Bielefeld University
Faculty of Technology

Neural Approaches to Relational Aspect-Based Sentiment Analysis

Exploring generalizations across words and languages

Soufian Jebbara

Submitted in partial fulfillment of the requirements for the degree of
Doctor rerum naturalium (Dr. rer. nat.)
January 2020

Reviewers:

Prof. Dr. Philipp Cimiano, Bielefeld University, Germany

Prof. Dr. Hinrich Schütze, Ludwig-Maximilians-Universität München, Germany

Prof. Dr. Barbara Hammer, Bielefeld University, Germany

Printed on non-aging paper according to ISO 9706.

Abstract

Everyday, vast amounts of unstructured, textual data are shared online in digital form. Websites such as forums, social media sites, review sites, blogs, and comment sections offer platforms to express and discuss opinions and experiences. Understanding the opinions in these resources is valuable for e.g. businesses to support market research and customer service but also individuals, who can benefit from the experiences and expertise of others.

In this thesis, we approach the topic of opinion extraction and classification with neural network models. We regard this area of sentiment analysis as a relation extraction problem in which the sentiment of some opinion holder towards a certain aspect of a product, theme, or event needs to be extracted. In accordance with this framework, our main contributions are the following:

i) We propose a full system addressing all subtasks of relational sentiment analysis.

ii) We investigate how semantic web resources can be leveraged in a neural-network-based model for the extraction of opinion targets and the classification of sentiment labels. Specifically, we experiment with enhancing pre-trained word embeddings using the lexical resource WordNet. Furthermore, we enrich a purely text-based model with SenticNet concepts and observe an improvement for sentiment classification.

iii) We examine how opinion targets can be automatically identified in noisy texts. Customer reviews, for instance, are prone to contain misspelled words and are difficult to process due to their domain-specific language. We integrate information about the character structure of a word into a sequence labeling system using character-level word embeddings and show their positive impact on the system's performance. We reveal encoded character patterns of the learned embeddings and give a nuanced view of the obtained performance differences.

iv) Opinion target extraction usually relies on supervised learning approaches. We address the lack of available annotated data for specific languages by proposing a zero-shot cross-lingual approach for the extraction of opinion target expressions. We leverage multilingual word embeddings that share a common vector space across various languages and incorporate these into a convolutional neural network architecture. Our experiments with 5 languages give promising results: We can successfully train a model on annotated data of a source language and perform accurate prediction on a target language without ever using any annotated samples in that target language.

Acknowledgements

I would like to thank my supervisor Prof. Dr. Philipp Cimiano for giving me the opportunity to work in an exceptional research group. Philipp, Hendrik ter Horst, Sherzod Hakimov, Maximilian Panzner, Basil Ell, Matthias Hartung and all other members of the Semantic Computing Group supported and inspired me throughout the years with fruitful collaborations, feedback and discussions.

I am thankful to Prof. Dr. Hinrich Schütze and Prof. Dr. Barbara Hammer for accepting to review this thesis.

My special thanks goes to my family, friends, and my lovely wife Natascha for the unlimited support and encouragement during the last few years and without whom I would not have reached this point in my life.

A large part of this thesis was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

Lastly, I would like to express my sincere gratitude to the company Semalytix that enabled me to finish this thesis in an awesome working environment among friends.

"If you give a man an answer, all he gains is a little fact. But give him a question and he'll look for his own answers."

— Patrick Rothfuss, The Wise Man's Fear

Contents

Chapter 1

Introduction

Chapter Overview *This chapter provides an introduction into Aspect-based Sentiment Analysis which constitutes a central research topic of this thesis. We motivate the importance of this particular branch of sentiment analysis and demonstrate its challenges with tangible real-world examples. These lead us to our research questions and consequently to the contributions of this thesis.*

1.1 Motivation

The technological advances of information technologies in recent decades opened up new communication ways over the internet. These new formats of communication allow vast amounts of people to exchange ideas, experiences, opinions, and more about every conceivable topic. A large number of websites and platforms support and rely on this exchange between people. Nowadays, anyone can share a story in online forums, talk to friends and strangers on social media sites, write an opinion piece, news article, commentary, or review. The quantity of available information is plainly staggering and making use of it to its fullest extent is practically impossible with manual effort alone. In fact, the travel website TripAdvisor alone reached an amount of roughly 730 million shared reviews in 2018[1]. For the business-review website Yelp, the number of gathered reviews reached 192 million in 2019[2].

Being able to automatically understand and summarize the masses of information is clearly valuable for many interest groups. Businesses can profit from customer feedback by understanding the customer needs more clearly. Recognizing genuine criticism about e.g. a product allows a business to address these issues quickly and in turn increase customer satisfaction. Furthermore, the diversity of

[1]Retrieved 7 November 2019: `https://www.statista.com/topics/3443/tripadvisor/`
[2]Retrieved 7 November 2019: `https://www.yelp-press.com/company/fast-facts/`

the available information can aid in market research to identify unmet needs that are otherwise hard to uncover. While the commercial interest of understanding opinions is obvious, consumers benefit from the experiences and opinions of others as well. The emergence of review sites such as Yelp, TripAdvisor, IMDb, Glassdoor, Goodreads, Jameda, or StreetAdvisor underline the important role that reviews play in the decision making of consumers. These sites allow to share experiences relating to a broad spectrum of topics. Consumers often rely on such reviews when forming a decision about a product. In many cases, the decision making process is influenced by only a handful of reviews [Askalidis and Malthouse, 2016] with only few people taking the time to browse beyond ten reviews[3]. Lackermair et al. [2013] confirm that customer reviews are an important source of information for customers. However, they find that the unstructured nature of textual reviews reduces their perceived helpfulness. This is due to the fact that comparing these reviews is a laborious process for the customer. This clearly shows the need for a concise representation or summarization of large amounts of customer reviews in order to inform about the expressed opinions.

While opinions can be expressed through various modalities, be it visually or auditory [Pérez-Rosas et al., 2013; Poria et al., 2016b], a large part of the openly shared experiences are in textual form. These texts are often semi- or unstructured, that is, they consist of free text and are sometimes supplemented by other information such as user ratings. Given the example of user reviews, the provided ratings are often coarse-grained in that they represent an overall rating of the product, service, or experience. For instance, the e-commerce platform Amazon provides a 5-star rating scale to summarize the entirety of the customer's satisfaction with a product. However, the expressed opinions in reviews are often more nuanced and touch upon several aspects of an experience which are not accurately represented by the overall rating. Many potential applications for customers and businesses require a more fine-grained view on the given information beyond the document-level. This need is addressed by the growing research field of Aspect-based Sentiment Analysis.

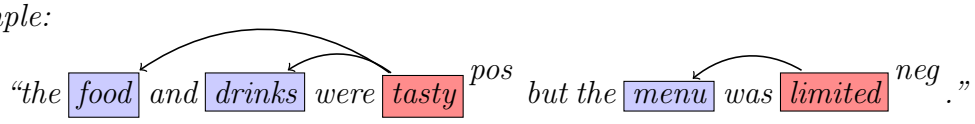
1.1.1 Aspect-Based Sentiment Analysis

Aspect-based Sentiment Analysis (ABSA) is concerned with the detection of opinions, evaluations, or emotions towards entities of interest. The range of possible entities is diverse and highly dependent on the considered domain. Among others, targeted entities can be political topics, products, services, places, events, or parts thereof. An entity, can be associated with different aspects i.e. a collection of parts

[3]Retrieved 7 November 2019: `https://www.vendasta.com/blog/50-stats-you-need-to-know-about-online-reviews`

or attributes of the entity. As an example, an opinion holder who is reviewing a restaurant might not refer to it as a whole but rather to individual aspects that were particularly important to his experience:

Example:



In the above example, an opinion holder expresses three distinct opinions. A positive sentiment is expressed towards the two aspect phrases *food* and *drinks*, indicated by the opinion phrase *tasty*. A third, negative opinion is expressed through the term *limited* and targets the *menu*. Breaking down the review into individual opinions with a clear target is of particular value especially when the opinion holder has conflicting opinions about different aspects that cannot simply be subsumed by an overall sentiment label.

The importance of automatically finding fine-grained opinions targeted at individual aspects of an entity is apparent. ABSA makes it possible to group, classify, and sort experiences automatically by various domain-specific aspects depending on the particular information need. As an example, a person looking for an apartment might be interested in neighborhoods that are well connected to the public transportation system while nearby nightlife and dining opportunities are irrelevant. Yet, the opposite might be true for others. The possibility to filter reviews by what is relevant to an individual greatly reduces the effort of the exploration process. For businesses, the analysis of reviews and customer feedback in general allows to identify problematic aspects of a product or service. ABSA offers a way to alleviate manual effort in analyzing this feedback. It helps to discover previously unknown themes in the expressed opinions and allows to access them in a structured manner. We summarize that ABSA is a valuable means for addressing the strong demand of structuring large collections of freely expressed opinions and making them semantically searchable. Before we consider typical challenges that arise for ABSA, we briefly introduce the relevant terminology that is used in this thesis.

1.2 Terminology

Throughout this thesis, we support the view that opinions are often expressed in complex ways that elude an accurate summarization by a simple rating scale such as a *positive* or *negative* sentiment label. Rather, opinions are composed of several contributing parts. In the following, we describe these parts as we use them in this thesis:

Aspect An *aspect* usually refers to a part of a product, service, event on a conceptual level. It can be a concrete part of a product e.g. the *display* of a phone, but also more abstract things such as the *ambience* of a restaurant.

Aspect Phrase In a text, the aspect *ambience* can be referred to with different *aspect phrases*, e.g. "*decor*", "*interior design*", or "*atmosphere*". These are the actual terms that are used to speak of a particular aspect.

Opinion Holder The *opinion holder* is the person that expresses an opinion. As we are focusing on customer reviews in this thesis, the opinion holder is implicitly the writer of the review. Therefore, we usually assume the opinion holder to be already identified and irrelevant to our analysis. We omit it from further discussions.

Opinion We refer to an *opinion* as a complex structure that usually defines i) what the target of the opinion is e.g. the "*food*", ii) what its sentiment label is (i.e. *positive*, *negative*, etc.), and iii) what the reason for the sentiment is e.g. "*tasty*".

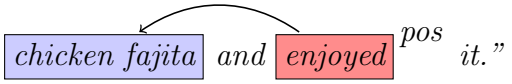
Sentiment Label An opinion usually expresses a positive or negative attitude towards an aspect and the *sentiment label* (or *sentiment polarity*) is a discrete representation of this. In our examples, we visualize this as a superscript.

Opinion Phrase The phrases in a text that explicitly carry subjective information are referred to as *opinion phrases*, also called *opinion expressions* or *subjective expressions*. In many cases, these play an important role in identifying the polarity of an opinion as they may convey its reason. Where relevant, we mark these with red boxes in our examples.

Opinion Target Expression An *opinion target expression* is an aspect phrase towards which an opinion holder expresses an opinion. It is thus the target of the opinion. For clarification, consider the following examples:

Example:

"I ordered the chicken fajita."



In both sentences, the *chicken fajita* is an aspect phrase that is potentially rateable. However, only in the second example is it also the target of an opinion

and thus an opinion target expression. We mark these with blue boxes in our examples.

Opinion Target Relation In subsequent parts of this thesis, we are concerned with sentiment analysis as a relation extraction problem. In short, we advocate to subdivide ABSA into smaller subtasks, one of which is the identification of *opinion target relations*. Such a relation connects an opinion phrase to an opinion target expression and indicates that the expressed sentiment refers to the targeted aspect. Relations are visualized as arrows connecting the respective phrases.

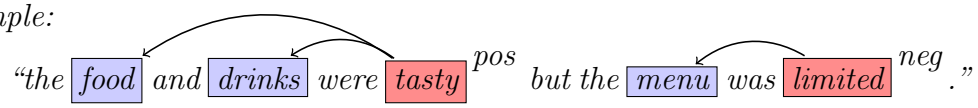
Aspect Category Some formalizations of ABSA assign every aspect to one of several predefined categories which provide a further abstraction from the textual expressions. The SemEval 2016 workshop [Pontiki et al., 2015], defines aspect categories as pairs of *Entities* (e.g. *Laptop*, *Keyboard*, *Customer Support*, *Restaurant*, *Food*) and *Attributes* (e.g. *Usability*, *Quality*, *Price*). As the classification of aspect categories is out of scope of this thesis, we merely mention these here for the sake of completeness.

1.3 Challenges

Automatically extracting complex opinions from customer reviews is decidedly valuable given the wealth of available information. It is, however, challenging for several reasons. At the core of all challenges resides the fact that natural language is complicated and evolving and has, in general, a high variability. This affects Natural Language Processing (NLP) in general and fuels the continuous growth of NLP research. For ABSA, we highlight the following challenges that are of particular importance to this thesis:

Multiple Opinions As we already saw before, an opinion holder can state several opinions with conflicting polarities for the same entity, such as a restaurant, in a single sentence:

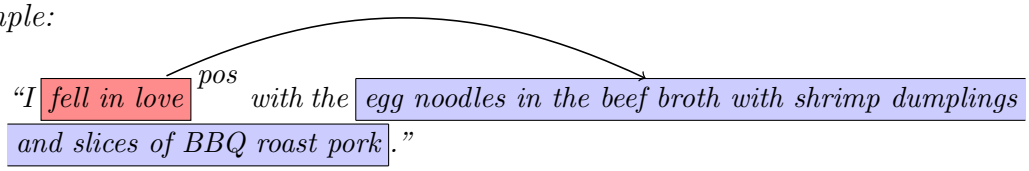
Example:



Document-level classification approaches are unsuited for this problem as they lack the required granularity. Among these, Bag-of-Words approaches are particularly unsuited for inferring the sentiment towards an aspect as they are oblivious to the sentence structure. To address this issue, further steps need to be taken to enable aspect-specific inference (see Chapters 3 and 4).

Variation of Aspects For most domains and applications, an ontology is not available so that the exact set of potential target entities and their aspects is usually unknown. This is complicated even further since an aspect such as *ambience* can be referenced in a multitude of ways, including, but not limited to the aspect phrases "*decor*", "*vibe*", and "*atmosphere*". Furthermore, the evolving nature of the web leads to the emergence of new entities and aspects as new services or products become available or events take place. Another form of variation is expressed through very specific, and in some cases long aspect phrases:

Example:



This examples makes clear, that it is infeasible to collect a finite set aspect phrases as they can be strongly compositional. Rather, it is necessary to infer targeted aspect phrases from the text itself.

Ambiguities For ABSA, ambiguous words, phrases, and sentences complicate the extraction of opinions. Adjectives like "*cold*" or "*small*" often carry an indication of a sentiment polarity, however, this can only be accurately determined in conjunction with the modified aspect. As an example, a "*cold beer*" is generally considered a good thing whereas "*cold food*" tends to be perceived negatively.

Implicit Opinions Moreover, it is not unusual for an opinion to be expressed in an implicit way. For instance, an opinion phrase which explicitly indicates the polarity of an opinion can be omitted while the expressed sentiment remains clearly discernable for a human reader:

Example:

> *"Please take my advice, go and try this place."*

> *"There is something about their atmosphere that makes me come back nearly every week."*

Other times, the opinion target is unspecified as well, such that the only part of an opinion that can be identified is an overall sentiment:

Example:

> *"And I hate to say this but I doubt I'll ever go back."*

Background Knowledge Going further, some statements are only clear in a larger context or require a fair amount of background knowledge to understand them correctly:

Example:

> *"Well, I guess maybe it might be beneficial for the management to pay a visit to Tutta Bella in Columbia City for some tips."*

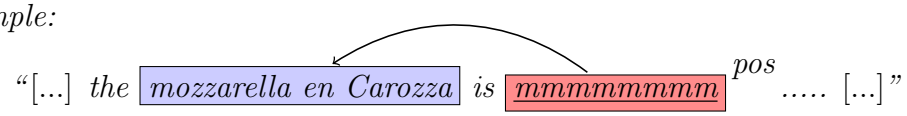
Domain-Specific Language While the manner in which an opinion holder expresses an opinion can pose challenges for automatic extraction systems, the same is true for the source domain of the texts. Each domain has its peculiarities that affect how opinions are expressed and what constitutes a valid target aspect. Target aspects are heavily domain dependent and make use of terms and phrases that are specific to that domain. The following excerpt from a laptop review illustrates this:

Example:

> *"[...] and its radeon 5850 would have DDR5 instead of DDR3."*

Besides the target aspect, opinion expressions and other sentiment indicators vary from domain to domain:

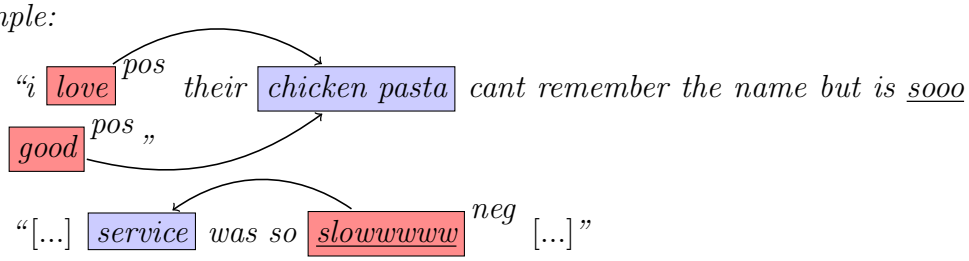
Example:



The onomatopoetic expression "*mmmmmmmm*", while understandable in the restaurant domain, would usually not be used as such in e.g. a laptop review.

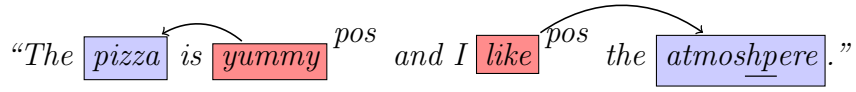
Writing Style Customer reviews are particularly challenging as they often feature an informal writing style. Among common phenomena are elongated words that emphasize a sentiment:

Example:



Besides these deliberate peculiarities, customer reviews frequently contain spelling errors:

Example:



These characteristics impede many traditional, token-based machine learning approaches and require extensive preprocessing and normalization.

Cost of Annotation The aforementioned challenges are mostly rooted in the complexity of natural language itself. Besides these challenges, there are more technical hurdles as well. Most existing approaches for ABSA rely on supervised machine learning methods. For these, manually annotated training data is needed to optimize a model for the specific task. The annotation process itself is laborious due to the intricate structure of opinions. And as the performance of these approaches often depends on the quality and quantity of these annotations, the generation of such a training set can be quite costly. This issue has not been addressed to a satisfactory extent by previous research efforts.

1.4 Research Questions

The key research questions of this thesis concern the automatic extraction of opinions from natural language texts. We are concerned with four major questions that correspond to the core chapters of this thesis. We approach these research questions through more nuanced sub-questions which each focus on different facets. To give the reader an overview of the focus of this thesis, we summarize each research question briefly:

RQ1 *How can fine-grained sentiment analysis be addressed in its complexity?*

Sentiment analysis is traditionally addressed in a strongly simplified way that cannot model the intricacies of naturally expressed opinions. We advocate the framework of *relational sentiment analysis* and propose solutions for all its subtasks.

RQ1.1 *How can relevant phrases be identified?*

Key tasks within relational sentiment analysis are the identification of aspects and opinion phrases. These phrases are classified and linked in later steps to represent expressed opinions. We offer solutions capable of the automatic extraction of these phrases.

RQ1.2 *What is the impact of domain-specific word embeddings on the task?*

A central method to represent texts is through the use of word embeddings. These are usually pretrained on a large unlabeled dataset and fine

tuned on specific downstream tasks. We experiment with domain-specific word embeddings and assess their contribution to the extraction of target phrases.

RQ1.3 *How can dependencies between aspect and opinion phrases be leveraged?*

Aspect and opinion phrases often co-occur as they are semantically linked. We show that jointly extracting these phrases is beneficial for aspects and opinion phrases alike.

RQ1.4 *How can we infer the sentiment pertaining to a particular expression in a document?*

It is not unusual that a single document contains multiple opinion expressions with contrasting polarities. Classifying the subjective content of a phrase is difficult as it heavily depends on its context. We propose a novel model capable of opinion-phrase specific sentiment classification using learned distance embeddings.

RQ1.5 *How can we identify corresponding targets for an opinion phrase?*

Opinion phrases usually refer to one or more explicit target phrases. To determine the opinion towards a specific target aspect, we perform relation extraction between identified aspect and opinion phrases and surpass prior approaches.

RQ2 *How can external, structured knowledge be incorporated into a model for ABSA?*

Structured knowledge bases have the potential to support machine learning models for ABSA. We evaluate two approaches to leverage this knowledge.

RQ2.1 *What is the benefit of lexical knowledge?*

Word embeddings are often computed in a purely data-driven fashion and lack explicit lexical knowledge. We experiment with existing techniques to introduce such external knowledge into precomputed word embeddings.

RQ2.2 *What improvements can be expected when including semantic knowledge?*

While word embeddings can encode semantic information to some extent, this knowledge is not explicit. Learning this from a training corpus might not always be possible. We measure the impact of explicit, concept-level information on the task of ABSA.

RQ3 *How can we alleviate the difficulty of opinion target extraction for noisy textual data?*

ABSA is often performed on user-generated texts. As such, the documents are noisy and feature an informal writing style that complicate an automated analysis. We propose a solution to mitigate these effects.

RQ3.1 *How can a model learn relevant morphological information?*

Our solution leverages character-level word embeddings that we optimized for the extraction of opinion target expressions. The learned embeddings show a desirable clustering according to word suffixes.

RQ3.2 *Which challenges are alleviated with subword information?*

We hypothesize that the learned embeddings help to disambiguate Out-of-Vocabulary words and multi-word expressions. We evaluate these ideas and show an improvement for multi-word expressions.

RQ4 *How can we reduce the annotation effort for the creation of training data for under-resourced languages?*

Most of the proposed approaches throughout this thesis make use of very few task-specific features and instead rely on annotated training datasets for supervised learning. These annotations can be costly to produce, especially so for multiple domains and languages. We explore an approach to mitigate the reliance on annotated data for a specific language.

RQ4.1 *To what degree is a multilingual model capable of performing OTE extraction for unseen languages?*

Being able to perform Opinion Target Expression (OTE) extraction without any target language annotations is very valuable. We propose an approach capable of zero-shot prediction for unseen languages and compare it across language pairs.

RQ4.2 *What is the benefit of training a model on more than one source language?*

Our approach can leverage multiple source languages at once and thus make use of more available training data. We show in how far such a model improves over using a single language.

RQ4.3 *How much annotation effort can be saved by harnessing cross-lingual learning?*

Having training samples for the target language is still a valuable source of information for opinion target extraction. We show how much annotation effort can be saved by our approach.

RQ4.4 *How big is the impact of the used alignment method on the OTE extraction performance?*

The alignment of word embeddings across languages plays a central role in the proposed transfer learning. We compare two methods for obtaining such embeddings.

1.5 Contributions and Structure of the Thesis

This thesis comprises 7 chapters. Chapters 1 and 2 lay the groundwork of the thesis and motivate the overall goal and research questions. The core chapters (Chapter 3 - Chapter 6) follow a similar, overall structure and i) start with an introduction to a particular research problem, including motivations, related work, research questions, and our contributions, ii) proceed with a description of our approaches, iii) evaluate the approaches by carrying out experiments, iv) discuss the results, and finally v) conclude the chapter with a brief summary and an outlook on future work. The thesis finally concludes with a summary of the main findings and general future research directions in Chapter 7. The remainder of this thesis is organized as follows:

Chapter 2 *Background*

This chapter lays the groundwork for this thesis as it introduces the fundamentals of natural language processing with neural networks. We give a brief introduction into artificial neural networks and its modern variants and introduce word embeddings as a method for representing textual data. We briefly describe span representations as a means to sequence tagging. The chapter is concluded with the definition of frequently used evaluation metrics.

Chapter 3 *Relational Sentiment Analysis*

We introduce the framework of *relational sentiment analysis* in an effort to adequately model the complexity of sentiment analysis in a modular way. Each such module is implemented and evaluated. In this chapter, we address the following research questions:

RQ1 *How can fine-grained sentiment analysis be addressed in its complexity?*

RQ1.1 *How can relevant phrases be identified?*

RQ1.2 *What is the impact of domain-specific word embeddings on the task?*

RQ1.3 *How can dependencies between aspect and opinion phrases be leveraged?*

RQ1.4 *How can we infer the sentiment pertaining to a particular expression in a document?*

RQ1.5 *How can we identify corresponding targets for an opinion phrase?*

Chapter 4 *Aspect-Based Sentiment Analysis and Structured Resources*

In this chapter, we follow a simplified formulation of the problem of sentiment analysis that reduces it to two subtasks: i) the extraction of opinion target phrases, and ii) the classification of sentiment labels for specific target phrases. We provide a focused look on the benefits of external, structured resources and evaluate their contribution for the task. We address the following research questions:

RQ2 *How can external, structured knowledge be incorporated into a model for ABSA?*

RQ2.1 *What is the benefit of lexical knowledge?*

RQ2.2 *What improvements can be expected when including semantic knowledge?*

Chapter 5 *Subword Information for Opinion Target Extraction*

This chapter concerns the extraction of opinion targets in particular as it poses an important early step in our sentiment analysis framework. We focus on the effects of noisy, user-generated data and present an approach to mitigate these issues. We address the following research questions:

RQ3 *How can we alleviate the difficulty of opinion target extraction for noisy textual data?*

RQ3.1 *How can a model learn relevant morphological information?*

RQ3.2 *Which challenges are alleviated with subword information?*

Chapter 6 *Zero-Shot Cross-Lingual Opinion Target Extraction*

This chapter addresses opinion target extraction with regard to the lack of available annotated data for under-resourced languages. We reduce the gap between high-resource languages (e.g. English) and low-resource languages (e.g. Russian) for this task by harnessing NLP techniques for cross-lingual learning. We address the following research questions:

RQ4 *How can we reduce the annotation effort for the creation of training data for under-resourced languages?*

RQ4.1 *To what degree is a multilingual model capable of performing OTE extraction for unseen languages?*

RQ4.2 *What is the benefit of training a model on more than one source language?*

RQ4.3 *How much annotation effort can be saved by harnessing cross-lingual learning?*

RQ4.4 *How big is the impact of the used alignment method on the OTE extraction performance?*

Chapter 7 *Conclusion*

In this final chapter, we summarize our results and discuss the main findings with respect to the posed research questions. We conclude this thesis with open questions and an outlook on potential research opportunities.

1.6 Publications

The main material of this thesis has been published as the following peer-reviewed publications (listed in reverse-chronological order):

[JC19] S. Jebbara and P. Cimiano. Zero-Shot Cross-Lingual Opinion Target Extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2486–2495, 2019.

[JC17] S. Jebbara and P. Cimiano. Improving Opinion-Target Extraction with Character-Level Word Embeddings. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 159–167, 2017.

[JC16b] S. Jebbara and P. Cimiano. Aspect-Based Sentiment Analysis Using a Two-Step Neural Network Architecture. In *Semantic Web Challenges. Third SemWebEval Challenge at ESWC 2016. Revised Selected Papers*, pages 153–170, 2016.

[JC16a] S. Jebbara and P. Cimiano. Aspect-Based Relational Sentiment Analysis Using a Stacked Neural Network Architecture. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 1123–1131, 2016.

Further publications that were developed and published during the research for this thesis include:

[JBCC19] S. Jebbara, V. Basile, E. Cabrio, and P. Cimiano. Extracting common sense knowledge via triple ranking using supervised and unsupervised distributional models. *Semantic Web*, 10(1): 139–158, 2019.

[HJC19] S. Hakimov, S. Jebbara and P. Cimiano. Evaluating Architectural Choices for Deep Learning Approaches for Question Answering over Knowledge Bases. In *13th IEEE International Conference on Semantic Computing (ICSC)*, pages 110–113, 2019.

[HJC17] S. Hakimov, S. Jebbara and P. Cimiano. AMUSE: Multilingual Semantic Parsing for Question Answering over Linked Data. In *The Semantic Web - ISWC 2017. 16th International Semantic Web Conference*, pages 329–346, 2017.

[HKJC17] M. Hartung, F. Kaupmann, S. Jebbara, and P. Cimiano. Learning Compositionality Functions on Word Embeddings for Modelling Attribute Meaning in Adjective-Noun Phrases. In *Proceedings of the 15th Meeting of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 54-64, 2017.

[HTJHC16] S. Hakimov, H. ter Horst, S. Jebbara, M. Hartung, and P. Cimiano. Combining Textual and Graph-based Features for Named Entity Disambiguation Using Undirected Probabilistic Graphical Models.

In *Knowledge Engineering and Knowledge Management (EKAW)*. pages 288-302, 2016.

[BJCC19] V. Basile, S. Jebbara, E. Cabrio, and P. Cimiano Populating a Knowledge Base with Object-Location Relations using Distributional Semantics In *Proceedings of 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, pages 34-50, 2016.

Throughout this thesis, I use the personal pronoun *we* in acknowledgment of my co-authors.

Chapter 2

Background

Chapter Overview *This chapter introduces the foundations of this thesis. We give a brief introduction into artificial neural networks in general and some of their underlying principles. We present modern concepts of neural computation that are relevant for this thesis and show how these can be applied to NLP. In particular, we introduce the concept of word embeddings that forms an important part of many modern NLP applications. We then introduce the concept of sequence labeling and related ideas. Finally, the chapter is concluded with the definition of relevant evaluation metrics.*

2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a type of computational processing systems that are loosely motivated by biological neural networks [McCulloch and Pitts, 1943]. Similar to their biological counterparts, ANNs are a capable of learning patterns and functions from input-output pairs and are thus a popular machine learning approach. As ANNs play an important role in this thesis, we give a brief introduction into the inner workings of ANNs and the core concepts surrounding their usage.

An ANN consists of a number of interconnected *neurons*, also called *units*. The connections between the neurons are weighted and usually one-way connections. Each neuron in the network acts as a small computation unit that receives input signals and, dependent on the connection weights, computes a corresponding output signal. Traditionally, the computation of a single neuron can be separated into the *pre-activation* z, i.e. the weighted sum of its inputs, and the *activation* a, i.e. the non-linear transformation of the pre-activation. Using a succinct vector

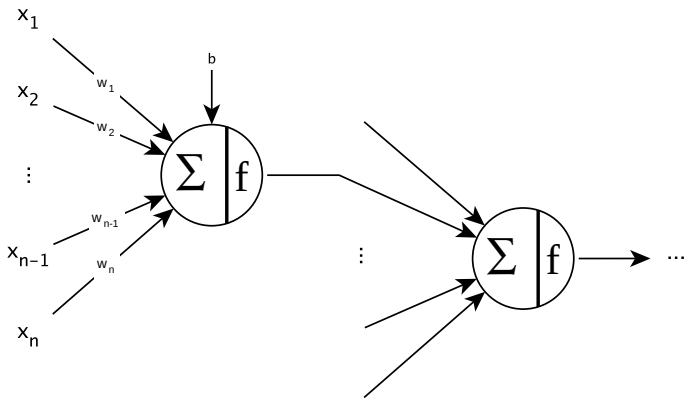


Figure 2.1: A schematic visualization of a neural network. A neuron (left) computes an activation based on an input vector. Its output is passed to a subsequent neuron (right) as an input value.

notation, this can be written as:

$$z = w^\top x + b \tag{2.1}$$

$$a = f(z) \tag{2.2}$$

where x is the input for this neuron, w is the vector of connection weights, b a bias weight that shifts the pre-activation independently from the received input, and f a non-linear activation function. The connection weights w together with the bias b form the weights of the neuron. The neurons of a network are connected such that the output of one neuron acts as an input for another neuron. The network thus forms a complex function, parameterized by the weights. In fact, it has been shown that sufficiently complex networks with adequate *topology*, i.e. arrangement of neurons, are universal function approximators [Hornik, 1991]. Figure 2.1 shows a schematic visualization of interconnected neurons.

Optimization The connection weights of a neural network substantially influence the network's internal computations and with that, its outputs. For a network to approximate a desired behavior, its weights need to be configured accordingly. The optimization process is called the *training* of the network. It is characterized by a *forward pass* and a *backward pass*. At the beginning of the optimization process, the network weights are set randomly. In the forward pass, we present inputs to the network and successively compute outputs for all following neurons, until we reach the output neurons. Given the current network weights, we thus obtain a

predicted output vector. We compare the predicted output to an expected output and compute the *loss* (also called the *error*) with a *loss function*.

Since our goal is to reduce the error of the network, and thus obtain more accurate predictions, we adapt the network weights accordingly. A popular approach to update the weights is a method called *gradient descent*. Gradient descent iteratively updates the weights of a network by following the gradient of the loss function such that the computed error is reduced. The efficient calculation of gradients is performed in a backwards pass through the network using a procedure called *backpropagation* [Werbos, 1974] that leverages the chain-rule for differentiating composed functions. In practice, the weight updates are averaged over mini-batches of data samples to obtain a more stable gradient and to improve the overall training speed [Dekel et al., 2012]. Additionally, the training procedure may require several iterations over the training dataset, called *epochs*. The training is stopped when the updates of the weights converge or the error on a validation set stops decreasing. The latter is a regularization technique called *early stopping* [Caruana et al., 2000]. Popular variations of gradient descent are RMSProp [Tieleman and Hinton, 2012] and Adam [Kingma and Ba, 2014] that track the evolution of weights over time and adjust their update rates accordingly. Especially Adam is established as a robust optimization strategy for neural networks.

Activation Functions The activation functions of a neural network greatly influence its behavior during the training and prediction process. The *sigmoid* function has been a long-standing candidate for the activation function of hidden and output units:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

The sigmoid function bounds the neurons activation between 0 and 1 which is particular desirable for output neurons where the activation can be interpreted as a probability. The shape of the activation function is depicted in Figure 2.2a.

The *Hyperbolic Tangent (tanh)* has a similar shape as the sigmoid function but ranges from -1 to 1:

$$tanh(x) = 2\sigma(2x) - 1 \tag{2.4}$$

Its shape is depicted in Figure 2.2b.

Both the tanh and sigmoid function suffer from vanishing gradients during the training procedure of deep networks [Hochreiter, 1998] as their gradients approach 0 towards either ends. The *Rectified Linear Unit (ReLU)* [Nair and Hinton, 2010] alleviates this problem with its piece-wise linear nature:

$$ReLU(x) = \begin{cases} 0 & x \leq 0 \\ x & x > x \end{cases} \tag{2.5}$$

The ReLU has been a key in training deeper neural networks as it has a constant gradient of 1 for $x > 0$ and thus allows the backpropagation of errors to continue through several neurons. It is one of the most widely used activation functions in modern neural networks. The ReLU is depicted in Figure 2.2c.

