DEVELOPING A COMPREHENSIVE FRAMEWORK FOR SENTIMENT ANALYSIS IN TURKISH

by

Cem Rıfkı Aydın

B.S., Computer Engineering, Bahçeşehir University, 2011

M.S., Computer Engineering, Boğaziçi University, 2013

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Computer Engineering

Boğaziçi University

2020

DEVELOPING A COMPREHENSIVE FRAMEWORK FOR SENTIMENT ANALYSIS IN TURKISH

APPROVED BY:

Prof. Tunga Güngör (Thesis Supervisor)	.....
Assist. Prof. Tevfik Aytekin	.....
Prof. Fikret S. Gürgen	.....
Assoc. Prof. Arzucan Özgür	.....
Assist. Prof. Reyyan Yeniterzi	.....

DATE OF APPROVAL: 28.07.2020

ACKNOWLEDGEMENTS

I am more than grateful to my splendid supervisor, Tunga Güngör for his great contributions to this thesis, approaching me always with caring, support, encouragement and understanding, and invaluable guidance in both my M.Sc. and Ph.D. studies, and in my life overall.

I would like to thank the members of my thesis committee, Fikret Gürgen, Tevfik Aytekin, Arzucan Özgür, and Reyyan Yeniterzi for their invaluable feedbacks and contributions to this work. I would like to also thank my professor Arzucan Özgür again, also Haşim Sak and Cumali Türkmenoğlu for providing me with data and tools.

I would especially like to thank my family members, my mother, father, and sister, without whom I would not have succeeded in anything in life. I also thank my late aunt, my friends, relatives, professors, and every other person, whom I have met and who has contributed to my life in terms of love, sharing, learning, teaching, and making me myself.

This work was supported in part by the Boğaziçi University Research Fund (BAP) under Grant 6980D, and in part by the Turkish Directorate of Strategy and Budget under the TAM Project number 2007K12-873. The work of Cem Rıfkı Aydın was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under Grant BİDEB 2211.

ABSTRACT

DEVELOPING A COMPREHENSIVE FRAMEWORK FOR SENTIMENT ANALYSIS IN TURKISH

In this thesis, we developed a comprehensive framework for sentiment analysis that takes its many aspects into account mainly for Turkish. We have also proposed several approaches specific to sentiment analysis in English only. We have accordingly made five major and three minor contributions. We generated a novel and effective feature set by combining unsupervised, semi-supervised, and supervised metrics. We then fed them as input into classical machine learning methods, and outperformed neural network models for datasets of different genres in both Turkish and English. We created a polarity lexicon with a semi-supervised domain-specific method, which has been the first approach applied for corpora in Turkish. We performed a fine morphological analysis for the sentiment classification task in Turkish by determining the polarities of morphemes. This can be adapted to other morphologically-rich or agglutinative languages as well. We have built a novel neural network architecture, which combines recurrent and recursive neural network models for English. We built novel word embeddings that exploit sentiment, syntactic, semantic, and lexical characteristics for both Turkish and English. We also redefined context windows as subclauses in modelling word representations in English. This can also be applied to other linguistic fields and natural language processing tasks. We have achieved state-of-the-art and significant results for all these original approaches. Our minor contributions include methods related to aspect-based sentiment in Turkish, parameter redefinition in the semi-supervised approach, and aspect term extraction techniques for English. This thesis can be considered the most detailed and comprehensive study made on sentiment analysis in Turkish as of July, 2020. Our work has also contributed to the opinion classification problem in English.

ÖZET

TÜRKÇE İÇİN KAPSAMLI BİR DUYGU ANALİZİ ÇATISI GELİŞTİRME

Bu çalışmada, Türkçe için geniş kapsamlı bir duygu analizi çatısı oluşturulmuştur. Ayrıca, İngilizce'de duygu analizine özel bazı yaklaşımlar da geliştirilmiştir. Bu kapsamda bu problemin bir sürü yönü göz önünde bulundurulmuştur. Beş tane ana katkı ve üç tane küçük katkıda bulunulmuştur. Denetimsiz, yarı denetimli ve denetimli yöntemlerle orijinal ve etkili öznitelik kümeleri oluşturulmuştur. Bundan sonra, bu öznitelikler klasik makine öğrenme metotlarına girdi olarak verilmiştir ve Türkçe ile İngilizce dillerindeki veri setleri için, yapay sinir ağlarının performansı geçilmiştir. Türkçe dili için ilk defa yarı denetimli yöntemlerle duygu sözlükleri oluşturulmuştur. Sondan eklemeli Türkçe dili için detaylı bir biçimbilimsel analiz gerçekleştirilmiş ve eklerin duygu skorları tespit edilmiştir. Bu, biçimsel olarak zengin diğer dillere de uygulanabilmektedir. Tekrarlamalı ve yinelemeli sinir ağı modellerini birleştiren özgün bir yapay sinir ağı mimarisi geliştirilmiştir. Duygusal, sözdizimsel, anlamsal ve sözlüksel öznitelikler hesaba katılarak, hem Türkçe hem İngilizce için özgün kelime vektörleri oluşturulmuştur. Ayrıca, bağlam pencereleri yancümlecik olarak belirlenerek İngilizce için kısmen orijinal bir yöntemle kelime vektörleri modellenmiştir. Bu, diğer dilbilimsel alanlara ve doğal dil işleme alanlarına uyarlanabilmektedir. Bütün bu özgün yaklaşımlar için, ölçüt olarak alınan çalışmaların başarı oranları geçilmiştir. Küçük çaplı katkılarımız, Türkçe için yön bazlı duygu analizi, yarı denetimli metot için parametrelerin değiştirilmesi ve İngilizce'de duygu analizi için yorumlarda yönlerin tespit edilmesi olarak belirtilebilir. Bu tez, Temmuz, 2020 itibariyle Türkçe için geliştirilmiş en kapsamlı çalışma olarak atfedilebilir. Bu çalışma aynı zamanda İngilizce'de duygu analizi alanı için önemli katkılarda bulunmuştur.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES x
LIST OF TABLES xii
LIST OF SYMBOLS xv
LIST OF ACRONYMS/ABBREVIATIONS xvi
1. INTRODUCTION 1
1.1. Sentiment Analysis Applications 1
1.2. Sentiment Analysis Research 2
1.2.1. Levels of Analysis 2
1.2.2. The Use of Sentiment Lexicons 3
1.3. Opinion Spamming Detection 5
1.4. This Thesis 6
1.4.1. Motivations, Research Questions, and Contributions 6
1.5. Outline of the Thesis 11
2. SENTIMENT ANALYSIS AND ITS BASIC CONCEPTS 13
2.1. Opinion Definition 13
2.2. Extraction of the Opinion Information 15
2.3. Types of Sentiments 17
2.3.1. Regular and Comparative Opinions 17
2.3.2. Explicit and Implicit Opinions 18
2.4. Author and Reader "Confliction" 18
3. RELATED WORK 20
3.1. A General Overview of Sentiment Analysis at Word- and Document-Level 20
3.1.1. Unsupervised Approaches 20
3.1.2. Semi-Supervised Approaches 21
3.1.3. Supervised Approaches 21

3.2. Studies on Sentiment Analysis in Turkish 24
3.3. Morphological Analysis for Sentiment Analysis 24
3.4. Aspect-Based Sentiment Analysis 26
3.4.1. Aspect Extraction from Reviews 28
3.4.2. Aspect-Based Polarity Detection 28
3.4.2.1. SemEval-2014, Task 4 - Aspect Polarity Detection . . . 29
3.4.2.2. Recurrent Neural Network Models 29
3.4.2.3. Recursive Neural Network Models 30
3.4.2.4. Rhetorical Structure Models 31
3.4.2.5. Ensemble Models 32
3.5. Word and Document Embeddings for Sentiment Analysis 35
3.6. Context Windows for Word Embeddings and Other NLP tasks 37
4. METHODOLOGIES AND EXPERIMENTATIONS 39
4.1. Preprocessing and Feature Extraction Methods 40
4.1.1. Basic Preprocessing Techniques 41
4.1.2. Additional Preprocessing Step: Partial Surface Forms 47
4.2. Major Contribution #1: Unsupervised, Semi-Supervised and Supervised Approaches, and Their Combination 50
4.2.1. Unsupervised and Semi-supervised Approaches 51
4.2.1.1. Domain-Independent 52
4.2.1.2. Domain-Specific 54
4.2.2. Supervised Approaches 60
4.2.2.1. Conventional Machine Learning Models and Feature Sets 61
4.2.2.2. Neural Network Approaches 62
4.2.3. Combining Unsupervised and Supervised Feature Metrics 65
4.2.4. Experimental Evaluation 66
4.2.4.1. Datasets 66
4.2.4.2. Hyper-parameters 69
4.2.4.3. Results 69
4.2.5. Conclusion and Future Work 77

4.3. Major Contribution #2: Morphological Analysis for Sentiment Analysis in Turkish . . . 79
4.3.1. Results . . . 79
4.3.2. Conclusion . . . 81
4.4. Major Contribution #3: Combining Recurrent and Recursive Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations . . . 82
4.4.1. Methodology . . . 85
4.4.1.1. Submodel 1: Recurrent Model . . . 85
4.4.1.2. Submodel 2: Recursive Model . . . 86
4.4.1.3. Ensemble Framework Combining Recursive and Recurrent Models . . . 95
4.4.2. Experimental Evaluation . . . 96
4.4.2.1. Datasets . . . 97
4.4.2.2. Hyper-parameters . . . 97
4.4.2.3. Results . . . 98
4.4.3. Conclusion and Future Work . . . 102
4.5. Major Contribution #4: Generating Word and Document Embeddings for Sentiment Analysis . . . 103
4.5.1. Methods Developed . . . 105
4.5.1.1. Corpus-Based Approach . . . 105
4.5.1.2. Dictionary-Based Approach . . . 107
4.5.1.3. Supervised Contextual 4-scores . . . 109
4.5.1.4. Combination of Word Vectors . . . 110
4.5.1.5. Creating Document Embeddings . . . 110
4.5.2. Corpora . . . 111
4.5.3. Experiments . . . 112
4.5.3.1. Preprocessing . . . 112
4.5.3.2. Hyper-parameters . . . 113
4.5.3.3. Results . . . 113
4.5.4. Conclusion . . . 116

4.6. Major Contribution #5: Redefining Context Windows as Subclauses for Word Vector Models . . . 117
4.6.1. The Proposed Approach . . . 118
4.6.2. Experimental setup . . . 119
4.6.3. Datasets . . . 120
4.6.4. Results . . . 120
4.6.4.1. Context Window Type . . . 122
4.6.4.2. Window Size and Orientation . . . 122
4.6.4.3. Word Embedding Size . . . 123
4.6.4.4. Stop Word Removal . . . 123
4.6.5. Conclusion . . . 123
4.7. Minor Contributions . . . 124
4.7.1. Minor Contribution #1: Semi-Supervised Approach with Tweaked Parameters . . . 124
4.7.2. Minor Contribution #2: Aspect Term Extraction for English . . 124
4.7.3. Minor Contribution #3: Aspect-Based Sentiment Classification for Turkish . . . 125
5. CONCLUSIONS AND FUTURE WORK . . . 127
5.1. Publications Obtained in this Thesis . . . 132
REFERENCES . . . 134

LIST OF FIGURES

Figure 4.1. The flowchart of the proposed model. 50

Figure 4.2. Algorithm for unsupervised approach. 56

Figure 4.3. The visual summary of the semi-supervised approach [1]. We tweaked the parameters of this approach and improved the success rates when evaluating the model on Turkish and English datasets. 58

Figure 4.4. Algorithm for semi-supervised approach. 60

Figure 4.5. Algorithm for supervised approaches. 65

Figure 4.6. Algorithm for the combination of unsupervised and supervised approaches. 67

Figure 4.7. Effects of fine-grained morphological analysis with respect to different top (most discriminative) morpheme percentages on the Turkish movie corpus. 80

Figure 4.8. Effects of fine-grained morphological analysis with respect to different top (most discriminative) morpheme percentages on the Turkish movie corpus. 81

Figure 4.9. The architecture of the baseline study [2]. AASR denotes *Aspect-Aware Sentence Representation.* 86

Figure 4.10. A sample review parsed into its chunks/phrases along with their polarity categories, which range from very negative to very positive (--, -, 0, +, ++). 88

Figure 4.11. Example of a sentence decomposed into its relationships. 90

Figure 4.12. The text *"The vibe is very relaxed and cozy, service was great and the food was excellent!"* decomposed into its sub-reviews. Each encircled part in the figure is a sub-review. 93

Figure 4.13. Visual summary of our proposed ensemble framework. 96

Figure 4.14. The impact of leveraging the labels of tokens as well in the dictionary-based approach. This figure visualises how words with the opposite polarities get far away from each other in the VSM. . 109

Figure 4.15. The flowchart of our system. 111

Figure 5.1. Visual summaries of the five major contributions. 127

LIST OF TABLES

Table 3.1. Summary of the related works covered thus far. 27

Table 3.2. Summary of the related works on ABSA. 34

Table 4.1. A sample tweet in Turkish and the impacts of each preprocessing technique throughout the pipeline. 46

Table 4.2. Different seed words of opposite polarities chosen manually for the domain-independent and the domain-specific methods. 55

Table 4.3. Contribution of each weighting scheme to the performances (%) of the supervised approaches for the two datasets. 70

Table 4.4. Summary of performances (%) of the unsupervised, semi-supervised and supervised methods (using the 3-feats technique), and their combinations for the two datasets. 72

Table 4.5. Performances (%) for two neural network models using different word embedding types for the two datasets. 73

Table 4.6. Contributions of preprocessing modules to the performance (%) of the SVM method using the delta tf-idf metric. 75

Table 4.7. Summary of performances (%) of the unsupervised, semi-supervised and supervised methods, and their combinations for the three datasets in English. 76

Table 4.8. Summary of the impacts of fine-grained morphological analyses on the success rates (%) for the two Turkish corpora with respect to the 3-feats scheme and the SVM method. 81

Table 4.9. Example sentences and the sub-sentences for each aspect term built employing the constituency and dependency parsers. The gold aspect terms are underlined. 92

Table 4.10. Details of the datasets we used in this approach. 97

Table 4.11. Hyper-parameters made use of in the recursive and recurrent neural networks. *"Optimal size"* corresponds to the optimal size of embeddings at intermediary nodes in trees. 98

Table 4.12. Hyper-parameters made use of in the baseline model. 98

Table 4.13. Accuracies (%) obtained for the baseline method and our ensemble approach which integrates the recursive neural network model into the baseline recurrent network. Results for different embedding size and other settings are listed. 100

Table 4.14. Accuracies (%) that are obtained in the baseline method (IARM) and our ensemble framework. SA stands for single aspect scenario, MA for multiple aspect scenario. 101

Table 4.15. Performances (%) for different features used as input in the SVM model when detecting the polarities of documents. The baseline approach is chosen as the word2vec algorithm. 114

Table 4.16. Most similar words to given queries with regard to the corpus-based approach and the baseline algorithm (word2vec). 116

Table 4.17. Possible values chosen as hyper-parameters for the experiments. . . 120

Table 4.18. Accuracy (%) across all models and metrics with various hyper-parameters trained on different corpora. 121

LIST OF SYMBOLS

a_{ij}	Aspect j of entity i
c_s	Coefficient of the supervised score
c_u	Coefficient of the unsupervised score
$cos(.)$	Cosine similarity
$\mathbf{E}$	Edge matrix representing cosine similarities between words
e_i	Entity i
$f_{w,d}$	Frequency of the word w in document d
g	Target constituting of entity and aspect
$\mathbf{g}$	Global propagation factor in the semi-supervised approach
h_k	Opinion holder k
$\mathbf{M}$	Positive pointwise mutual information matrix
$\mathbf{M}_{i,j}$	Positive pointwise mutual information between $word_i$ and $word_j$
N	Corpus of positive polarity
N'	Corpus of negative polarity
$\mathbf{P_P}$	Vector of positive polarity word scores
$\mathbf{P_N}$	Vector of negative polarity word scores
s_{ijkl}	Sentiment expressed towards the aspect j of entity i by the person k in the time period l
$\mathbf{s}$	Seed polarity vector used in the semi-supervised approach
t_l	Time l
$\mathbf{U}$	Real or complex unitary matrix of the singular value decomposition
$\mathbf{v}$	Vocabulary
$\mathbf{V}$	Real or complex unitary matrix of the singular value decomposition
$\mathbf{\Sigma}$	Diagonal matrix of the singular value decomposition

LIST OF ACRONYMS/ABBREVIATIONS

3-feats	Minimum, mean, and maximum sentiment scores per review
AASR	Aspect-Aware Sentiment Representation
ABSA	Aspect-Based Sentiment Analysis
BOW	Bag-of-Words
CML	Classical Machine Learning
CNN	Convolutional Neural Networks
combSC	Combination Score
CRF	Conditional Random Fields
C-RNN	Convolutional Neural Networks - Recurrent Neural Networks
DNNs	Deep Neural Networks
IARM	Inter-Aspect Relation Modelling
idf	Inverse Document Frequency
IR	Information Retrieval
kNN	k-Nearest Neighbour
LDA	Latent Dirichlet Allocation
LS	Log Scoring
LSA	Latent Semantic Analysis
LSTM	Long Short-Term Memory
MA	Multiple Aspects
MWE	Multi-Word Expression
NB	Naïve Bayes
NER	Named Entity Recognition
NLP	Natural Language Processing
OOV	Out-of-Vocabulary
PMI	Pointwise Mutual Information
PPMI	Positive Pointwise Mutual Information
POS	Part-of-Speech
R-CNN	Recurrent Neural Networks - Convolutional Neural Networks

RecNN	Recursive Neural Networks
RNN	Recurrent Neural Networks
RST	Rhetorical Structure Theory
SA	Single Aspect
SC	Sentiment Score
SVD	Singular Value Decomposition
SVM	Support Vector Machines
TDK	Türk Dil Kurumu (Turkish Language Institution)
tf-idf	Term Frequency—Inverse Document Frequency
URL	Uniform Resource Locator
VSM	Vector Space Model

1. INTRODUCTION

Sentiment analysis is the field of study, where subjective information in source materials is identified and extracted, referring to the use of natural language processing (NLP), machine learning, text analysis and computational linguistics. This sentiment classification task is heavily employed in many fields, such as marketing, social media analyses, and customer service, making use of reviews. Relevant studies in the literature have mostly been conducted for the sentiment analysis problem in English and other most commonly spoken languages. There is a lack of bulky datasets and sentiment lexicons in Turkish to be useful for this classification problem. Hence, this topic can be considered immature for Turkish and this gap needs to be filled by generating comprehensive datasets and lexicons. Also, developing approaches to generating even more robust and novel machine learning algorithms, and feature sets for this language may be helpful, which can be adapted to other languages and even other NLP problems as well.

In this thesis, we present several contributions, which make our work one of the most detailed analyses performed on the sentiment classification task for Turkish. Our several approaches are cross-domain and also portable to other languages. We also developed some models specific to sentiment analysis in English only, which may be applied to different languages with minor changes. A brief introductory overview of this domain is given below. We have not modelled all of the below scenarios in our study. Our contributions will be discussed in Section 1.4.1.

1.1. Sentiment Analysis Applications

Sentiment analysis has been one of the hottest and most alluring topics in the recent decades in NLP, since it helps us understand and analyse the trends, movements, product likeability, and several other implicit and explicit factors at play. By using the techniques specific to this domain, companies can increase their profits through

marketing analyses or even political parties may take actions, as was the case in the Arab Spring [3]. In the past, when people needed to ask an opinion, they used to consult their friends or families. On the other hand, when an organisation needed to gather consumer or public opinions, they resorted to conducting surveys and opinion polls. However, due to the explosive growth of the social media (e.g. Facebook and Twitter) people are mostly making decisions, based on what they come across in the reviews, blogs, and posts. However, extracting sentiments from these kinds of texts manually on a large scale is virtually impossible. This is why automated sentiment analysis techniques have to be developed. While data on the Web is mostly utilised for detecting polarities of the reviews, organisations employ the internal data as well for this task. For example, these can be the data collected from call centres, e-mails sent to them by their customers, or other feedback media, such as surveys. Taking into account the importance of this domain, many start-up companies are built, and several companies, such as Microsoft and Google, have built their own in-house capabilities [4].

1.2. Sentiment Analysis Research

Sentiment analysis, as an NLP task, has become popular in the recent years, and is also widely being researched in the Web mining and information retrieval (IR) fields of study. This task has in fact started spreading from computer science domain into management sciences. The following subsections represent the overall aspects and characteristics of sentiment analysis.

1.2.1. Levels of Analysis

Different levels in sentiment analysis can be taken into consideration, which are document-level, sentence-level, and entity-levels, in a coarse-to-fine order. In document-level analysis, the whole review, which can consist of one or several sentences, or even several paragraphs, is assigned a sentiment score or a polarity label. On the other hand, in sentence-level analysis, factual information (i.e. objective sentence(s)) is distinguished from the subjective information [5]. In entity-level analysis,

aspects in the sentence(s) are assigned sentiment scores or labels. For example, the below sentence is given to exemplify this process:

"Samsung S20+ is packed with a good screen resolution and an awesome video quality overall. However, I observed that it has a bad call quality."

Here, the screen aspect and video quality features are each assigned a positive opinion. However, it is the opposite case for the call quality. Another challenging task is that we can prefer using comparative opinions over regular opinions. In regular opinions, a comment may be made only on an aspect or an entity. On the other hand, in the comparative type, two aspects or entities are compared to each other, where disambiguation may occur. For example, a reviewer may make a comment like:

"I have recently bought an iPhone and my girlfriend preferred to buy an Android phone despite my contrary opinions. However, I should confess that the sound quality of her smartphone has turned out to be much better than that of mine."

Here, the aspect *"Android phone's sound quality"* is represented by the word *"hers"*, and it is a laborious task to extract that information [6]. Therefore, techniques from different NLP areas can also be employed to help extract the corresponding target aspects and their polarities.

1.2.2. The Use of Sentiment Lexicons

Sentiment lexicons are some of the major resources we may make use of when extracting sentiments from source materials, since mostly unigram words represent opinions. To name a few, *"good"*, *"excellent"*, and *"amazing"* are positive words. On the other hand, some other words, such as *"terrible"*, *"poor"*, and *"repulsive"* are of negative polarity. Nevertheless, taking into account only unigrams may be considered

insufficient, since idioms may represent a sentiment independently of its constituent words, such as *"cost someone an arm and leg"*. Another problematic issue is that some words may express a positive sentiment for a domain and a negative sentiment for some other domains. For example, the word *"suck"* carries a negative sentiment in the sentence *"This camera sucks!"* Nonetheless, the same word is of the opposite polarity (i.e. positive) in the sentence *"This vacuum cleaner sucks well."* Another word exemplifying this case can be *"loud"*, which may have a positive sentiment in a comment made on a headphone. But it could be of negative polarity in a car review, as in *"The car I have just bought is too loud and has not met my expectations."*

Another challenging issue related to this domain is as follows. Even if a sentiment word appears in a sentence, this does not exactly mean that it has a sentiment. For example, in the sentence *"Do you know any good laptop I can buy?"*, the reviewer does not express a positive or negative opinion. In the other example *"If you did not like the screen quality of your phone, sell it on eBay and buy this Huawei phone with an impressive AMOLED screen instead."*, a positive sentiment is expressed about the Huawei smartphone. However, a negative comment is in fact not made on the phone the user is currently using (the owner of the phone may be thinking negatively about it). Thus, interrogative and conditional sentences may bring problems along with it, and much further research is needed in this field.

We can detect the polarity of a review through the words, which appear in sentiment lexicons or training corpora. Additionally, it is also possible that an opinion may express a sentiment without the use of a polarity word. For instance, in the sentence *"This washer uses up a lot of water."*, a polarity word does not appear, while it clearly has a negative opinion. Also, in *"Most of the people in modern society sustain a life with copy-pasted and learnt emotions instead of innate feelings."*, a word of negative polarity does not occur. However, in fact, the word *"copy-pasted"* indirectly and implicitly expresses a negative sentiment. Identifying such cases is difficult.

Another issue that is prevalently observed in this domain is the use of sarcastic opinions. For instance, in the review *"OMG, I am so impressed with the car I have just bought! It stopped working just in two days."*, a positive sentiment seems like to be expressed in the first sentence when taking account of the word *"impressed"*. However, we can observe that this is not the case when we look at the second sentence. Sarcasm generally occurs in political discussions, but not as frequently in product or service reviews. This problem is one of the hardest tasks to handle in the sentiment classification task.

1.3. Opinion Spamming Detection

Social media has enabled people to express their ideas and opinions without needing to disclose their true identity. This way of expressing a person's own views is highly valuable, since we may be informed of the "true" sentiments and what he or she "really" thinks. However, we should not overlook the fact that people may fake identities so as to make undesirable comments about products. The plausible reasons may be to discredit or promote a target product, service, or organisation. For instance, even fake profiles are being created to defame the products of fierce competitor companies. A profile generated for a mobile phone company can defame the product of another cell phone company in several ways, although most of the customers of the second company can be satisfied with nearly all of the characteristics of this product [7]. This problem can be solved by analysing the posting behaviours of individuals. For example, a reviewer may give a score of 10/10 to almost all the products of a company. The usage of these products may not be closely related to each other (e.g. computers, mobile cell phones, televisions, refrigerators). Also, if the same reviewer gives rather low scores to the products of a competitor company, irrespective of what the product is, it could be considered a suspicious act. Another case would be that in which an individual posts the same (or "weirdly", very similar) negative review for all the computer products of a corporation. Even several tactics are followed in generating spam opinions. For example, so as to make a fake comment (of positive or negative polarity) the top comment, fake ratings can also be generated.

That is, other fake profiles can favour that review and *"find it useful"*. Another way to detect spamming is that in which we analyse how frequently and periodically an individual makes reviews or gives star ratings to the comments of other reviewers. If he or she makes many (specifically similar) comments about the product of a company in a span of a few seconds, that can be thought of as another suspicious behaviour.

1.4. This Thesis

Given the background, in this thesis, we present a comprehensive framework for sentiment analysis in Turkish. We propose several approaches to performing document-level and aspect-level sentiment analysis in Turkish. We perform a novel morphological analysis for this agglutinative language and boost the classification performance. We adapted our several novel methods to English as well and achieved state-of-the-art results for both languages. We have also developed a novel aspect-based polarity detection framework for English only and have outperformed all the participating teams in the most reputable competition on sentiment analysis, which has taken place in 2014. Additionally, we define novel word vector models that uses the sentiment information in the reviews. Apart from these, we also propose a novel rule set for defining context windows, which are not only adaptable to sentiment analysis, but also to other NLP tasks. These are all explained briefly below.

1.4.1. Motivations, Research Questions, and Contributions

In this work, we mainly addressed the problems in the sentiment classification task for Turkish. Although several studies for sentiment analysis in Turkish handle some aspects of it, these are not as comprehensively taken into account as in English and other widely-used languages. In our study, our main goal has been to fill this gap for Turkish by developing novel approaches and achieve better performances than the existing approaches. We list the main contributions and the corresponding research questions below.

(i) *Combination of Unsupervised and Supervised Features:* In this sub-approach, we generate a feature set which combines unsupervised, semi-supervised, and supervised techniques. We simply assign averaged scores to each word, whereby supervised scores are weighed more heavily compared to semi-supervised scores. Using only these word scores and after generating three polarity scores (mean, maximum, and minimum values) for each review, we feed these features as input into an ensemble classifier. Accordingly, we achieve the best results and even outperform neural network models for several corpora of different genres and languages. This proves that classical machine learning algorithms fed with effective feature engineering techniques can yield better results than deep learning frameworks. Employing semi-supervised scores instead of unsupervised scores and combining them with supervised features yields the best results.

The two *research questions* we addressed for this sub-approach are as follows. Can we achieve a better performance when combining supervised, semi-supervised, and unsupervised techniques? Is it possible that classical machine learning models fed with effective features perform better than neural network approaches?

(ii) *Morphological Analysis for Turkish:* Turkish is an agglutinative language, whereby morphemes attached to the root forms of words can also be indicative of polarity in sentiment analysis and other classification tasks. Hence, we performed a fine-grained analysis, in which morphemes are also assigned polarity scores relying on the labels of reviews. These reviews consist of words, which are in surface forms. Incorporating the sentiment information of these morphemes as well helps us boost the performance even more rather than when making use of only the root or surface forms of words. We think that this sub-approach can be adopted for other morphologically-rich and agglutinative languages, and other NLP tasks as well with minor changes.

The corresponding *research question* is: Does comprehensive morphological analysis contribute to the sentiment classification task compared to using only the root or surface forms of words for morphologically-rich languages?

(iii) *Combining Recurrent and Recursive Neural Networks for Sentiment Analysis Using Inter-Aspect Relations:* Most of the works employ either recurrent or recursive

neural network models when performing aspect-based sentiment analysis (ABSA). Recurrent neural networks (RNN) can capture and represent the temporal information. On the other hand, recursive neural networks (RecNN) can model the grammatical and syntactic information more robustly. However, only a few studies combine these two models in an ensemble form. In this approach, we extend a baseline study, which employs only a recurrent model. Their work performs aspect-based sentiment analysis, whereby aspects are provided in advance. They also model inter-aspect relations. That is, it is assumed that the sentiment of an aspect can affect those of preceding and following aspects. Our approach does not only use this recurrent model, but also incorporates recursive model into the framework. We achieve this by splitting each review into sub-reviews (or subclauses) in an original way and training them separately by utilising the sentiment information. We then incorporate these vectors containing sentiment, grammatical, and syntactic information into the baseline recurrent network. By relying on this ensemble neural network model, we outperform the baseline study with a significant margin for two domains.

Research questions addressed for this approach are as follows. Can we enhance recurrent neural networks by including distant sentiment information? Can recursive and recurrent neural network models be merged in an ensemble framework? If so, would it boost the performance? What would be the reason for an ensemble form to perform better than non-ensemble frameworks of recurrent and recursive neural network models? Can reviews be partitioned into sub-reviews, which contain only the relevant sentiments about the aspect(s)? Which one can be considered a better-performing and robust parsing method for the sentiment classification task, constituency or dependency parsing?

(iv) *Generating Sentiment and Semantic Embeddings:* Most studies in the literature use off-the-shelf word vectors, such as word2vec [8] or GloVe vectors [9], when implementing NLP tasks. These embeddings can represent the syntactic and semantic information of tokens. However, these kinds of vectors cannot capture the sentiment information as robustly. Therefore, in this approach, we generate several novel embeddings techniques, which incorporate the sentiment information

into the vector models. When we feed these embeddings into a sentiment classifier, we achieve the best performances for several corpora, which are of different genres and in two different languages. We also observe that, in several cases, even employing unsupervised techniques can lead to better performances than when utilising supervised methods in creating word vectors. We empirically show that this original approach is cross-lingual and portable to other languages as well.

Here, the relevant *research questions* are as follows. Does including the sentiment information in word embeddings lead to better performances rather than when incorporating only co-occurrence statistics? Is it possible for vectors generated by unsupervised techniques to outperform those created with supervised methods in a classification task?

(v) *Generating Context Windows for NLP Tasks:* In the literature, context windows are generally defined as sliding windows, which consist of a limited number of words, around the target words. In this approach, we propose a novel rule set for generating sub-sentences as context windows and employ this information when generating word vectors. Accordingly, we define the context window to be subclauses, in which a word occurs, rather than to be a fixed number of words in their left and right contexts. Thus, words in the same subclauses are in general more related to each other compared to those in others. We utilise this context window information when modelling the off-the-shelf embedding algorithms, which are the word2vec and GloVe methods, and then feed these vectors into two classifiers. These two NLP classification tasks are sentiment analysis and spam detection problems. We outperform the baseline approach, which makes use of the sliding windows technique, for these tasks. That is, this approach is adaptable to several embedding generation models and other NLP tasks as well. We think that this window context definition can also be used in other models, such as convolutional neural networks (CNN), in lieu of sliding windows.

The related *research questions* are as follows. Can we redefine context windows for them to model the sequences more robustly compared to sliding windows of fixed-length? Are the words in the same subclause are syntactically and semantically more similar to each other compared to those in sliding windows, even if they are

physically more distant in the first technique? If so, why?

In summary, in this thesis, these are the five main contributions for the sentiment analysis task. These are not only restricted to proposing novel approaches for Turkish and specific domains, some of them are also contributive to languages other than Turkish, are cross-domain, and can even be adapted to other NLP tasks with minor changes. In addition to these five contributions, we can list three minor contributions as follows.

(i) *Generating Polarity Scores Using Search Engine Method and Tweaking the Parameters of a Semi-Supervised Approach:* Polarities of words can be detected by querying search engines. In this respect, using syntactic relations can help identify the sentiment and semantic similarities between words. In a study, seed polarity words are defined in advance and the sentiment scores of other words are detected by their co-occurrence statistics. For example, if the word *"awesome"* co-occurs with the word *"good"* more frequently rather than with the word *"annoying"* in context windows (e.g. sliding windows), this word can be assigned a positive score. Several different syntactic contexts can be defined in this respect. For example, words of the same polarities are more frequently connected to each other with the conjunction *"and"*. In our study, we define several syntactic relations specific to Turkish when querying the search engine and employ them. That is, our contribution in this sub-approach is minor in this regard.
In a baseline study, domain-specific polarities are computed by using statistical information. In their work, researchers define a seed set of words, which are specific to a corpus, and then find the sentiments of other words in a corpus by employing propagation methods. The motivation behind their study is that some words may have opposite polarities for different domains. For example, the word *"unpredictable"* may have a negative sentiment in the automobile domain, as in *"This car has an unpredictable steering."* On the other hand, the same word may express a positive polarity for movie reviews, as in *"This film's plot was really unpredictable. This totally got me!"* In our approach, we have adopted this

method as well and adapted it to Turkish by performing minor changes. To our knowledge, our work is the first to employ this semi-supervised approach for the sentiment classification task in Turkish. When we change several formulae when applying the propagation technique, we achieved better results for this language. However, apart from it, we have not made a broad contribution to this approach. *Research question* addressing the two above sub-approaches is: What is the effect of employing sentiment lexicons generated in a semi-supervised manner on the performance of the classifier?

(ii) *Aspect Term Extraction Method for English:* In this thesis, we also extracted aspects from raw text in English for sentiment analysis. We generated a set of features to identify whether or not a target word is aspect. Among these features are the GloVe embeddings of words, part-of-speech (POS) vectors, and several others as will be discussed later. When we fed the concatenated forms of these word vectors into a support vector machine (SVM) classifier, we achieved a similar performance as that of the baseline. The features we employed alone are not very novel and our contribution for this sub-approach in this respect can be considered minor.

