments to RDF format in the OpenML [106] framework, its extension of that provides direct support to the OPMW and indirect to the PROV-O [208] ontology, as well as the possible extension to elucidate the description of DL experiments.

[19] https://onnx.ai/

CHAPTER 7

Conclusion and Future Directions

This thesis studies the research problem of automatizing the fact-checking validation task.

In particular, we tackle the problems of Entity Recognition on noisy data (Chapter 3), Web Credibility (Chapter 4), Automated Fact-Checking (Chapter 5), and Reproducible Research (Chapter 6). In the following sections, we summarize our contributions and discuss the main findings that corroborate our research questions.

7.1 Overall contributions and conclusions

To tackle the underlying challenges defined for this thesis we defined four research questions.

First, we tackled the problem of detection and classification of entities in a noisy environment, and answer the following research question.

> **RQ1**: Can images along with news improve the performance of the named entity recognition models on noisy text?

The Web often deals with more informal and colloquial languages, which lack implicit linguistic formalism. Thus, they have more unstructured properties, are shorter, they lack context and present more grammatical and spelling errors. With respect to that – not surprisingly – the performance of SOTA degrades significantly on the Web content, evidencing the sensibility of the proposed models when dealing with noisy and out-of-domain text. To answer RQ1, we need a methodology that adapts itself better to the noisy scenario of Web. In consequence, we proposed the HORUS approach, a novel named entity recognition approach based on computer vision and text mining techniques. The HORUS approach is able to detect entities on noisy data by extracting heuristics from images and text.

HORUS has been designed to exploit the semantics encoded in the data collected from web

sources and produce global vectors to represent each entity. The key components that allows HORUS to exploit semantics is its ability to process images and associated text, clustering its vectors in a high-dimensional vector space. We empirically demonstrate the advantages of using HORUS to detect named entities on microblogs, showing he benefits of this architecture. Our theoretical and empirical findings indicate that – in comparison with the state-of-the-art – our proposed integration approach HORUS is able to slightly outperform state of the art architectures. Besides, the great advantage is that it does not require any kind of encoded rule and is language-agnostic per nature. Additionally, the HORUS approach remains flexible and applicable to a variety of applications, since its vectors can be embedded into other domains, such as Entity Linking. We formally and empirically prove that the semantics encoded in HORUS are useful resources to perform named entity on noisy data. Based on our findings, we contribute to the state-of-the-art in the area of named entity recognition on noisy data:

We propose a novel methodology to extract relevant information from tokens which is based on the concept of images and news.

We released a novel NER Framework named HORUS, which implements the concepts behind this methodology, generating a set of heuristics to boost the task.

An empirical evaluation to assess the effectiveness of HORUS for the NER task on noisy text. Experiments are executed over the most famous datasets for the task: Ritter, WNUT-15, WNUT-16 and WNUT-17.

The second problem we tackle, it is determining the level of credibility of a given information source and answers the following research question.

RQ2: How to calculate a credibility score for a given information source?

With the growth of the internet, the number of *fake-news* online has been proliferating every year. The consequences of such phenomena are manifold, ranging from lousy decision-making process to bullying and violence episodes. An important step to detect fake-news is to have access to a credibility score for a given information source. However, most of the widely used Web indicators have either been shut-down to the public (e.g., Google PageRank) or are not free for use (Alexa Rank). To answer RQ2 first we review the state-of-the-art approaches and select Likert as trustworthiness scale. The proposed model automatically extracts source reputation cues by transforming HTML metadata into a sequence-to-sequence problem. It then computes a credibility factor, providing valuable insights which can help in belittling dubious and confirming trustful unknown websites. Although further credibility databases exist, they are short-manually curated lists of online sources, which do not scale. Finally, most of the research on the topic is theoretical-based or explore confidential data in a restricted simulation environment. To avoid the need for an expert domain intervention, we propose WebCred, a novel web credibility framework. The empirical evaluations demonstrate the benefits of using our approach to support the problem of detecting fake-news. To test the accuracy of WebCred, we compared its results with the state-of-the-art methods. The proposed model automatically extracts source reputation cues and computes a credibility factor, providing valuable insights which can help in belittling dubious and confirming trustful unknown websites. Experimental results outperform state of

the art in the 2-classes and 5-classes setting. We formally and empirically prove that the use of HTML metadata improves the accuracy of the task of detecting credibility scales for information sources. Based on our findings, we provide the following contributions to the state-of-the-art:

A novel methodology to compute trustworthiness indicators for websites.

A novel web credibility Framework named WebCred, which implements the concepts behind this methodology and is 100% open-source.

An empirical evaluation to assess the effectiveness of WebCred for the web credibility task. Experiments are executed over the most famous datasets for the task: Microsoft and 3C Corpus.

An updated release of the Microsoft Credibility dataset.

The third problem we solve in this thesis, it is the problem of automating the fact-checking task itself. The research question we answer by solving this problem is the following:

RQ3: How to determine the veracity of a given claim?

Fake news is serious problem which has attracted global attention. The information on the internet suffers from noise and corrupt knowledge that may arise due to human and mechanical errors. To further exacerbate this problem, an ever-increasing amount of fake news on social media, or internet in general, has created another challenge to drawing correct information from the web. This huge sea of data makes it difficult for human fact checkers and journalists to assess all the information manually. To answer question RQ3, first, we have proposed a multilingual fact-validation framework named DeFacto to focus on simple claims over structured data. DeFacto is a supervised learning approach which performs verification of triple-like claims using the Web as information source. To perform natural language generation, it relies on a library to perform multi-lingual natural-language patterns transformation. Besides, we manually annotated and generated a gold standard dataset for structured claim, dubbed FactBench. FactBench is a full-fledged multilingual[1] benchmark for the evaluation of fact validation algorithms and is based on FreeBase and DBpedia. All facts in FactBench are scoped with a timespan in which they were true, enabling the validation of temporal relation extraction algorithms. Finally, we have extended DeFacto to also perform verification in a more realistic setting, i.e. complex claims (i.e., natural language). We demonstrate that our approach is able to perform fact-validation over structured claims extracted from KBs. DeFacto achieves an F1 measure of 84.9% on the most realistic fact validation test set (FactBench mix) on DBpedia as well as Freebase data. The temporal extension shows a promising average F1 measure of 70.2% for time point and 65.8% for time period relations. The use of multi-lingual patterns increased the fact validation F1 by 1.5%. We show that DeFacto is able to successfully perform triple validation. Results of the empirical evaluation suggest that DeFacto is able to effectively validate wrong triples existing across different KBs.

In order to explore further scenarios, we have extended DeFacto to also perform verification in a

[1] FactBench currently supports English, German and French

more realistic setting, i.e. complex claims (i.e., natural language). In this release, we also have updated the natural language generation module to handle predicates without restriction, when comparing to the first analysis, which had a limitation in the number of allowed predicates. The experiments suggest that the new component based on word embeedings implemented in DeFacto is able to comprehensively perform the fact-validation task. DeFacto can be applied in numerous use cases, e.g., related to journalism or political debates. To validate the comprehensiveness of DeFacto, we participated inn the most difficult and famous fact-checking challenge, FEVER: Fact Extraction and VERification. It consists of 185,445 claims generated by altering sentences extracted from Wikipedia and subsequently verified without knowledge of the sentence they were derived from. DeFacto outperformed the neural-based baseline system by a decent margin: (0.43×0.18), (0.51×0.48) and (0.38×0.27) for $F1$, *accuracy* and balanced FEVER *Score*, respectively. Last, but not least, we explored methods to further improve the efficiency of the model. In our last study, we focused on the *evidence* retrieval part, proposing a 2-connected layer based on LSTM to merge the extracted proofs. We then compared this to classical and state-of-the-art methods to perform the task. Our experiments show that our model outperforms all the baselines. Compared to the best baseline(XGBoost), it outperforms it by 11%, 10.2% and 18.7% on the FEVER-Support, FEVER-Reject and 3-Class tasks, respectively. We show empirically that DeFacto is able to automatize the fact-checking task, both in structured as well as unstructured scenarios. We enhanced the approach to allow the validation of a variety of different claims, as well as studied and developed a new method to improve the evidence extraction phase. Based on our findings, we provide the following contributions to the state-of-the-art:

1. The development and extension of DeFacto, a RDF fact-checking framework .
2. The extension of DeFacto to increase its coverage on the validation task w.r.t. the allowed predicates.
3. The extension of DeFacto to support triple ranking.
4. The improvement of its architecture w.r.t. new evidence extraction methods.
5. An empirical evaluation to assess the effectiveness of DeFacto for the fact-checking task in the most important fact-checking challenge. Experiments performed validate the viability of the framework on real-use cases scenarios.

The fourth and final question we want to answer in the scope of this thesis is the following:

RQ4: Are existing reproducible research methods sufficient to enable reproducibility?

Over the last decades diverse new scientific methods have been proposed, leading to a massive amount of new articles. However, reproducing them is most of the time a tricky task. In order to compare machine-learning experiment results, they need to be performed thoroughly on the same computing environment, i.e., replication must be performed over the same configurations such as datasets, algorithm and hyperparameters. With this respect, interoperability and metadata management also play an important role. Still, practical experience shows that scientists

and engineers tend to have large output data in their experiments, which is both difficult to analyze and archive properly without provenance metadata. To answer RQ4 we studied existing paradigms and approaches to enable 1) interoperability, 2) scalability and 3) interpretability. These three pillars are the cornerstone to enable reproducible research. We first survey existing attempts to bridge this gap, which were predominantly designed on a workflow-system base. Due to the complexity of the scenario, these attempts imply in several constraints, allowing its implementation only in specific domains (e.g., Bioinformatics). To tackle this problem, ontologies were proposed. However, proposing a high level of granularity, which naturally comes at a price of more complexity to represent the experiments. In order to enable reproducibility of machine learning experiments, in general, we proposed MEX. MEX is a lightweight specification based on Linked Data for interchanging machine-learning metadata over different architectures to achieve a higher level of interoperability. To further foster the dissemination of the protocol, we designed an extensive list of tools and frameworks.