However, as the ReLU has a gradient of 0 for $x \leq 0$, neurons can get stuck after a large gradient update that pushes their activation below 0 for all x. Thus, these neurons cease to activate again. The *Exponential Linear Unit (ELU)* [Clevert et al., 2016] addresses this as it has the same linear piece for $x > 0$, but a modification for $x \leq 0$ that has a non-zero gradient:

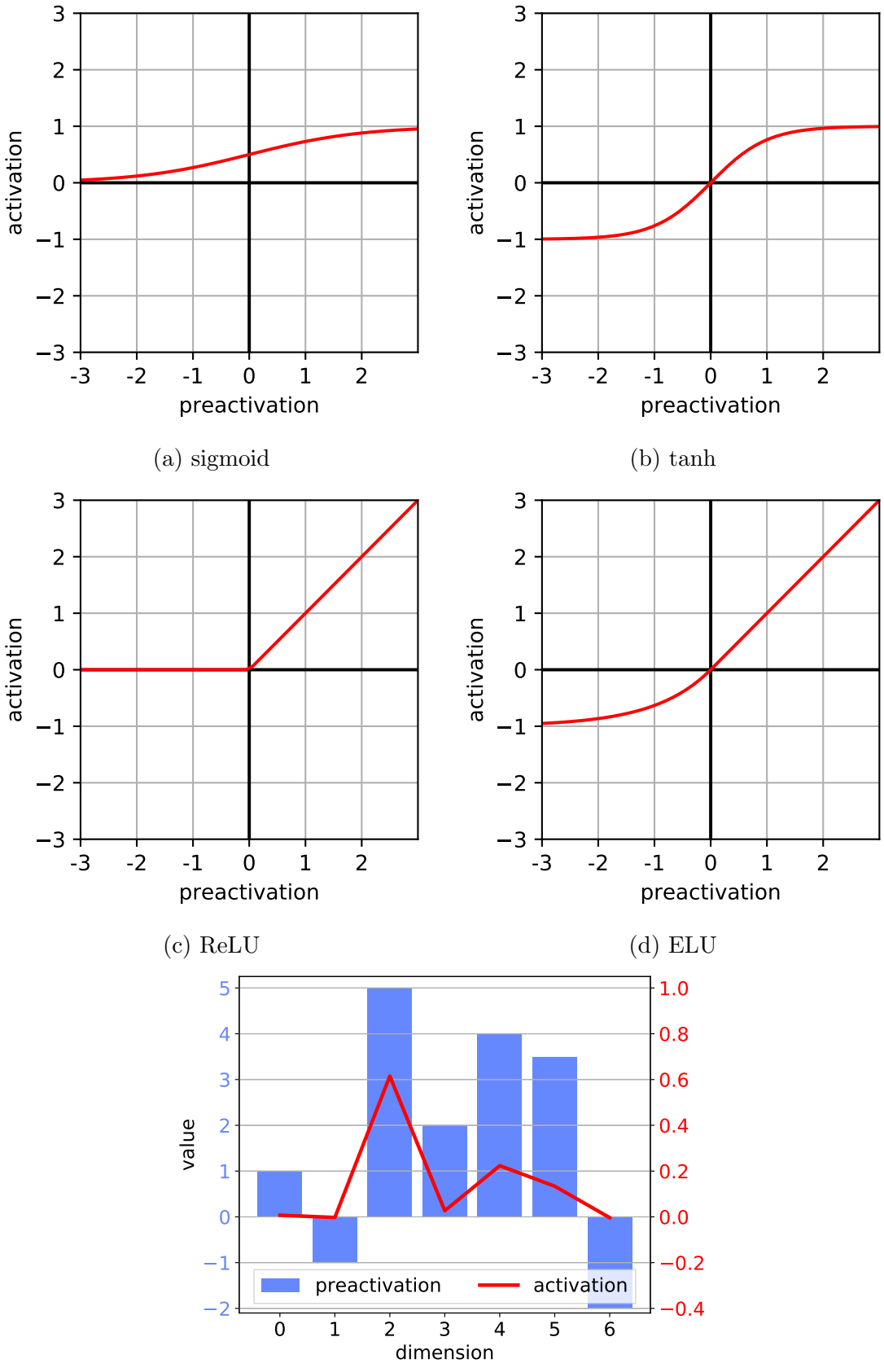
$$ELU(x) = \begin{cases} \alpha(e^x - 1) & x \leq 0 \\ x & x > x \end{cases} \tag{2.6}$$

Besides introducing a valuable non-linearity into the computation process of the network, activation functions are commonly used in output units to map their output (pre-activation) to the desired range. As we briefly mentioned, the sigmoid function maps the input values to a range of $(0, 1)$. Sigmoid units are commonly used to realize learnable and differentiable *gates* in modern networks. For an example, see the Gated Recurrent Unit (GRU) network introduced in Section 2.1.3. Furthermore, a network with a single sigmoid output unit could be used to predict a probability that corresponds to a binary class label. A network with n such sigmoid output units can produce n probability values and can thus be used to address multi-label classification. In many cases, however, we are interested in predicting a single class of n possible classes, i.e. multi-class classification. To enforce that the network output conforms to a probability distribution over all classes, we can employ the *softmax* activation function:

$$softmax(o) = \frac{e^{o_i}}{\sum_{j=1}^{n} e^{o_j}} \tag{2.7}$$

where $o = (o_1, \ldots, o_n)$ denotes the vector of pre-activations of the n output units. The softmax functions normalizes the pre-activations such that $softmax(o)_i \in (0, 1)$ for all $i = 1 \ldots n$ and $\sum_{i=1}^{n} softmax(o)_i = 1$. The softmax therefore always produces a valid distribution over the n classes and lends itself very well for multi-class classification problems. The normalization of an example input vector with the softmax function is depicted in Figure 2.2e.

Loss Functions We established that neural networks are trained by predicting an output for an input sample and then update the weights to better match a target output. The direction and step size of the weight updates is, determined by the error of the prediction as compared to the target output. The exact error is



(a) sigmoid

(b) tanh

(c) ReLU

(d) ELU

(e) softmax

Figure 2.2: An overview of common activation functions employed in this work.

computed with a differentiable loss function. A common choice for the loss function is the *categorical cross-entropy* that is used in conjunction with the softmax activation function for multi-class classification problems. For a single training sample it is defined as:

$$\mathcal{L}(q, p) = -\sum_{i=1}^{C} q_i log(p_i) \tag{2.8}$$

where q is the target distribution across all C classes and p is the predicted distribution. In most multi-class classification scenarios, the target distribution is in fact a one-hot vector with a probability of 1 for the target class c and probabilities of 0 for all other classes. In this case, the formulation can be simplified to:

$$\mathcal{L}(c, p) = -log(p_c) \tag{2.9}$$

The theoretical capabilities and versatility of artificial neural networks lead to their application to wide variety of research problems. In recent decades, the original design of the ANN has been developed further and new network topologies, unit types, and activation functions have been proposed to suit individual problems even further. In the following, we describe the main types of networks that are relevant to this thesis.

2.1.1 Feed-Forward Neural Networks

Feed-Forward Neural Networks (FFNNs) are a special case of Artificial Neural Networks with a constrained topology. In a FFNN, the computation flows in one direction, i.e. from the network input to the network output. The network features no feedback connections and contains no loops in general.

Layers Traditionally, these networks are separated into three types of layers:

1. the *input layer*,
2. several *hidden layers*, and
3. the *output layer*.

Each node in these layers is only connected to the nodes in the previous and the following layer. A network with this topology is also called a Multilayer Perceptron (MLP). Figure 2.3 depicts an instance of such a network.

When an input vector is passed to the input layer of an MLP, the vector is "passed through" the following network layers which each compute their own output vector using their connection weights and activation functions. At last,

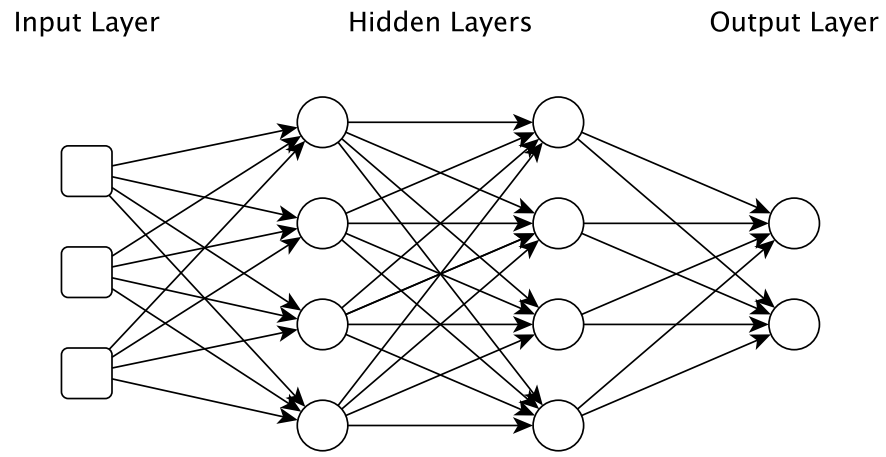


Figure 2.3: A feed-forward neural network with 3 input units, two layers with each 4 hidden units, and a final layer with 2 output units.

the output layer produces the overall network output: the *prediction* of the model given the input. The computation of a single layer can be formalized as follows:

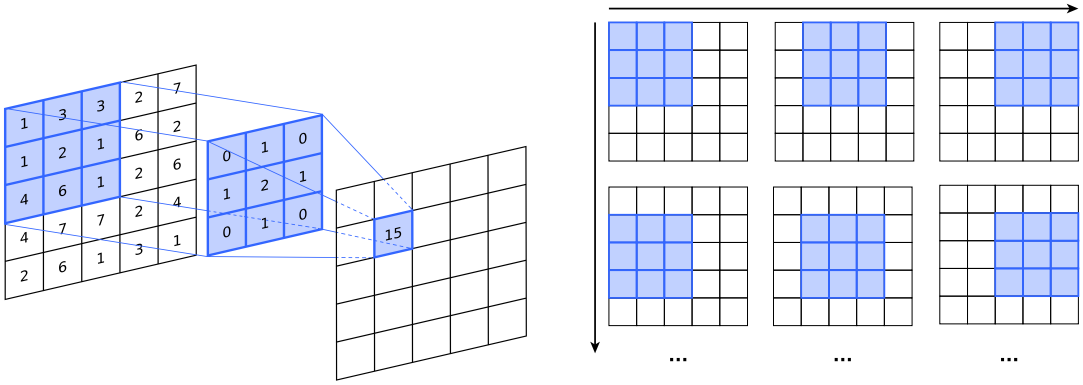
$$h = f(\mathbf{W}x + b) \tag{2.10}$$

where x is the layer input, $\mathbf{W}$ the weight matrix, b the bias vector, f an activation function, and h the layer output. We refer to such a layer as a *feed-forward* or *dense* layer.

In the areas of Computer Vision and Natural Language Processing, more advanced types of neural networks are commonly used that are better suited to deal with images and sequential data. In the following, we give a brief introduction to Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) which are a common ingredient of modern neural-network-based models.

2.1.2 Convolutional Neural Networks

Convolutional Neural Network (CNN) are a variant of feed-forward neural networks that originated in the field of computer vision [Fukushima, 1980]. These networks are specifically designed to process 2nd and 3rd order tensors (i.e. pixel grids) of different shapes. Inspired by the visual system, CNNs modify the standard, fully-connected topology of layers by restricting it to *local connections* for each neuron. Contrary to standard MLPs, each neuron in a CNN layer is connected only to a small set of input units of the previous layer. This set of input units is sometimes called the *receptive field* [Luo et al., 2016] of that neuron and is usually characterized by a *kernel size*, that specifies the width and height of the receptive



(a) The kernel of a CNN layer is applied to a window of the input centered around a pixel. The resulting response value determines the value of the feature map at that position.

(b) The application of a convolution layer can be interpreted as a sliding window operation applied to the entire input. The example above shows a 2-dimensional kernel of size 3×3 (blue area) that is applied to a 5×5 input matrix.

Figure 2.4: The application of the convolution operation.

field. In practice, the connection weights of a neuron to its receptive field are usually called a *kernel* or *filter* and are shared between all neurons of the same layer. This reduces the number of parameters of the network and introduces a measure of shift invariance into the network. With this formulation, CNNs can be interpreted as a regular feed-forward layer that is applied to a sliding window over the input. For an image that is represented as the matrix $\mathbf{x}$, the computation for one such window around pixel x_{ij} can be written as:

$$z_{ij} = \sum_{a=1}^{m} \sum_{b=1}^{m} w_{ab} x_{(i+a-c)(j+b-c)} \quad \text{with} \quad c = \frac{m-1}{2} \tag{2.11}$$

Here, $w \in \mathbb{R}^{m \times m}$ is the square kernel of the convolution layer[1]. The result of the convolution operation $\mathbf{z}$ over the entire input is called a *feature map* that corresponds to the size of the input. Figure 2.4a shows the application of the kernel to a single patch of an input image. The application of the sliding window over the image is depicted in Figure 2.4b. As with feed-forward layers, the feature map can be transformed by a non-linear activation function before passing it to following layers in order to increase the networks representational power. The convolution operation is commonly combined with pooling operations such as the maximum or average over a small window of the feature map. The pooling operations reduce

[1]For simplicity, we define the convolution layer only for square kernels with odd kernel sizes.

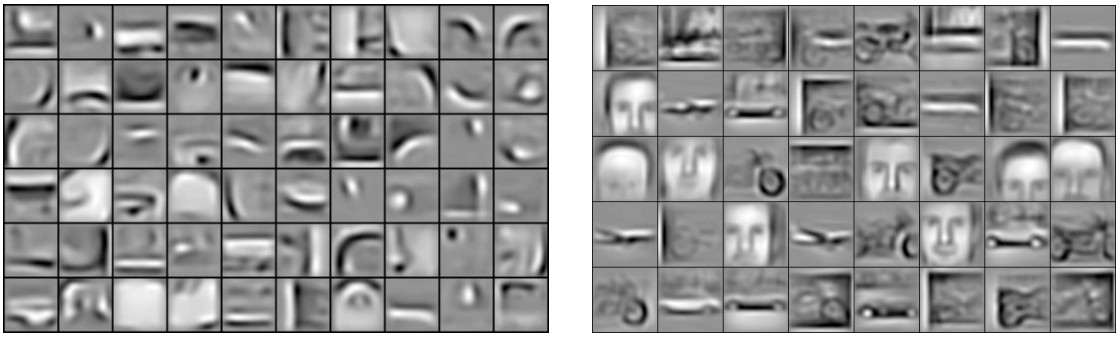


Figure 2.5: Stacking convolution layers enables to learn feature detectors of increasing complexity that correspond with intuitive concepts. The kernel weights learned from images from the categories faces, cars, airplanes, and motorbikes are shown above. The learned weights of the second layer (left) correspond to parts of faces and objects while the third layer (right) can be visualized to reveal full faces and vehicles. Images from Lee et al. [2009].

the dimensionality of an input and make the overall network invariant to small translations of input features [Weng et al., 1993].

A single convolution layer can be seen as a feature detector for an input. Since the convolution weights are part of the trainable network parameters, this feature detector can be optimized for a specific objective. It is possible to learn feature detectors with increasing complexity by stacking several convolution layers on top of each other. For a CNN network trained to classify images, it can be shown that the feature detectors in each layer correspond to an intuitive hierarchy of concepts such as *edges* → *shapes* → *parts* → *faces* [Zeiler and Fergus, 2014]. Figure 2.5 shows a visualization of learned feature weights for the second and third layers of a convolutional deep belief network [Lee et al., 2009].

CNNs for NLP Though originally designed for computer vision applications, CNNs can be applied to sequential data (e.g. texts) by using 1D kernels instead of 2D kernels[2]. Since a CNN layer slides its kernel over the input, it can be applied to inputs of varying size. This makes it especially suitable for NLP applications where the inputs are usually documents of varying lengths. When classifying texts of variable lengths, a common approach is to learn several layers of feature detectors using CNN layers and then produce a fixed length vector using pooling operations over the entire feature representation sequence. This pooling operation in text application is often called pooling *over time* alluding to the sequential nature

[2]Since each word in a text is represented by a vector, the kernel is in fact 2-dimensional. It is generalized to include several input *channels* that correspond to the input vector dimensions. However, the entire kernel is moved only along one axis and is hence referred to as 1D.

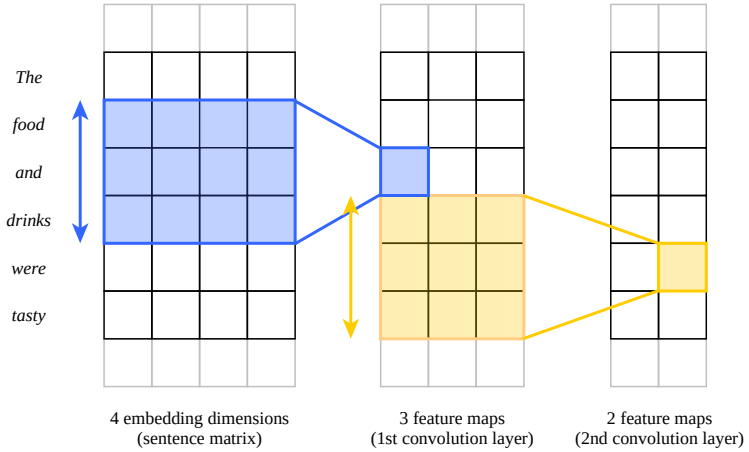


Figure 2.6: A simple CNN for text classification. The blue segment visualizes the application of a single kernel of the first convolution layer that slides over the input sequence and which outputs a single corresponding feature map. Two more kernels (not visualized here) separately produce the remaining feature maps of the first layer. The yellow segment depicts a kernel of the second convolution layer that receives the three feature maps of the first convolution layer as its input and outputs its own feature map.

of the text. By reducing the text sequences to a fixed length vector, the final classification can be carried out using regular feed-forward layers. In subsequent chapters, we refer to the computation of a single CNN layer with a shorthand notation: $\text{CNN}(\mathbf{x}) = \mathbf{h}$.

2.1.3 Recurrent Neural Networks

In the previous section, we introduced the CNN as a type of neural network that can be conveniently applied to texts due to a *sliding window* approach. The locally restricted connections of the CNN make it an efficient way to learn and compute a feature representation for e.g. a word by considering its immediate context. However, a more intuitive approach to processing sequential data such as texts is given by Recurrent Neural Networks (RNNs). RNNs process a sequence step by step (also called *timesteps*) and keep an internal *state* for each step. Additionally, RNNs extend the idea of feed-forward networks by introducing feedback connections into the network topology [Elman, 1990; Jordan, 1997]. A simple recurrent network

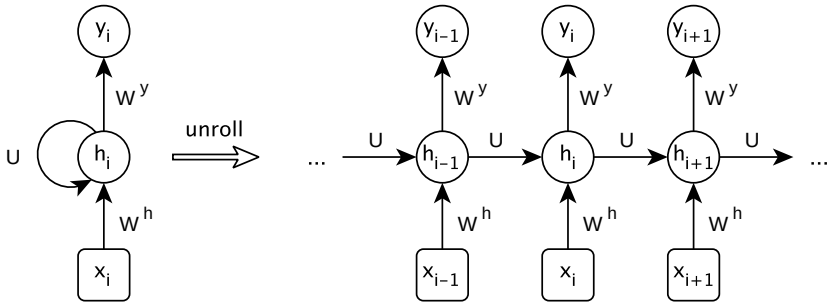


Figure 2.7: An RNN layer showing feed-forward and feedback connections. The timesteps of an RNN can be *unrolled*. The weights of each unrolled step are shared.

can be formalized as:

$$h_i = f^h(\mathbf{W}^h x_i + \mathbf{U} h_{i-1} + b^h)$$
$$y_i = f^y(\mathbf{W}^y h_i + b^y)$$

where $\mathbf{x} = (x_1, \ldots, x_n)$ is a sequence of input vectors and i denotes the current timestep. $\mathbf{W}^h$, $\mathbf{U}$, b^h, $\mathbf{W}^y$, and b^y are trainable parameters of the model. f^h and f^y are activation functions. With these feedback connections, a unit does not only receive its input from the previous layer but also from its own previous *state*. By including the previous state in the computation for the output of the current step, the network has access to the (transformed) history of the entire sequence that was processed so far. Figure 2.7 depicts an RNN that reads in a input sequence and produces a corresponding out put sequence. For the purpose of visualization, we *unroll* the individual timesteps of the RNN.

RNNs are naturally applicable to textual data by reading in a text word for word and producing a sequence of output vectors accordingly. The sequence of output vectors can be processed in further steps to address problems such as sequence labeling. RNNs are also capable of encoding an entire text as a single vector. This is usually done in one of three ways:

1. The text is processed word for word and the final output y_n of the RNN is selected as a *summary* that represents the entire text (cf. Sutskever et al. [2014]).

   $$s = y_n \tag{2.12}$$

   This is an intuitive approach, since the output of the final state can, in theory, incorporate features from the entire input text and thus provide a task-relevant summary.

2. Alternatively, a pooling operation (usually maximum or average) is applied

on the entire sequence of output vectors that aggregates the information in a single vector. This closely resembles the pooling operations in CNNs.

3. A weighted average of the output sequence is computed where the computation of the individual weights are optimized alongside the entire network. This is commonly referred to as an *attention mechanism* [Bahdanau et al., 2015].

Gated Recurrent Units

Standard recurrent neural networks are difficult to train in that they suffer from vanishing and exploding gradients during the backpropagation step [Hochreiter, 1991, 1998]. The result is that long-range dependencies between inputs and outputs are rarely modeled in the inner workings of an RNN. These issues can be alleviated by introducing gated connections. The introduction of gates allows to retain information in the memory for longer steps and thus learn tasks with long-range dependencies [Hochreiter et al., 2001]. The most famous RNN variants using gated connections are the Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] and GRU [Cho et al., 2014]. In this work we focus on GRUs due to their lower complexity in terms of computation and parameters but competitive performance [Chung et al., 2014a] with respect to LSTMs.

The GRU uses a combination of update and reset gates to improve its ability to learn long-range information comparable to Long Short-Term Memory cells [Chung et al., 2014a]. The computation of a single GRU layer at timestep i is defined as follows:

$$
\begin{aligned}
r_i &= \sigma(\mathbf{W}^r x_i + \mathbf{U}^r g_{i-1} + b^r) \\
z_i &= \sigma(\mathbf{W}^z x_i + \mathbf{U}^z g_{i-1} + b^z) \\
h_i &= f(\mathbf{W}^h x_i + \mathbf{U}^h (r_i \odot g_{i-1}) + b^h) \\
g_i &= (1 - z_i) \odot g_{i-1} + z_i \odot h_i
\end{aligned}
$$

where x_i is an element of an input sequence and g_i the computed output. σ is the sigmoid activation function and f is a non-linear activation function, originally the element-wise tanh function. The operator $\odot$ denotes the element-wise multiplication. The forget gate r_i regulates how much of the past output g_{i-1} can influence the current internal state h_i. A value close to 0 effectively blocks out past information. z_i is the update gate that controls how the past output g_{i-1} and the current internal state h_i should be combined for the final output. Here, a gate value close to 1 entirely exposes the current internal state at the output, while a value of 0 effectively copies over the last output. The flow of information for the GRU is visualized in Figure 2.8. In subsequent chapters, we refer to the

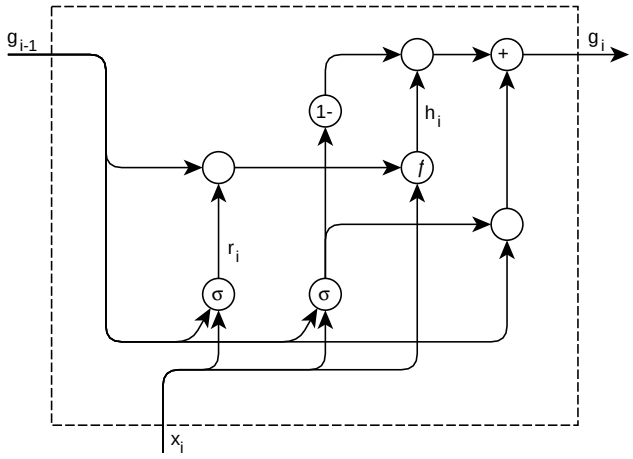


Figure 2.8: The Gated Recurrent Unit (GRU) uses a combination of update and reset gates to control the flow of information through between steps.

computation of a single GRU layer with a shorthand notation: $\text{GRU}(\mathbf{x}) = \mathbf{g}$.

Bidirectional RNNs

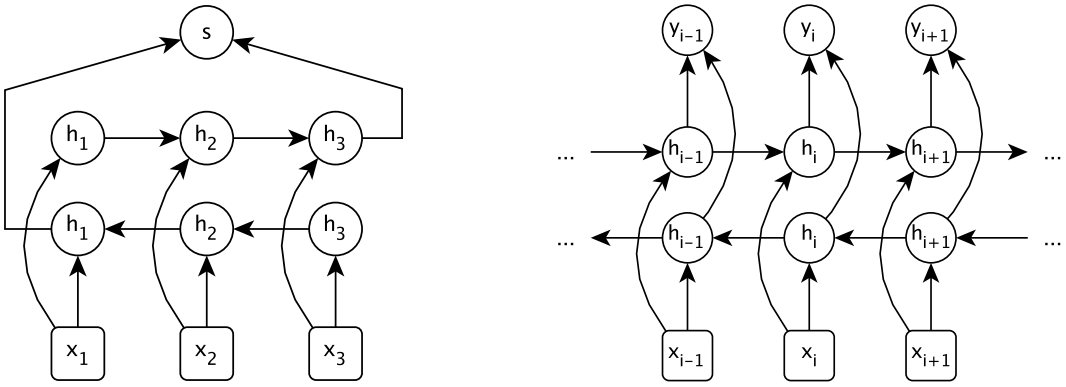
As we have seen before, RNNs process sequences in one direction. For text, this is usually the reading direction. This entails that information appearing later in the input text does not contribute to earlier steps of the RNN computations and can thus not influence the predicted output for these earlier steps. This is a problem for many sequence labeling tasks such as Named Entity Recognition (NER) that are more accurately modeled with contextual information from both directions. This problem can be mitigated with Bidirectional Recurrent Neural Networks (BiRNNs) [Schuster and Paliwal, 1997]. A BiRNN is a combination of two separate RNNs: a forward RNN and a backward RNN, denoted as $\overrightarrow{RNN}$ and $\overleftarrow{RNN}$, respectively. The forward RNN processes the input text from start to end while the backward RNNs reads in the text in reverse order. The outputs of the two RNNs are then concatenated whereby the combination procedure depends on the required output.

If a single summary vector for a text is required, the output of the forward RNN for the last word is concatenated with the output of the backward RNN for the first word:

$$\overrightarrow{RNN}(\mathbf{x}) = \overrightarrow{\mathbf{y}} \tag{2.13}$$

$$\overleftarrow{RNN}(\mathbf{x}) = \overleftarrow{\mathbf{y}} \tag{2.14}$$

$$s = \overrightarrow{y}_n \oplus \overleftarrow{y}_1 \tag{2.15}$$



(a) A bidirectional summary vector for a text that combines the final output of a forward-directed RNN with that of a backward-directed RNN.

(b) A bidirectional output sequence for a text that combines the individual outputs of a forward-directed RNN with a backward-directed RNN for every element in the sequence.

Figure 2.9: A BiRNN can (a) summarize an entire sequence, or (b) encode the individual elements of a sequence.

By doing so, both output vectors $\overrightarrow{y}_n$ and $\overleftarrow{y}_1$ correspond to the *last* step in the processing order of the respective RNNs. This procedure is often used in text classification models. Figure 2.9a illustrates this scenario.

In applications such as Part-of-Speech (POS) tagging or NER, an output sequence of vectors is required so that each input word has a corresponding output representation. In this case, the output vectors of the forward and backward RNN for each word are combined in separation:

$$\overrightarrow{RNN}(\mathbf{x}) = (\overrightarrow{y_1}, \ldots, \overrightarrow{y_n}) \tag{2.16}$$

$$\overleftarrow{RNN}(\mathbf{x}) = (\overleftarrow{y_1}, \ldots, \overleftarrow{y_n}) \tag{2.17}$$

$$\mathbf{y} = (\overrightarrow{y_1} \oplus \overleftarrow{y_1}, \ldots, \overrightarrow{y_n} \oplus \overleftarrow{y_n}) \tag{2.18}$$

$$\tag{2.19}$$

This way, the produced output vectors for every word are a function of the entire text to its left *and* its right allowing complete information flow from every input step to every output step. Figure 2.9b illustrates this idea.

2.2 Representing Textual Data

Many machine learning algorithms, and especially so neural networks, ultimately process data in vectorized form. To translate data into such a vector form is not always trivial and, in many ways, influences the performance of the algorithms. As an example, images are usually represented as matrices that reflect pixels in terms of colors and intensities. However, representing textual data in an adequate vectorized form is difficult. A common approach to represent a text is as a sequence of discrete tokens.

Each token is represented by a one-hot vector: a large, sparse vector with a single dimension with value 1 that corresponds to a specific word in a vocabulary V of possible words. While this representation is intuitive for categorical data in general, it has major drawbacks: Firstly, the one-hot vector is very large since it has a separate dimension for every single word in V. For English, this vocabulary comprises tens of thousands of words. This is especially problematic when materializing the representation for a large document of hundreds or thousands of words. For some NLP problems it is adequate to reduce the sequential representation of a document to a single fixed-sized Bag-of-Words vector by adding the individual one-hot vectors. The resulting Bag-of-Words representation comprises a dimension for each word in V with the value being the frequency of the word in the document. While this reduces the size of the document representation, it also discards the order of the words and thus omits most syntactic information. For many NLP tasks, such as Dependency Parsing, this is a poor representation that does not provide the necessary information for the task.

Another problem of representing words as independent dimensions is that this does not reflect the similarity of words. The one-hot vectors of any two words are orthogonal regardless of the similarity of words. As a result, a similarity in the vector space generally does not represent a similarity in the *meaning space* of words. A favorable representation would embed words with similar meanings or functions close to each other in the vector space to facilitate learning and generalization.

2.2.1 Word embeddings

Word embeddings (and sometimes called *word vectors*) address some of these issues by representing words with dense vectors in a latent space that better captures the meaning and functions of words. Some of the established approaches for computing word embeddings are inspired by the distributional hypothesis which states: *"A word is characterized by the company it keeps."* [Firth, 1957]. In recent years, a variety of word embedding methods have been put forth that are capable of leveraging large amounts of unlabeled textual data to compute general purpose word representations [Mikolov et al., 2013b; Pennington et al., 2014; Bojanowski

et al., 2017]. In the following, we introduce a prominent and efficient approach to compute word embeddings that plays an important role throughout this thesis.

Continuous Skip-Gram

The Continuous Skip-Gram model [Mikolov et al., 2013b], often just called Skip-Gram model, is a word embedding algorithm based on a shallow neural network architecture. The algorithm leverages unlabeled text datasets by constructing a simple language modeling tasks. Following the notation of Mikolov et al. [2013b] closely, the general objective of the algorithm is to maximize the average log probability of surrounding context words:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\ p(w_{t+j} \mid w_t) \tag{2.20}$$

where $w_1, \ldots, w_T$ is a sequence of words, and c the size of the training context. The probability of a context word w_c given a word w_i is defined as:

$$\frac{exp\big({v'_{w_c}}^{\top} v_{w_i}\big)}{\sum_{w=1}^{W} exp\big({v'_{w}}^{\top} v_{w_i}\big)} \tag{2.21}$$

where v_w and v'_w are the vector representations for w as a center word and a context word, respectively.

Due to the large size of the vocabulary, a naive implementation of the algorithm is intractable for large datasets. Each single classification would require the computation of the full probability distribution over the entire vocabulary at the output layer [Mikolov et al., 2013b]. The issue can be addressed in two ways:

1. Hierarchical softmax, or
2. Negative Sampling.

For 1, the full softmax function is approximated with a binary tree with words in its leaves. Each word is reachable via a distinct path through the tree. For 2, a modification of the Noise Contrastive Estimation [Gutmann and Hyvärinen, 2012] is employed. For each context word that is to be predicted, k *negative samples* are drawn from the vocabulary in accordance to a *noise distribution*. The modified objective is then to distinguish the correct context word from the randomly sampled noise words:

$$log\ \sigma({v'_{w_c}}^{\top} v_{w_i}) + \sum_{c'=1}^{k} \mathbb{E}_{w_{c'}\sim P_n(w)}\big[log\ \sigma({v'_{w_{c'}}}^{\top} v_{w_i})\big] \tag{2.22}$$

By performing a large amount of these seemingly simple classification tasks, it is possible to learn useful vector representations for a large vocabulary that are in line with our understanding of semantic similarity. The learned vector space has interesting properties. Firstly, the dimensionality of the final word vectors is a hyperparameter that is usually much smaller than the size of the original vocabulary. Secondly, the semantic similarity of words, albeit a vague concept, is often reflected by the similarity in the vector space. As an example, the four closest words in the learned vector space for the word *"bartender"* are *"waitress"*, *"bartenders"*, *"waiter"*, and *"barman"*[3]. Furthermore, the embeddings exhibit properties that support simple arithmetic operations in the vector space. The learned vectors allow us to resolve analogies such as "*man* is to *waiter* as *woman* is to _____" with simple vector arithmetic:

$$v_{waiter} - v_{man} = v_{?} - v_{women}$$
$$v_{?} = v_{waiter} - v_{man} + v_{women}$$

Retrieving the closest neighbor of the resulting vector $v_{?}$ from the pretrained word embeddings with respect to the cosine similarity gives the expected answer *"waitress"*.

2.3 Sequence Labeling

A central point of this thesis is the extraction of relevant phrases and expressions from documents. Finding these phrases in an automatic fashion can be approached in several ways: In some domains, relevant phrases can be extracted by designing linguistic rules based on POS and dependency patterns. Such an approach has been used for aspect extraction in opinion mining [Liu et al., 2015b; Poria et al., 2016a]. A drawback of this approach is the importance of expert knowledge in crafting these rules and the reliance on advanced linguistic tools that are not necessarily available for every domain and language. On the other hand, rule-based approaches are usually not supervised learning algorithms, alleviating the cost of annotating a suitable amount of training data.

An alternative approach is given with factor graphs [Frey, 1998]. An approach based on factor graphs can model phrases explicitly by sampling over possible phrase candidates and scoring individual solutions. Such an approach has been used to detect drug mentions in medical documents [ter Horst et al., 2017] but also subjective phrases and their targets in customer reviews [Klinger and Cimiano, 2013b]. Factor graphs are an interesting choice as they allow to model dependencies

[3]Using publicly available word embeddings trained on the Google News corpus: `https://code.google.com/archive/p/word2vec/`

explicitly through specific factors. On the contrary, the sampling space of these methods can be quite large making the sampling procedure costly.

The prevalent approach to extract phrases is to formulate the task as a sequence labeling problem by using an appropriate tagging scheme. In the following section, we elaborate this approach and present tagging schemes that are used throughout this thesis.

2.3.1 Tagging Schemes

The problem of identifying phrases of variable length in a text is commonly formulated as a sequence labeling problem. Given a text containing various phrases of interest, we can represent these phrases by assigning a tag to each single word according to a tagging scheme. The resulting sequence is a one-to-one mapping of words to tags and encodes which words are part of a phrase and which are not. In the following, we introduce some of the predominantly used tagging schemes in NER and sequence labeling in general.