(iii) *Aspect-Based Sentiment Analysis in Turkish:* In this sub-approach, we extract aspects from a corpus in Turkish for the sentiment classification task. However, it is mostly the same as in a baseline study [10]. Our minor contributions in this respect are that (1) we take into account negation, amplifiers and downtoners and (2) we rely on sentiment lexicons generated by a semi-supervised approach when assigning polarities to aspects. These two techniques are not original in their way, since they are adapted to the sentiment classification task in English. However, to the best of our knowledge, they have not been adapted to Turkish thus far in such a fine-grained analysis.

1.5. Outline of the Thesis

The remainder of the thesis is organised as follows. In Chapter 2, we give a brief overview on sentiment analysis and its main concepts. In Chapter 3, we present and

discuss the existing works on different approaches used in the sentiment classification task for Turkish and other languages. We describe the proposed approaches and show the experimental results in Chapter 4. We also discuss the main contributions of the proposed approaches in the same section. We incorporate these different parts of the thesis work into this same section, since we developed several different comprehensive approaches and we do not want to discuss experiments in a faraway chapter. That is, we include the results of experimentations and their contributions immediately after each corresponding sub-approach. In Chapter 5, we conclude the paper.

2. SENTIMENT ANALYSIS AND ITS BASIC CONCEPTS

If we analyse a problem by breaking it down into sub-problems, we would understand the relevant topic better with respect to its underlying structure. By doing so, researchers can create a more maintainable, consistent, accurate, and robust framework, which can provide solutions and be enhanced more easily later. Therefore, several major sub-components of the sentiment analysis domain and the main problems existing in the current techniques related to this domain are discussed below.

2.1. Opinion Definition

In order to represent the definition of what is an opinion, the below example will be referred to:

"(1) I bought this iPhone about six months ago. (2) I definitely loved it. (3) Its screen resolution is fascinating and is what I have been looking for for ages. (4) Its battery life is also long. (5) Although I liked it so much, my sister keeps telling that it is too lousy."

The important points about this review are as follows:

(i) In the second sentence, a positive sentiment is expressed about the phone overall. In the third sentence, an opinion of positive polarity is stated about its screen resolution. Sentence (4) expresses another positive sentiment towards its battery life. Lastly, in sentence (5), a negative opinion is stated. Hence, the following observation can be made.

Observation: An opinion can consist of two components, which are a target g and a sentiment s, that is,

$$(g,\ s),$$

where g can be an entity, or an aspect of an entity, whereas s can be the sentiment expressed as negative, neutral or positive, or as a rating score, which typically ranges from 1 through 5 (or 10), in increments of 0.5 or 1. In sentence (2), target is the entity iPhone, whereas in sentence (4), it is an aspect (i.e. battery life) of the entity iPhone.

(ii) The sentiments in this review are expressed by two people, called opinion holders or opinion sources. In sentences (2-4), the opinion holder is the person who makes this review, whereas in sentence (5), it is the sister of the reviewer.

(iii) The exact date of this review is stated explicitly (*"six months ago"*) and this information is also leveraged in the sentiment classification task. This important aspect can be taken into account, when keeping track of the trends. Here, we can detect how differently people can react regarding a topic in various periods. For example, reviewers may start making negative reviews on some politicians after some corruption scandals come out, or a wave of demonstrations and civil unrest, such as *"Gezi Parkı"* protests, happen. These all can be used in the context of social and political analyses.

To summarise these, the following definition can be referred to:

Definition (Opinion): An opinion is a quadruple consisting of four key components as follows:

$$(g,\ s,\ h,\ t),$$

where g stands for target, s for sentiment, h for the opinion holder, t for the time when the opinion was expressed.

Although sentence (3) just makes mention of screen resolution, the entity referred to is *"iPhone's screen resolution"*. A target can therefore be broken down into an entity and an attribute thereof, and be represented as a pair as follows:

(iPhone, screen resolution)

Entity can be defined as the target object as a whole and its aspects. A relevant example to this is given above. Aspects are generally considered to have a meronymy relation in the sense that these are parts of the entity. Aspects are also called features. However, it should not be confused with feature concept used in the machine learning domain. In this regard, the opinion definition can be refined as follows:

$$(e_i,\ a_{ij},\ s_{ijkl},\ h_k,\ t_l),$$

where the only difference to the previously defined set is that the aspect component is added and the description is more fine-grained. Among these subscripts/modules, i corresponds to a specific entity, j to an aspect, k to an opinion holder, and l to a specific time, when the opinion is expressed. For instance, s_{ijkl} stands for the sentiment expressed towards the aspect j of the entity i by the person k in the time period l. It is a more sensible notation, since we may be interested in detecting which components of a product are of good quality and liked, and which features are not. For example, many people can make a positive comment on the call quality of Google Pixel's new smartphone, but they can also express negative sentiments on its camera quality. Hence, the aspect components and their implicit or explicit polarities should be analysed separately. The companies may thereby start ameliorating the disliked or flawed features of their products, based on a finer analysis of the reviewers' comments.

2.2. Extraction of the Opinion Information

First, the opinion quintuples $(e_i,\ a_{ij},\ s_{ijkl},\ h_k,\ t_l)$ need to be extracted from each review (or called document). The first key component we have to obtain is entity, which can be performed by named entity recognition (NER) or other several techniques [11–13]. A problematic scenario here is that an entity name can be written in different ways. For instance, *"McDonald's"* can be expressed as *"Mc"*, *"McDo"*, etc.

Specific normalisation techniques can be employed in this case, which is another NLP task. Secondly, aspects have to be extracted, where different tools, such as dependency parsers, can also be utilised [14]. The following definitions can be referred to while carrying out the aspect extraction task:

Definition (explicit aspect): Those are the aspects that can be expressed by nouns or noun phrases. For instance, *"battery life"* in *"Battery life of this laptop is rather long."* represents an explicit aspect.

Definition (implicit aspect): Those are the aspects, which are not defined by nouns or noun phrases. For example, in *"This camera cannot fit in a pants pocket."*, when expressing the phrase *"fit in the pocket"*, the reviewer refers to the size of the camera.

Out of the remaining three components (s_{ijkl}, h_k, t_l), opinion holders also play a key role when performing social analyses. For example, the location of the reviewers (a metadatum about the opinion source) may be detected and this information can be used and analysed in the case of political turmoils or other significant events. For instance, when Russia had invaded Crimea in 2014, most of the negative reviews have been made by the Ukrainians, whereas the supportive comments were mostly made by the Russian people. Polarities and their target aspects are considered the main components in the sentiment analysis domain, since the services, organisations, or people may be informed of whether or not people are, in general, dissatisfied with their products or some of their features in general. The below example is given to exemplify these processes:

"My friend adores his camera and its picture quality."

The following quintuples are generated from it:

(Canon, GENERAL, positive, Cem's friend, 08/05/2020)

(Canon, picture quality, positive, Cem's friend, 08/05/2020)

Disambiguation occurring here stems from the use of *"its"*, which can be solved by the anaphora resolution task in NLP. That is, we should know that it refers to the Canon camera.

2.3. Types of Sentiments

Opinions, which have thus far been discussed are called regular opinions. Additionally, comparative opinions are another type of opinions. Based on how they are expressed, these can be called explicit or implicit opinions.

2.3.1. Regular and Comparative Opinions

Regular opinions are those, in which a comparison does not exist. The two subtypes are direct and indirect opinions. The first type is easy to handle, since the opinion is directly expressed on the entity or its aspects, as in *"The sound quality is amazing."* Analysing the latter comparison type is more difficult compared to the former type, since it is not as clearly stated. In the sentence *"After injection, my elbows felt much better."*, the effect of the injection indirectly made the patient feel better. However, some disambiguation may occur in indirect opinions, as in the sentence *"Since I had headaches, the doctor put me on these drugs."* Here, there is not a sentiment expressed on the drugs, because these are prescribed after the headaches started happening.

Comparative opinions, as it is clarified by its name itself, express ideas, which represent the similarities or differences between entities or their target aspects. These can be in comparative forms, as in *"better"* or *"worse"*, or superlative, as in *"best"* or *"worst"*. Apart from these, verbs are also sometimes utilised to express these kinds of opinions, as in *"I prefer the books of Nietzsche over those of Jean-Paul Sartre among*

modern philosophers."

2.3.2. Explicit and Implicit Opinions

As mentioned, explicit opinions can be regular and comparative opinions. They clearly express what the reviewer intends to say, as shown in the following examples:

"Lipton Ice Tea tastes great."

"Nestea tastes even better than Lipton Ice Tea."

Implicit (or called implied) opinions imply the sentiments, which are not voiced explicitly. This is exemplified by the following comments:

"This ultrabook has a long battery life."

"However, this laptop has a longer battery life than the other one."

In comparison to the explicit opinions, much less research has been conducted for the implicit opinions [15, 16].

2.4. Author and Reader "Confliction"

A reviewer may make a positive comment on something, but other readers may perceive it negatively. For instance, in the sentence *"I am so happy that housing prices went down."*, those prospective readers who are likely to rent or buy an apartment may consider this comment as a positive review. On the other hand, for home sellers, the implied sentiment would be negative. In another example, which is *"Trump is most likely going to be re-elected in 2020, I cannot help being in euphoria!"*, a positive

opinion is likely indicated for those who are pro-neoliberal globalisation, but not for social democrats.

This chapter was given to provide readers with a general overview about the sentiment classification task. Among the sub-modules discussed here, we have not taken into account comparative opinions, opinion holder and time components in this thesis. Only regular opinions, entities, aspects, and polarities discussed in this chapter have been leveraged in this thesis. Additionally, we have developed several other approaches, which will be given in Chapter 4 in detail.

3. RELATED WORK

Since we have performed a comprehensive sentiment analysis, where we take its several aspects into account in this thesis, we discuss the related works in separate subsections based on their topics to make them more readable for the reader. We categorised them with respect to the different aspects of the sentiment analysis problem (their application in different languages, the effects of using different machine learning models, feature sets, etc.). Although opinion holder and time components for the sentiment analysis problem are described and explained in Chapter 2, we do not review any study here, which models these scenarios. The reason is that we have not performed and implemented these scenarios in this thesis. This should be noted that, in the following subsections, several related works can be repeatedly discussed and overlap, because they handle more than one component of the quintuple model and subtasks of the sentiment classification domain.

3.1. A General Overview of Sentiment Analysis at Word- and Document-Level

In this section, we discuss related works, which mostly perform sentiment analysis at word- or document-level, and employ machine learning algorithms or simpler statistical methods. Most of these conduct binary or ternary sentiment classification, in which a finer analysis (e.g. aspect-based analysis) is not performed. Most of those studies focus on opinion mining in English.

3.1.1. Unsupervised Approaches

Several studies predict the polarities of sentences or reviews based on the sentiment orientations of words, which are generated through unsupervised techniques. In [17], sentiments of words and phrases are detected by relying on a search engine. Then, sentiment of a document is predicted by averaging all the scores of the words

therein. The performance is reported to be better for car reviews (80-84%) than for movie reviews (66%). We also adopted the same approach by performing minor changes. We have tried out several different query words and operators, and evaluated this technique on the movie and Twitter datasets in Turkish. It is also reported that making use of static corpora may be more robust and less erratic compared to the use of search engines [18]. We also utilised static corpora for our semi-supervised approach, albeit not for the unsupervised approach. In [19], a graph of adjective words is generated by employing conjunctional relationships. For example, if two words are connected with the conjunction *"and"* too frequently, they are assumed to be closer in the graph. Lastly, they perform clustering and assign each word to either positive or negative class in the end.

3.1.2. Semi-Supervised Approaches

Annotators can specify a limited number of seed sentiment words and their effect can be propagated across a graph built of corpus words in a semi-supervised manner. In [1], several sentiment words specific to each genre are chosen and a graph is generated in the sense that words that co-occur frequently have weightier links, that is, they are closer to each other. The sentiment scores of these seed words propagate through this graph and those nodes who are closer to these seeds are assigned similar polarities. The main advantage of this approach is that domain-specific sentiment lexicons are induced in lieu of generic polarity lexicons, such as SentiWordNet [20]. In [21], unsupervised and supervised approaches are combined by using polarity lexicons. However, their approach does not generate domain-specific lexicons as we do.

3.1.3. Supervised Approaches

Although there is a large body of work, which relies on the use of unsupervised and semi-supervised approaches, the majority of the related studies employ supervised techniques at document-level. Most of the supervised approaches utilise Boolean and term frequency—inverse document frequency (tf-idf) techniques as features when mak-

ing use of classical machine learning methods [22,23]. Apart from these, the delta tf-idf technique proves to be more effective, since sentiment information is taken into account on a word-basis [24]. In this thesis, we model these scenarios as feature engineering techniques as well, in addition to several others. In [25], three polarity scores are extracted from each review, and these are fed as input into machine learning classifiers. We also employ a similar technique in that we induce the minimum, mean, and maximum sentiment scores from each document. We thereby achieve the best performances with this effective feature set. Reference [26] generates a feature set, which consists of character n-grams and several other metrics, and empirically show that it solves the data sparsity problem for sentiment analysis for the Twitter datasets. Several other features (e.g. POS tags and the number of emoticons or emojis) are also leveraged in another study [27].

It is reported that punctuation marks and repeated letters in the reviews boost the strength of polarities of the immediately preceding sentiment [28]. In [29], bigrams are stated to capture modified verbs and nouns. Hence, modelling this scenario produces better results for this classification task compared to the bag-of-words (BOW) metric. Another study [30] takes into account the presence and absence of specific words for aspect-based sentiment classification. For instance, conditional words and the presence of *wh* words are mostly indicative of expressing a negative polarity.

In [31], a fine-grained sentiment analysis is performed employing the financial microblogs and news headlines datasets. For this domain-specific approach, they employ four feature sets, which are linguistic features, sentiment lexicon features, domain-specific features, and word vectors. They choose punctuation marks, numbers, and metadata as domain-specific features. For example, if a word is preceded by the "-" mark, it is indicative of a negative polarity for the finance domain. They achieve the best results when relying on ensemble regression classifiers. We instead only choose a limited number of seed words in this thesis and do not define any domain-specific rules for each review. In [32], authors create sentiment-specific vectors, which are also language-independent. They utilise a supervised approach over the word2vec model

layer. Emoticons and emojis are processed in a distant-supervised manner such that reviews are automatically annotated. They outperform the word2vec model by also incorporating this sentiment information into their approach. Sentiment lexicons are induced by propagating the effects of those polarity labels across the graph.

SVMs are generally the best-performing classical machine learning model, since these can avoid the overfitting problem through the use of the kernel trick. The other reason is that these are defined by a convex optimisation problem, in which local minima do not exist [29]. However, deep neural networks (DNNs) have more recently gained much more popularity, since they provide better models in terms of accuracy, efficiency, and flexibility compared to classical machine learning classifiers and regressors. Although it can take a great effort and intense time for a person to manually generate features which are fed into classical machine learning classifiers, DNNs extract features automatically. As inputs, in general, word embeddings, such as word2vec, are employed. In a study [33], sentiment-aware vectors are generated by using the sentiment information of movie reviews. The word2vec embeddings capture only the syntactic and semantic characteristics of words. Therefore, incorporating sentiment information into those word representations as well is reported to boost the classification performance. We also combine unsupervised, semi-supervised, and supervised features on a word-basis. However, in the corresponding approach, we generate scalar values, not sentiment-aware multi-dimensional embeddings. In [34], emojis are generated by using a distant supervision technique, whereafter a bi-directional long short-term memory (bi-LSTM) is employed to conduct multi-class sentiment analysis. In the same study, sarcasm is also handled. Reference [35] builds a framework, in which message-level and topic-based sentiment analyses in a hybrid form are performed. In their study, a two-layer bi-LSTM is utilised and an attention mechanism is employed over the last layer for the message-level sentiment analysis. In the topic-based classification module, a *"Siamese"* bi-directional LSTM framework is modelled by using context-aware attention. No sentiment lexicons or hand-crafted features are relied on in their study. However, they rank first (tie) in the Subtask A of the SemEval-2017 competition.

3.2. Studies on Sentiment Analysis in Turkish

For the sentiment classification task in Turkish, most of the studies in the literature employ supervised techniques. In [36], political columns in Turkish are subjected to the classification task, using n-gram language model, maximum entropy, naïve Bayes (NB), and SVM approaches. The performances they obtain range from 65% to 77%. In another study [37], active learning is employed for sentiment analysis for the Twitter data in Turkish. They use a similar weighting technique as we have done. In [38], the researchers translate a polarity lexicon in English into Turkish. They also utilise several supervised methods. In their study, employing presence and absence suffixes in Turkish increases the accuracy of their models. The accuracy they obtain is 89.5% in their study. Other studies on sentiment analysis in Turkish, which rely on unsupervised or semi-supervised approaches, translate sentiment lexicons in English into Turkish [39] or induce sentiment lexicons using corpus statistics [40]. However, the approach that generates polarities based on the use of search engines [17] has thus far not been implemented for the Turkish language to the best of our knowledge. Our study, in this respect, is the first to use this technique as well. Apart from these, most of the studies in the literature take into account only the root forms of words and a limited number of affixes (e.g. the negation morpheme) for Turkish. Nonetheless, we conduct a fine-grained morphological analysis for sentiment analysis in Turkish and leverage every morpheme attached to the root forms. That is, we generate polarities for each morpheme. Our approach can be adapted to other agglutinative languages and even to other NLP tasks with minor changes.

3.3. Morphological Analysis for Sentiment Analysis

In a study [41], for the opinion mining problem in Arabic, inflectional and derivational morphemes are employed in addition to the root forms of words. With this comprehensive feature set, they have outperformed the baseline model, which only utilises the stems of words as features. In [42], authors conduct sentiment analysis in the Hindi language by relying on a three-stage system. In their study, if there is an annotated

dataset in this language, they perform classification by using supervised techniques. If not, they translate the reviews into English and conduct the classification task in this language. If none of those two cannot be applied, they induce a sentiment lexicon also by taking into account the morphologically-rich structure of the Hindi language and perform classification by utilising a majority voting scheme. In [43], when classifying movie reviews, the researchers use only seed tokens and Chinese morphemes that can be mono-syllabic characters. They do not employ any external sentiment resources. None of these studies feed the surface forms of words as input into their classifier models, since they report that this can increase the size of features to a great extent and can thereby hamper the performance. They instead choose a specific subset of morphemes (mostly suffixes) as features in addition to root forms. In our study, we also expand our feature set by taking into account all morphemes at first. However, after detecting the polarities of these morphemes, we keep the most discriminative ones and eliminate those which do not express much of a sentiment.

A sentiment classification framework is developed for the Korean language in [44]. In their work, they employ morphemes, which express sentiment, in addition to the root forms for this morphologically-rich language. They feed the movie reviews as input into the sentiment analysis model and leverage several sentiment lexicons. Additionally, a chunking technique which is based on the dependency relationships between morphemes is used in their study. For example, negation morphemes have an impact on the sentiments of words and morphemes only in the same sub-chunks. Similarly, in our study, we only utilise a subset of suffix set, which consists of the most sentimentally informative morphemes. We also leverage negation-shifters in addition to other contextual features. In [45], sentiment classification is conducted for the Sinhala language, which is also a morphologically-rich language. In his study, the researcher analyses the impact of the selection of specific adjectives and adverbs on the classification performance. Several weighting schemes are employed to predict the polarity scores. In another study [46], sentiment lexicons for the Sinhala language are generated. In that study, the researcher models a three-stage scenario as follows. First, a cross-lingual approach is followed such that a sentiment lexicon in English is

translated into Sinhala. Second, morphological structures are handled and the lexicon's features are expanded. Third, a graph-based approach is proposed by employing the polarity lexicon in the source language (i.e. English). In this study, the researcher defines rule-based methods, where negations and intensifiers are handled. This study is reported to be adaptable to other morphologically-rich or agglutinative languages as well. In this thesis, we do not translate the sentiment lexicons into Turkish. We instead follow a semi-supervised approach, which requires only a minor effort to be made by researchers. This is adaptable to other languages and domains with minor changes. Here, only a set of seed words specific to a domain has to be determined in advance. We also determine the polarities of morphemes in Turkish by employing two corpora in this language [47].

We summarise the related works, which have been covered up to this point in Table 3.1 with respect to their methods and feature engineering techniques. Here, we include only the settings that lead to the best performances. In this table, except [1], all the studies conduct the binary sentiment classification task. We show only a subset of the related works in the table, since there is a large overlap between these studies in that most of them employ the same or very similar features and methods. We only intend to make the reader to get an overview here. However, most of these studies are not directly comparable, since these are evaluated on datasets of different genres and in various languages.

3.4. Aspect-Based Sentiment Analysis

Polarities are, in general, predicted on a review/document basis. However, it is also possible to predict the labels of aspects. In another scenario, aspects and their corresponding orientations can be identified together in a hybrid approach. Most studies leverage statistical methods, knowledge-based techniques, or their hybrid combinations in this regard [48].

Table 3.1. Summary of the related works covered thus far.

Study	**Language**	**Techniques**	**Domain**	**Performance (%)**
[38]	Turkish	tf-idf + SVM	Movie	89.50
[17]	English	Co-occurrence statistics through the use of search engine (unsupervised)	Movie	66.00
[1]	English	Domain-specific (semi-supervised) for ternary classification	Sports	63.10
[36]	Turkish	n-gram + SVM	Politics	77.00
[24]	English	Delta tf-idf + SVM	Movie	91.26
[33]	English	Sentiment-aware vectors + Logistic regression	Movie	88.90
[41]	Arabic	Sentiment lexicon + Morphological analysis	Newswire	73.43
[44]	Korean	Morphological analysis + Parsers	Movie	94.75

In this section, relevant and recent studies, which conduct the aspect-based sentiment classification task, are discussed. Again, these are given in separate subsections based on different underlying machine learning models and structures. There are two scenarios in this case: 1. Aspects can be extracted from the reviews. 2. Only the polarities of the given aspects can be detected. We model the second scenario comprehensively in this scenario [49].

3.4.1. Aspect Extraction from Reviews

In the literature, it is stated that aspects and their sentiments can be learned separately or in a joint model [50]. In [51], aspects are extracted by employing a 7-layer deep CNN model. Features fed as input into the system are word embeddings, POS tags, and also several linguistic patterns. In another study [52], the semantic meanings of target aspects are captured and these model aspect representations. The researchers utilise an LSTM model over their framework, which also takes into account the syntactic information. Reference [53] leverages target-level and sentence-level attention mechanism to predict aspect words from the reviews. They also incorporate common sense knowledge into their framework. In this thesis, we also tried to identify aspects in English reviews. We generate a limited number of features for each word, such as POS embeddings, a Boolean value indicating whether or not the word is a noun, and a few others. We then fed them as input into an SVM classifier. Although we outperformed the simple baseline approach, this approach we developed is not very original and the results are not much significant and state-of-the-art. That is, we cannot say that we have made a significant contribution in terms of the aspect extraction task.

3.4.2. Aspect-Based Polarity Detection

In this section, we review studies, which predict the polarities of the aspects given in advance. That is, most of these studies do not focus on extracting aspect words from reviews. The papers are discussed in separate subsections based on the models and structures used. Since we evaluate our aspect-based sub-approach only on

SemEval-2014, Task 4 datasets, we first give an overview about this task.

3.4.2.1. SemEval-2014, Task 4 - Aspect Polarity Detection. Task 4 of the SemEval-2014 competition has four tasks, which are (1) aspect term detection, (2) aspect term polarity prediction, (3) aspect category detection, and (4) aspect category polarity detection [54]. For this approach, we have only focussed on aspect term polarity prediction by evaluating our methods on the laptop and restaurant datasets [55]. 26 teams have participated in this subtask. Both of the two top-performing teams [56, 57] generate features, such as parse trees and n-grams, and feed them as input into an SVM classifier. In [56], private sentiment lexicons are used. One of them is an in-domain sentiment lexicon, which is generated through the Amazon laptop reviews. The others are out-of-domain polarity lexicons, comprehensive Twitter sentiment lexicons and three manually-curated lexicons. On the other hand, [57] makes use of publicly available sentiment lexicons. Some of the words in the lexicon are eliminated by the authors. Some words are also manually added to the lexicons, and these are adapted to the restaurant and laptop datasets. In our corresponding approach, we rely on deep neural networks and not on hand-crafted feature engineering techniques. That is, we do not conduct feature engineering tasks for this approach. None of the participating teams develop a novel ensemble neural network framework as we do.

3.4.2.2. Recurrent Neural Network Models. In the sentiment analysis problem, many studies employ recurrent neural network models, since they can capture the sequential model and temporal information effectively. As mentioned, those models can help predict the label of preceding or succeeding words more successfully as compared to other neural model approaches. LSTM and gated recurrent units (GRUs) are the two of the most preferred recurrent neural network models [58].

The baseline study for this approach [2] conducts the aspect-based sentiment classification task by employing inter-aspect relations. They only predict the labels of the given aspects. That is, they do not detect the aspects separately. To model

this scenario, they propagate the effect of sentiments through the text. They average aspect term embeddings in each aspect group and they feed these into a bi-directional GRU model. In our approach, we develop a novel ensemble neural network model, such that recursive and recurrent neural models are combined. Recurrent model as in the baseline is employed to capture temporal knowledge. We enrich it with a recursive neural model, such that syntactic, grammatical, and distant sentiment information are also incorporated into the model.

In [59], a recurrent neural network model with an attention mechanism is relied on for aspect-based sentiment analysis. In that study, the sentiments of target aspects are predicted by employing position-weighted memory and recurrent attention memory. They perform better than the baseline approaches. Reference [60] leverages layer-wise relevance propagation in recurrent neural network models. The authors apply a limited number of rules for the propagation process. They conduct five-class sentiment classification and outperform the baseline methods. In [61], a novel model is proposed, in which aspect representations are merged with those of their contexts using an attention mechanism. In another study [62], authors perform sentiment analysis for tweets in Spanish. They generate a hybrid approach such that sentiment information from polarity lexicons are incorporated into a recurrent neural network model (LSTM). We also rely on sentiment lexicons, but not in the same manner. In lieu of using the sentiment lexicon scores in a recurrent neural network, we train constituency and dependency parse trees by using a recursive neural network model. Then, we employ the root vectors of these trees in the ensemble framework. Reference [35] employs a 2-layer bi-directional LSTM network for the message-level sentiment detection task. The researchers boost their performance by using an attention mechanism over the last layer. A Siamese bi-directional model is also used for the detection of topic-level sentiments. Their approach rank first in the SemEval-2017 competition, Task 4 *"Sentiment Analysis in Twitter"*.

3.4.2.3. Recursive Neural Network Models. In contrast to recurrent neural networks, RecNNs can model the syntactic, grammatical, and other relational representations in

the text. In [63], the authors decompose reviews into their chunks using a constituency parser and the Stanford sentiment treebank (SST). Each node in the trees can be assigned a sentiment score, whose value ranges from 0 (very negative) to 4 (very positive) in increments of 1. The values at these nodes are updated while training the recursive tree models. This model is more representative and robust than the BOW or n-gram techniques, since it can capture the structural relations effectively (e.g. subclauses can be affected with the conjunctions *"although"* and *"however"*). These can play a role in shifting the sentiment of a phrase, subclause, or the whole sentence. We leverage this model when generating our recursive model with a constituency parser. In [64], a dependency parser is employed for the detection of sentiments. They propose a model, which is especially adapted to morphologically-rich languages (e.g. Polish and Turkish). Their proposed approach is based on ternary sentiment classification. A joint model co-extracts aspects and opinions in another study [65]. The researchers employ a recursive neural network model and the conditional random fields (CRF) approach. They arrive at state-of-the-art results in this study. They model a scenario, in which discriminative features are employed and sentiment information is propagated between related opinions and aspects through a dependency parser. The authors also make use of hand-crafted feature engineering techniques to have a better performance. Reference [66] defines a specific set of rules to combine constituency and dependency parsers. They create target-dependent binary phrase dependency trees. The researchers predict the sentiments of aspects by training these ensemble forms of neural models and evaluate their approach on the SemEval datasets. We also combine recurrent and recursive neural network models, but in a different manner. We do not merge the output of the two models. We instead train those models separately and feed the outputs of the recursive sub-model per aspect term group as input into the recurrent sub-model. In advance, we also model a novel scenario, in which a set of rules are used to generate sub-review(s) per review.

3.4.2.4. Rhetorical Structure Models. Although recursive and recurrent neural network models can represent the syntactic, semantic, and sentimental characteristics of a review or the aspects therein, these may function on a sentence-basis only. If a

document/review consists of two or more sentences, rhetorical structures can model the scenario for the sentiment classification task more effectively and comprehensively. In these structures, nuclei are considered the main components of the text, whereas the satellites are secondary, contributive to the nuclei. In [67], rhetorical structures are employed, where polarities of words are given in advance by relying on sentiment lexicons. In these structures, nucleus spans are weighed more heavily in terms of the sentiment score and they use a genetic algorithm to determine the optimal weights of their model.

In [68], the authors use a parser called *"Sentence-level PArsing for DiscoursE"* to perform polarity detection at document-level. Several rhetorical relations are reported to have more significance for the sentiment analysis problem. They also state that some specific relations, such as the contrast relation, may shift the overall polarity of the review. Another study [69] models rhetorical structure theory (RST) for sentiment classification by employing sentiment lexicons and propagating polarities across sentences or even paragraphs. They use the *"HIgh-Level Discourse Analyzer"* when conducting the sentiment classification task on a review-basis. In [70], the researchers identify the sentiments of words through pointwise mutual information. For that goal, they query search engines. An SVM classifier is relied on to detect on- and off-topics, after which a weighting scheme is employed. In their study, negation, intensifying, and downtoning are handled as well. A decision tree algorithm is used by them to conduct sentiment classification at document-level in the end.

3.4.2.5. Ensemble Models. In ensemble frameworks, two or more sub-models are combined such that these each compensate for what the others lack. In [71], first, a convolutional neural network approach is employed over the word embedding layer. Accordingly, the context information of words are captured and these are convolved into vectors. Over this layer resides a recursive neural network that uses a constituency parser. In this sub-model, grammatical, syntactic, semantic, and sentimental characteristics are modelled, taking the convolved embeddings as input. They evaluate their methods on the SST. In our approach, we first decompose each review into sub-reviews

for each aspect group. We then train the recursive neural network model by relying on constituency and dependency parsers. Lastly, we feed their inputs separately into the recurrent gated recurrent unit model.

In [72], a bi-LSTM model is trained and then a CNN model is applied over this layer (R-CNN). Also, the opposite mechanism is leveraged (C-RNN). The authors combine these two methods (i.e. R-CNN and C-RNN) under the name of *"fusion gates"*. That is, both temporal and local features are both taken into account in their study. In another study [73], CNN and LSTM frameworks are separately trained and the authors compute the averaged probability scores of the outputs generated by these sub-models. The cut-off threshold value between positive and negative sentiments was set as 0.5. We do not follow such an approach. We instead use the outputs of recursive neural trees per aspect group as the sentiment embeddings in the recurrent model. Reference [74] proposes a model, in which, first, CNN is applied to capture the local information. Then they employ an LSTM model over this layer. The effect of each word is propagated through the sequence. They state that this approach can be expanded for use in other NLP tasks as well.

Apart from the ensemble frameworks discussed above, some studies model hybrid approaches combining ontology-based learning schemes with deep learning neural networks. A study [75] uses conceptual values. These conceptual values refer to the three cases that are (1) polarity orientation (positive or negative), (2) aspect mention (e.g. *"atmosphere"* is associated with the aspect category/concept *"ambiance"*), or (3) polarity mention (e.g. the word *"cheap"* has a negative connotation in the sentence *"This place has a cheap atmosphere."*, whereas the opposite applies in *"It has a cheap price."*). If these conceptual frameworks cannot model the sentiment information robustly, the authors use bi-LSTMs. They can therefore capture the most informative words in the right and left context windows of a target word or phrase. In [76], a similar approach is followed. A work [77] ameliorates this hybrid model by enriching it with regularisation parameters and several CNN layers.

We summarise these related works on ABSA in Table 3.2. We do not include the studies conducted only on aspect extraction here, because we have not carried out an aspect detection study comprehensively in an original way. However, as mentioned, our model that identifies the polarities of the given aspects is novel. Only some of these works are covered in this table to give the reader an overview. They mostly perform ABSA. They rely on different approaches, such as classical machine learning methods, recursive and recurrent neural network models, RST, ensemble, or hybrid approaches. These all evaluate their methods on corpora in English. However, these results are not directly comparable, since datasets can be of different genres. CML in the table stands for classical machine learning approaches.

Table 3.2. Summary of the related works on ABSA.

Study	**Approaches**	**Dataset**	**Performance (%)**
[56]	*CML:* *n*-gram, public lexicons + SVM	Laptop	70.48
[57]	*CML:* *n*-gram, public lexicons + SVM	Laptop	70.50
[67]	*RST:* Rhetorical structures + sentiment lexicons	Restaurant	60.00
[70]	*RST:* Rhetorical structures + sentiment lexicons + topic classifier	Movie	76.00
[66]	*Recursive* neural networks combining dependency and constituency parsers	Restaurant	66.20
[65]	*Recursive* neural networks + CRF	Restaurant	84.11

Table 3.2. Summary of the related works on ABSA. (cont.)

Study	**Approaches**	**Dataset**	**Performance (%)**
[2] (*baseline*)	*Recurrent:* Inter-aspect relations with LSTM	Restaurant	80.00
[35]	*Recurrent:* 2-layer LSTM and Siamese bi-LSTM	Twitter	86.30
[72]	*Ensemble:* LSTM + CNN	SST	85.80
[71]	*Ensemble:* CNN + recursive neural networks	SST	90.39
[76]	*Hybrid:* Lexicalised domain ontology + neural attention	Restaurant	86.31
[77]	*Hybrid:* Lexicalised domain ontology + regularised neural attention	Restaurant	87.10

3.5. Word and Document Embeddings for Sentiment Analysis

In the literature, it is reported that employing dense word vectors is a better choice than utilising sparse embeddings in most of the NLP tasks. Priorly, latent semantic analysis (LSA) was proven to be a robust word representation model. In [78], this approach is employed to model word vectors that takes into account co-occurrence statistics. These are reported to perform better than the sparse vectors models by a significant margin for many classification tasks [79]. However, more recently developed word embeddings, such as word2vec and GloVe, can model the word representations better. These vectors are constructed through shallow neural networks or co-occurrence

statistics. Although a large body of studies, which take into account semantic and sentimental aspects of words, exist for English, this topic is immature for the Turkish language.

In addition, latent Dirichlet allocation (LDA) is employed in a study [80] to generate the mixture of latent topics. However, these modelled scenarios focus on extracting latent topics and the generation of word meanings is ignored. Therefore, more robust embeddings, which model the semantic, syntactic, and sentiment information of words, have more recently been developed as given below.