1. We define the first lightweight standard to represent machine learning experiments, a vocabulary dubbed MEX;

2. We design LOG4MEX - a library which allows to export configurations and experiment outcomes;

3. Furthermore, we propose WEB4MEX, a REST interface to export configurations through web calls;

4. Also, MEX-Interfaces is part of the project and defines a set of class interfaces that allow machine learning metadata generation without explicit code implementations;

5. WASOTA has been proposed as a prototype to store experimental metadata online;

6. Finally, we propose ML-Schema, an upper level ontology which maps state-of-the-art ontologies and adopts high level of provenance in a single representation, achieving the maximum level of interoperablity and intepretability; possible.

Through the successful conclusion of these projects, we are able to answer RQ4 and conclude that throught linked data technologies is possible to enable reproducibility of scientific experiments. The proposed methodology and frameworks provide a prompt method to describe experiments with a special focus on data provenance and fulfills the requirements for a long-term maintenance.

7.2 Outlook

In this final section, we describe what we envisage as the future directions for this work.

In the scope of this thesis, we focused on different underlying challenges to automatize the validation task in the fact-checking context. Although very interesting findings have been presented, there is still room to improve the results in each of the problems presented in this thesis. In the following items we summarize the future directions on each of the main contributions of this thesis.

Regarding the challenge of recognizing named entities on noisy data (RQ1: HORUS), we see the following directions to continue and improve the integration approach:

1. Extend HORUS approach to performing classification of other common entities beyond PER, LOC and ORG (e.g., MISC); although the exploration of images and text mining techniques have been extensively tackled in the scope of the thesis, the inclusion on new target classes - especially specific *emerging entities* [82] - have not yet been explored.

2. Our proposed approach has a remarkable negative aspect w.r.t. performance. Currently, its performance is close to 2-3 seconds per token, which is considerably slow and possibly a significant issue in terms of the production environment. The bottleneck is the image feature extraction pipeline. A distributed solution to cache the extracted features would considerably speed up the model‘s response.

3. The metadata provided by HORUS has the potential to become an essential asset to several other applications in NLP, such as entity linking and question answering, for instance. This integration should be validated and may push the state of the art, especially in noisy text.

4. Finally, to improve performance and reduce costs with search engine calls, we propose the integration with open Knowledge Bases, such as DBpedia, YAGO. The trade-off might be worth and shall be validated in terms of costs and performance.

Regarding the trustworthiness model (RQ2: WebCred), it can be extended with the following ideas:

1. Integrate our new proposed method with graph-based solutions [23] to assess credibility in order to improve user experience and give more insights to the end-user.

2. The concept of credibility strongly relies on the user‘s feedback. An interesting approach would be extending the framework to an open crowdsource based architecture where users could interact with the system, giving feedback to the model‘s response. This information could be either used as a feature to a supervised model as well as to design a reinforcement learning-based strategy to tackle the problem.

3. Documents which have a certain number of false statements should be automatically labeled as non-credible sources. The idea of performing macro fact-checking over all statements existing in a given information source can be an exciting approach to derive a final credibility score.

Regarding our automated fact-checking approach (RQ3: DeFacto) we plan the following to the future:

1. All new models proposed (HORUS and WebCred) have a high potential to improve the performance of our fact-checking framework and thus should be validated in DeFacto.

2. An exciting line of research would be to study visual fake-news. Generative Adversarial Networks (GANs) [210] have had massive success since they were first proposed in 2014.

With the adoption of this technology, videos and photos can be used to spread fake-news in a different level of difficulty for existing automated fact-checking models.

3. The usage of word embeddings to enhance the evidence extraction phase is a promising research line. In this case, we suggest to continue and extend the evidence extraction method we proposed in this thesis. Possible extensions could be considering not only the explicit knowledge encoded available in related documents but the implicit knowledge too, which can be acquired through *common-sense* frameworks. Transformers also have great potential to boost the task.

4. In the document retrieval component, we incentive the study of entity-linking methods to potentially improve recall. Our current architecture does not perform disambiguation of entities, which is an important step in the pipeline.

5. Untimely, we also suggest integrating into DeFacto a feedback module, shifting the architecture to a human-in-the-loop paradigm. Positive and negative feedback from specialists (e.g., journalists) can be a great asset to improve the overall model‘s performance.

Future of reproducible research (RQ4: MEX and ML-Schema) should encompass more transparent methods to represent data and metadata:

1. The most challenging issue regarding this topic is indirectly related to time management. Scientists face a never-ending competition to solve complex problems within the minimum amount of time. They are thus constantly pushed to deliver more and faster, which implies in the lack of proper representation of scientific experiments. In the engineering side, one exciting solution would be integrating ML vocabularies and ontologies through the proposed canonical format ML-Schema into famous ML frameworks, such as scikit-learn.

2. A very ambitious solution lies in constructing machines to automatically reading source-code and generating metadata representation out of it, without human interference.

In terms of the applicability of the approaches we see a lot of opportunities to solve integration problems on the following domains:

In the health domain, HORUS can be adopted as a novel solution to detect unusual entities for NER models, e.g., products and ingredients. Due to its language-agnostic characteristic, it may be applied to several languages without restrictions.

Furthermore, DeFacto can be applied to verify knowledge graphs containing information of alimentary diet. For instance, by checking relationships among the food and diseases (e.g., "*carrots are a weight loss friendly food and have been linked to lower cholesterol levels and improved eye health.*"

After most famous web credibility tools have been shut down to the public, WebCred can stand as the only free solution to bring credibility information to internet users if deployed in large scale.

All tools and frameworks designed under the MEX Project have a high potential to create

a general controlled environment to foster reproducibility.

7.3 Closing Remarks

With the increase in false fact circulation across different social media platforms, it has become pertinent to validate the claims and statements released online. In this thesis, we shed light on existing challenges and propose solutions to automatize the fact-checking validation task. Future research work can build upon the contributions presented in this thesis to devise more accurate approaches. Additionally, the pieces of software produced during the development of this thesis are impacting several application domains (trustworthiness, fact-checking, named entity recognition and reproducible research). The results of this thesis have been appreciated by the Semantic Web Research and Natural Language Processing Communities. Moreover, the results of this thesis are part of further European Union research proposals and projects.

Bibliography

[1] Y. Oulabi and C. Bizer,
"Estimating missing temporal meta-information using Knowledge-Based-Trust",
CEUR workshop proceedings, vol. 1959, RWTH, 2017 Paper (cit. on p. 1).

[2] J. Dalton, L. Dietz and J. Allan,
"Entity Query Feature Expansion Using Knowledge Base Links",
Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '14, ACM, 2014 365,
ISBN: 978-1-4503-2257-7, URL: http://doi.acm.org/10.1145/2600428.2609628
(cit. on p. 1).

[3] J. H. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubézy, H. Eriksson, N. F. Noy and S. W. Tu,
The evolution of Protégé: an environment for knowledge-based systems development,
International Journal of Human-computer studies **58** (2003) 89 (cit. on p. 1).

[4] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann and S. Auer,
Quality assessment for linked data: A survey, Semantic Web **7** (2015) 63 (cit. on p. 1).

[5] L. Berti-Équille and J. Borge-Holthoefer, *Veracity of Data: From Truth Discovery Computation Algorithms to Models of Misinformation Dynamics*,
Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2015
(cit. on p. 1).

[6] B. Saha and D. Srivastava, "Data quality: The other face of Big Data",
IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014, 2014 1294 (cit. on p. 1).

[7] B. Saha and D. Srivastava, "Data quality: The other face of Big Data",
2014 IEEE 30th International Conference on Data Engineering, 2014 1294 (cit. on p. 1).

[8] K. Fridkin, P. J. Kenney and A. Wintersieck, *Liar, Liar, Pants on Fire: How Fact-Checking Influences Citizens' Reactions to Negative Advertising*,
Political Communication **32** (2015) 127,
eprint: https://doi.org/10.1080/10584609.2014.914613,
URL: https://doi.org/10.1080/10584609.2014.914613 (cit. on p. 2).

[9] A. Vlachos and S. Riedel, *Fact Checking: Task definition and dataset construction*,
ACL 2014 (2014) 18 (cit. on pp. 2, 18, 19).

[10] M. Rashid, M. Torchiano, G. Rizzo, N. Mihindukulasooriya and O. Corcho,
A quality assessment approach for evolving knowledge bases, Semantic Web (2018) 1
(cit. on p. 2).

[11] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann and S. Auer, *Quality Assessment for Linked Data: A Survey*, Semantic Web Journal (2015), URL: http://www.semantic-web-journal.net/content/quality-assessment-linked-data-survey (cit. on p. 2).

[12] G. Karadzhov, P. Nakov, L. Màrquez, A. Barrón-Cedeño and I. Koychev, *Fully Automated Fact Checking Using External Sources*, CoRR **abs/1710.00341** (2017), arXiv: 1710.00341, URL: http://arxiv.org/abs/1710.00341 (cit. on pp. 2, 18, 52).

[13] S. Vosoughi, D. Roy and S. Aral, *The spread of true and false news online*, Science **359** (2018) 1146 (cit. on p. 2).

[14] M. Himma-Kadakas et al., *Alternative facts and fake news entering journalistic content production cycle*, Cosmopolitan Civil Societies: An Interdisciplinary Journal **9** (2017) 25 (cit. on p. 2).

[15] N. Hassan, B. Adair, J. T. Hamilton, C. Li, M. Tremayne, J. Yang and C. Yu, *The quest to automate fact-checking*, world (2015) (cit. on pp. 3, 53).