IO scheme The simplest tag set is the IO scheme introduced by [Tjong Kim Sang and Veenstra, 1999]. Following this scheme, we distinguish between two types of tokens:

1. Tokens that are part of a phrase, i.e. **I**nside, are tagged with the **I** tag.

2. All other tokens, i.e. tokens **O**utside of any relevant phrase, are tagged with **O**.

This tagging scheme is straight-forward and boils down to a simple binary classification per token. However, decoding such a tag sequence is inherently ambiguous as there is no indication to separate adjacent phrases. This is especially an issue in domains with colloquial writing styles, where punctuation symbols are omitted that would otherwise mark the boundaries between phrases. As such, this tagging format is only reasonably applicable when adjacent phrases are not expected or only a single phrase per document is possible.

IOB2 Scheme The IOB2 schemes addresses the limitations of the IO format by introducing a new tag for the beginning of a phrase. In the IOB2 scheme, the **B** tag is assigned to the first token of each phrase while the remaining tokens of a phrase still receive the **I** tag. This extension removes the ambiguity for adjacent phrases, thus making the encoding and decoding lossless.

	We	enjoyed	pizza	santa	fe	chopped	salad	and	fish	and	chips	.
IO	O	O	I	I	I	I	I	O	I	I	I	O
IOB	O	O	I	B	I	I	I	O	I	I	I	O
IOB2	O	O	B	B	I	I	I	O	B	I	I	O
IOBES	O	O	S	B	I	I	E	O	B	I	E	O

Figure 2.10: Four common tagging formats for representing text chunks.

IOB Scheme As a slight variation of this, the IOB format restricts the usage of the **B** tag and only applies it to the first token of a phrase that immediately follows another phrase. In all other cases, the full phrase is tagged with **I** tags. Figure 2.10 shows the three tagging schemes.

Besides these formats, several other variations can be found in the literature. The IOE and IOE2 formats correspond to IOB and IOB2, respectively, yet mark the last word of a chunk with an **E** instead of the **B** tag for the initial word. The IOBES extends these by using both **B** and **E** tags for every multi-word chunk and a single **S** tag for single-word chunks. Some studies report performance differences of several chunk representations for different tasks, languages, and learning algorithms with mixed results: Ratinov and Roth [2009] show that the IOBES (BILOU) representations performs significantly better in NER tasks compared to the common IOB (BIO) format. In contrast, Konkol and Konopík [2015] show that the performance of a tagging scheme also depends on the language of the tagged documents and the used sequence labeling algorithm. Their findings suggest that the IOBES representation is not particularly good for the NER task. Rather, the IOE and IOE2 formats *"seem to be the best or at least reasonable choice for almost all languages and methods."[Konkol and Konopík, 2015].* Cho et al. [2013] present a method for combining segment representations and achieve improved results when combining the most complex and the least complex tagging formats. The authors surmise that the complex format support the model in developing a higher discriminative power while the simpler format alleviates problems associated with data sparseness.

2.4 Metrics

In the course of this thesis, we propose a variety of machine learning models that address complex annotation tasks in an automatic fashion. To assess the performance of these models, we perform experiments on manually labeled datasets and compare the predicted results with the given true labels. In this section, we establish the core metrics that are commonly used in the field of ABSA.

F_1-Score, Precision, and Recall *Precision* and *recall* are metrics that are widely used to evaluate NLP tasks that are concerned with the retrieval of a set of interesting items (e.g. entities) from a potentially very large set of candidates (e.g. every possible phrase). An example of this is NER [Derczynski, 2016]. In this context, precision measures the fraction of *retrieved items* that are correct as compared to a set of *target items*.

$$Precision = \frac{|\{\text{target items}\} \bigcap \{\text{retrieved items}\}|}{|\{\text{retrieved items}\}|}$$

A system that is very selective and only returns a few but correct items can thus reach high precision values. It is maximized when all retrieved items are correct.

On the other hand, recall measures the fraction of *target items* that were correctly retrieved.

$$Recall = \frac{|\{\text{target items}\} \bigcap \{\text{retrieved items}\}|}{|\{\text{target items}\}|}$$

To achieve a high recall, a system needs to retrieve as many target items as possible even if it retrieves a few wrong items as well. The score is maximized when all target items have been retrieved.

Put together, these two scores characterize the performance of a retrieval system well. A combination of the two scores is given by the *F_1-score*. The F_1-score, also commonly referred to as the *F_1-measure*, is defined as the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

It gives a balanced view on the extraction performance and is the de facto standard metric for NER and opinion target extraction. In these areas, the aforementioned items are phrases that are to be identified in a given text. It is worth noting that opinion target extraction is usually evaluated with exact matches. That means that the identified start and end positions of a phrase need to be *exactly* the same as those of the target phrases. Partial matches are considered errors in this formulation. In the following, we give an example for the computation of precision, recall, and F_1-score for a set of predicted phrases given a target set of phrases[4]:

Targets:

"The [*signs*], *the* [*specials menus*], [*food*], *and even all the* [*waitstaff*]

[4]In practice, we represent phrases as start and end offsets in order to properly account for phrases with the same surface form.

are ALL TOTALLY Japanese."

→ { *"signs"*, *"specials menus"*, *"food"*, *"waitstaff"* }

Predictions:

"The signs, the specials menus, food, and even all the waitstaff are ALL TOTALLY Japanese."

→ { *"signs"*, *"menus"*, *"food"*, *"the waitstaff"*, *"Japanese"* }

Given these sets, we can calculate the precision, recall, and F_1-score with the formulas above as:

$$Precision = \frac{|\{\text{“signs”, “food”}\}|}{|\{\text{“signs”, “menus”, “food”, “the waitstaff”, “Japanese”}\}|} = 0.4$$

$$Recall = \frac{|\{\text{“signs”, “food”}\}|}{|\{\text{“signs”, “specials menus”, “food”, “waitstaff”}\}|} = 0.5$$

$$F_1 = 2 \cdot \frac{0.4 \cdot 0.5}{0.4 + 0.5} = 0.\overline{4}$$

Accuracy The *accuracy* of a classification is characterized by the fraction of correctly classified instances compared to the total amount of instances:

$$Accuracy = \frac{|\{\text{correct instances}\}|}{|\{\text{all instances}\}|}$$

The accuracy constitutes an intuitive metric and is commonly used when the label distribution is roughly balanced. We use the accuracy throughout this thesis to evaluate the classification of sentiment polarity.

Chapter 3

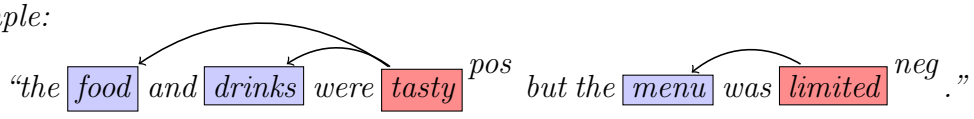
Relational Sentiment Analysis

Chapter Overview *This chapter presents the formulation of sentiment analysis as a relation extraction problem. We approach this problem by dividing it into four subtasks that we separately address using novel architectures. Our findings suggest that a combination of CNN and RNN layers outperforms alternatives for opinion target extraction. By jointly addressing opinion target extraction and opinion phrase extraction in a single model, we can further achieve performance improvements. Moreover, we propose a position-aware RNN for phrase-specific sentiment classification and relation extraction and achieve large improvements over baseline approaches.*

3.1 Introduction

Sentiment Analysis (SA) is the task of classifying expressed opinions in natural language texts. Although Sentiment Analysis (SA) has been addressed in the form of a classification task of documents to a large extent [Liu et al., 2010; Yang et al., 2016], this is not sufficient. As seen in Chapter 1, opinions can be expressed in complex ways. While a document might have an overall sentiment theme, the actual opinions are often more nuanced. In fact, it is common that opinion holders express multiple opinions towards different targets in a single sentence. The following example clearly shows that a single sentiment label cannot accurately represent the expressed opinions in that sentence:

Example:



In this example, an opinion holder expresses three distinct opinions. A positive sentiment is expressed towards the two aspects *food* and *drinks*, indicated by the

term *tasty*. A third, negative opinion is expressed through the term *limited* and targets the *menu*. To provide accurate analyses of expressed opinions, these need to be modeled at a more fine-grained level than the document- or sentence-level.

In the course of this chapter, we advocate to model sentiment analysis as a relation extraction problem. In this framework, we decompose it into four subtasks:

1. the extraction of *opinion targets* that specify the aspect of a product, theme or event that is assessed,
2. the extraction of *opinion phrases* which indicate the reason and polarity of an opinion,
3. the labeling of these opinion phrases with a *sentiment label* (e.g. "positive", "neutral", "negative"), and
4. the extraction of *relations* between targets and opinion phrases.

The entirety of relational sentiment analysis can thus be tackled in a modularized way, facilitating the otherwise complex problem. The modularization allows us to develop machine learning components that are focused on the respective subtask. As the subtasks are in line with existing research areas (e.g. sequence tagging or relation extraction), we can build on findings of these fields. Overall, our work is in line with the growing interest of providing more fine-grained, aspect-based sentiment analysis, going beyond a mere text classification or regression problem that aims at predicting an overall sentiment for a text. In the following, we give an overview of related research fields that this chapter touches upon.

Opinion Extraction The area of Opinion Role Labeling (ORL) is concerned with identifying relevant text fragments that define an expressed opinion. Often these include *opinion holders* and *opinion targets* but can also encompass *opinion phrases*. Such an annotation scheme is realized by the MPQA Opinion Corpus [Wiebe et al., 2005] and the VERB corpus [Wiegand et al., 2015] and approached by many researchers [Josef Ruppenhofer and Wiebe, 2008; Yang and Cardie, 2013; Katiyar and Cardie, 2016; Marasović and Frank, 2018]. In this thesis, we are not concerned with detecting opinion holders, as the considered datasets are customer reviews. In reviews, the opinion holder does not contribute much to the understanding of the expressed opinions as it is implicitly clear that the opinion is held by the author of a review. In accordance with that, the formulation of ABSA by Pontiki et al. [2014, 2015, 2014] that is concerned with customer reviews, omits the opinion holder entirely and focuses on the opinion target. Since the first iteration of the SemEval Challenge on ABSA, Pontiki et al. [2014] model sentiment analysis with opinion targets, target-specific sentiment labels, and aspect categories. In

contrast to the aforementioned ORL, the SemEval ABSA formulation does not include explicit opinion phrases that express the sentiment polarity. A different route to opinion extraction is taken by Hu and Liu [2004]. They address opinion extraction as the summarization of reviews. They present an approach that summarizes reviews based on the product features for which an opinion is expressed using data mining and natural language processing techniques. Similarly, Titov and Mcdonald [2008] describe a statistical model for joint aspect and sentiment modeling for the summarization of reviews. The method is based on Multi-Grain Latent Dirichlet Allocation which models global and local topics extended by a Multi-Aspect Sentiment Model.

Opinion Target and Opinion Phrase Extraction San Vicente et al. [2015] present an approach that addresses opinion target extraction as a sequence labeling problem based on a perceptron algorithm with local features. The system also implements a sentiment polarity classifier to classify individual opinion targets. The approach uses a window of words around a given opinion target and classifies it with a Support Vector Machine (SVM) classifier based on a set of features such as word clusters, POS tags and polarity lexicons. Opinion target and opinion phrase extraction for sentiment analysis has also been addressed using probabilistic graphical models. Toh and Wang [2014] for instance propose a Conditional Random Field (CRF) as a sequence labeler that includes a variety of features such as POS tags and dependencies, word clusters and WordNet [Fellbaum, 1998] taxonomies. Additionally, the authors employ a logistic regression classifier to address opinion target polarity classification. Jakob and Gurevych [2010] follow a very similar approach that addresses opinion target extraction as a sequence labeling problem using CRFs. Their approach includes features derived from words, POS tags and dependency paths, and performs well in a single and cross-domain setting. Klinger and Cimiano [2013a,b] have modeled the task of joint target and opinion phrase extraction using probabilistic graphical models and rely on Markov Chain Monte Carlo methods for inference. They have demonstrated the impact of a joint architecture on the task with a strong impact on the extraction of opinion targets, but less so for the extraction of opinion phrases. The approach to semantic role labeling proposed by Fonseca and Rosa [2013] is closely related to our approach to opinion target extraction in that the task is phrased as a sequence tagging problem using convolutional neural networks. Most relevant in terms of opinion target and opinion phrase extraction are the works of Liu et al. [2015a] and İrsoy and Cardie [2014]. Liu et al. [2015a] address the extraction of opinion expressions while İrsoy and Cardie [2014] focus on the extraction of opinion targets. Both approaches are framed as a sequence labeling using RNNs.

Targeted Sentiment Analysis Zhang et al. [2016] propose an RNN-based classifier to predict the sentiment label of target phrases in a text. The model processes the target phrase, as well as the left and right context separately and is thus able to extract target-specific sentiment. Brun et al. [2016] propose a model for aspect category and target sentiment classification. The model relies on a rich, term-centric feature representation involving lexico-semantic features and syntactic dependency features that are classified using an Elastic Net regression model. Our work is related to other approaches using deep neural network architectures for sentiment analysis. Lakkaraju et al. [2014] present a recursive neural network architecture that is capable of extracting multiple aspect categories and their respective sentiments jointly in one model or separately using two softmax classifiers.

In the course of this chapter, we propose a position-aware RNN that leverages distance embeddings to perform targeted classification. The benefits of distance and position embeddings are explored and supported by [dos Santos et al., 2015b; Zeng et al., 2014; Sun et al., 2015]. More recently, the transformer architecture [Vaswani et al., 2017] shows the benefit of distance embeddings in combination with an attention mechanism as a way to encode structural information in a text without recurrent or convolutional networks.

Relation Extraction In Section 3.2, we phrase fine-grained sentiment analysis as a relation extraction problem. Our approach for relation extraction is inspired by the work of Zeng et al. [2014] who address relation extraction between pairs of entities using a convolutional neural network architecture. Their approach combines lexical features for both entities with sentence level features learned by a CNN model. The authors use learned position-embeddings to instill positional information about the subject and object entities into the CNN. We adopt a similar strategy as presented in [Zeng et al., 2014] to address relation extraction. In contrast to [Zeng et al., 2014] however, we use a recurrent neural network instead of a convolutional architecture to perform the actual relation extraction. Katiyar and Cardie [2016] model relations between opinion holders and opinion phrases, as well as opinion targets and opinion phrases in a single LSTM model. To extract relations between tokens, the model predicts relative distances for each token to its relation counterpart or 0 if no relation is assumed. In contrast to the model of Zeng et al. [2014] which assumes the entities as given, the LSTM model of Katiyar and Cardie [2016] has no explicit knowledge of entities when predicting relations but rather predicts both simultaneously.

3.1.1 Contributions and Structure of the Chapter

The aforementioned works address some form of sentiment analysis and relation extraction to some extent. However, we argue that sentiment analysis is a com-

plexly structured problem that is often framed in a too simplified way. Rather, it ought to be addressed as a relation extraction task between opinion expressions and their target phrases. In order to address this lack, we answer the following overarching research question throughout this chapter:

RQ1 *How can fine-grained sentiment analysis be addressed in its complexity?*

We particularly focus on the following facets of this question:

RQ1.1 *How can relevant phrases be identified?*

RQ1.2 *What is the impact of domain-specific word embeddings on the task?*

RQ1.3 *How can dependencies between aspect and opinion phrases be leveraged?*

RQ1.4 *How can we infer the sentiment pertaining to a particular expression in a document?*

RQ1.5 *How can we identify corresponding targets for an opinion phrase?*

In the scope of these questions, our contributions are the following:

i) We present a complete architecture that addresses sentiment analysis as a relation extraction problem to offer a very fine-grained analysis. The architecture works in modularized way by extracting targets and opinion phrases, opinion-phrase specific sentiment and opinion-target relations separately. For all four subtasks, we present a neural network based component that achieves competitive and state-of-the-art results without extensive, task-specific feature engineering.

ii) We show the impact of training the target aspect and opinion extraction component with word embeddings initialized from a domain-specific corpus, showing that using domain-specific embeddings increases performance by 6.5% F-Measure as compared to randomly initialized embeddings.

iii) We show the impact of a component performing target aspect and opinion phrase extraction jointly versus predicting each type of phrase separately, demonstrating that joint prediction increases F-measure performance by 1% for target phrases and 5% for opinion phrases.

iv) We present a novel approach that is able to extract opinion-phrase-specific sentiment. Our position-aware RNN achieves performances high above our baselines and provides the first results on the USAGE dataset for the task of opinion-phrase-specific sentiment prediction, thus setting a strong baseline for future research.

v) Finally, we show that the relation extraction component is applicable to the sentiment analysis problem and show that our approach outperforms the current state-of-the-art in opinion-target relation extraction by 15% F-measure.

The remaining part of the chapter is structured as follows: In Section 3.2, we propose our framework for relational aspect-based sentiment analysis. We suggest a set of subtasks for relational sentiment analysis that can be addressed by individual components. In Section 3.3, we address the subtasks of extracting opinion phrases and its targeted aspect phrases from texts. We first describe the overall procedure of extracting opinion phrases and opinion targets framed as a sequence labeling problem. Thereafter, we describe the core features underlying our models. Here, we motivate the usage of domain-specific word embeddings and Part-of-Speech tags. We introduce our baseline models based on RNNs and CNNs for sequence labeling and give intuitive explanations about their suitability for the task. Then, we propose two extensions to these baseline models: Firstly, we propose an extension that combines both types of networks in a stacked architecture. Secondly, as another extension, we propose a model capable of jointly extracting opinion phrases and opinion targets. Section 3.4 attends to the classification of opinion phrases with respect to one of four sentiment classes. A position-aware recurrent neural network extracts the expressed sentiment of each opinion phrase by using POS tags, word and distance embedding features. In the following Section 3.5, we present our component for extracting relations between target aspect and opinion phrases. We describe our data representation and classification model that is based on a RNN architecture similar to the one presented in Section 3.4. The relation model classifies extracted target and opinion phrases in a pair-wise fashion. We evaluate the proposed components of our framework in Section 3.6. The evaluation is carried out on two datasets which allow us to investigate different properties of the overall system. We conclude this chapter in Sections 3.6.7 and 3.7 where we summarize our findings, point out shortcomings and potential improvements.

3.2 A Relational Framework for Aspect-Based Sentiment Analysis

As we motivated before, sentiment analysis needs to be addressed in a fine-grained way. To simply perform sentiment analysis at the document or sentence level is insufficient as this cannot adequately model multiple, potentially conflicting opinions in a single sentence. Instead, aspect phrases ought to be considered which offer crucial information about the target of an expressed opinion. Furthermore,

to deduce the reasons for an opinion, the corresponding opinion phrases and its sentiment label (e.g. positive, negative) needs to be identified, as well. To this end, we advocate to frame sentiment analysis as a relation extraction problem. This approach offers possibilities for more fine-grained analysis. To reach a sufficient level of granularity, sentiment analysis can be divided into the following subtasks:

(i) Opinion Target Extraction

(ii) Opinion Phrase Extraction

(iii) Sentiment Classification

(iv) Relation Extraction

In the framework of relational sentiment analysis, we first detect occurrences of target aspect and opinion phrases. The detection of aspects and opinions can be done separately but, as we see in Section 3.6.4, solving these tasks jointly is beneficial to both sides. As a next step, we classify the previously detected opinion phrases into one of several sentiment classes. This classification considers the phrase in the context of the document to allow to disambiguate otherwise ambiguous phrases e.g. *"cold"* as in *"cold food"* or *"cold beers"*. Finally, we infer relations between extracted phrases which allows us to explicitly associate the subjective expression with the opinion target. A schematic visualization of the complete architecture can be seen in Figure 3.1.

As a first step of assessing the relational sentiment analysis framework, we propose and evaluate possible models for each of the subtasks. The extraction of target and opinion phrases can essentially be regarded as a tagging task and can potentially be tackled by sequence tagging techniques such as Hidden Markov Models, CRFs etc. Besides these, neural networks have sparked increasing interest in recent years and have been applied very successfully to a great variety of NLP-related tasks. In particular, CNNs have been proposed as a general method for solving sequence tagging [Collobert et al., 2011] and sequence classification [Kim, 2014]. Similarly, RNNs have been applied to a variety of NLP-related tasks [Cho et al., 2014] as they are a natural candidate for processing textual data (see Section 2.1.3). In this chapter, we build on these encouraging results and employ several neural network based components which we combine into a single system to address relational sentiment analysis in four steps.

All models are based on neural networks and use only a limited amount of task-specific features. This makes each model easily extensible and allows for an easier transfer to other datasets, domains, and related tasks. In the following, we present our components for each subtask.

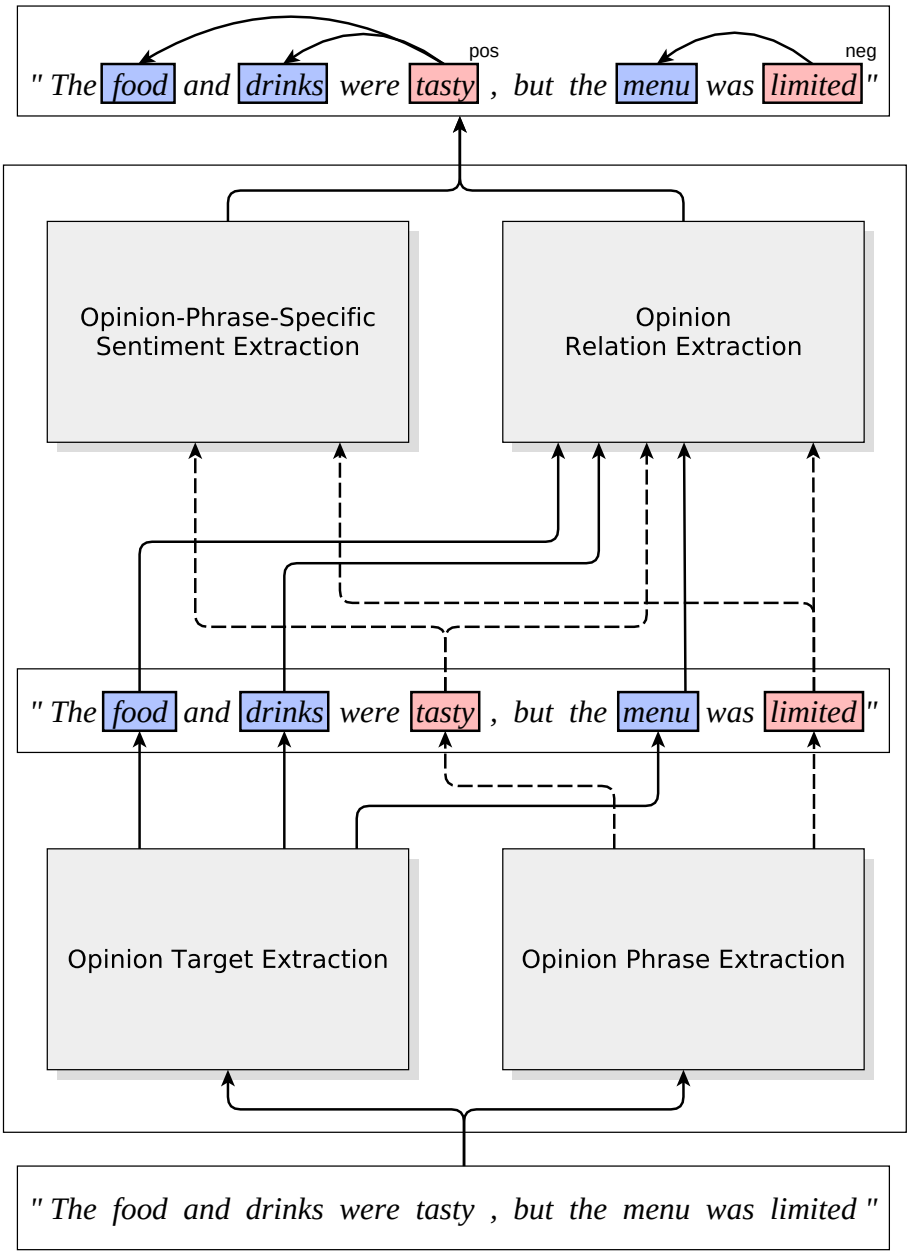


Figure 3.1: An architecture for sentiment analysis as a relation extraction problem. The architecture comprises four components that each address a subtask of the problem: i) The extraction of opinion target phrases, ii) the extraction of opinion phrases, iii) the classification of opinion-phrase-specific sentiment, and iv) the identification of relations between opinion phrases and opinion targets.

3.3 Opinion Target and Opinion Phrase Extraction

In this section, we propose different choices for neural-network-based components for the task of aspect and opinion phrase extraction. As both subtasks are structured similarly, we address both tasks using the same approaches. We interpret the problem as a sequence labeling task [Toh and Wang, 2014; Fonseca and Rosa, 2013] and predict sequences of tags for sequences of words. We use the IOB2 scheme [Tjong Kim Sang and Veenstra, 1999] to represent our aspect and opinion annotations as a sequence of tags (see Section 2.3). According to this scheme, each word in our text receives one of three tags, namely **I**, **O** or **B** that indicate if the word is at the **B**eginning, **I**nside or **O**utside of an annotation:

The	*tuna*	*and*	*wasabe*	*potatoes*	*are*	*excellent*	.	(Text)
O	**B**	O	**B**	**I**	O	O	O	(Target Phrases)
O	O	O	O	O	O	**B**	O	(Opinion Phrases)

In this example the *tuna* and the *wasabe potatoes* are opinion target annotations that we encoded with the IOB2 scheme. Similarly, the opinion phrase *excellent* is encoded in a separate tag sequence. Ultimately, each tag is represented as a one-hot vector:

$$I = \begin{pmatrix}1\\0\\0\end{pmatrix}, O = \begin{pmatrix}0\\1\\0\end{pmatrix}, B = \begin{pmatrix}0\\0\\1\end{pmatrix}.$$

As shown above, we encode opinion target annotations separately from opinion phrase annotations thus resulting in two separate tag sequences per review. This procedure is reasonable since overlapping opinion target expressions and opinion phrases are in principle possible[1]. Encoding these into a single tag sequence would require to use a larger, more complicated tag set as it would need to cover single *and* overlapping annotations, e.g. {`I-Target`, `I-Opinion`, `I-Target-Opinion`, `B-Target`, `B-Opinion`, `B-Target-Opinion`, `O`}. We expect this to hinder the learning procedure. For most experiments, we instantiate two separate models — one for extracting opinion targets and another for extracting opinion phrases — and train both separately to predict their respective tag sequences. However, as we later show in Section 3.3.6, it is also possible to extract aspect and opinion phrases jointly without using a larger tag set.

For the core of the sequence labeling model, we compare convolutional neural networks, recurrent neural networks and combinations thereof. While both CNNs

[1] We orient our design choices in accordance with the USAGE dataset that allows overlapping annotations.

and RNNs are capable of handling sequential data, it is unclear which of these is to be preferred for aspect and opinion phrase extraction. In the following, we discuss the input representation used by our systems, which includes both POS tags as well as word embeddings. The word embeddings are learned from a domain-specific corpus of Amazon reviews using a skip-gram model (see Section 2.2.1 in Chapter 2). Then, we present our different choices of components that we experimentally examine for the aspect and opinion phrase extraction subtask in Sections 3.6.4 and 3.6.4. We describe a convolutional network architecture as well as a recurrent network architecture as baseline systems. We then propose a stacked architecture that feeds features of multiple convolutional layers to a recurrent layer that, in turn, produces an output tag sequence. Additionally, we specify a joint model that shares parameters for the extraction of aspects and opinion phrases.

3.3.1 Domain-Specific Word Embeddings

Distributed word embeddings have been proven to be a useful feature in many NLP tasks [Collobert et al., 2011; dos Santos and Zadrozny, 2014; Le and Mikolov, 2014] since they often encode semantically meaningful information of words [Mikolov et al., 2013a; dos Santos and Zadrozny, 2014]. For word embeddings to be effective features in downstream tasks, they are usually trained on large unlabeled text collections. Besides different techniques for computing word embeddings from these data collections, the choice of the data itself is important. We argue that different domains use a different vocabulary and phrasing which ultimately affects the computed word representations. Since the meaning of words can differ between domains, we expect that precomputing word embeddings on domain specific data might lead to better overall results for downstream tasks acting in the same domain.

To confirm this hypothesis for the downstream task of ABSA, we use three sets of word embeddings. The first set of embeddings are randomly initialized without using any pretraining. The second set of embeddings are pretrained on English Wikipedia articles and act as our baseline word embeddings for a general domain. The third set are domain-specific embeddings trained on product reviews. We pretrain the word embeddings using the skip-gram model with hierarchical softmax as it is implemented in the topic modeling library *gensim* [Řehůřek and Sojka, 2010]. To quantify the impact of using these domain-specific embeddings, we also compute word embeddings on a domain-independent corpus of Wikipedia articles. As we show in Section 3.6.3, the domain-specific word embeddings indeed outperform the more general Wikipedia word embeddings.

3.3.2 Part-of-Speech Tags

While word embeddings encode features of a word that are useful to many NLP tasks, it is often beneficial to introduce further, more linguistically motivated features. We argue that POS tags are a helpful feature for subtasks of relational sentiment analysis and that they complement pretrained word embeddings. To test this hypothesise, we evaluate the use of POS tags as additional word-level features in our sequence tagger models. We obtain these tags for a document with the Stanford POS tagger [Manning et al., 2014]. The underlying tag set contains 45 tags to which we add one additional *padding* tag. We encode these tags as one-hot vectors which results in a POS tag feature vector $p_i \in \mathbb{R}^{46}$ for each word.

3.3.3 Convolutional Neural Network Model

While CNNs were originally intended for image processing, they have been successfully applied to several NLP tasks, as well [Poria et al., 2015; Kim, 2014; dos Santos et al., 2015a]. When working with sequences of words, convolutions allow to extract local features around each word.

The CNNs component for sequence tagging that we propose is composed of several sequentially applied layers that transform an initial sequence of words (i.e. a review) into a sequence of IOB2 tags. This sequence of tags encodes predicted aspect (or opinion) phrase annotations for the given review. More formally, the process from word sequence to tag sequence can be described as follows:

Given a sequence of n of words:

$$\mathbf{w} = (w_1, \ldots, w_n)$$

that correspond to a vocabulary V, our model applies a word embedding layer to each sequence element to retrieve a sequence of word embeddings $x_i \in \mathbb{R}^{D_{word}}$:

$$\mathbf{x} = (x_1, \ldots, x_n).$$

This is done by treating the embedding matrix $W_{word} \in \mathbb{R}^{D_{word} \times |V|}$ as a lookup table and returning the column vector that corresponds to the respective word index. Optionally, we are able to use POS tags as additional features. To do so, we concatenate the corresponding one-hot vector p_i to the word embedding x_i and use the resulting sequence:

$$\mathbf{x}' = (x_1 \oplus p_1, \ldots, x_n \oplus p_n)$$

in place of $\mathbf{x}$ in the next steps. We refer to $\mathbf{x}$ (or $\mathbf{x}'$, respectively) as our input representation.

After we obtain the initial feature representation for the input text, each window of l_{conv} consecutive vectors in $\mathbf{x}$ around x_i is convolved into a single vector $h_i^1 \in \mathbb{R}^{D_{conv}}$, where D_{conv} specifies the number of feature maps for this convolution[2]. Precisely, the convolution is performed on the concatenated sliding-window $z_i \in \mathbb{R}^{D_{word} \cdot l_{conv}}$ that we define as:

$$z_i = x_{n-(l_{conv}-1)/2} \oplus \cdots \oplus x_{n+(l_{conv}-1)/2},$$

where $\oplus$ marks the concatenation operation. The convolution at position n in the sequence is then:

$$h_i^1 = f(W_{conv} z_i + b_{conv}),$$

where the kernel matrix $W_{conv} \in \mathbb{R}^{D_{conv} \times D_{word} \cdot l_{conv}}$ and the bias vector b_{conv} are shared across all windows for this convolution. The function f is an element-wise, non-linear activation function such as the ReLU function. See Chapter 2 for a description of the ReLU function. The resulting hidden sequence as obtained by the first convolution layer is then:

$$\mathbf{h}^1 = (h_1^1, \ldots, h_n^1).$$

For simplicity, we denote the computation of a single CNN layer as

$$\mathbf{h} = \mathrm{CNN}(\mathbf{x})$$

This convolution operation can be applied several times (with different weights and on the output sequence of the previous convolution) to yield a sequence of more abstract representations:

$$\mathbf{h}^m = (h_1^m, \ldots, h_n^m) = \mathrm{CNN}(\mathbf{h}^{m-1}).$$

In a last step, we apply a standard feed-forward layer with a softmax activation function to each individual sequence element that projects the hidden representation h_i^m to a vector of $D_{IOB2} = 3$ probabilities for the corresponding tags I, O or B:

$$t_i = softmax(W_{tag} h_i^m + b_{tag}).$$

Figure 3.2a depicts the architecture.

The network is trained by minimizing the cross-entropy loss for the classification of each token which we recall from Eq. (2.9) in Chapter 2 as the negative

[2]Since we want to apply the convolution operation to the first and the last element in a sequence, too, we pad the input sequence with vectors of 0s.

logarithm of the probability of the expected tag $c_i \in \{I, O, B\}$ of word w_i:

$$\mathcal{L}_{tok}(c_i, t_i) = -log(t_{ic_i})$$

The loss for a training text is then defined as the average loss of its tokens:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{tok}(c_i, t_i) \tag{3.1}$$

which we try to minimize for the entire training corpus. If not stated otherwise, the formulation of the loss is the same for following sequence tagging models.

3.3.4 Recurrent Neural Network Model

In Section 2.1.3 of Chapter 2, we stressed that RNNs are naturally capable of processing textual data as they are designed to handle sequential inputs and outputs. To leverage this, we define a second, RNN-based baseline component to perform aspect and opinion phrase extraction that we compare to the CNN model from the previous section. We chose the GRU layer [Cho et al., 2014] to power our RNN model. To avoid repetition, we refer to Section 2.1.3 of Chapter 2 for a more detailed description of the GRU.

The RNN model receives the same input representation $\mathbf{x}$ as the CNN model which comprises the word embeddings of the input sequence and, optionally, the sequence of POS tags. We apply a single recurrent GRU layer and obtain a sequence of hidden states:

$$\mathbf{h} = (h_1, \ldots, h_n) = GRU(\mathbf{x})$$

Analogously to the CNN approach, the produced hidden states are then mapped to tag probabilities for each token:

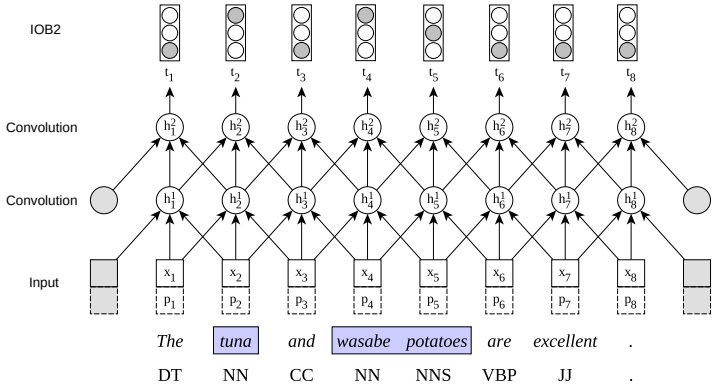
$$t_i = softmax(W_{tag} h_i + b_{tag}).$$

As before, we train the network weights by minimizing the cross-entropy loss for each token. Figure 3.2b depicts the architecture.

