UNIVERSIDADE FEDERAL FLUMINENSE

JONNATHAN DOS SANTOS CARVALHO

Exploiting Different Types of Features to Improve Classification Effectiveness in Twitter Sentiment Analysis

NITERÓI

2019

UNIVERSIDADE FEDERAL FLUMINENSE

JONNATHAN DOS SANTOS CARVALHO

Exploiting Different Types of Features to Improve Classification Effectiveness in Twitter Sentiment Analysis

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense, in partial fulfillment of the requirements for the degree of Doctor of Science. Area: Systems and Information Engineering.

Advisor:
ALEXANDRE PLASTINO

NITERÓI

2019

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

C331e Carvalho, Jonnathan dos Santos
Exploiting Different Types of Features to Improve Classification Effectiveness in Twitter Sentiment Analysis / Jonnathan dos Santos Carvalho ; Alexandre Plastino, orientador. Niterói, 2019.
116 f. : il.

Tese (doutorado)-Universidade Federal Fluminense, Niterói, 2019.

DOI: http://dx.doi.org/10.22409/PGC.2019.d.10470309776

1. Mineração de opiniões (Computação). 2. Twitter (Site de relacionamentos). 3. Produção intelectual. I. Plastino, Alexandre, orientador. II. Universidade Federal Fluminense. Instituto de Computação. III. Título.

CDD -

Bibliotecária responsável: Fabiana Menezes Santos da Silva - CRB7/5274

JONNATHAN DOS SANTOS CARVALHO

EXPLOITING DIFFERENT TYPES OF FEATURES TO IMPROVE CLASSIFICATION EFFECTIVENESS IN TWITTER SENTIMENT ANALYSIS

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense, in partial fulfillment of the requirements for the degree of Doctor of Science. Area: Systems and Information Engineering.

Approved in December of 2019.

EXAMINATION BOARD

Prof. Alexandre Plastino - Advisor, UFF

Profa. Aline Marins Paes Carvalho, UFF

Prof. José Viterbo Filho, UFF

Prof. Eduardo Bezerra da Silva, CEFET/RJ

Profa. Kate Cerqueira Revoredo, PPGI/UFRJ

Niterói

2019

Acknowledgements

Meu Deus, eu agradeço a você antes de qualquer outro. Agradeço por ter escolhido as pessoas que entraram na minha vida durante este processo. Agradeço por ser o condutor da minha vida, por me capacitar para a realização desse objetivo e por não permitir que eu perdesse o rumo. Maria Santíssima, minha mãe no céu, também te agradeço por sempre interceder por mim a Deus. "E a quem Deus prometeu, nunca faltou".

Agradeço aos meus pais, Jonas e Edir, por terem sido o meu porto seguro, por me ensinarem com gestos de amor e carinho a como ser um ser humano melhor a cada dia. Cada abraço e cada oração que recebi nesses últimos anos me deram a força necessária para seguir em frente. Sou muito abençoado por ter vocês como pais e inspiração de vida. Agradeço também ao meu irmão, Thiago, por me incentivar e acreditar em mim.

Sobre as pessoas que Deus coloca em nossas vidas, agradeço a Ele por ter me dado como namorada a mulher mais compreensiva e amiga que já conheci. Paula, meu amor, você foi usada por Deus para ser os meus olhos quando o meu olhar desviava do objetivo. A sua fé me fortaleceu e o seu amor me preencheu. "Tu vicias e eu viciei para sempre".

Agradeço também a minha grande amiga, Renatinha, por estar sempre comigo, em qualquer lugar, a qualquer hora, e por sido a minha companhia em tantos momentos de solidão. Levarei você para sempre comigo, onde quer que eu (você) vá.

De forma muito especial e com muito respeito, quero agradecer ao meu querido orientador, Alexandre Plastino, a quem com muito orgulho chamo de Professor. Obrigado por tanto! Você me ensinou muito mais do que um orientador poderia. Obrigado por todas as palavras e por me guiar de forma tão incrível na conclusão da minha pesquisa. Você é um grande amigo!

Por fim, agradeço a todos os meus amigos e familiares pelo carinho e amizade. Agradeço ainda ao Instituto Federal Fluminense pelo suporte, à CAPES pelo apoio financeiro e a todos do Instituto de Computação da UFF por me receber tão bem. Muito obrigado!

Resumo

Análise de sentimentos é a tarefa que identifica automaticamente opiniões expressas em textos, tais como tweets, que são mensagens limitadas a 140 caracteres publicadas no Twitter. Este tipo de mensagem tem sido o foco da análise de sentimentos nos últimos anos, em função da grande quantidade de opiniões expressas, a todo instante, sobre assuntos diversos. Nesse contexto, muitos trabalhos utilizam diferentes métodos para determinar a polaridade dessas opiniões e incluem, em sua maioria, técnicas de aprendizado de máquina supervisionado. Assim, diferentes tipos de atributos têm sido propostos para aumentar o poder preditivo da classificação das opiniões expressas em tweets. Estes atributos incluem n-gramas, *meta-features* e atributos derivados de *word embeddings*.

Com essa grande quantidade de atributos disponíveis, nesta tese, propõe-se investigar a aplicação de diferentes conjuntos de atributos na classificação de opiniões em tweets de domínios distintos. Com esse objetivo, é realizada uma avaliação de cada conjunto de atributos, em vinte e duas bases de dados, para determinar o conjunto de atributos mais relevante na análise de sentimentos em tweets. Além disso, esta tese apresenta um estudo das *meta-features* e de diferentes modelos pré-treinados de *word embeddings* propostos na literatura. Mais precisamente, é proposta a categorização de *meta-features* identificadas na literatura para determinar os tipos de *meta-features* mais adequados para esta tarefa. Ainda, é apresentado um estudo da qualidade de diferentes modelos de *word embeddings*, genéricos e afetivos, na classificação do sentimento expresso em tweets.