Several studies [81–83] leverage the sentimental characteristics of words when modelling word embeddings. In [33], a framework relies on both semantic and sentimental aspects in generating word representations. That study employs the star ratings of reviews for the sentiment information. A study [1] induces a sentiment lexicon by leveraging domain-specific co-occurrence statistics. They outperform the word2vec when they prefer the singular value decomposition (SVD) technique over the word2vec model.

A study [84] on generating word embeddings for the sentiment classification task employs hand-crafted features. After they create word vectors, they produce document embeddings and feed these into the SVM classifier to predict the polarities of the whole document (review). We outperform this baseline study by generating more effective word vectors by using both unsupervised and supervised feature sets. In [85], the researchers encode sentiment information into word vectors employing a hybrid approach, which combines a CNN model and external sentiment lexicons.

Most studies in the literature employ distant supervision techniques to generate sentiment-aware vectors for English. In a study [34], emojis are leveraged as a sentiment and emotion source, and a bi-LSTM model is relied on to create word vectors. Another study [86] creates sentiment-aware embeddings by using the contextual and sentiment information of words. In that study, emoticons help capture the sentimental aspect of a

nearby word. Our embedding models do not use shallow or deep neural networks. We instead employ contextual, lexical, supervised, and unsupervised techniques. However, we outperform neural networks by a significant margin. Our approach is portable to other languages and can be adapted to other domains as well [87].

3.6. Context Windows for Word Embeddings and Other NLP tasks

Distributional word vector models can represent words by taking into account several factors such as semantics, syntax, and even sentiments. They can encode information more robustly and densely as compared to sparse embeddings [88]. These vectors are fed as input into machine learning approaches in many natural language processing tasks to better capture the grammatical and semantic structures of the texts [89–92]. The two of the most widely-used word vector models are GloVe and word2vec, which employ co-occurrence statistics. The word2vec approach relies on a shallow neural network and negative sampling techniques. On the other hand, the GloVe model generates matrices based on the corpus statistics and matrix factorisation methods.

There are only a few studies focussing on the definition of context windows used in word vector models. Performance for word representation models is affected more by the differences in hyper-parameters rather than the specific algorithm used [93]. An approach developed in a study [94] is very relevant and similar to ours in that they also generate subclauses when modelling word representations. However, the dependency rule set they use when creating these subclauses is different to ours. In [95], the effects of word embeddings generated by dependency parsers are investigated with respect to functional and domain similarities among other factors. Count-based approaches are generally outperformed by distributional semantic approaches [96]. In [97], hyper-parameters of context windows (e.g. left or right windows, cross-sentential contexts) are tested and evaluated. A grid search is performed to find the optimal context window size for the speech recognition task in [98]. In [49], sub-reviews from reviews are extracted using a dependency parser for the sentiment classification task. We utilise the

same approach to defining subclauses in sentences. However, different from that work, we use the subclauses when training word embedding models. The SVD technique is reported to outperform the word2vec approach [1, 87] in modelling word vectors. We use both GloVe and SVD models by defining context windows as subclauses, which has not previously been applied in exactly the same manner to the best of our knowledge.

4. METHODOLOGIES AND EXPERIMENTATIONS

In this thesis, we have developed many approaches, which are adapted to several aspects of the sentiment classification task. Our methods are also applicable for different languages with minor changes. Therefore, we describe our approaches and discuss the corresponding experimentations and results in separate subsections.

Before we introduce our approaches, we give detailed explanations on our preprocessing and feature extraction methods employed differently for the Turkish and English languages. We then elaborate on these aspects specific to the following, corresponding approaches when needed. Before conducting the sentiment classification tasks for the two languages, we have to perform two main different preprocessing methods. On one hand, since Turkish is an agglutinative language, we preprocess tokens using different libraries specific to Turkish, taking into account its linguistic patterns. A specific preprocessing approach we combined with a supervised metric, which we call *"partial surface forms"* and will be explained later, is original for sentiment analysis in an agglutinative language. On the other hand, since English is not a morphologically-rich language, the corresponding processes take much less effort. Besides these different preprocessing techniques for these two languages, the several proposed approaches are language-independent and can be adapted to various other languages. Most of our preprocessing techniques and approaches are cross-lingual provided that similar grammatical libraries (morphological analyser, lemmatiser, dependency parser, etc.) exist for a target language.

Thereafter, we explain the five main proposed approaches in separate sections, since these each have contributions in various aspects. Several of these are broken down into sub-approaches, since they are broad in focus. We discuss the several datasets used for each of these frameworks. A little repetition occurs throughout the explanations of these different approaches for the corpora. There are two reasons why we do not create a separate datasets section as we do for the preprocessing methods. One of them

is that we mostly employ different corpora in two different languages and of various genres. Some of these sets are specific to only one aspect of the sentiment classification task, such as ABSA. The other reason is that we again do not want to explain the datasets far away from the described frameworks and aspire to make the narrative more effective and consistent. Here, we do not separate minor and major contributions made by us, since some of these are dependent on each other. The reason is that that would fragment the narration to a great extent and make the whole presentation more incomprehensible.

We also elaborate on the hyper-parameters in the respective subsections. In the corresponding experimentations and results subsections, we present detailed analyses of our evaluations. We point out our minor or major contributions. We discuss the pros and cons of our described frameworks and we conjecture how they can be adapted to other corpora and languages, and even to other NLP tasks.

4.1. Preprocessing and Feature Extraction Methods

Before applying our approaches, we perform several basic preprocessing and feature extraction schemes for datasets in Turkish and English. These are important since, in the sentiment analysis problem, the overall opinions of reviews can be predicted based on sentiment words, which are extracted through these methods. Among these are techniques based on unsupervised or supervised characteristics of raw texts. For example, the term frequency and tf-idf features are popular engineering techniques, which indicate how important a token is with respect to a document. On the other hand, intensifiers and downtoners can increase or decrease the effect of a word they modify. Also, schemes like delta tf-idf exploit the label information of data. In addition to these widely-used metrics, we also develop a novel feature engineering technique, which generates a polarity lexicon for Turkish morphemes only. We explain all of these and others in detail below.

4.1.1. Basic Preprocessing Techniques

The feature set that we employed in our framework for the sentiment classification of reviews is as follows:

- *Term frequency:* A token's frequency is indicative about the importance of that token. As in many types of classification tasks, term frequencies are widely utilised in the sentiment analysis problem. A word's frequency is indicative about the importance of this word. We leveraged the tf-idf technique as a variation of term frequency.
- *POS tag:* Several part-of-speech tags can be more informative than the other tags for this classification task. For example, adjectives are in general considered to be the key sentiment-bearing words. Nonetheless, all POS tags may be taken into account as well. We made use of both of these scenarios when generating the feature set. Various morphological analysis and disambiguation tools could be made use of for different languages to identify the POS tags of the document tokens.
- *Sentiment words and phrases:* As said, adjectives are useful in expressing sentiments. However, verbs (e.g. *"love"*), adverbs (e.g. *"hilariously"*), and out-of-vocabulary (OOV) words (e.g. *"9/10"*), may also function as features. Additionally, we can also employ phrases to express opinions (e.g. *"cost someone an arm and a leg"*). Therefore, we used them all to assess and evaluate how these affect the performance of our framework.
- *Sentiment shifting:* Negators can lead the sentiment to shift. For instance, in *"I did not like it much."*, a negative polarity is expressed, although the review has a positive polarity word, such as *"like"*. However, employing the negation word (*"not"*) does not always necessitate the sentiment to change, as in *"I adored not only her elegance, but also her intelligence."* In such a case, the opinion word *"adore"* is still assigned a positive connotation. The Turkish negators (be it words or suffixes) are employed for opinion shifting.

- *Opinion rules:* Several rules can be applied when defining a set of sentiment features. For example, when *"less"* or *"more"* is used in the text, the opinion word following it can be assigned a different score. Another example is that, if an entity consumes a resource in large quantities, as in *"This washer uses up a lot of water."*, a negative score would be assigned to the document. On the other hand, if it rather contributes to the production, a positive score would be assigned. We applied a set of intensifiers that cause a relative increase or decrease in the scores of the words.

Using all the words except those whose frequency is below a threshold value improved the performance for several cases. However, for a metric (i.e. 3-feats), all words with a non-neutral score contributes to the success rates, no matter what their raw frequencies are. In the feature selection and extraction stages, we performed the preprocessing operations as follows:

- When spelling a text in the Turkish language, people can type English characters for the corresponding Turkish characters from time to time, such as u (*"guzel"*) for ü (*"güzel"* (beautiful)). We rely on the Zemberek tool [99] for the correction of such characters.
- Emoticons, such as *":)"* and *":("*, are not filtered out in this study, because these are indicative of polarity. If the constituent characters of these emoticons are repeated in a row, we normalise them in such a way that we do not lose this extra information (e.g. *":(((("* is normalised as *":(("*). So as to be able to capture all of these kinds of emoticons, we use a specific set of regular expressions.
- We filter out all punctuation marks except *"?"*, *"(!)"*, and *"!"*, since these do not affect the overall sentiment of a document. If a commentator uses the exclamation mark(s) repetitively in a row (e.g. *"?!"*), a negative polarity is likely to be intended to be expressed. For example, in *"And you say that she is graceful??!!"*, the commentator emphasises a negative aspect about an entity, which is a woman in this case.

- As in the repetition of punctuation marks, we think that the words, in which a repetition of characters (e.g. *"müthişşşş"* (greatttt)) occurs, are utilised for an emphasis and we boost their polarity scores. Additionally, we increase the scores of uppercased words. Nonetheless, if all the words of a document are in uppercase form, we hypothesise that there is no emphasis for specific tokens and there is no extra information to exploit for a specific case of the opinion mining task.
- After conducting the above normalisation processes, we additionally use the İTÜ NLP tool [100] to detect other unnormalised tokens which the mentioned methods cannot find. On the other hand, for the English datasets, we tokenise and lemmatise all words using the spaCy library [101]. As mentioned, since English is not an agglutinative language, we do not need to perform a comprehensive morphological analysis.
- We then use the morphological parsing [102] and disambiguation tools [103], and obtain the morphological analysis of every token. In the techniques which we employ in this thesis, we take account of three scenarios for the words: (1) root form, (2) surface form, and (3) partial surface form (Section 4.1.2). Since Turkish is a morphologically-rich and agglutinative language, it is possible that the removing morphemes can impact the performance negatively. We observe the effects of these morphemes in the sentiment classification problem by employing these three different metrics.

In addition to the above preprocessing techniques, we perform intensification, negation handling, and stop word removal when we generate the feature set. The negator feature set for Turkish is defined as follows.

- The word *"değil"* (not) changes the sentiment/semantic orientation of the word it immediately follows. For example, the polarity of the word *"hoş"* (nice) in the review *"Hoş değil bu şarkı."* shifts from positive to negative.
- Several verbal suffixes in Turkish (e.g. *"ma/me"*) can switch the sentimental and semantic orientation of the verbs. For instance, *"Beğendim."* (I liked.) has a positive sentiment, whereas *"Beğenmedim."* (I did not like.) is of the opposite

orientation (i.e. negative).

- The word *"Yok."* (There is not.) makes the preceding word "absent". For example, in the text *"Umut yok artık."* (There is no hope any more.), the effect of the word *"umut"* (hope) is neutralised when followed by the word *"yok"*.
- The suffixes *"sız", "siz", "suz"*, and *"süz"* are other negators used in the Turkish language. For instance, *"ümitsiz"* (desperate) has a negated meaning, whose polarity switches from positive to negative.

The morphological parser and disambiguation tools we employ finds out if a negation suffix is attached to a word root. If a negation word or suffix is detected, we simply append an underscore at the end of the roots of these negated words in the classical machine learning methods. For instance, if the word is *"beğenmedim"* (I did not like), the corresponding feature is stated as *"beğen_ "*. In the case that negation words or morphemes appear explicitly in a sequence, the effect of the negation is eliminated. For instance, in *"şekersiz değil"* (it is not sugarless), a negation suffix and negator word occur consecutively. Hence, the negative orientation of the statement is lost. This does not therefore necessitate the negation of the token *"şeker"* (sugar) in this case.

When performing stop word removal, we do not remove some specific words which function as stop words in generic/common domain, but which are also helpful in modifying or identifying opinions. For example, the words *"bayağı"* (quite) and *"çok"* (very) boost the strength of the polarities of the following words. When performing these, the intensifiers are removed, however, the polarity scores of the words they modify are increased. If such intensifiers occur consecutively, as in *"çok çok hoş"* (quite quite nice), the corresponding, modified polarity score is increased even more. Several words have also a heavier impact on the intensification of sentiments as compared to some others. For example, the word *"daha"* (more) has a less impact on the following word compared to the intensifier *"en"* (most). Hence, we take into account different coefficient scores for these modifiers that are to be relied on in the classification phase.

The sentiment scores of the words has a range of $(-\infty, \infty)$. When a negator modifies a word, its polarity is multiplied by -1, causing the polarity to shift. When negators appear repetitively in a row, the sentiment score remains the same. We group the intensifiers into several categories. When employing the intensifier *"more"* and the similar intensifiers (*"very"*, *"quite"*, etc.), we multiply the sentiment score of the modified word by a score of 1.2. Contrarily, if the modifier *"less"* and similar downtoners (*"so so"*, *"a little"*, etc.) explicitly appear in a text, the multiplication coefficient is chosen as 0.8. If intensifiers occur consecutively in the text, the multiplicative factor is defined as 1.2^x for the polarity boosters and 0.8^x for the downtoners, whereby x is the number of intensifiers or dowtoners which come one after another. In the case of the booster *"most"*, the word's polarity score is multiplied by 1.5. On the other hand, for the downtoner *"least"*, the multiplication factor is set as 0.5. We found out the optimal weight parameters by trying them out empirically on two datasets and observing their impact on the performance. These schemes for both Turkish and English are applied separately in this thesis.

In this thesis, we developed sentiment classification approaches which could be evaluated on corpora of different genres. One of these has reviews using standard spelling (i.e. movie corpus) and the other has a different jargon (i.e. Twitter corpus). Words appearing less than a threshold number are removed to combat the noise problem. For the movie dataset, this threshold value is chosen as 20 and, for the Twitter dataset, the corresponding value is set as five. However, for a scheme generated with a supervised technique (e.g. 3-feats), as will be described later, words with a low frequency can have a high polarity score. Therefore, when we did not eliminate any words for this metric, we boosted the performance. For us to cover the Twitter case, we update the normalisation process further. Uniform resource locators (URL) are filtered out, for these do not have an impact on the sentiments of documents (tweets). Nonetheless, we do not remove hashtags from the tweets, because they are reported to have a contribution to the opinions expressed [104]. This was also supported by our preliminary experiments. We give an example in Table 4.1, which shows the effects of preprocessing techniques on a tweet in Turkish. In this example, the roots of the

Table 4.1. A sample tweet in Turkish and the impacts of each preprocessing technique throughout the pipeline.

Technique	**Text**
Raw Text	Cok gusel hareketler degil mi bunlar yaa! :)))) [very nice move+plr not ? these interjection! :))))]
Tokenisation	Cok gusel hareketler degil mi bunlar yaa ! :))))
Deasciification	Çok gusel hareketler değil mi bunlar yaa ! :))))
Further normalisation steps	çok güzel hareketler değil mi bunlar ya ! :))
Morphological parsing and disambiguation	çok çok[Det] güzel güzel[Adj] hareketler hareket[Noun]+lAr[A3pl]+[Pnon]+[Nom] değil değil[Conj] mi mi[Ques]+[Pres]+[A3sg] bunlar bu[Pron]+[Demons]+lAr[A3pl]+[Pnon]+[Nom] ya ya[Conj] ! ![Punc] :)) @smiley[:)][Unknown]
Negation and intensification handling	çok_güzel hareket_ mi bu ya ! :))

words are relied on. When evaluating our approaches on English datasets, we employ a specific tokeniser built for the Twitter jargon specific to this language only [105].

Apart from making use of only unigrams (tokens) as the feature set, we employ patterns and multi-word expressions (MWE) as well that are formed of word n-grams, where $n > 1$. We rely on the Turkish official dictionary (Türk Dil Kurumu (TDK) - Turkish Language Institution) to identify multi-word expressions (e.g. idioms). A multi-word term can contribute to the polarity orientation which may not be determined by the constituent words therein. For example, the multi-word expression *"nalları dikmek"* (to kick the bucket, literally *"to raise the horseshoes up in the air"*) is of

negative polarity, while its constituent words have neutral orientations. On the other hand, patterns are defined as bigrams, in which a predefined set of rules based on the POS tags, as defined in [17], is used and applied for two words occurring consecutively. For instance, the pattern rule "adverb+adjective" describes the scenario, whereby an adverb is followed by an adjective, which is a bigram feature as a whole. As we will explain later, whereas employing MWEs in addition to the unigram features has an impact on the classification success rates, this is not the case for the patterns scheme.

4.1.2. Additional Preprocessing Step: Partial Surface Forms

This part applies only to the sentiment classification task for the Turkish language. The morphologically-rich, agglutinative structure of the Turkish languages enables us to obtain additional features (i.e. morphemes) when conducting NLP tasks. In the literature, researchers mostly take into account either the the surface forms or root forms of the words appearing in the texts [23]. However, in sentiment analysis, some suffixes are more associated with sentiment information compared to the remaining others. For example, the morpheme *"-cağız"* as in *"zavallıcağız"* (poor man/woman) expresses an opinion of pitying someone. As another example, conditional morphemes in Turkish, such as *"-se/-sa"* as in *"Keşke sevse."* (I wish he or she loved.), express wishes or sentiments in an indirect way. Therefore, in this study, we employ these suffixes when generating features in addition to the use of the surface and root forms of the words. For the Turkish corpora we relied on in this thesis, the morpheme POS set consists of 114 different tags defined overall.

In this preprocessing method, a two-staged scenario is modelled. First, we compute the sentiment scores of all the text words that are in their surface form, using the delta tf-idf technique (see Eq. (4.7)). After that, we parse and then disambiguate the words utilising the morphological analysis tools [102, 103] to extract their suffixes. We assume that a morpheme in a word has the same sentiment score as that word, the polarity score of all morphemes is calculated by averaging the scores of the words, to which they are attached. For example, if the suffix *"-se/-sa"* explicitly appears only in

two words in the corpus, whose surface forms are *"çalışsa"* (I wish he or she worked) and *"okusa"* (I wish he or she had an education), the polarity score of the morpheme *"-se/-sa"* is calculated by averaging the delta tf-idf scores of the two mentioned words. Then, we take into account a percentage of the morphemes that has the highest confidence scores. When choosing a specific set of morphemes, we address the following factors:

(i) *Negator morphemes:* Irrespective of their sentiment scores, we do not remove the negator suffixes. They have a significant effect in determining the polarities of words by shifting the sentiments on a word-basis.

(ii) *Number and type of discriminative morphemes:* When we build the discriminative suffix set, we set the number of the top positive and negative morphemes as the same. When we do not follow an equal polarity distribution for top negative and positive morpheme sets, we observe that the results become biased towards the sentiment of the majority class. When conducting experimentations, we evaluate the effects of using various percentages of top discriminative suffixes, which range from 0% to 100% in increments of 10. When this number is 100%, words are considered to be in their surface forms (i.e. all morphemes are processed). When this is set at 0%, their root forms are used (i.e. all suffixes but the negation morphemes are filtered out). We thereby perform a comparative morpheme-based analysis.

(iii) *Root forms:* Only the morphemes which have the least discriminative polarity scores are filtered out in this preprocessing stage. That is, we do not eliminate any word roots no matter what their delta td-idf scores are. We perform a morphological analysis here and the root forms of words are not related to this process.

(iv) *Processing more than one corpus:* In the training phase, the polarity scores of suffixes are calculated using the training set of the corpus. Apart from this setting, we analyse the impact of using other corpora as well on the performance. In this case, when determining the sentiment scores of morphemes, we rely on the training set of one corpus (e.g. movie dataset) and the whole set of another

corpus (e.g. tweet dataset), when inducing a morpheme polarity lexicon for the movie dataset. A morpheme's polarity is calculated by averaging the sentiment scores of that word with respect to the movie and Twitter datasets. The same process can be generalised and applied for three or more datasets as well. As we will empirically show and discuss in the experimentation sections, we have found out that exploiting additional knowledge by relying on other datasets can lead to a good generalisation and this helps combatting overfitting. Root forms of words can have different sentiments across different corpora. However, the sentiment orientations of morphemes are independent of corpora. For example, the suffix *"-cağız"* expresses the same polarity across all the texts written in Turkish, although this does not hold for the word *"unpredictable"* as said.

In the second stage, after the morpheme polarity lexicon is generated, the suffixes which are defined as weakly discriminative morphemes are stripped off the surface forms and the remaining parts only are processed. In other words, the words are defined as their root forms in addition to top discriminative morphemes attached to them. For example, when the suffix *"-se/-sa"* has a high polarity score, but if the score of the possessive suffix *"-m"* is below a predefined threshold, then the word *"yapsam"* (if I did/made) is redefined as *"yapsa"* by filtering our the suffix *"-m"*. Hence, the words *"yapsam"* and *"yapsa"* are updated and changed as the same partial form. After the partial surface forms are obtained, we employed them in the proposed methods as in the same way as the root forms and the surface forms. We will empirically show in the corresponding experimentation sections that employing partial surface form outperforms the use of the root and surface form schemes. We attribute it to the reason that partial forms are additionally more informative about the overall sentiment, where several morphemes are not neutral-like. However, we should state that this metric is applicable only if a labelled dataset exists in the related agglutinative language.

4.2. Major Contribution #1: Unsupervised, Semi-Supervised and Supervised Approaches, and Their Combination

In this section, we first describe the supervised, semi-supervised, and unsupervised approaches we employed. None of them alone have a major contribution. We only tweak their parameters which can be considered a minor contribution to some degree. However, the way we combine these methods in a novel way can be thought of as a major contribution, especially when thereby achieving significant and state-of-the-art results. The architecture of the proposed approach is visually summarised in Figure 4.1.

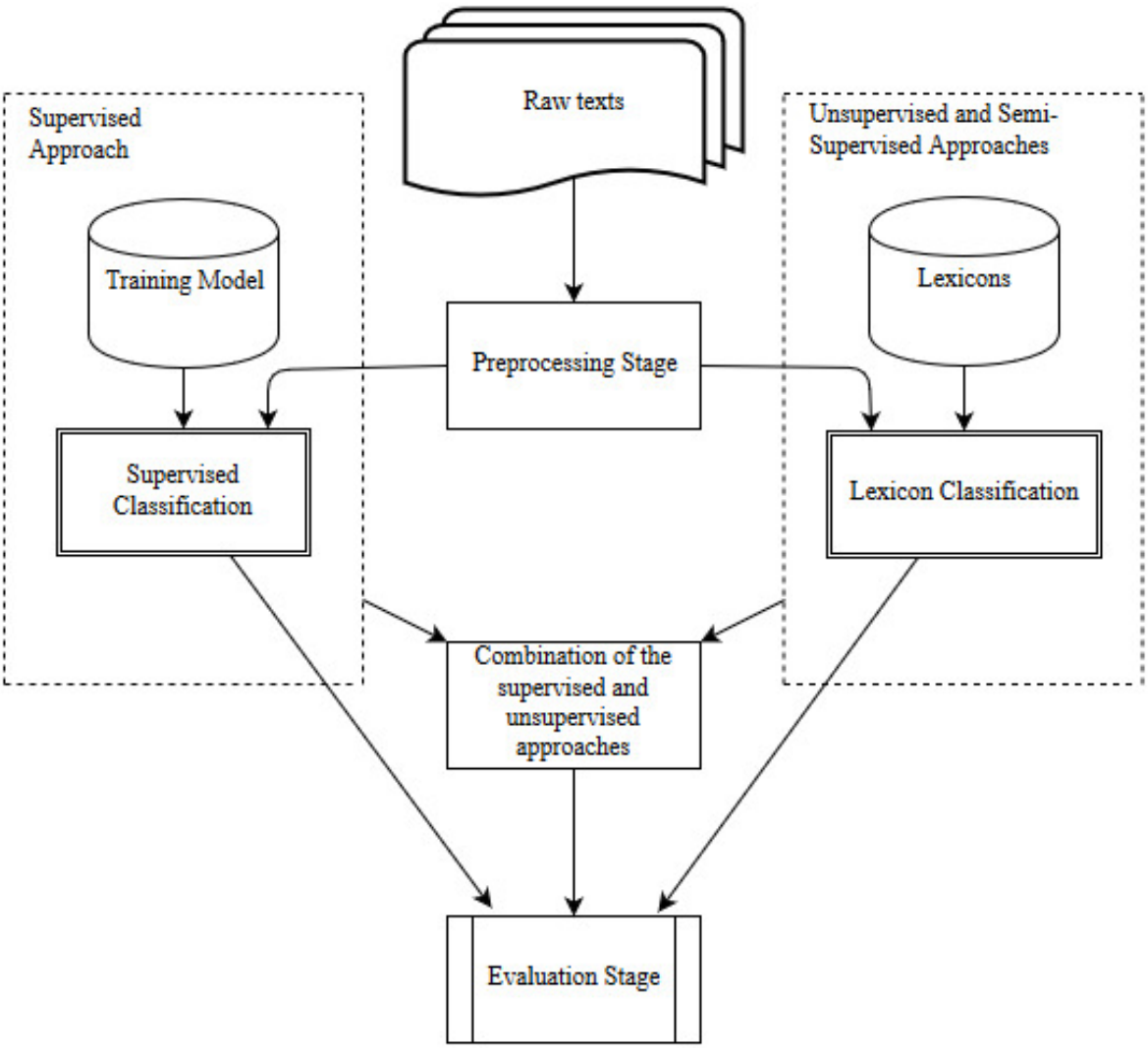

Figure 4.1. The flowchart of the proposed model.

4.2.1. Unsupervised and Semi-supervised Approaches

In the literature, most studies conducted for the sentiment analysis problem in Turkish employ supervised methods [106]. Such works perform many feature extraction and selection methods. Although supervised approaches are diverse and can vary based on their definition, the only unsupervised model used in identifying the sentiments of the Turkish text is the one which refers to the translation of polarity lexicons into their Turkish versions in addition to conducting some further analyses [38, 39]. However, translating sentiment lexicons and lexical databases has three drawbacks which we list below:

(i) It requires an intense effort by human annotators. It is also prone to errors.

(ii) A word that expresses a positive opinion in a language can have the opposite sentiment for another language. For instance, in some parts and countries of Asia, people can make use of the word *"smile"* in the text to cover up embarrassment, shame, or humiliation that is apparently a negative polarity. In Viet Nam, this word is used as a substitute for the expression *"I'm sorry"* or other patterns of behavioural expressions [107]. The word *"smile"* may therefore not be considered to have the same polarity orientation and score in Vietnamese as in English or French, which constitutes as a subjective topic.

(iii) Besides the cultural aspect, even people who speak the very same language may translate words into their forms in the target language differently. Thus, a word can be labelled different sentiments with respect to different translators.

Because of these drawbacks, in lieu of translating an available sentiment lexicon into its Turkish form, we developed approaches that calculate the polarity scores of the words in an automatic manner without resorting to annotating a bulky set of words manually by humans. We rely on two algorithms, which are domain-independent and domain-specific methods, for this goal. The first model computes a generic polarity score for each word without exploiting any domain-specific knowledge. We construe this approach as an unsupervised technique, because it leverages only a manually-generated

small set of seed words with generic polarities. We also develop a domain-specific approach by observing the impact of different domains on the sentiments of corpus words. This is a semi-supervised method, since we manually define the seed words by employing the domain and polarity information. When conducting experimentations, we have observed that the domain-specific methods have performed better than the domain-independent method as we expected.

4.2.1.1. Domain-Independent. In the domain-independent model, we build a polarity lexicon by calculating a sentiment score (SC) for each word. In this respect, we use the pointwise mutual information (PMI) metric [17]. Given a word *word*, the polarity score is computed in accordance with the formula given below:

$$SC(word) = \log_2\left(\frac{hits(word\ NEAR\ `harika')}{hits(`harika')} \times \frac{hits(`berbat')}{hits(word\ NEAR\ `berbat')}\right) \quad (4.1)$$

The words *"harika"* (great) and *"berbat"* (awful) are antonyms determined manually. $NEAR$ is an operator that corresponds to the co-occurrence of the words w_1 and w_2 in the pattern *"w_1 NEAR w_2"*. That is, this models the statistical scenario that these two words co-occur in sliding context windows. We employ three different operators and patterns in this model, as will be described and discussed in the following paragraphs. *hits(query)* denotes the number of hits returned by a search engine with regard to the given query.

Eq. (4.1) is defined by performing division of the PMI formula for the pair *word* and *"harika"* (great) by the PMI formula for the pair *word* and *"berbat"* (awful), and then filtering out the effect of the *hits(word)* term in the denominator in the first part and the *hits(word)* term in the numerator in the second part. In this way, the polarity scores of given words are identified by computing the ratio of the statistics of its

co-occurrence with a positive word (e.g. *"harika"* (great)) and its co-occurrence frequency with the predefined negative word (e.g. *"berbat"* (awful)), and then performing normalisation with regard to the total frequencies of the words of this antonym pair. We also conduct smoothing operation by adding a small number (0.001) to the *hits* function to handle the problem of zero-occurrences. The higher the score of Eq. (4.1) is, the higher positive sentiment orientation a word has (the more probably it expresses a positive sentiment). If the opposite holds, this is most likely of negative polarity.

Unlike in other relevant studies in Turkish, we compute the hit frequency values through a search engine (Yandex) [108], in lieu of relying on manually-curated datasets. We assess and analyse the performance of the search mechanism by employing one operator and additionally two collocational patterns. The operator *"w_1 NEAR(k) w_2"*, as stated above, returns the number of co-occurrences of the given tokens in the same window of the length k. The collocational patterns are *"w_1 ve w_2" ("w_1 and w_2")*, such as *"zeki ve güzel"* ("intelligent and beautiful") and *"hem w_1 hem w_2"* ("both w_1 and w_2"), such as "hem cin fikirli hem çalışkan" ("both astute and hard-working"). We take into account these types of conjunctions as operators, for they are used in connecting words, phrases, or clauses with similar sentiment orientations or semantic meanings. For instance, the phrase *"hard-working and beautiful"* is more sensible than *"lazy and beautiful"*. That is, the conjunction *"and"* in general links two words of the same or similar sentiments.

We have found out that the operator $NEAR$ produces more hits and leads to more accurate and robust polarity values than the patterns of collocations. We used several values for the context window length and observed that the optimal window length is $k = 12$ for the operator $NEAR$. In other words, when we look up the six words on both the left and the right of a target word, we achieve the best results. The optimal value was detected by assessing the final performance of the polarity classification model taking account of word polarity scores produced with respect to these various context window lengths. When we increased the value k more, words and phrases of opposite polarities may tend to occur more frequently and the context

is assumed to be too broad. By contrast, when we set the context window size as a smaller value, we miss the co-occurrences of the informative collocations of words, which makes the context too narrow and insufficient.

We have built a seed word set consisting of ten antonym pairs that are non-stop words and that are among the most frequently appearing tokens, as we show in the upper part of Table 4.2. The reason that ten antonym pairs as seed words are chosen in this thesis is that the same applies in most studies in the literature as well [40]. The size preferred could be considered to be small and insufficient. Nonetheless, these antonym pairs are powerful indicators of widely-used and highly representative sentiments. These can capture the implicit and explicit sentiments of the text words effectively and accurately.

We calculate the sentiment values of given words by Eq. (4.1) for each antonym words and lastly average these. The reason that we rely on ten antonym pairs and their average, in lieu of a single antonym pair, is to combat the bias that may occur towards a single pair. In other words, we do not want to base our metric on a very specific context. The sentiment score of a document/review is produced by averaging all the polarity scores of words occurring in that text. When the averaged sentiment value exceeds zero, the predicted sentiment is positive. If the opposite holds, the sentiment is predicted as a negative. First, only the words with the part-of-speech tags, which are noun, adjective, verb, and adverb, are taken account of in the reviews. In the preliminary experimentations, we observe that other tokens, such as conjunctions, are not generally indicative of the overall polarity of the document for some approaches. However, some tokens, such as *"10/10"* with the POS tag *"unknown"*, can also contribute to the performance for several modules and datasets (e.g. the 3-feats technique as will be discussed later). This unsupervised method is summarised in Figure 4.2.

4.2.1.2. Domain-Specific. The approach described in the above section predicts the polarity score of tokens without exploiting the domain information of a review. Nevertheless, the sentiment orientation and score of a token can differ based on corpora

Table 4.2. Different seed words of opposite polarities chosen manually for the domain-independent and the domain-specific methods.

Technique	**Positive**	**Negative**
Domain independent	sevgi (love)	nefret (hate)
	harika (great)	berbat (awful)
	tatlı (sweet)	acı (painful)
	olumlu (positive)	olumsuz (negative)
	:) (happy emoticon)	:((sad emoticon)
	güzel (beautiful)	çirkin (ugly)
	doğru (correct)	yanlış (wrong)
	zevkli (enjoyable)	sıkıcı (boring)
	iyi (good)	kötü (bad)
	sevimli (lovely)	sevimsiz (unlovely)
Domain specific	sürükleyici (gripping)	yorucu (wearisome)
	şaşırtıcı (unpredictable)	öngörülebilir (predictable)
	sıradışı (unusual)	kalıplaşmış (cliché)
	başyapıt (masterpiece)	vasat (mediocre)
	büyüleyici (fascinating)	iğrenç (awful)
	güzel (beautiful)	çirkin (ugly)
	doğru (correct)	yanlış (wrong)
	zevkli (enjoyable)	sıkıcı (boring)
	iyi (good)	kötü (bad)
	sevimli (lovely)	sevimsiz (unlovely)

with different genres. For example, as said, the word *"unpredictable"* has a positive polarity for the movie domain (*"unpredictable plot"*), whereas it has a negative polarity for the reviews made in the car domain (*"unpredictable steering"*). In order to model this domain-specific scenario based on these factors, we adopt the methods described in a study [1] for the Turkish language by implementing minor changes in our version. We evaluated this approach on the Twitter and movie datasets. However, this may be

for each word **w** in training dataset **do** //induce a polarity lexicon

compute SC(**w**) by Eq. (4.1)

for each document doc in test dataset **do** //evaluate on test documents

doc-sent ← sum(SC(**w**)), $\forall\, \boldsymbol{w} \in \boldsymbol{doc} \wedge POS(\boldsymbol{w}) \in \{N, V, Adj, Adv\}$

if doc-sent > 0 **then** predict as *'positive'* **else** predict as *'negative'*

Figure 4.2. Algorithm for unsupervised approach.

adapted to other domains as well after conducting modifications. In this section, we describe and explain the approach for the scenario modelled for the film corpus.