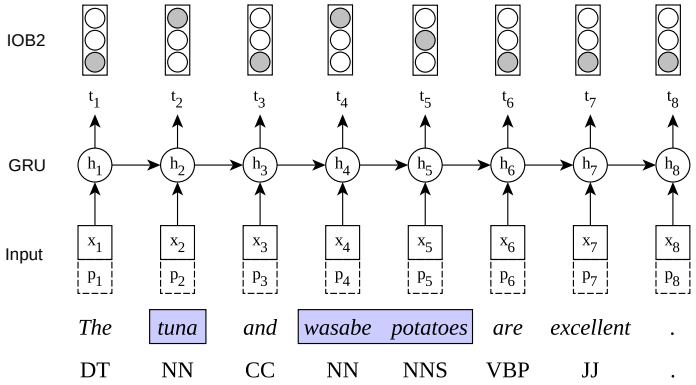
3.3.5 Stacked Model

The previous two sections describe the baseline models that we consider to tackle opinion target and opinion phrase extraction. In this section, we propose a combination of the previous two models.

The intuition for combining the CNN and the RNN model is that both models



(a) The CNN architecture for sequence labeling. The classification of each word depends on its local neighborhood by means of convolution operations.



(b) The RNN architecture for sequence labeling. The model reads the text from left to right and classifies each word given all the preceding words.

Figure 3.2: The baseline networks for sequence labeling. The words marked with boxes are annotated opinion targets and are tagged with the correct IOB2 tags at the output layer. The input to the network are word embeddings and optionally the corresponding POS tags. The gray vectors are padding vectors for the convolution operation. For simplicity, we show an instantiation of the CNN model with only two convolution layers.

present quite different approaches for the same task. While the CNN uses locally connected weights to produce a localized feature representation of a words immediate context, the RNN uses recurrent connections and allows for a much larger context. The RNN makes it possible to capture important pieces of information from preceding and potentially distant parts of the text. A combination of both models might benefit from both the local connectivity of the CNN and the larger available history of the RNN. Related ideas are expressed in Zhou et al. [2015a] and He et al. [2016].

We design the new, combined model as a stack of CNN and RNN layers. First, we apply a stack m convolutional layers to the input sequence, similar to the baseline CNN model, yet only up to the final hidden layer:

$$\mathbf{h}^1 = (x_1, \ldots, x_n) = CNN(\mathbf{x})$$
$$\mathbf{h}^m = (h_1^m, \ldots, h_n^m) = CNN(\mathbf{h}^{m-1})$$

On top of this sequence of high-level features, we stack a GRU layer that learns temporal dependencies of its input sequence:

$$\mathbf{h}^{m+1} = (h_1^{m+1}, \ldots, h_n^{m+1}) = CNN(\mathbf{h}^m)$$

Finally, a dense layer with a softmax activation is used to map the recurrent hidden states to tag probabilities:

$$t_i = softmax(W_{tag} h_i^{m+1} + b_{tag}). \tag{3.2}$$

The full model is trained end-to-end just as the tagging models before. Figure 3.3 visualizes the architecture.

3.3.6 Joint Opinion Target and Opinion Phrase Extraction

The aforementioned neural networks are general sequence labeling architectures that predict a single output tag for each input token. To extract two different types of phrases, i.e. opinion targets and opinion phrases, we can instantiate two separate models and train these to predict their respective tag sequences. This formulation of the problem is straightforward. However, we argue that opinion phrases and their targets are interdependent and knowledge about opinion phrases in a text could help extracting opinion targets and vice versa. A single model could leverage interactions by jointly predicting opinion phrases and opinion targets. In fact, the idea of extracting opinion targets and opinion phrases jointly has been addressed by Klinger and Cimiano [2013a,b] using a factor graph model and later by Wang et al. [2017a] in a neural network architecture. A similar idea is expressed

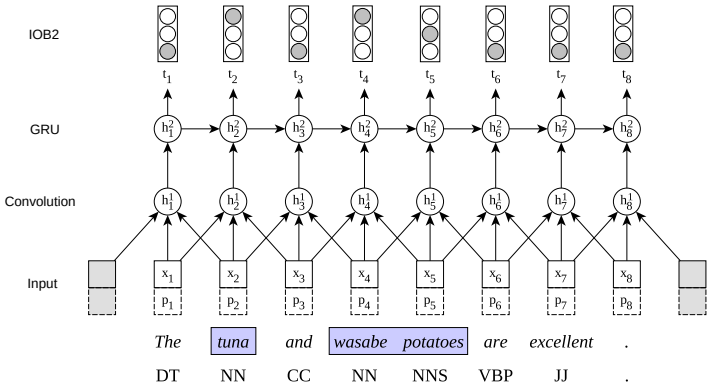


Figure 3.3: The stacked CNN-RNN architecture. The extracted features from the convolution layers are further processed by a GRU layer. In this example, we only show a single convolution layer.

by Katiyar and Cardie [2016] who jointly extract opinion holder entities, opinion targets and their relations.

In the following, we propose an architecture that performs the extraction of opinion targets and opinion phrases jointly. The network shares large parts of its parameters between the two tasks but features task-specific output layers. By training the network to perform well for both objectives the model is constrained to learn an internal feature representation that is beneficial for both tasks. We realize this concept by introducing two separate, task-specific output layers. While the shared parts of the network are optimized with training signals from both objectives, the output layers are optimized for single tasks only: either opinion phrase extraction or opinion target extraction. Formally, we modify Eq. (3.2) for the output layer of the stacked model as such:

$$t_i^T = softmax(W_{tag}^T h_i^{m+1} + b_{tag}^T)$$
$$t_i^O = softmax(W_{tag}^O h_i^{m+1} + b_{tag}^O)$$

This gives us two output tag sequences $\mathbf{t}^T$ and $\mathbf{t}^O$. During the training of the network, we optimize the weights such that $\mathbf{t}^T$ produces the tag sequence corresponding to the target aspects and $\mathbf{t}^O$ the sequence for the opinion phrases. This is achieved by redefining the loss function in (3.1) as the sum of the losses for the individual tag sequences:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{tok}(c_i^T, t_i^T) + \mathcal{L}_{tok}(c_i^O, t_i^O)$$

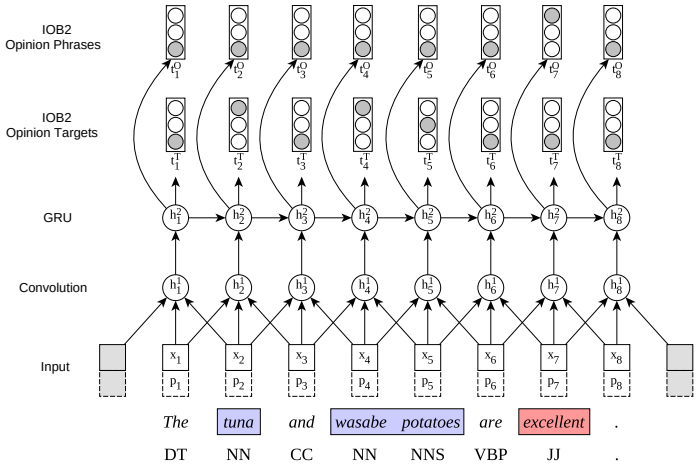


Figure 3.4: The CNN-RNN architecture for joint target and opinion phrase extraction. This model is capable of predicting aspect and opinion phrases jointly by featuring two separate output layers.

With this modification we obtain a single model that can jointly produce two distinct tag sequences. See Figure 3.4 for a depiction of the joint architecture.

3.4 Opinion-Phrase-Specific Sentiment Classification

With the previously described models, we are able to detect mentioned aspect and opinion phrases. The second step in our pipeline for relational sentiment analysis is to predict sentiment labels for these opinion phrases. The difficulty here is that it is not enough to extract an overall sentiment for a given review. Rather, the sentiment needs to be extracted with respect to one of possibly several opinion phrases in a text. In this section, we propose a position-aware recurrent neural network for this task: an RNN augmented with learned distance embedding features for targeted sentiment classification.

3.4.1 Position-Aware Recurrent Neural Network

In Chapter 2, we pointed out that a vanilla RNN is capable of encoding a text sequence as a single summary vector by e.g. selecting its final hidden state as the summary vector. If trained correctly, this summary vector contains condensed information that is relevant to the training objective. Many RNN-based text classification models use such a summary vector as input to a classification layer.

However, the summary vector is not conditioned on a particular phrase but rather on the entire sentence and is thus not suitable to classify the sentiment pertaining to a particular phrase in a text.

In this thesis, we propose a position-aware RNN to challenge this problem. The proposed network uses additional positional information about a specific phrase and computes a summary vector that is conditioned on this phrase. The conditioned summary is different for each phrase and allows the network to capture relevant sentiment information for this particular phrase. Let us consider the following example:

Coffee	*stays*	*fresh*	*and*	*hot*	*in*	*the*	*Carafe*	(Text)
NNP	VBZ	JJ	CC	JJ	IN	DT	NN	(POS)
-1	0	0	1	2	3	4	5	(Distances)

In this example, an opinion holder expresses a positive opinion towards the *Coffee* which is indicated by the opinion phrase *stays fresh*. Another opinion is expressed through the phrase *hot* which is also targeted at the *Coffee*. Below, the corresponding sequence of POS tags is given. Our procedure for classifying the sentiment value of a phrase classifies each opinion phrase separately. In the given example, we focus on the phrase *stays fresh*. To encode the position of this phrase in the full text, we compute the relative distance of each token to the opinion phrase in question. These distance values are displayed in row (Distances). We convert each word into its respective vector representation using the pretrained lookup table of word embeddings, thus obtaining a sequence of word vectors. Similar to this, we also use an embedding layer for the relative distances that provides us with a vector representation of dimensionality $D_{dist} = 10$ for each distance value. The distance embeddings are treated as learnable parameters and are optimized alongside other weights of the model.

The individual vectors of the three sequences — word embeddings x_i, POS tags p_i and distance embeddings d_i — are concatenated, resulting in a single sequence with $D_{word} + D_{pos} + D_{dist}$ dimensional elements:

$$\mathbf{x}' = (x_1 \oplus p_1 \oplus d_1, \ldots, x_n \oplus p_n \oplus d_n) \tag{3.3}$$

We feed the resulting sequence to a recurrent neural network consisting of three layers. The first hidden layer is a GRU layer with D_{gru} hidden units that reads in the sequence of vectors and produces a sequence of hidden states:

$$\mathbf{h} = (h_1, \ldots, h_n) = GRU(\mathbf{x}').$$

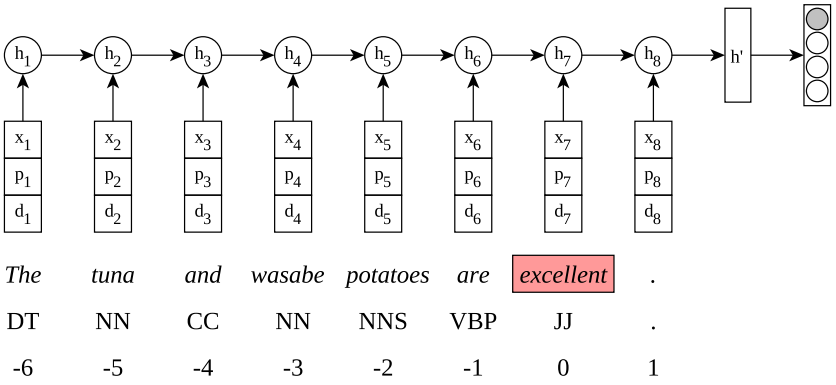


Figure 3.5: The position-aware RNN for opinion-phrase-specific sentiment classification. The red box at the input marks an opinion phrase. Below, the POS tags and relative word distances are shown. The input vectors are composed of three parts: one part for the word embeddings, one part for the POS tag vectors, and a final part for the distance embeddings. The output layer contains one unit for each possible sentiment label: `positive`, `neutral`, `negative`, and `unknown`.

We define the summary vector s as the final hidden state:

$$s = h_n.$$

We recall that the input sequence comprises the relative distances to a specific opinion phrases and that s is thus conditioned on that phrase.

The second layer is a feed-forward layer of ReLU units that transforms the summary vector s into a vector h' of D_{pol} dimensions. Lastly, another feed-forward layer maps the previous hidden layer to the desired number of output classes using a softmax activation function. For our problem at hand, the output units correspond to one of four possible sentiments: `positive`, `neutral`, `negative`, and `unknown`. Finally, the sentiment with the highest probability at the corresponding output unit is the predicted sentiment. Figure 3.5 visualizes the network. In practice, we do not pass the entire sentence to the RNN for each opinion phrase. Rather, we prune the input text by extracting a window of $l_{pol} = 20$ words centered around the opinion phrase[3]. Using just a subsequence of words and POS tags around each opinion phrase is reasonable since the lengths of the considered reviews reach up to several hundred words. We expect that relevant sentiment information is generally located close to the opinion phrase and that pruning the input helps focusing on these parts of the text. An evaluation of the proposed architecture is given in Section 3.6.5.

[3]Sequences for review texts with less than l_{pol} words are padded at the left with zeros.

3.5 Relation Extraction

In the previous section, we proposed a position-aware RNN model that uses relative distance embeddings in order to provide positional information about a phrase to a classification layer. In this section, we extend the position-aware RNN to be capable of addressing relation extraction between opinion targets and opinion phrases. As outlined earlier, our relational sentiment analysis pipeline first extracts opinion targets and opinion phrases and, in a later step, establishes the corresponding relations between the inferred phrases. We approach the extraction of opinion-target relations as a classification problem: Given a text $\mathbf{w}$, an opinion phrase o, and a target phrase t we predict a binary label $r \in \{0, 1\}$ that indicates whether the text expresses a relation between o and t.

Again, we employ a position-aware neural network. To encode information about the distance between the aspect and opinion phrases with respect to their position in the review, we follow a similar approach as in Section 3.4 which was originally inspired by Zeng et al. [2014] and dos Santos et al. [2015b].

Our approach employs four types of features for a given pair of target and opinion phrase: i) the sequence of word embeddings for the review text, ii) the sequence of corresponding POS tags for each word, iii) a sequence of relative distances of each word to the target phrase, and iv) a sequence of relative distances of each word to the opinion phrase. For a real world example, this could look as follows:

I	*like*	*all*	*the*	*different*	*features*	.	(Text)
PRP	VBP	PDT	DT	JJ	NNS	.	(POS)
-4	-3	-2	-1	0	0	1	(Distances^T)
-1	0	1	2	3	4	5	(Distances^O)

Here, the word *like* is an opinion phrase that indicates the sentiment toward the target phrase *different features*. Below, the corresponding sequence of POS tags is shown. The rows labeled with (Distances^T) and (Distance^O), show the sequence of relative distances of each word to the target and opinion phrase, respectively.

The first step in classifying the triple $(\mathbf{w}, o, t)$ is a pruning heuristic that automatically rejects all instances where the opinion phrase o and the target phrase t are more than 20 words apart from each other. While this does reject some valid relations, we can still predict 98% of relations correctly for this maximum distance of 20 words. For those pairs which are below the rejection distance, our model extracts a context of 20 of words centered around the two phrases[4]. Analogously, the sequence of POS tags is extracted.

[4]Sequences for review texts with less than 20 tokens are padded at the left with 0s.

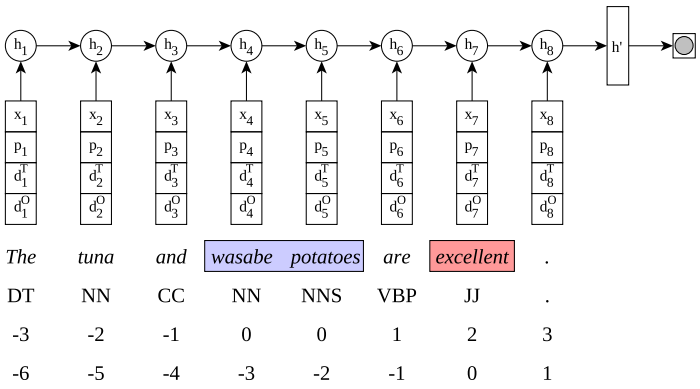


Figure 3.6: The position-aware RNN for opinion-target relation extraction. The colored boxes at the input mark the opinion-target pair that is to be classified. Below, the POS tags and relative word distances to the aspect and opinion phrase are shown. Each of the four input sequences are concatenated and passed to the GRU layer. The final output layer contains one single sigmoid unit. An output value above 0.5 indicates a relation between the aspect and opinion phrases.

Again, we use an embedding layer to obtain a sequence of $D_{dist} = 10$ dimensional embedding vectors for the relative distances. The individual vectors of the four sequences — word embeddings x_i, POS tags p_i, target distance embeddings d_i^T and opinion distance embeddings d_i^O — are concatenated:

$$\mathbf{x}' = (x_1 \oplus p_1 \oplus d_1^T \oplus d_1^O, \ldots, x_n \oplus p_n \oplus d_n^T \oplus d_n^O)$$

We feed the resulting sequence to a GRU layer with D_{gru} hidden units that computes a sequence of hidden states and a summary vector for the input:

$$\mathbf{h} = (h_1, \ldots, h_n) = GRU(\mathbf{x}') \tag{3.4}$$

$$s = h_n. \tag{3.5}$$

The summary vector s is passed on to a feed-forward layer h' of D_{rel} maxout units [Goodfellow et al., 2013], a generalization of the ReLU. As a last step, h' is projected to a single output value using a feed-forward maxout layer with a sigmoid activation function that projects the output to a value between 0 and 1. We interpret the network's output as the probability that the text expresses a relation between the pair of target and opinion phrase. Figure 3.6 visualizes the network. With this model for the extraction of relations between opinion phrases and opinion targets, we established the final component of the relational sentiment analysis framework.

	Reviews	Sentences	Annotations
Train	254	1315	1654
Test	96	685	845

Table 3.1: An overview of the SemEval 2015 ABSA dataset for the restaurant domain.

3.6 Experimental Evaluation and Discussion

This section evaluates our proposed system for relational sentiment analysis. Our evaluation targets the individual components of the overall system, measuring their performances in isolation. For the evaluation, we consider two datasets which offer a different granularity in their annotations. The SemEval dataset does not provide annotations for opinion phrases, opinion-phrase-specific sentiment or opinion-target relations. As such, we only evaluate our component for opinion target extraction on this dataset. The USAGE dataset, on the other hand, offers annotations for all our subtasks, allowing us to evaluate all our components on this dataset. Where possible, we give precision, recall, and F_1-score for our own and baseline approaches. All experiments were performed with the deep learning library *Keras* [Chollet, 2015] and employ many of its pre-implemented algorithms. Tag sequences predicted by our approach are post-processed to yield only sequences that are valid according to the IOB2 scheme.

3.6.1 Datasets

For the evaluation of this work, we employ two datasets that provide annotated reviews for the task of aspect-based or relational sentiment analysis.

SemEval 2015

The SemEval 2015 Task 12 dataset [Pontiki et al., 2015] is used to evaluate our systems for the subtask of opinion target extraction. The dataset provides a collection of review sentences from different domains (Restaurant, Laptops, Hotels). We only make use of the data from the restaurant domain as it contains annotations for explicitly mentioned opinion targets. The datasets for the laptop and hotel domains only contain annotations for aspect categories without annotated textual mentions. Note that we can only use this dataset to evaluate our architecture on opinion target extraction since the dataset is not annotated with respect to opinion phrases, and therefore also without explicit target-opinion relations. An overview of the dataset is given in Table 3.1.

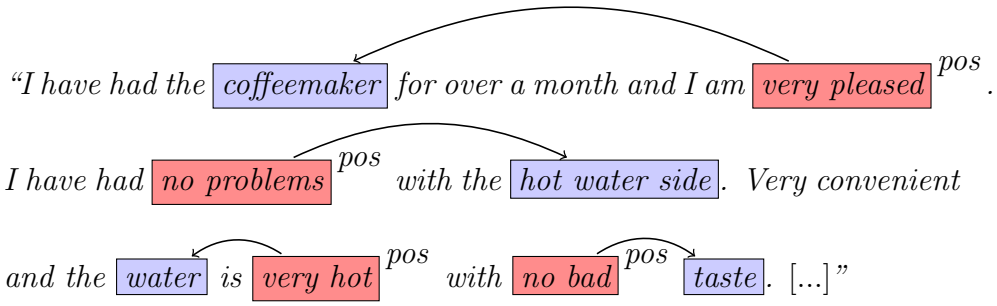
Reviews	Opinion Targets	Opinion Phrases	Relations	Categories
622	8545	5321	4481	8

Table 3.2: An overview of the USAGE dataset for relational sentiment analysis.

USAGE

The USAGE corpus [Klinger and Cimiano, 2014] is a collection of annotated English and German Amazon[5] reviews of different product categories. The annotations include (among others) mentioned aspect phrases, opinion phrases marked with a sentiment label and relations between aspect and opinion phrase. In contrast to the SemEval dataset, the reviews are not separated into sentences but rather treated as a whole. An example from the dataset is given below:

Example:



This dataset provides annotations for all considered subtasks, hence we will evaluate all our components on this dataset. However, we restrict our use of this corpus to the annotations for the English reviews. A high-level overview of the dataset is given in Table 3.2. For a more detailed description, we refer to Klinger and Cimiano [2014].

3.6.2 Experimental Settings

This section briefly outlines our training procedure. For comparability with previous works, we comply with established evaluation practices for the used datasets.

The standard procedure for evaluations on the SemEval 2015 dataset uses an official train and test split. Therefore, our experiments for opinion target extraction in Sections 3.6.3 and 3.6.4 follow this setup. Since many of our network

[5] https://www.amazon.com/

parameters are initialized randomly and our training data is processed in a random order, the performance of our model might be influenced by these external factors. To mitigate these effects, we perform each experiment on this dataset three times and averaged the obtained results. The USAGE dataset does not provide official splits for the data. Instead, prior work performed experiments in a 10 fold cross-validation on the available data which we adopt for our experiments on this dataset.

In all experiments, we keep the word embedding dimension fixed at $D_{word} = 100$. If not specified otherwise, the architecture of the CNN component is fixed as well. It is composed of 3 convolution layers with each $D_{conv} = 50$ kernels, a kernel size of $l_{conv} = 3$, and ReLU activation functions. We do not employ a max pooling operation after convolutions in order to retain the initial sequence length. However, we employ dropout [Srivastava et al., 2014] with a drop probability of 0.5 after each convolution as a regularization to prevent overfitting. Furthermore, we set the dimensionalities D_{gru}, D_{pol}, and D_{rel} of the hidden layers of the other models to 100.

3.6.3 Evaluation: Domain-Specific Word Embeddings

In our first experiment, we compare the performance impact of initializing the weights of the word embedding lookup table in the stacked CNN-RNN model with:

i) randomly initialized embeddings,

ii) pretrained embeddings trained on Wikipedia articles, and

iii) pretrained embeddings trained on domain-specific product reviews.

Our intuition in using domain-specific embeddings is that we expect these to reflect the target domain more closely and thus constitute a favorable text representation for relational sentiment analysis on reviews. With the following evaluation we aim to answer the research question:

RQ1.2 *What is the impact of domain-specific word embeddings on the task?*

The dataset of domain-specific reviews is provided by McAuley et al. [2015b] and consists of roughly 83 million reviews from 1996 to 2014. All reviews are lowercased and the dimensionality of the word vectors is set to $D_{word} = 100$. Rare words that appear less than 10 times in the corpus are replaced with a special token `<UNK>`. This token is later used to represent previously unseen words in order to provide a vector for each word at test time. The resulting vocabulary

Word	speed	quality	display
Nearest Neighbors	*spped*	*qualtiy*	displays
	speeds	*qualilty*	*diplay*
	speeed	*qulaity*	*dislay*

Table 3.3: Three commonly used words in product reviews and their 3 nearest neighbors in the embedding space. Often, misspelled versions (*italic*) of the original word are among its closest neighbors.

contains about 1 million unique words which we trim to the 200000 most frequent words to reduce the memory footprint of our model.

A welcome side effect of using this huge dataset of reviews is that we obtain word embeddings for misspelled forms of a word that appear commonly in reviews. As shown in Table 3.3, the learned representation of a misspelled word is in many cases very close[6] to its correctly spelled counterpart. The informal writing style of customer reviews is therefore, to some extend, reflected in the representation of words themselves.

For all three types of embeddings, we train the stacked CNN-RNN model on the training portion of the SemEval 2015 dataset and measure the extraction performance of opinion targets on the official test data. We measure the performance in terms of exact matches using the F_1-score (see Section 2.4 in Chapter 2). The results are summarized in Table 3.4. The experiments show that our model achieves on average an F_1-score of 0.581 with the randomly initialized embeddings. The initialization with the Wikipedia embeddings leads to a performance improvement of 3.7 points, and the domain-specific review embeddings give an even greater improvement of 6.5 points. As expected, we can observe a large benefit in pretraining our word embeddings on large collections of natural language texts with the largest gain using domain-specific embeddings. Considering these first results, we only use the domain-specific review embeddings as initialization in further experiments.

3.6.4 Evaluation: Opinion Target and Opinion Phrase Extraction

This section evaluates our relational sentiment analysis architecture focusing on the component for opinion target and opinion phrase extraction. We perform the evaluation in two steps: First, we evaluate the component on the SemEval 2015 dataset measuring its performance for opinion target extraction only. Keep in

[6]We use the euclidean vector distance as a distance measure.

Embeddings	F_1
Random	0.581
Wikipedia	0.618
Reviews	**0.646**

Table 3.4: Results for opinion target extraction on the SemEval 2015 test dataset using different initializations of the word embedding layer of the CNN+RNN model. The domain-specific embeddings pretrained on product reviews achieve the best result.

mind that we cannot evaluate the other subtasks on this particular dataset due to limitations of the provided annotation. These subtasks are evaluated in later sections.

Sequence Encoder

This section evaluates our opinion target extraction component. We compare different choices for the underlying sequence encoding and evaluate the use of POS tags as additional linguistic features. With this, we answer the research question:

RQ1.1 *How can relevant phrases be identified?*

We train and test the CNN, the RNN and the stacked CNN-RNN model from the Sections 3.3.3, 3.3.4, and 3.3.5 on the SemEval 2015 dataset using the official training and test split. Again, we perform each experiment thrice to account for differences due to the networks' initializations and training sample orders. Table 3.5 shows the average precision, recall, and F_1-score for each model.

The model based on convolutional layers and the model based on recurrent layers both perform similarly regarding the overall F_1-score. Combining both types of models in a stack-like architecture results in an increased averaged F_1-score that is statistically significant ($p = 0.05$). Providing additional features in the form of POS tags also improves the extraction performance. While we still perform not quite as good as the current state-of-the-art system EliXa [San Vicente et al., 2015], we do perform well with respect to the overall ranking of the SemEval 2015 task as can be seen in [Pontiki et al., 2015]. The benefit of our method is that it is not restricted to the mere extraction of opinion targets but that it is capable of extracting opinion targets and opinion phrases jointly. The following section shows that our method achieves state-of-the-art performance on the USAGE dataset with this joint extraction objective. Since our method uses only a very limited set of features, i.e. word embeddings and POS tags, we believe that improvements can

	Opinion Targets		
Model	P	R	F_1
RNN	0.592	0.646	0.618
CNN	0.558	0.702	0.621
Stack	0.599	**0.703**	0.646
Stack+POS	**0.633**	0.689	0.659
EliXa [San Vicente et al., 2015]	–	–	**0.701**

Table 3.5: Precision (P), Recall (R), and F_1-scores for opinion target extraction on the SemEval 2015 test dataset using different model architectures and additional POS tag features.

be obtained using a similarly rich feature set as that of EliXa and other competing systems. We leave the confirmation of this hypothesis to future work. Based on the results of these experiments, we perform all following opinion target and opinion phrase extraction tasks using the CNN-RNN model with additional POS tag features.

Joint Objective

The previous part of the evaluation focused on the performance differences of CNNs and RNNs as the core sequence encoder of the tagging model. This section investigates the benefits of predicting opinion target and opinion phrases jointly in one model, in contrast to two separate models. The evaluation addresses the research question:

RQ1.3 *How can dependencies between aspect and opinion phrases be leveraged?*

In this experiment, we compare a model with a joint training objective for opinion target and opinion phrase extraction to baseline models that address both tasks separately. To account for the added complexity that the joint training objective poses for the model, we consider two variations. The model *Joint* is instantiated with the same layer dimensions as the models *Targets only* and *Opinion only*. For the model *Joint (large)*, we doubled the layer sizes to observe the impact of added capacity on the joint model. Similar to Klinger and Cimiano [2014], we evaluate our models in a 10-fold cross-validation and report the averaged scores. Table 3.6 shows the results for the joint and the separate models.

We can see that extracting opinion target and opinion phrases jointly does indeed enhance the models performance, but only so for a larger network configuration. The extraction of opinion phrases benefits from the joint setting in

	Opinion Targets			Opinion Phrases		
Model	P	R	F_1	P	R	F_1
Klinger and Cimiano [2014]	–	–	0.56	–	–	0.48
Targets only	0.63	**0.70**	0.66	–	–	–
Opinion only	–	–	–	0.44	0.48	0.45
Joint	0.57	0.65	0.61	0.40	0.40	0.40
Joint (large)	**0.65**	0.69	**0.67**	**0.47**	**0.53**	**0.50**

Table 3.6: Precision (P), Recall (R), and F_1-scores for aspect and opinion phrase extraction on the USAGE dataset for joint and separate models.

particular. Here we observe a drastic improvements of the F_1-score by 5 points over a model that only extracts opinion phrases. However, this might also be attributed to the increased size of the hidden layers in our neural architecture. The extraction of opinion phrases might simply require more network parameters which it is able to claim in the larger joint architecture.

3.6.5 Evaluation: Opinion-Phrase-Specific Sentiment Classification

In previous experiments, we assessed our proposed components for the extraction of opinion targets and opinion phrases. Next, we present our evaluation of the position-aware RNN as proposed in Section 3.4. The model analyzes individual opinion phrases in a wide context and predicts one of four sentiment labels: `positive`, `neutral`, `negative`, or `unknown`. With this evaluation, we address the following question:

RQ1.4 *How can we infer the sentiment pertaining to a particular expression in a document?*

The subtask of opinion-phrase-specific sentiment classification is evaluated on the USAGE dataset which offers gold standard annotations for opinion phrases and their sentiment labels. We perform the sentiment classification on the these gold standard phrases in order to measure the suggested component in isolation. To keep the experimental settings consistent across our different components, the evaluation is performed as a 10-fold cross-validation.

At the time of publication, there are no comparable approaches published that address opinion-phrase-specific sentiment classification on the USAGE dataset. In the absence of published reference results, we also report the results of two baseline

Model	All	Conflicting
Majority (Positive)	0.646	0.439
baseline RNN	0.793	0.541
position-aware RNN	**0.836**	**0.731**

Table 3.7: Accuracy for opinion-phrase-specific sentiment classification. The proposed position-aware RNN is particularly beneficial for samples with conflicting sentiment expressions.

models: i) a naive baseline that always predicts the (most frequent) sentiment label `positive` (dubbed *Majority*), and ii) a regular RNN sequence classifier, i.e. an RNN that does not use distance embeddings (dubbed *baseline RNN*). We keep the results comparable across models and perform all experiments on the same cross-validation splits. Furthermore, the baseline RNN and the proposed position-aware RNN operate on the same window of tokens around each opinion phrase and are instantiated with identical hyperparameters. The only difference is the presence or absence of distance embeddings that inform about the exact position of the opinion phrase.

Besides reporting results for different models, we also compute the accuracy of all models on two test sets: i) the full test set of opinion phrases (dubbed *All*), and ii) the subset of opinion phrases that feature other phrases of different polarity in their near context[7] (dubbed *Conflicting*). The evaluation on the full test set gives an overview of the classification performances on the entire range of expressed opinions and facilitates comparability with potential future research results. On the other hand, the evaluation on the subset of conflicting opinion phrases gives us a clearer understanding of the benefits of the distance embeddings. It emphasizes difficult cases that cannot be adequately addressed by models that lack positional information. Table 3.7 shows the results for all models and test sets.

We see that the majority baseline gives an accuracy of 0.646 on the full test set which indicates a slight imbalance towards the positive sentiment in this dataset. Both RNN models score high above this naive baseline. The regular RNN reaches an accuracy of 0.793 on this test set. Our position-aware RNN surpasses this result and scores an average accuracy of 0.836. This shows that the additional distance embeddings improve the models overall performance as compared to the regular RNN model.

Looking at the subset of conflicting opinions, we can see that the baseline RNN is not capable of classifying these samples reliably and only reaches an average accuracy of 0.541. This can be attributed to the lack of positional information.

[7]The context is equivalent to the window of $l_{pol} = 20$ words centered around the opinion phrase as described in Section 3.4

Model	P	R	F_1
Klinger and Cimiano [2014]	–	–	0.65
Our Approach	0.87	0.75	**0.81**

Table 3.8: Precision (P), Recall (R), and F_1-scores for opinion-target relation extraction.

On the other hand, the position-aware RNN performs well yielding an average accuracy of 0.731 even for these difficult cases. While we do observe a small drop in performance for this subset, the relative benefit of the proposed method as compared to the baseline is especially pronounced here. We can conclude that the position embeddings are vital for the task and enable opinion-phrases-specific sentiment classification. Hence, with this work, we contribute the first results for this task on the USAGE dataset and set a strong baseline for future research.

3.6.6 Evaluation: Opinion-Target Relation Extraction

So far, we assessed the extraction of opinion target and opinion phrases as well as the classification of opinion phrases into one of four sentiment categories. The final component in the overall architecture for relational sentiment analysis is concerned with the extraction of relations between opinion phrases and its target aspects. With the evaluation of the proposed position-aware RNN for relation extraction, we provide an answer to the research question:

RQ1.5 *How can we identify corresponding targets for an opinion phrase?*

We perform the relation extraction on the opinion targets and opinion phrases from the gold standard annotations of the USAGE corpus, in order to measure the performance for relation extraction in isolation. This methodology is adopted from Klinger and Cimiano [2014] and allows us to compare our method to their work. Table 3.8 shows the results of a 10-fold cross-validation of our proposed component. The results show that our RNN-based model improves relation extraction by 15 points F_1-score compared to the probabilistic graphical model of Klinger and Cimiano [2014]. The strong result suggests that our model is capable of leveraging the word and distance embeddings alongside the POS tags to accurately link an opinion phrase to its correct target.