Apesar dos diferentes tipos de atributos propostos na literatura, nenhum trabalho avalia de forma detalhada como atributos de diferentes tipos podem se complementar na classificação das opiniões expressas em tweets. Nesse contexto, nesta tese, é apresentado um estudo para avaliar a eficácia da combinação destes diferentes conjuntos de atributos, isto é, n-gramas, *meta-features* e atributos derivados de *word embeddings*, utilizando uma técnica de concatenação de vetores de atributos e métodos de combinação de classificadores ou *ensemble* de classificadores. Para os métodos de *ensemble*, é explorada uma abordagem que adota classificadores base obtidos através de diferentes algoritmos de classificação, cada um utilizando, como entrada, um conjunto específico de atributos.

O estudo conduzido nesta tese indica que todos os tipos de atributos podem contribuir na classificação do sentimento expresso em tweets, incluindo a controversa representação derivada dos n-gramas. Esta representação acarreta na esparsidade das bases de dados, devido à grande quantidade de termos infrequentes. Nesse contexto, é proposta uma estratégia de enriquecimento dos termos contidos nos tweets, que identifica e utiliza termos do vocabulário que possuam alguma relação semântica, com o objetivo de enriquecer a representação esparsa derivada dos n-gramas. Os resultados obtidos demonstram que esta estratégia contribui de forma efetiva na análise de sentimentos em tweets.

Palavras-chave: análise de sentimentos, Twitter, n-gramas, *meta-features*, *word embeddings*, *ensemble* de classificadores, esparsidade

Abstract

Sentiment analysis is the task of automatically determining the opinion expressed on subjective data, such as microblog messages, like tweets. Tweets are short messages sent by Twitter users, limited to 140 characters. This type of message has been the target of sentiment analysis in many recent studies, since they represent a rich source of opinionated texts. Thus, to determine the opinion expressed in tweets, different studies have employed distinct approaches, which mainly include supervised machine learning strategies. For this purpose, a plenty of distinct kinds of features have been engineered in the literature, trying to improve the predictive performance of the sentiment classification of tweets. These features include n-grams, meta-features and word embedding-based features.

Having this large set of features at hand, we investigate whether the classification of tweets from different domains can benefit from those distinct sets of features. To this end, using a collection of twenty-two datasets of tweets, we conduct an experimental evaluation of each feature set, in order to detect which one may provide the core information in Twitter sentiment analysis. Furthermore, we present an underlying study of a large set of meta-features and pre-trained word embedding models developed in the literature over the years. Specifically, we propose to group a rich set of meta-features into different categories, and we evaluate each of these categories to figure out how relevant their features are in the task of Twitter sentiment classification. Also, we present a comparative evaluation of a significant collection of publicly available generic and affective pre-trained word embedding models in the sentiment classification of tweets.

Although many different types of features have been proposed in the literature, none of the state-of-the-art studies have properly exploited how those distinct sets of features may complement each other in Twitter sentiment analysis. In this context, we fill this gap by conducting an assessment study of the combination effectiveness of the different feature sets investigated in this thesis, i.e., n-grams, meta-features, and embedding-based features, using as strategies for combination a feature concatenation approach and ensemble learning methods. For the ensemble methods, we exploit an approach that uses different algorithms as base classifiers, each one using distinct feature sets as input.

As we shall see, all feature sets exploited in this thesis can contribute to the sentiment classification of tweets if properly combined, including the controversial n-gram representation, in which a highly number of infrequent features are derived from, leading to the data sparsity problem. In this regard, we propose an enrichment approach to Twitter sentiment analysis that uses semantically related terms from tweets to increase the knowledge provided by the n-gram features, and we show that this approach contributes to improve the classification effectiveness in the sentiment detection task on tweets.

Keywords: sentiment analysis, Twitter, n-grams, meta-features, word embeddings, ensemble learning, data sparsity

List of Figures

List of Tables

Contents

Chapter 1

Introduction

In recent years, much attention has been given to the content generated by the users of the Web. Since people can communicate their opinions and emotions about any target, such as products, services, and events around the globe, many consumers and companies can make decisions based on these ever-growing opinionated content. However, since a huge amount of opinions are being published every day, manually seeking for these opinions and identifying them as carrying a positive or negative connotation may be impractical. In this context, sentiment analysis or opinion mining is the field of study that analyzes people's opinions, sentiments, appraisals, attitudes, and emotions toward entities and their attributes expressed in written text [53].

Sentiment analysis has been extensively used to automatically determine the overall opinion expressed about different targets in many types of user-generated documents on the Web, such as user reviews, blog comments, news articles, etc. Many companies have taken advantage of the area of sentiment analysis by automatically extracting the opinions expressed by consumers about their products and services, eliminating the need of extensive and expensive researches and facilitating the decision making process.

One of the key challenges in this field concerns the automatic identification of opinions and emotions expressed in short informal texts, such as tweets. Tweets, which are short texts published on Twitter[1], make the task of sentiment analysis very tricky due to their inherent characteristics, such as their informal linguistic style, the presence of misspelling words, and the careless use of grammar [58]. In order to determine the sentiment expressed in this type of message, different approaches have been proposed in the literature. These approaches mainly include supervised machine learning methods and lexicon-based strategies, and they usually focus on the polarity classification of tweets, that is, whether

[1] `http://www.twitter.com`

the sentiment expressed on them carries a positive or negative connotation.

Supervised machine learning methods classify the sentiment of tweets by exploring their contents in order to learn characteristics, commonly referred to as features, that can distinguish the positive tweets from the negative ones. The learning process is accomplished through the knowledge extracted from manually labelled tweets toward their sentiment orientation, i.e., positive or negative, from which a classifier is trained. Using a distinct approach, lexicon-based strategies aim at determining the sentiment expressed in tweets by relying on existing annotated dictionaries or lexicons, in which the sentiment of a tweet is determined from the prior sentiment information of its words or phrases, extracted from lexicon resources.