As for the domain-independent method, we define a seed word set consisting of positive and negative antonym pairs. In this case, words are sentimentally specific to the movie corpus. We list these antonym sentiments in the lower part of Table 4.2. As shown, there is an overlap between the seed words defined for the domain-independent and domain-specific scenarios. This is indicative of the fact that some words (e.g. *"doğru"* (correct) and *"yanlış"* (wrong/incorrect)) can be considered to be too specific to and generic/broad for a domain at the same time.

Using those manually chosen seed words, a propagation method is utilised to generate a polarity lexicon for a specific domain/dataset. The underlying mechanism of this approach is that we can hypothesise two words are similar to each other, if they co-occur in the same context windows directly or indirectly. Accordingly, a graph is built such that vertices represent corpus words. On the other hand, edges between nodes represent how frequently the linked words co-occur in sliding windows. If the link's weight is heavier, we assume that those two words are closer to each other in the vector space model (VSM).

So as to compute the weights of these edges, a matrix $\mathbf{M}$ whose entries correspond to the positive PMI (PPMI) values is built. These PPMI values (i.e. $\mathbf{M}_{i,j}$) are

computed as in the following equation, where w_i and w_j are the connected words in the graph.

$$\mathbf{M}_{i,j} = max(\log(\frac{p(w_i, w_j)}{p(w_i) \times p(w_j)}), 0) \tag{4.2}$$

In this equation, $p(w)$ refers to the frequency of a word in the sliding window contexts across the whole corpus, whereas $p(w_i, w_j)$ is the probability that those two words co-occur in the sliding windows of a fixed length. We have taken into consideration several hyper-parameters which are the context window sizes ranging from 10 through 30 in increments of 5. We achieve the highest performance when choosing this value as 15. The max equation in the formula is employed, since co-occurrence statistics are not reasonable when the value is below 0. When building this matrix, a vector per row word w is created. We then compute the edge/link weights of this graph. We calculate the edge weight $\mathbf{E}_{i,j}$ between two words by using the cosine similarity metric between these two embeddings.

While generating this graph, a random walk approach to propagating the effects of sentiments across the corpus words is followed. In this method, if some tokens co-occur with a predetermined seed polarity word frequently and are close to this seed token, these tokens are assigned the same or similar polarities scores. This is visually summarised in Figure 4.3. The equations are formed of the following notations. $\mathbf{v}$ is the vocabulary set of corpus words. The below algorithms identify the positive and negative sentiments independently. $\mathbf{P}_N$ and $\mathbf{P}_P$ correspond to negative and positive polarity scores, respectively. Before starting to iterate, the ${\mathbf{P}_P}^{(0)}$ and ${\mathbf{P}_N}^{(0)}$ embeddings are initialised. In this respect, every entry of these two vectors is assigned the score of $\frac{1}{|\mathbf{v}|}$. We then update the sentiment score embeddings at each iteration as shown in the below equation. (In this formulation, we only explain the case for the positive class. The analogous process applies for the negative vectors as well.)

$$\mathbf{P}_P{}^{(k+1)} = (1-g)\mathbf{E}\mathbf{P}_P{}^{(k)} + g\mathbf{s} \tag{4.3}$$

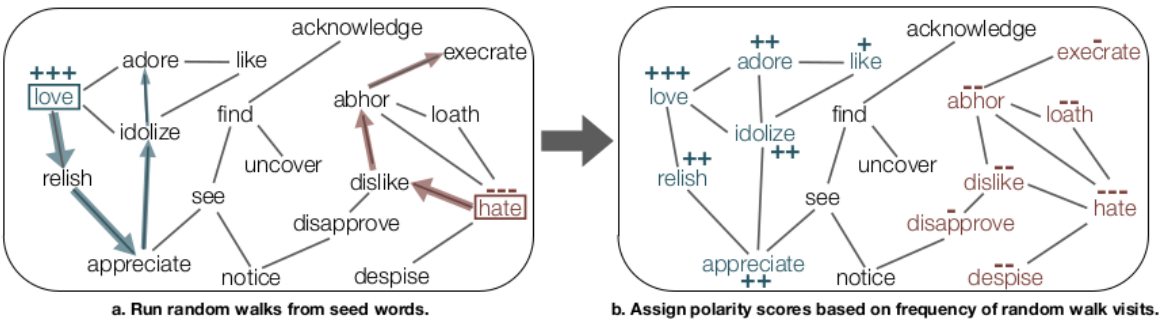

Figure 4.3. The visual summary of the semi-supervised approach [1]. We tweaked the parameters of this approach and improved the success rates when evaluating the model on Turkish and English datasets.

In this formula, k refers to the number of the iteration. $\mathbf{s}$ denotes the vector utilised for seed words, whereby the predetermined seed word entries have the weight of $\frac{1}{|\mathbf{s}|}$, whereas the other entries have a score of 0. As such, the formula is formed of two constituents. The $\mathbf{E}$ matrix holds the co-occurrence statistics information. The vector $\mathbf{s}$ is utilised that favours positive seed tokens. The constant $\boldsymbol{g}$ is determine the effect of local contextual information. If this has a low score, local information is favoured. Otherwise, global information is given more importance such that seed polarity words have a larger impact on the final sentiments of all words.

While performing the propagation process across the graph, the sentiment values of tokens keep converging to a value at each iteration. This shows that tokens that co-occur with seed polarity words directly or indirectly too frequently have the same or similar polarities. On the other hand, the scores of those words far away from the seed words in the graph are affected and updated to a lesser degree, which probably means that these are neutral-like tokens. Therefore, the absolute polarity values of such words are lower. To combat this bias, we update and increase the local contextual consistency value, where we steadily decrease the $\boldsymbol{g}$ value, differently from the baseline approach [1]. The main intuition behind this scenario is that the set of seed polarity

words we determined in advance may not be the optimal ones and can be misleading for some corpus words. We perform the iteration until we meet the convergence or meet a maximum number of iterations (i.e. N).

Later, we generate the positive sentiment score vector (i.e. $\mathbf{P}_P$) as shown below:

$$\mathbf{P}_P = \sum_{k=1}^{N} \frac{\mathbf{P}_P{}^{(k)}}{k!} \tag{4.4}$$

Here, we rely on the vectors built throughout the iterations, and then generate a series and lastly sum these vectors column-wise. In this sequence, at each step k, we perform division of these vectors by the factorial value k. The reason why we chose the factorial function in lieu of the exponential decay or some other functions is that we wanted to give priority to those words similar to the seed polarity tokens in the VSM. We observed a slight increase of 1% in the performance by employing Eq. (4.4) as compared to the baseline approach [1]. We have thereby shown the effects of tweaking the parameters in the formula. As said, those tokens far away from seed polarity words are assigned more neutral-like sentiments in the end.

Lastly, we generate the polarity score vector which is notated by $\mathbf{P}$. The entries of this sentiment vector, each of which is referred to as w, are computed by the formulation in Eq. (4.5). When the score of $\mathbf{P}(w)$ exceeds 0, the word is considered a positive token. If the opposite holds, it is considered to be of negative polarity. Different coefficients are employed in Eq. (4.5) than the baseline study [1]. As mentioned, we have therefore observed a small increase in the performance.

$$\mathbf{P}(w) = \log(\frac{\mathbf{P}_P(w)}{\mathbf{P}_N(w)}) \tag{4.5}$$

After we identify the polarity scores of each corpus word, we sum all the polarity values in a document. If the summed score of a document is above 0, the overall sentiment is predicted as positive; otherwise, it is predicted as negative. For instance, in Table 4.1, polarity scores are found to be *"çok_güzel" [+3] and ":))"* [+2]. The overall summed score is therefore +5. That is, this review is predicted to be positive. We summarise this semi-supervised method in Figure 4.4.

for $\forall w_i, w_j \in$ *training dataset* **do**
  calculate $\mathbf{M}_{i,j}$ using Eq. (4.2) //compute PPMI value
for $\forall w_{i,j} \in$ *training dataset* **do** //build edge matrix
  $\mathbf{E}_{i,j} \leftarrow \cos(\mathbf{M}_{[i,:]}, \mathbf{M}_{[j,:]})$
initialise $\mathbf{P}_P^{(0)}$ and $\mathbf{P}_N^{(0)}$ //initialise polarity vectors of words
loop until convergence //random walk algorithm
  calculate $\mathbf{P}_P^{(k+1)}$ and $\mathbf{P}_N^{(k+1)}$ using Eq. (4.3)
calculate $\mathbf{P}_P$ and $\mathbf{P}_N$ using Eq. (4.4)
calculate **P** using Eq. (4.5) //build sentiment lexicon
for each document **doc** in test dataset **do** //process test documents
  doc-sent $\leftarrow$ sum(**P**(**w**)), $\forall \boldsymbol{w} \in \boldsymbol{doc} \wedge POS(\boldsymbol{w}) \in \{N, V, Adj, Adv\}$
  if doc-sent > 0 **then** predict as *'positive'* **else** predict as *'negative'*

Figure 4.4. Algorithm for semi-supervised approach.

4.2.2. Supervised Approaches

We categorise the supervised approaches we utilised in this thesis under two scenarios. In the first case, we make use of conventional machine learning models, such as SVM. We have initially created several feature engineering techniques and fed these into classifiers. In the latter case, we rely on deep learning models. In this way, we do not need to perform comprehensive preprocessing, feature selection and extraction operations, since these are mostly handled automatically by these architectures.

4.2.2.1. Conventional Machine Learning Models and Feature Sets. For the classical machine learning architectures, the choice of the "right" feature sets rather than classifier models can lead to more successful results. Several feature weighting metrics are accordingly relied on for this approach. The first two schemes are the delta idf and delta tf-idf techniques that are given in Eqs. (4.6) and (4.7), respectively [24]:

$$\text{delta idf}_{\text{w}} = \log \frac{\frac{N_{P,w}}{N_P} + 0.001}{\frac{N_{N,w}}{N_N} + 0.001} \tag{4.6}$$

$$\text{delta tf-idf}_{\text{w, d}} = (0.5 + 0.5 \times \frac{f_{w,d}}{max_{\{w' \in d\}} f_{w',d}}) \times \text{delta idf}_{\text{w}} \tag{4.7}$$

Since these two exploit the sentiment information present in the training dataset, they generally lead to state-of-the-art performances. In [24], it is stated that this metric performs much better compared to the tf-idf scheme, which does not use the sentiment information. In Eq. (4.6), delta idf_{w} denotes the polarity value of the word w in the dataset. N_P and N_N are the number of positive reviews and negative reviews, respectively. $N_{P,w}$ and $N_{N,w}$ are the counts of documents in the positive and negative dataset, where the word w explicitly occurs, respectively. Numerator and denominator are both normalised to handle the class imbalance problem. For smoothing, we add a value of 0.001 to both numerator and denominator. The delta idf_{w} metric is utilised to denote the weight of word w, ignoring the frequency statistics on a single review-basis.

Eq. (4.7) is based on a variation of the tf-idf scheme. $f_{w,d}$ denotes the frequency of the word w in the document d. We normalise these values such that we divide these frequencies by the most frequently appearing token in the d, which is formulated as by $max_{\{w' \in d\}} f_{w',d}$. We thereby prevent a bias towards lengthier documents. delta $\text{tf-idf}_{\text{w, d}}$ takes into consideration both the raw frequency of a word in a document and the supervised score.

The third weighting metric utilised is the classical tf-idf scheme. In this scenario, we do not exploit the sentiment information unlike the above-mentioned ones. We

calculate tf-idf scores for the words occurring in both positive and negative datasets. The other metric models the scenario, in which minimum, mean, and maximum word polarity scores calculated by Eq. (4.7) are additionally concatenated on a review-basis, in contrast to those other scores that are built on a word-basis. As will be shown in the results, these three scores in general lead to the best success rates. We also combine the three polarities scheme with the tf-idf metric by performing column-wise concatenation. In other words, unsupervised and supervised techniques are combined in this last scheme.

After building all those features, we used classifier models to predict the polarity labels of documents. The conventional machine learning models employed for this approach are SVM, decision tree (J-48), NB, and k-nearest neighbour (kNN) algorithms. In the literature, generative models are generally stated to be outperformed by discriminative models [109]. Our experimental results also demonstrate that this holds true, as will be discussed in the results section.

Apart from the machine learning models, we utilise a simple and basic supervised method as well that we name *"log scoring"* (LS). In this method, we only sum the delta tf-idf scores of all the document words. If the overall summed score exceeds the threshold 0, the sentiment is predicted as positive; otherwise, as negative. We have observed that such a basic technique achieves almost as high success rates as the SVM classifier produces. The LS method even performs better than the kNN model.

4.2.2.2. Neural Network Approaches. In addition to conventional machine learning models, we have also utilised two deep learning architectures, which are the CNN and LSTM models. In these approaches, only tokenisation is performed as a preprocessing technique. We do not generate explicit feature sets either for these two networks. As mentioned, these deep learning models can generate features automatically on their own.

LSTM is an RNN model consisting of LSTM units [110]. Every cell in this network "remembers" values over arbitrary time intervals and accounts for memory in the architecture. The three gates in these cells are in fact conventional artificial neurons. An activation function is applied taking the weighted sum of the outputs in these gates as input. These architectures are superior over classical RNNs in the sense that these can remember the previous states more effectively and robustly in a long-term dependency manner.

In the LSTM model we relied on, we conduct sentiment classification task on a document-basis. At the most-bottom layer, we feed word embeddings into the architecture. Word embeddings have been trained with the skip-gram setting on a large corpus that consists of 951M words [111]. A shallow neural model is used to generate word vectors as in [8]. This algorithm employs a log-linear classifier for exploiting the statistical information. Contextual windows are accordingly defined and relied on to create dense embeddings. Those words that do not co-occur in the same context windows directly or indirectly are modelled to have less similar representations. The size of those vectors is chosen as 300. The number of LSTM units is selected as 196. So as to prevent overfitting, we determined the dropout rate to be 0.4. The dropout process is performed only in the training phase, not for the development and testing stages. A softmax classifier is employed over the top layer of the architecture. Here, the categorical cross-entropy problem is handled. The maximal number of epochs is set at 200. However, if the model converges or the performance for the validation set starts to decrease after a specific epoch, we choose the optimal model parameters according to the development phase.

Convolutional neural networks have recently gained popularity mostly in the areas of the image and video recognition tasks, and also in NLP tasks [58]. As a conventional feature selection method, the use of n-grams with a fixed length may lead to a loss in the information. For example, in two trigrams, if the first two words are the same in both these trigrams, but the last word is different, these two are considered to be completely different, although this should not be the case. Nonetheless, the CNN

architectures can handle this issue to some degree. These models capture the most informative local information and use this in the classification stage.

We relied on a publicly available CNN code repository [112]. We slightly tweaked the parameters of this repository to adapt it to the Turkish languages and to feed off-the-shelf word embeddings into the system. We use the character encoding as UTF-8 and filtered out several tokenisation rules that are specific to English only. We have thereby incorporated several tokenisation and other preprocessing techniques for the Turkish language. So as to feed static word vectors into the model, we have updated the repository with a few lines of code.

The scores obtained in the initial layers are convolved into scalar values by employing several filters with different lengths. First, word embedding values are reduced to single numbers, when these are scanned through sliding windows. We perform max-pooling such that only the most important and maximum values are taken into account. Other parameters, such as dropout regularisation and L2 values among others are employed to combat the overfitting issue and are fine-tuned by the use of a development set. Finally, we perform opinion classification over the last layer, on a fully connected softmax classifier module.

Non-static word vectors that are learnt from scratch during the training phase and static (pre-trained) word vectors [113] are both employed and tested as input independently. We utilise those word vectors as in the LSTM classifier. We preferred the skip-gram model over the continuous bag-of-words metric for the word2vec approach, since this generates more robust embeddings when a large corpus is available. The length of these pre-trained embeddings is 300. Dense embedding models can combat the data sparsity problem and are more informative as compared to the PMI approach that is employed in the unsupervised and semi-supervised metrics. In other words, exploiting the co-occurrence information about two words cannot be as effective as in employing a neural network model, when fed with a large corpus as input. We summarise all the supervised approaches in Figure 4.5.

Classical Machine Learning Models

for each word **w** in training dataset **do** //build feature set

calculate delta idf_w (Eq. (4.6)), delta tf-$idf_{w,d}$ (Eq. (4.7)), tf-idf_w, 3-feats

run classifier //use machine learning approach (SVM, J-48, kNN, NB) or LS method

Neural Network Models

gather word vectors that are pre-trained or learn them from scratch

run classifier using these static and non-static embeddings //use deep learning models //(i.e. LSTM, CNN)

Figure 4.5. Algorithm for supervised approaches.

4.2.3. Combining Unsupervised and Supervised Feature Metrics

Another approach we rely on is the combination of the unsupervised, semi-supervised, and supervised methods, which is a novel and effective feature engineering method developed by us. In this method, if the unsupervised and supervised sentiment values of a token are of different polarity signs (i.e. one positive, the other negative), we consider that word to be sentimentally ambiguous and take into consideration only a fraction of the supervised score. That is, we multiply the supervised score by a specific coefficient. Otherwise, we average these unsupervised and supervised scores, as shown below:

$$combSC_w = \begin{cases} c_s \times supervised\_score(w), & \text{if } opposite \\ & signs \\ c_u \times unsupervised\_score(w) + c_s \times supervised\_score(w), & \text{otherwise} \end{cases} \tag{4.8}$$

In this formulation, the unsupervised score and the supervised score correspond to the scores calculated in, respectively, Eq. (4.1) and Eq. (4.7). For the supervised case, we prefer the delta tf-idf metric (Eq. (4.7)) over other weighting schemes, since it leads to better performances. c_u and c_s are the coefficients of the unsupervised and

supervised scores, respectively, where $c_u + c_s = 1$. So as to find the most optimal coefficient values, we first split the whole corpus into training, validation, and test sets. After that, grid search is performed on the development set using nested 10-fold cross-validation, where coefficients can range from 0.1 through 0.9 in increments of 0.1. Another scenario would have been that a supervised regression approach could be followed. Nonetheless, our simple approach was effective, robust, and reliable as well, so we employed our technique. We found out that the optimal values are $c_u = 0.3$ and $c_s = 0.7$ for both datasets. This indicates that, as can be expected, the supervised score is more reliable and informative about a word's polarity, thus it contributes more to the overall score. Nonetheless, the unsupervised weight also has an effect and ignoring it causes the performance to decrease. When the signs of the unsupervised and supervised polarity scores are opposite, we employ only the supervised polarity score multiplied by c_s and the unsupervised component is ignored. The intuition behind is that the supervised score is more reliable; however, due to ambiguity, we lessen its impact. After the feature scores are computed in a combined manner, we feed them into classical machine learning algorithms. As will be discussed later, employing this basic combined approach improves the performance in some cases, especially when we do not perform intensification and negation handling. We give the summary of this approach in Figure 4.6.

4.2.4. Experimental Evaluation

We evaluated the proposed approach on text data in two domains for the binary sentiment classification problem in Turkish at document-level. In order to test whether our approaches are portable to other languages, we evaluated the methods on three English corpora as well. In this section, we first give the details of the corpora and the possible values used for the hyper-parameters. Then, we give a detailed explanation on the experiments and make comments on the results obtained.

4.2.4.1. Datasets. In this approach, we first used two datasets of different genres. The first one is a movie dataset consisting of movie reviews in Turkish collected from

```
for c_s = 0.0; c_s <= 1.0; c_s += 0.1 do                    //grid search
   c_u ← (1 - c_s)
   for each word w in training corpus do
      compute unsup_score(w) using Eq. (4.1)
      compute sup_score(w) using Eq. (4.7)
      compute combSC(w) using Eq. (4.8)
   run classifier on development corpus
   measure success rate                                     //find optimal c_s and c_u values
run classifier using optimal combSC values
```

Figure 4.6. Algorithm for the combination of unsupervised and supervised approaches.

a popular website [114]. This is the same corpora in the work that we consider as a baseline for comparison [38]. The size of the dataset is 20,244 reviews, where the average number of words in reviews is 39. The reviews have star scores ranging from 0.5 to 5 in increments of 0.5 points. If the overall sentiment score of a review is equal to or lower than 2.5, then we considered it to be negative, whereas if the score is equal to or higher than 4 it is considered as positive. In overall, there are 13,224 positive reviews and 7,020 negative reviews in this movie dataset.

The second corpus is a Twitter dataset formed of tweets in Turkish. The tweets in this corpus are much shorter and noisier compared to the reviews in the movie dataset described above. The training set is composed of 3,000 tweets and was taken from the website of the Kemik NLP group of Yıldız Technical University. Tweets in this set are about two pioneering Turkish mobile network operators, which are Turkcell and Avea. The tweets in the development and test sets which cover various topics were collected and curated by two undergraduate students and were annotated as positive, negative, or neutral. We filtered out the neutral reviews from the dataset since we perform binary sentiment classification only for this approach. We merged and combined all the sets, shuffled them, and applied 10-fold cross-validation on them. We thereby take into account several topics and domains in the same corpus and test how it impacts

the performance. In total, there are 1,716 tweets, of which 743 are positive and 973 are negative. The annotators measured the Cohen's Kappa inter-annotator agreement score to be 0.82.

We also utilised three other datasets in English to test the portability of our approaches across languages. One of them is again a movie corpus collected from the web [115]. In this dataset, there are 5,331 positive reviews and 5,331 negative reviews in this dataset. The other is a Twitter dataset containing nearly 1.6 million tweets annotated by using a distant supervised method [116]. In the method leveraged, if a tweet includes a positive emoticon, it is labelled as positive and, if it includes a negative emoticon, it is labelled as negative. Otherwise, it is labelled as neutral. That is, these tweets have positive, neutral, and negative labels. We have chosen 7,020 positive tweets and 7,020 negative tweets randomly. As a third dataset, we used the dataset collected for the widely-known SemEval 2017 competition [117] to compare our performance against the contestants. Since we perform binary sentiment analysis, we chose the Subtask B dataset for "tweet classification according to a two-point scale". This corpus is formed of tweets including topic information along with sentiment. The training dataset consists of 20,508 tweets, whereas the test dataset is composed of 6,185 tweets. We removed neutral and "off topic" reviews from the training set and processed the remaining 18,962 tweets.

Since there are separate training and test datasets for the SemEval task, we have not performed cross-validation for this corpus. We split 20% of the training dataset as validation set to find the optimal values for the combination of supervised and unsupervised approaches. In the other corpora, in the approach combining the supervised and unsupervised methods described previously, we used nested ten-fold cross-validation. In all the other experiments, we employed non-nested ten-fold cross-validation. We partition the data into three parts: as 80% for training set, 10% for development set, and 10% for test set. The validation portion is used to determine whether the convergence criterion is met, to prevent overfitting, and to find the optimal hyper-parameter values.

4.2.4.2. Hyper-parameters. We used the scikit-learn framework [118] for the classical machine learning algorithms (J-48, SVM, kNN, and NB). In SVM, we relied on linear kernel since sentiment analysis is generally considered to be a linearly classifiable problem. For the kNN method, we chose the nearest neighbour number k as three and used the cosine similarity metric. We chose k as an odd number to prevent ties while classifying a review. We label a document as positive or negative, depending on the majority of the sentiments of its nearest neighbours.

In the LSTM framework, we applied only tokenisation and lemmatisation in the preprocessing stage, since the model carries out most of the feature selection and extraction tasks inherently and automatically. In the case of CNN, we chose the number of filters as 128 and set the filter sizes as 3, 4, and 5. In this network, there are four layers, which are the word vector layer, convolutional layer, max-pooling layer, and softmax layer. We set the dropout rate at 0.5 and chose the L2 regularisation value as 0.005. We did not perform negation handling and multi-word expression extractions, since this model is assumed to achieve this effect via the sliding windows. We have trained the network until the convergence criterion is met or for at most 200 epochs, as in the LSTM case. If the loss on the validation set stopped decreasing, we consider the early stopping criterion is met and we cut the training phase at that point.

4.2.4.3. Results. We first show the performances (F1-score) of the five feature weighting schemes in Table 4.3. Each metric was applied to both datasets using the traditional machine learning methods. We should state that in this experiment, we do not measure the success of a metric on a word-basis. That is, we are not interested in how accurately a metric (say delta idf_w) determines the polarity of a word. Instead, we measure the performance on a review-basis by employing the polarity scores of the words in the review determined by the metric, and then predicting the review as positive or negative.

In the experiments, we tested the methods using the surface forms of the words, the root forms of the words, and the forms obtained by concatenating the roots with

Table 4.3. Contribution of each weighting scheme to the performances (%) of the supervised approaches for the two datasets.

Weighting scheme & **Dataset**		**J-48**	**SVM**	**kNN**	**NB**	**LS**
delta idf	Movie	89.71	89.89	73.32	89.18	88.32
	Twitter	78.75	78.56	65.12	77.00	75.84
delta tf-idf	Movie	89.92	90.01	74.25	89.53	88.64
	Twitter	78.65	79.29	66.26	78.40	76.05
tf-idf	Movie	88.82	89.45	72.89	88.87	-
	Twitter	76.38	77.56	64.28	77.01	-
3-feats	Movie	89.87	**90.98**	74.73	89.53	-
	Twitter	78.74	**79.54**	66.43	78.48	-
tf-idf + 3-feats	Movie	89.80	90.01	74.00	89.43	-
	Twitter	78.01	78.12	64.54	77.07	-

the morphemes having the highest confidence scores (partial surface forms). We found out that using the partial surface forms of words as features produces the best results. We have also observed that, unlike the unsupervised and semi-supervised approaches, using all the tokens gives better success rates than only using words with several specific POS tags (e.g. noun, adjective, verb, adverb). The main reason behind it is that some tokens that are not morphologically correct words (e.g. the token *"10/10"* in a movie review) may carry sentimental information. The figures in Table 4.3 correspond to partial surface form and all tokens cases.

We find out that the best success rates were achieved using the 3-feats model for both datasets, which is a simple model that relies on only three features. This is an interesting finding, which can be attributed to the specific nature of the sentiment classification problem. In this problem, the average sentimental meaning of a document accompanied with the maximum and minimum word values may successfully signal the variance of sentiment of the review. That is, it takes into account the most extreme values occurring in a review. If the maximum, minimum, and mean polarity scores of a review differ from each other to a great extent, we would hypothesise that it has

"borderline" traits. This is akin to how extreme values are captured in the pooling operation of CNNs architecture. For instance, the first reviewer may make a comment like: *"Graphics were hilarious; the plot was OK. However, the actress was beyond terrible."* On the other hand, the second commentator's review could be *"All the graphics, plot, and actors were OK."* Although these two reviews might have the same mean polarity score, the variance occurring in the first review is much greater. That is, it can be considered an outlier review, which affects the performance in a different manner.

Table 4.4 shows the results of the unsupervised (Section 4.2.1.1) and semi-supervised (Section 4.2.1.2) approaches, compares them with the supervised case, and also shows the effect of various combination strategies on the success rates. For the supervised classifiers alone, we include the results of the 3-feats method obtained with SVM, which prove to be the most successful ones. All the experimentations were conducted with both the combination of unigrams with MWEs and patterns. We observe that the use of patterns results in worse performance. The success rates for the unsupervised and semi-supervised methods are much lower than those of the supervised learning approaches as can be expected. We tested several combinations of the four supervised methods using majority voting scheme. We achieve the best results with the ensemble combination of J-48, SVM, and NB. The results obtained under this ensemble of classifiers outperformed SVM by a small margin. This result indicates that in some cases the instances misclassified by a classifier are compensated by the other classifiers, and a more robust framework is built. The final result we obtained is the combination of the unsupervised and supervised methods. This combination also using the majority voting scheme yields the highest performances. This signals the necessity and contribution of the unsupervised metric in sentiment classification. The result obtained for the movie corpus (91.17%) outperforms the baseline study in Turkish [38] by a significant margin, in which an accuracy of 89.5% is reported to be obtained on the same dataset. Our success rate is also the highest for the binary sentiment classification task in Turkish on the movie dataset, to the best of our knowledge.

Table 4.4. Summary of performances (%) of the unsupervised, semi-supervised and supervised methods (using the 3-feats technique), and their combinations for the two datasets.

Method	**Unigram + MWE**		**Pattern**	
	Movie	**Twitter**	**Movie**	**Twitter**
Unsupervised	70.12	67.82	61.42	59.88
Semi-supervised	72.28	68.59	65.86	61.97
J-48	89.87	78.74	84.21	73.28
SVM	90.98	79.54	84.67	73.44
kNN	74.73	66.43	70.16	63.12
NB	89.53	78.48	84.12	73.76
Supervised Majority Voting (J-48, SVM, NB)	91.03	80.48	85.10	74.09
Unsupervised + Supervised Majority Voting (J-48, SVM, NB)	**91.17**	**80.59**	85.36	74.55

Although the neural network architectures we used are supervised models, we show them separately from the classical supervised models, since they are not subjected to different feature weighting schemes as in Table 4.3. Table 4.5 shows the performances of the deep learning architectures. Employing static word vectors (word2vec) results in worse performance compared to the non-static word vectors, which are learnt from scratch. The reason is that in the non-static case we learn the embeddings during the training phase by exploiting the sentiment information. On the other hand, the static vectors are pre-trained, and we do not use class labels in the learning phase of word representations. As in all other experiments, the use of word embeddings and neural models led to lower success rates for the Twitter dataset compared to the movie dataset. This can be attributed to the Twitter's distinctive and noisy jargon, which makes it more difficult to normalise the tweets successfully, and to generate meaningful and comprehensive representations of tweets. There are many OOV words that could not be normalised, thus they lack their corresponding word embeddings. Since there are

relatively less OOV words in the movie corpus, we could feed their available embeddings into the network and obtain better performance as compared to the Twitter dataset.

Table 4.5. Performances (%) for two neural network models using different word embedding types for the two datasets.

Embedding type	**Dataset**	**LSTM**	**CNN**
Static	Movie	87.98	89.69
	Twitter	75.84	78.58
Non-static	Movie	88.04	**90.25**
	Twitter	75.86	**78.74**

An interesting point that we observed in the CNN experiments is that when we perform intensification and negation handling, and take account of multi-word expressions, the success rates drop by about 2%. Since the filtering mechanism within the framework carries out these tasks inherently, our intervention proves to be not only unnecessary, but also harmful. For example, when we perform negation handling (e.g. *"etkileyici değil"* (not impressive) is changed as *"etkileyici_"* (impressive_)), SVM generates better results, whereas CNN does not. Thus, some feature engineering techniques may have opposite effects on different machine learning algorithms.

In the literature, neural network models used in this work are in general reported to outperform classical/traditional learning methods especially when fed with bulky datasets. However, the success rates we obtained using the LSTM and CNN models are not as high as those produced by the supervised methods used, especially with the 3-feats metric and the majority voting scheme (i.e. the ensemble classifier). That is, we found out that conventional machine learning methods could perform better than the two popular neural network models by relying on different and effective feature engineering techniques. For several scenarios, the neural network models could not compensate for the advanced features additionally implemented for the classical machine learning schemes. For example, in the SVM approach, we made use of multi-word expressions, whereas in the neural networks word embeddings for those expressions and

phrases were absent. Another case is that we can intensify the polarity strengths of words and feed these modified input vectors into the classical machine learning methods. However, we cannot intensify or modify the values of word embeddings that are generated in [111]. However, when we do not employ those feature selection methods, neural networks outperform the classical machine learning methods. We think that this is on account of the advanced nature of the feature extraction processes used in the conventional methods.

It can be argued that the performance of the neural models will improve if the OOV words are handled properly. There are several ways to decrease the number of OOV words and generate embeddings for such words as discussed in the literature [119, 120]. For example, the embedding of an OOV word can be taken as the average of the embeddings of words appearing in the same context window(s) [121]. The success rates of the neural models may increase if we include these words in the models. However, since most of the OOV words occur rarely and we remove the words that occur below a frequency threshold for several cases, this may result only in a slight increase in the performance. We leave a detailed analysis of this issue for future study, possibly for post-doctoral researches.

We also analyse the contribution of the preprocessing operations on the success rates. We show the effect of each operation in Table 4.6. We use the delta tf-idf$_{w,\,d}$ metric and SVM. Normalisation refers to the operations of deasciification, punctuation removal, and other tasks provided by the İTÜ normalisation tool. Noise elimination is the removal of tokens occurring less than seven times. In the MWE step, multi-word terms are included in addition to the unigram features. The steps denoted by emoticons, negation handling, and intensification correspond to including also emoticons in the feature set, processing negation words and suffixes, and taking intensifiers into account, respectively. In the partial surface forms step, we use partial forms formed of 90% and 50% of the top morphemes for the movie and Twitter datasets, respectively. As shown in the table, it increases the success rates with a significant margin for the

movie dataset and with a small margin for the Twitter dataset. The detailed analysis of partial forms will also be given in the next section. Finally, the *"all POS"* step refers to the case where all the text words are processed rather than only the four categories, namely noun, adjective, verb, and adverb, as in the previous steps. We see that each step contributes to the success rate. In the Twitter case, normalisation and removing the POS restriction also boost the performance because of the idiosyncratic nature of the medium and the large number of OOV words. In summary, normalisation, using partial surface forms, and all the tokens regardless of their POS tags are found to be the most effective methods to increase the success rates.

Table 4.6. Contributions of preprocessing modules to the performance (%) of the SVM method using the delta tf-idf metric.

Module	**Movie Dataset**	**Twitter Dataset**
No normalisation	83.09	59.75
Normalisation	85.88	68.21
Normalisation + noise elimination	86.81	69.57
Normalisation + noise elimination + MWE	86.98	69.71
Normalisation + noise elimination + MWE + emoticons	87.12	70.16
Normalisation + noise elimination + MWE + emoticons + negation handling	87.41	72.67
Normalisation + noise elimination + MWE + emoticons + negation handling + intensification	87.47	75.02
Normalisation + noise elimination + MWE + emoticons + negation handling + intensification + partial surface forms	89.91	76.12
Normalisation + noise elimination + MWE + emoticons + negation handling + intensification + partial surface forms + All POS	**90.98**	**79.54**

As a final experiment, we also evaluated the proposed approaches on three datasets in English in order to test the portability of the approaches across languages. To give an overview, we show the most important findings which are produced by a subset of the methods and the feature engineering techniques. We do not apply morphological operations on the words (negation suffixes, partial surface forms, etc.), since English has a simple inflectional, derivational, and morphological structure. The results are given in Table 4.7. For the SemEval 2017 dataset, in addition to the F1-scores shown in the table, we also computed the "average recall" values as stated in the relevant paper in order to reasonably compare our results with those of other participants. In the "unsupervised + supervised majority voting" scheme, we obtained a score of 77.9% and ranked 15th among 24 teams in total, including us. For this scheme, we found the optimal coefficients to be 0.6 and 0.4 for the supervised and unsupervised components, respectively, by leveraging the validation set. When we apply our other approaches and schemes, we rank worse overall. As in the case of Turkish, combining the unsupervised and supervised methods gives rise to the best success rates. When we compare the classical machine learning models and the neural models, we see that they have similar performances. We do not observe a large difference between the two paradigms as in Turkish, probably due to the simpler nature of the feature extraction process in this case (English).