3.6.7 Discussion and Future Work

Our experiments and evaluations touched upon several facets of ABSA. We saw that domain-specific embeddings provide sensible embeddings for common misspelled words and are ultimately beneficial for OTE extraction. However, pretrained word-level embeddings can only represent *known* words and spelling errors, i.e. words that appeared in the pretraining corpus. A more dynamic approach is to model words at the character-level to learn relevant information about the word structure itself. We examine this idea in Chapter 5.

In Section 3.6.4, we evaluated different sequence encoders for the extraction of OTEs. Among the considered architectures were CNNs, RNNs, and combinations thereof. Our results clearly showed that a combined architecture obtains the best results. However, more specialized models with richer feature sets such as proposed by San Vicente et al. [2015] are required to reach state-of-the-art performance on this subtask. For the other subtasks in scope, further experiments need to be performed to determine the best architecture. Recently proposed encoder types such as attention-based RNNs [Bahdanau et al., 2015] or Transformers [Vaswani et al., 2017] are interesting candidates, especially so for targeted-sentiment analysis and relation extraction. The use of an attention mechanism is especially interesting for our position-aware RNN that, equipped with an attention mechanism, could use the positional information more explicitly. Zhang et al. [2017] propose a similar idea and report improvements for a slot filling task. In general, we expect our RNN-based components to benefit from bidirectional recurrent connections [Schuster and Paliwal, 1997] so that words appearing later in a sentence can be taken into account.

With our experiments in Section 3.6.4, we confirm results of prior work that show the effectiveness of jointly modelling opinion expressions and their target phrases [Klinger and Cimiano, 2013a]. How to extend the joint objective to include sentiment classification and relation extraction is not obvious and an interesting direction for future research. Potential avenues are outlined by research on joint entity and relation extraction [Gupta et al., 2016; Bekoulis et al., 2018].

The pipeline approach that we followed in this chapter allows for an easy modularization and evaluation of the overall system. We evaluated all components in isolation to allow for comparability with earlier works. An end-to-end evaluation from plain text to fully extracted opinion could give more insight into interdependencies of tasks and might reveal an alternative separation and order of execution of subtasks for this pipeline approach.

3.7 Conclusion

In this chapter, we presented a modular architecture that addresses sentiment analysis as a relation extraction problem. The proposed system divides the problem into four subtasks and addresses each with a dedicated component. This highly flexible approach offers a fine-grained solution for sentiment analysis.

As part of this overall architecture, we presented possible implementations for the individual components: First, we presented different neural network models that are capable of opinion target and opinion phrase extraction and which achieved competitive and state-of-the-art results on different datasets. We reported a benefit for this task in using domain-specific word embeddings compared to domain-independent and randomly initialized embeddings. We investigated the extraction of opinion targets and opinion phrases separately and jointly and found the joint approach to produce superior results in one setting. Thus, we confirm previous results from Klinger and Cimiano [2013a] who used a probabilistic graphical model instead of a neural network model. Secondly, we addressed opinion-phrase-specific sentiment extraction with a position-aware recurrent neural network using distance embedding features and achieved promising results that surpassed those of two baseline approaches by a large margin. The benefit was especially pronounced for difficult samples with conflicting sentiment expressions. Our results are the first sentiment extraction results on the considered dataset and provide a strong baseline for future research. Thirdly, we designed and evaluated a model for the extraction of relations between opinion targets and opinion phrases which outperformed prior results on the same dataset by 15 points in F_1-score.

This chapter shows that it is possible to divide sentiment analysis in a flexible and fine-grained way using a highly modular architecture. While the components we presented are all exchangeable, they do pose advantages in their current implementation. All proposed components stand out by their minimal use of hand-engineered features that are strongly tuned to their specific tasks. The only external resources that were used are machine-generated POS tags and word embeddings which were created with a data-driven approach. Nevertheless, all components perform competitive on their individual subtasks. It is easily conceivable to provide further task-specific features to improve the performances of the individual components even further. We address this in following chapters by leveraging structured, semantic knowledge bases and by focusing on problems arising from i) noisy, user-generated data (see Chapter 5), and ii) limited, annotated training data (see Chapter 6).

Chapter 4

Aspect-Based Sentiment Analysis and Structured Resources

Chapter Overview *In this chapter, we discuss how structured knowledge bases can support aspect-based sentiment analysis. This is achieved by integrating graph-based knowledge bases into a two-step machine learning pipeline. We demonstrate the positive impact of lexical and semantic knowledge for target-specific sentiment classification.*

4.1 Introduction

In the previous chapter, we showed that the complexity of ABSA can be modeled in a relational framework to a fine degree. In that framework, we subdivide the problem into four smaller sub-problems:

1. the extraction of opinion phrases,
2. the extraction of opinion targets,
3. the classification of opinion phrases into a set of sentiment categories, and
4. the extraction of opinion-target relations

In this chapter, we approach ABSA with a simplified framework that exchanges the expressivity of the relational framework for a simpler two-step setup. We adopt a formalization comprising: i) the extraction of opinion targets, and ii) the classification of target-specific sentiment, that is, the inference of the sentiment of an aspect *without* explicitly extracting opinion phrases and relations first. The intuition behind this approach is that is the expressed sentiment in a text is sometimes only implicitly stated and hard to pinpoint to a specific phrase. This

difficulty is reflected by the lower agreement between annotations of different annotators for opinion phrases compared to opinion targets in the USAGE dataset [Klinger and Cimiano, 2014]. In the following examples, it is easy to see that the sentiment towards the respective aspects is positive:

Example:

> *"Please take my advice, go and try this* place *."*

> *"There is something about their* atmosphere *that makes me come back nearly every week."*

However, it is difficult to locate a single succinct phrase that holds this information. By directly assigning the sentiment to the opinion target, we can circumvent this problem and classify implicit sentiment statements the same way as explicit statements. Furthermore, the annotation effort for the generation of a dataset of training samples can be reduced, albeit at the cost of losing fine-granular information about opinion phrases.

A finding of the previous chapter shows that it is possible to train supervised machine learning models for ABSA. The proposed models can learn from annotated training data and do not necessarily need much feature engineering. However, the regularities in natural language texts underlying ABSA are complex and might not be fully inferable from the annotated datasets alone. On the contrary, high-level features such as POS tags prove to be beneficial for opinion target extraction. In light of this, this chapter explores the usage of structured, semantic resources to boost the opinion extraction process. Semantic knowledge graphs provide human knowledge distilled into a machine readable format that we aim to leverage for opinion extraction. How to harness the structured knowledge is dependent on the task at hand and the knowledge base itself. In this thesis, we consider two resources which we motivate in the following.

As we established in Section 2.2, word embeddings encode words in such a way that similar words tend to be close together in the vector space. Ultimately, this similarity of two words stems from an overlapping set of common context words since those play a crucial part in the computation of word embeddings. However, this encoding sometimes struggles with words of opposite polarity which, despite their difference in sentimental meaning, often share similar word embeddings since they appear in similar contexts [Tang et al., 2014; Yu et al., 2017]. This issue extends to antonyms in general. Querying the pretrained and publicly available word embeddings[1] of the Google News corpus by Mikolov et al. [2013a], we can see that seemingly opposite words appear close together: The nearest neighbor of the

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

word *small* in that embedding space is the word *large*. We address this issue using the lexico-semantic database WordNet [Fellbaum, 1998]. WordNet is a graph that organizes words according to their meanings, called *synsets*. The graph explicitly models various semantic relations between synsets such as hypernyms, hyponyms, meronyms, and many more. We argue that inducing this knowledge into word embeddings could be beneficial for the extraction of opinion targets or sentiment classification. To this end, we employ a technique called *retrofitting* [Faruqui et al., 2015] that post-processes pretrained embeddings in a way that reflects the structure of the used semantic lexicon. We give a more detailed description in Section 4.3.

While WordNet offers lexical knowledge, it does not contain explicit information about the sentiment of words or synsets. To some extent, sentiment and subjectivity are encoded in word embeddings but only implicitly. We can observe this by querying the closest neighbors for a particular sentiment-laden word such as *good*. Using the publicly available word embeddings from Mikolov et al. [2013a] as before, we obtain the 5 closest neighbors of the word *good*, namely *great*, *bad*, *terrific*, *decent*, and *nice*. From the given example, it is apparent that the subjectivity of words contributes to their measured similarity. However, we also observe (again) that words with opposite sentimental meaning are assessed as being similar which might hamper the performance of an ABSA system. Considering this, we expect that external, sentiment-focused information can be a helpful indicator for the automatic sentiment analysis of texts. We address this hypothesis using the common sense knowledge base SenticNet 3 [Cambria et al., 2014], a concept-level resource for sentiment analysis. SenticNet provides us a means to induce explicit sentimental knowledge into a model. We give a more detailed description of SenticNet 3 and our method of leveraging it in Section 4.4.

The value of knowledge bases as a source of features for document-level sentiment analysis has been explored in prior work [Schouten and Frasincar, 2015; Dragoni et al., 2014; Chung et al., 2014b]. Schouten and Frasincar [2015] enrich a feature representation of a document with bag-of-concept features that are extracted from WordNet. The features are used in conjunction with an SVM and evaluated for the tasks of sentence-level polarity classification and aspect category detection. A similar bag-of-concept approach for document-level sentiment classification is presented by Chung et al. [2014b]. Both works differ from our approach in that these approaches address document-level sentiment analysis and thus omit the positional information of the concept. The extracted sentiment information is thus not associated with an individual aspect but rather with the entire document. Aprosio et al. [2015] target opinion frame detection and combine a large feature set including SenticNet features with a CRF tagger and an SVM to extract opinion phrases, opinion targets, opinion holders and polarities. Toh and

Wang [2014] propose a CRF as a sequence labeler that includes (among others) features derived from WordNet taxonomies to address opinion target extraction. While WordNet-based features have been used in some ABSA applications, so far, retrofitted word embeddings have not been explored. Concept-level information as provided by SenticNet has been shown to be effective for document-level sentiment analysis but no research is available for such a benefit for target-specific sentiment analysis.

4.1.1 Contributions and Structure of the Chapter

In this chapter, we investigate in how far semantic resources such as WordNet and SenticNet support ABSA. We present an alternative approach for ABSA that follows a two-step framework. The complexity of ABSA is addressed by i) the extraction of opinion target phrases and ii) the assignment of a polarity label directly to each extracted opinion target. At the core of this chapter are two neural network models that target the two subtasks. The models are inspired by the findings of Chapter 3 and follow similar principles. In this chapter, we answer the main research question:

RQ2 *How can external, structured knowledge be incorporated into a model for ABSA?*

We specifically look at two types of structured knowledge and answer the related questions:

RQ2.1 *What is the benefit of lexical knowledge?*

RQ2.2 *What improvements can be expected when including semantic knowledge?*

The approach was originally developed in the context of the ESWC 2016 Challenge on Semantic Sentiment Analysis and was awarded as the best performing and most innovative sentiment analysis system among all submissions. Below, we list the key contributions of this chapter:

i) We propose an RNN-based model that is capable of classifying aspect-term-specific sentiment. The model follows the same principles established in Chapter 3.

ii) We show the effect of infusing lexical knowledge from WordNet into word embeddings for aspect extraction and targeted sentiment classification. We observe that retrofitted embeddings, contrary to our intuition, do not support the extraction of opinion targets.

iii) We assess SenticNet as a source for sentiment-related knowledge that aids ABSA. We derive word-level features from the knowledge graph and show that it is particularly helpful for target-specific sentiment classification. Our evaluation shows a substantial positive effect on training time and classification accuracy.

The remaining chapter is structured as follows: In the following Section 4.2, we present our overall setup and describe its two main components as well as the core features we employ. Section 4.3 introduces the concept of *retrofitting* word vectors to semantic lexicons. Here, we show how we incorporate knowledge form WordNet into our model. In Section 4.4, we give a brief introduction to SenticNet and describe the features we derive from it. The two steps of our approach and its extensions are evaluated in Section 4.5. We first assess the performance of the opinion target extraction in a cross-validation on the training data. Following, we evaluate the target-specific sentiment classification accordingly and show a benefit in leveraging knowledge from SenticNet. As a last step in our evaluation, we report results on the official test data of the sentiment analysis challenge. We conclude the chapter in Section 4.6 and give suggestions for future research.

4.2 A Two-Step Approach for Aspect-Based Sentiment Analysis

The previous chapter showed us that ABSA can be modeled in a relational framework which provides a fine-grained view of expressed opinions. In this chapter, we rephrase this formulation in favor of a reduced complexity by focusing on the identification of OTEs and the classification of sentiment polarity. In contrast to our earlier formulation, we omit the detection of explicit opinion phrases and opinion relations. The expressed sentiment is thus not attributed to a phrase in the text, but rather directly assigned to the opinion target expression. A visualization of this formulation could look like this:

Example:

> *"The* *food*^{pos} *and* *drinks*^{pos} *were tasty but the* *menu*^{neg} *was limited"*

We see that the *food* and *drinks* are perceived as positive while the *menu* is mentioned as a negative aspect of the experience.

We follow this two-step formulation and design a system that is capable of extracting an opinion holder's sentiment towards certain aspects of an entity. As a first step, given a text, the system extracts OTEs, i.e. aspect phrases that are

the target of an opinion. Secondly, each extracted opinion target expression is processed individually considering its context and a sentiment value is assigned. The following sections elaborate on our design and feature choices for our target phrase and sentiment extraction components.

4.2.1 Domain-Specific Word Embeddings

A central feature in our models are pretrained word embeddings which have been successfully employed in numerous NLP tasks [Collobert et al., 2011; dos Santos and Zadrozny, 2014; Le and Mikolov, 2014; Mikolov et al., 2013a; Pennington et al., 2014].

In the previous chapter, we highlighted the importance of domain-specific word embeddings for ABSA. By training a skip-gram model on domain-specific data, the resulting embeddings capture the semantics of each word for our targeted domain more closely than embeddings trained on domain-independent data. We evaluated this effect in Section 3.6.3 of Chapter 3. In this chapter, we use the same skip-gram word embeddings from earlier experiments but limit the vocabulary to the 100,000 most frequent words. The resulting vocabulary is denoted as V.

Word embeddings constitute an integral part of the input representation for our approaches. Throughout this chapter, we refer to the sequence of word embeddings for an input sentence as:

$$\mathbf{w} = (w_1, \ldots, w_n) \text{ with } w_i \in \mathbb{R}^{100}.$$

4.2.2 Part-of-Speech Tags

Apart from these word embeddings, we enrich the input representation by providing POS tags for each word in a sentence as we could show a positive impact of POS tags on the extraction of opinion target expressions in Chapter 3. When including POS tags, we employ a one-hot encoding that transforms each tag into a k-dimensional vector that represents this specific tag. Specifically, we use the Stanford POS Tagger [Manning et al., 2014] with a tag set of $k = 45$ tags. These vectors are then concatenated with their respective word vectors before being fed to the extraction components. The sequence of POS tag vectors for a sentence with words $1 \ldots n$ is denoted as:

$$\mathbf{p} = (p_1, \ldots, p_n) \text{ with } p_i \in \mathbb{R}^{45}.$$

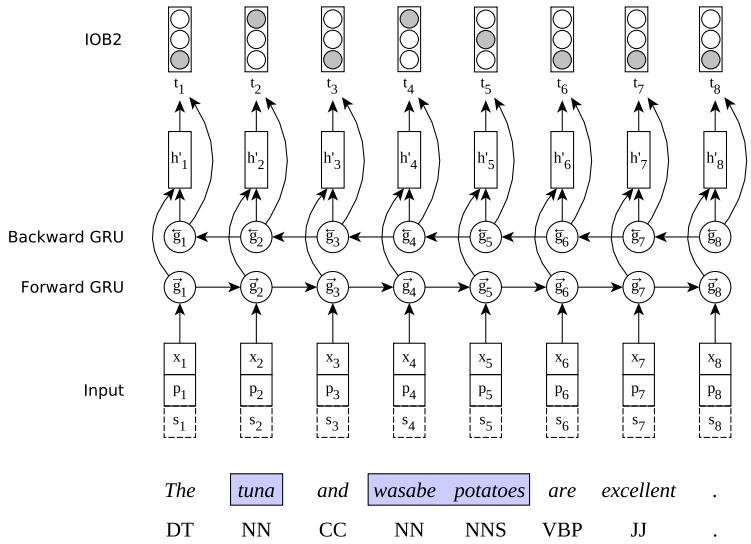


Figure 4.1: The opinion target extraction component. The network processes the input sentence as a sequence of word vectors w_i, sentic vectors s_i and POS tags p_i using a bidirectional GRU layer and feed-forward layers. The output of the network is a predicted tag sequence in the IOB2 format. The opinion targets that are to be predicted are marked in the input sentence.

4.2.3 Opinion Target Extraction

Our first step in extracting aspect-based sentiment from a text is the extraction of mentioned opinion targets. We follow the approach from Chapter 3 which allows us to extract an arbitrary number of opinion targets from a given text by framing the extraction as a sequence labeling problem. For this, we encode expressed opinion targets using the IOB2 tagging scheme [Tjong Kim Sang and Veenstra, 1999]. According to this scheme, each word in our text receives one of three tags, namely **I**, **O** or **B** that indicate if the word is at the **B**eginning, **I**nside or **O**utside of an annotation:

The	sake	menu	should	not	be	overlooked	!
O	**B**	**I**	O	O	O	O	O

This tagging scheme allows us to encode multiple non-overlapping opinion targets at once.

We design a neural network based sequence tagger that reads in a sequence of words and predicts a sequence of corresponding IOB2 tags that encode the detected opinion targets. Figure 4.1 depicts the neural network component. The procedure to generate a tag sequence for a given word sequence can be described as

follows: First, the sequence of words is mapped to a sequence of word embedding vectors $\mathbf{w} = (w_1, \ldots, w_n)$ and POS tag vectors $\mathbf{p} = (p_1, \ldots, p_n)$ using the resources described in Sections 4.2.1 and 4.2.2. We concatenate each word vector with its corresponding POS tag vector to receive the sequence:

$$\mathbf{x} = (x_1, \ldots, x_n) = (w_1 \oplus p_1, \ldots, w_n \oplus p_n) \text{ with } x_i \in \mathbb{R}^{100+45}. \tag{4.1}$$

where $\oplus$ denotes the concatenation of two vectors. The resulting sequence is passed to a bidirectional layer [Schuster and Paliwal, 1997] of Gated Recurrent Units (GRU, [Cho et al., 2014]) that produces an output sequence of recurrent hidden states:

$$\mathbf{g} = \textsc{BiGRU}(\mathbf{x}) = (g_1, \ldots, g_n) \text{ with } g_i \in \mathbb{R}^{50}, \tag{4.2}$$

using a combination of update and reset gates in each recurrent hidden unit. For a mathematical formulation of the GRU, we refer to Section 2.1.3 in Chapter 2. We implement the bidirectional GRU layer as two separate GRU layers. One layer processes the input sequence in a forward direction (left-to-right) while the other processes it in reversed order (right-to-left). The sequences of hidden states of each GRU layer are concatenated element wise in order to yield a single sequence of hidden states:

$$\mathbf{g} = (\overrightarrow{g}_1 \oplus \overleftarrow{g}_1, \ldots, \overrightarrow{g}_n \oplus \overleftarrow{g}_n) \text{ with } \overrightarrow{g}_i, \overleftarrow{g}_i \in \mathbb{R}^{25}, \tag{4.3}$$

where $\overrightarrow{g}_i$ and $\overleftarrow{g}_i$ are the output vectors for the forward and backward GRU layer, respectively. Each vector g_i is passed to a regular feed-forward layer that produces a transformed representation $h'_i \in \mathbb{R}^{50}$ for that vector. Lastly, a final layer in the network projects each h'_i of the previous layer to a probability distribution q_i over all possible output tags, namely I, O or B, using a softmax activation function:

$$q_i = softmax(W_{tag} h'_i + b_{tag}). \tag{4.4}$$

For each word, we choose the tag with the highest probability as its predicted IOB2 tag. Since the prediction of each tag can be interpreted as a classification, the network is trained to minimize the categorical cross-entropy between expected tag distribution p_i and predicted tag distribution q_i of each word i:

$$\mathcal{L}(p_i, q_i) = -\sum_{t \in \mathcal{T}} p_i^t \log(q_i^t), \tag{4.5}$$

where $\mathcal{T} = \{I, O, B\}$ is the set of IOB2 tags, $p_i^t \in \{0, 1\}$ is the expected probability of tag t and $q^t \in [0, 1]$ the predicted probability. The network's parameters are

optimized using the stochastic optimization technique *Adam* [Kingma and Ba, 2014].

For further processing, a predicted tag sequence can be decoded into opinion target annotations using the IOB2 scheme. Note that we do not enforce the syntactic correctness of the predicted IOB2 scheme on a network-level. It is possible that the network produces a tag sequence that is not correct in terms of the employed IOB2 scheme. Thus, we post process each predicted tag sequence such that it constitutes a valid IOB2 tag sequence. Specifically, we replace each I tag that follows an O tag with a B in order to properly mark the beginning of an opinion target.

4.2.4 Target-Specific Sentiment Classification

The second step in our two-step architecture for aspect-based sentiment analysis is the prediction of a polarity label given a previously detected opinion target. We adopt our approach from the previous chapter and employ a position-aware recurrent neural network for the targeted classification. In contrast to the previous chapter which was concerned with the classification of the sentiment label for an opinion phrase, we are now using the position-aware RNN to predict the sentiment label for an opinion *target* directly. Let us briefly recap the position-aware RNN: In order to predict a polarity label for a *specific* opinion target in a sentence, we need to mark the opinion target in question. We tag each word in the input sentence with its relative distance to the opinion target, as follows:

Great	*service*	,	*great*	*food*	.	(Text)
-1	0	1	2	3	4	(Distance)

where the word *service* is the selected opinion target phrase that is to be classified. The word *food* marks another opinion target that is processed separately in a later step. The relative distance to the selected opinion target is shown below each word. This sequence of relative distances encodes the position of the opinion target in question in the sentence. We do not use the raw distance values directly but represent them as 10 dimensional distance embedding vectors similar as in [dos Santos et al., 2015b; Zeng et al., 2014; Sun et al., 2015] and treat them as learnable parameters in our network. We further denote the sequence of distance embedding vectors for a sentence of n words as:

$$\mathbf{d} = (d_1, \ldots, d_n) \text{ with } d_i \in \mathbb{R}^{10}. \tag{4.6}$$

Figure 4.2 depicts the neural network component.

Given the position-aware RNN, the procedure for predicting a polarity label for an opinion target can be described as follows: Assume we have a sentence

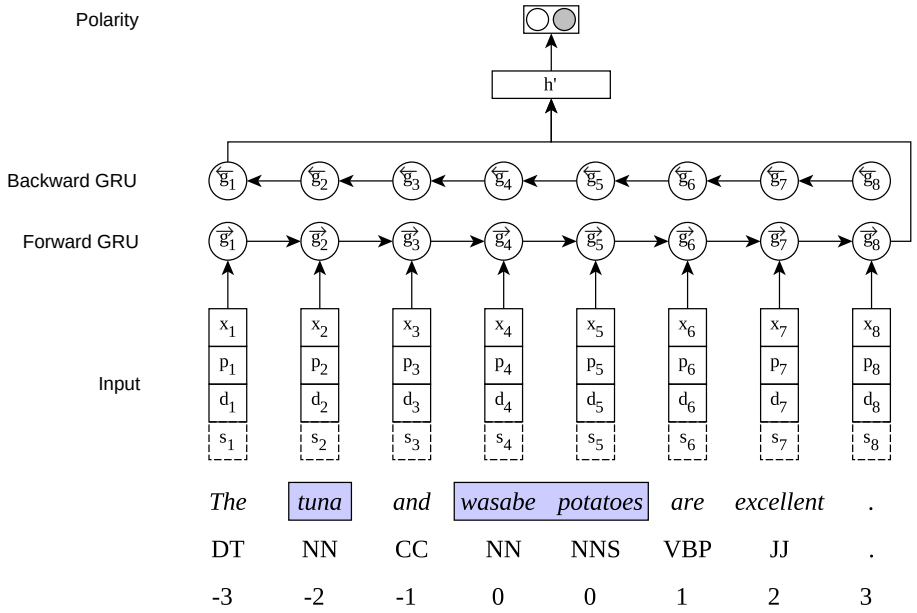


Figure 4.2: The target-specific sentiment classification component. The network processes the input sentence as a sequence of word vectors w_i, sentic vectors s_i, POS tags p_i and distance embeddings d_i using a bidirectional GRU layer and regular feed-forward layers. The output of the network is a single predicted polarity label for the opinion target of interest. The opinion target for which a polarity label is to be predicted is encoded in the input sentence with relative distance values.

and an already extracted opinion target. We concatenate each word vector with its corresponding POS tag vector and distance vector to receive the input feature sequence:

$$\mathbf{x} = (w_1 \oplus p_1 \oplus d_1, \ldots, w_n \oplus p_n \oplus d_n) \text{ with } x_i \in \mathbb{R}^{100+45+10}. \tag{4.7}$$

The resulting sequence is passed to a bidirectional GRU layer that produces an output sequence of recurrent states:

$$\mathbf{g} = \text{BiGRU}(\mathbf{x}) = (\overrightarrow{g}_1 \oplus \overleftarrow{g}_1, \ldots, \overrightarrow{g}_n \oplus \overleftarrow{g}_n) \text{ with } \overrightarrow{g}_i, \overleftarrow{g}_i \in \mathbb{R}^{25}. \tag{4.8}$$

We take the final output vector $\overrightarrow{g}_n$ of the forward GRU and the final output vector $\overleftarrow{g}_1$ of the backward GRU[2] and concatenate them to receive a fixed sized representation $h = (\overrightarrow{g}_n \oplus \overleftarrow{g}_1) \in \mathbb{R}^{50}$ of the opinion target in the whole input sentence. We denote h the *summary* of the target and its context. Next, the network passes

[2]Since this GRU processes the sequence in a reversed direction, the final output vector is the output vector for the first word.

the summary h through a densely connected feed-forward layer producing another hidden representation $h' \in \mathbb{R}^{50}$. As a last step, a final densely connected layer with a softmax activation function projects h' to a 2-dimensional vector $q \in \mathbb{R}^2$ representing a probability distribution over the two polarity labels `positive` and `negative`. We consider the label with the highest estimated probability to be the predicted polarity label for the given opinion target.

Again, we train the network to minimize the categorical cross-entropy between expected polarity label distribution p and predicted polarity label distribution q of each opinion target:

$$\mathcal{L}(p, q) = -\sum_{l \in \mathcal{P}} p^l \log(q^l), \tag{4.9}$$

where $\mathcal{P} = \{\texttt{positive}, \texttt{negative}\}$ is the set of polarity labels and p^l and q^l the expected and predicted probability, respectively, for label l. As before, we use the *Adam* optimizer to update network parameters.

4.3 Retrofitting Word Embeddings to WordNet

Although word embeddings have been shown to encode semantic and syntactic features of their respective words well [Mikolov et al., 2013b; dos Santos and Zadrozny, 2014; Mikolov et al., 2013a], we try to enhance their encoded semantic information by using an external lexical resource. For this, we employ a technique called *retrofitting* [Faruqui et al., 2015]. The idea behind retrofitting is to iteratively adapt precomputed word vectors to better fit the relations modeled in a given knowledge graph. The graph-based algorithm gradually "moves" each word vector towards the word vectors of its neighboring nodes while still staying close to its original position.

Following the notation of Faruqui et al. [2015], let $V = \{v_1, \ldots, v_{N_V}\}$ be the considered vocabulary and $W = (w_1, \ldots, w_{|V|})$ with $w_i \in \mathbb{R}^d$ the corresponding precomputed word vectors. $G = (V, E)$ is the graph of semantic relationships to which we want to fit the word vectors with $(v_i, v_j) \in E \subseteq V \times V$ denoting the edges between words. With $\hat{W} = (\hat{w}_1, \ldots, \hat{w}_{|V|})$ being the fitted word vectors, the algorithm tries to minimize the following objective function:

$$\Psi(W, \hat{W}) = \sum_{i=1}^{|V|} \left[\alpha_i ||\hat{w}_i - w_i||^2 + \sum_{(i,j) \in E} \beta_{ij} ||\hat{w}_i - \hat{w}_j||^2 \right] \tag{4.10}$$

cold		*warm*		*small*	
original	retrofit	original	retrofit	original	retrofit
frigid	*colder*	*toasty*	*warms*	*large*	*large*
colder	*coldest*	*warmer*	*warmed*	*big*	*smaller*
warm	*frigid*	*chilly*	*warmer*	*tiny*	*smallest*
coldest	*coldness*	*cold*	*warmest*	*smallish*	*larger*
winter	*colds*	*warms*	*warmth*	*larger*	*big*
big		*slow*		*fast*	
original	retrofit	original	retrofit	original	retrofit
huge	*bigger*	*slower*	*slower*	*quick*	*quick*
small	*biggest*	*snail's*	*slows*	*quickly*	*faster*
bigger	*small*	*slows*	*slowed*	*superfast*	*fasted*
too	*large*	*sloooow*	*sluggish*	*speedy*	*quickly*
large	*little*	*fast*	*slowest*	*super-fast*	*speedy*

Table 4.1: The 5 closest nearest neighbors with respect to the cosine similarity of words in the original embedding space and the retrofit embedding space. We underline words that we judge as antonymic to the query word.

The online update rule for each w_i is then:

$$\hat{w}_i = \frac{\sum_{j:(i,j)\in E} \beta_{ij}\hat{w}_j + \alpha_i w_i}{\sum_{j:(i,j)\in E} \beta_{ij} + \alpha_i} \tag{4.11}$$

where α and β are parameters of the retrofitting procedure.

In this work, we chose WordNet [Fellbaum, 1998] as our lexico-semantic resource. We construct a subgraph of the WordNet relations that links each word in our vocabulary to all its synonyms (lemma names) in the WordNet graph. We set all $\alpha_i = 1$ and all $\beta_{ij} = 1/degree(i)$ and run the retrofitting algorithm for 10 iterations as suggested by Faruqui et al. [2015]. The resulting embeddings are still very similar to their original embeddings yet incorporate part of the semantics of WordNet. Table 4.1 shows the nearest neighbors for a selected set of words before and after the retrofitting process. When using retrofitted word embeddings in our models, we modify Eqs. (4.1) and (4.7) by replacing the plain word embeddings $\mathbf{w}$ with retrofitted embeddings $\hat{\mathbf{w}}$. The subsequent equations remain unchanged. We investigate the benefit of using these retrofitted word embeddings in comparison to their original counterparts in Section 4.5.

4.4 Extracting Word-Level Features from SenticNet

SenticNet 3 [Cambria et al., 2014] is a graph-based, concept-level resource for semantic and affective knowledge. Among other common- and common sense knowledge entries and relations, SenticNet 3 comprises entries for adjectives such as *good* or *horrible* that convey an unambiguous sentiment regardless of the targeted noun. Beyond that, ambiguous adjectives such as *small* or *cold* assume a polarity only in conjunction with the targeted noun. This can be seen in examples such as *cold beer* and *cold food* where the adjective *cold* acts as a positive and negative modifier, respectively. SenticNet 3 addresses these cases by storing them as multi-word concepts that convey an unambiguous sentiment only in conjunction. For each of the 30,000 concepts that are part of the knowledge graph, SenticNet 3 provides real-valued scores for 5, so called, sentics: *pleasantness*, *attention*, *sensitivity*, *aptitude*, and *polarity*. These sentics encode information about the semantics and polarity of a concept. Making this knowledge accessible to a classification model for target-specific sentiment could improve the models accuracy when compared to a baseline that has no explicit knowledge about sentiment. In order to leverage the stored knowledge in SenticNet, we need to detect expressed concepts in a given input text. Moreover, we require that the obtained concepts are localized in the text, such that we can attribute the obtained sentiment information to certain words. The positional information enables our position-aware RNN to relate the expressed sentiment to specific target aspects mentioned in the text instead of relating it to the document as a whole.

While Cambria et al. [2014] provide a concept parser for SenticNet 3, the available implementation exposes the detected concepts as a bag-of-concepts that is oblivious to their locations in the text. Since the modification of the automatic concept parser is out-of-scope of this work, we follow a simple approach and omit all ambiguous multi-word concepts such as `notice_problem` or `beautiful_music`. Instead, we only consider unambiguous single-word concepts such as `experience` or `improvement` that we can easily detect and pinpoint to a position in a text. Doing this, we can inject the real-valued sentics directly at the word-level.

For each word w_i in a text, we perform a lookup in SenticNet 3 and obtain the associated vector of sentics:

$$\begin{aligned} s_i &= Sentics(w_i) \\ &= (pleasantness_{w_i}, attention_{w_i}, sensitivity_{w_i}, aptitude_{w_i}, polarity_{w_i}) \end{aligned}$$

If we cannot find a concept for a word w_j, we define $s_j = (0, 0, 0, 0, 0)$. We extend the baseline models by concatenating the sentic vectors with the other word-level

Dataset	#Sentences	#Tokens	#Opinions	#Targets
Laptops (train)	3048	51840	1864	1864
Restaurants (train)	2000	29278	2406	1808

Table 4.2: An overview of the ESWC 2016 ABSA dataset.

features in Eqs. (4.1) and (4.7). The remaining processing steps for the opinion target extraction with the position-aware RNN remain unchanged.

4.5 Experimental Evaluation and Discussion

In this section, we assess the proposed lexically and semantically enriched models in comparison to our baseline models. We perform the evaluation of the two subtasks in isolation to obtain a clearer understanding of the observed effects.

4.5.1 Dataset

We carry out the evaluation of the proposed models on the ABSA dataset of the ESWC 2016 Challenge on Semantic Sentiment Analysis. For every sentence of a review, the dataset provides annotations for opinion target expressions specified as character offsets and their respective sentiment labels which can either be `positive` or `negative`. The organizers of the challenge provide data for the domains *restaurants* and *laptops* which we combine into a single dataset for our experiments. An overview of the dataset is given in Table 4.2.

4.5.2 Opinion Target Extraction

We evaluate opinion target extraction using precision, recall, and F_1-score. Table 4.3 shows the results for opinion target extraction for different feature combinations. Here, `WE` denotes the usage of regular word embeddings, `WE-Retro` denotes the retrofitted embeddings, `POS` specifies additional POS tag features and `Sentics` indicates the usage of sentic vectors. Comparing the models in Table 4.3, we can see that including either type of structured knowledge surprisingly degrades the networks performance. Both the retrofitted embeddings and the sentic vectors negatively affect the F_1-score to a similarly small extent. The only small increase with respect to the measured precision can be observed for the model using sentic vectors. Overall, the extraction of OTEs does not seem to profit from either type of structured semantic knowledge as it is induced here.

Features	F_1	Precision	Recall
WE + POS	**0.684**	0.659	**0.710**
WE + POS + Sentics	0.679	**0.663**	0.697
WE-Retro + POS	0.678	0.651	0.708
WE-Retro + POS + Sentics	0.679	0.655	0.706

Table 4.3: Results of 5-fold cross-validation for opinion target extraction using different feature combinations.