Regarding supervised learning methods, which are the focus of this thesis, much effort has been made in the literature of Twitter sentiment analysis to achieve an effective and efficient representation of tweets [1, 2, 3, 4, 6, 8, 9, 13, 16, 21, 23, 24, 25, 26, 32, 35, 40, 42, 44, 45, 47, 48, 50, 51, 54, 62, 63, 66, 68, 70, 78, 80, 82, 84, 88, 91, 97, 99]. In this context, distinct types of features have already been proposed, from the simple n-gram-based representation to meta-level features, and word embeddings.

N-grams are the most basic feature representation when dealing with text classification problems, and have motivated the early works on Twitter sentiment analysis [40, 68]. In that case, raw sequences of n words extracted from tweets constitute a sparse and high-dimensional feature space for the classification task. Later, trying to deviate from the sparsity issue, many state-of-the-art studies have proposed different sets of features by developing an abstract representation of tweets, comprising meta-information extracted from their textual content [6]. Those features, also called meta-level features, can capture insightful new information from tweets, regarding their peculiarities. More recently, distributed representations of words generated from deep learning approaches, namely word embeddings, have emerged as an efficient feature representation for text documents [59]. They are currently the main focus of most works on sentiment detection of tweets. Word embeddings encode linguistic patterns of words from a vast corpus of text data and can represent the textual content of tweets in low-dimensional feature vectors.

As far as we know, despite the efforts on designing effective and efficient feature representation in the literature of Twitter sentiment analysis, there is a gap concerning the effect of combining such distinct types of features proposed in state-of-the-art works. In this study, we recognize three main groups of features regarding their structural properties and how they are engineered, such as the n-gram language model, meta-level features,

and word embedding-based features. Each of these groups encloses a rich disjoint set of features that might boost the classification effectiveness if appropriately combined.

Moreover, regarding meta-level features, we have observed that only a small and different fraction of features are employed on each work in the literature. Then, we propose to fill another gap by aggregating meta-level features designed in different works. We believe that combining them into a unique set might benefit the sentiment detection on tweets, as we shall see later. Also, we categorize this aggregated set of meta-level features, putting together features that share similar aspects, to examine whether the sentiment classification of tweets can benefit from different categories of meta-level features.

1.1 Research Questions

The main purpose behind the work proposed in this thesis is to improve the polarity classification effectiveness of the sentiment expressed via tweets. To this end, our main focus is on supervised machine learning methods, since they have been largely and successfully used in Twitter sentiment analysis. In this context, the study presented in this thesis is conducted in order to respond to the following research questions.

– RQ1. Which group of features is the most effective in Twitter sentiment analysis?

Given the large number of features from distinct kinds designed and employed in the literature, such as n-grams, meta-level features, and word embedding-based features, we propose to perform a comparative evaluation of their predictive performances, by using a large collection of datasets of tweets. Our goal is to detect the most powerful feature set in the sentiment classification of tweets from various domains.

It is important to note that an improper choice of a learning algorithm to be used with a specific feature set may degrade the classification performance. As a consequence, it might prevent the classifier from learning how to assign a sentiment label to tweets correctly. In this context, in order to take maximum advantage of the features from each feature set, we leverage the best classifiers constructed for each feature set, instead of comparing them by merely relying on the same learning algorithm. More clearly, we respond to the intermediate question — "*Which classification strategies are the most suitable for each feature set?*", by evaluating distinct supervised learning algorithms for each feature set. After identifying the best classifiers under the individual evaluation of

each feature set, we then conduct a fair comparative evaluation of their predictive power.

As a result of the comparative study among the best classifiers for each feature set, as we shall see, a classifier made up of a concise yet rich set of meta-level features from well-referenced works [1, 6, 13, 16, 25, 26, 40, 42, 48, 50, 51, 63, 70, 88, 99] achieves improved results, which may be a piece of evidence that such feature set plays an essential role in this task. Going further, we propose to categorize this rich set of meta-level features. This categorization is an extension of our previous study [18]. In this thesis, the categories proposed in [18] are revisited, and we include some new meta-level features. In addition to this categorization, we investigate whether the classification of tweets from different domains can benefit from these distinct categories of meta-level features. For this purpose, we evaluate the predictive power of those categories to give a more general understanding of the relevance of the most common meta-level features proposed in the literature.

Lastly, regarding the word embedding-based features, we also present an underlying evaluation of a significant collection of generic and affective pre-trained embedding models that we have identified in the literature, in order to acknowledge the most effective one on the polarity classification of tweets. Pre-trained models are publicly available embedded representations of words, trained with different deep learning methods. While generic pre-trained models comprise word vector trained for general purpose, the affective ones are specifically trained for the sentiment and emotion detection tasks.

– ***RQ2. Can the concatenation of the different features proposed in the literature boost the classification performance in Twitter sentiment analysis?***

We propose to evaluate distinct combinations of the feature sets investigated in this thesis, (i.e., n-grams, meta-level features, and word embedding-based features), considering that features from different groups might complement each other, leading to an improvement in detecting the polarity of tweets. Our goal is to determine which combinations of distinct feature sets may provide the core information in the task of Twitter sentiment analysis. To this end, we adopt a simple feature concatenation approach that aims at combining features from distinct groups into a unique feature vector. We investigate whether the concatenation of all feature sets, as well as pairs of distinct feature sets, can improve the sentiment classification effectiveness.

Furthermore, despite the acknowledged use of SVM due to its robustness on large feature spaces [18, 42, 47, 63], to the best of our knowledge, no study in the literature evaluates the effectiveness of different learning strategies in the presence of features from

different feature sets. We believe that some learning algorithms may be more effective than others when features from distinct natures are put together, depending on their intrinsic properties and how the learning algorithms can deal with them. In this context, we also conduct experiments to identify which classification strategies are the most suitable when combining features from different types.

– ***RQ3. Can the sentiment classification of tweets benefit from the use of ensemble classification strategies having the best classifiers for each feature set as base learners?***

Another approach to combine the discriminative power of different sets of features is through ensemble classification methods. Ensemble methods are learning algorithms that create a set of classifiers, also called base classifiers or base learners, which are used to classify new instances by taking a vote of their predictions [30].