[16] D. M. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild et al., *The science of fake news*, Science **359** (2018) 1094 (cit. on p. 3).

[17] L. Derczynski, D. Maynard, G. Rizzo, M. van Erp, G. Gorrell, R. Troncy, J. Petrak and K. Bontcheva, *Analysis of named entity recognition and linking for tweets*, Information Processing & Management **51** (2015) 32 (cit. on p. 3).

[18] D. Esteves, R. Peres, J. Lehmann and G. Napolitano, "Named Entity Recognition in Twitter using Images and Text", *3rd International Workshop on Natural Language Processing for Informal Text (NLPIT 2017)*, 2017, URL: https://www.researchgate.net/publication/317721565_Named_Entity_Recognition_in_Twitter_using_Images_and_Text (cit. on pp. 3, 25).

[19] A. Ritter, S. Clark, O. Etzioni et al., "Named entity recognition in tweets: an experimental study", *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2011 1524 (cit. on pp. 3, 19, 26, 30, 31).

[20] X. Liu, M. Zhou, F. Wei, Z. Fu and X. Zhou, "Joint inference of named entity recognition and normalization for tweets", *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, Association for Computational Linguistics, 2012 526 (cit. on pp. 3, 20, 26).

[21] A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan and A. Doan, *Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-based Approach*, Proc. VLDB Endow. **6** (2013) 1126, ISSN: 2150-8097, URL: http://dx.doi.org/10.14778/2536222.2536237 (cit. on pp. 3, 26).

[22] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition", *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, 2009 147 (cit. on p. 3).

[23] S. Nakamura, S. Konishi, A. Jatowt, H. Ohshima, H. Kondo, T. Tezuka, S. Oyama and K. Tanaka, *Trustworthiness analysis of web search results*, Research and advanced technology for digital libraries (2007) 38 (cit. on pp. 4, 118).

[24] R. Mochales and M.-F. Moens, *Argumentation mining*, Artificial Intelligence and Law **19** (2011) 1 (cit. on p. 4).

[25] G. Rocha, H. L. Cardoso and J. Teixeira, "ArgMine: A Framework for Argumentation Mining", *Computational Processing of the Portuguese Language-12th International Conference, PROPOR*, 2016 13 (cit. on p. 4).

[26] M. Lippi and P. Torroni, *Argumentation mining: State of the art and emerging trends*, ACM Transactions on Internet Technology (TOIT) **16** (2016) 10 (cit. on pp. 4, 13).

[27] D. Gerber and A.-C. Ngonga Ngomo, "From RDF to Natural Language and Back", *Towards the Multilingual Semantic Web*, Springer, 2014 (cit. on p. 4).

[28] S. Riedel, L. Yao, A. McCallum and B. M. Marlin, *Relation extraction with matrix factorization and universal schemas*, (2013) (cit. on p. 4).

[29] J. Pasternack and D. Roth, "Generalized fact-finding", *Proceedings of the 20th international conference companion on World wide web*, ACM, 2011 99 (cit. on p. 13).

[30] G. Rocha, H. Lopes Cardoso and J. Teixeira, "ArgMine: a framework for argumentation mining", *Computational Processing of the Portuguese Language-12th International Conference, PROPOR*, 2016 13 (cit. on p. 13).

[31] D. Gerber, D. Esteves, J. Lehmann, L. Bühmann, R. Usbeck, A.-C. N. Ngomo and R. Speck, *Defacto—temporal and multilingual deep fact validation*, Web Semantics: Science, Services and Agents on the World Wide Web **35** (2015) 85 (cit. on pp. 14, 15, 39, 43, 51, 52, 54, 55, 72).

[32] M. Samadi, M. M. Veloso and M. Blum, "OpenEval: Web Information Query Evaluation.", *AAAI*, Citeseer, 2013 (cit. on pp. 14, 15).

[33] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld and A. Yates, "Web-scale Information Extraction in Knowitall: (Preliminary Results)", *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, ACM, 2004 100, ISBN: 1-58113-844-X, URL: http://doi.acm.org/10.1145/988672.988687 (cit. on pp. 14, 15).

[34] H. Bast, B. Buchhold and E. Haussmann, "Relevance scores for triples from type-like relations", *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2015 243 (cit. on pp. 14, 16, 17).

[35] B. Ding, Q. Wang and B. Wang, "Leveraging Text and Knowledge Bases for Triple Scoring: An Ensemble Approach—The BOKCHOY Triple Scorer at WSDM Cup 2017", *WSDM Cup 2017 Notebook Papers, February 10, Cambridge, UK*, ed. by M. Potthast, S. Heindorf and H. Bast, CEUR-WS.org, 2017, URL: http://www.wsdm-cup-2017.org/proceedings.html (cit. on pp. 14, 17).

[36] F. Hasibi, D. Garigliotti, S. Zhang and K. Balog, "Supervised Ranking of Triples for Type-Like Relations—The Cress Triple Scorer at the WSDM Cup 2017", *WSDM Cup 2017 Notebook Papers, February 10, Cambridge, UK*, ed. by M. Potthast, S. Heindorf and H. Bast, CEUR-WS.org, 2017, URL: http://www.wsdm-cup-2017.org/proceedings.html (cit. on pp. 14, 17).

[37] V. Zmiycharov, D. Alexandrov, P. Nakov, I. Koychev and Y. Kiprov, "Finding People's Professions and Nationalities Using Distant Supervision—The FMI@SU "goosefoot" team at the WSDM Cup 2017 Triple Scoring Task", *WSDM Cup 2017 Notebook Papers, February 10, Cambridge, UK*, ed. by M. Potthast, S. Heindorf and H. Bast, CEUR-WS.org, 2017, URL: http://www.wsdm-cup-2017.org/proceedings.html (cit. on pp. 14, 17).

[38] S. G. Hong and M. Y. Yi, *Plausibility Assessment of Triples with Instance-based Learning Distantly Supervised by Background Knowledge*, Submitted to the Semantic Web Journal (under review) (2017), URL: http://www.semantic-web-journal.net/system/files/swj1546.pdf (cit. on pp. 13, 14).

[39] G. A. Miller, *WordNet: A Lexical Database for English*, Commun. ACM **38** (1995) 39, ISSN: 0001-0782, URL: http://doi.acm.org/10.1145/219717.219748 (cit. on p. 14).

[40] C. Macleod, R. Grishman, A. Meyers, L. Barrett and R. Reeves, "NOMLEX: A Lexicon of Nominalizations", *In Proceedings of Euralex98*, 1998 187 (cit. on p. 14).

[41] S. G. Hong, S.-h. Cho and M. Y. Yi, "Unsupervised Verb Inference from Nouns Crossing Root Boundary", *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, Dublin City University and Association for Computational Linguistics, 2014 1248, URL: http://www.aclweb.org/anthology/C14-1118 (cit. on p. 14).

[42] X. Li, W. Meng and C. Yu, "T-verifier: Verifying Truthfulness of Fact Statements", *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, IEEE Computer Society, 2011 63, ISBN: 978-1-4244-8959-6, URL: http://dx.doi.org/10.1109/ICDE.2011.5767859 (cit. on p. 14).

[43] M. Samadi, P. Talukdar, M. Veloso and M. Blum, "ClaimEval: Integrated and Flexible Framework for Claim Evaluation Using Credibility of Sources", *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, AAAI Press, 2016 222, URL: http://dl.acm.org/citation.cfm?id=3015812.3015845 (cit. on p. 14).

[44] F. Li, X. L. Dong, A. Langen and Y. Li, *Knowledge Verification for LongTail Verticals*, PVLDB **10** (2017) 1370 (cit. on p. 15).

[45] S. Soderland, O. Etzioni, T. Shaked and D. Weld, "The use of Web-based statistics to validate information extraction", *AAAI-04 Workshop on Adaptive Text Extraction and Mining*, 2004 21 (cit. on p. 15).

[46] C. Cortes and V. Vapnik, *Support-Vector Networks*, Mach. Learn. **20** (1995) 273, ISSN: 0885-6125 (cit. on p. 15).

[47] D. Gerber and A.-C. Ngonga Ngomo,
"Extracting Multilingual Natural-Language Patterns for RDF Predicates",
Proceedings of EKAW, 2012 (cit. on p. 16).

[48] J. Thorne and A. Vlachos,
Automated Fact Checking: Task formulations, methods and future directions,
arXiv preprint arXiv:1806.07687 (2018) (cit. on p. 18).

[49] N. Lee, C.-S. Wu and P. Fung, "Improving Large-Scale Fact-Checking using Decomposable Attention Models and Lexical Tagging", *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018 1133 (cit. on p. 18).

[50] J. Thorne, A. Vlachos, O. Cocarascu, C. Christodoulopoulos and A. Mittal,
The Fact Extraction and VERification (FEVER) Shared Task,
arXiv preprint arXiv:1811.10971 (2018) (cit. on pp. 18, 19, 52).

[51] M. Taniguchi, T. Taniguchi, T. Takahashi, Y. Miura and T. Ohkuma,
"Integrating Entity Linking and Evidence Ranking for Fact Extraction and Verification",
Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), 2018 124 (cit. on p. 18).

[52] K. Popat, S. Mukherjee, A. Yates and G. Weikum,
DeClarE: Debunking Fake News and False Claims using Evidence-Aware Deep Learning,
CoRR **abs/1809.06416** (2018), arXiv: 1809.06416,
URL: http://arxiv.org/abs/1809.06416 (cit. on pp. 18, 53).

[53] M. Tosik, A. Mallia and K. Gangopadhyay,
Debunking Fake News One Feature at a Time, arXiv preprint arXiv:1808.02831 (2018) (cit. on pp. 18, 77–80).