Features	Accuracy
WE + POS + Dist	0.776
WE + POS + Dist + Sentics	**0.811**
WE-Retro + POS + Dist	0.776
WE-Retro + POS + Dist + Sentics	0.809

Table 4.4: Results of 5-fold cross-validation for target-specific sentiment extraction using different feature combinations.

4.5.3 Target-Specific Sentiment Classification

To evaluate the target-specific sentiment classification, we extract polarity labels for all opinion targets of the ground truth annotations. By separating opinion target extraction and sentiment extraction, we can better evaluate the sentiment extraction in isolation. Again, we only consider unique opinion targets that are either labeled with a `positive` or `negative` polarity. We report the performance of our sentiment extraction in terms of the accuracy of the system for different feature combinations. Table 4.4 shows the results for the 5-fold cross-validation on the training data. `WE`, `WE-Retro`, `POS` and `Sentics` are defined as before, while `Dist` denotes the obligatory distance embedding features of the position-aware RNN. While the retrofitted embeddings do not contribute positively to the performance for sentiment extraction either, a notable gain is achieved using the sentic vectors in our component for target-specific sentiment extraction. Here, we observe a gain of 3.5 points accuracy compared to using only word embeddings, distance embeddings and POS tags. Apart from that, the usage of sentic vectors drastically reduces the training time needed to achieve these results. The best results for the `WE + POS + Dist` and `WE-Retro + POS + Dist` model were achieved with 102 iterations over the training portion of the data, while the `WE + POS + Dist + Sentic` and `WE-Retro + POS + Dist + Sentic` model reached their best performances for only 12 and 9 iterations, respectively. See Figure 4.3 for a visualization of the system's accuracy with respect to the employed features and the iteration

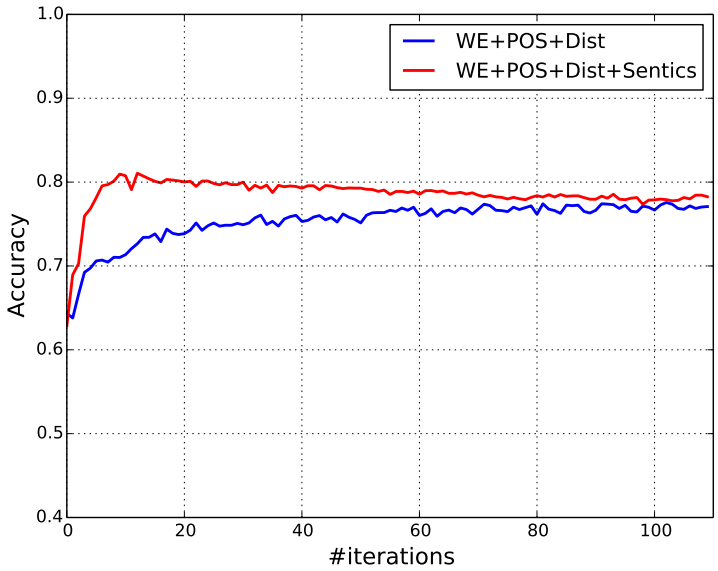


Figure 4.3: Visualization of the performance gain of using sentic vectors with respect to the number of iterations over the training data. By using additional sentic vectors we achieve better results with less training needed.

over the training data.

4.6 Conclusion

In this chapter, we empirically evaluated the benefit of introducing structured, semantic knowledge into a two-step approach for ABSA. We addressed the extraction of opinion target expressions and corresponding sentiment labels using neural network architectures that were inspired by our findings from earlier chapters. We considered the two knowledge graphs WordNet and SenticNet as sources for semantic knowledge that have the potential to support ABSA. For our approach to leverage WordNet, we experiment with a technique called *retrofitting*. This technique iteratively adapts pretrained word embeddings such that the word similarities in the embedding space correspond more strongly to the given structure of a knowledge graph such as WordNet. As a means to induce affective knowledge into our baseline models, we make use of SenticNet. SenticNet provides real-valued scores for thousands of concepts that we use as additional word-level features in our models. Contrary to our expectation, the retrofitted embeddings neither support the extraction of opinion target expression, nor the classification of target-specific sentiment. Here, a baseline model with skip-gram embeddings and

POS tag features alone obtains better results. On the other hand, regarding the classification of target-specific sentiment using a position-aware RNN, the sentics obtained from SenticNet prove to be a valuable feature that increase accuracy and shorten training time considerably.

Chapter 5

Subword Information for Opinion Target Extraction

Chapter Overview *In this chapter, we investigate whether character-level models can improve the performance of the identification of opinion target expressions. We integrate information about the character structure of a word into a sequence labeling system using character-level word embeddings and show their positive impact on the system's performance. In further experiments, we reveal encoded character patterns of the learned embeddings and take a closer look at the performance differences as compared to a baseline system.*

5.1 Introduction

A key task within aspect-based sentiment analysis consists of identifying Opinion Target Expressions (OTEs). OTEs are segments of a text that specify the target of an opinion and, therefore, form a crucial part of understanding an opinion. A particular challenge involved in OTE identification in online reviews stems from the fact that these reviews can be of low quality or feature an informal writing style. This results in a couple of phenomena which can only be handled to a limited degree by models based on classic word embeddings. For example, reviews often contain misspelled words:

Example:

> *"The* *pizza* *is yummy and I like the* *atmoshpere* *."*

These can affect the quality of an OTE extraction model that relies on a fixed-sized vocabulary of words as is often the case when using precomputed word embeddings. A similar effect is presented by the elongation of words. Opinions expressed in

informal texts are often emphasized by repeating certain characters, especially in phrases indicating the sentiment itself:

Example:

> *"i love their chicken pasta cant remember the name but is sooo good"*
>
> *"[...] service was so slowwwww [...]"*

Due to the high variability of possible surface forms, a fixed-sized vocabulary might not contain all relevant words for accurately detecting sentiment and targets. We can observe another form of emphasis in which single words or entire phrases are capitalized to inform the reader about the importance of this phrase:

Example:

> *"The prices were CHEAP compared to the quality of service and food."*
>
> *"The signs, the specials menus, food, and even all the waitstaff are ALL TOTALLY Japanese."*
>
> *"I LOOOVE their eggplant pizza, as well as their pastas!"*

While it is possible to normalize the texts by e.g. lowercasing the entire text, this would discard useful information for the understanding of the opinion.

The aforementioned issues are particularly problematic for models that rely on classic word embeddings that have a fixed vocabulary such as Continuous Skip-Gram (SG), Continuous Bag-of-Words (CBOW) or GloVe embeddings [Pennington et al., 2014]. These embeddings are usually pretrained on a separate, unlabeled dataset and can only represent words that were encountered in that pretraining phase. A notable exception to this are fastText embeddings [Joulin et al., 2017; Bojanowski et al., 2017] that extend the ideas of SG and CBOW to include embeddings for character n-grams.

A further challenge is that opinion target expressions can span multiple tokens. As we show in this chapter, extracting multi-word expressions is difficult and much harder than detecting single-word targets. This phenomenon is aggravated by variations in spelling and hyphenation that represent identical concepts as either single words or multiple words, which ultimately affects the representation of the phrase in the model. As an example, compare the spelling variations of the target concept *waitstaff*:

Example:

"the waitstaffs are nice though."

"[...] absurdly arrogant wait-staff who don't recognize they work at a glorified diner [...]"

"Cute place, nice wait staff but would never go there again."

or the usage of a hyphenated compound phrase:

Example:

"Wine list selection is good and wine-by-the-glass was generously filled to the top."

Improper tokenization might lead to subpar word and phrase representations which, in turn, affects the extraction of target phrases.

Considering these issues, we hypothesize that including subword information is beneficial in the context of OTE extraction, allowing a model to be robust to spelling variations as well as to generalize to unseen words and multi-word expressions. In the following, we point out notable works for OTE extraction that leverage subword information to some extent. Beyond that, we give a brief overview of recent research on character-level NLP in general.

Subword Information for Opinion Target Extraction San Vicente et al. [2015] present a system that addresses opinion target extraction as a sequence labeling problem based on a perceptron algorithm with local features. Besides word clusters and context features, they make use of prefix and suffix features to inform the system about the morphology of the words. Kumar et al. [2016] propose a CRF-based model using an extensive collection of lexical and semantic features. Similar to San Vicente et al. [2015], the model includes prefix and suffix features but also character n-grams for every word. Hercig et al. [2016] also consider character n-grams in the context of ABSA for aspect-polarity and aspect-category classification. However, they do not use these for the extraction of aspect target phrases. All approaches include subword information in the form of discrete n-grams into the respective machine learning models, yet do not model the word structure as a whole.

Character-Level Natural Language Processing Character-level neural network models are gaining interest in many research areas such as language modeling [Kim et al., 2016], spelling correction [Sakaguchi et al., 2017], text classification [Zhang et al., 2015] and more. Most similar works from the area of character-level

word representations can be found in [dos Santos and Zadrozny, 2014; dos Santos et al., 2015a; Ma and Hovy, 2016]. In these works, word and character level representations are successfully learned and combined to improve POS tagging and NER. dos Santos and Zadrozny [2014] and dos Santos et al. [2015a] apply a CNN to the raw character sequence that detects character patterns and represents them as a fixed-sized embedding vector. The concatenated sequence of word and character-level embeddings is then used to predict POS or NER tags for each word. Ma and Hovy [2016] use a similar CNN-based word structure model. However, the subsequent processing of the embedded word sequence is carried out using a bidirectional LSTM network.

An example of character-level text classification not requiring any tokenization is given by Zhang et al. [2015]. In their work, the authors perform text classification using character-level CNNs on very large datasets and obtain comparable results to traditional models based on words. Their findings suggest that the standard tokenization of text is indeed something to be reconsidered.

In a recent work, Akbik et al. [2018] present a character-level LSTM that is pretrained on a language modeling task. They use the internal LSTM states to obtain contextualized character-level word embeddings for a sentence which are then passed to a subsequent sequence labeling model for NER. Their approach using the character-level language-model embeddings achieved state-of-the-art results at that time.

Overall, an increasing research interest in subword-informed models can be observed. In the case of OTE extraction, however, character-level features have been limited to n-gram indicator features. So far, the research of end-to-end character-based models has not been extended to ABSA and the extraction of OTEs.

5.1.1 Contributions and Structure of the Chapter

User-generated texts pose a series of challenges for opinion target extraction. Motivated by these challenges, we investigate whether a character-based approach for the extraction of OTEs is capable of using the additional low-level information to improve upon a standard word-based baseline. We hypothesize that character-level word embeddings capture relevant information for OTE extraction that regular (skip-gram) word embeddings lack. In the course of this chapter, we answer the following research question:

RQ3 *How can we alleviate the difficulty of opinion target extraction for noisy textual data?*

We particularly focus on the following facets of this question:

RQ3.1 *How can a model learn relevant morphological information?*

RQ3.2 *Which challenges are alleviated with subword information?*

In answering these questions, our contributions are as follows:

i) We propose a neural network model that learns and utilizes character-level word embeddings to extract opinion target expressions. To our knowledge, this is the first model for OTE extraction to learn character-level word embeddings in an end-to-end fashion.

ii) We perform an experimental analysis and show that with an increase of 3.3 points F_1-score compared to a word-only baseline, the character information is indeed valuable for the task.

iii) We shed light on the learned properties of character-level word embeddings and show that they encode useful suffix information.

iv) We narrow down the observed performance improvement in focused experiments. We attribute the improvement to a more accurate detection of multi-word expression.

The rest of the chapter is structured as follows: In Section 5.2, we describe our baseline approach to address opinion target extraction. The model follows a simple sequence labeling architecture which we already investigated in previous chapters. Section 5.3 introduces the extension to the baseline model that we use to measure the impact of character information on the task. The extended model is equipped with an RNN-based component that processes the character sequence of each word and encodes it in a single vector. The character-informed word representation is included in the tagging model and trained in an end-to-end fashion alongside other parts of the model. We carry out our evaluation in Section 5.4 and examine the learned character-level word embeddings in more detail. Here, we describe our experimental settings and outline our hyperparameter optimization. In Section 5.4.3, we report results for a cross-validation on the training data as well as results for an official held-out test set. To obtain more insights into the inner workings of the proposed model we perform further, more focused experiments: We probe the learned character-level word embeddings in Section 5.4.4 and reveal that they encode suffix information. We hypothesize that this additional information is particularly helpful for OTE extraction. Furthermore, in Sections 5.4.5 and 5.4.6, we explore the performance differences on various subsets of the data and find that the proposed extension is particularly helpful in identifying multi-word expressions but does not alleviate problems associated with out-of-vocabulary terms. Finally, Section 5.5 summarizes our findings and presents directions for future work.

5.2 Baseline Model

We approach the task of extracting opinion target expressions by phrasing it as a sequence labeling problem. As we established in previous chapters, OTEs can be extracted using a neural network model in combination with a span encoding scheme such as the **IOB** scheme [Tjong Kim Sang and Veenstra, 1999]. We reiterate that, according to the **IOB** tagging scheme, each word in our text receives one of three tags, namely **I**, **O** or **B** that indicate if the word is at the **B**eginning, **I**nside or **O**utside of an expression: Doing so allows us to extract an arbitrary number of multi-word expressions in a given text.

I	*LOOOVE*	*their*	*eggplant*	*pizza*	*,*	*as*	*well*	*as*	*their*	*pastas*	*!*
O	O	O	**I**	**I**	O	O	O	O	O	**I**	O

The task is thus reduced to mapping a sequence of words to a sequence of tags. In this chapter, we model the sequence labeling task using RNNs which allow us to integrate character-level knowledge into the model in the form of character-level word embeddings. To quantify the impact of these embeddings, we compare a baseline model that only uses word-level embeddings to an extended model that learns character-level word embeddings in an end-to-end fashion. Let us formally describe the components of our model.

Gated Recurrent Unit The core of our sequence labeling model is the GRU. The GRU uses a combination of update and reset gates to improve its ability to learn long range information comparable to Long Short-Term Memory cells [Chung et al., 2014a]. We reiterate its general mathematical formulation for a single timestep i:

$$r_i = \sigma(\mathbf{W}^r x_i + \mathbf{U}^r g_{i-1} + b^r) \tag{5.1}$$
$$z_i = \sigma(\mathbf{W}^z x_i + \mathbf{U}^z g_{i-1} + b^z) \tag{5.2}$$
$$h_i = f(\mathbf{W}^h x_i + \mathbf{U}^h (r_i \odot g_{i-1}) + b^h) \tag{5.3}$$
$$g_i = (1 - z_i) \odot g_{i-1} + z_i \odot h_i \tag{5.4}$$

where x_i is an element of an input sequence and g_i the computed output at timestep i. z_i is the update gate and r_i the forget gate, σ is the sigmoid activation function and f the ELU [Clevert et al., 2016] function. The operator $\odot$ denotes the element-wise multiplication. For simplicity, we denote the application of Eqs. (5.1) – (5.4) as

$$\text{GRU}(\mathbf{x}) = \mathbf{g}$$

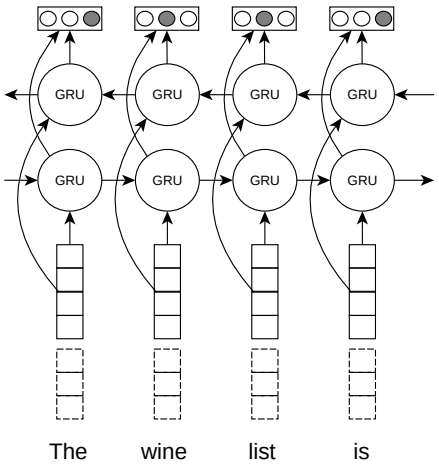


Figure 5.1: Illustration of the RNN sequence labeling model. The dashed boxes represent the character-level word embeddings that are only present in the character-enhanced model.

For an explanation of the individual equations of the GRU, we refer to Section 2.1.3 in Chapter 2.

Bidirectional GRU The Bidirectional Gated Recurrent Unit (BiGRU) is a variant of the GRU that processes the input sequence in forward and backward direction. The individual hidden states of the forward GRU and the backward GRU are concatenated to produce a single hidden state sequence:

$$\text{BiGRU}(\mathbf{x}) = \overleftrightarrow{\mathbf{g}} = (\overrightarrow{g}_1 \oplus \overleftarrow{g}_1, \ldots, \overrightarrow{g}_n \oplus \overleftarrow{g}_n), \tag{5.5}$$

where $\overrightarrow{g}_i$ and $\overleftarrow{g}_i$ are the output vectors for the forward and backward GRU layer, respectively, and $\oplus$ denotes the vector concatenation. The bidirectional connections allow the model to include words appearing before and after each timestep into the computation of the hidden states. The resulting sequence of hidden states $\overleftrightarrow{\mathbf{g}}$ incorporates the context for each word in its corresponding hidden state.

We combine the above concepts in a recurrent neural architecture for OTE extraction. The proposed baseline model is a BiGRU that receives a word sequence $\mathbf{w} = (w_1, \ldots, w_n)$ as input features and predicts an output sequence of IOB tags $\mathbf{t} = (t_1, \ldots, t_n)$. Figure 5.1 illustrates the network. Formally, the word sequence is passed to a word embedding layer that maps each word w_i to its d^{wrd}-dimensional embedding vector x_i^{wrd} by means of an embedding matrix $\mathbf{W}^{wrd} \in \mathbb{R}^{d^{wrd} \times |V^{wrd}|}$:

$$x_i^{wrd} = \mathbf{W}^{wrd} e^{w_i} \tag{5.6}$$

where V^{wrd} is the vocabulary of the word embeddings and e^{w_i} is a one-hot vector of size $|V^{wrd}|$ representing the word w_i. The representation of words x_i^{wrd} is processed by a BiGRU layer that computes a contextualized representation for each word:

$$\mathbf{g}^{wrd} = BiGRU(x_i^{wrd}) \tag{5.7}$$

In a last step, each hidden state g_i^{wrd} is projected to a probability distribution q_i over all possible output tags, namely **I**, **O** and **B**, using a standard feed-forward layer with a softmax activation function:

$$q_i = softmax(\mathbf{W}^{tag} g_i^{wrd} + b^{tag}) \tag{5.8}$$

with $\mathbf{W}^{tag} \in \mathbb{R}^{d^{tag} \times r^{wrd}}$ and $b^{tag} \in \mathbb{R}^{d^{tag}}$. For each word, we choose the tag with the highest probability as the predicted IOB tag. In all experiments, we fix the dimensionality of the parameters of the word-level GRU layers such that $\overrightarrow{g}_i, \overleftarrow{g}_i \in \mathbb{R}^{r^{wrd}/2}$, where r^{wrd} is a hyperparameter of the model.

5.3 Character-Enhanced Model

We propose a variation of the aforementioned baseline model that incorporates character-level information in the process of opinion target extraction. Our goal is to confirm the hypothesis that character information poses a valuable source of information for the task of opinion target extraction. Following previous work in this direction, we incorporate the character information in the form of character-level word embeddings [Ma and Hovy, 2016]. These embeddings are computed dynamically for each word in an input sequence using an RNN sub-model. The RNN processes each word as a sequence of characters. Every character is represented as an embedding vector and the entire character sequence is encoded as a single summary vector for each word. The resulting vector is incorporated in the text-level model to support the sequence tagging procedure. Figure 5.2 illustrates the character-level word model.

More formally, the character-enhanced model can be described as follows: Given the character sequence $\mathbf{c} = (c_1, \ldots, c_m)$ of a word w, we first transform each character c_i to its corresponding d^{chr}-dimensional character embedding x_i^{chr} using a character embedding matrix $W^{chr} \in \mathbb{R}^{d^{chr} \times |V^{chr}|}$:

$$x_i^{chr} = \mathbf{W}^{chr} e^{c_i}. \tag{5.9}$$

Analogously to the procedure for word embeddings, V^{chr} is the character vocabulary and e^{c_i} is a one-hot vector of size $|V^{chr}|$ representing the character c_i. As before, the sequence of character embeddings is passed through a bidirectional

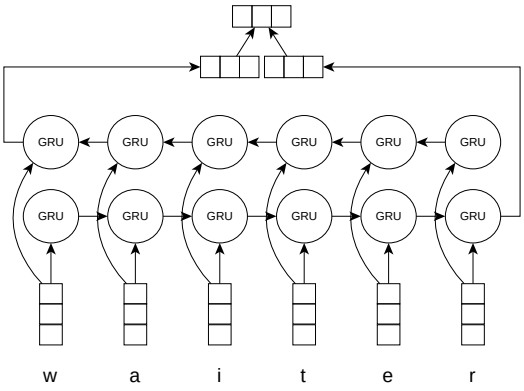


Figure 5.2: Illustration of the RNN character-level word embedding model. The output of this sub network is later concatenated with the regular word embeddings.

GRU layer that produces two sequences of hidden states, $\overrightarrow{\mathbf{g}}^{chr}$ and $\overleftarrow{\mathbf{g}}^{chr}$:

$$\overrightarrow{\mathbf{g}}^{chr} = \overrightarrow{\mathrm{GRU}}(\mathbf{x}^{chr}), \tag{5.10}$$

$$\overleftarrow{\mathbf{g}}^{chr} = \overleftarrow{\mathrm{GRU}}(\mathbf{x}^{chr}). \tag{5.11}$$

We choose the dimensionality of the parameters such that $\overrightarrow{g}_i^{chr}, \overleftarrow{g}_i^{chr} \in \mathbb{R}^{d^{chr}}$. To represent the sequence of characters as a fixed-sized summary vector, we concatenate the final hidden states of both sequences and obtain a single summary representation $g^{chr} = \overrightarrow{g}_m^{chr} \oplus \overleftarrow{g}_1^{chr}$ for the character sequence. Lastly, the concatenated hidden state g^{chr} is transformed to the final character-level word embedding using a linear feed-forward layer:

$$x^{cw} = \mathbf{W}^{cw} g^{chr} + b^{cw} \tag{5.12}$$

with $\mathbf{W}^{cw} \in \mathbb{R}^{d^{chr} \times 2 \cdot d^{chr}}$ and $b^{cw} \in \mathbb{R}^{d^{chr}}$.

To incorporate the word model in the overall neural network model, we pass the corresponding character sequence of each word in $\mathbf{w} = (w_1, \ldots, w_n)$ through the character model to obtain $\mathbf{x}^{cw} = (x_1^{cw}, \ldots, x_n^{cw})$. The resulting character-level embeddings are then concatenated with the word-level embeddings:

$$\widetilde{\mathbf{x}} = (x_1^{wrd} \oplus x_1^{cw}, \ldots, x_n^{wrd} \oplus x_n^{cw}) \tag{5.13}$$

The augmented sequence $\widetilde{\mathbf{x}}$ replaces $\mathbf{x}^{wrd}$ in Eq. (5.7) and subsequent equations of the baseline model and is passed through the remaining layers of the network. Since $\widetilde{\mathbf{x}}$ contains word and character-level information, the subsequent RNN and projection layers can make use of the additional information to improve the ex-

traction of opinion target expressions.

5.4 Experimental Evaluation and Discussion

In this section, we evaluate the impact of using character-level word embeddings on the task of extracting opinion target expressions from user-generated reviews. We start by describing the dataset that we use in our evaluation and give relevant statistics to put the evaluation into perspective. Afterwards, we compare the character-enhanced model from Section 5.3 to the baseline RNN of Section 5.2. To select a fitting set of hyperparameters for each model, we perform a 5-fold cross validation on the training portion of our dataset. Using the best hyperparameters, we evaluate both models on the test portion of the data and investigate the models' properties with respect to the induced character information in Section 5.4.3. Evaluation is carried out in terms of F_1-score of expected opinion target expressions and retrieved opinion target expressions using exact matches[1]. Following this, we probe the learned character-based word representations with respect to the encoded information in Section 5.4.4 and their performance on selected subsets of the test data in Sections 5.4.5 and 5.4.6.

5.4.1 Dataset

In our experiments, we use the data for the SemEval 2016 ABSA Task 5 [Pontiki et al., 2016]. The used dataset consists of English review sentences from the restaurant domain with annotations for opinion target expressions. Table 5.1 gives a summary of the dataset. Notably, the training portion of the datasets contains 2000 review sentences with a total of 1880 OTEs. The length of the target phrases ranges from 2 characters per phrase up to 80 characters. An example sentence with a particularly long target phrase is given in the example below:

Example:

> *"* *noodles with shrimp and chicken and coconut juice* *is the MUST!"*

5.4.2 Experimental Settings

The optimization of the model parameters is done by minimizing the classification error for each word in the sequence using the cross-entropy loss. The optimization is carried out using a mini-batch size of 5 with the stochastic optimization technique *Adam* [Kingma and Ba, 2014]. We clip the norm of the gradients to 5 and

[1] We use the provided evaluation code from the organizers of the SemEval 2016 challenge.

Dataset	Sentences	OTEs	Chars. per OTE	Words per OTE
Train	2000	1880	2 – 80	1 – 15
Test	676	650	3 – 50	1 – 11

Table 5.1: Relevant statistics of the SemEval 2016 dataset (Task 5, restaurant domain).

regularize our network quite rigorously using L2 regularization of 10^{-5} on $\mathbf{W}^{tag}$ and $\mathbf{W}^{cw}$, as well as using Dropout [Srivastava et al., 2014] in various positions in our network. Specifically, we apply Dropout with a drop probability of 0.5 to the character and word embeddings, the output of the character-level GRUs, as well as the input and hidden sequence of the word-level GRUs as proposed in [Gal and Ghahramani, 2016]. Initial experiments suggested that this strong regularization is necessary due to the moderate size of the training dataset. The networks are implemented using the machine learning framework *Keras* [Chollet, 2015].

The word embedding matrix $\mathbf{W}^{wrd}$ is initialized with a pretrained matrix of skip-gram embeddings trained on a corpus of Amazon reviews [McAuley et al., 2015a]. With these embeddings, we rely on our findings of Chapter 3 that using a domain-specific corpus in the pretraining stage significantly improves performance for opinion target extraction.

5.4.3 Overall Performance

Hyperparameter Selection

We set the dimensionality d^{wrd} of the pretrained word embeddings to 100 and perform a grid search on a subset of the hyperparameters to find a suitable solution to be used in the final system configuration. We evaluate each candidate set of hyperparameters using a 5-fold cross validation on the training data. The search is performed for each model (`word-only` and `char+word`). We experiment with:

- the size of the word vocabulary[2] $|V^{wrd}| \in \{10000, 20000, 50000\}$ (with respect to the most frequent words),
- the size of the sentence level RNN hidden layer $r^{wrd} \in \{60, 100, 200\}$,
- and the size of the character-level RNN and the corresponding character-level word embedding vector $d^{chr} \in \{20, 50, 100\}$.

Table 5.2 shows the best hyperparameters for each model. As expected, the search indicates that it is always better to increase the size of the word vocabulary

[2]The size of the word vocabulary is the main factor in terms of (GPU) memory usage.

Model	$\lvert V^{wrd}\rvert$	r^{wrd}	d^{chr}	∅ F_1
`word-only`	50000	60	–	0.6713
`char+word`	50000	100	100	**0.6936**

Table 5.2: Results of a search for hyperparameters. The column ∅ F_1 gives the best mean F_1-score for the best performing training epoch across cross-validation models.

Model	F_1-score
`word-only`	0.6260
`char+word`	**0.6586**

Table 5.3: Results on the test set for best performing hyperparameters. The previous findings of the usefulness of character-level word embeddings are confirmed by the results of the test set.

V^{wrd}. The best model using both word and character-level information performs on average about 2.2 points F_1-score better than the best model that only uses word-level information. For the following evaluations, we instantiate and train our models according to these hyperparameters.

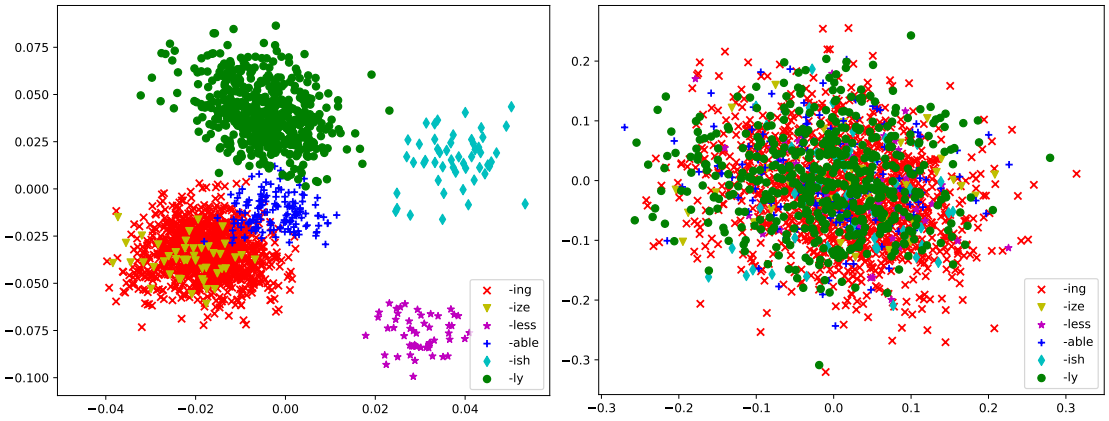
Performance on Held-Out Data

For the evaluation on the test set, we use the previously found hyperparameters and instantiate our models. We train both models on 80% of the training set and use the remaining 20% as a validation set for early stopping [Caruana et al., 2000]. The `word-only` model reaches its best performance at epoch 35 and the `char+word` model peaks at epoch 73. The performances of both models are given in Table 5.3. The results confirm our hypothesis and the findings from the cross validation that the character-level word embeddings offer a substantial improvement (3.3 points F_1-score) over the `word-only` baseline model.

5.4.4 Suffix Information

Our empirical results show that learned character-level word embeddings improve the extraction performance for OTE. We hypothesize that the character-level word embeddings encode morphological features of a word. To confirm this assumption, we visualize the learned embeddings using suffix information.

From a large collection of reviews, we extract a subset of the 2000 most frequently occurring words that end on one of the following suffixes: *-ing*, *-ly*, *-able*, *-ish*, *-less*, *-ize*. We project the character-level word embeddings of the



(a) Character-level embeddings (b) Word-level embeddings

Figure 5.3: Visualization of suffix information of the two employed types of embeddings.

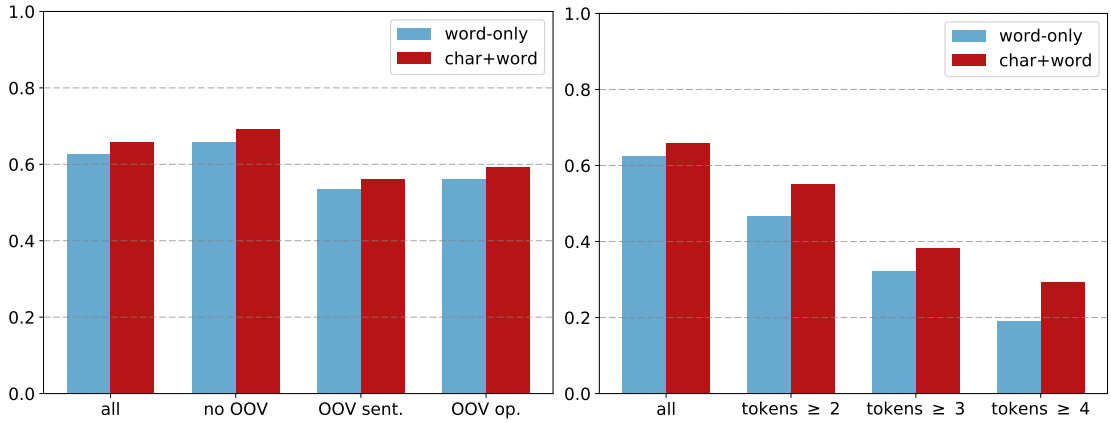
words to a 2 dimensional space using the dimensionality reduction technique t-SNE [van der Maaten and Hinton, 2008]. The resulting low-dimensional embeddings are plotted as a scatter plot. By highlighting each word according to its suffix, we see that the character-level embeddings are grouped according to their suffixes (see Figure 5.3a). Performing the same procedure with the regular skip-gram word embeddings results in no clear separation between the 6 suffix groups (see Figure 5.3b).

In the earlier Chapter 3, we could show a positive impact of POS tag features for the extraction of opinion phrases and opinion target expressions. It stands to reason if the character-level word embeddings act in a similar way. The morphological information of character-level word embeddings (as shown in Figure 5.3a) might help to disambiguate word occurrences with respect to their linguistic function in the sentence, similar to the positive effect of POS tags for this task. We leave the verification of this hypothesis for future work.

5.4.5 Out-of-Vocabulary Errors

Next, we are interested in seeing if the improvement in F_1-score can be traced back to Out-of-Vocabulary (OOV) word errors. For this, we compute the F_1-score on 3 different subsets of sentences for the `word-only` model and the `char+word` model:

- `no OOV`: This subset only contains sentences for which all words are part of the known vocabulary.
- `OOV sent.`: This subset contains sentences that contain an unknown word at some position in the sentence.



(a) Performance on OOV-related subsets. (b) Performance on multi-word phrases.

Figure 5.4: Illustration of performance differences for different subsets of sentences.

Word	atmosphere	restaurant	service
Nearest Neighbors	atomosphere	restaraunt	customer
	ambience	eatery	serivce
	atmoshpere	restuarant	costumer

Table 5.4: Three commonly used words in restaurant reviews and their 3 nearest neighbors in the embedding space. Often, misspelled versions of the original word are among its closest neighbors. We highlight these with wavy lines.

- `OOV op.`: The subset of sentences that contain at least one opinion target expression with an unknown word.

Figure 5.4a shows F_1-scores for different subsets. Surprisingly, we can see that the F_1-scores rise and fall similarly for both models regardless of the evaluated subset. This suggests, that the positive influence of the character information does not particularly help in those cases where the text contains previously unseen words (e.g. misspelled words). We assume that the positive impact on these cases is mitigated since the domain specific skip-gram word embeddings already contain various writing errors that frequently occur in customer reviews. This can be seen in Table 5.4, which shows the nearest neighbors of exemplary words in the skip-gram embedding space. We see that common writing mistakes are often already captured by the word embeddings.

5.4.6 Multi-Word Expressions

Another possible cause for the performance difference of both models might be related to the length of opinion target expressions. This hypothesis is motivated by the idea that e.g. variations in spelling with respect to hyphenation (e.g. *bartenders* vs. *bar tenders* or *wait staff* vs. *wait-staff*) could have less of an influence on the character-based model than on the word-based model. To test this idea, we consider subsets of sentences that contain at least one OTE that is a multi-word expression of more than or equal to k words. The performance differences for $k \in \{2, 3, 4\}$ are visualized in Figure 5.4b.

The first thing to notice is that both models are strongly affected by the length of the OTEs. Longer expressions seem to be harder to extract in general. However, we can observe that the character model is influenced by the length of an OTE to a lesser degree. While the difference in F_1-score for all sentences between the `word-only` model and `char+word` model is about 3.3, the differences for OTEs composed of more than or equal to 2, 3, and 4 words are 8.4, 6.1 and 10.4, respectively.