Table 4.7. Summary of performances (%) of the unsupervised, semi-supervised and supervised methods, and their combinations for the three datasets in English.

Approach	**Movie**	**Twitter**	**SemEval**
Unsupervised	66.67	64.47	61.13
Semi-supervised	68.18	64.53	63.45
SVM (tf-idf)	70.99	72.02	72.02
SVM (3-feats)	73.97	74.72	73.48
Supervised majority voting	73.99	74.02	74.42
Unsupervised + supervised majority voting	**74.78**	**74.98**	**75.86**
CNN (Non-static)	74.11	74.53	73.73
LSTM (Non-static)	73.92	74.52	74.84

The most similar work to ours carried out for sentiment analysis in Turkish is the work proposed in [38]. They rely on unigram and bigram features with tf-idf weights, and achieve an accuracy of 89.5% on the same movie dataset. We observed in our work that combining the supervised algorithms with unsupervised approaches leads to better overall performance. We used paired t-test and the approximate randomisation technique to measure the significance of the difference between the two works. The result obtained with the combined method (91.17%) significantly outperforms the success rate obtained in [38] on the same dataset at $p = 0.05$. In [36], several supervised machine learning methods are also used and evaluated on the domain of political news. They analysed the effect of different types of features, such as unigrams and bigrams, adjectives, and a predefined list of polarity words, with term frequency and Boolean weights. They obtained an accuracy of 77%. To the best of our knowledge, employing a semi-supervised domain-specific approach, combining unsupervised and supervised approaches, utilising the partial surface forms method, and making use of both static and non-static word embeddings in neural architectures for Turkish are the novel aspects in this proposed approach.

4.2.5. Conclusion and Future Work

In this approach, we have developed a framework for two-class sentiment classification problem in Turkish. The reviews and tweets were first subjected to comprehensive preprocessing operations tailored for the language and then features were extracted based on the normalised texts. We proposed methods that follow three main approaches. In the unsupervised and semi-supervised approaches, the polarity score of words were determined using a set of antonym seed words and then the sentiment of a document is obtained as the sum of the scores of the words it contains. In the supervised approach, we used several feature weighting metrics including novel metrics. The supervised methods were analysed in two groups, which are the traditional learning algorithms and the deep learning models. Lastly, the third approach formed ensembles of classifiers and also combined the unsupervised and supervised approaches in a novel way.

We have evaluated all these methods on two Turkish datasets with different characteristics. We conclude that combining unsupervised/semi-supervised and supervised methods yields the best results in both datasets. This indicates that incorporating knowledge obtained in an unsupervised manner into the classification process seems to be necessary to obtain more successful results. To the best of our knowledge, this is the first thesis work in Turkish that extracts sentiment scores of words using search engines, that employs domain-specific sentiment analysis, and also that combines unsupervised and supervised schemes. We have also evaluated our methods on three corpora in English and achieved the best results with this combination scheme as well. This proves the portability of this approach to other languages. The unsupervised and semi-supervised approaches can be adapted to other domains, and languages besides English and Turkish easily by choosing a relevant set of antonym pairs.

The experiments have shown that the feature engineering techniques used in determining the features and using suitable weighting schemes are the most important aspects in the supervised case. We observed that some preprocessing operations and specific feature selection methods may hamper the performance for some of the learning algorithms, such as negation handling and intensification in the CNN framework. Another finding is related to the agglutinative nature of the language in the sense that, in addition to the root forms of the words, making use of the morphemes within the surface forms gives rise to better results after performing normalisation. We have also shown that conventional machine learning algorithms using novel and effective feature engineering techniques can outperform the deep learning models, which are the hottest topic in the field of machine learning as of 2020.

We will extend our approach in the future by (1) applying it to other domains, such as hotel or restaurant reviews, (2) implementing different feature engineering techniques, and (3) performing error analysis for the "unsupervised + supervised method" to see the differences in the decisions of the classifiers. We also plan to (4) utilise sentiment lexicons in English, translate them into Turkish, and combine these polarity scores in lieu of unsupervised scores with supervised scores as discussed previously. A

combination of supervised and unsupervised approaches can also be employed in neural network models and word embeddings in a similar manner.

4.3. Major Contribution #2: Morphological Analysis for Sentiment Analysis in Turkish

In this approach, we examined the effect of a fine-grained morphological analysis for sentiment analysis in Turkish. Although the previous approach is applied on datasets in two languages, this approach is applicable only for Turkish and other agglutinative languages. In this case, partial surface forms as described beforehand are employed as features. After generating these feature sets, where some morphemes attached to their root forms are sentimentally more discriminative, we fed document vectors into classical machine learning methods to predict the polarities on a review-basis.

4.3.1. Results

We performed an elaborative analysis about the partial surface forms technique we used for the Turkish texts. We examined the impacts of employing various ratios of suffixes and inducing morpheme polarity lexicons based on more than one corpus, and performed a comparative analysis with respect to surface and root forms of words. We empirically show the results in Figure 4.7 (movie corpus) and Figure 4.8 (the corpus of tweets), which are the same corpora discussed in the previous subsection. The performances have been achieved by employing the 3-feats technique and the SVM model. The horizontal axis in these figures represents the ratio of suffixes with the maximal confidence scores that are covered by the partial surface forms. The ratio 0% corresponds to the root form, whereas 100% corresponds to the surface form. The partial surface forms have been experimented in two different settings: We either employ one dataset (only the training set of the whole corpus) or process two datasets (again, only the training set of the target corpus and both the training and test sets of the other corpus). These figures show that when relying on only a subset of all

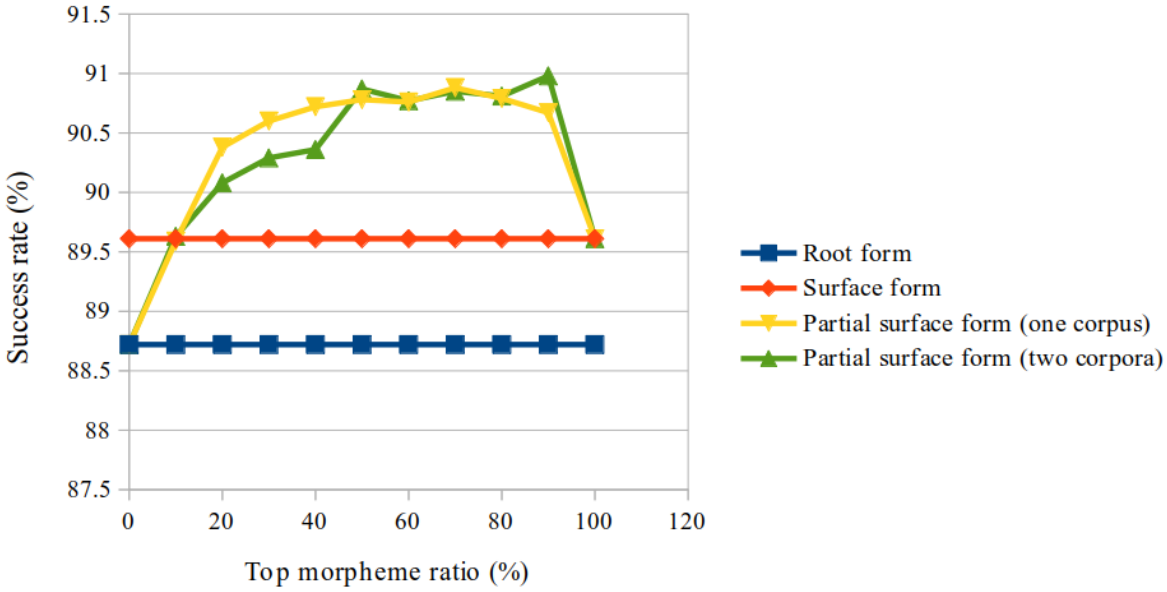

Figure 4.7. Effects of fine-grained morphological analysis with respect to different top (most discriminative) morpheme percentages on the Turkish movie corpus.

the suffixes by taking account of their discriminative strength, our approach could perform better than the models using the root form and the surface form for almost all the top (morpheme) percentage values. We relate this good performance of partial surface forms to various factors. Some morphemes that have neutral-like polarities had better be filtered out, since these do not carry sentiment information. By processing only sentimentally the most discriminative morphemes by exploiting the polarity label information, we generate more robust sentiment representations for tokens occurring in an agglutinative language. When using several datasets at the same time in inducing morpheme polarity lexicons, we found out that this is especially useful for the Twitter dataset (Figure 4.8). We hypothesise that the reason is that the additional movie corpus used as a supplementary dataset is a less noisy and a bulkier corpus carrying more information.

Table 4.8 summarises the impacts of differing morphological settings on the performance. In the partial surface form scenario, we include only the best success rates given in Figures 4.7 and 4.8, which correspond to 90% of the suffixes for the movie corpus and 50% for the tweet corpus. The best performances are achieved by employ-

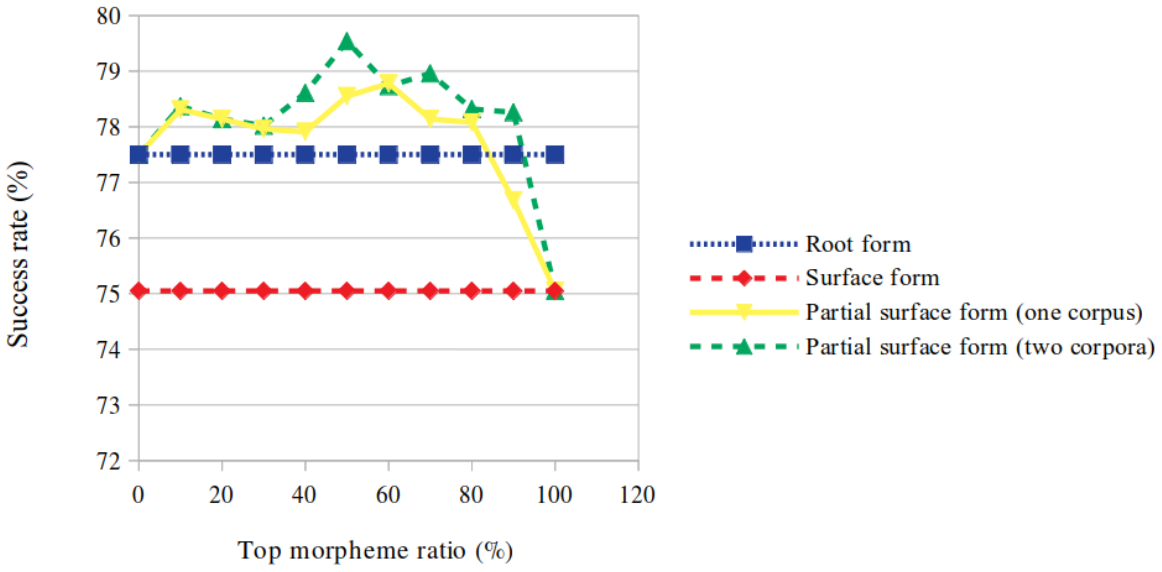

Figure 4.8. Effects of fine-grained morphological analysis with respect to different top (most discriminative) morpheme percentages on the Turkish movie corpus.

ing the partial surface forms metric. Thus, we have observed that making use of a set of discriminative morphemes is the most effective method for sentiment analysis in Turkish. Our review classification results are significant at $p = 0.05$, when we compare the partial surface forms technique to root and surface forms.

Table 4.8. Summary of the impacts of fine-grained morphological analyses on the success rates (%) for the two Turkish corpora with respect to the 3-feats scheme and the SVM method.

Morphological setting	**Movie**	**Twitter**
Root forms	88.72	77.50
Surface forms	89.61	75.05
Partial surface forms	**90.98**	**79.54**

4.3.2. Conclusion

Since Turkish is a morphologically-rich language, taking into account morphological features as well boosted the performance. Some morphemes are observed to be neutral-like, whereas some others are likely to express sentiments. That is, sur-

face forms and root forms can both show weaknesses compared to the incorporation of the most sentimentally effective morphemes. The supervised algorithm we generated may be adopted for other morphologically-rich and agglutinative languages without producing any hand-crafted sentiment features specific to morphemes, and successful results could be arrived at as long as labelled data are available for one corpus (or more). In the case that no training data exist for a language, the previously mentioned unsupervised and semi-supervised methods can be used to generate polarity lexicons in the first place, and then the morpheme sentiment lexicon can be induced. However, as can be expected, we would not achieve as high accuracies for these cases.

As future work, we plan to (1) determine the sentiment scores of morphemes by implementing a neural network model which can more effectively capture this information and to (2) perform an error analysis.

4.4. Major Contribution #3: Combining Recurrent and Recursive Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations

In the literature, most studies conducted on sentiment analysis use either recursive or recurrent neural network frameworks alone. Recurrent networks can model the temporal effect and the sentiment information can accordingly be propagated throughout a text. By contrast, recursive neural networks can capture syntactic structures of the texts and the polarity information can be exploited in the training phase. Only several related works merge both these models in an ensemble neural network for polarity detection. In this approach, an original neural network model is proposed such that both these neural models are combined for ABSA. Relying on constituency and dependency parsers, we initially break down each review into sub-reviews (subclauses) that bear the sentiment information related to only the corresponding aspect terms. After we generate and train the recursive neural models based on the parse trees of the sub-reviews, we use their output in the recurrent model. This ensemble approach is evaluated on two corpora of different categories in English. Results we arrived at are

state-of-the-art and we outperform a study (an RNN-only framework) which we chose as a baseline by a significant margin for the two corpora.

A single review in general bears a sentiment about an entity, such as a service, product, or political act. However, reviewers can also comment on different aspects of the same entity in a single review. For example, in *"I found the ambiance of the restaurant great overall; however, the main dish was served a bit cold and lately."*, opinions expressed towards the aspects *"ambiance"* and *"dish"* are positive and negative, respectively. Therefore, we need to perform a fine-grained analysis rather assign a single overall polarity score to a review.

If multiple aspects are commented about in a single review, the polarity of these each is likely to affect the following or preceding aspects as well. For instance, in *"I liked the taste of pizza more than that of the chips."*, it is observed that the negative sentiment of the aspect *"chip"* is expressed in an indirect way by the first aspect *"pizza"*. That is, the polarities of aspects are likely to affect one another in the same review. Additionally, several conjunctions (e.g. *"and"*, *"also"*, *"however"*, and *"but"*) lead to aspects sharing their polarities with other aspects or influencing the polarities of other aspects in the same review. In the sentence *"The quality of the display of this laptop is so sensational, so is the price thereof."*, a correlation exists between the polarities of the aspects because of the conjunction *"so"*. A work [2] develops a recurrent neural network approach by exploiting such inter-aspect relations, when modelling such scenarios.

Such RNN models utilise the sequence information in a series of objects. The effect of the sentiments is thereby propagated across the same text, be it in a forward or reverse setup. Nonetheless, the grammatical information is not covered by such models. For example, after a sentence is broken down and parsed into its constituent phrases, the words in the same subtrees/subgroups are more likely to be semantically, syntactically, and sentimentally more akin to each other than those words in other subtrees. Hence, relying on recursive neural networks is useful in that the words in

the same subtrees can be of the same or similar sentiment scores. If we merge this structural, sentimental information into other neural network models, such as RNNs exploiting the temporal information, we can arrive at more comprehensive sentiment analysis frameworks. As such, sub-models in this ensemble models can each compensate for what the other lacks.

In this approach, a novel framework is proposed for ABSA [55]. To exploit the sentiment information of aspects, we merge recursive and recurrent neural network models. In the recurrent sub-model, we rely on an off-the-shelf framework [2] which employs gated recurrent units. In the recursive sub-model, we describe original approaches to extracting sub-reviews, each corresponding to aspect term groups, from whole reviews. All sub-reviews are captured such that every one of them is modified by one polarity at most. Each sub-review is considered a separate review and these are trained by recursive neural networks. We extract the root vectors from these sub-reviews in a distant-supervised manner, whereby these root embeddings represent the aspects therein. We then feed these vectors as input into the recurrent neural networks and the effect of sentiments is propagated along with other information.

We test and evaluate the original approach using two corpora, which are restaurant and laptop datasets provided in the Task 4 of SemEval-2014. Our approach performs better than the baseline study [2] significantly. Employing the recurrent network only cannot capture polarity information as effectively. The recursive model helps create the *"optimal"* sentiment root embeddings of the sub-reviews, which are later merged with the related aspect component vectors in the recurrent model. When incorporating a recursive model trained with a distant-supervised approach into a recurrent model and, as such, forming a novel ensemble framework, we boosted the performance, where an increase of 1.6% on average is observed for the two datasets.

Our research objective in this approach is to obtain better performances for ABSA. To achieve this goal, as mentioned in a preceding chapter, we address the following research questions. Could we enhance recurrent neural networks by incorpo-

rating distant sentiment information? Can we merge recursive and recurrent neural models in an ensemble form? If so, what is the effect? Why do such ensemble approaches perform better than those modelling only one of its sub-models? Can we break down reviews into sub-reviews/subclauses using syntactic parsers such that each sub-review bears only one related sentiment expressed towards the aspect or aspects therein? Is employing dependency parsers a better choice over the use of constituency parsers in ABSA? If so, what is the reason?

4.4.1. Methodology

We conduct ternary aspect-based sentiment classification in this approach, where the aspects are either negative, positive, or neutral. First, we perform basic tokenisation and other preprocessing operations, such as lowercasing, on texts employing the spaCy library [101]. Then, we generate sub-reviews from whole reviews corresponding to each aspect by using a constituency and a dependency parser. Each of these sub-reviews may contain one aspect or more. In the end, we merge recurrent and recursive neural networks into an ensemble form as mentioned. In the remainder of this section, we first give a brief overview on the baseline recurrent model. Then the proposed recursive neural network model and the ensemble model based on these two sub-models are described.

4.4.1.1. Submodel 1: Recurrent Model. In this sub-model, the framework [122] developed by the study [2] which we regard as the baseline method is used and enhanced by us. This model employs inter-aspect relations in such a way that aspects can have an impact on the polarities of the previous or succeeding aspects. We keep the hyperparameters of this model the same to perform a consistent, comparative analysis. We arrive at better results for two corpora of different genres when integrating this recurrent network with a recursive network in an original way.

We explain the baseline study briefly as follows. The GloVe vectors are fed as input into the system. In aspect-aware sentiment representation (AASR) modules, the

contextual information is propagated throughout the word sequence. Additionally, an attentive mechanism is relied on to identify and boost the effects of the polarities that are expressed towards the modified aspects. This is repeated for each aspect term group so that its polarity has an impact on those of the others. A softmax classifier is used to identify the sentiments of the given aspects over the top layer. A mechanism named "multiple hops" is also used in their study. This concept is defined as follows: The hidden outputs of the text are repeatedly fed as input into the system several times. Hereby, a fine-grained representation of aspect terms is modelled. This approach is visually summarised in Figure 4.9.

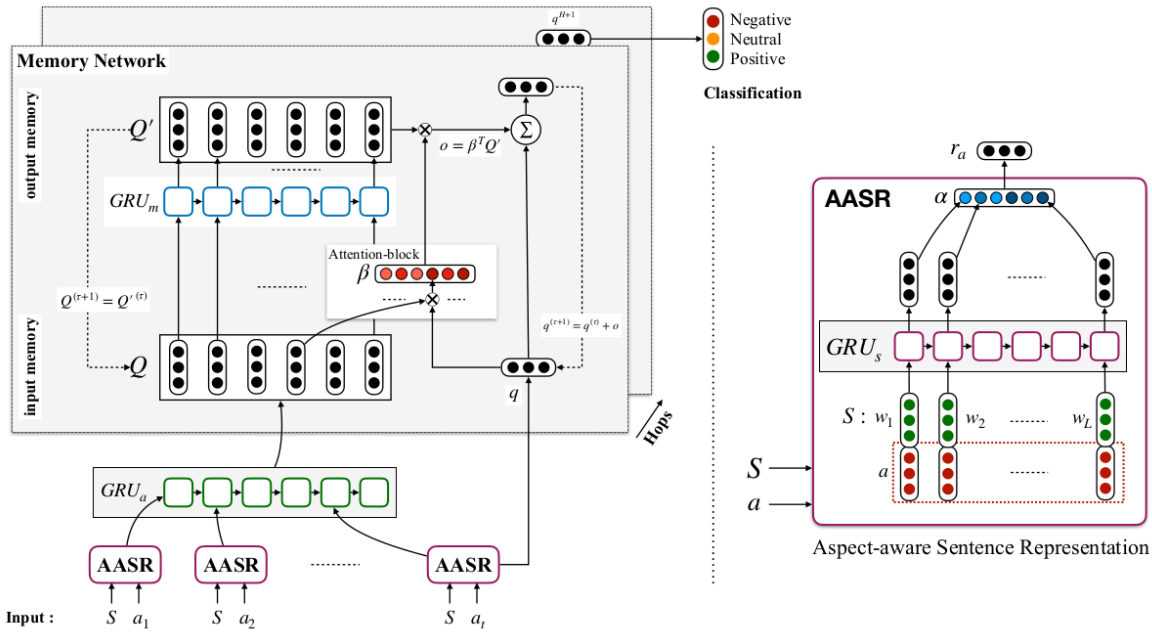

Figure 4.9. The architecture of the baseline study [2]. AASR denotes *Aspect-Aware Sentence Representation.*

4.4.1.2. Submodel 2: Recursive Model. Recursive neural network architectures are made use of to train models by modelling the grammatical structures of the sentences. For example, in the review *"I adored this flick!"*, the verb *"adore"* bears a positive polarity orientation and this modifies an object noun, which is *"flick"*. The sentimental and semantic information are thereby captured from the text more saliently and successfully as compared to other models (feed-forward neural networks, n-grams, bag-of-

words, etc.). In the remainder of this section, the constituency and dependency parser sub-models we developed in this sub-approach are described. We combine these models with the recurrent network model as will be explained and discussed in Section 4.4.1.3.

Constituency Parser Sub-model

Constituency parsers generate parse trees by decomposing sentences into chunks. For the sentiment analysis problem, breaking down the texts into such phrases can help organise polarity information in a well-defined, structured manner. For instance, if a negator (e.g. *"not"*) explicitly appears in a chunk, this would lead to a sentimental change in an opposite direction. A similar mechanism holds true if a phrase is followed by a contrastive conjunction, such as *"however"*. Modelling such structures along with the polarity information can help us achieve a better performance [63].

According to this sub-model, we rely on the Stanford CoreNLP framework [123] to parse the sentences into constituency phrases/chunks along with polarity labels. We give an example that visualises the recursive structure of a review in Figure 4.10. In this example, every node of the parse tree is modified by a polarity category, which may be very negative (--), negative (-), neutral (0), positive (+), or very positive (++). These polarities are identified by the above-mentioned tool in a distant-supervised way. That is, this approach does not exploit the label information present in the training dataset. As said earlier, this sub-model can effectively capture the negation and scope relations in such parse trees.

Before using the parse trees in our recursive model, we first identify aspect groups/chunks. In other words, per aspect term, we create a subtree covering the aspect term. We then train these sub-reviews/subtrees in the recursive model separately. The reason behind this approach is that the same review may have various aspects with opposite sentiments. Therefore, when we break down reviews into sub-reviews with respect to aspect terms, this helps us gather the relevant sentiments of these aspects. The fine-grained analysis can thereby boost the performance for ABSA.

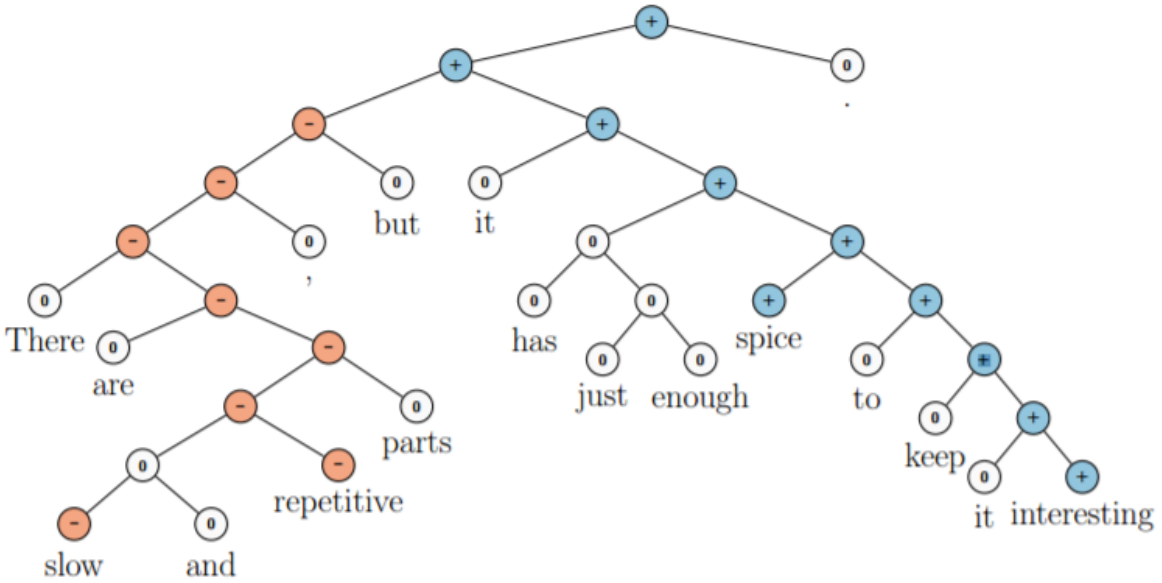

Figure 4.10. A sample review parsed into its chunks/phrases along with their polarity categories, which range from very negative to very positive (--, -, 0, +, ++).

Our original approach to creating a subtree per aspect term is described as follows. We first start scanning a tree from the leaves (terminals) that represent aspect term words which are labelled in the training data and then go in upward direction in the tree. If a node whose polarity is non-neutral is encountered, we hypothesise that this sentiment word modifies the corresponding leaf aspect term and we cut the parse tree at that point. As such, we take this node and all of its child nodes as the aspect subtree for the corresponding aspect term. Accordingly, this subtree represents the aspect term that is covered in it. It is therefore probable that the same sub-reviews/subtrees can encompass different aspect terms, as in *"I loved the ambiance and service overall."* Additionally, several rules are defined for negators. For instance, when a negating word appears at a higher level than the node where we cut the tree, we keep expanding the subtree in an upward direction until we cover the negator node and all of its children into the same subtree. Then, we cut the parse tree at that point. This is usually the case that the negating words lie at higher node levels compared to the words modified by them. Lastly, after generating these sub-reviews, we train them disjointly as if they are different reviews by the conventional recursive neural model [63].

We arrived at better results as compared to the baseline when taking sub-reviews into consideration and training them separately for our ensemble framework. Nevertheless, this sub-model has a drawback. As mentioned, three gold polarity labels are defined for aspects provided in the SemEval-2014 datasets, which are used in the evaluation stage. However, if a node of positive or negative polarity is encountered, we do not expand the tree per aspect term in the bottom-up manner any more. In other words, we do not take account of the case whereby an aspect term can be neutral. Yet another deficiency about the constituency sub-model is that we can fail in generating robust or meaningful subtrees from time to time. We hypothesise that the reason may be that constituency parsers have an incompetence: They cannot function as effectively as dependency parsers for the opinion mining task. Dependency parsers define relationships (e.g. modification) much more clearly in contrast to constituency parsers. In dependency parser outputs, sub-reviews are in general linked to each other with predefined relationships. Thus, we are more capable to accurately extract these relevant subclauses from the whole sentences. Such a capability is not provided by constituency parsers.

After having generated the sub-reviews using the constituency parser tool, we rely on an open source recursive neural network framework [124] when training these trees beforehand. To handle the overfitting problem, we make use of a development dataset. The maximum epoch number is chosen as 30, since when exceeding this value, in general overfitting occurs. We treat the vector size as a hyper-parameter and define the value set as {30, 50, 100}. The embeddings at the nodes encode the sentimental information in the parse tree and are updated in the training phase. After the training epochs are completed, we employ the root vector of each sub-review per aspect in the ensemble model.

Dependency Parser Sub-model

As said, when creating sensible aspect subtrees to be used in the ensemble framework, constituency parser can be unsuccessful. In the literature, it is also reported that

these parsers cannot usually perform as well as dependency parsers in the sentiment analysis domain [125]. Therefore, dependency parsers are also made use of in this approach to generating sentiment embeddings for aspect terms, which are later fed into our ensemble approach.

Words in a text are connected to each other based on the binary relationships between them by dependency parsers. Vertices in the tree are tokens in the sentence(s) and edges are labelled by relationships. Parents, which are the source of an edge, modify their child nodes. For instance, in *"There are slow and repetitive parts."*, the words *"slow"* and *"repetitive"* modify the word *"parts"*. This example is visualised in Figure 4.11. In this example, the relationship *"amod"* stands for adjectival modifier. In dependency parse trees, for a word can be linked to another word with a modifier relationship, these parsers can help capture the sentiment information as well more effectively and accurately than constituency parsers do. The spaCy library is employed to generate dependency relationships from a text.

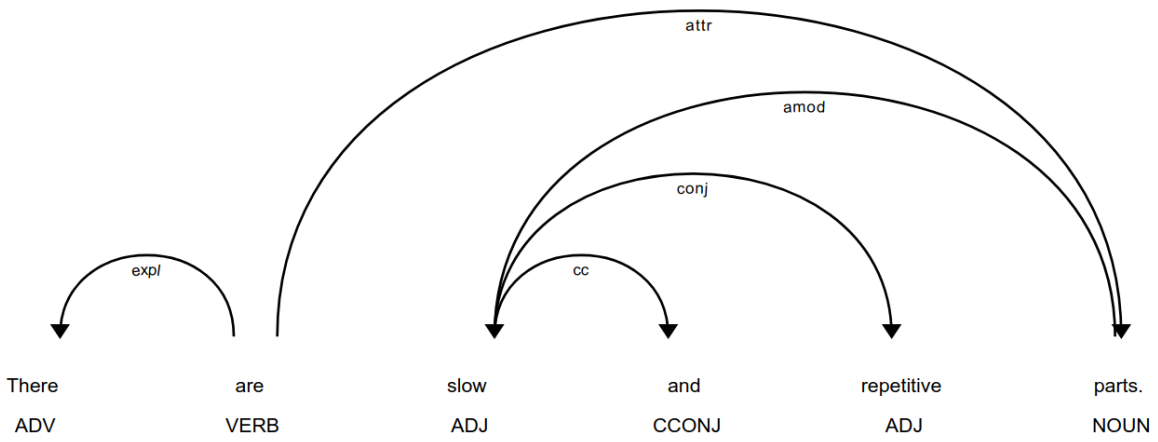

Figure 4.11. Example of a sentence decomposed into its relationships.

As is the case in the constituency parser model, first, we extract sub-reviews per aspect term from whole reviews relying on the dependency parser. To achieve this, a few rules are defined to decompose a whole dependency tree generated for a review into sub-review trees. If the word's part-of-speech tag is a verb (be it in gerund, infinitive, or any other form) and this is linked to conjunction or clausal component relationship,

this marks the existence of a sub-review. In other words, all the child nodes of a verb that are linked through relationships other than those two are incorporated into the corresponding sub-review in a recursive manner. For example, in *"There are slow and repetitive parts, but it has just enough spice to keep it interesting."*, the word *"are"* is the main verb of this sentence. The conjunction relationship links it to the secondary verb *"has"*. The dependency tree is therefore cut at this point. We cover all the children of the verb *"are"* except *"has"* (and its respective children) in a recursive manner and build a sub-review. We perform the same process for the word *"has"* with regard to its children. Hence, the resulting sub-reviews are *"There are slow and repetitive parts."* and *"But it has just enough spice to keep it interesting."*

After generating sub-reviews, we remove subtrees (subclauses) which have no aspects and are therefore considered unnecessary. We then filter out the preceding and following redundant conjunctions, such as *"and"*, if any. We also discard the punctuation marks (e.g. *"!"*) at the beginning and end of the sub-reviews in the first place. Then, we append the punctuation mark appearing at the end of the whole review (e.g. *"!"*) to the end of each sub-review instead to arrive at consistency. In case there are not any clausal components or conjunctions in a review, we hypothesise that there are not any possible subclauses/sub-reviews and the review has only one single sentiment. However, a single sub-review (subtree) can consist of one aspect or more than one aspect. We accordingly obtain more robust sub-reviews conforming to these rules compared to the outputs of the constituency parser.

We give a sample text that visualises sub-reviews created by the constituency and dependency parser models in Table 4.9. The reader can see that all the sub-reviews built are relevant only to the aspects therein for the dependency parser model. Additionally, the subtree covers aspect-specific sentiments expressed only towards their aspects. Nevertheless, in the constituency parser model, when decomposing a review into sub-reviews, outputs shown in the table are not as consistent. For instance, the model predicts the words *"French"* and *"gourmet"* as positive by a polarity score of 3 according to the Stanford NLP library. The bottom-up approach defined for the

constituency parser therefore stops scanning the tree, when a word of a positive or negative polarity is come across by the algorithm. That is, we cut the parse tree at that point. We visually show the last sentence (review) broken down into its sub-sentences (sub-reviews) in Figure 4.12. Each of the three encircled parts in the figure is a sub-review.

Table 4.9. Example sentences and the sub-sentences for each aspect term built employing the constituency and dependency parsers. The gold aspect terms are underlined.

Review	**Sub-reviews generated**	
	Dependency	**Constituency (Bottom-up)**
It may be a bit packed on weekends, but the vibe is good and it is the best French food you will find in the area.	**1.** It may be a bit packed on weekends. **2.** The vibe is good. **3.** It is the best French food you will find in the area.	**1.** Packed on weekends. **2.** The vibe is good. **3.** French food.
I love and I know gourmet food by excellence!	**1.** I love and I know gourmet food by excellence!	**1.** Gourmet food.
The vibe is very relaxed and cozy, service was great and the food was excellent!	**1.** The vibe is very relaxed and cozy! **2.** Service was great! **3.** The food was excellent!	**1.** The vibe is very relaxed and cozy! **2.** Service was great! **3.** The food was excellent!