[54] Y. Yang, L. Zheng, J. Zhang, Q. Cui, Z. Li and P. S. Yu,
TI-CNN: Convolutional Neural Networks for Fake News Detection,
CoRR **abs/1806.00749** (2018), arXiv: 1806.00749,
URL: http://arxiv.org/abs/1806.00749 (cit. on p. 18).

[55] s. li sizhen, S. Zhao, B. Cheng and H. Yang,
"An End-to-End Multi-task Learning Model for Fact Checking",
Proceedings of the First Workshop on Fact Extraction and VERification (FEVER),
Association for Computational Linguistics, 2018 138,
URL: http://aclweb.org/anthology/W18-5523 (cit. on p. 18).

[56] C. Conforti, M. T. Pilehvar and N. Collier, "Towards Automatic Fake News Detection: Cross-Level Stance Detection in News Articles",
Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), 2018 40 (cit. on p. 18).

[57] W. Yin and D. Roth,
TwoWingOS: A Two-Wing Optimization Strategy for Evidential Claim Verification,
CoRR **abs/1808.03465** (2018), arXiv: 1808.03465,
URL: http://arxiv.org/abs/1808.03465 (cit. on p. 18).

[58] J. Thorne, A. Vlachos, C. Christodoulopoulos and A. Mittal,
FEVER: a large-scale dataset for Fact Extraction and VERification,
CoRR **abs/1803.05355** (2018), arXiv: 1803.05355,
URL: http://arxiv.org/abs/1803.05355 (cit. on pp. 18, 52, 53).

[59] R. Baly, G. Karadzhov, D. Alexandrov, J. R. Glass and P. Nakov,
Predicting Factuality of Reporting and Bias of News Media Sources,
CoRR **abs/1810.01765** (2018), arXiv: 1810.01765,
URL: http://arxiv.org/abs/1810.01765 (cit. on pp. 19, 52).

[60] K. Popat, S. Mukherjee, J. Strötgen and G. Weikum, "Where the truth lies: Explaining the credibility of emerging claims on the web and social media",
Proceedings of the 26th International Conference on World Wide Web Companion,
International World Wide Web Conferences Steering Committee, 2017 1003
(cit. on pp. 19, 52, 53).

[61] G. L. Ciampaglia, P. Shiralkar, L. M. Rocha, J. Bollen, F. Menczer and A. Flammini,
Computational fact checking from knowledge networks, PloS one **10** (2015) e0128193
(cit. on p. 19).

[62] J. R. Finkel, T. Grenager and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling",
Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics,
Association for Computational Linguistics, 2005 363 (cit. on p. 19).

[63] J. Baldridge, *The OpenNLP project*,
URL: http://opennlp.apache.org/index.html, (accessed 17 May 2016) (2005)
(cit. on p. 19).

[64] D. Nadeau, *Balie—baseline information extraction: Multilingual information extraction from text with machine learning and natural language techniques*, tech. rep.,
Technical report, University of Ottawa, 2005 (cit. on p. 19).

[65] P. Ferragina and U. Scaiella,
"Tagme: on-the-fly annotation of short text fragments (by wikipedia entities)",
Proceedings of the 19th ACM international conference on Information and knowledge management, ACM, 2010 1625 (cit. on p. 19).

[66] L. Ratinov and D. Roth,
"Design Challenges and Misconceptions in Named Entity Recognition", *CoNLL*, 2009,
URL: http://cogcomp.cs.illinois.edu/papers/RatinovRo09.pdf (cit. on p. 19).

[67] G. Luo, X. Huang, C.-Y. Lin and Z. Nie,
"Joint named entity recognition and disambiguation", *Proc. EMNLP*, 2015
(cit. on p. 19).

[68] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. Kuksa,
Natural language processing (almost) from scratch,
Journal of Machine Learning Research **12** (2011) 2493 (cit. on p. 19).

[69] R. Al-Rfou, V. Kulkarni, B. Perozzi and S. Skiena,
"Polyglot-NER: Massive multilingual named entity recognition", *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, British Columbia, Canada*,
SIAM, 2015 (cit. on p. 19).

[70] J. P. Chiu and E. Nichols, *Named entity recognition with bidirectional LSTM-CNNs*, arXiv preprint arXiv:1511.08308 (2015) (cit. on p. 19).

[71] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer, *Neural architectures for named entity recognition*, arXiv preprint arXiv:1603.01360 (2016) (cit. on pp. 19, 26, 30, 34).

[72] K. Bontcheva, L. Derczynski, A. Funk, M. A. Greenwood, D. Maynard and N. Aswani, "TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text.", *RANLP*, 2013 83 (cit. on pp. 19, 31).

[73] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic et al., *Text Processing with GATE (Version 6)(2011)*, University of Sheffield Department of Computer Science **15** (2011) (cit. on p. 20).

[74] N. Limsopatham and N. Collier, *Bidirectional lstm for named entity recognition in twitter messages*, WNUT 2016 (2016) 145 (cit. on pp. 20, 26).

[75] C. Lopez, I. Partalas, G. Balikas, N. Derbas, A. Martin, C. Reutenauer, F. Segond and M.-R. Amini, *CAp 2017 challenge: Twitter Named Entity Recognition*, arXiv preprint arXiv:1707.07568 (2017) (cit. on p. 20).

[76] R. Peres, D. Esteves and G. Maheshwari, "Bidirectional LSTM with a Context Input Window for Named Entity Recognition in Tweets", *Proceedings of the Knowledge Capture Conference*, ACM, 2017 42 (cit. on p. 20).

[77] H. He and X. Sun, "A Unified Model for Cross-Domain and Semi-Supervised Named Entity Recognition in Chinese Social Media.", *AAAI*, 2017 3216 (cit. on p. 20).

[78] D. Esteves, R. Peres, J. Lehmann and G. Napolitano, *Named Entity Recognition in Twitter using Images and Text*, arXiv preprint arXiv:1710.11027 (2017) (cit. on pp. 20, 25, 32, 34, 35).

[79] Q. Zhang, J. Fu, X. Liu and X. Huang, "Adaptive Co-attention Network for Named Entity Recognition in Tweets.", *AAAI*, 2018 (cit. on p. 20).

[80] T. Baldwin, M.-C. de Marneffe, B. Han, Y.-B. Kim, A. Ritter and W. Xu, "Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition", *Proceedings of the Workshop on Noisy User-generated Text*, 2015 126 (cit. on pp. 20, 37).

[81] B. Strauss, B. Toma, A. Ritter, M.-C. de Marneffe and W. Xu, "Results of the wnut16 named entity recognition shared task", *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, 2016 138 (cit. on p. 20).

[82] L. Derczynski, E. Nichols, M. van Erp and N. Limsopatham, "Results of the WNUT2017 shared task on novel and emerging entity recognition", *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017 140 (cit. on pp. 20, 118).

[83] J. Sobel, *A theory of credibility*, The Review of Economic Studies **52** (1985) 557 (cit. on p. 20).

[84] B. Fogg and H. Tseng, "The elements of computer credibility", *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ACM, 1999 80 (cit. on p. 20).

[85] B. Fogg, J. Marshall, O. Laraki, A. Osipovich, C. Varma, N. Fang, J. Paul, A. Rangnekar, J. Shon, P. Swani et al., "What makes Web sites credible?: a report on a large quantitative study", *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 2001 61 (cit. on p. 20).

[86] B. J. Fogg, C. Soohoo, D. R. Danielson, L. Marable, J. Stanford and E. R. Tauber, "How do users evaluate the credibility of Web sites?: a study with over 2,500 participants", *Proceedings of the 2003 conference on Designing for user experiences*, ACM, 2003 1 (cit. on pp. 20, 21, 42).

[87] S. Nakamura, S. Konishi, A. Jatowt, H. Ohshima, H. Kondo, T. Tezuka, S. Oyama and K. Tanaka, "Trustworthiness Analysis of Web Search Results", English, *Proceedings of the 11th European conference on Research and Advanced Technology for Digital Libraries*, ed. by L. Kovács, N. Fuhr and C. Meghini, vol. 4675, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2007 38, ISBN: 978-3-540-74850-2, URL: http://dx.doi.org/10.1007/978-3-540-74851-9_4 (cit. on pp. 20, 21).

[88] B. J. Fogg, "Prominence-interpretation theory: Explaining how people assess credibility online", *CHI'03 extended abstracts on human factors in computing systems*, ACM, 2003 722 (cit. on pp. 21, 49).

[89] J. Schwarz and M. Morris, "Augmenting web pages and search results to support credibility assessment", *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2011 1245 (cit. on pp. 21, 22, 43).

[90] K. D. Giudice, "Crowdsourcing credibility: The impact of audience feedback on Web page credibility", *Proceedings of the 73rd ASIS&T Annual Meeting on Navigating Streams in an Information Ecosystem-Volume 47*, American Society for Information Science, 2010 59 (cit. on p. 21).

[91] S. Liu, M. d'Aquin and E. Motta, "Towards Linked Data Fact Validation through Measuring Consensus.", *LDQ@ ESWC*, 2015 (cit. on pp. 21, 40).

[92] A. A. Shah, S. D. Ravana, S. Hamid and M. A. Ismail, *Web credibility assessment: affecting factors and assessment techniques*, Information Research **20** (2015) 20 (cit. on pp. 21, 42, 49).

[93] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun and W. Zhang, *Knowledge-based Trust: Estimating the Trustworthiness of Web Sources*, Proc. VLDB Endow. **8** (2015) 938, ISSN: 2150-8097, URL: https://doi.org/10.14778/2777598.2777603 (cit. on pp. 21, 42, 49).

[94] H. Singal and S. Kohli,
Trust Necessitated through Metrics: Estimating the Trustworthiness of Websites,
Procedia Computer Science **85** (2016) 133 (cit. on p. 21).