According to Zhang and Duin [98], in practice, there exist two main kinds of ensemble strategies. In the first, the predictions of homogeneous classifiers are combined according to some rule. In the other, heterogeneous classifiers are used. While homogeneous classifiers use the same learning algorithm with different representations of the feature space, the heterogeneous ones apply different classification algorithms to the same input features. In this work, we exploit a hybrid approach to ensemble learning.

Specifically, given the various kinds of features studied in this work, we use different learning algorithms as base classifiers, each one fed with a specific feature representation for the same dataset of tweets (i.e., n-grams, meta-level features, or embedding-based features). For most situations, we show that those classifiers can complement each other in the sentiment detection of tweets, dealing properly with the specificities of the data that might be uncovered by some of them. In addition, we provide an in-depth analysis of the correlation among the base classifiers, showing that there is sufficient diversity among them, which is a necessary condition for ensemble strategies to succeed [30].

The results achieved by evaluating both strategies for combination, i.e., feature concatenation and ensemble learning, show that all feature sets investigated in this thesis can contribute to the sentiment classification of tweets if appropriately combined, including the arguable n-gram-based features. It has already been acknowlegded that the n-gram features, specially when applied to short informal texts, increase the level of data sparsity due to the limited number of characters or words in each message, resulting in a large

number of infrequent terms [75]. In this context, we also investigate how semantically related terms from the vocabulary can be used to enrich the sparse n-gram-based representation of tweets, which motivates the next research question.

– RQ4. Is it possible to use semantically related terms to enrich the sparse representation of tweets and boost the predictive performance of the n-gram-based features?

Intending to enrich the natural sparse representation derived from the n-gram-based features in the sentiment classification of tweets, in which most features have low frequencies, we propose an enrichment approach to Twitter sentiment analysis. This approach uses the existing semantic information of terms in lexicon resources to augment the inherent knowledge of Twitter data, trying to make the tweets more informative to the classifier.

Specifically, our enrichment approach leverages the prior polarity information conveyed by emoticons, as well as the synonymy relation among terms in WordNet [36]. WordNet is a large lexical database of English, in which words are grouped together based on their semantic meanings[2]. The results of the proposed approach show that enriching the n-gram representation of tweets with semantically related terms from the vocabulary improve the overall predictive performance of Twitter sentiment analysis.

1.2 Contributions

In summary, the main contributions of this thesis are:

1. Considering the large amount of heterogeneous features that have already been proposed in the literature of supervised sentiment classification of tweets [1, 2, 3, 4, 6, 8, 9, 13, 16, 21, 23, 24, 25, 26, 32, 35, 40, 42, 44, 45, 47, 48, 50, 51, 54, 62, 63, 66, 68, 70, 78, 80, 82, 84, 88, 91, 97, 99], we present a literature review of the most common features used to represent these short texts, which have been employed in a relevant set of well-referenced works in Twitter Sentiment Analysis, including n-grams, meta-level features, and word embedding-based features.

2. We investigate whether the classification of tweets from different domains can benefit from those distinct sets of features identified in the literature. For this purpose, we

[2] `https://wordnet.princeton.edu`

perform an experimental evaluation of each feature set, using a collection of twenty-two datasets of tweets. To the best of our knowledge, this is the first study that evaluates distinct types of features for a significant number of datasets of tweets, in order to give a more general understanding of the relevance of the most common feature representation in Twitter sentiment analysis. The feature sets examined in this thesis are the popular n-gram language model, an aggregated rich set of meta-level features from the literature, and word embedding representations designed for general purpose and for the sentiment and emotion detection tasks on tweets.

3. Although a plenty of distinct types of meta-level features have emerged in the literature, and even though many of these features share similar characteristics, none of the state-of-the-art studies has organized them into categories. In this context, we propose to group this rich set of meta-level features into different categories, in such a way that features that are similar in structural aspects fall into the same category. In addition to categorizing the meta-level features identified in the literature, we evaluate these categories to figure out how relevant the features from each category are in the task of Twitter sentiment classification.

4. Regarding the increasing number of pre-trained word embedding models developed in the literature, we present an underlying comparative evaluation of a significant collection of publicly available pre-trained embedding models in the sentiment classification of tweets.

5. We address the combination of distinct sets of features proposed in the literature of Twitter sentiment analysis over the years. To this end, we present an assessment study of the combination effectiveness of the different feature sets investigated in this thesis, i.e., n-grams, meta-level features, and word embedding-based features, using as strategies for combination a simple feature concatenation approach and ensemble learning methods. For the ensemble strategies, we exploit a hybrid approach, in which different learning algorithms are used as base classifers, each one using distinct and disjoint feature sets as input.

6. Intending to enrich the feature representation of the naturally sparse Twitter data obtained with the n-gram language model, we propose an enrichment approach as a preprocessing step that precedes the sentiment classification process, and we show that this approach can effectivelly contribute to improve the overall classification effectiveness in Twitter sentiment analysis.

1.3 Thesis Organization

To conduct the work proposed in this thesis, we designed a pipeline of our work under each chapter, as depicted in Figure 1.1, where arrows across chapters mean that an element created in one chapter is used by a method presented in the other.