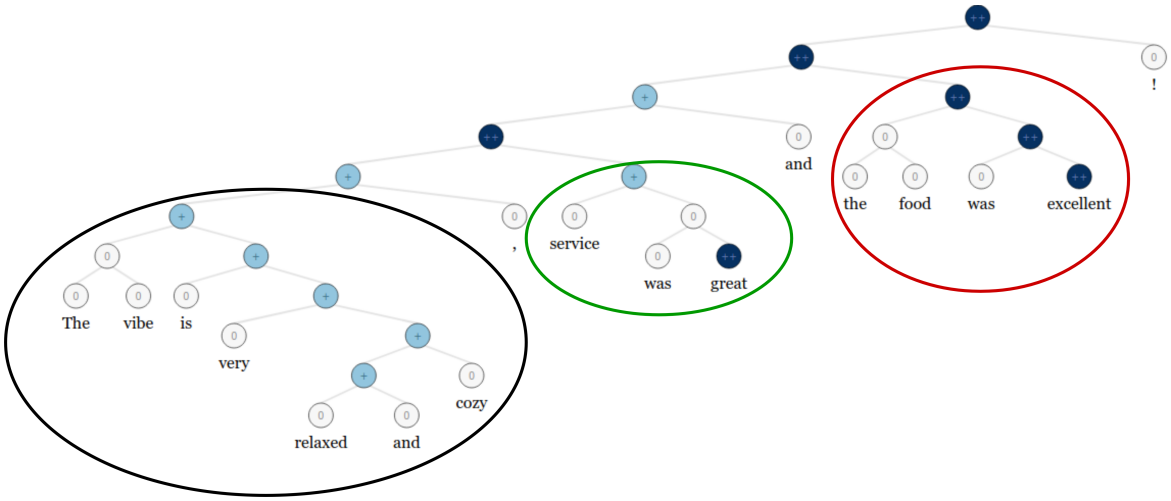

Figure 4.12. The text *"The vibe is very relaxed and cozy, service was great and the food was excellent!"* decomposed into its sub-reviews. Each encircled part in the figure is a sub-review.

After we build sub-reviews from reviews making use of the dependency parser, we train the recursive sentiment model. Accordingly, each parse tree representing a sub-review is trained in a recursive way, independently of one another. These are then fed as input into the ensemble framework. We employ a publicly available code repository for this sub-model [64]. We slightly changed the parameters of this framework to adapt it to our corpora. In this sub-model, as in the constituency parser model, each node is modified by a polarity label. A sentiment lexicon [1] in which the words are labelled as -1 (negative), 0 (neutral), or 1 (positive) is leveraged to identify the polarities of words. If this lexicon does not include a token, we assign it the score of 0. This polarity lexicon is used in identifying the sentiment scores of each node in the dependency parse tree. Despite the fact that the labels can belong to one of the five sentiments for the constituency parser model, conducting a coarser analysis in the dependency parser model helped us achieve higher success rates in the final ensemble framework. Exploiting these sentiment labels, we recursively train these vectors at the nodes of the parse tree.

Our models have been evaluated for various vector lengths of 30, 50, and 100. After the training phase for the recursive neural network is completed, the root vectors for each aspect group are fed as input into the ensemble framework. As we empirically explain and show in Section 4.4.2.3, we have separately assessed the performances of vectors of different lengths. As will be shown Section 4.4.2.3, the dependency parser model performs significantly better compared to both the baseline model alone and the ensemble approach with the constituency parser module.

Apart from this basic approach, two additional, different settings are also tested such that gold polarity labels of aspects corresponding to the nodes in the parse tree are taken into consideration. For the first variant, the leaves in the tree representing aspect terms are labelled as the same gold polarity values of aspects that are present in the training data instead of using the information in the sentiment lexicon. These sentiment scores can be either -1, 0, or 1. These are conducted for the training stage. On the other hand, in the testing phase, we perform majority voting scheme to identify the polarity of an aspect term in the test data by looking up the polarity values in the training dataset of this aspect. In the second scenario, the polarity of the root node in the tree is assigned the gold label of the aspect it covers. The reason behind that is that the root node is the most effective component when propagating sentiment through the tree. The above-mentioned majority voting process is also performed for this variant as well. Nonetheless, although we used the gold sentiment labels present in the training data, we observed a decrease in the accuracies. We think that it stems from the fact that combining sentiment information coming from different sources disarranges the consistent structure of the tree. For instance, a word's sentiment score can be -1 according to the polarity lexicon. On the other hand, this score could be defined as 0 in the training data as a gold label. Thus, this would lead to an ambiguity. Therefore, when we used the sentiment scores relying on only one source (i.e. sentiment lexicon), we arrived at better results. That is, this approach produces a more consistent sentiment structure for the text and the model would remain more robust. We will discuss the related results in Section 4.4.2.3.

4.4.1.3. Ensemble Framework Combining Recursive and Recurrent Models. The above-described recursive and recurrent models are integrated into an ensemble form in a hierarchical manner. In this scenario, the root embeddings of the parse trees trained in the recursive network are employed as input in the recurrent network. The baseline study [2] for this approach only makes use of GRUs, a recurrent model, with an attention mechanism, as explained in Section 4.4.1.1. Our recurrent sub-model is the same as this. The second sub-model we relied on is a recursive neural network using a constituency or dependency parser. We also extract sub-sentences from whole sentences by leveraging these tree parsers in an original manner. As mentioned, the recurrent sub-model captures the temporal information being propagated throughout the text, whereas the recursive sub-model extracts the grammatical and sentiment information from the reviews in a fine-grained scope. That is, those two sub-models can be considered to complete each other in areas where they cannot perform as robustly and we obtain more powerful representations when combining these two. We show the ensemble framework in Figure 4.13.

As mentioned, in this approach, in addition to the recurrent baseline model, we have also employed the corresponding root embeddings of the parse tree that covers the affected aspect and incorporated it into the ensemble model. The module named memory network is the same as that in the baseline study, which is only a recurrent model. The sample given on the bottom left demonstrating the sub-review generation process is the same as in Figure 4.12. $Subtree_i$ corresponds to the second sub-review. In this case, since three sub-sentences are generated from a single sentence, three AASR modules are produced. In this approach, we first break down reviews into sub-reviews, each of which holds the relevant sentiment about the affected aspect. Then we train those sub-reviews separately in parse trees. After that, we feed the root embeddings of these trees into the recurrent model. In AASR modules, aspect term embeddings and root embeddings of the related aspect subtrees are concatenated along with all sentence vectors. The GloVe embeddings of sentence words are utilised. If an aspect consists of more than one word, we take their average to represent the aspect group as a whole.

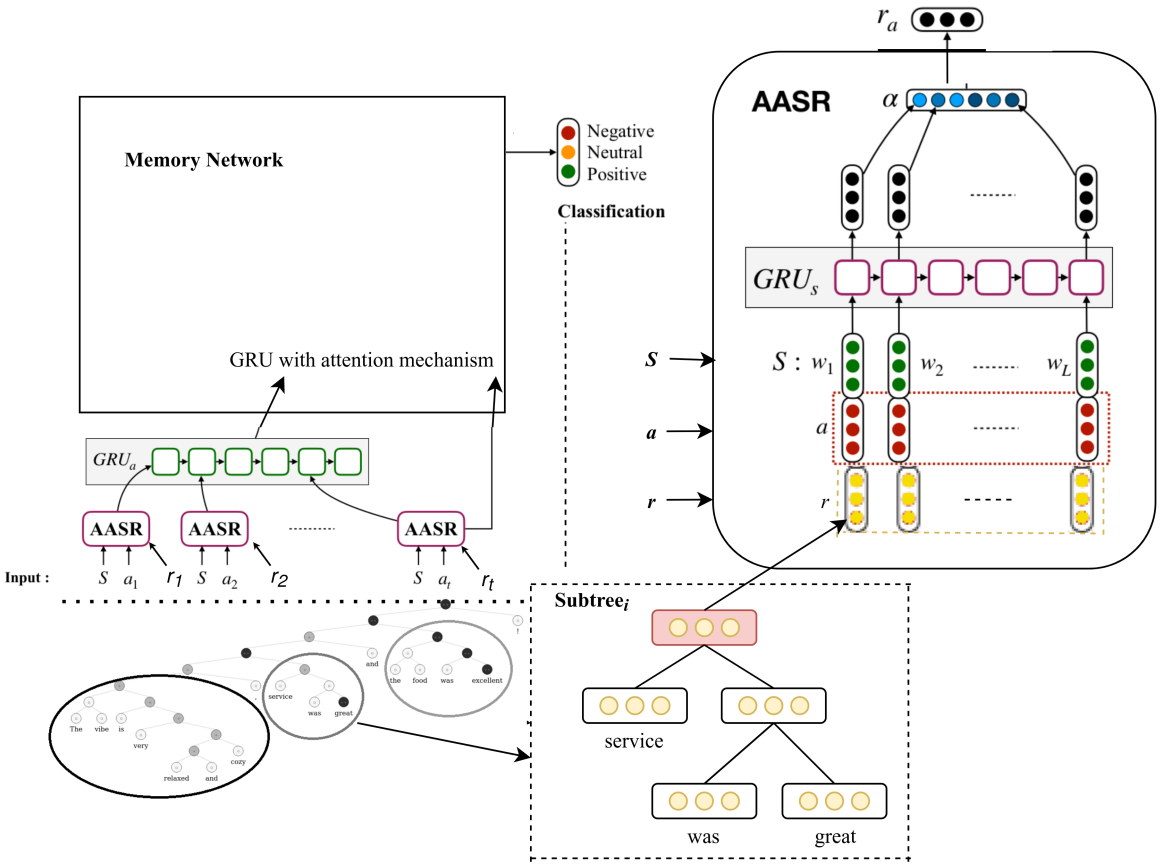

Figure 4.13. Visual summary of our proposed ensemble framework.

The main intuition behind our using root vectors is that these embeddings capture the overall sentiment, semantic, and syntactic information of the sub-review tree more comprehensively compared to the nodes below its level. The propagation process of sentiments throughout the parse trees is mostly affected by this root node. When we have also taken these sub-review parse trees and root embeddings into consideration, we observed that the recurrent model is enriched by incorporating additional grammatical and sentiment information in a novel way as well.

4.4.2. Experimental Evaluation

In the remainder of this section, first, we describe the corpora we utilised for ABSA. We then explain the hyper-parameters we have used. Lastly, we give a detailed explanation on and discuss the experimental settings and the results we obtained.

4.4.2.1. Datasets. We evaluated our methods on two datasets which are provided in the Task 4 of the SemEval-14 competition. These two corpora are restaurant and laptop reviews. In both these datasets, aspects are given in advance. That is, the competitors only try to find out what the polarities of these aspect terms are. Therefore, we have not developed a specific approach to extracting aspect terms from the datasets for this task. Each aspect is assigned a sentiment gold label, which can be negative, neutral, or positive. These are visualised in Table 4.10. The statistics state the distribution of sentiments across the datasets.

Table 4.10. Details of the datasets we used in this approach.

Dataset		**Positive**	**Neutral**	**Negative**
Restaurant corpus	Training	2,159	632	800
	Test	730	196	195
Laptop corpus	Training	980	454	858
	Test	340	171	128

We have not performed cross-validation, since the datasets are split as the training and test sets in advance. We split 10% of the training set as the development set to handle the overfitting issue in our neural network models and to detect the optimal values among hyper-parameters.

4.4.2.2. Hyper-parameters. Three publicly available code repository are utilised for this approach. We show the hyper-parameter sets in Table 4.11. Two of these are defined for the recursive neural network models, in which dependency and constituency parse trees are trained disjointly and independently of each other. Hidden states at each node are trained in these parse trees. The other setting is the same recurrent model as in the baseline. Hidden vector sizes are set at 30, 50, and 100, and we tried to figure out which one of them produces the optimal results. We used a validation set to find these values maximising the performance. When we choose the size of hidden vectors as 100, we observe the overfitting to occur for all cases. We think the reason is that that makes the model too complex. At the bottom-most layer, we feed the GloVe

vectors of the size 300 corresponding to tokens in the text. On the other hand, the vectors at intermediate nodes can be of various lengths. For the constituency parser, we learn the leaf embeddings from scratch, because the code repository we relied on does not allow us to use pre-trained vectors. The hyper-parameters utilised in the baseline approach are listed in Table 4.12.

Table 4.11. Hyper-parameters made use of in the recursive and recurrent neural networks. *"Optimal size"* corresponds to the optimal size of embeddings at intermediary nodes in trees.

Hyper-parameter		**Constituency parser model**	**Dependency parser model**	**Recurrent model**
Learning rate		0.001	0.05	0.001
Maximum number of epochs		30	30	30
Batch size		20	25	30
Optimal size	Restaurant	50	50	-
	Laptop	30	30	-

Table 4.12. Hyper-parameters made use of in the baseline model.

Hyper-parameter	**Laptop**	**Restaurant**
Hop Count	10	3
GRU Hidden State Size	400	300
GRU Output Size	400	350

4.4.2.3. Results. We list the accuracies obtained for the two corpora in Table 4.13. In the first row, the results for the baseline approach are given. The two rows below it are the accuracies produced for different recursive neural network model settings, which are later integrated to the ensemble framework. First, the results for the constituency parser are given. Then, those for the dependency parser are listed. The three approaches described in Section 4.4.1.2 are labelled *"root"*, *"leaves"*, and *"gold aspect"*.

In the *"root"* metric, the root vectors of the trained parse trees are fed into the GRUs network. For the *"leaves"* feature, instead of root vectors, leaf aspect embeddings learnt are employed. On the other hand, *"gold aspect"* models the scenario in which the sentiment labels present in the training dataset are exploited as explained in the previous subsections. That is, the label of the root node is assigned the same polarity of the affected aspect in the corresponding sub-review. We also utilised gold labels only for the leaf nodes in the parse trees, that is, not for the root node. As mentioned, for identifying the polarities of the nodes in the test parse trees in advance, we performed a majority voting scheme taking account of the frequency statistics in the training data. However, if an aspect word in the test data does not occur in the training data, we exploit the label information in the sentiment lexicon. However, we do not list the results of this last scheme in the table, since it does not produce state-of-the-art and significant results. If the reader has a hard time comprehending these concepts, he or she can refer to Section 4.4.1.2. The embedding size corresponds to the lengths of vectors at the nodes in the parse trees. Here, we test and assess the impact of embedding sizes only for the recursive neural sub-model. The reason is that we wanted to make a consistent comparison between our model and the baseline approach utilising the same hyper-parameter settings.

We empirically show that we outperform the baseline approach (IARM) for both corpora when the root scheme is utilised. This holds true for both the dependency and constituency parser settings. According to these results, for the laptop dataset, we outperformed all the 26 teams including us that participated in the Task 4, aspect-polarity detection in the SemEval-2014 competition. When it comes to the restaurant reviews, we rank second among all the teams. We attribute our success to the following reason. The top-performing teams rely on the SVM classifiers, into which they feed hand-crafted features among others. However, in our approach, a robust ensemble framework is built combining two different deep neural network models. It is well-stated in the literature that in general conventional machine learning methods are not as effective as deep neural networks and can hardly compete with them [58]. Deep learning networks generate features on their own and rely on more complex and more

Table 4.13. Accuracies (%) obtained for the baseline method and our ensemble approach which integrates the recursive neural network model into the baseline recurrent network. Results for different embedding size and other settings are listed.

Model	**Feature**	**Recursive NN vector size**	**Accuracy (%)**	
			Restaurant	**Laptop**
Baseline (IARM) [2]			80.00	73.80
Constituency Parser + Recurrent NN	Leaves	30	77.98	72.23
		50	79.20	72.10
		100	78.10	72.00
	Root	30	79.48	**74.20**
		50	**80.27**	73.73
		100	78.10	72.97
Dependency Parser + Recurrent NN	Gold aspect label	30	78.30	72.90
		50	78.57	72.85
		100	77.90	72.48
	Root	30	79.90	**76.15**
		50	**80.90**	75.75
		100	79.82	73.70

robust, representative models.

When performing a fine-grained analysis, we observe that dependency parsers produce more successful results as compared to constituency parsers. The main intuition behind it is that the modifier relationship can be directly captured in dependency parser module unlike in constituency parser module. These modifier relations are useful in both generating sub-reviews and identifying sentiment expressions made towards aspects. The root scheme is observed to help arrive at the best results. The reason is that the model is mostly affected by the root nodes in terms of both sentiments and syntax. That is, these nodes have an impact on a wide scope. For the dependency

parser module, exploiting the gold aspect labels in addition to sentiment lexicons leads to inconsistency and it causes the performance to decrease. That is using the labels present in both the training data and sentiment lexicon, which can be different for the same words, can lead to contradiction and ambiguity. Lastly, vector lengths of 30 and 50 lead to the best performances for the recursive neural network models, whereas the size 100 gives rise to overfitting.

Additionally, the results for two different scenarios, which are a single aspect (SA) or multiple aspects (MA) modules, are listed and a comparison between our approach and the baseline is made in Table 4.14. Here, only the best accuracies (dependency parser module with the *"root"* setting) for our approach are shown. SA is the case where there exists only one aspect in the review, whereas MA refers to the case where there is more than one aspect in the review. Each review in the dataset has either an SA or MA. That is, every review has at least one aspect. We demonstrate the impact of the use of the MA scenario on the performance, modelling the interplay of sentiments between different aspect terms. We arrived at better performances for the MA module. We show that we outperformed the baseline method for all cases. We performed the Stuart-Maxwell test and observed that the results we obtained are significant $p = 0.05$.

Table 4.14. Accuracies (%) that are obtained in the baseline method (IARM) and our ensemble framework. SA stands for single aspect scenario, MA for multiple aspect scenario.

Approach	**Restaurant**		**Laptop**	
	SA	MA	SA	MA
Baseline (IARM)	78.6	80.48	73.4	74.1
Our ensemble framework	**79.5**	**81.38**	**75.75**	**76.45**

In summary, our incorporating a recursive neural network model into a recurrent model and forming an ensemble framework boosts the performance for ABSA. We enrich the recurrent model encoding temporal information with a recursive model capturing grammatical and sentiment information. We originally generate sub-reviews

from whole reviews and train them in the recursive model. When the mentioned two neural networks are merged, a more robust representation is modelled, thereby leading to better accuracies.

4.4.3. Conclusion and Future Work

We have developed an ensemble approach to performing ternary ABSA. Our framework is composed of a recurrent and recursive sub-model. We first extract sub-reviews from reviews in a novel way. Each of these sub-reviews encompasses only one sentiment expressed towards the aspects covered therein. A single sub-sentence parse tree may include one aspect or more. We include only the relevant sentiment, ignoring other opinions expressed towards other aspects in the same review. Constituency and dependency parser modules are utilised when training our recursive neural models. In the end, the root vectors of these parse trees are fed as input into a GRU network, which is a recurrent model.

It is observed that combining recursive and recurrent neural networks provides us with an enriched and a more robust model. In the recurrent model, we capture the temporal information, whereas in the recursive model, we can extract inherent grammatical and sentimental features from the texts. That is, both these compensate for what the other module lacks. Our experimental results show that we significantly outperformed the baseline approach for two datasets. When using the dependency parser, we achieve a better performance as compared to the use of constituency parser. The intuition behind it is that dependency parsers can directly capture modifier relationships. These modifiers indicate which sentiment is expressed towards which aspect or noun. We conjecture that our ensemble classifier model can be applied to other NLP classification tasks or computational linguistics fields.

As future work as post-doctoral research, we plan to extend this approach by (1) incorporating a CNN framework to enrich the system with the contextual representation of aspect terms, (2) enhancing the constituency parser module to effectively

capture the neutral opinions as well, (3) relying on other sentiment lexicons induced by semi-supervised approaches as mentioned or use other supervised techniques to better model the polarities of words in the trees, (4) training both the recurrent and recursive neural network models at the same time to arrive at more successful results, and (5) using and adapting our ensemble approach to other languages. In addition to these, we also plan to (6) compare our approach to works based on RST and hybrid solutions that incorporate concept-based reasoning into ontology-based reasoning using deep learning models and (7) evaluate our methods on other corpora, such as the SemEval-2015 corpora, where reviews can consist of more than one sentence.

We also plan to (6) evaluate our methods on other datasets with different categories (e.g. the SemEval-2015 corpora) where reviews may consist of more than one sentence and (7) compare our performances to those of RST studies and the hybrid solutions which combine concept- and ontology-based reasoning with deep learning models.

4.5. Major Contribution #4: Generating Word and Document Embeddings for Sentiment Analysis

Polarities of words can differ based on the corpus. Generating common sentiment lexicons for a language and making use of them cannot lead successful results to be obtained for corpora of different genres. In this approach, we combine the lexical, contextual, and supervised characteristics of the text words. In this regard, the semantic definitions of words appearing in the dictionary are leveraged. The tokens surrounding a target word represent domain-specific information and supervised scores of words capture their sentiments. We observed that combining labelled information of words (i.e. positive or negative) with lexical features generated from dictionaries boosts the performance. We perform an ablative combination of contextual, dictionary-based and supervised feature sets, and create novel embeddings. Additionally, we incorporate hand-crafted features into the word2vec embeddings learnt from scratch. We generate domain-specific sentiment embeddings for two datasets that are the movie and the

Twitter corpora in Turkish. When we create document embeddings and feed them as input into the SVM method, our approaches outperform the baseline studies for Turkish by a significant margin. These original models are evaluated on two English datasets of different genres as well. These also perform better than the word2vec approach. This empirically shows that our methods are cross-domain and portable to other languages [87].

Neural network models in general perform better than the classical machine learning algorithms for most classification and regression tasks, including opinion mining [88], when fed with bulky datasets as input. In these models, dense word embeddings are employed to combat the data sparsity issue. These provide us with more *"meaningful"* and robust representations. These vectors indicate how similar and close the words are to one another in the VSM.

As mentioned, in the literature, most of the studies rely on vectors, such as word2vec [8], in which only the syntactic and semantic representations of the words are taken into account. Ignoring the sentimental aspects of words can cause the tokens with opposite sentiments to be closer to each other in the space model, provided that their semantic and syntactic features are similar.

In the Turkish language, there exist only a few studies that use sentiment information when inducing word and document vectors. In contrast to the studies conducted for English and other commonly-used languages, we employ the official Turkish dictionary in this thesis, and incorporate both supervised and unsupervised features into the model at the same time so as to produce a unified score per dimension of each word embeddings.

The main contribution of this approach is generating novel and effective word embeddings which capture semantic, syntactic and sentimental characteristics of tokens, and exploit all this information when creating word vectors. We rely on the word2vec embeddings trained on our corpora as the baseline approach as well. Apart from util-

ising those vectors, we additionally build a hand-crafted feature set on a review-basis and generate document embeddings. We test and evaluate these vectors on several corpora of different genres. We will empirically show that we perform better than the approaches which do not leverage the sentiment knowledge. We outperform the other works carried out on the sentiment classification task in Turkish media as well. We evaluated our original vector models on two different datasets in English as well in the end. We also outperformed the baseline models for the English language.

4.5.1. Methods Developed

We create various word embeddings that can represent the contextual, lexical, semantic, and sentimental characteristics of words. Additionally, as a baseline, we also induce word2vec vectors of the corpus words by training this vector model on these corpora. After producing those dense vectors, we also incorporate hand-crafted features into review (document) embedding representations and conduct binary classification, as we will explain and discuss later.

4.5.1.1. Corpus-Based Approach. Contexts are informative in the sense that similar words in general occur in the same contextual windows. For instance, the word *"smart"* is more probably to co-occur with the word *"hard-working"* than with the word *"lazy"*. The similarity may be represented semantically, syntactically, and sentimentally. In this sub-approach, we capture all these characteristics and create vectors for the corpus tokens.

First, we build a matrix whose entries correspond to the number of co-occurrences of the row and column words with regard to sliding context windows. In this matrix, diagonal entries are the numbers of sliding context windows that the corresponding row word occurs in all the reviews in the dataset. Then, we perform normalisation by dividing scores in each row by the maximal score in the same row.

Second, we use the principal component analysis (PCA) method to decrease the number of dimensions. It extracts latent meanings from data and relies on high-order co-occurrence statistics after performing noise removal. We set the reduced column (attribute) number of the matrix as 200. We thereafter calculate the cosine similarity score between each row pair w_i and w_j as shown in Eq. (4.9) to detect the similarity of two different word (row) embeddings.

$$cos(w_i, w_j) = \frac{w_i^\intercal w_j}{||w_i||||w_j||} \tag{4.9}$$

Third, all the scores in the matrix are subtracted from 1 so that we have a dissimilarity matrix. We then use this matrix as an input in the fuzzy c-means clustering algorithm. The number of clusters is chosen to be 200, as this is stated to be one of the standard values for word embeddings in the literature. After having performed clustering, the dimension i for a corresponding token is indicative of the degree to which this token belongs to the cluster i. The main intuition behind this method is that if two tokens are close to each other in the VSM, these tend to appear in the same or similar clusters with similar probabilities. Lastly, each word in the dataset is represented by a 200-dimensional embedding.

In addition to this method, we rely on the SVD method taking the co-occurrence matrices as input, where we compute the matrix $\boldsymbol{M}^{PPMI} = \boldsymbol{U}\boldsymbol{\Sigma}V^T$. Here, the $\boldsymbol{U}$ is an $m \times m$ real or complex unitary matrix, $\boldsymbol{\Sigma}$ is an $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, and $\boldsymbol{V^T}$ is an $n \times n$ real or complex unitary matrix. Positive pointwise mutual information (PPMI) score between each word is computed and the truncated SVD is performed. We take into consideration only the $\boldsymbol{U}$ matrix per word. We have again set the singular value number at 200. That is, every corpus word is modelled by a 200-dimensional embedding, as shown below:

$$\boldsymbol{w}_i = (\boldsymbol{U})_i \tag{4.10}$$

4.5.1.2. Dictionary-Based Approach. In Turkish, there are not well-established sentiment lexicons as in English. In this sub-approach, the official Turkish dictionary (TDK) is employed to compute word polarities [126–129]. Although this dictionary does not include polarities specific to words, when we combine it with domain-specific sentiment scores extracted from the labelled dataset, we achieve state-of-the-art performances.

First, we build a matrix whose row entries correspond to words in the dataset and column entries are the words appearing in their dictionary definitions. Boolean approach is preferred in this model. For example, for the row word *"cat"*, the entries corresponding to the column words explicitly appearing in its dictionary definition are each assigned a score of 1. The matrix entries corresponding to those column words not occurring in the dictionary definition of *"cat"* have a score of 0 in the same row.

When we perform clustering on the matrix, we observe that words with similar meanings are grouped in the same or nearby clusters. Nevertheless, sentiment information cannot be effectively captured in this model. For example, the words *"happy"* and *"unhappy"* are grouped in the same cluster, for they have common words in their dictionary definitions, such as *"feeling"*. On the contrary, words of different polarities should actually be located far away from each other in the VSM.

We therefore rely on an approach to move such words as far away from one another as possible in this space model, even if there are several common words in their lexical definitions. Each score in a row is multiplied by the raw sentiment score of the word in the same row. Hence, we obtain more robust clusters. Only exploiting the training set, the sentiment score for each word is computed as shown in Eq. (4.11). In fact, this formula is the same as Eq. (4.6) except that only the smoothing value chosen for this case is different.

$$w_t = \log \frac{\frac{N_t}{N} + 0.01}{\frac{N'_t}{N'} + 0.01} \tag{4.11}$$

Here, w_t is the polarity score of the word t, N_t is the total number of reviews/tweets in which t appears in the corpus containing positive reviews. N is the vocabulary size of the positive dataset. N'_t and N' denote the corresponding values for the corpus of negative sentiment. We normalise the values to handle the imbalance problem by adding a small number to both numerator and denominator for smoothing.

On the other hand, apart from the multiplication of the raw supervised sentiment values with unsupervised scores, we additionally multiply all row scores with only +1 if the corresponding row word expresses a positive sentiment or with -1 if this expresses a negative sentiment. It is observed that we thereby achieve better performance.

The impact of this multiplicative process is visualised in Figure 4.14, visualising the positions of word embeddings in the VSM. These *"x"* words are of negative polarity, whereas *"o"* words are of positive polarity. In the upper sub-plot, the words with different sentiments are close to each other, due to their having common dictionary definition words. Only the information in the dictionary definitions is relied on there, ignoring the supervised scores. Nevertheless, when employing the supervised score (+1 or -1), tokens with different sentiments (e.g. *"happy"* and *"unhappy"*) move far away from one another as they are translated across different coordinate regions. Now, words with a positive sentiment are in quadrant 1 and words with a negative sentiment are in quadrant 3. Thus, words with similar sentiment scores get closer in the VSM and could be more accurately grouped in the same or close clusters in the VSM. Apart from clustering, we used the SVD algorithm as well to conduct dimensionality reduction for the fully unsupervised dictionary approach and made use of the newly created matrix by performing ablative combinations with other sub-approaches. The number of dimensions is again set at 200 with regard to the $\boldsymbol{U}$ matrix. These are elaborated in the later sections. When evaluating this sub-approach on the datasets in English, we

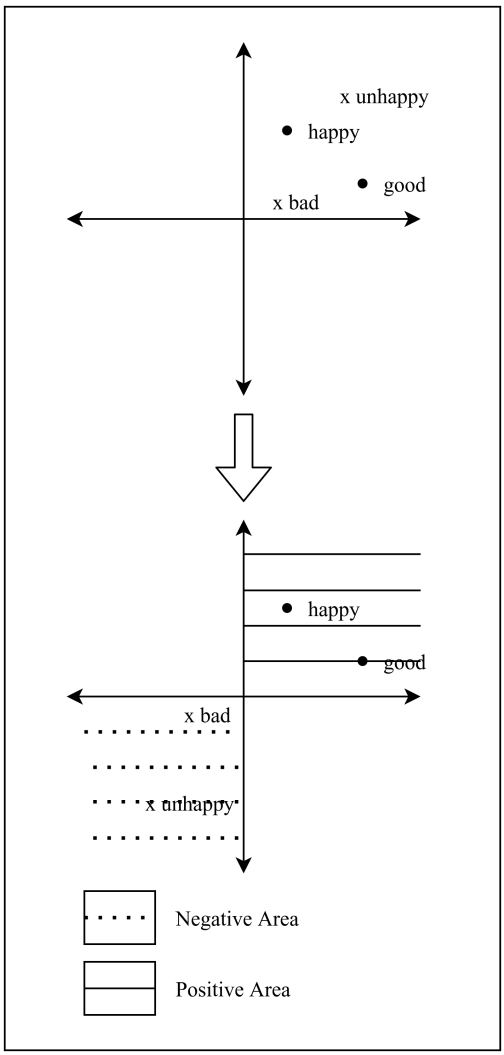

Figure 4.14. The impact of leveraging the labels of tokens as well in the dictionary-based approach. This figure visualises how words with the opposite polarities get far away from each other in the VSM.

relied on the SentiWordNet lexicon [20]. When preferring the SVD reduction approach over the use of clustering method, we achieve better performances.

4.5.1.3. Supervised Contextual 4-scores. The last module we used is a basic scheme which extracts four supervised scores per token in the corpora. These values are generated as follows. For each target token in the dataset, we scan through all of its context windows. In addition to the sentiment value of a target word (which we name "the self-score"), maximum, minimum, and averaged scores are used out of all the sentiment

scores of words appearing in the same context windows as the target token. We calculated the word polarity scores by using Eq. (4.11). These scores are obtained by relying on the training set.

In this way, those four scores per word are taken into account, which is more informative compared to the use of a single self-polarity score. However, this metric is entirely supervised in contrast to the previous two sub-modules.

4.5.1.4. Combination of Word Vectors. Apart from employing these three sub-modules disjointly, all the matrices generated in the previous approaches are also combined. In other words, we perform concatenation of the reduced forms (SVD - U) of dictionary-based, corpus-based, and the 4-score vectors of each word column-wise. Accordingly, each token is modelled by a 404-dimensional embedding, for dictionary-based corpus-based embedding components have each 200 dimensions, whereas the 4-score embedding component is formed of four values, whereby the sum is 404.

The intuition behind this ensemble approach is that some sub-methods compensate for what the others lack. For instance, the corpus-based approach can capture the domain-specific, syntactic, and semantic characteristics. On the other hand, the 4-scores module models label information and the dictionary-based approach helps capture the general lexical and semantic information. When we combined these three approaches, we have produced more representative and robust word models.

4.5.1.5. Creating Document Embeddings. After generating various vector types as mentioned, we generate document embeddings. In this respect, documents can be reviews or tweets. Per document, we average all the embeddings of those words appearing therein. Additionally, we define a set of hand-crafted features, which are maximal, minimal, and mean polarity scores of words appearing in a document. Those sentiment values are calculated as shown in (4.11). That is, these values are generated on a review-basis, not on a word-basis. For instance, if a document is composed of six

words, this would have six polarity scores and only three of these polarity values are employed as explained. In the end, we concatenate the averaged document vectors to these three sentiment scores.

Thus, each document is modelled by the averaged word embedding and three additional hand-crafted sentiment scores. We thereafter use these vectors in an SVM classifier to predict the polarity of a review. We give the flowchart of the proposed approach in Figure 4.15. When we combine the embeddings generated by word-based features with the overall three sentiment scores created on a review-basis, better-performing and state-of-the-art results are obtained by us.

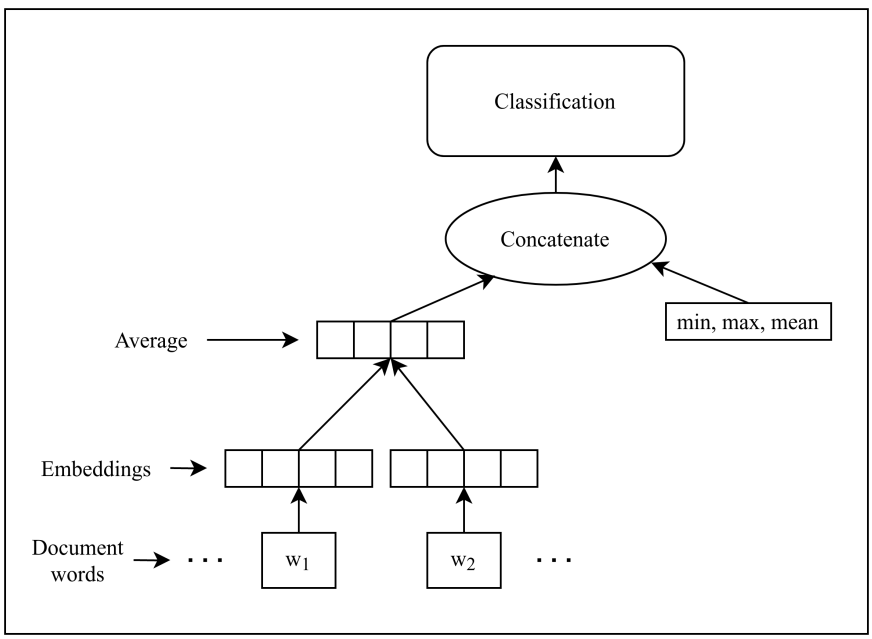

Figure 4.15. The flowchart of our system.