5.5 Conclusion

There is a growing interest in character and subword-level models for natural language processing in recent years. Tokenization is a crucial step for many applications, yet neglects the information that can be gained from the character structure of a word itself.

In this chapter, we were able to show that character-level information assists in the task of opinion target extraction, an important step in aspect-based sentiment analysis. We compared a model using only word-level features to a more sophisticated model that also includes character-level word embeddings. We showed that the more complex character model consistently outperforms the baseline model with a substantial margin of 3.3 points F_1-score. A visualization of the learned embeddings revealed encoded morphological regularities that we could not find in our skip-gram word embeddings. Through experiments on different subsets of the data, we linked the positive influence of the character-level word embeddings to the difficulty of extracting multi-word expressions. We did not observe a performance difference for Out-of-Vocabulary cases. However, it is not entirely clear how exactly the additional character information contributes to the task of extracting opinion target expression. In general, we suspect that the morphological information of character-level word embeddings helps to disambiguate word occurrences similarly to the positive effect of POS tags for OTE extraction. A confirmation of this hypothesis remains for future work. An interesting direction for future work

is the pretraining of parts of the network to enrich the character-based word representation. To a degree, this is approached by [Akbik et al., 2018] who initialize early parts of their network with a pretrained character-level language models.

The positive results of this work and remaining open questions suggest a need to focus further research effort in the direction of character-level neural network models in order to improve token-based approaches or even replace the need for tokenization altogether.

Chapter 6

Zero-Shot Cross-Lingual Opinion Target Extraction

Chapter Overview *In this chapter, we address the lack of available annotated data for specific languages by proposing a zero-shot cross-lingual approach for the extraction of opinion target expressions. We leverage multilingual word embeddings that share a common vector space across various languages and incorporate these into a convolutional neural network architecture for OTE extraction. Our experiments with 5 languages give promising results: We can successfully train a model on annotated data of a source language and perform accurate prediction on a target language without ever using any annotated samples in that target language. Furthermore, we can increase the prediction accuracy by performing cross-lingual learning from multiple source languages.*

6.1 Introduction

A key task within aspect-based sentiment analysis consists of the identification of Opinion Target Expressions (OTEs). Opinion target expressions are segments of a text that specify the target of an opinion and, therefore, form a crucial part of understanding an opinion. To automatically extract OTEs, supervised learning algorithms are usually employed which are trained on manually annotated corpora. The creation of these corpora is labor-intensive and sufficiently large datasets are therefore usually only available for a very narrow selection of languages and domains. As a large part of the research community focuses on English, the available datasets for English are large compared to other, less researched languages.

For a particular domain, such as restaurant reviews, opinion targets are fairly similar across languages. In the following example, we can see parallels between an English, a Dutch, and a Spanish review that all praise the *"wine list"* of their

respective restaurants[1]:

Example:

> *"The wine list is excellent."*
>
> *"Er is een uitstekende wijnkaart die regelmatig vernieuwd wordt."*
> (*"There is an excellent wine list that is regularly updated."*)
>
> *"Magnífica atención, buena carta de vinos, muy buen paella."*
> (*"Magnificent attention, good wine list, very good paella."*)

The shared semantic and grammatical regularities of these languages suggest that cross-lingual learning could be beneficial for OTE extraction by effectively allowing a model to use *more* training data. Motivated by this, in this chapter, we are concerned with the transfer of trained models across languages to alleviate the need for multilingual training data. We explore the overarching question of how we can leverage the available data in one language to support the detection of OTEs in other languages.

This chapter brings together the domains of opinion target extraction on the one side and cross-lingual learning on the other side. In the following, we give a brief overview of both domains and point out parallels and shortcomings of the existing research.

Opinion Target Extraction for Multiple Languages Kumar et al. [2016] present a CRF-based model for opinion target extraction that makes use of a variety of morphological and linguistic features and is one of the few systems that submitted results for more than one language for the SemEval 2016 ABSA challenge. The strong reliance on high-level NLP features, such as dependency trees, named-entity information and WordNet features restricts its wide applicability to resource-poor languages. Àlvarez-López et al. [2016] propose a CRF model using only word, lemma, bi-gram, and POS tag features features. The approach is trained on English and on Spanish data separately and achieved the best performance for the latter. While the results for the individual languages are good, the model does not leverage features across languages. Another CRF model is presented by Brun et al. [2016] who use POS tags, word lemmas, and lexico-semantic features. The approach is tested on English and French data with successful results for French. Again, the model is trained for both languages separately without any cross-lingual learning. In a recent work, Agerri and Rigau [2019] present a system for opinion target extraction that reaches state-of-the-art results on several languages using the same perceptron architecture for all languages. Their work differs

[1]Translations obtained from DeepL: https://www.deepl.com/translator

from ours in that their focus is to develop a single architecture that works well for multiple languages if the required training data is provided for that language. In contrast, our aim is to leverage the existing training data for a given language in order to support languages with fewer resources. For a more comprehensive overview of ABSA and OTE extraction approaches in general we refer to earlier chapters and Pontiki et al. [2016].

Cross-Lingual and Zero-Shot Learning for Sequence Labeling With the CLOpinionMiner, Zhou et al. [2015b] present a method for cross-lingual opinion target extraction that relies on machine translation. The approach derives an annotated dataset for a target language by translating the annotated source language data. Part-of-Speech tags and dependency path-features are projected into the translated data using the word alignment information of the translation algorithm. The approach is evaluated for English to Chinese reviews. A drawback of the presented method is that it requires access to a strong machine translation algorithm for source to target language that also provides word alignment information. Additionally, it builds upon NLP resources that are not available for many potential target languages. Addressing the task of zero-shot Spoken Language Understanding (SLU), Upadhyay et al. [2018] follow a similar approach as our work. They use the aligned embeddings from Smith et al. [2017] in combination with a BiRNN and target zero-shot SLU for Hindi and Turkish. Their evaluation is limited to the two language pairs English-Turkish and English-Hindi and does not consider other approaches for word embedding alignment.

Overall, our work differs from the related work by presenting a simple model for the accurate cross-lingual and zero-shot extraction of opinion target expressions. Similar to Upadhyay et al. [2018], our model leverages multilingual word embeddings [Smith et al., 2017; Lample et al., 2018] that share a common vector space across various languages. The shared space allows us to transfer a model trained on source language data to predict OTEs in a target language for which no (i.e. zero-shot setting) or only small amounts of data are available. The multilingual word embeddings themselves mostly rely on unlabeled text corpora and can be generated with very limited supervision [Smith et al., 2017] or none at all [Lample et al., 2018]. By using no annotated target data or elaborate NLP resources, such as Part-of-Speech taggers or dependency parsers, our approach is easily applicable to many resource-poor languages.

6.1.1 Contributions and Structure of Chapter

In this chapter, we are concerned with the annotation effort that is required for training a tool for the automatic extraction of OTEs. As most NLP research

focuses on English, other languages remain underrepresented in terms of the availability of tools and resources. To alleviate this issue, we are interested in answering the following research question:

RQ4 *How can we reduce the annotation effort for the creation of training data for under-resourced languages?*

We approach this question by proposing a model that is capable of accurate zero-shot cross-lingual OTE extraction, thus reducing the reliance on annotated data for every language. We evaluate this approach with regards to a set of focused questions:

RQ4.1 *To what degree is a multilingual model capable of performing OTE extraction for unseen languages?*

RQ4.2 *What is the benefit of training a model on more than one source language?*

RQ4.3 *How much annotation effort can be saved by harnessing cross-lingual learning?*

RQ4.4 *How big is the impact of the used alignment method on the OTE extraction performance?*

Our main contributions can be summarized as follows:

i) We present the first approach for zero-shot cross-lingual opinion target extraction and achieve up to 87% of the performance of a monolingual baseline.

ii) We investigate the benefit of using multiple source languages for cross-lingual learning and show that we can improve by 6 to 8 points in F_1-score compared to a model trained on a single source language.

iii) We investigate the benefit of augmenting the zero-shot approach with additional data points from the target language. We observe that we can save hundreds of annotated data points by employing a cross-lingual approach.

iv) We compare two methods for obtaining cross-lingual word embeddings on the task.

The remaining part of the chapter is structured as follows: Section 6.2 presents our mono-lingual baseline model for opinion target extraction. The model follows the same principles for the extraction of OTEs that we established in previous chapters. In Section 6.3, the mono-lingual model is extended with multilingual word embeddings that enable transfer learning between arbitrary combinations of source and target languages. Section 6.4 describes our experiments in which we evaluate the proposed approach with respect to our research questions. The chapter is concluded with Section 6.5 which summarizes the results and highlights avenues for future work.

6.2 Monolingual Model

Throughout this thesis, we presented several variations of neural-network-based models that can be trained to extract OTEs by framing the task as a sequence labeling problem. We reiterate that an annotated text can be represented as a sequence of tokens, where each token is labeled with a tag:

The	*wine*	*list*	*is*	*also*	*really*	*nice*	.
O	**I**	**I**	O	O	O	O	O

In this example, we see the opinion target *wine list* which is encoded as a sequence of tags according to the IOB tagging scheme. For a more detailed description of the encoding, see Section 2.3.

In this chapter, we use a multi-layer CNN as our sequence tagging model. The model receives a sequence of words as input features and predicts an output sequence of IOB tags. In order to keep our model simple and our results clear, we restrict our input representation to a sequence of word embeddings. While additional features such as POS tags are known to perform well in the domain of OTE extraction (cf. Section 3.6.4 and [Toh and Su, 2016; Kumar et al., 2016]), they would require a separately trained model for POS-tag prediction which can not be assumed to be available for every language. Furthermore, we refrain from using more complex architectures such as memory networks as our goal is mainly to investigate the possibility of performing zero-shot cross-lingual transfer learning for OTE prediction. Being the first approach proposing this, we leave the question of how to increase the performance of the approach by using more complex architectures to future work.

In the following, we describe our monolingual CNN model for OTE extraction which we use as our baseline model. Afterwards, we show how we adapt this model for a cross-lingual and even zero-shot regime. Our monolingual baseline model consists of a word embedding layer, a stack of convolution layers, a standard feed-forward layer followed by a final output layer. Formally, the word sequence:

$$\mathbf{w} = (w_1, \ldots, w_n)$$

is passed to the word embedding layer that maps each word w_i to its embedding vector x_i using an embedding matrix $\mathbf{E}$. The sequence of word embedding vectors:

$$\mathbf{x} = (x_1, \ldots, x_n)$$

is processed by a stack of L convolutional layers[2], each with d^{conv} kernels, a kernel

[2]The input sequences are padded with zeros to allow the application of the convolution operations to the edge words.

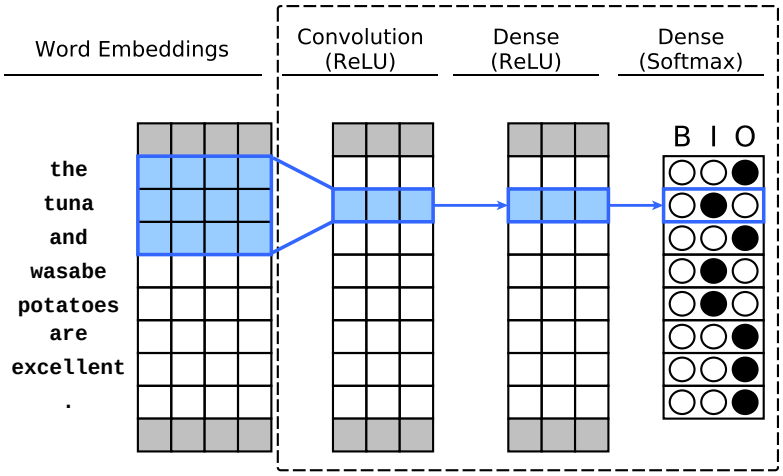


Figure 6.1: Model for sequence tagging using convolution operations. For simplicity, we only show a single convolution operation. The gray boxes depict padding vectors. The layers inside the dashed box are shared across multiple languages.

width of l^{conv}, and ReLU activation functions f (see Chapter 2). The final output of these convolution layers is a sequence of abstract representations:

$$\mathbf{h}^L = CNN(\mathbf{h}^{L-1}) = (h_1^L, \ldots, h_n^L)$$

that incorporate the immediate context of each word by means of the learned convolution operations. The hidden states $\mathbf{h}^L$ of the last convolution layer are processed by a regular feed-forward layer to further increase the model's capacity and the resulting sequence is passed to the output layer. In a last step, each hidden state is projected to a probability distribution over the possible output tags B, I, and O using a standard feed-forward layer with weights $\mathbf{W}^{tag}$, bias b^{tag} and a softmax activation function. As in previous chapters, we optimize the network parameters for minimal categorical cross entropy loss regarding the expected and predicted tag probabilities. Figure 6.1 depicts the sequence labeling architecture.

6.3 Cross-Lingual Model

Our cross-lingual model works purely with word embeddings that have been trained on monolingual datasets and in a second step have been aligned across languages. In fact, the embeddings are precomputed in an offline fashion and are not adapted while training the convolutional network on data from a specific language. As the inputs to the convolutional network are only the cross-lingual embeddings, the network can be applied to any language for which the embeddings have been

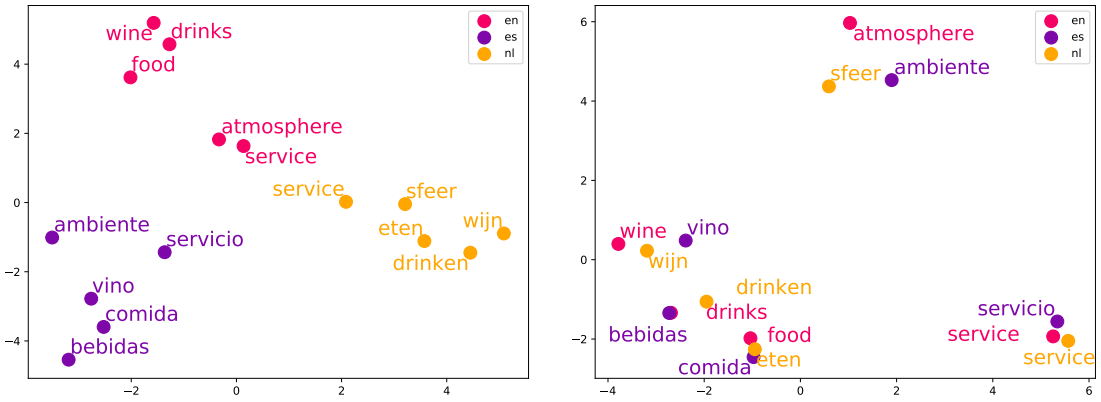
aligned. In the following, we i) introduce two approaches to obtain multilingual word embeddings and ii) describe how we can adapt the monolingual baseline to handle multiple languages at once.

6.3.1 Aligning Word Embeddings across Languages

Word embedding trained on unlabeled data for a particular language often exhibit useful semantic and syntactic relationships between words of that language [Andreas and Klein, 2014]. However, when comparing the embedding space for independently trained models for different languages, we quickly see that these relations do not hold across languages. While both vector spaces have a similar internal structure (e.g. "*wine*" is close to "*drinks*" and "*vino*" is close to "*bebidas*"), these structures are not aligned and thus not comparable (e.g. "*wine*" is not close to "*vino*"). In an effort to alleviate this problem, several approaches have been proposed that align pretrained word embeddings across languages [Smith et al., 2017; Lample et al., 2018; Chen and Cardie, 2018]. As a result of these alignment procedures, the aligned embeddings share a common vector space and are thus comparable and, to an extend, exchangeable. In this chapter, we rely on two approaches to compute embeddings that are aligned across languages. Both methods rely on fastText [Bojanowski et al., 2017] to compute monolingual embeddings trained on Wikipedia articles. The first method is the one proposed by Smith et al. [2017], which computes a Singular Value Decomposition (SVD) on a dictionary of translated word pairs to obtain an optimal, orthogonal projection matrix from one space into the other. We refer to this method as `SVD-aligned`. We use these embeddings in our experiments in Sections 6.4.3, 6.4.4 and 6.4.6.

The second method proposed by Lample et al. [2018] performs the alignment of embeddings across languages in an unsupervised fashion, without requiring translation pairs. The approach uses adversarial training to initialize the cross-lingual mapping and a synthetically generated bilingual dictionary to fine-tune it with the Procrustes algorithm [Schönemann, 1966]. We refer to the multilingual embeddings from Lample et al. [2018] as `ADV-aligned`. These are used in Section 6.4.5.

To illustrate the effect of the alignment procedure, we visualize a small subset of the embeddings for three languages in Figure 6.2 before and after the alignment with `SVD-aligned`. For the purpose of the visualization, the dimensionality of the embeddings is reduced to 2 dimensions using a PCA projection [Pearson, 1901; Hotelling, 1933]. Before the alignment, the original embeddings for the separate languages are strongly separated and do not show meaningful similarities between translation triples such as "*wine*", "*vino*", and "*wijn*". Afterwards, these triples exhibit the desired similarities while the closeness to other semantically similar words, such as "*drinks*", is retained.



(a) Original embeddings. (b) Aligned embeddings.

Figure 6.2: Word embeddings for English, Spanish, and Dutch before (left) and after (right) the alignment procedure.

6.3.2 Cross-lingual Transfer

The monolingual baseline model from Section 6.2 uses only word embeddings as the feature representation of the input text. In the monolingual setting, the word embedding matrix is pretrained for a particular language and fixed such that the vector representation does not change over the course of the model training. To extend this model to a cross- or multilingual setting, we employ the aforementioned multilingual word embeddings.

Suppose we have a dataset of training samples for a *source language* s and we want to leverage those to train a model for a *target language* t. We also assume that we have precomputed, monolingual word embeddings $\mathbf{E}^s$ and $\mathbf{E}^t$ for source and target language, respectively[3]. As a first step, we align the two sets of embeddings using the previously mentioned techniques and obtain *aligned* embeddings $\hat{\mathbf{E}}^s$ and $\hat{\mathbf{E}}^t$. Due to this alignment, semantically similar words in $\hat{\mathbf{E}}^s$ and $\hat{\mathbf{E}}^t$ are also similar in a mathematical sense, i.e. in terms of cosine similarity. We then instantiate our monolingual model and initialize the word embedding layer of the model with the aligned embeddings for the source language $\hat{\mathbf{E}}^s$. We proceed to train the model using the training samples for s as we would do in a monolingual setting. In the course of the training, the model parameters for the convolution layers and the fully-connected layers are optimized to minimize the loss for the given training data while the word embeddings are kept fixed. After the training of the network is finished, we substitute the source language embeddings with the aligned

[3]Obtaining word embeddings for under-represented languages is usually not a problem as most embedding approaches merely require unlabeled datasets which are available in sufficient size for many languages.

embeddings for the target language $\hat{\mathbf{E}}^t$. Since the word embeddings for source and target language share a common vector space, the remaining parts of the network are able to process target language samples and perform accurate zero-shot opinion target extraction.

Multiple Source and Target Languages The aforementioned procedure allows us to train a model on a source language and perform predictions on a target language. Since the core layers of the network are language agnostic, we can extend this approach to any number of source and target languages, provided that the embeddings of the considered languages share a common vector space. One problem in this regard is that both alignment methods from Section 6.3.1 are conceived for producing bilingual word embeddings, that is, they can align the embeddings of a *single* source language to a *single* target language. The simultaneous alignment of *several* languages is not provided as part of the original methods. However, we can enable this by using a proxy language. By mapping all languages to the same target language, e.g. English, all these languages effectively share the same vector space. The training of a cross-lingual model for OTE extraction that leverages the training data of multiple source languages is then a straightforward extension of the procedure for a single source and target language.

6.4 Experimental Evaluation and Discussion

In this section, we investigate the proposed zero-shot cross-lingual approach and evaluate it on the widely used ABSA dataset of the SemEval 2016 workshop. With our evaluation, we answer our final major research question:

RQ4 *How can we reduce the annotation effort for the creation of training data for under-resourced languages?*

We are particularly concerned with the following questions that each relate to different aspects:

RQ4.1 *To what degree is a multilingual model capable of performing OTE extraction for unseen languages?*

RQ4.2 *What is the benefit of training a model on more than one source language?*

RQ4.3 *How much annotation effort can be saved by harnessing cross-lingual learning?*

RQ4.4 *How big is the impact of the used alignment method on the OTE extraction performance?*

Dataset	#Sent.	#Tokens	#Targets
en (train)	2,000	29,278	1,880
en (test)	676	10,080	650
es (train)	2,070	36,164	1,937
es (test)	881	13,290	731
nl (train)	1,711	24,981	1,283
nl (test)	575	7,690	394
ru (train)	3,655	53,734	3,159
ru (test)	1,209	17,856	972
tr (train)	1,232	12,702	1,385
tr (test)	144	1,360	159

Table 6.1: Statistics of the SemEval 2016 ABSA dataset for the restaurant domain.

Before we answer these questions, let us first give a brief overview of the used datasets and resources.

6.4.1 Datasets

As part of Task 5 of the SemEval 2016 workshop [Pontiki et al., 2016], a collection of datasets for aspect-based sentiment analysis on various languages and domains was published. Due to its relatively large number of samples and high coverage of languages and domains, the datasets are commonly used to evaluate ABSA approaches. To answer our research questions, we make use of a selection of the available datasets. We evaluate our cross-lingual approach on the available datasets for the restaurant domain for the 5 languages Dutch (nl), English (en), Russian (ru), Spanish (es) and Turkish (tr)[4]. Table 6.1 gives a brief overview of the used datasets.

6.4.2 Experimental Settings

In all our experiments, we report F_1-scores for the extracted opinion target expressions computed on exact matches of the character spans as in the original SemEval task [Pontiki et al., 2016]. As described in Section 6.3, our model relies on pre-trained multilingual embeddings. For both of the used alignment approaches, namely SVD-aligned and ADV-aligned, we use the embeddings as provided by the original authors[5]. However, we restrict our vocabulary to the most frequent

[4]The dataset for French reviews was no longer available.

[5]https://github.com/Babylonpartners/fastText_multilingual and https://github.com/facebookresearch/MUSE

50,000 words per language[6] to reduce memory consumption. For all experiments, we fix our model architecture to 5 convolution layers with each having a kernel size of 3, a dimensionality of 300 units and a ReLU activation function [Nair and Hinton, 2010]. The penultimate feed-forward layer has 300 dimensions and a ReLU activation, as well. We apply dropout [Srivastava et al., 2014] on the word embedding layer with a rate of 0.3 and between all other layers with 0.5. The word embeddings and the penultimate layer are L1-regularized with $\lambda = 0.0001$ [Ng, 2004]. The network's parameters are optimized using the stochastic optimization technique *Adam* [Kingma and Ba, 2014]. We optimize the number of training epochs for each model using early stopping [Caruana et al., 2000] but do not tune other hyperparameters of our models. We always pick 20% of our available training data for the validation process. For the zero-shot scenario, this entails that we optimize the number of epochs on the source language and not on the target language to simulate true zero-shot learning.

6.4.3 Zero-Shot Transfer Learning

In this section, we present our evaluation for zero-shot learning. We first examine a setting with a single source language. Then, we evaluate the effect of cross-lingual learning from multiple source languages.

Single Source Language

This part of our evaluation addresses our first research question:

RQ4.1 *To what degree is a multilingual model capable of performing OTE extraction for unseen languages?*

To answer this question, we perform a set of experiments in the zero-shot setting. We train a model on the training portion of a source language and evaluate the model performance on all possible target languages. In total, we consider 25 language pairs. Figure 6.3 shows the obtained scores. The reported results are averaged over 10 runs with different random seeds to account for variations due to the initialization of model parameters. The main diagonal represents results of models where both source and target language are the same. We considered these our monolingual baselines. The remaining 20 results correspond to true zero-shot settings enabled by the use of multilingual word-embeddings.

In general, the proposed approach achieves relatively high scores for some language pairs, although with large performance differences depending on the exact source and target language pairs. Looking at the absolute scores, the best

[6] As appearing in the respective embedding files.

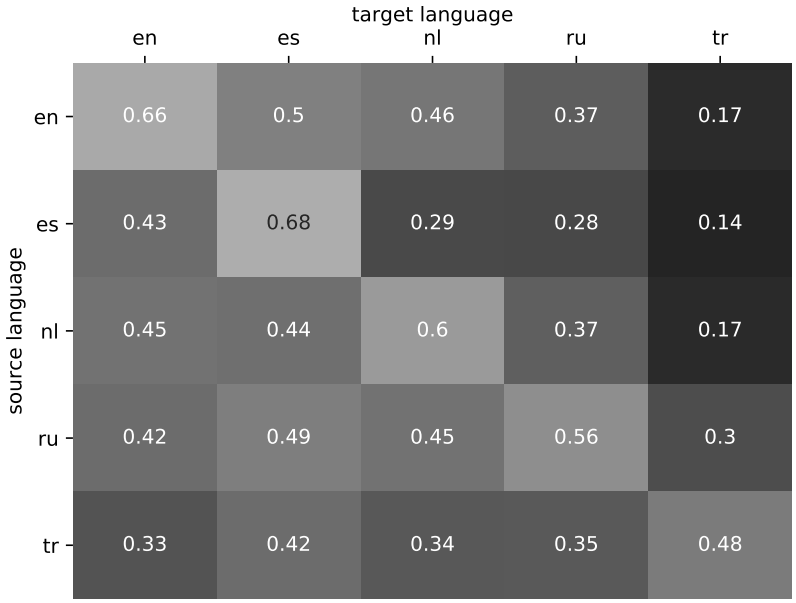


Figure 6.3: Zero-shot F_1-scores for cross-lingual learning from a single source to a target language.

performing cross-lingual language pair is `en`→`es` with an F_1-score of 0.5. This is followed by `en`→`nl` at 0.46. The lowest is `es`→`tr` with an F_1-score of 0.14. When considering the results relative to their respective monolingual baselines, the highest relative performance is achieved by `en`→`nl` at 77% of a `nl`→`nl` model, followed by `en`→`es` and `ru`→`nl`, which both reach an F_1-score of about 74%. The weakest performing language pair is still `es`→`tr` at 29% relative performance. In general, the Turkish language seems to benefit the least from the cross-lingual transfer learning, while Russian is on average the best source language in terms of relative performance achievement for the target languages.

Overall, the presented results show that it is in fact possible for most considered languages to train a model for OTE extraction without ever using any annotated data in that target language.

Multiple Source Languages

In the next experiment, we want to address our second research question:

RQ4.2 *What is the benefit of training a model on more than one source language?*

As we explained in Section 6.3.2, our approach allows us to train and test on any number of source and target languages, provided that we have aligned word embeddings for each considered language.

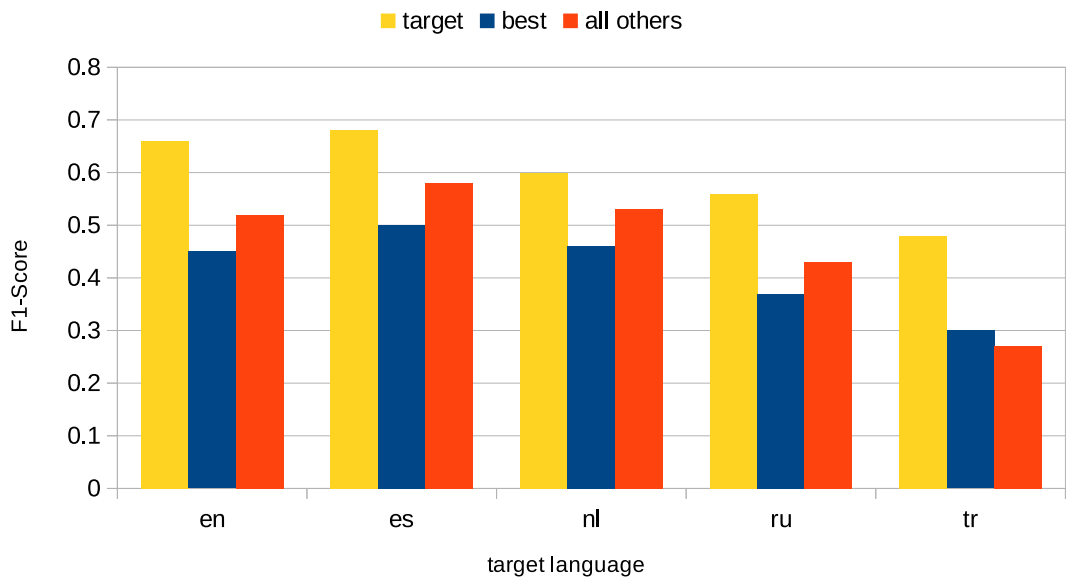


Figure 6.4: Zero-shot results for cross-lingual learning from multiple source languages to a target language. The yellow bars show the monolingual scores that act as a baseline. The blue bars represent the best performing cross-lingual model from Figure 6.3 for each target language. The orange bars show the results for training on all languages except for the target language.

In order to answer our second research question, we train a model on the available training data for all but one language and perform prediction on the test data for the left-out language. The results for these experiments are summarized in Figure 6.4. For every target language (x-axis), the figure shows the performance for three configurations:

1. a monolingual setting where source and target language are the same (yellow),
2. a zero-shot setting highlighting the single best performing source language (blue),
3. a zero-shot setting that exploits the training data for all available languages save for the target language (orange).

We can see that all languages with the exception of Turkish substantially profit from a zero-shot transfer setting with multiple source languages compared to using only a single source language. The absolute improvements are in the range of 6 to 8 points in F_1-score while the performance on Turkish samples drops by 3 points.

We can summarize that we can obtain substantial improvements for most languages when training on a combination of multiple source languages. In fact, for `en`, `es`, `nl` and `ru`, the results of our cross-lingual models trained on all other languages reach between 78% to 87% relative performance of a monolingual model.

6.4.4 Cross-Lingual Transfer Learning with Additional Target Language Data

While our goal is to reduce the effort of annotating huge amounts of data in a target language to which the model is to be transferred, it might still be reasonable to provide a few annotated samples for a target language. Our next research question addresses this issue:

RQ4.3 *How much annotation effort can be saved by harnessing cross-lingual learning?*

We answer this question by training our models jointly on a source language dataset as well as a small amount of target language samples and compare this to a baseline model that only uses target language samples. By gradually increasing the available target samples, we can directly observe their benefit on the test performance. Figure 6.5 shows a visualization for the source language `en` and the target languages `es`, `nl`, `ru`, and `tr`.

We can immediately see that a monolingual model requires at least 100 target samples to produce meaningful results as opposed to a cross-lingual model that performs well with source language samples alone. Training on increasing amounts of target samples improves the model performances monotonically for each target language and the model leveraging the bilingual data consistently outperforms the monolingual baseline model. The benefits of the source language data are especially pronounced when very few target samples are available, i.e. less than 200. As an example, a model trained on bilingual data using all available English samples and 200 Dutch samples is competitive to a monolingual model trained on 1000 Dutch samples (0.55 vs. 0.56).

As one would expect, the results in Figure 6.4 and Figure 6.5 suggest that training the model on more data samples leads to a better performance. Since our model can leverage the data from all languages simultaneously, we can exhaust our resources and train an instance of our model that has access to all training data samples from all languages including the target training data. This is reflected by the dashed line in Figure 6.5. We see that we reach our highest scores for all languages when exploiting all available source languages. However, the model cannot leverage the other source languages far beyond what it already achieves in a bilingual setting. Nevertheless, for the considered languages, we can conclude

that enriching the original monolingual training dataset with further samples from other languages is always beneficial for OTE extraction.

6.4.5 Comparison of Alignment Methods

The previous experiments show that we can achieve good performance in a cross-lingual setting for OTE extraction using the multilingual word embeddings proposed by Smith et al. [2017]. Now we address our final research question:

RQ4.4 *How big is the impact of the used alignment method on the OTE extraction performance?*

With our final research question, we compare our previous results to an alternative method of aligning word embeddings in multiple languages. We repeat our experiments in Section 6.4.3 using the embeddings of Lample et al. [2018] which we refer to as `ADV-aligned`. To enable a direct comparison to the zero-shot results in Section 6.4.3, we report absolute differences in F_1-score to the scores obtained with `SVD-aligned` for all source and target language combinations.

As can be seen in Figure 6.6, the two methods both perform well overall, albeit different for specific language pairs. In a monolingual setting (i.e. main diagonal), `ADV-aligned` performs slightly worse than `SVD-aligned` with the exception of `en`→`en`. Using `ADV-aligned`, Spanish appears to be a more effective source language than using `SVD-aligned` as the average performance is about 2.9 points higher. It can also be observed that the cross-lingual transfer learning works better for English as a target language using `ADV-aligned` since the average performance is about 2.2 points higher than for `SVD-aligned`. The opposite is true for Dutch as a target language, which shows a reduction in performance by 2.1 points on average. The largest difference is observed for the language pair `tr`→`nl` where the performance of `ADV-aligned` drops by 6.2 points F_1-score compared to `SVD-aligned`. Overall, we see that both types of embeddings perform comparably well: For 13 of the 25 language pairs, the embeddings based on `SVD-aligned` perform better than embeddings aligned with `ADV-aligned` with an average of only 0.2 points higher scores.

6.4.6 Comparison to State-of-the-Art

In this last part of our evaluation, we want to put the results of this chapter into perspective of state-of-the-art systems for opinion target extraction on the SemEval 2016 restaurant datasets. We report results for our multilingual model that is trained on the combined training data of all languages and evaluated on the corresponding test datasets. We compare our model to the respective state-of-the-art for each language in Table 6.2.