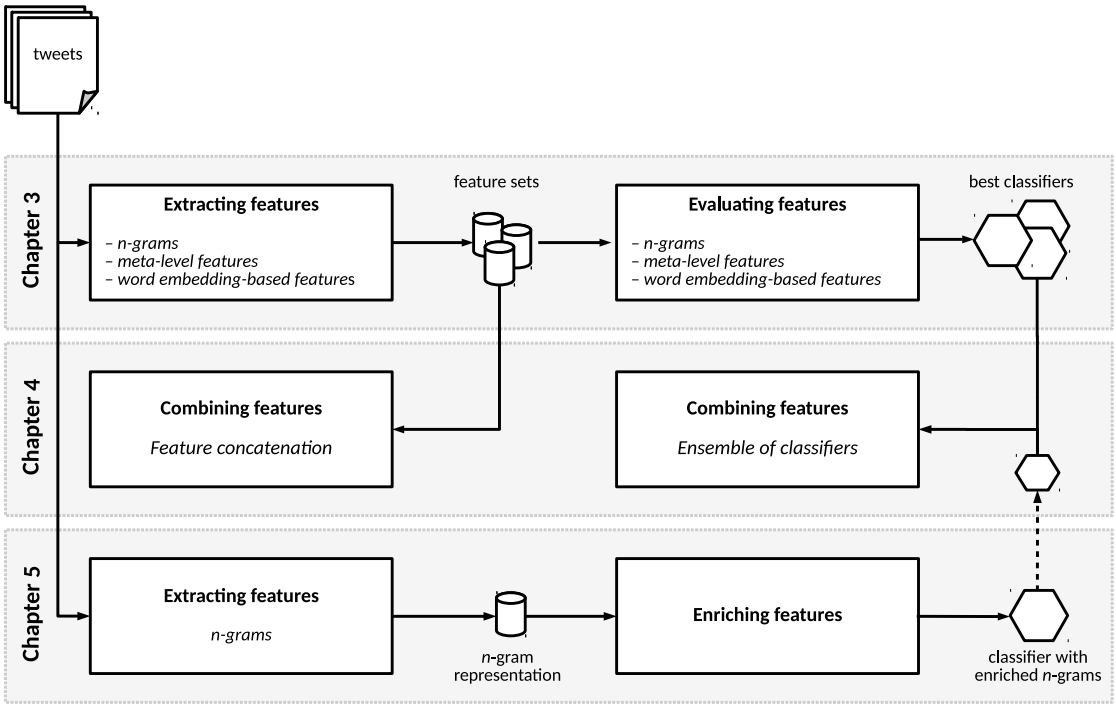

Figure 1.1: Pipeline of the work proposed in this thesis.

In this context, the remainder of this thesis is organized as follows:

- Chapter 2: Identifying Features in Twitter Sentiment Analysis. In this chapter, we describe the most common features identified in a set of well-referenced works in Twitter sentiment analysis, such as the n-gram language model, meta-level features and word embedding-based features. Besides, we propose the categorization of the meta-level features, based on their similar characteristics.

- Chapter 3: Evaluating Features in Twitter Sentiment Analysis. This chapter reports the results of the experimental evaluation of the different feature sets introduced in Chapter 2. Specifically, we evaluate distinct state-of-the-art learning algorithms to identify the best classifiers for each feature set. Also, we present the evaluation of the categories of meta-level features, as well as an underlying evaluation of a relevant set of pre-trained word embeddings from the literature.

- Chapter 4: Combining Features in Twitter Sentiment Analysis. In this chapter, we present the results of the experimental evaluation of different strategies for combination of the features, such as feature concatenation and ensemble learning strategies.

- Chapter 5: An Enrichment Approach to Twitter Sentiment Analysis. In this chapter, we describe and evaluate the enrichment approach to Twitter sentiment analysis proposed in this thesis, which uses the semantic information of terms in existing lexicon resources to increase the knowledge of the naturally sparse n-gram representation. Furthermore, we examine whether the sentiment classification of tweets can benefit from the combination of this enriched set of n-grams with the other feature sets, i.e., meta-level features and embedding-based features.

- Chapter 6: Conclusions and Future Work. This chapter presents the conclusions of the work proposed in this thesis and directions for future research.

Chapter 2

Identifying Features in Twitter Sentiment Analysis

2.1 Introduction

Sentiment analysis has been widely employed to determine the polarity of subjective data, that is, whether the sentiment expressed in opinionated text (movie reviews, blogs, microblogs, etc.) has a positive or negative connotation. For this purpose, different approaches have already been proposed in the literature, which mainly include supervised machine learning methods and lexicon-based strategies.

Regarding supervised machine learning methods, one of the most significant challenges when dealing with text classification problems is related to feature engineering, especially in short texts such as tweets. Among the broad set of features that have emerged in the literature of Twitter sentiment analysis, the n-gram features have been widely employed because of their simplicity in representing tweets [1, 3, 4, 6, 8, 9, 21, 23, 24, 25, 26, 32, 40, 42, 44, 45, 47, 48, 51, 54, 62, 63, 66, 68, 78, 80, 82, 91, 99]. N-gram features are contiguous sequences of n words from a text. Despite their simplicity, it has already been acknowledged that this type of feature may negatively impact the predictive performance of the classification because of the large number of uncommon words in Twitter [75] and because people tend to use much less than the 140-character limit of messages [24]. Indeed, analyzing a corpus of 1.6M tweets, Go et al. [40] have reported that the average length of a tweet is 14 words or 78 characters. Further, in [78], it was observed that 93% of the words in a corpus of 60,000 tweets are highly infrequent, occurring less than ten times. These drawbacks make the data very sparse due to the curse of dimensionality, which can sometimes prevent the classifier to correctly learn how to assign a sentiment label to

unseen tweets.

Beyond the sparsity issue, another factor that makes the sentiment classification even harder is related to the challenging nature of tweets, such as their informal linguistic style and the careless use of grammar [58]. In this context, different studies have explored feature engineering by designing hand-crafted features or meta-level features. Meta-level features are usually extracted from other features and can capture insightful new information about the data [17]. These features include summations and counts of: part-of-speech of words [1, 6, 13, 40, 51, 63], punctuation marks [1, 6, 26, 42, 48, 63], specific characteristics of Twitter and short messages, such as hashtags, user mentions, retweets (RT), abbreviations, etc. [1, 6, 42, 48, 51, 63, 99], emoticons [1, 25, 42, 63], and lexicon features [1, 13, 25, 42, 48, 50, 51, 63, 88], which use the prior sentiment information of words annotated in existing lexicon resources. For example, Mohammad et al. [63] have pointed out the importance of a set of lexicon-based features. In [63], they have designed lexicon-based features such as the total number of positive and negative tokens from a tweet, the overall and the maximal score of a tweet, and the score of the last token of a tweet. All those features were extracted for each of five different sentiment lexicons. The results of the experiments have shown that the most influential features for the two assessed datasets of tweets were the lexicon-based ones, which led to an improvement of 8.5% in terms of the macro-averaged F-score of the positive, negative, and neutral classes.