[95] M. Kakol, R. Nielek and A. Wierzbicki, *Understanding and predicting Web content credibility using the Content Credibility Corpus*,
Information Processing & Management **53** (2017) 1043 (cit. on pp. 22, 43).

[96] A. Abbasi, Z. Zhang, D. Zimbra, H. Chen and J. F. Nunamaker Jr,
Detecting fake websites: the contribution of statistical learning theory,
Mis Quarterly (2010) 435 (cit. on p. 22).

[97] S. Afroz and R. Greenstadt, *PhishZoo: Detecting Phishing Websites by Looking at Them*,
2012 IEEE Sixth International Conference on Semantic Computing **00** (2011) 368
(cit. on p. 22).

[98] D. G. Lowe, "Object Recognition from Local Scale-Invariant Features",
Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2,
ICCV '99, IEEE Computer Society, 1999 1150, ISBN: 0-7695-0164-8,
URL: http://dl.acm.org/citation.cfm?id=850924.851523 (cit. on p. 22).

[99] A. Olteanu, S. Peshterliev, X. Liu and K. Aberer,
"Web credibility: features exploration and credibility prediction",
European conference on information retrieval, Springer, 2013 557
(cit. on pp. 22, 42, 43, 47, 49, 50).

[100] A. Wawer, R. Nielek and A. Wierzbicki,
"Predicting webpage credibility using linguistic features",
Proceedings of the 23rd international conference on world wide web, ACM, 2014 1135
(cit. on pp. 22, 42, 43, 47, 50).

[101] P. J. Stone and E. B. Hunt, "A Computer Approach to Content Analysis: Studies Using the General Inquirer System",
Proceedings of the May 21-23, 1963, Spring Joint Computer Conference,
AFIPS '63 (Spring), ACM, 1963 241,
URL: http://doi.acm.org/10.1145/1461551.1461583 (cit. on pp. 22, 45).

[102] C. M. Micheel, S. J. Nass, G. S. Omenn et al.,
Evolution of translational omics: lessons learned and the path forward,
National Academies Press, 2012 (cit. on pp. 23, 81).

[103] D. D. Roure, C. Goble and R. Stevens, *The Design and Realisation of the myExperiment Virtual Research Environment for Social Sharing of Workflows*,
Future Generation Computer Systems **25** (2009) 561 (cit. on pp. 23, 24).

[104] Y. Gil, V. Ratnakar, J. Kim, P. A. González-Calero, P. T. Groth, J. Moody and E. Deelman, *Wings: Intelligent Workflow-Based Design of Computational Experiments*,
IEEE Intelligent Systems (2011) 62 (cit. on pp. 23, 82, 83).

[105] O. Tcheremenskaia, R. Benigni, I. Nikolova, N. Jeliazkova, S. E. Escher, M. Batke, T. Baier, V. Poroikov, A. Lagunin, M. Rautenberg et al.,
"OpenTox predictive toxicology framework: toxicological ontology and semantic media wiki-based OpenToxipedia", *Journal of biomedical semantics*, vol. 3, 1,
BioMed Central, 2012 S7 (cit. on pp. 23, 24, 82, 83, 87).

[106] J. Vanschoren, J. N. van Rijn, B. Bischl and L. Torgo,
OpenML: Networked Science in Machine Learning, SIGKDD Explor. Newsl. (2014) 49,
ISSN: 1931-0145 (cit. on pp. 23, 24, 83, 111).

[107] J. Kim, E. Deelman, Y. Gil, G. Mehta and V. Ratnakar,
Provenance Trails in the Wings-Pegasus System,
Journal of Concurrency And Computation: Practice And Experience (2008) 587,
ISSN: 1532-0626 (cit. on p. 24).

[108] C. M. Keet, A. Ławrynowicz, C. d'Amato, A. Kalousis, P. Nguyen, R. Palma, R. Stevens and M. Hilario, *The Data Mining OPtimization Ontology*,
Web Semantics: Science, Services and Agents on the World Wide Web (2015) 43,
ISSN: 1570-8268,
URL: http://www.sciencedirect.com/science/article/pii/S1570826815000025
(cit. on pp. 24, 83).

[109] P. Panov, L. Soldatova and S. Džeroski,
"OntoDM-KDD: ontology for representing the knowledge discovery process",
International Conference on Discovery Science, Springer, 2013 126
(cit. on pp. 24, 83, 91).

[110] J. Vanschoren and L. Soldatova, "Exposé: An ontology for data mining experiments",
International workshop on third generation data mining: Towards service-oriented knowledge discovery (SoKD-2010), 2010 31 (cit. on pp. 24, 83, 91, 111).

[111] H. Blockeel and J. Vanschoren, "Experiment Databases: Towards an Improved Experimental Methodology in Machine Learning", English,
Springer Berlin Heidelberg, 2007 6, ISBN: 978-3-540-74975-2,
URL: http://dx.doi.org/10.1007/978-3-540-74976-9_5 (cit. on pp. 24, 83).

[112] S. A. Coelho, D. Moussallem, G. C. Publio and D. Esteves,
TANKER: Distributed Architecture for Named Entity Recognition and Disambiguation,
arXiv preprint arXiv:1708.09230 (2017), URL: https://arxiv.org/abs/1708.09230
(cit. on p. 25).

[113] R. Peres, D. Esteves and G. Maheshwari, "Bidirectional LSTM with a Context Input Window for Named Entity Recognition in Tweets",
Proceedings of the 9th International Conference on Knowledge Capture, ACM, 2017
(cit. on p. 25).

[114] D. Esteves, A. Fischer, R. Peres and J. Lehmann, "Boosting Named Entity Recognition Architectures using Images and News Extracted from the Web",
Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18 (Manuscript submitted for publication), 2018 (cit. on p. 25).

[115] M. Tkachenko and A. Simanovsky, "Named entity recognition: Exploring features.",
KONVENS, 2012 118 (cit. on p. 26).

[116] Z. Huang, W. Xu and K. Yu, *Bidirectional LSTM-CRF models for sequence tagging*,
arXiv preprint arXiv:1508.01991 (2015) (cit. on pp. 26, 30, 32, 34).

[117] Y.-s. Chang and Y.-H. Sung, *Applying name entity recognition to informal text*,
Recall **1** (2005) 1 (cit. on p. 26).

[118] X. Liu, S. Zhang, F. Wei and M. Zhou, "Recognizing named entities in tweets", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, Association for Computational Linguistics, 2011 359 (cit. on p. 26).

[119] D. G. Lowe, "Object recognition from local scale-invariant features", *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, 1999 1150 (cit. on p. 28).

[120] J. Sivic, A. Zisserman et al., "Video google: A text retrieval approach to object matching in videos.", *iccv*, vol. 2, 2003 1470 (cit. on p. 28).

[121] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching", *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2007 1 (cit. on p. 28).

[122] J. MacQueen et al., "Some methods for classification and analysis of multivariate observations", *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, 14, Oakland, CA, USA., 1967 281 (cit. on p. 28).

[123] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories", *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, IEEE, 2005 524 (cit. on p. 28).

[124] L. Fei-Fei, R. Fergus and P. Perona, *Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories*, Computer Vision and Image Understanding **106** (2007) 59 (cit. on p. 28).

[125] S. K. O. Tursun, "A Challenging Big Dataset for Benchmarking Trademark Retrieval", *IAPR Conference on Machine Vision and Applications*, 2015 (cit. on p. 28).

[126] T. Fletcher, *Support vector machines explained*, Online]. http://sutikno. blog. undip. ac. id/files/2011/11/SVM-Explained. pdf.[Accessed 06 06 2013] (2009) (cit. on p. 28).

[127] L. Breiman, *Random Forests*, Mach. Learn. **45** (2001) 5, ISSN: 0885-6125, URL: https://doi.org/10.1023/A:1010933404324 (cit. on p. 30).

[128] N. Nguyen and Y. Guo, "Comparisons of sequence labeling algorithms and extensions", *Proceedings of the 24th international conference on Machine learning*, ACM, 2007 681 (cit. on p. 30).

[129] L. Derczynski and K. Bontcheva, "Passive-aggressive sequence labeling with discriminative post-editing for recognising person entities in tweets", *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, 2014 69 (cit. on p. 30).

[130] L. Derczynski, E. Nichols, M. van Erp and N. Limsopatham, "Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition", *Proceedings of the 3rd Workshop on Noisy User-generated Text*, Association for Computational Linguistics, 2017 140, URL: http://aclweb.org/anthology/W17-4418 (cit. on p. 30).

[131] S. Hochreiter and J. Schmidhuber, *Long Short-Term Memory*, Neural Comput. **9** (1997) 1735, ISSN: 0899-7667, URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735 (cit. on p. 30).

[132] X. Ma and E. Hovy, *End-to-end sequence labeling via bi-directional lstm-cnns-crf*, arXiv preprint arXiv:1603.01354 (2016) (cit. on pp. 30, 34).

[133] D. Etter, F. Ferraro, R. Cotterell, O. Buzek and B. Van Durme, *Nerit: Named entity recognition for informal text*, The Johns Hopkins University, the Human Language Technology Center of Excellence, HLTCOE 810Wyman Park Drive Baltimore, Maryland 21211, Tech. Rep. (2013) (cit. on p. 31).

[134] L. Derczynski, S. Chester and K. S. Bøgh, "Tune your brown clustering, please", *International Conference Recent Advances in Natural Language Processing, RANLP*, vol. 2015, Association for Computational Linguistics, 2015 110 (cit. on p. 32).

[135] J. Turian, L. Ratinov and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning", *Proceedings of the 48th annual meeting of the association for computational linguistics*, Association for Computational Linguistics, 2010 384 (cit. on p. 32).

[136] D. Esteves, A. J. Reddy, P. Chawla and J. Lehmann, "Belittling the Source: Trustworthiness Indicators to Obfuscate Fake News on the Web", *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, 2018 50 (cit. on p. 39).