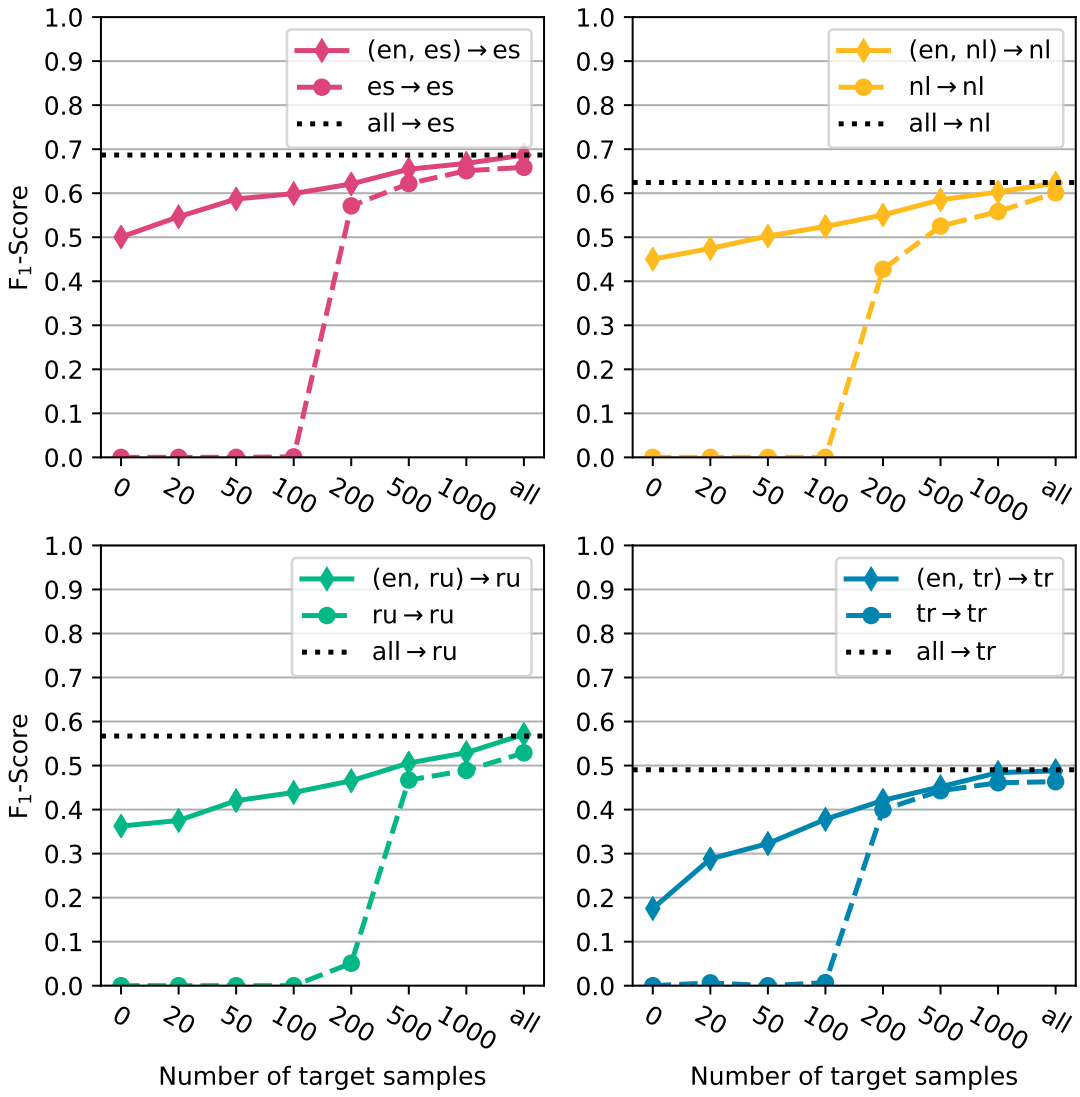


Figure 6.5: Cross-lingual results for increasing numbers of training samples from the target language.

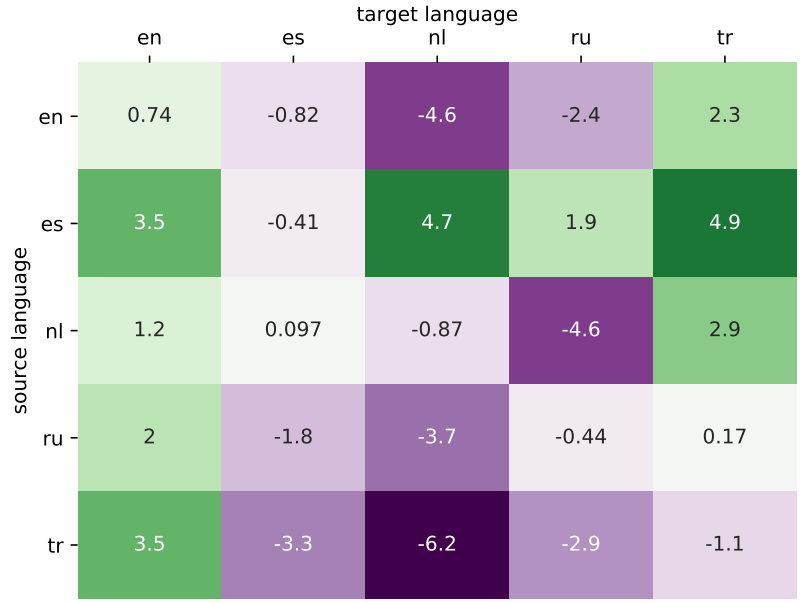


Figure 6.6: Zero-shot results comparing the multilingual embeddings `ADV-aligned` to `SVD-aligned`. A positive value means higher absolute F_1-score for `ADV-aligned` and vice versa. For readability, score differences are scaled by a factor of 100.

System	en	es	nl	ru	tr
Toh and Su [2016]	0.723	–	–	–	–
Àlvarez-López et al. [2016]	0.666	0.685	–	–	–
Kumar et al. [2016]	0.685	0.697	0.644	–	–
Pontiki et al. [2016]*	0.441	0.520	0.506	0.493	0.419
Li and Lam [2017]	0.734	–	–	–	–
Agerri and Rigau [2019]	0.735	0.699	0.664	0.655	0.602
`all→target` (Ours)	0.660	0.687	0.624	0.567	0.490

Table 6.2: Overview of the current state-of-the-art for opinion target extraction for 5 languages. Our model is trained on the combined training data of all languages and evaluated on the respective test datasets. The row marked with * is the baseline provided by the workshop organizers.

We can see that the competition is strongest for English and very recently also Russian and Turkish. Here, we fall behind latest monolingual systems. For English, this corresponds to rank 7 of 19 of the original SemEval competition. However, we see that we clearly outperform the original baseline system from Pontiki et al. [2016] for all languages by at least 7 points in F_1-score. Regarding Spanish and Dutch, we see that we are close to the best published systems.

Overall, we present the first approach on this task to achieve such promising performances for a variety of languages with a single, multilingual model. While we do not surpass dedicated monolingual models, we believe that a more complex model architecture as proposed by Li and Lam [2017] or a richer feature set as by Agerri and Rigau [2019] could close the gap.

6.4.7 Discussion

The presented experiments shed light on the performance of our proposed approach under various circumstances. In the following, we want to discuss its limitations and consider explanations for performance differences of different language pairs.

Model Limitations

The core of our proposed sequence labeling approach consists of aligned word embeddings and shared CNN layers. Due to the limited context of a CNN layer, the model can only base its decisions for each word on the local information around that word. In many cases, this information is sufficient since most opinion target expressions are adjective-noun phrases[7] which are well enough identified by the local context for most considered languages. Nevertheless, it is worth to investigate in how far our findings translate to more complex model architectures.

Language Characteristics

Due to the inherent variability of natural languages and of the used datasets, it is difficult to identify the exact reasons for the observed performance differences between language pairs. However, we suspect that language features such as word order, inflection, or agglutination affect the *compatibility* of languages. As an example, Turkish is considered a highly agglutinative language, that is, complex words are composed by attaching several suffixes to a word stem. This sets Turkish apart from the other four languages which we considered in this chapter and might explain some of the obtained results. This language feature might present a difficulty in our approach since the appending of suffixes is not optimally reflected

[7] 90% of OTEs in the English dataset consist of zero or more adjectives followed by at least one noun.

in the tokenization process and the used word embeddings. An approach that performs alignment of languages on subword units might alleviate this problem and lead to performance gains for language pairs with similar inflection rules.

Syntactic regularities, such as word order might also play a role in our transfer learning approach. It is reasonable to assume that the CNN layers of our approach pick up patterns in the word order of a source language that are indicative of an opinion target expression, e.g. *"the* [NOUN] *is good"*. When applying such a model to a target language with drastically different word order regularities, these patterns might not appear as such in the target language and ultimately reduce the model's performance.

For the considered languages, we see the following characteristics: Where English and Spanish are generally considered to follow a Subject-Verb-Object (SVO) order, Dutch largely exhibits a combination of SOV and SVO cases. Turkish and Russian are overall flexible in their word order and allow a variety of syntactic structures. In the case of Turkish, its morphological and syntactic features seem to explain some of the relatively low results. However, with the small sample of languages and the many potential influencing factors at play, we are aware that it is not possible to draw any strong conclusions. Further research has to be conducted in this direction to answer remaining questions.

6.5 Conclusion

In this chapter, we presented a method for cross-lingual and zero-shot extraction of opinion target expressions which we evaluated on 5 languages. Our approach uses multilingual word embeddings that are aligned into a single vector space to allow for cross-lingual transfer of models. By training a model equipped with these embeddings on annotated samples in a source language, we obtain a pretrained model for any target language through the replacement of the source language word embeddings with the corresponding set in the target language.

Using English as a source language in a zero-shot setting, our approach was able to reach an F_1-score of 0.50 for Spanish and 0.46 for Dutch. This corresponds to relative performances of 74% and 77% compared to a baseline system trained on target language data. By using multiple source languages, we increased the zero-shot performance to F_1-scores of 0.58 and 0.53, respectively, which correspond to 85% and 87% in relative terms. We investigated the benefit of augmenting the zero-shot approach with additional data points from the target language. Here, we observed that we can save several hundreds of annotated data points by employing a cross-lingual approach. Among the 5 considered languages, Turkish seemed to benefit the least from cross-lingual learning in all experiments. The reason for this might be that Turkish is the only agglutinative language in the dataset. Further,

we compared two approaches for aligning multilingual word embeddings in a single vector space and found their results to vary for individual language pairs but to be comparable overall. Lastly, we compared our multilingual model with the state-of-the-art for all languages and saw that we achieve promising performances and even come close to leading monolingual models for Spanish and Dutch.

Our results show that the quality of the transfer learning between language pairs is dependent on the exact source and target languages. As future work, it is interesting to investigate which language characteristics, if any, are critical for effective cross-lingual learning. Furthermore, it is worth to investigate in how far our findings translate to more complex model architectures that have been proposed for OTE extraction, such as LSTMs with memory interactions [Li and Lam, 2017] or attention-based models [Wang et al., 2017b]. We are especially interested in cross-lingual learning with character- or subword-level models such as the model presented in Chapter 5. Modeling words at the morpheme level could alleviate the observed difficulties with opinion target extraction for Turkish texts. Apart from that, a transfer of our approach to targeted sentiment classification as shown in Chapters 3 and 4 could yield similarly positive results and boost the classification performance for under-resourced languages.

Chapter 7

Conclusion

Chapter Overview *In this final chapter, we summarize the main findings of this thesis. We provide a brief overview of the answers to the underlying research questions that were addressed throughout the thesis. We end this chapter by giving perspectives for future research opportunities on aspect-based sentiment analysis.*

7.1 Summary

This thesis revolves around aspect-based sentiment analysis and a related formulation coined relational sentiment analysis. We recall from Chapter 3 that in the framework of relational sentiment analysis, an expressed opinion is divided into four parts:

i) an opinion phrase that carries sentimental meaning,

ii) a sentiment label that marks the opinion as e.g. positive or negative,

iii) an aspect phrase that constitutes the target of the opinion, and

iv) a relation that explicitly links an opinion phrase to one or more target phrases.

In Chapter 3, we were concerned with the automatic extraction of all four of these parts from natural language texts. We presented a complete, modular architecture for relational sentiment analysis and evaluated each proposed component (RQ1). Our evaluation showed that opinion phrases and their targets can be extracted with sequence labeling techniques based on CNNs and RNNs. A model combining CNN and RNN layers performed best (RQ1.1) and could be improved by providing POS tags and domain-specific word embeddings (RQ1.2).

We also observed that we can benefit from a jointly formulated objective for aspect and opinion phrase extraction (RQ1.3). Furthermore, we proposed an approach for opinion-phrases-specific sentiment classification using a position-aware RNN (RQ1.4). With this, our work provides the first results for targeted sentiment analysis on the considered dataset. Following a similar architectural design, we adapted the position-aware RNN for aspect-opinion relation extraction and achieved substantial improvements of 15 points F_1-score over prior work (RQ1.5).

Chapter 4 was concerned with the integration of external structured knowledge into neural architectures for ABSA (RQ2). Firstly, we explored the effect of retrofitting pretrained word embeddings to a graph of lexico-semantic relations but could not observe any positive effects on opinion target extraction or target-specific sentiment classification (RQ2.1). Secondly, we proposed to include sentiment information from a concept knowledge base as word-level features. Our experiments showed that the sentiment information, while not helpful for OTE extraction, offers substantial improvements of 3.5 points F_1-score for sentiment classification and a drastically improved training speed (RQ2.2).

In Chapter 5, we were able to show that character-level information assists in the task of opinion target extraction, an important step in aspect-based sentiment analysis. We compared a model using only word-level features to a more sophisticated model that also includes character-level word embeddings (RQ3). We showed that the more complex character model consistently outperforms the baseline model with a substantial margin of 3.3 points F_1-score. A visualization of the learned embeddings revealed encoded morphological regularities that we could not find in regular pretrained skip-gram word embeddings (RQ3.1). Via experiments on different subsets of the data, we linked the positive influence of the character-level word embeddings to the difficulty of extracting multi-word expressions (RQ3.2). However, we did not observe a performance difference for Out-of-Vocabulary cases.

Chapter 6 addresses the lack of research dedicated to aspect-based sentiment analysis for under-resourced languages. We specifically addressed this gap by proposing a model for opinion target extraction capable of cross-lingual and zero-shot transfer learning (RQ4). Using pretrained multilingual word embeddings, our proposed approach is capable of performing predictions for a target language without ever seeing any samples in that language. Using English as a source language in a zero-shot setting, our approach was able to reach an F_1-score of 0.50 for Spanish and 0.46 for Dutch. This corresponds to relative performances of 74% and 77% compared to a baseline system trained on target language data (RQ4.1). Overall, our results show that the quality of the transfer learning between language pairs is dependent on the exact source and target languages. By training a model with access to training samples of multiple source languages, we increased

the zero-shot performance to F_1-scores of 0.58 and 0.53, respectively, which correspond to 85% and 87% in relative terms (RQ4.2). We investigated the benefit of augmenting the zero-shot approach with additional data points from the target language. This allowed us to estimate how much annotation effort can be avoided by harnessing available data from multiple languages. Here, we observed that we can save several hundreds of annotated data points by employing the proposed cross-lingual approach (RQ4.3). Further, we quantified the impact of different methods of aligning word embeddings across languages and found their results to vary for individual language pairs but to be comparable overall (RQ4.4).

7.2 Outlook

Although this thesis provides answers for challenging research questions regarding aspect-based sentiment analysis, there remain open questions and further research opportunities. In the following, we give perspectives that are more general in nature in order to avoid overlap with specific research directions that were mentioned in previous chapters:

Relational Sentiment Analysis A drawback with arranging interdependent machine learning modules in a pipeline structure is the propagation and amplification of errors from one step to the next. As every single step in such a pipeline can produce incorrect results, the individual errors may add up in the course of the processing chain and lead to flawed end results. In the scope of this thesis, this pertains to the relational sentiment analysis framework and its subtasks, presented in Chapter 3. We already showed that the joint extraction of target aspects and opinion phrases leads to improvements for both subtasks (see Section 3.6.4 of Chapter 3). It is worth investigating whether a single model that addressed all subtasks with a joint objective yields further improvements and alleviates issues arising from error propagation in our pipeline. Particularly, we hypothesize that the extraction of OTEs profits from such a joint objective as this could reduce false positive phrases. The intuition behind this can be seen in the following example where the same phrases might conceivably be recognized as a target aspect despite the lack of an expressed sentiment statement:

"I ordered the *chicken fajita*" (False Positive)
"I ordered the *chicken fajita* *and enjoyed it.".* (True Positive)

Multi-Task and Transfer Learning A further research question arises from the available annotated training data. Annotated datasets for aspect-based sentiment analysis usually consist of a few hundred or thousand samples. While the

annotation effort is already high to provide such complex and reliable annotations, the datasets are still rather small when comparing these to datasets for other natural language processing problems. As an example, machine learning models for document-level sentiment classification or machine translation are usually trained on datasets with orders of magnitudes more samples. In Chapter 6, we addressed this problem from the perspective of cross-lingual transfer learning and obtained promising results. Beyond this approach, we are interested in two related avenues: Firstly, how can we leverage unlabeled textual data to improve aspect-based sentiment? We already saw the large impact of pretraining word embeddings, especially on domain-specific data (see Section 3.6.3 in Chapter 3). Similarly, we expect improvements from pretraining entire sequence processing components [Peters et al., 2018; Akbik et al., 2018; Howard and Ruder, 2018]. Secondly, we are interested in leveraging other datasets by approaching aspect-based sentiment analysis in a multi-task setting [Søgaard and Goldberg, 2016]. By selecting the right auxiliary tasks, we could harness already annotated datasets for these tasks in order to support our main objective of aspect-based sentiment analysis. While it is not clear, how to identify helpful auxiliary tasks for multi-task learning [Bjerva, 2017], we believe that low-level NLP problems such as chunking, NER, POS tagging, constituency parsing, or dependency parsing are promising candidates. The latter is particularly promising for targeted sentiment and relation extraction since syntactic dependencies are known to be an effective source of information for relation extraction in general [Fundel et al., 2007; Zhang et al., 2018; Gupta et al., 2019].

Following up on our work in Chapter 6, we see opportunities in extending the cross-lingual, zero-shot approach for OTE extraction to include subword information. We proposed a zero-shot approach that uses aligned word embeddings to represent words. It remains to be seen if such an approach can be applied to the subword-level, e.g. by performing the alignment on the level of morphemes.

Text Representation In Chapter 6, we observed performance differences of mono-lingual models across languages for the extraction of opinion target expressions. We surmise that the word-level representation of textual data might not be optimal for all languages. A simple tokenization based on naïve indicators (white spaces, punctuation) could pose problems for languages that are highly agglutinative (e.g. Turkish), fusional (e.g. Russian), or do not explicitly delimit words (e.g. Chinese). These languages might profit from other text representation approaches that take into account the subword structure of words [Akbik et al., 2018; Schuster and Nakajima, 2012]. To an extent, we could show such a benefit of subword representations for English in Chapter 5. An extension of this analysis to other languages might help explaining the observed phenomena and reveal similar benefits for opinion target extraction.

List of Tables

List of Figures

Acronyms

ABSA Aspect-based Sentiment Analysis.

ANN Artificial Neural Network.

BiGRU Bidirectional Gated Recurrent Unit.

BiRNN Bidirectional Recurrent Neural Network.

CBOW Continuous Bag-of-Words.

CNN Convolutional Neural Network.

CRF Conditional Random Field.

ELU Exponential Linear Unit.

FFNN Feed-Forward Neural Network.

GRU Gated Recurrent Unit.

LSTM Long Short-Term Memory.

MLP Multilayer Perceptron.

NER Named Entity Recognition.

NLP Natural Language Processing.

OOV Out-of-Vocabulary.

ORL Opinion Role Labeling.

OTE Opinion Target Expression.

POS Part-of-Speech.

ReLU Rectified Linear Unit.

RNN Recurrent Neural Network.

SA Sentiment Analysis.

SG Continuous Skip-Gram.

SLU Spoken Language Understanding.

SVD Singular Value Decomposition.

SVM Support Vector Machine.

tanh Hyperbolic Tangent.

Bibliography

Agerri, R. and Rigau, G. (2019). Language independent sequence labelling for Opinion Target Extraction. *Artificial Intelligence*, 268:85–95.

Akbik, A., Blythe, D., and Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Àlvarez-López, T., Juncal-Martínez, J., Fernández Gavilanes, M., Costa-Montenegro, E., and Javier González-Castaño, F. (2016). GTI at SemEval-2016 Task 5: SVM and CRF for Aspect Detection and Unsupervised Aspect-Based Sentiment Analysis. In *SemEval@NAACL-HLT*, pages 306–311. The Association for Computer Linguistics.

Andreas, J. and Klein, D. (2014). How much do word embeddings encode about syntax? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 822–827, Baltimore, Maryland. Association for Computational Linguistics.

Aprosio, A. P., Corcoglioniti, F., Dragoni, M., and Rospocher, M. (2015). Supervised opinion frames detection with RAID. In Gandon, F., Cabrio, E., Stankovic, M., and Zimmermann, A., editors, *Semantic Web Evaluation Challenges - Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia*, volume 548 of *Communications in Computer and Information Science*, pages 251–263. Springer.

Askalidis, G. and Malthouse, E. C. (2016). The value of online customer reviews. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 155–158, New York, NY, USA. ACM.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bekoulis, G., Deleu, J., Demeester, T., and Develder, C. (2018). Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34 – 45.

Bethard, S., Cer, D. M., Carpuat, M., Jurgens, D., Nakov, P., and Zesch, T., editors (2016). *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*. The Association for Computer Linguistics.

Bjerva, J. (2017). Will my auxiliary tagging task help? estimating auxiliary tasks effectivity in multi-task learning. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 216–220, Gothenburg, Sweden. Association for Computational Linguistics.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Brun, C., Perez, J., and Roux, C. (2016). XRCE at SemEval-2016 task 5: Feedbacked ensemble modeling on syntactico-semantic knowledge for aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 277–281, San Diego, California. Association for Computational Linguistics.

Cambria, E., Olsher, D., and Rajagopal, D. (2014). SenticNet 3: a common and common-sense knowledge base for cognition-driven sentiment analysis. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1515–1521.

Caruana, R., Lawrence, S., and Giles, C. L. (2000). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In Leen, T. K., Dietterich, T. G., and Tresp, V., editors, *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 402–408. MIT Press.

Cer, D. M., Jurgens, D., Nakov, P., and Zesch, T., editors (2015). *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*. The Association for Computer Linguistics.

Chen, X. and Cardie, C. (2018). Unsupervised multilingual word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 261–270, Brussels, Belgium. Association for Computational Linguistics.

Cho, H.-C., Okazaki, N., Miwa, M., and Tsujii, J. (2013). Named entity recognition with multiple segment representations. *Information Processing & Management*, 49(4):954 – 965.

Cho, K., Van Merriënboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on EMNLP*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Chollet, F. (2015). Keras -theano-based deep learning library. https://github.com/fchollet/keras.

Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y. (2014a). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Deep Learning Workshop*.

Chung, J. K., Wu, C., and Tsai, R. T. (2014b). Polarity detection of online reviews using sentiment concepts: NCU IISR team at ESWC-14 challenge on concept-level sentiment analysis. In Presutti, V., Stankovic, M., Cambria, E., Cantador, I., Iorio, A. D., Noia, T. D., Lange, C., Recupero, D. R., and Tordai, A., editors, *Semantic Web Evaluation Challenge - SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece*, volume 475 of *Communications in Computer and Information Science*, pages 53–58. Springer.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. (2012). Optimal distributed online prediction using mini-batches. *J. Mach. Learn. Res.*, 13(1):165–202.

Derczynski, L. (2016). Complementarity, f-score, and nlp evaluation. In Chair), N. C. C., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

dos Santos, C., Guimaraes, V., Niterói, R., and de Janeiro, R. (2015a). Boosting named entity recognition with neural character embeddings. In *Proceedings of NEWS 2015 The Fifth Named Entities Workshop*, page 25.

dos Santos, C., Xiang, B., and Zhou, B. (2015b). Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 626–634, Beijing, China. Association for Computational Linguistics.

dos Santos, C. and Zadrozny, B. (2014). Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1818–1826.

Dragoni, M., Tettamanzi, A. G. B., and da Costa Pereira, C. (2014). A fuzzy system for concept-level sentiment analysis. In Presutti, V., Stankovic, M., Cambria, E., Cantador, I., Iorio, A. D., Noia, T. D., Lange, C., Recupero, D. R., and Tordai, A., editors, *Semantic Web Evaluation Challenge - SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece*, volume 475 of *Communications in Computer and Information Science*, pages 21–27. Springer.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2):179 – 211.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., and Smith, N. A. (2015). Retrofitting Word Vectors to Semantic Lexicons. *Proceedings of Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*, pages 1606–1615.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication. MIT Press, Cambridge, MA.

Firth, J. R. (1957). *A synopsis of linguistic theory 1930-55.*, volume 1952-59, pages 1–32. The Philological Society, Oxford.

Fonseca, E. R. and Rosa, J. a. L. G. (2013). A two-step convolutional neural network approach for semantic role labeling. In *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7.

Frey, B. J. (1998). *Graphical Models for Machine Learning and Digital Communication*. MIT Press, Cambridge, MA, USA.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202.

Fundel, K., Küffner, R., and Zimmer, R. (2007). RelEx—Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Gal, Y. and Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A. C., and Bengio, Y. (2013). Maxout networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1319–1327. JMLR.org.

Gupta, P., Rajaram, S., Schütze, H., and Runkler, T. A. (2019). Neural relation extraction within and across sentence boundaries. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019.*, pages 6513–6520. AAAI Press.

Gupta, P., Schütze, H., and Andrassy, B. (2016). Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction. In *Proccedings of the 26th International Conference on Computational Linguistics (COLING2016)*, pages 2537–2547.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361.

He, P., Huang, W., Qiao, Y., Loy, C. C., and Tang, X. (2016). Reading scene text in deep convolutional sequences. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 3501–3508. AAAI Press.

Hercig, T., Brychcín, T., Svoboda, L., and Konkol, M. (2016). UWB at SemEval-2016 Task 5: Aspect Based Sentiment Analysis. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 342–349.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1).

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116.

Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In Kremer, S. C. and Kolen, J. F., editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pages 168–177, New York, NY, USA. ACM.

İrsoy, O. and Cardie, C. (2014). Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 720–728.

Jakob, N. and Gurevych, I. (2010). Extracting opinion targets in a single-and cross-domain setting with conditional random fields. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1035–1045.

Jordan, M. I. (1997). Chapter 25 - serial order: A parallel distributed processing approach. In Donahoe, J. W. and Dorsel, V. P., editors, *Neural-Network Models of Cognition*, volume 121 of *Advances in Psychology*, pages 471 – 495. North-Holland.

Josef Ruppenhofer, S. S. and Wiebe, J. (2008). Finding the sources and targets of subjective expressions. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, B. M. J. M. J. O. S. P. D. T., editor, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Katiyar, A. and Cardie, C. (2016). Investigating LSTMs for Joint Extraction of Opinion Entities and Relations. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 919–929.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2016). Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 2741–2749.

Kingma, D. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, pages 1–15.

Klinger, R. and Cimiano, P. (2013a). Bi-directional inter-dependencies of subjective expressions and targets and their value for a joint model. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 848–854.

Klinger, R. and Cimiano, P. (2013b). Joint and pipeline probabilistic models for fine-grained sentiment analysis: Extracting aspects, subjective phrases and their relations. In *Proceedings - IEEE 13th International Conference on Data Mining Workshops, ICDMW 2013*, pages 937–944.

Klinger, R. and Cimiano, P. (2014). The usage review corpus for fine grained multi lingual opinion analysis. In Chair), N. C. C., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Konkol, M. and Konopík, M. (2015). Segment representations in named entity recognition. In Král, P. and Matoušek, V., editors, *Text, Speech, and Dialogue*, pages 61–70, Cham. Springer International Publishing.

Kumar, A., Kohail, S., Kumar, A., Ekbal, A., and Biemann, C. (2016). IIT-TUDA at SemEval-2016 Task 5: Beyond Sentiment Lexicon: Combining Domain Dependency and Distributional Semantics Features for Aspect Based Sentiment Analysis. In [Bethard et al., 2016], pages 1129–1135.

Lackermair, G., Kailer, D., and Kanmaz, K. (2013). Importance of online product reviews from a consumer's perspective. *Advances in Economics and Business*, 1:1–5.

Lakkaraju, H., Socher, R., and Manning, C. (2014). Aspect Specific Sentiment Analysis using Hierarchical Deep Learning. *NIPS Workshop on Deep Learning and Representation Learning*.

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., and Jégou, H. (2018). Word translation without parallel data. In *International Conference on Learning Representations*.

Le, Q. and Mikolov, T. (2014). Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1188–1196.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th International Conference on Machine Learning*, pages 609–616.

Li, X. and Lam, W. (2017). Deep Multi-Task Learning for Aspect Term Extraction with Memory Interaction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2886–2892. Association for Computational Linguistics.

Liu, B. et al. (2010). Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2(2010):627–666.

Liu, P., Joty, S., and Meng, H. (2015a). Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1433–1443. Association for Computational Linguistics.

Liu, Q., Gao, Z., Liu, B., and Zhang, Y. (2015b). Automated rule selection for aspect extraction in opinion mining. In Yang, Q. and Wooldridge, M., editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1291–1297. AAAI Press.

Luo, W., Li, Y., Urtasun, R., and Zemel, R. (2016). Understanding the effective receptive field in deep convolutional neural networks. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 4898–4906. Curran Associates, Inc.

Ma, X. and Hovy, E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, volume 1. Association for Computational Linguistics.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Marasović, A. and Frank, A. (2018). SRL4ORL: Improving opinion role labeling using multi-task learning with semantic role labeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 583–594, New Orleans, Louisiana. Association for Computational Linguistics.

McAuley, J. J., Pandey, R., and Leskovec, J. (2015a). Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

McAuley, J. J., Targett, C., Shi, Q., and van den Hengel, A. (2015b). Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5:115–133.

Mikolov, T., Corrado, G., Chen, K., and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pages 1–12.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Nair, V. and Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA. Omnipress.

Ng, A. Y. (2004). Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 78–, New York, NY, USA. ACM.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pérez-Rosas, V., Mihalcea, R., and Morency, L.-P. (2013). Utterance-level multimodal sentiment analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 973–982, Sofia, Bulgaria. Association for Computational Linguistics.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., Al-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., Clercq, O. D., Hoste, V., Apidianaki, M., Tannier, X., Loukachevitch, N. V., Kotelnikov, E. V., Bel, N., Zafra, S. M. J., and Eryigit, G. (2016). Semeval-2016 task 5: Aspect based sentiment analysis. In [Bethard et al., 2016], pages 19–30.

Pontiki, M., Galanis, D., Papageorgiou, H., Manandhar, S., and Androutsopoulos, I. (2015). Semeval-2015 task 12: Aspect based sentiment analysis. In [Cer et al., 2015], pages 486–495.

Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., and Manandhar, S. (2014). Semeval-2014 task 4: Aspect based sentiment analysis. In Nakov, P. and Zesch, T., editors, *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin,*

Ireland, August 23-24, 2014., pages 27–35. The Association for Computer Linguistics.

Poria, S., Cambria, E., and Gelbukh, A. (2015). Deep Convolutional Neural Network Textual Features and Multiple Kernel Learning for Utterance-Level Multimodal Sentiment Analysis. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2539–2544.

Poria, S., Cambria, E., and Gelbukh, A. (2016a). Aspect Extraction for Opinion Mining with a Deep Convolutional Neural Network. *Knowledge-Based Systems*, 108:42–49.

Poria, S., Cambria, E., Howard, N., Huang, G.-B., and Hussain, A. (2016b). Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomputing*, 174:50 – 59.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 147–155, Stroudsburg, PA, USA. Association for Computational Linguistics.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC Workshop on New Challenges for NLP Frameworks*, pages 45–50.

Sakaguchi, K., Duh, K., Post, M., and Durme, B. V. (2017). Robsut wrod reocginiton via semi-character recurrent neural network. In [Singh and Markovitch, 2017], pages 3281–3287.

San Vicente, I. n., Saralegi, X., and Agerri, R. (2015). Elixa: A modular and flexible ABSA platform. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*, pages 748–752, Denver, Colorado. Association for Computational Linguistics.

Schönemann, P. H. (1966). A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10.

Schouten, K. and Frasincar, F. (2015). The benefit of concept-based features for sentiment analysis. In Gandon, F., Cabrio, E., Stankovic, M., and Zimmermann, A., editors, *Semantic Web Evaluation Challenges - Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia*, volume 548 of *Communications in Computer and Information Science*, pages 223–233. Springer.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5149–5152.

Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Singh, S. P. and Markovitch, S., editors (2017). *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. AAAI Press.

Smith, S. L., Turban, D. H. P., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *International Conference on Learning Representations*.

Søgaard, A. and Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., and Wang, X. (2015). Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 1333–1339. AAAI Press.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., and Qin, B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

ter Horst, H., Hartung, M., and Cimiano, P. (2017). Joint Entity Recognition and Linking in Technical Domains Using Undirected Probabilistic Graphical Models. In Gracia, J., Bond, F., McCrae, J. P., Buitelaar, P., Chiarcos, C.,

and Hellmann, S., editors, *Language, Data, and Knowledge (Proceedings of the 1st International LDK Conference)*, volume 10318, pages 166–180. Springer.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Titov, I. and Mcdonald, R. (2008). A Joint Model of Text and Aspect Ratings for Sentiment Summarization. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 308–316.

Tjong Kim Sang, E. F. and Veenstra, J. (1999). Representing Text Chunks. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, pages 173–179. Bergen, Norway.

Toh, Z. and Su, J. (2016). NLANGP at SemEval-2016 Task 5: Improving Aspect Based Sentiment Analysis using Neural Network Features. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016*, volume 2015, pages 282–288.

Toh, Z. and Wang, W. (2014). DLIREC: Aspect Term Extraction and Term Polarity Classification System. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval)*, pages 235–240.

Upadhyay, S., Faruqui, M., Tur, G., Hakkani-Tur, D., and Heck, L. (2018). (Almost) Zero-Shot Cross-Lingual Spoken Language Understanding. In *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

van der Maaten, L. and Hinton, G. E. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Wang, W., Pan, S. J., Dahlmeier, D., and Xiao, X. (2017a). Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In [Singh and Markovitch, 2017], pages 3316–3322.

Wang, W., Pan, S. J., Dahlmeier, D., and Xiao, X. (2017b). Coupled Multi-Layer Attentions for Co-Extraction of Aspect and Opinion Terms. In [Singh and Markovitch, 2017], pages 3316–3322.

Weng, J. J., Ahuja, N., and Huang, T. S. (1993). Learning recognition and segmentation of 3-d objects from 2-d images. In *In Proceedings of the International Conference on Computer Vision*, pages 121–128.

Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University.

Wiebe, J., Wilson, T., and Cardie, C. (2005). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39:165–210.

Wiegand, M., Schulder, M., and Ruppenhofer, J. (2015). Opinion holder and target extraction for verb-based opinion predicates – the problem is not solved. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 148–155, Lisboa, Portugal. Association for Computational Linguistics.

Yang, B. and Cardie, C. (2013). Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1640–1649, Sofia, Bulgaria. Association for Computational Linguistics.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Yu, L.-C., Wang, J., Lai, K. R., and Zhang, X. (2017). Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, Copenhagen, Denmark. Association for Computational Linguistics.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In Fleet, D. J., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer.

Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation Classification via Convolutional Deep Neural Network. *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 2335–2344.

Zhang, M., Zhang, Y., and Vo, D.-T. (2016). Gated neural networks for targeted sentiment analysis. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 3087–3093. AAAI Press.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

Zhang, Y., Qi, P., and Manning, C. D. (2018). Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, Brussels, Belgium. Association for Computational Linguistics.

Zhang, Y., Zhong, V., Chen, D., Angeli, G., and Manning, C. D. (2017). Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Zhou, X., Hu, B., Lin, J., Xiang, Y., and Wang, X. (2015a). ICRC-HIT: A deep learning based comment sequence labeling system for answer selection challenge. In [Cer et al., 2015], pages 210–214.

Zhou, X., Wan, X., and Xiao, J. (2015b). CLOpinionMiner: Opinion Target Extraction in a Cross-Language Scenario. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23:1–1.