With the revival and success of deep learning techniques in traditional machine learning applications, distributed representations of words have emerged as a solution to the curse of dimensionality issue [7, 22, 59, 61, 71]. Bengio et al. [7] have discussed two main characteristics of the n-gram model that can lead to misclassification problems: the context and the similarity between words are not taken into consideration. Although some context can be caught by using higher-order n-grams, such as 5-grams, it does not consider contexts farther than n words. Besides that, it makes the dimensionality even higher. To overcome these problems, Collobert et al. [22] introduce a method that relies on largely unlabeled data and uses a multilayer neural network architecture to learn word representations, namely word embeddings. Word embeddings are dense, low-dimensional, and real-valued vectors, each one representing a word in the vocabulary, and encode linguistic patterns that can capture context from a massive corpus of textual data. This method has been successfully applied in many Natural Language Processing (NLP) tasks such as part-of-speech tagging, named entity recognition and semantic role labeling [22].

In the context of sentiment analysis, some works have effectively designed sentiment

and emotion-specific embedding learning methods [2, 35, 84, 97]. For example, Tang et al. [84] have observed that traditional methods for learning word embeddings ignore the sentiment information of text, which can become a problem since words that appear in similar contexts but carrying opposite polarities are mapped into close vectors (for example, *good* and *bad*). In [84], this issue is addressed by extending the method proposed in [22]. Specifically, Tang et al. have developed a sentiment-specific word embedding (SSWE) neural network that incorporates the sentiment information of texts into the embedding learning process, using a corpus of 10M tweets with emoticons as a noisy, distant-supervised training data. In the experiments conducted to evaluate their approach, Tang et al. have shown that the results achieved by the SSWE learning method are competitive with those achieved by the state-of-the-art meta-level features proposed in [63] (84.98% and 84.73% in macro-F1, respectively).

Considering the vast amount of features explored in the literature of supervised sentiment classification of tweets, we have identified three main types of features, which are: n-grams, meta-level features, and word embedding-based features. In the next sections, we describe each of these feature sets. Part of the study presented in this chapter has appeared in [18].

The remainder of this chapter is organized as follows. Section 2.2 introduces the n-gram features. In Section 2.3, we present the meta-level features identified in a set of well-referenced works in Twitter sentiment analysis, as well as the categorization of this rich set of meta-level features. Section 2.4 describes some characteristics of word embedding-based features. Finally, in Section 2.5, we present a summary of this chapter.

2.2 N-gram Features

Different types of features have been engineered and used in Twitter sentiment analysis, from the most common representation, such as n-grams, to meta-level features and word embeddings. N-grams are contiguous sequences of n tokens from a text. The most common representation of textual data is the bag-of-words model [79], in which each word of a tweet is considered as a feature. In general, the feature space is represented by a binary feature vector indicating whether each word of the vocabulary occurs in the tweet or not. In that case, the values 0 and 1 represent the absence and presence of each word in the tweet, respectively [68].

In the simple bag-of-words language model [79], tweets are represented as a sparse

matrix T, where each line constitutes a tweet t_i, and each column contains a word w_j present in the vocabulary of the entire corpus of tweets. Then, each cell T_{ij} represents the occurrence of word w_j in tweet t_i. Equation 2.1 illustrates a matrix T representing a corpus containing 3 tweets (t_1, t_2, and t_3) and a vocabulary of 5 words (w_1, w_2, w_3, w_4, and w_5), where, for example, tweet t_3 contains words w_1 and w_5.

$$\mathbf{T} = \begin{array}{c} \begin{matrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} t_1 \\ t_2 \\ t_3 \end{matrix} \tag{2.1}$$

The bag-of-words model does not consider the order of words in tweets. For that reason, tweets are regarded as bags that contain words. In this context, the n-gram language model tries to establish some order between words. In that case, contiguous sequences of n words from a tweet are used to represent it as features. For example, given the tweet "*This is a great book*", by setting $n \leq 3$, the following unigrams ($n = 1$), bigrams ($n = 2$), and trigrams ($n = 3$) are created: <*This*>, <*is*>, <*a*>, <*great*>, <*book*>, <*This is*>, <*is a*>, <*a great*>, <*great book*>, <*This is a*>, <*is a great*>, and <*a great book*>. As we can notice, although the n-gram model is able to add some context in representing tweets, it makes the sparsity problem even worse as the value of n increases.

In the task of sentiment analysis, Pang et al. [69] are the pioneer authors using n-grams as features to detect the polarity of movie reviews. In the sentiment classification of tweets, Go et al. [40] have used the same approach as in [69] to classify the sentiment expressed in tweets using a method called *distant supervision* to label tweets automatically. Specifically, rather than using hand-labeled tweets, which is time-consuming, this method relies on positive and negative emoticons as labels. If a tweet contains a positive (negative) emoticon, it is regarded as belonging to the positive (negative) class. Since then, n-grams have been one of the most adopted features in supervised learning strategies due to their simplicity in representing tweets [1, 3, 4, 6, 8, 9, 21, 23, 24, 25, 26, 32, 40, 42, 44, 45, 47, 48, 51, 54, 62, 63, 66, 68, 77, 78, 80, 82, 91, 99].

Table 2.1 presents an overview of the n-gram features used in the literature of Twitter sentiment analysis. As shown in this table, most studies in the literature discourage the use of higher-order n-grams, such as 4- and 5-grams, trying to deviate from the sparsity issue.

Table 2.1: Overview of the n-grams features used in the literature of Twitter sentiment classification, ordered by publication year (Year column).