[137] X. Li, X. L. Dong, K. Lyons, W. Meng and D. Srivastava, *Truth Finding on the Deep Web: Is the Problem Solved?*, Proc. VLDB Endow. **6** (2012) 97, ISSN: 2150-8097, URL: http://dx.doi.org/10.14778/2535568.2448943 (cit. on p. 40).

[138] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata and R. Procter, *Detection and Resolution of Rumours in Social Media: A Survey*, CoRR **abs/1704.00656** (2017), URL: http://arxiv.org/abs/1704.00656 (cit. on p. 40).

[139] Netcraft, *Netcraft Survey (2016)*, http://www.webcitation.org/6lhJlHtez, Accessed: 2017-10-01, 2016 (cit. on p. 40).

[140] A. Haas and J. Unkel, *Ranking versus reputation: perception and effects of search result credibility*, Behaviour & Information Technology **36** (2017) 1285 (cit. on pp. 40, 42).

[141] G. Erkan and D. R. Radev, *LexRank: Graph-based Lexical Centrality As Salience in Text Summarization*, J. Artif. Int. Res. **22** (2004) 457, ISSN: 1076-9757, URL: http://dl.acm.org/citation.cfm?id=1622487.1622501 (cit. on p. 44).

[142] J. Steinberger and K. Ježek, "Using latent semantic analysis in text summarization and summary evaluation", *In Proc. ISIM '04*, 2004 93 (cit. on p. 44).

[143] L. Si and J. Callan, "A Statistical Model for Scientific Readability", *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, ACM, 2001 574, ISBN: 1-58113-436-3,
URL: http://doi.acm.org/10.1145/502585.502695 (cit. on p. 44).

[144] M. Baker, *1,500 scientists lift the lid on reproducibility*, Nature News **533** (2016) 452 (cit. on p. 47).

[145] D. Esteves, D. Moussallem, C. B. Neto, T. Soru, R. Usbeck, M. Ackermann and J. Lehmann, "MEX Vocabulary: A Lightweight Interchange Format for Machine Learning Experiments",
Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS '15, ACM, 2015 169, ISBN: 978-1-4503-3462-4,
URL: http://doi.acm.org/10.1145/2814864.2814883 (cit. on p. 50).

[146] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne et al.,
The FAIR Guiding Principles for scientific data management and stewardship, Scientific data **3** (2016) (cit. on p. 50).

[147] A. J. Reddy, G. Rocha and D. Esteves, "DeFactoNLP: Fact Verification using Entity Recognition, TFIDF Vector Comparison and Decomposable Attention",
Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), Association for Computational Linguistics, 2018 132,
URL: http://aclweb.org/anthology/W18-5522 (cit. on p. 51).

[148] A. Rula, M. Palmonari, S. Rubbinaci, A. Ngonga, J. Lehmann and D. Esteves,
TISCO: Temporal Scoping of Facts,
Journal of Web Semantic JWS (Manuscript submitted for publication) (2018) (cit. on p. 51).

[149] D. Esteves, A. Rula, A. Reddy and J. Lehmann,
Towards Veracity Assessment in RDF Knowledge Bases – An Exploratory Analysis,
Journal of Data and Information Quality (JDIQ) (2018) (cit. on p. 51).

[150] R. Speck, D. Esteves, J. Lehmann and A.-C. Ngonga Ngomo,
"DeFacto - A Multilingual Fact Validation Interface",
14th International Semantic Web Conference (ISWC 2015), 11-15 October 2015, Bethlehem, Pennsylvania, USA (Semantic Web Challenge Proceedings),
ed. by S. Bechhofer and K. Kyzirakos,
Semantic Web Challenge, International Semantic Web Conference 2015, 2015,
URL: http://jens-lehmann.org/files/2015/swc_defacto.pdf (cit. on p. 51).

[151] J. Lehmann, D. Gerber, M. Morsey and A.-C. N. Ngomo, "Defacto-deep fact validation",
International Semantic Web Conference, Springer, 2012 312 (cit. on p. 51).

[152] J. Thorne, A. Vlachos, C. Christodoulopoulos and A. Mittal,
FEVER: a large-scale dataset for Fact Extraction and VERification,
arXiv preprint arXiv:1803.05355 (2018) (cit. on pp. 52, 76, 77).

[153] D. Chen, A. Fisch, J. Weston and A. Bordes,
Reading wikipedia to answer open-domain questions,
arXiv preprint arXiv:1704.00051 (2017) (cit. on pp. 52, 68).

[154] R. Baly, M. Mohtarami, J. R. Glass, L. Màrquez, A. Moschitti and P. Nakov, *Integrating Stance Detection and Fact Checking in a Unified Corpus*, CoRR **abs/1804.08012** (2018), arXiv: 1804.08012, URL: http://arxiv.org/abs/1804.08012 (cit. on p. 52).

[155] T. Mihaylova, P. Nakov, L. Màrquez, A. Barrón-Cedeño, M. Mohtarami, G. Karadzhov and J. R. Glass, *Fact Checking in Community Forums*, CoRR **abs/1803.03178** (2018), arXiv: 1803.03178, URL: http://arxiv.org/abs/1803.03178 (cit. on p. 52).

[156] D. Esteves, A. J. Reddy, P. Chawla and J. Lehmann, "Belittling the Source: Trustworthiness Indicators to Obfuscate Fake News on the Web", *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, Association for Computational Linguistics, 2018 50, URL: http://aclweb.org/anthology/W18-5508 (cit. on pp. 53, 54).

[157] A. Akbik, D. Blythe and R. Vollgraf, "Contextual string embeddings for sequence labeling", *Proceedings of the 27th International Conference on Computational Linguistics*, 2018 1638 (cit. on p. 53).

[158] A. Gatt and E. Krahmer, *Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation*, Journal of Artificial Intelligence Research **61** (2018) 65 (cit. on p. 53).

[159] S. Villata, G. Boella, D. M. Gabbay and L. Van Der Torre, "Arguing about the trustworthiness of the information sources", *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, Springer, 2011 74 (cit. on pp. 54, 77).

[160] R. M. Palau and M.-F. Moens, "Argumentation mining: the detection, classification and structure of arguments in text", *Proceedings of the 12th international conference on artificial intelligence and law*, ACM, 2009 98 (cit. on p. 54).

[161] A. Peldszus and M. Stede, *From argument diagrams to argumentation mining in texts: A survey*, International Journal of Cognitive Informatics and Natural Intelligence (IJCINI) **7** (2013) 1 (cit. on p. 54).

[162] L. Breiman, *Random Forests*, Mach. Learn. **45** (2001) 5, ISSN: 0885-6125, URL: https://doi.org/10.1023/A:1010933404324 (cit. on pp. 67, 70).

[163] J. R. Finkel, T. Grenager and C. Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling", *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, Association for Computational Linguistics, 2005 363, URL: https://doi.org/10.3115/1219840.1219885 (cit. on p. 68).

[164] V. I. Levenshtein, *Binary Codes Capable of Correcting Deletions, Insertions and Reversals*, Soviet Physics Doklady **10** (1966) 707 (cit. on p. 68).

[165] J. Thorne, A. Vlachos, C. Christodoulopoulos and A. Mittal, "FEVER: a Large-scale Dataset for Fact Extraction and VERification", *NAACL-HLT*, 2018 (cit. on p. 68).

[166] M. Sammons, V. Vydiswaran and D. Roth, "Recognizing Textual Entailment", *Multilingual Natural Language Applications: From Theory to Practice*, ed. by D. M. Bikel and I. Zitouni, Prentice Hall, 2012 209 (cit. on p. 68).

[167] S. R. Bowman, G. Angeli, C. Potts and C. D. Manning, "A large annotated corpus for learning natural language inference", *EMNLP*, The Association for Computational Linguistics, 2015 632 (cit. on p. 68).

[168] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep Contextualized Word Representations", *Proceedings of the 2018 Conference of NAACL: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, 2018 2227, URL: http://aclweb.org/anthology/N18-1202 (cit. on p. 68).

[169] A. Parikh, O. Täckström, D. Das and J. Uszkoreit, "A Decomposable Attention Model for Natural Language Inference", *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2016 2249, URL: http://www.aclweb.org/anthology/D16-1244 (cit. on p. 68).

[170] L. Pratt and B. Jennings, *A Survey of Transfer Between Connectionist Networks*, Connection Science **8** (1996) 163, eprint: https://doi.org/10.1080/095400996116866, URL: https://doi.org/10.1080/095400996116866 (cit. on p. 69).

[171] H. He and E. A. Garcia, *Learning from Imbalanced Data*, IEEE Trans. on Knowl. and Data Eng. **21** (2009) 1263, ISSN: 1041-4347, URL: https://doi.org/10.1109/TKDE.2008.239 (cit. on p. 69).

[172] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville and J. Pineau, "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.", *AAAI*, vol. 16, 2016 3776 (cit. on p. 75).

[173] M. Ballesteros, C. Dyer and N. A. Smith, *Improved Transition-Based Parsing by Modeling Characters instead of Words with LSTMs*, CoRR **abs/1508.00657** (2015), arXiv: 1508.00657, URL: http://arxiv.org/abs/1508.00657 (cit. on p. 75).

[174] P. Liu, S. Joty and H. Meng, "Fine-grained opinion mining with recurrent neural networks and word embeddings", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015 1433 (cit. on p. 75).

[175] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, *Enriching word vectors with subword information*, Transactions of the Association for Computational Linguistics **5** (2017) 135 (cit. on p. 75).

[176] T. Mikolov, K. Chen, G. Corrado and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, CoRR **abs/1301.3781** (2013), arXiv: 1301.3781, URL: http://arxiv.org/abs/1301.3781 (cit. on p. 75).