4.5.2. Corpora

We relied on two corpora of different genres in both Turkish and English when conducting experimentations and evaluating our approaches.

As the first Turkish dataset, we employed the same movie corpus consisting of 20,244 reviews that are collected from a popular website, as mentioned in Sec-

tion 4.2.4.1. We have randomly chosen 7,020 negative and 7,020 positive reviews, formed a balanced dataset, and processed only them. The second corpus for Turkish is the same Twitter dataset consisting of 1,716 tweets, which we explained in the preceding subsections.

We also relied on two other datasets which are in English to test the portability of our approaches to other languages. Among these, the first is a balanced movie dataset [115] downloaded from a website [112]. In total, there are 5,331 negative reviews and 5,331 positive reviews in this corpus, as discussed previously. The second dataset is a Twitter corpus that consists of approximately 1.6 million tweets which are labelled through a distant supervised algorithm [116]. These tweets are assigned negative, positive, and neutral polarities. We chose 7,020 positive tweets and 7,020 negative tweets in a random manner to create a balanced corpus for the binary classification task.

4.5.3. Experiments

4.5.3.1. Preprocessing. We performed similar preprocessing techniques to those given and discussed in Section 4.1 for this approach as well. We explain these as follows. The Turkish speaking people can sometimes spell English characters for the corresponding Turkish characters (e.g. *"i"* for *"ı"*, *"c"* for *"ç"*) when typing messages in an electronic format. So as to perform normalisation on these tokens, the Zemberek tool [99] is employed. Punctuation marks except only *"!"* and *"?"* are filtered out, since these do not have an effect on the overall sentiment of a review/tweet. In this approach, emoticons (e.g. *":((("*), and idioms (e.g.*"kafayı yemek"* (lose one's mind)), have not been removed, since two or more words can bear a sentiment together when forming an idiom, no matter what polarity its constituent words have. Due to the morphologically-rich structure of the Turkish language, the morphological parser and disambiguation tools [102, 103] are used. We additionally performed stop word removal and negation handling. When handling negator words/morphemes, an underscore is added to the end of a word in case negation occurs. For instance, *"sıkıcı değil"* (not boring) is redefined

as *"sıkıcı_"* (sıkıcı_) in the feature selection stage before generating supervised scores.

4.5.3.2. Hyper-parameters. We employed the LibSVM utility in the WEKA tool [130]. We relied on the linear kernel for the binary classification task. We generated word2vec vectors by training them on all the four datasets making use of the Gensim library [131] with the skip-gram method as the baseline. The length of these embeddings is set as 200 for a consistent comparison. As mentioned, other vectors, generated by relying on the clustering and the SVD modules, are of length 200 as well. In the c-means clustering sub-method, the maximal number of iterations is set at 25. If the model converges before reaching that number, we stop the training earlier.

4.5.3.3. Results. As mentioned, we evaluated all our approaches on four corpora that are the movie and Twitter datasets in Turkish and English. We have learnt all vectors from scratch for these datasets. As the performance criterion, the accuracy metric is preferred, since all the corpora are completely or almost balanced. 10-fold cross-validation is performed for all the corpora. We employed the approximate randomisation method to conduct the significance tests. In this approach, we try to predict the labels of reviews. That is, we evaluate and assess our models on a review-basis, not on a word-basis.

The results we obtained are shown in Table 4.15. The *"3-feats"* metric denotes the hand-crafted feature set, which are the maximum, minimum, and mean polarity scores of documents as previously described. As can be seen, at least one of our approaches performs better the baseline approach (word2vec) for all the English and Turkish datasets of different categories. As can be expected, all of our approaches also perform better when we exploit the sentimental characteristics that are extracted on a review-basis and when we concatenated them to word embeddings. In most cases, the supervised 4-scores metric produces the best-performing results, since it makes use of the annotation knowledge based on the sentiments of words.

Table 4.15. Performances (%) for different features used as input in the SVM model when detecting the polarities of documents. The baseline approach is chosen as the word2vec algorithm.

Word vector type	**Turkish (%)**		**English (%)**	
	Movie	**Twitter**	**Movie**	**Twitter**
Corpus-based + SVD (U)	76.19	64.38	66.54	**87.17**
Dictionary-based + SVD (U)	60.64	51.36	55.29	60.00
Supervised 4-scores	**89.38**	**76.00**	**75.65**	72.62
Concatenation of the above three	88.12	73.23	73.40	73.12
Corpus-based + Clustering	52.27	52.73	51.02	54.40
word2vec	76.47	46.57	57.73	62.60
Corpus-based + SVD (U) + 3-feats	88.45	72.60	76.85	**85.88**
Dictionary-based + SVD (U) + 3-feats	88.64	71.91	76.66	80.40
Supervised 4-scores + 3-feats	**90.38**	**78.00**	**77.05**	72.83
Concatenation of the above three + 3-feats	89.77	72.60	77.03	80.20
Corpus-based + Clustering + 3-feats	87.89	71.91	75.02	74.40
word2vec+ 3-feats	88.88	71.23	77.03	75.64

As shown in the table, the clustering approach generally leads to the lowest performance. We observed that the corpus - SVD module always performs better as compared to the clustering method. We think the reason is that, in SVD, we take into consideration the most important and informative singular values. The corpus - SVD method performs better than the word2vec model for several corpora. When the 3-feats metric is not employed, the corpus-based SVD approach produces the best-performing accuracies for the Twitter corpus in English. This has also empirically been shown that the use of basic and simple models can be a better and more successful choice compared to that of complex models. For instance, our statistical corpus-based approach outperforms the word2vec neural network model. We also found out that the accuracy decreases in several cases when we make use of the sentiment information, as in the case for the English tweets.

Since the official Turkish dictionary covers almost all the movie corpus words, the dictionary approach produces state-of-the-art results. On the other hand, the dictionary lacks many words, which appear in short and noisy tweets. Hence, its performance is not the best among all results. However, as can be expected, when we combine the dictionary method with the 3-feats metric, we observe a great improvement.

The accuracies produced for the movie corpora are much higher than those for the Twitter datasets for almost all of our methods. The reason is that the texts in the Twitter datasets are usually much noisier and shorter. We observed that when setting the p value as 0.05, our results are statistically significant as compared to the baseline approach for Turkish [84]. Our several approaches produce better success rates than those sentiment analysis models relied on for English [34, 86] as well. That is, our results are both significant and state-of-the-art for both these languages. As said, when we additionally leveraged three hand-crafted sentiment features, we arrived at the best results.

A CNN model is also used for this approach. Nonetheless, the SVM method that is a classical machine learning model performed as effectively. Therefore, here, we cover only the best results, ignoring those obtained for the CNN approach.

As a qualitative assessment of word representations, we have listed the most similar words to target words by exploiting the cosine similarity metric. We have performed a limited analysis to find out which words are the most similar with regard to different modules. In Table 4.16, we show the most akin words to the given query words. Tokens of the same polarity are observed to be the most similar and closest in the VSM. For instance, the most similar word to *"muhteşem"* (gorgeous) is *"10/10"*, both of which are of positive polarity. As shown in the table, our corpus-based approach is better at capturing domain-specific characteristics and features as compared to the word2vec model, which can generally model syntactic and semantic characteristics, however, not the sentiment information.

Table 4.16. Most similar words to given queries with regard to the corpus-based approach and the baseline algorithm (word2vec).

Query Word	**Corpus-based**	**word2vec**
Muhteşem (*Gorgeous*)	10/10	Harika (*Wonderful*)
Berbat (*Terrible*)	Vasat (*Mediocre*)	Kötü (*Bad*)
Fark (*Difference*)	İlginç (*Interesting*)	Tespit (*Finding*)
Kötü (*Bad*)	Sıkıcı (*Boring*)	İyi (*Good*)
İyi (*Good*)	Güzel (*Beautiful*)	Kötü (*Bad*)
Senaryo (*Script*)	Kurgu (*Plot*)	Kurgu (*Plot*)

4.5.4. Conclusion

In this approach, we have demonstrated that employing word embeddings which can capture only syntactic and semantic features and characteristics can be enhanced by incorporating sentimental aspects as well into the model. Our approach is cross-domain and portable to other languages. We conjecture that they can be applied to other datasets of different genres and also to other languages apart from Turkish and English, when minor changes are performed.

This approach is one of the few studies performing opinion mining for the Turkish language and exploiting sentiment characteristics of words when creating word embeddings. This approach outperforms all the other methods. Any of the sub-modules we proposed in this approach can be used disjointly and independently of the others. Our approach can be applied to other classification fields, such as concept extraction and topic classification, when supervised labels are adapted to new categories.

We have empirically shown that even methods based on unsupervised models, such as the corpus-based approach, may perform better than supervised models in classification tasks. Combining several approaches, each of which may compensate for what others lack, can lead to the generation of more robust word representations. Our

word embeddings are built and assessed by feeding them into conventional machine learning algorithms. However, these produce state-of-the-art results. Although we used a conventional machine learning algorithm (SVM) over a neural network classifier in order to identify the polarities of documents, we have obtained accuracies of over 90% for the Turkish movie dataset and an accuracy of about 88% for the English Twitter dataset at best.

In this approach, we have only performed binary sentiment analysis as in the literature. We plan to (1) expand our framework in future by also taking neutral reviews into consideration to test its generalisability to all sentiment categories. We would also (2) employ the Turkish WordNet to enhance the robustness of the word and document vectors.

4.6. Major Contribution #5: Redefining Context Windows as Subclauses for Word Vector Models

In this approach, we propose an approach, where a word context is defined in terms of a subclause rather than a sliding window. We generate subclauses by using a dependency parser. The rule set we defined for generating subclauses/sub-sentences is novel. As a sentence is broken down into subclauses, these are chosen as the context windows for the target words in the sentence. We thereby take into account linguistic, syntactic, and even sentimental patterns as opposed to sliding windows, which cannot capture these as robustly and effectively. In this way, we employ more semantically-oriented, variable-size contexts as compared to fixed-size relative position-based contexts. To evaluate the effectiveness of the proposed approach, we generate word embeddings by redefining the context windows for the GloVe and SVD models and perform binary classification by using the document vectors. We applied this approach only on English datasets. However, this can be adopted for other languages with minor changes as well, if a dependency parser library exists for the target language.

4.6.1. The Proposed Approach

In this approach, we change the local context window definition for both the GloVe and SVD models, and redefine these contexts as formed of subclauses. In this regard, we parse a sentence into subclauses as in [49]. As previously mentioned in the ensemble neural network model, subclauses are also identified using the same rule set for this approach. We explain this dependency-based subclause generation process as follows (albeit a bit repetitively). We first generate dependency tree of sentences using the spaCy library. Then we scan the nodes in these trees starting from the root. We iterate recursively every child node of words. If the POS tag of a word is verb and this word is not modified by a clausal component (i.e. ccomp) or a conjunction relationship (i.e. conj), we recursively add its children into the same group marked as a subclause. Otherwise, we cut the dependency tree at these points and start forming different subclauses. After generating these subclauses, if any, we remove the preceding and trailing conjunctions. We also add the punctuation mark of the sentence to the end of each subclause. An example was shown in Figure 4.12 that is relevant to this process. In that sample review, the sentence is broken down into three subclauses. Each subclause (subtree) is considered a separate context and is used as the context window for every word within the subtree when generating the embedding of that target word. Even those neighbouring words in the sentence could be considered semantically far away from each other, if they are in different subclauses. That is, those words in the same subclauses are more related to each other and this information is useful when generating word embedding models.

For the GloVe algorithm, we consider the context of a word as formed of the subclause in which the word occurs. For the SVD algorithm, the co-occurrence matrix is built by considering the subclauses separately. In other words, two words that co-occur in the same subclause add one to their co-occurrence count. After the co-occurrence matrix is factorised as $\boldsymbol{U}\boldsymbol{\Sigma}V$, we take the $\boldsymbol{U}$ matrix only, since it is reported to yield the best performance [1]. In this matrix, each row corresponds to a word's representation.

After forming context windows as subclauses for both of those word embedding models and generating word vectors, we evaluate the effectiveness of the embeddings on two document-based classification tasks. We represent a document as the average of the embedding vectors of the words in the document. These document vectors are then fed as input into an SVM classifier.

4.6.2. Experimental setup

In the literature, lexical similarity and analogy tasks are heavily leveraged for assessing the quality of word representations [132]. In this approach, we follow a different track and perform extrinsic evaluation on two tasks to measure the quality of the word embeddings based on subclausal contexts. Table 4.17 lists the hyper-parameters used for the Glove and SVD models. Context window type can be regarded as a hyper-parameter, where sliding window refers to the baseline approach for both the GloVe and SVD methods, and subclause to the proposed approach. Subclause generation value indicates what relationships in the dependency trees are used to create subclauses. Sliding window size and sliding window position are related to only the baseline approach. We did not take into account the left-side contexts alone, since, in a baseline study [97], the best-performing practices are reported to be with symmetric and right-side windows. However, this can differ depending on the linguistics properties of the language of corpus. The parameters which are word embedding size and stop word removal are applicable for both the baseline and the proposed approaches in this thesis.

We train the GloVe and SVD models with 100- and 300-dimensional vector settings separately. We use two versions of each model, one with all the words included and the other with stop words removed. The epoch number for the GloVe algorithm is set as 10.

Table 4.17. Possible values chosen as hyper-parameters for the experiments.

Hyper-parameter	**Possible values**
Context window types	{subclauses, sliding windows}
Sliding window size	{5, 10}
Sliding window position	{right, symmetric}
Subclause generation	{conj, conj+ccomp}
Word embedding size	{100, 300}
Stop word removal	{yes, no}

4.6.3. Datasets

We evaluated the performance of our models on two datasets corresponding to different classification tasks after training the word vectors on these corpora. The first dataset is a movie corpus labelled for binary (positive and negative) sentiment classification [115]. This is the same balanced dataset consisting of 10,662 reviews, as explained in Section 4.2.4.1. We split the whole dataset as 80% for training (8,530 reviews and 179,674 tokens) and 20% for testing (2,132 reviews and 44,391 tokens). In addition to the sentiment analysis problem, we also evaluated this framework on a dataset that belongs to another NLP classification task to test its portability across various domains. This second one is a nearly balanced dataset used for the spam e-mail binary classification problem [133]. The training set consists of 5,729 e-mails (797,892 tokens) and the test set is composed of 1,708 e-mails (567,903 tokens). We use the spaCy library to tokenise words. We use the same library to generate dependency parse trees and to obtain the subclauses for each review/e-mail. We perform lemmatisation and filter out words whose frequency is lower than four.

4.6.4. Results

In the experiments, we compare the performance of the proposed approach that makes use of subclauses as contexts with that of the baseline approach. Additionally, we perform an ablation study to assess the impact of each hyper-parameter on the

performance. In this regard, we observe the success of the proposed approach with respect to different scenarios. All the experiments were performed in Python and C++ on an Ubuntu Linux-based laptop, with the Intel i5 processor and a 4 GB of RAM. The training phase lasted for 32 minutes and 5 minutes on the average for the spam and sentiment tasks, respectively. The execution of the subclause extraction module took about one and a half times as long, since it takes time for the dependency parser to generate all relationships. We show the results in Table 4.18 in terms of accuracy.

Table 4.18. Accuracy (%) across all models and metrics with various hyper-parameters trained on different corpora.

Word vector	**Domain**	N	**Subclauses**		**Sliding windows**			
					$k = 5$		$k = 10$	
			conj + ccomp	conj	right	sym	right	sym
GloVe	Sentiment	100	59.56	**60.22**	58.77	59.05	58.44	59.89
		300	**62.42**	61.58	58.02	60.41	58.11	61.39
	Spam	100	**86.99**	86.42	85.70	80.84	82.19	81.83
		300	89.45	**90.27**	83.47	86.93	81.89	81.66
SVD - U	Sentiment	100	**69.51**	69.44	65.66	66.55	67.16	67.87
		300	70.45	**74.38**	68.99	68.80	69.60	70.35
	Spam	100	95.60	96.12	95.54	93.67	96.25	**96.48**
		300	**97.24**	97.18	96.77	95.84	96.89	97.12

In the table, the value N stands for the embedding size. The value k denotes the window length. *sym* stands for symmetric context windows. *conj* indicates that subclauses are generated by cutting the sentence dependency tree based on only the conjunction relationship. *conj + ccomp* denotes the same scenario based on both the conjunction and clausal component relationships. The above experiments correspond to the scenarios, where stop word removal is not performed. As can be seen, our approach (defining subclauses as context windows) outperforms most of the baseline sliding windows modules. We performed McNemar's significance test and found out that our results are significant ($p < 0.10$) except only one case where we could not outperform the baseline (SVD, Spam detection, $N = 100$). We conjecture that our

approaches can perform significantly well on larger corpora as well, which should be verified by future research. The results of the experiments and evaluations are detailed below for each hyper-parameter.

4.6.4.1. Context Window Type. The proposed sub-approach where context windows are formed of subclauses outperforms the baseline sliding windows method in almost all cases. This shows that subclauses can semantically and syntactically capture the relationships between the words therein more robustly and successfully as compared to fixed-size windows for word vector models. We observed that, in some cases, also taking into account the *"comp"* relationship in addition to the *"conj"* relationship when splitting the trees into subclauses can boost the performance or vice versa. We attribute it to the following factors. When generating narrower, that is, more specific and separate contexts for both these associations, we focus on a different aspect of the sentence. Therefore, those words in these narrower contexts can be more strictly related to each other. On the other hand, the *"comp"* relationship (e.g. when immediately followed by *"which"*) can specify an attribute/event about the preceding words by modifying them in broader contexts. Hence, these can also be considered related to each other and redefining them as a whole subclause can be considered a better approach in a way.

4.6.4.2. Window Size and Orientation. These hyper-parameters are related to only the baseline context window method. In regard to the sizes of sliding windows, the window size of 10 performs better than the size of 5 as can be expected. The reason might be that a wider context of words can extract lexically and semantically more informative analogies and relationships between words. For the window position parameter, we observed that using context from both sides outperforms the case where only the right context is used. This can be attributed to the fact that words around a target word in both relative positions carry more relevant information about the target word than the words on one side only. Making use of all these words can therefore model the vector representation of a target word more successfully.

4.6.4.3. Word Embedding Size. A larger size of a vector can encode more information using distributional semantics. In parallel with this, we observed an increase in the performance when using 300-dimensional embeddings as compared to 100-dimensional vectors.

4.6.4.4. Stop Word Removal. When we filtered out stop words from the corpora, we observed a decrease in the performance. This is the opposite of the results given in the baseline paper [97], where removing stop words was reported to boost the performance. We attribute the reason to two factors. The first is that we employ a dependency parser and the parser needs to take into account all the words during parsing. Therefore, every token has a functionality in generating a consistent parse and subclauses. The second is that our corpora are much smaller in size compared to those used in the baseline paper. Hence, stop words add to the sizes of the datasets and are likely to contribute to the informative relationships between words. However, we do not include the effect of stop word removal in the table. Instead, we show the results, where stop words are not filtered out, since this is the best-performing scenario.

4.6.5. Conclusion

In this approach, we showed that using subclauses as context windows in word vector models outperforms the sliding window approach. Relying on linguistic patterns and capturing semantic, syntactic, and sentimental relationships between words within subclauses provides us with a more robust word representation. Our approach is cross-domain and cross-model.

As future work, we plan to extend this study to non-English corpora, to the word2vec algorithm, and to other machine learning models as the context windows used in convolutional neural networks and other similar architectures. We also plan to perform error analysis, conduct lexical similarity tasks, use rhetorical structure theory [134] in lieu of parsers, and leverage larger corpora to generate more powerful representations.

4.7. Minor Contributions

4.7.1. Minor Contribution #1: Semi-Supervised Approach with Tweaked Parameters

As described and discussed in Section 4.2.1.2, we employed the semi-supervised approach for Turkish and English datasets similarly as in [1]. Accordingly, we propagate the polarities across the graph built by relying on co-occurrence statistics. However, as mentioned, we changed the formula such that we take into account factorial values in matrix and vector processes instead of the exponential or a linearly increasing function. Since this has been discussed in a previous section, we do not delve into its details again here. As said, this increased the success rates significantly. Nonetheless, since we have not made a significant contribution besides changing the formula parameters for the semi-supervised approach, we consider it to be a minor contribution.

4.7.2. Minor Contribution #2: Aspect Term Extraction for English

Another minor contribution made by us is a method that extracts aspects from English corpora employing a supervised technique. We rely on SemEval datasets, in which aspects are manually annotated and are provided in the training set. Approach we developed is as follows. First, we perform feature extraction by defining a set of rules. We extract nouns and noun phrases, and give more importance to these words, because mostly nouns can be a candidate aspect. Among other features are GloVe vectors, POS tag vectors of words, and dependency relationship vectors. That is, for each noun or noun phrase word(s), we take into account these embeddings and concatenate them, and finally feed these vectors as input into an SVM classifier, which predicts a noun/noun phrase as an aspect or a non-aspect word. If this is a noun phrase, we average all the corresponding vectors therein. GloVe vectors are indicative of a word's semantic properties and aspects can generally be considered to have similar dense representations. POS tag vectors are also important, since, as said, words that are noun or noun-like tokens are more likely to be an aspect. Lastly, dependency

relationships model the scenario, wherein aspects are more likely to be modified by specific grammatical and syntactic associations. For example, adjectives and verbs in general modify candidate aspects, as in *"I loved its sound quality!"* and *"This smartphone has a good resolution."* However, these features alone are not very novel besides concatenating them all. On the other hand, we have not achieved significantly high success rates as compared to the baseline method. Therefore, we do not include the experimental results related to this approach in this thesis book.

As future work, we plan to define a broader set of features and feed them as input into a neural network architecture, which is a more effective model than the SVM classifier for most cases. Also, we would employ dense POS tag vectors in lieu of the Boolean POS tag embeddings, since the latter cannot capture the part-of-speech similarities and properties as effectively. For instance, a gerund verb and an infinitive verb have similar representations according to the dense model, whereas they are completely different when the Boolean metric is relied on.

4.7.3. Minor Contribution #3: Aspect-Based Sentiment Classification for Turkish

In this approach, we perform aspect-based analysis on Turkish raw text data. We first extract nouns using the parsing and disambiguation tools, and consider them all to be aspects. Then, we employ a dependency parser developed for Turkish [100] and if we detect that an adjective or a verb modifies them, we assign the sentiment of that word to that candidate aspect. However, this approach is very similar to that in [10]. Our minor contributions in this respect are as follows. First, polarity lexicon is generated beforehand by using the domain-specific semi-supervised approach discussed in Section 4.2.1.2. This is a much more sensible alternative compared to the use of a common/generic polarity lexicon. Second, we take into account intensifiers, downtoners, and negators. For instance, when the word *"most"* occurs in the text before an adjective, we increase the strength of its sentiment and assign that score to the modified aspect/noun. So as to evaluate the performance of this approach,

we annotated a small set of the movie dataset as aspects and their sentiments. We achieved a success rate of around 70% for this small set. However, since we have not performed a comprehensive analysis for the ABSA in Turkish, we do not include the experimental results here.

As future work, we plan to employ a common corpus and filter out some words as non-aspect if they appear in both the target dataset and this generic corpus. The reason is that some aspects are specific to a domain and those words occurring in generic sentences very frequently can be weighed less as aspect candidates.

5. CONCLUSIONS AND FUTURE WORK

In this thesis, we have made five major and three minor contributions on the sentiment classification task in Turkish and English, which are visually summarised in Figure 5.1. We addressed several aspects of and issues about sentiment analysis that exist for both Turkish and English. As discussed in the above separate sections, we elaborated on all our approaches. However, we summarise them all here as well as the final chapter, albeit a bit repetitively.

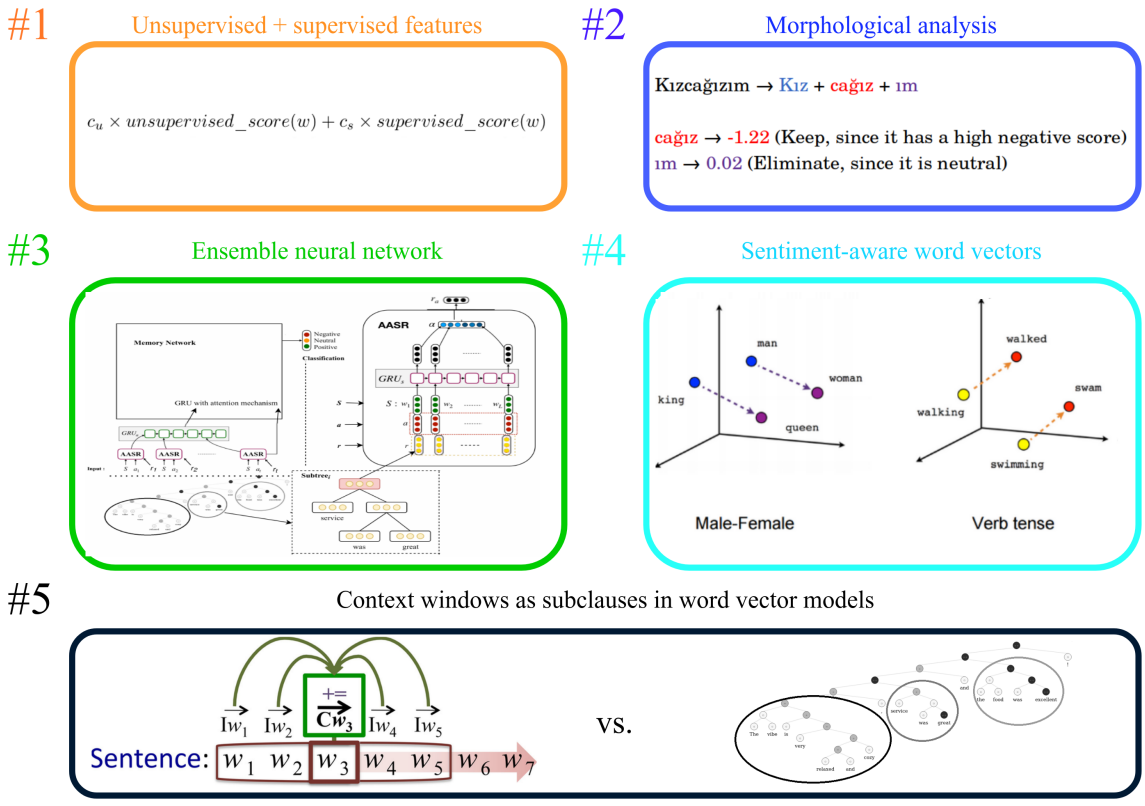

Figure 5.1. Visual summaries of the five major contributions.

(i) *Major contribution #1:* We have generated effective and novel feature engineering techniques by combining unsupervised, semi-supervised, and supervised metrics. Our research question was whether taking unsupervised characteristics into account in addition to supervised features in an ensemble form could contribute to the representations of words. After generating a polarity lexicon on a word-basis,

we employed several classical machine learning models taking these scalar word sentiment scores as inputs. We thereby outperformed neural network models for several corpora in English and Turkish. That shows that classical machine learning methods still have the potential to compete with neural network models, when employing robust feature engineering techniques. This model is cross-domain and is also portable to other languages with all results being significant.

(ii) *Major contribution #2:* We have performed a fine morphological analysis specific to Turkish, an agglutinative language, for sentiment analysis. In the literature, mostly only root forms of words are employed and their polarities are computed when conducting opinion mining. In our work, we have originally specified a set of rules and applied them when assigning a sentiment score to each morpheme attached to the root form of a word. We have observed that some suffixes are more expressive of sentiments and filtering out these morphemes that are not sentimentally discriminative boosts the performance for Turkish. We employed two corpora in this language to build a generic and generalisable morpheme polarity lexicon. This approach is proved to be cross-domain with significant performances. We conjecture that this approach can also be applied to the sentiment classification task for other morphologically-rich languages and even to other NLP classification tasks. The source code for this and the above approach is publicly available [135].

(iii) *Major contribution #3:* Related works on ABSA in English mostly rely on either recurrent or recursive neural network models; however, only a few of them merge these two architectures and generate an ensemble classifier. In this approach, we combine recursive and recurrent neural networks in an original way by improving a model described in a baseline study. In this baseline study, the researchers only employ a recurrent neural network model exploiting the inter-aspect relations. They model the propagation effect of sentiments on aspects throughout the text. We enrich this system by incorporating a recursive neural network architecture. We first extract sub-reviews/subclauses in a novel way for each aspect in a review. Then, we generate a recursive parse tree for each aspect and train them in a distant-supervision manner using a dependency and constituency parser.

Thereafter, we feed the root vectors of these trees as input into the recurrent model (baseline) per aspect. We thereby take into account temporal and grammatical characteristics at the same time in this original ensemble neural model architecture. Temporal information is modelled by the recurrent model, whereas grammatical structures represent the sentiment information more robustly. We again achieve state-of-the-art and significant results for this approach for two datasets of different genres in English. We rank first and second in a worldwide competition for both these corpora. The corresponding source code for this approach is publicly available as well [136].

(iv) *Major contribution #4:* In the literature, word embeddings in general can ignore the sentiment information, and exploit only the semantic and syntactic information. Accordingly, we addressed this issue by generating word vectors that also take into account sentimental and lexical information in a novel way in addition to capturing syntactic and semantic knowledge. We have found out that we could even outperform the supervised word embedding generation approach by employing an unsupervised model when modelling word representations. We also performed better than a baseline model, which is the popular word2vec, a shallow neural network architecture, for almost all the scenarios, whereby we exploit the above-mentioned characteristics. We evaluated our word generation approaches on corpora of different genres in two languages, which are Turkish and English. This approach is also cross-domain and portable to other languages, with results being significant. The relevant source code for this approach is publicly available [137].

(v) *Major contribution #5:* When generating word representation models, statistical information based on context windows is leveraged. In the literature, this contextual knowledge is in general extracted based mostly on windows of a fixed-size length. Researchers look up the words in the symmetrical or asymmetrical windows of a target token when modelling its embedding. However, this rigid approach may overlook several aspects about a word. There are only a few studies in the literature that define subclauses as context windows. However, in our approach, the rule set we develop for generating sub-sentences is novel. As com-

pared to the use of windows with a fixed-size, when we employ subclauses as the context windows for target words, we arrive at more robust representations, since these can capture the relevant syntactic, semantic, and even sentimental information more sensibly. For example, even two neighbour words in a window with a fixed length could be considered far away from each other semantically and syntactically. We thereby addressed this issue in this approach. We evaluated our methods by training two word embedding models on two corpora in English. After generating word representation models, we averaged all the vectors of words appearing in documents and built document vectors. We then fed them as input into two NLP tasks, which are sentiment and spam classifications. We arrived at better results for almost all hyper-parameter values and cases as compared to the baseline method, which only uses fixed-size context windows. This approach is cross-model and cross-model. Despite the fact that this contribution is not specific to sentiment analysis unlike the other contributions, we conjecture this definition of context windows can also be applied to other NLP or machine learning problems, such as CNN and LSTM, with minor changes and adaptations. The corresponding source code for this approach is publicly available as well [138].

Minor contributions #1, #2, and #3: Our minor contributions for the sentiment analysis problem are listed as follows. (1) We tweaked the parameters of a semi-supervised approach so that we could generate more robust domain-specific polarity lexicons, which can be adapted to any domain and language with minor changes. (2) We extracted aspects from raw text in English using several feature engineering techniques that are not very much novel and cannot outperform the baseline by a significant margin. (3) We performed aspect-based sentiment analysis for Turkish that has a few novel characteristics inherent to it compared to a baseline approach, for which the source code is publicly available as well [139].

In summary, in this thesis we addressed issues specific to not only Turkish, but also English for several aspects of the sentiment analysis problem. We generated several polarity lexicons, feature sets, original neural network models, and other several ap-

proaches that have thus far not been handled by researchers for both of these languages. We think our work has been the most detailed and comprehensive research conducted for sentiment analysis in Turkish as of July, 2020 [140]. Some of our methods for the English language also have major contributions that fill the gap for this classification task by the mentioned novel aspects in this thesis. Several other approaches developed by us also boost the performance for the existing machine learning models. We have additionally built several methods, such as subclause and context window generations, which are specific to not only sentiment analysis domain, but also to generic machine learning models, linguistic models, and other NLP tasks. Our frameworks can be applied to several other agglutinative or non-agglutinative languages and datasets for sentiment analysis, and other classification problems.

As future work, in post-doctoral research, we plan to merge all these major and minor contributions into a single, more comprehensive opinion mining framework by performing minor changes. We also plan to develop a "complete" framework for the sentiment classification task in Turkish. In this scenario, we would also (1) take account of the time and opinion holder components as defined in the quintuple representation, which we have not been modelled in this thesis. Additionally, for all our approaches alone, we would (2) perform a more comprehensive error analysis. We also plan to (3) rely on neural networks when generating a morpheme polarity lexicon for the Turkish language, (4) apply our word context window definitions to other corpora, languages, and NLP tasks, (5) enhance our model for the aspect-based sentiment analysis in Turkish as discussed, and (6) improve our ensemble neural network framework by incorporating more robust sentiment information into the model as mentioned. When these would be implemented as well, we would make a broader contribution to the sentiment analysis domain for Turkish, other agglutinative languages when applied with minor changes, English, and even to other computational linguistics and NLP tasks.

5.1. Publications Obtained in this Thesis

We have three scientific papers published during the thesis stage, one of which is a long conference paper [87] and two of which are journal papers [47, 49]. In addition, we have submitted one of our approaches to a conference, from which the notification of acceptance is yet to be made. These publications and papers are as follows:

(i) *IEEE Access*

Aydın, C. R. and Güngör, T., "Combination of Recursive and Recurrent Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations," in IEEE Access, vol. 8, pp. 77820-77832, 2020, doi: 10.1109/ACCESS.2020.2990306.

(ii) *CICLing 2019*

Aydın, C. R., Güngör, T., and Erkan, A., "Generating Word and Document Embeddings for Sentiment Analysis", 16th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2019), Ed. A.Gelbukh, April 2019, La Rochelle, France.