Year	Reference	$n = 1$ bag-of-words	$n = 2$	$n = 3$	$n = 4$	$n = 5$
2009	Go et al. [40]	✓	✓			
2010	Barbosa and Feng [6]	✓				
	Bermingham and Smeaton [8]	✓	✓	✓		
	Bifet et al. [9]	✓				
	Davidov et al. [26]	✓	✓	✓	✓	✓
	Pak and Paroubek [68]	✓	✓	✓		
2011	Agarwal et al. [1]	✓				
	Jiang et al. [48]	✓	✓			
	Kouloumpis et al. [51]	✓	✓			
	Speriosu et al. [82]	✓	✓			
2012	Narr et al. [66]	✓	✓			
	Saif et al. [78]	✓				
	Wang et al. [91]	✓				
2013	Mohammad et al. [63]	✓	✓	✓	✓	
2014	da Silva et al. [25]	✓				
	Saif et al. [77]	✓				
	Tang et al. [83]	✓	✓	✓	✓	
2015	Chikersal et al. [21]	✓	✓	✓		
	Hagen et al. [42]	✓	✓	✓	✓	
	Hamdan et al. [45]	✓	✓			
	Zhang et al. [99]	✓				
2016	Cozza and Petrocchi [23]		✓			
	da Silva et al. [24]	✓	✓	✓		
	Hamdan [44]	✓	✓	✓		
	Lochter et al. [54]	✓				
	Siddiqua et al. [80]	✓				
2017	Araque et al. [3]	✓	✓			
	Jabreel and Moreno [47]	✓	✓	✓	✓	
	Miranda-Jiménez et al. [62]	✓	✓	✓		
2018	Arif et al. [4]	✓	✓			
2019	Emadi and Rahgozar [32]	✓	✓			

2.3 Meta-level Features

Meta-level features, also called hand-crafted features, are usually extracted from other features and can capture insightful new information about the data [17], exploring the content of tweets more efficiently than merely relying on raw sequences of words. In this study, we consider as meta-level features those referred to counts and summations, which are, in general, secondary information extracted from tweets. Meta-level features are referred to hereafter as meta-features.

2.3.1 Categorizing Meta-level Features

In this section, we present and categorize the most common types of meta-features we have examined in a set of well-referenced works in supervised sentiment classification of tweets [1, 6, 13, 16, 25, 26, 40, 42, 48, 50, 51, 63, 70, 88, 99]. This categorization is an extension of the study presented in [18]. In this thesis, the categories proposed in [18] are revisited. For this purpose, considering that features sharing structural aspects should fall into the same group, we have categorized them into five categories, namely: Microblog, Part-of-speech, Surface, Emoticon, and Lexicon-based features. In the following, we describe each of these categories of meta-features.

2.3.1.1 Microblog Features

The Microblog category refers to those features that leverage the syntax and the vocabulary used in tweets and microblog messages, as used in [1, 6, 42, 48, 51, 63, 99]. More specifically, some characteristics of how microblog posts are written may be good indicators of sentiment, such as the use of repeated letters and internet slang present in the vocabulary of this type of text. Furthermore, Twitter-specific tokens, such as user mentions (followed by the special character @), retweets (indicated by RT), URLs, and hashtags (followed by the special character #) have also been explored in the literature.

Twitter hashtags, which are often used as keywords for tweets, are a very informative mechanism. Thus, they may be a good evidence of positive or negative sentiment, as employed in [1, 6, 42, 48, 63, 99]. Similarly, others Twitter-specific tokens are taken as features in the literature, such as the presence of user mentions and retweets [6].

Regarding the 140-character limit of tweets, a very common trick established among Twitter users is the use of word shortcuts and internet slang (for example, “love” becomes “luv”). Another interesting aspect of tweets is the use of repeated letters as intensifiers (for example, in “looooove”). Thus, some works have defined these characteristics as meta-features as well [1, 42, 51, 63].

2.3.1.2 Part-of-Speech Features

Although some studies have already acknowledged that part-of-speech features are not useful for sentiment classification [40, 69], this category of features is still used to determine the sentiment of tweets, in combination with other features [1, 6, 13, 40, 51, 63]. For example, assuming that some adjectives and verbs are good indicators of positive and

negative sentiment, Barbosa and Feng [6] map each word in a tweet to its part-of-speech, being able to identify nouns, verbs, adjectives, adverbs, interjections, and others. Similarly, Agarwal et al. [1] consider the number of adjectives, adverbs, verbs, and nouns as features. In order to capture the informal aspects of tweets, some works [13, 63] use a part-of-speech tagset, presented in [39], to identify some special characteristics of short and noisy texts, such as misspelling words.

2.3.1.3 Surface Features

Surface features capture superficial stylistic content of the tweet, such as the number of words, capitalized words, words with all caps, capital letters, and punctuation marks [1, 6, 26, 42, 48, 51, 63, 70, 83].

Punctuation play an important role in sentiment detection of microblog messages. Thus, punctuation features have also been explored in the literature [1, 6, 26, 42, 48, 63, 70, 83]. The most usual meta-features in this category are the number of exclamation and question marks, as appearing in [1, 6, 26, 42, 48, 70]. Some works have already proposed more sophisticated meta-features, such as the number of contiguous sequences of exclamation and question marks [42, 63, 83], regarding their use in microblog messages to convey intonation. For example, Bermingham and Smeaton [8] observed that the exclamation mark is the most discriminative unigram according to the Information Gain measure, in a corpus of 1,000 tweets labeled as being positive and negative. They also point out that the question mark and sequences of exclamation marks (for example, as "!!!") are in the top 10 most relevant features.

2.3.1.4 Emoticon Features

The polarity of emoticons may also be another relevant characteristic for Twitter sentiment analysis. Since emoticons are used by microblog users to summarize the sentiment they intend to communicate, some works have also extracted meta-features from emoticons, such as the number of positive and negative emoticons in a tweet, as employed in [1, 25, 42, 63, 70, 83].