[177] J. Pennington, R. Socher and C. D. Manning,
"GloVe: Global Vectors for Word Representation",
Empirical Methods in Natural Language Processing (EMNLP), 2014 1532,
URL: http://www.aclweb.org/anthology/D14-1162 (cit. on p. 75).

[178] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov,
Bag of Tricks for Efficient Text Classification, CoRR **abs/1607.01759** (2016),
arXiv: 1607.01759, URL: http://arxiv.org/abs/1607.01759 (cit. on p. 75).

[179] A. P. Parikh, O. Täckström, D. Das and J. Uszkoreit,
"A Decomposable Attention Model for Natural Language Inference", *EMNLP*, 2016
(cit. on pp. 76, 77).

[180] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters,
M. Schmitz and L. S. Zettlemoyer,
"AllenNLP: A Deep Semantic Natural Language Processing Platform", 2017,
eprint: arXiv:1803.07640 (cit. on pp. 76–79).

[181] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean,
Distributed Representations of Words and Phrases and their Compositionality,
CoRR **abs/1310.4546** (2013), arXiv: 1310.4546,
URL: http://arxiv.org/abs/1310.4546 (cit. on p. 79).

[182] D. Esteves, D. Moussallem, C. B. Neto, T. Soru, R. Usbeck, M. Ackermann and
J. Lehmann, "MEX Vocabulary: A Lightweight Interchange Format for Machine
Learning Experiments", *11th International Conference on Semantic Systems
(SEMANTiCS 2015), 15-17 September 2015, Vienna, Austria*, 2015,
URL: http://svn.aksw.org/papers/2015/SEMANTICS_MEX/public.pdf
(cit. on pp. 81, 111).

[183] D. Esteves, P. N. Mendes, D. Moussallem, J. C. Duarte, A. Zaveri, J. Lehmann,
C. B. Neto, I. Costa and M. C. Cavalcanti,
"MEX Interfaces: Automating Machine Learning Metadata Generation",
*12th International Conference on Semantic Systems (SEMANTiCS 2016), 12-15
September 2016, Leipzig, Germany*, 2016,
URL: https://www.researchgate.net/publication/305143958_MEX_
InterfacesAutomating_Machine_Learning_Metadata_Generation (cit. on p. 81).

[184] D. Esteves, D. Moussallem, J. Lehmann, M. Claudia Cavalcanti Reis and J. Cesar
Duarte, "Interoperable Machine Learning Metadata using MEX",
International Semantic Web Conference 2015 (ISWC2015), Demos & Posters, 2015
(cit. on pp. 81, 84, 91).

[185] D. Esteves, *MEX Ontology - Documentation*, 2015,
URL: http://www.dne5.com/machine-learning/ontology/mexdocumentation.html
(visited on 02/04/2015) (cit. on pp. 81, 84).

[186] R. D. Peng, *Reproducible research in computational science*, Science **334** (2011) 1226
(cit. on p. 81).

[187] D. De, R. Carole and G. R. Stevens, *The design and realisation of the myexperiment
virtual research environment for social sharing of workflows*, (2008) (cit. on pp. 82, 83).

[188] T. R. Gruber, *A translation approach to portable ontology specifications*,
KNOWLEDGE ACQUISITION **5** (1993) 199 (cit. on p. 83).

[189] P. Panov, S. Džeroski and L. Soldatova, "OntoDM: An ontology of data mining", *Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference on*, IEEE, 2008 752 (cit. on pp. 83, 111).

[190] C. M. Keet, A. Lawrynowicz, C. d'Amato, A. Kalousis, P. Nguyen, R. Palma, R. Stevens and M. Hilario, *The Data Mining OPtimization Ontology*, J. Web Sem. **32** (2015) 43, URL: https://doi.org/10.1016/j.websem.2015.01.001 (cit. on pp. 83, 111).

[191] D. Esteves, D. Moussallem, C. B. Neto, T. Soru, R. Usbeck, M. Ackermann and J. Lehmann, "MEX vocabulary: a lightweight interchange format for machine learning experiments", *Proceedings of the 11th International Conference on Semantic Systems*, ACM, 2015 169 (cit. on p. 84).

[192] D. Esteves, D. Moussallem, T. Soru, C. Baron Neto, J. Lehmann, A.-C. Ngonga Ngomo and J. C. Duarte, "LOG4MEX: A Library to Export Machine Learning Experiments", *WI '17 - IEEE/WIC/ACM International Conference on Web Intelligence*, ACM, 2017 7, URL: http://jens-lehmann.org/files/2017/wi_log4mex.pdf (cit. on pp. 84, 85).

[193] D. Esteves, P. N. Mendes, D. Moussallem, J. C. Duarte, A. Zaveri and J. Lehmann, "MEX Interfaces: Automating Machine Learning Metadata Generation", *Proceedings of the 12th International Conference on Semantic Systems*, SEMANTiCS 2016, ACM, 2016 17, ISBN: 978-1-4503-4752-5, URL: http://doi.acm.org/10.1145/2993318.2993320 (cit. on pp. 84, 85).

[194] J. C. Duarte, M. C. R. Cavalcanti, I. de Souza Costa and D. Esteves, "An Interoperable Service for the Provenance of Machine Learning Experiments", *Proceedings of the International Conference on Web Intelligence*, WI '17, ACM, 2017 132, ISBN: 978-1-4503-4951-2, URL: http://doi.acm.org/10.1145/3106426.3106496 (cit. on pp. 84, 92).

[195] C. B. Neto, D. Esteves, T. Soru, D. Moussallem, A. Valdestilhas and E. Marx, "WASOTA: What Are the States Of The Art?", *SEMANTiCS (Posters, Demos, SuCCESS)*, 2016 (cit. on pp. 84, 93).

[196] D. Esteves, A. Lawrynowicz, P. Panov, L. Soldatova, T. Soru and J. Vanschoren, *ML Schema Core Specification*, draft report, http://www.w3.org/2016/10/mls/: W3C Machine Learning Schema Community Group, 2016 (cit. on pp. 84, 94).

[197] *W3C PROV-O Ontology*, http://www.w3.org/TR/prov-o/ (cit. on p. 85).

[198] D. Esteves, D. Moussallem, C. B. Neto, T. Soru, R. Usbeck, M. Ackermann and J. Lehmann, "MEX Vocabulary: A Lightweight Interchange Format for Machine Learning Experiments", *Proceedings of the 11th International Conference on Semantic Systems*, ACM, 2015 169 (cit. on pp. 85, 90).

[199] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva and H. T. Vo, "VisTrails: Visualization Meets Data Management", *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, ACM, 2006 745, ISBN: 1-59593-434-0, URL: http://doi.acm.org/10.1145/1142473.1142574 (cit. on p. 87).

[200] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, *The WEKA Data Mining Software: An Update*, SIGKDD Explor. Newsl. **11** (2009) 10, ISSN: 1931-0145, URL: http://doi.acm.org/10.1145/1656274.1656278 (cit. on p. 87).

[201] C. M. Keet, A. Ławrynowicz, C. d'Amato, A. Kalousis, P. Nguyen, R. Palma, R. Stevens and M. Hilario, *The Data Mining OPtimization Ontology*, Web Semantics: Science, Services and Agents on the World Wide Web (2015) 43, ISSN: 1570-8268, URL: http://www.sciencedirect.com/science/article/pii/S1570826815000025 (cit. on p. 91).

[202] A. Guazzelli, M. Zeller, W.-C. Lin and G. Williams, *PMML: An Open Standard for Sharing Models*, The R Journal **1** (2009) 60, URL: http://journal.r-project.org/archive/2009-1/RJournal_2009-1_Guazzelli+et+al.pdf (cit. on p. 91).

[203] R. R. Brinkman, M. Courtot, D. Derom, J. M. Fostel, Y. He, P. Lord, J. Malone, H. Parkinson, B. Peters, P. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, L. N. Soldatova, C. J. Stoeckert, J. A. Turner, J. Zheng and the OBI consortium, *Modeling biomedical experimental processes with OBI*, Journal of Biomedical Semantics **1** (2010) S7, ISSN: 2041-1480, URL: https://doi.org/10.1186/2041-1480-1-S1-S7 (cit. on p. 97).

[204] D. Garijo and Y. Gil, "Augmenting PROV with Plans in P-PLAN: Scientific Processes as Linked Data", *Second International Workshop on Linked Science: Tackling Big Data (LISC), held in conjunction with the International Semantic Web Conference (ISWC)*, 2012, URL: http://www.isi.edu/~gil/papers/garijo-gil-lisc12.pdf (cit. on p. 109).

[205] K. Belhajjame, J. Zhao, D. Garijo, M. Gamble, K. M. Hettne, R. Palma, E. Mina, Ó. Corcho, J. M. Gómez-Pérez, S. Bechhofer, G. Klyne and C. A. Goble, *Using a suite of ontologies for preserving workflow-centric research objects*, J. Web Sem. **32** (2015) 16, URL: http://dx.doi.org/10.1016/j.websem.2015.01.003 (cit. on p. 109).

[206] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan and J. V. den Bussche, *The Open Provenance Model Core Specification (V1.1)*, Future Gener. Comput. Syst. **27** (2011) 743, ISSN: 0167-739X, URL: http://dx.doi.org/10.1016/j.future.2010.07.005 (cit. on p. 109).

[207] P. Missier and L. Moreau, *PROV-DM: The PROV Data Model*, W3C Recommendation, http://www.w3.org/TR/2013/REC-prov-dm-20130430/: W3C, 2013 (cit. on p. 109).

[208] S. Sahoo, T. Lebo and D. McGuinness, *PROV-O: The PROV Ontology*, W3C Recommendation, http://www.w3.org/TR/2013/REC-prov-o-20130430/: W3C, 2013 (cit. on pp. 109, 111).