(iii) *Natural Language Engineering (Cambridge University Press)*

Aydın, C. R. and Güngör, T. (2020). "Sentiment Analysis in Turkish: Supervised, Semi-Supervised, and Unsupervised Techniques", Natural Language Engineering, 1-29. doi:10.1017/S1351324920000200

(iv) *EMNLP 2020*

Submitted; the notification of acceptance is yet to be announced. (A short paper entitled "Redefining Context Windows as Subclauses for Word Vector Models")

The *"IEEE Access"* journal in which our paper has been published is the third best journal in the field of Engineering & Computer Science according to Google Scholar as of July, 2020 [141]. On the other hand, the conference *"CICLing"* and the journal *"Natural Language Engineering"* that our works have been accepted for and published in are both among the top 20 publications on computational linguistics and NLP domains according to Google Scholar as of July, 2020 [142]. We have also submitted one

short paper to EMNLP 2020, the second best publication in the field of computational linguistics and NLP according to the same Google Scholar URL reference given above, from which the notification of acceptance is yet to be made.

REFERENCES

1. Hamilton, W. L., K. Clark, J. Leskovec and D. Jurafsky, "Inducing Domain-Specific Sentiment Lexicons from Unlabeled Corpora", *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 595–605, Association for Computational Linguistics, Austin, Texas, Nov. 2016, https://www.aclweb.org/anthology/D16-1057.

2. Majumder, N., S. Poria, A. Gelbukh, M. S. Akhtar, E. Cambria and A. Ekbal, "IARM: Inter-Aspect Relation Modeling with Memory Networks in Aspect-Based Sentiment Analysis", *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3402–3411, Association for Computational Linguistics, Brussels, Belgium, Oct.-Nov. 2018, https://www.aclweb.org/anthology/D18-1377.

3. Stepanova, E., "The Role of Information Communication Technologies in the Arab Spring", *PONARS Eurasia Policy Memo*, pp. 1–6, Jan. 2011.

4. Liu, B., *Sentiment Analysis and Opinion Mining*, Morgan & Claypool Publishers, 2012.

5. Wiebe, J. M., R. F. Bruce and T. P. O'Hara, "Development and Use of a Gold-Standard Data Set for Subjectivity Classifications", *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 246–253, Association for Computational Linguistics, College Park, Maryland, USA, Jun. 1999, https://www.aclweb.org/anthology/P99-1032.

6. Jindal, N. and B. Liu, "Identifying Comparative Sentences in Text Documents", *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pp. 244–251, Association for Computing Machinery, New York, NY, USA, 2006,

https://doi.org/10.1145/1148170.1148215.

7. Jindal, N. and B. Liu, "Opinion Spam and Analysis", *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pp. 219–230, Association for Computing Machinery, New York, NY, USA, 2008, https://doi.org/10.1145/1341531.1341560.

8. Mikolov, T., I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality", *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 3111–3119, Curran Associates Inc., Red Hook, NY, USA, 2013.

9. Pennington, J., R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation", *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Association for Computational Linguistics, Doha, Qatar, Oct. 2014, https://www.aclweb.org/anthology/D14-1162.

10. Dehkharghani, R., B. Yanikoglu, Y. Saygin and K. Oflazer, "Sentiment Analysis in Turkish: Towards a Complete Framework", *Natural Language Engineering*, pp. 1–20, 2015.

11. Hobbs, J. R., "Information Extraction", N. Indurkhya and F. J. Damerau (Editors), *Handbook of Natural Language Processing, Second Edition*, pp. 511–532, Chapman and Hall/CRC, 2010.

12. Mooney, R. J. and R. Bunescu, "Mining Knowledge from Text Using Information Extraction", *SIGKDD Explor. Newsl.*, Vol. 7, No. 1, p. 3–10, Jun. 2005, https://doi.org/10.1145/1089815.1089817.

13. Sarawagi, S., "Information Extraction", *Found. Trends Databases*, Vol. 1, No. 3,

pp. 261–377, Mar. 2008, https://doi.org/10.1561/1900000003.

14. Hu, M. and B. Liu, "Mining and Summarizing Customer Reviews", *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pp. 168–177, Association for Computing Machinery, New York, NY, USA, 2004, https://doi.org/10.1145/1014052.1014073.

15. Zhang, L. and B. Liu, "Identifying Noun Product Features That Imply Opinions", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pp. 575–580, Association for Computational Linguistics, USA, 2011.

16. Greene, S. and P. Resnik, "More than Words: Syntactic Packaging and Implicit Sentiment", *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pp. 503–511, Association for Computational Linguistics, USA, 2009.

17. Turney, P., "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews", *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 417–424, Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, Jul. 2002, https://www.aclweb.org/anthology/P02-1053.

18. Taboada, M., C. Anthony and K. Voll, "Methods for Creating Semantic Orientation Dictionaries", *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pp. 427–432, European Language Resources Association (ELRA), Genoa, Italy, May 2006.

19. Hatzivassiloglou, V. and K. R. McKeown, "Predicting the Semantic Orientation of Adjectives", *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Com-*

putational Linguistics, pp. 174–181, Association for Computational Linguistics, Madrid, Spain, Jul. 1997, https://www.aclweb.org/anthology/P97-1023.

20. Baccianella, S., A. Esuli and F. Sebastiani, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining", *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pp. 2200–2204, European Language Resources Association (ELRA), Valletta, Malta, May 2010.

21. Martínez-Cámara, E., M. Martín-Valdivia, M. D. González and J. Perea-Ortega, "Integrating Spanish Lexical Resources by Meta-Classifiers for Polarity Classification", *Journal of Information Science*, Vol. 40, pp. 539–554, July 2014.

22. Gang Li and Fei Liu, "A clustering-based approach on sentiment analysis", *2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering*, pp. 331–337, 2010.

23. Yıldırım, E., F. Çetin, G. Eryiğit and T. Temel, "The Impact of NLP on Turkish Sentiment Analysis", *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, Vol. 8, April 2014.

24. Martineau, J. and T. Finin, "Delta TFIDF: An Improved Feature Space for Sentiment Analysis", *ICWSM*, pp. 1–4, The AAAI Press, 2009.

25. Farhadloo, M. and E. Rolland, "Multi-Class Sentiment Analysis with Clustering and Score Representation", *2013 IEEE 13th International Conference on Data Mining Workshops*, pp. 904–912, 2013.

26. dos Santos, C. and M. Gatti, "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts", *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 69–78, Dublin City University and Association for Computational Linguistics, Dublin, Ireland,

Aug. 2014, https://www.aclweb.org/anthology/C14-1008.

27. Lango, M., D. Brzezinski and J. Stefanowski, "PUT at SemEval-2016 Task 4: The ABC of Twitter Sentiment Analysis", *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 126–132, Association for Computational Linguistics, San Diego, California, 2016, https://www.aclweb.org/anthology/S16-1018.

28. Thelwall, M., K. Buckley and G. Paltoglou, "Sentiment Strength Detection for the Social Web", *J. Am. Soc. Inf. Sci. Technol.*, Vol. 63, No. 1, pp. 163–173, Jan. 2012, https://doi.org/10.1002/asi.21662.

29. Wang, S. and C. Manning, "Baselines and Bigrams: Simple, Good Sentiment and Topic Classification", *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 90–94, Association for Computational Linguistics, Jeju Island, Korea, Jul. 2012, https://www.aclweb.org/anthology/P12-2018.

30. Guha, S., A. Joshi and V. Varma, "SIEL: Aspect Based Sentiment Analysis in Reviews", *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 759–766, Association for Computational Linguistics, Denver, Colorado, Jun. 2015, https://www.aclweb.org/anthology/S15-2129.

31. Jiang, M., M. Lan and Y. Wu, "ECNU at SemEval-2017 Task 5: An Ensemble of Regression Algorithms with Effective Features for Fine-Grained Sentiment Analysis in Financial Domain", *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 888–893, Association for Computational Linguistics, Vancouver, Canada, Aug. 2017, https://www.aclweb.org/anthology/S17-2152.

32. Saroufim, C., A. Almatarky and M. Abdel Hady, "Language Independent Sentiment Analysis with Sentiment-Specific Word Embeddings", *Proceedings of the*

9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 14–23, Association for Computational Linguistics, Brussels, Belgium, Oct. 2018, https://www.aclweb.org/anthology/W18-6204.

33. Maas, A. L., R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng and C. Potts, "Learning Word Vectors for Sentiment Analysis", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Association for Computational Linguistics, Portland, Oregon, USA, Jun. 2011, https://www.aclweb.org/anthology/P11-1015.

34. Felbo, B., A. Mislove, A. Søgaard, I. Rahwan and S. Lehmann, "Using Millions of Emoji Occurrences to Learn Any-Domain Representations for Detecting Sentiment, Emotion and Sarcasm", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1615–1625, Association for Computational Linguistics, Copenhagen, Denmark, Sep. 2017, https://www.aclweb.org/anthology/D17-1169.

35. Baziotis, C., N. Pelekis and C. Doulkeridis, "DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis", *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 747–754, Association for Computational Linguistics, Vancouver, Canada, Aug. 2017, https://www.aclweb.org/anthology/S17-2126.

36. Kaya, M., G. Fidan and I. H. Toroslu, "Sentiment Analysis of Turkish Political News", *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, Vol. 1, pp. 174–180, 2012.

37. Çetin, M. and M. Fatih Amasyalı, "Active learning for Turkish sentiment analysis", *2013 IEEE INISTA*, pp. 1–4, 2013.

38. Türkmenoğlu, C. and A. C. Tantuğ, "Sentiment Analysis in Turkish Media", *Workshop on Issues of Sentiment Discovery and Opinion Mining, International Con-*

ference on Machine Learning (ICML), 2014.

39. Vural, A. G., B. B. Cambazoglu, P. Senkul and Z. O. Tokgoz, "A Framework for Sentiment Analysis in Turkish: Application to Polarity Detection of Movie Reviews in Turkish", E. Gelenbe and R. Lent (Editors), *Computer and Information Sciences III*, pp. 437–445, Springer London, London, 2013.

40. Dehkharghani, R., Y. Saygin, B. Yanikoglu and K. Oflazer, "SentiTurkNet: A Turkish Polarity Lexicon for Sentiment Analysis", *Lang. Resour. Eval.*, Vol. 50, No. 3, pp. 667–685, Sep. 2016, https://doi.org/10.1007/s10579-015-9307-6.

41. Abdul-Mageed, M., M. Diab and M. Korayem, "Subjectivity and Sentiment Analysis of Modern Standard Arabic", *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 587–591, Association for Computational Linguistics, Portland, Oregon, USA, Jun. 2011, https://www.aclweb.org/anthology/P11-2103.

42. Joshi, A., B. A. R and P. Bhattacharyya, "A Fall-back Strategy for Sentiment Analysis in Hindi: A Case Study", *Proceedings of the 8th ICON*, 2010.

43. Yang, H.-L. and A. F. Chao, "Sentiment Analysis for Chinese Reviews of Movies in Multi-Genre Based on Morpheme-Based Features and Collocations", *Information Systems Frontiers*, Vol. 17, No. 6, pp. 1335–1352, Dec. 2015, https://doi.org/10.1007/s10796-014-9498-1.

44. Jang, H. and H. Shin, "Language-Specific Sentiment Analysis in Morphologically Rich Languages", *Coling 2010: Posters*, pp. 498–506, Coling 2010 Organizing Committee, Beijing, China, Aug. 2010, https://www.aclweb.org/anthology/C10-2057.

45. Medagoda, N., "Sentiment Analysis on Morphologically Rich Languages: An Artificial Neural Network (ANN) Approach", S. Shanmu-

ganathan and S. Samarasinghe (Editors), *Artificial Neural Network Modelling*, pp. 377–393, Springer International Publishing, Cham, 2016, https://doi.org/10.1007/978-3-319-28495-8_17.

46. Medagoda, N., *Framework for Sentiment Classification for Morphologically Rich Languages: A Case Study for Sinhala*, Ph.D. Thesis, Auckland University of Technology, 2017.

47. Aydın, C. R. and T. Güngör, "Sentiment Analysis in Turkish: Supervised, Semi-Supervised, and Unsupervised Techniques", *Natural Language Engineering*, pp. 1–29, 2020.

48. Cambria, E., "Affective Computing and Sentiment Analysis", *IEEE Intelligent Systems*, Vol. 31, No. 2, pp. 102–107, Mar. 2016, https://doi.org/10.1109/MIS.2016.31.

49. Aydın, C. R. and T. Güngör, "Combination of Recursive and Recurrent Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations", *IEEE Access*, Vol. 8, pp. 77820–77832, 2020.

50. Schouten, K. and F. Frasincar, "Survey on Aspect-Level Sentiment Analysis", *IEEE Trans. on Knowl. and Data Eng.*, Vol. 28, No. 3, pp. 813–830, Mar. 2016, https://doi.org/10.1109/TKDE.2015.2485209.

51. Poria, S., E. Cambria and A. Gelbukh, "Aspect Extraction for Opinion Mining with a Deep Convolutional Neural Network", *Know.-Based Syst.*, Vol. 108, No. C, pp. 42–49, Sep. 2016, https://doi.org/10.1016/j.knosys.2016.06.009.

52. He, R., W. S. Lee, H. T. Ng and D. Dahlmeier, "Effective Attention Modeling for Aspect-Level Sentiment Classification", *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1121–1131, Association for Computational Linguistics, Santa Fe, New Mexico, USA, Aug. 2018,

https://www.aclweb.org/anthology/C18-1096.

53. Ma, Y., H. Peng and E. Cambria, "Targeted Aspect-Based Sentiment Analysis via Embedding Commonsense Knowledge into an Attentive LSTM", pp. 5866–5873, Feb. 2018, https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16541.

54. Semantic Evaluation, *SemEval-2014 Task 4, Task Description: Aspect Based Sentiment Analysis (ABSA)*, 2014, http://alt.qcri.org/semeval2014/task4/, accessed in November 2019.

55. Pontiki, M., D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos and S. Manandhar, "SemEval-2014 Task 4: Aspect Based Sentiment Analysis", *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 27–35, Association for Computational Linguistics, Dublin, Ireland, Aug. 2014, https://www.aclweb.org/anthology/S14-2004.

56. Kiritchenko, S., X. Zhu, C. Cherry and S. Mohammad, "NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews", *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 437–442, Association for Computational Linguistics, Dublin, Ireland, Aug. 2014, https://www.aclweb.org/anthology/S14-2076.

57. Wagner, J., P. Arora, S. Cortes, U. Barman, D. Bogdanova, J. Foster and L. Tounsi, "DCU: Aspect-based Polarity Classification for SemEval Task 4", *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 223–229, Association for Computational Linguistics, Dublin, Ireland, Aug. 2014, https://www.aclweb.org/anthology/S14-2036.

58. Goldberg, Y. and G. Hirst, *Neural Network Methods in Natural Language Processing*, Morgan & Claypool Publishers, 2017.

59. Chen, P., Z. Sun, L. Bing and W. Yang, "Recurrent Attention Network on Memory for Aspect Sentiment Analysis", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 452–461, Association for Computational Linguistics, Copenhagen, Denmark, Sep. 2017, https://www.aclweb.org/anthology/D17-1047.

60. Arras, L., G. Montavon, K.-R. Müller and W. Samek, "Explaining Recurrent Neural Network Predictions in Sentiment Analysis", *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 159–168, Association for Computational Linguistics, Copenhagen, Denmark, Sep. 2017, https://www.aclweb.org/anthology/W17-5221.

61. Ma, D., S. Li, X. Zhang and H. Wang, "Interactive Attention Networks for Aspect-Level Sentiment Classification", *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pp. 4068–4074, AAAI Press, 2017.

62. Araque, O., R. Barbado, J. F. Sánchez-Rada and C. A. Iglesias, "Applying Recurrent Neural Networks to Sentiment Analysis of Spanish Tweets", *TASS 2017: Workshop on Semantic Analysis at SEPLN*, pp. 71–76, Murcia, Spain, 2017.

63. Socher, R., A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng and C. Potts, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank", *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Association for Computational Linguistics, Seattle, Washington, USA, Oct. 2013.

64. Korbak, T. and P. Zak, "Fine-Tuning Tree-LSTM for Phrase-Level Sentiment Classification on a Polish Dependency Treebank. Submission to PolEval task 2", *CoRR*, Vol. abs/1711.01985, 2017, http://arxiv.org/abs/1711.01985.

65. Wang, W., S. J. Pan, D. Dahlmeier and X. Xiao, "Recursive Neural Conditional Random Fields for Aspect-based Sentiment Analysis", *Proceedings of the*

2016 Conference on Empirical Methods in Natural Language Processing, pp. 616–626, Association for Computational Linguistics, Austin, Texas, Nov. 2016, https://www.aclweb.org/anthology/D16-1059.

66. Nguyen, T. H. and K. Shirai, "PhraseRNN: Phrase Recursive Neural Network for Aspect-based Sentiment Analysis", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2509–2514, Association for Computational Linguistics, Lisbon, Portugal, Sep. 2015, https://www.aclweb.org/anthology/D15-1298.

67. Hoogervorst, R., E. Essink, W. Jansen, M. Helder, K. Schouten, F. Frasincar and M. Taboada, "Aspect-Based Sentiment Analysis on the Web Using Rhetorical Structure Theory", pp. 317–334, Jun. 2016.

68. Heerschop, B., F. Goossen, A. Hogenboom, F. Frasincar, U. Kaymak and F. de Jong, "Polarity Analysis of Texts Using Discourse Structure", *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pp. 1061–1070, Association for Computing Machinery, New York, NY, USA, 2011, https://doi.org/10.1145/2063576.2063730.

69. Hogenboom, A., F. Frasincar, F. de Jong and U. Kaymak, "Using Rhetorical Structure in Sentiment Analysis", *Commun. ACM*, Vol. 58, No. 7, pp. 69–77, Jun. 2015, https://doi.org/10.1145/2699418.

70. Taboada, M., K. Voll and J. Brooke, "Extracting Sentiment as a Function of Discourse Structure and Topicality", *SFU, School of Computing Science*, Burnaby, BC, Canada, 2008.

71. Van, V. D., T. Thai and M.-Q. Nghiem, "Combining Convolution and Recursive Neural Networks for Sentiment Analysis", *Proceedings of the Eighth International Symposium on Information and Communication Technology*, SoICT 2017, pp. 151–158, Association for Computing Machinery, New York, NY, USA, 2017,

https://doi.org/10.1145/3155133.3155158.

72. Yang, F., C. Du and L. Huang, "Ensemble Sentiment Analysis Method based on R-CNN and C-RNN with Fusion Gate", *Internation Journal of Computers Communications & Control*, Vol. 14, No. 2, pp. 272–285, 2019, http://univagora.ro/jour/index.php/ijccc/article/view/3375.

73. Minaee, S., E. Azimi and A. Abdolrashidi, "Deep-Sentiment: Sentiment Analysis Using Ensemble of CNN and Bi-LSTM Models", *CoRR*, Vol. abs/1904.04206, 2019, http://arxiv.org/abs/1904.04206.

74. Chen, G., D. Ye, Z. Xing, J. Chen and E. Cambria, "Ensemble Application of Convolutional and Recurrent Neural Networks for Multi-Label Text Categorization", *IJCNN-2017*, pp. 2377–2383, Anchorage, AK, USA, May 2017.

75. Wallaart, O. and F. Frasincar, *A Hybrid Approach for Aspect-Based Sentiment Analysis Using a Lexicalized Domain Ontology and Attentional Neural Models*, pp. 363–378, Portorož, Slovenia, May 2019.

76. Meskele, D. and F. Frasincar, "ALDONA: A Hybrid Solution for Sentence-level Aspect-based Sentiment Analysis Using a Lexicalised Domain Ontology and a Neural Attention Model", *The 34th ACM/SIGAPP Symposium On Applied Computing (SAC)*, pp. 2489–2496, Limassol, Cyprus, Apr. 2019.

77. Meškele, D. and F. Frasincar, "ALDONAr: A Hybrid Solution for Sentence-Level Aspect-Based Sentiment Analysis Using a Lexicalized Domain Ontology and a Regularized Neural Attention Model", *Inf. Process. Manage.*, Vol. 57, No. 3, May 2020, https://doi.org/10.1016/j.ipm.2020.102211.

78. Turney, P. D. and P. Pantel, "From Frequency to Meaning: Vector Space Models of Semantics", *J. Artif. Int. Res.*, Vol. 37, No. 1, pp. 141–188, Jan. 2010.

79. Cortes, C. and V. Vapnik, "Support-Vector Networks", *Mach. Learn.*, Vol. 20,

No. 3, pp. 273–297, Sep. 1995, https://doi.org/10.1023/A:1022627411411.

80. Blei, D. M., A. Y. Ng and M. I. Jordan, "Latent Dirichlet Allocation", *J. Mach. Learn. Res.*, Vol. 3, pp. 993–1022, Mar. 2003.

81. Li, F., M. Huang and X. Zhu, "Sentiment Analysis with Global Topics and Local Dependency", *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, pp. 1371–1376, AAAI Press, 2010.

82. Lin, C. and Y. He, "Joint Sentiment/Topic Model for Sentiment Analysis", *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pp. 375–384, Association for Computing Machinery, New York, NY, USA, 2009, https://doi.org/10.1145/1645953.1646003.

83. Boyd-Graber, J. and P. Resnik, "Holistic Sentiment Analysis Across Languages: Multilingual Supervised Latent Dirichlet Allocation", *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 45–55, Association for Computational Linguistics, Cambridge, MA, Oct. 2010, https://www.aclweb.org/anthology/D10-1005.

84. Ertugrul, A. M., I. Onal Ertugrul and C. Acarturk, "Does the Strength of Sentiment Matter? A Regression Based Approach on Turkish Social Media", *22nd International Conference on Applications of Natural Language to Information Systems (NLDB)*, pp. 149–155, Liege, Brussels, Jun. 2017.

85. Ye, Z., F. Li and T. Baldwin, "Encoding Sentiment Information into Word Vectors for Sentiment Analysis", *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 997–1007, Association for Computational Linguistics, Santa Fe, New Mexico, USA, Aug. 2018, https://www.aclweb.org/anthology/C18-1085.

86. Tang, D., F. Wei, B. Qin, T. Liu and M. Zhou, "Coooolll: A Deep

Learning System for Twitter Sentiment Classification", *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 208–212, Association for Computational Linguistics, Dublin, Ireland, Aug. 2014, https://www.aclweb.org/anthology/S14-2033.

87. Aydın, C. R., T. Güngör and A. Erkan, "Generating Word and Document Embeddings for Sentiment Analysis", *20th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2019), Ed. A. Gelbukh*, pp. 1–12, La Rochelle, France, Apr. 2019.

88. Goldberg, Y., "A Primer on Neural Network Models for Natural Language Processing", *J. Artif. Int. Res.*, Vol. 57, No. 1, pp. 345–420, Sep. 2016.

89. Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. P. Kuksa, "Natural Language Processing (Almost) from Scratch.", *Journal of Machine Learning Research*, Vol. 12, pp. 2493–2537, 2011, http://dblp.uni-trier.de/db/journals/jmlr/jmlr12.html.

90. Lin, C.-C., W. Ammar, C. Dyer and L. Levin, "Unsupervised POS Induction with Word Embeddings", *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1311–1316, Association for Computational Linguistics, Denver, Colorado, May–Jun. 2015, https://www.aclweb.org/anthology/N15-1144.

91. Zhou, J. and W. Xu, "End-to-End Learning of Semantic Role Labeling Using Recurrent Neural Networks", *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)*, pp. 1127–1137, Association for Computational Linguistics, Beijing, China, Jul. 2015, https://www.aclweb.org/anthology/P15-1109.

92. Lample, G., M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer,

"Neural Architectures for Named Entity Recognition", *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 260–270, Association for Computational Linguistics, San Diego, California, Jun. 2016, https://www.aclweb.org/anthology/N16-1030.

93. Levy, O., Y. Goldberg and I. Dagan, "Improving Distributional Similarity with Lessons Learned from Word Embeddings", *Transactions of the Association for Computational Linguistics*, Vol. 3, pp. 211–225, 2015, https://www.aclweb.org/anthology/Q15-1016.

94. Levy, O. and Y. Goldberg, "Dependency-Based Word Embeddings", *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 302–308, Association for Computational Linguistics, Baltimore, Maryland, Jun. 2014, https://www.aclweb.org/anthology/P14-2050.

95. MacAvaney, S. and A. Zeldes, "A Deeper Look into Dependency-Based Word Embeddings", *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 40–45, Association for Computational Linguistics, New Orleans, Louisiana, USA, Jun. 2018, https://www.aclweb.org/anthology/N18-4006.

96. Bengio, Y., R. Ducharme, P. Vincent and C. Janvin, "A Neural Probabilistic Language Model", *J. Mach. Learn. Res.*, Vol. 3, pp. 1137–1155, Mar. 2003.

97. Lison, P. and A. Kutuzov, "Redefining Context Windows for Word Embedding Models: An Experimental Study", *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pp. 284–288, Association for Computational Linguistics, Gothenburg, Sweden, May 2017.

98. Ravanelli, M. and M. Omologo, "Automatic Context Window Composition for

Distant Speech Recognition", *Speech Communication*, Vol. 101, pp. 1–31, May 2018.

99. Akın, A. A. and M. D. Akın, "Zemberek, an open source NLP framework for Turkic Languages", *Structure*, Vol. 10, pp. 1–5, 2007.

100. Torunoğlu, D. and G. Eryiğit, "A Cascaded Approach for Social Media Text Normalization of Turkish", *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pp. 62–70, Association for Computational Linguistics, Gothenburg, Sweden, Apr. 2014, https://www.aclweb.org/anthology/W14-1308.

101. Honnibal, M. and M. Johnson, "An Improved Non-monotonic Transition System for Dependency Parsing", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1373–1378, Association for Computational Linguistics, Lisbon, Portugal, Sep. 2015, https://www.aclweb.org/anthology/D15-1162.

102. Sak, H., T. Güngör and M. Saraçlar, "Turkish Language Resources: Morphological Parser, Morphological Disambiguator and Web Corpus", *Proceedings of the 6th International Conference on Advances in Natural Language Processing (GoTAL '08)*, pp. 417–427, Springer-Verlag, 2008, http://dx.doi.org/10.1007/978-3-540-85287-2_40.

103. Sak, H., T. Güngör and M. Saraçlar, "Morphological Disambiguation of Turkish Text with Perceptron Algorithm", *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2017)*, pp. 107–118, CICLing Press, 2007, http://dx.doi.org/10.1007/978-3-540-70939-8_10.

104. Davidov, D., O. Tsur and A. Rappoport, "Enhanced Sentiment Learning Using Twitter Hashtags and Smileys", *Coling 2010: Posters*, pp.

241–249, Coling 2010 Organizing Committee, Beijing, China, Aug. 2010, https://www.aclweb.org/anthology/C10-2028.

105. Bird, S., E. Klein and E. Loper, *Natural Language Processing with Python*, O'Reilly Media, Inc., 1st edn., 2009.

106. Kulcu, S. and E. Dogdu, "A Scalable Approach for Sentiment Analysis of Turkish Tweets and Linking Tweets to News", *Proceedings of the 2016 IEEE Tenth International Conference on Semantic Computing*, pp. 471–476, Noida, India, Feb. 2016.

107. Fontes, L. A., *Interviewing Client Across Cultures: A Practitioner's Guide*, 2009.

108. Yandex, *Yandex Search Engine*, 2015, https://www.yandex.com.tr, accessed in April 2016.

109. Ng, A. Y. and M. I. Jordan, "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes", T. G. Dietterich, S. Becker and Z. Ghahramani (Editors), *Advances in Neural Information Processing Systems 14* , pp. 841–848, MIT Press, 2002.

110. Hochreiter, S. and J. Schmidhuber, "Long Short-Term Memory", *Neural Comput.*, Vol. 9, No. 8, pp. 1735–1780, Nov. 1997, https://doi.org/10.1162/neco.1997.9.8.1735.

111. Yildiz, E., C. Tirkaz, H. B. Sahin, M. T. Eren and O. Sonmez, "A Morphology-Aware Network for Morphological Disambiguation", *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 2863–2869, AAAI Press, 2016.

112. Britz, D., *Convolutional Neural Network for Text Classification in TensorFlow*, 2016, https://github.com/dennybritz/cnn-text-classification-tf, accessed in July 2018.

113. Güngör, O. and E. Yıldız, "Linguistic Features in Turkish Word Representations", *2017 25th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2017.

114. BeyazPerde, *Beyazperde: Film Haberleri, Eleştirileri, Sinema Seansları, Fragmanlar, Videolar, TV Programları, TV Dizileri!*, 2013, `http://www.beyazperde.com`, accessed in March 2014.

115. Pang, B. and L. Lee, "Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales", *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 115–124, Association for Computational Linguistics, Ann Arbor, Michigan, Jun. 2005.

116. Go, A., R. Bhayani and L. Huang, "Twitter Sentiment Classification Using Distant Supervision", *Processing*, Vol. 150, Jan. 2009.

117. Rosenthal, S., N. Farra and P. Nakov, "SemEval-2017 Task 4: Sentiment Analysis in Twitter", *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 502–518, Association for Computational Linguistics, Vancouver, Canada, Aug. 2017.

118. Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-Learn: Machine Learning in Python", *J. Mach. Learn. Res.*, Vol. 12, pp. 2825–2830, Nov. 2011.

119. Chen, R. and K. Sun, "Fast OOV words incorporation using structured word embeddings for neural network language model", *International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pp. 6119–6123, Association for Computational Linguistics, Calgary, Canada, 2018.

120. Garneau, N., J.-S. Leboeuf and L. Lamontagne, "Predicting and Interpreting Em-

beddings for out of Vocabulary Words in Downstream Tasks", *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 331–333, Association for Computational Linguistics, Brussels, Belgium, Nov. 2018, https://www.aclweb.org/anthology/W18-5439.

121. Horn, F., "Context Encoders as a Simple but Powerful Extension of word2vec", *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 10–14, Association for Computational Linguistics, Vancouver, Canada, Aug. 2017, https://www.aclweb.org/anthology/W17-2602.

122. Majumder, N., S. Poria, A. Gelbukh, M. S. Akhtar, E. Cambria and A. Ekbal, *IARM: Inter-Aspect Relation Modeling with Memory Networks in Aspect-Based Sentiment Analysis*, 2018, https://github.com/SenticNet/IARM, accessed in November 2019.

123. Manning, C. D., M. Surdeanu, J. Bauer, J. Finkel, P. Inc, S. J. Bethard and D. Mcclosky, "The Stanford CoreNLP natural language processing toolkit", *In ACL, System Demonstrations*, pp. 55–60, Baltimore, Maryland, USA, 2014.

124. Cheng, Y., *Recursive Neural Network for Sentiment Analysis with PyTorch*, 2018, https://github.com/yc930401/RecNN-pytorch, accessed in 2019.

125. Dong, L., F. Wei, S. Liu, M. Zhou and K. Xu, "A Statistical Parsing Framework for Sentiment Classification", *Comput. Linguist.*, Vol. 41, No. 2, pp. 293–336, Jun. 2015, https://doi.org/10.1162/COLI_a_00221.

126. Aydın, C. R., A. Erkan, T. Güngör and H. Takçı, "Sözlük Tabanlı Kavram Madenciliği: Türkçe için bir Uygulama", *30. Ulusal Bilişim Kurultayı*, pp. 1–6, Ankara, Turkey, Nov. 2013.

127. Aydın, C. R., *An Approach for Dictionary-Based Concept Mining in Turkish*, Master's Thesis, Boğaziçi University, 2013.

128. Aydın, C. R., A. Erkan, T. Güngör and H. Takçı, "Sözlük Kullanarak Türkçe için Kavram Madenciliği Metotları Geliştirme", *Akademik Bilişim Konferansı*, pp. 1–6, Mersin University, Mersin, Turkey, Feb. 2014.

129. Aydın, C. R., A. Erkan, T. Güngör and H. Takçı, "Dictionary-based Concept Mining: An Application for Turkish", *International Conference on Foundations of Computer Science and Technology (CST 2014)*, pp. 1–12, Zürich, Switzerland, Jan. 2014.

130. Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The WEKA Data Mining Software: An Update", *SIGKDD Explor. Newsl.*, Vol. 11, No. 1, pp. 10–18, Nov. 2009, https://doi.org/10.1145/1656274.1656278.

131. Řehůřek, R. and P. Sojka, "Software Framework for Topic Modelling with Large Corpora", *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pp. 45–50, ELRA, Valletta, Malta, May 2010, http://is.muni.cz/publication/884893/en.

132. Linzen, T., "Issues in Evaluating Semantic Spaces Using Word Analogies", *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pp. 13–18, Association for Computational Linguistics, Berlin, Germany, Aug. 2016, https://www.aclweb.org/anthology/W16-2503.

133. Noman, A. A., *Detect E-mail Spam Using Python*, 2013, https://www.codeproject.com/Articles/1232758/Detect-E-mail-Spam-Using-Python, accessed in April 2014.

134. Zeldes, A., D. Das, E. G. Maziero, J. Antonio and M. Iruskieta, "Introduction to Discourse Relation Parsing and Treebanking (DISRPT): 7th Workshop on Rhetorical Structure Theory and Related Formalisms", *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pp. 1–

6, Association for Computational Linguistics, Minneapolis, MN, Jun. 2019, https://www.aclweb.org/anthology/W19-2701.

135. Aydın, C. R., *Sentiment Analysis Framework*, 2020, https://github.com/cemrifki/sentiment-analysis, accessed in July 2020.

136. Aydın, C. R., *Combining Recursive and Recurrent Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations*, 2020, https://github.com/cemrifki/sentiment-recnn-rnn-ensemble-IARM, accessed in July 2020.

137. Aydın, C. R., *Generating Word and Document Embeddings for Sentiment Analysis*, 2019, https://github.com/cemrifki/sentiment-embeddings, accessed in July 2020.

138. Aydın, C. R., *Redefining Context Windows as Subclauses for Word Vector Models*, 2020, https://github.com/cemrifki/redefining-context-windows-as-subclauses-for-word-vector-models, accessed in July 2020.

139. Aydın, C. R., *Aspect-Based Sentiment Analysis in Turkish*, 2020, https://github.com/cemrifki/aspect-based-sentiment-analysis-in-turkish, accessed in July 2020.

140. Boğaziçi University, *Boğaziçi Üniversitesi'nde Türkçe için Yapılan En Detaylı Duygu Analizi Çalışması Tamamlandı*, 2019, https://haberler.boun.edu.tr/en/node/19417, accessed in June 2020.

141. Google Scholar, *Engineering & Computer Science (general) - Google Scholar Metrics*, 2020, https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng_enggeneral, accessed in July 2020.

142. Google Scholar, *Computational Linguistics - Google Scholar Metrics*, 2020,

https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng_computationallinguistics, accessed in July 2020.