[209] H. H. Bauer, *Scientific literacy and the myth of the scientific method*, University of Illinois Press, 1994 (cit. on p. 111).

[210] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Nets", *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, MIT Press, 2014 2672,
URL: http://dl.acm.org/citation.cfm?id=2969033.2969125 (cit. on p. 118).

APPENDIX A

Complete List of Publications

The following is the complete list of publications peer-reviewed during the development of this Ph.D. thesis (07-2014 to 12-2018).

Journal Articles:

1. Gustavo Publio, Agnieska Ławrynowicz, Larisa Soldatova, Panče Panov, **Diego Esteves**, Joaquin Vanschoren and Tommaso Soru. *ML-Schema: An interchangeable format for description of machine learning experiments* In Journal of Web Semantics (JWS), 2019 - submitted;.

2. **Diego Esteves**, Anisa Rula, Aniketh Reddy, Jens Lehmann. *Toward Veracity Assessment in RDF Knowledge Bases: An Exploratory Analysis.* In Journal of Data and Information Quality (JDIQ), 2018.

3. Anisa Rula, Matteo Palmonari, Simone Rubbinaci, Axel Ngonga, Jens Lehmann, **Diego Esteves**. *TISCO: Temporal Scoping of Facts.* In Journal of Web Semantics (JWS), 2018.

4. Daniel Gerber, **Diego Esteves**, Jens Lehmann, Lorenz Bühmann, Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, René Speck. *DeFacto - Temporal and Multilingual Deep Fact Validation.* In Web Semantics: Science, Services and Agents on the World Wide Web (SWJ), 2015.

Conference Papers:

5. **Diego Esteves**, Asja Fischer, Piyush Chwala, Saad Khan, Jens Lehmann and Rafael Peres. *Beyond Lexical Features: Named Entity Recognition on Noisy Data through the Web.* In Proceedings of the 2019 Conference of the Association for Computational Linguistics (ACL'19) - submitted;

6. Piyush Chwala and **Diego Esteves** *Automating Fact-Checking: Why is it so difficult?.* In Proceedings of the 2019 Conference of the Association for Computational Linguistics (ACL'19) - submitted;

7. **Diego Esteves**, Aniketh Janardhan Reddy, Piyush Chawla and Jens Lehmann. *Belittling the Source: Trustworthiness Indicators to Obfuscate Fake News on the Web.* In Proceedings of Fact Extraction and VERification (FEVER) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

8. Aniketh Janardhan Reddy, Gil Rocha and **Diego Esteves**. *DeFactoNLP: Fact Verification using Entity Recognition, TFIDF Vector Comparison and Decomposable Attention.* In Proceedings of Fact Extraction and VERification (FEVER) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

9. Tommaso Soru, Edgard Marx, André Valdestilhas, **Diego Esteves**, Diego Moussallem and Gustavo Publio. *Neural Machine Translation for Query Construction and Composition.* In Proceedings of 2st Neural Abstract Machines & Program Induction Workshop (NAMPI) at NIPS co-located with the 35th International Conference on Machine Learning (ICML 2018), Stockholm Sweden, 2018.

10. Gustavo Correa Publio, **Diego Esteves**, Agnieszka Ławrynowicz, Panče Panov, Larisa Soldatova, Tommaso Soru, Joaquin Vanschoren and Hamid Zafar. *ML-Schema: Exposing the Semantics of Machine Learning with Schemas and Ontologies.* In Proceedings of the 2nd Reproducibility in Machine Learning Workshop (MLTRAIN@RML) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Stockholm Sweden, 2018.

11. Diego Moussallem, Mohamed Ahmed Sherif, **Diego Esteves**, Marcos Zampieri and Axel-Cyrille Ngonga Ngomo. *LIDIOMS: A Multilingual Linked Idioms Data Set.* In Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018);

12. Rafael Peres, **Diego Esteves** and Gaurav Maheshwari. *Bidirectional LSTM with a Context Input Window for Named Entity Recognition in Tweets.* In Proceedings of the 9th International Conference on Knowledge Capture (K-CAP 2017)

13. Julio Cesar Duarte, Maria Claudia Reis Cavalcanti, Igor de Souza Costa and **Diego Esteves**. *An Interoperable Service for the Provenance of Machine Learning Experiments.* In Proceedings of the International Conference on Web Intelligence (WI2017)

14. Agnieszka Lawrynowicz, **Diego Esteves**, Pance Panov, Tommaso Soru, Sašo Dzeroski and Joaquin Vanschoren. *An Algorithm, Implementation and Execution Ontology Design Pattern.* In Proceedings of Advances in Ontology Design and Patterns 32 (2017) - co-located with the 15th International Semantic Web Conference (ISWC 2016) in Kobe, Japan.

15. **Diego Esteves**, Pablo N. Mendes, Diego Moussallem, Julio Cesar Duarte, Amrapali Zaveri, Jens Lehmann and Ciro Baron Neto, Igor Costa and Maria Claudia Cavalcanti. *MEX Interfaces: Automating Machine Learning Metadata Generation.* In Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016).

16. **Diego Esteves**, Diego Moussallem, Ciro Baron Neto, Tommaso Soru, Ricardo Usbeck, Markus Ackermann and Jens Lehmann. *MEX Vocabulary: A Lightweight Interchange Format for Machine Learning Experiments.* In Proceedings of the 11th International Conference on Semantic Systems (SEMANTiCS 2015), 15-17 September 2015, Vienna, Austria.

17. **Diego Esteves**, Diego Moussallem, Tommaso Soru, Ciro Baron Neto, Jens Lehmann, Axel-Cyrille Ngonga Ngomo and Julio Cesar Duarte. *LOG4MEX: A Library to Export Machine Learning Experiments.* In Proceedings of the International Conference on Web Intelligence (WI'17), Leipzig, Germany.

18. **Diego Esteves**, Rafael Peres, Jens Lehmann and Giulio Napolitano. *Named Entity Recognition in Twitter using Images and Text.* In Proceedings of the 3rd International Workshop on Natural Language Processing for Informal Text (NLPIT 2017), Rome, Italy.

19. Ciro Baron Neto, Dimitris Kontokostas, Gustavo Publio, **Diego Esteves**, Amit Kirschenbaum and Sebastian Hellmann. *IDOL: Comprehensive & Complete LOD Insights.* In Proceedings of the 13th International Conference on Semantic Systems (SEMANTiCS 2017), 11-14 September 2017, Amsterdam, Holland.

Posters and Demos:

20. **Diego Esteves**. *Named Entity Recognition on Noisy Data using Images and Text.* In Proceedings of The 4th Workshop on Noisy User-generated Text (W-NUT) co-located with the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018), Brussels, Belgium, 2018.

21. René Speck, **Diego Esteves**, Jens Lehmann, Axel-Cyrille Ngonga Ngomo. *DeFacto - A Multilingual Fact Validation Interface.* In Proceedings of the 14th International Semantic Web Conference (ISWC 2015, Semantic Web Challenge)

22. **Diego Esteves**, Diego Moussallem, Ciro Baron Neto, Jens Lehmann, Maria Claudia Cavalcanti, Julio Cesar Duarte. *Interoperable Machine Learning Metadata using MEX.* In Proceedings of the 14th International Semantic Web Conference (ISWC 2015, Posters & Demos)

23. Edgard Marx, Tommaso Soru, **Diego Esteves**, Axel-Cyrille Ngonga Ngomo and Jens Lehmann. *An Open Question Answering Framework.* In Proceedings of the 14th International Semantic Web Conference (ISWC 2015, Posters & Demos)

24. Sandro A. Coelho, Diego Moussallem, Gustavo C. Publio, **Diego Esteves**. *TANKER: Distributed Architecture for Named Entity Recognition and Disambiguation.* In Proceedings of the Posters and Demos Track of the 13th International Conference on Semantic Systems - co-located with the 13th International Conference on Semantic Systems (SEMANTiCS 2017), Amsterdam, The Netherlands, September 11-14, 2017.

25. Tommaso Soru, Edgard Marx, Diego Moussallem, Gustavo Publio, André Valdestilhas,

Diego Esteves, Ciro Baron Neto. *SPARQL as a Foreign Language.* In Proceedings of the 13th International Conference on Semantic Systems (SEMANTiCS 2017), Amsterdam, The Netherlands, September 11-14, 2017.

26. Ciro Baron Neto, **Diego Esteves**, Tommaso Soru, Diego Moussallem, Andre Valdestilhas and Edgard Marx. *WASOTA: What Are the States of the Art?* In Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016), 12-15 September 2016, Leipzig, Germany (Posters & Demos)

The following is the complete list of preprints submitted during the development of this Ph.D. thesis.

Arxiv:

27. Tommaso Soru, Ruberto Stefano, Diego Moussallem, Edgard Marx, **Diego Esteves** and Axel-Cyrille Ngonga Ngomo. *Expeditious Generation of Knowledge Graph Embeddings.* In arXiv preprint arXiv:1711.01283, 2018.

28. Tommaso Soru, **Diego Esteves**, Edgard Marx, Axel-Cyrille Ngonga Ngomo. *Mandolin: A Knowledge Discovery Framework for the Web of Data.* In arXiv preprint arXiv:1711.01283, 2017.

29. Ciro Baron Neto, Sebastian Hellmann and **Diego Esteves**. *Towards an Approximated Accurate Measure of Links on the Data Web and a Scalable Generation Method for Dynamic LOD Cloud Diagrams.* In arXiv preprint arXiv:1711.01283, 2015.

Technical Reports:

30. **Diego Esteves**, Agnieszka Lawrynowicz, Pance Panov, Larisa Soldatova, Tommaso Soru and Joaquin Vanschoren. *ML-Schema Core Specification.* In Technical report, World Wide Web Consortium (W3C), 2016.

List of Figures

List of Tables