2.3.1.5 Lexicon-based Features

A different manner of exploring the content of tweets in order to determine the sentiment expressed in them is from using existing sentiment lexical resources or dictionaries in the

literature. These lexicons consist of lists of words with positive and negative terms, such as Bing Liu's opinion lexicon [52], NRC-emotion [64], and OpinionFinder lexicon [94], as well as lexical resources containing words and phrases that are scored on a range of real values, such as AFINN [67], SentiWordNet (SWN) [5], NRC-hashtag [63], and Sentiment140 lexicon [63]. Meta-features of this category have been widely explored in sentiment classification of tweets [1, 13, 16, 25, 42, 48, 50, 51, 63, 83, 88], especially the total count of positive and negative words.

It has already been acknowledged that negation can affect the polarity of an expression [93]. Indeed, the expression *not good* is the opposite of *good*. In this context, an interesting meta-feature proposed in the literature to handle negation is the number of negated contexts [63]. Mohammad et al. [63] have defined a negated context as a segment of a tweet that starts with a negation word, such as *shouldn't*, and ends on the first punctuation mark after the negation word.

Regarding irony, Reyes et al. [74] argue that it represents a meaningful obstacle for determining the polarity of texts accurately. For example, in domains like politics, health campaigns, and natural disasters, Twitter users post ironic messages blaming the government, and most sentiment analysis models cannot deal properly with those messages. To this end, in [74], they proposed features to help capture irony in text, such as the number of counter-factuality words (e.g., *nonetheless*, *nevertheless*) and temporal compression words (e.g., *suddenly*, *now)*, which have been used in Twitter sentiment analysis [16]. As described in [74], while counter-factuality words are discursive terms that hint at contradiction in a text, temporal compression words are focused on identifying elements related to the opposition in time, i.e., words that indicate an abrupt change in a narrative.

An overview of the meta-features and their respective categories are presented in Table 2.2. The number in parentheses right below the name of each category corresponds to the total number of features in that category.

2.4 Word Embedding-based Features

In recent research, with the increasing interest in deep learning approaches for NLP applications, distributed representations of words in a vector space, or word embeddings, have received much attention due to their ability to achieve high performance in many text classification tasks. Although the well-known bag-of-words and n-gram representations have been extensively used regarding their simplicity, they make the feature space highly

Table 2.2: Overview of the meta-features proposed in the literature of Twitter sentiment classification, split by categories. The number of features are presented in parentheses.

Category	Features
Microblog (*10 features*)	⇒ *Whether the tweet has*: retweet, hashtag, user mention, URL, repeated letters, abbreviation, internet slang (7) ⇒ *Number of*: repeated letters, abbreviations, internet slangs (3)
Part-of-speech (*25 features*)	⇒ *Number of*: common noun, proper noun, personal pronoun, common noun + possessive, common noun + verb, proper noun + possessive, proper noun + verb, verb, adjective, adverb, interjection, punctuation, determiner, pre or post-position, conjunction, verb particle, predeterminer, predeterminer + verb, hashtag, emoticon, user mention, discourse marker ("RT" and ":" in retweet), URL or email address, numeral, symbol (25)
Surface (*15 features*)	⇒ *Whether the tweet has*: question mark, exclamation mark (2) ⇒ *Whether last token contains*: question mark, exclamation mark (2) ⇒ *Number of*: words, capitalized words, words with all letters capitalized, capital letters, punctuation marks, question marks, exclamation marks, sequence of question marks, sequence of exclamation marks, sequence of both question and exclamation marks (10) ⇒ *Average number of characters in words* (1)
Emoticon (*10 features*)	⇒ *Whether the tweet has*: emoticon, positive emoticon, negative emoticon (3) ⇒ *Whether the last token is*: positive emoticon, negative emoticon (2) ⇒ *Number of*: emoticons, positive emoticons, negative emoticons, extremely positive emoticons, extremely negative emoticons (5)
Lexicon-based (*70 features*)	⇒ *Number of*: positive adjectives, negative adjectives, positive nouns, negative nouns, positive adverbs, negative adverbs, positive verbs, negative verbs, negated contexts, negation words, intensifier words, counter factuality words, temporal compression words (13) ⇒ $\sum$ *scores of the adjectives, adverbs, verbs, and nouns* (1) ⇒ *For each sentiment lexicon* (AFINN, Bing Liu's lexicon, NRC-emotion, NRC-hashtag, OpinionFinder, Sentiment140, and SentiWordNet): (56) – *Number of*: positive words, negative words (2×7 *lexicons*) – *Total score of*: positive words, negative words (2×7 *lexicons*) – *Maximal score of*: positive words, negative words (2×7 *lexicons*) – *Balance score of the tweet* (1×7 *lexicons*) – *Score of the last token* (1×7 *lexicons*)

dimensional leading to the curse of dimensionality, as discussed in Section 2.2. On the other hand, word embeddings can capture the semantic and syntactic relations between words from a large amount of unlabeled text data, representing them in dense real-valued vectors that can be used as features in supervised machine learning frameworks. As described in [7], the feature vectors associated with each word are learned from large corpora, and each value represents a different aspect, or dimension, of the word. The main idea is that words that frequently occur together in the same contexts are mapped to similar regions of the vector space [2].

In [59], Mikolov et al. have designed the word2vec tool (w2v), comprising the Continuous Bag-Of-Words (CBOW) and the Skip-gram models, which are three-layer neural networks to train word embeddings. More specifically, given a massive text corpus, these architectures learn vector representation of words based on its vocabulary. As described in [59], the CBOW method predicts the source word based on its context, while the Skip-gram predicts nearby words given a source word. Later, in [61], Mikolov et al. have improved the Skip-gram model, making it much more computationally efficient. In [61], they have used an internal dataset of news articles from Google with one billion words to train the model, generating a 300-dimensional word vector.

Figure 2.1 illustrates the CBOW and Skip-gram architectures [59]. In the Skip-gram architecture, given a massive text corpus with vocabulary V and a target word $w(t)$, a neural network is trained on V to predict nearby words from $w(t)$ in a window of size c. More precisely, $w(t)$ is the central word and the network predicts the surrounding context of it, i.e., words that occur before and after $w(t)$ in V. For example, considering a window of size $c = 2$, the network will try to predict the words $w(t-2)$, $w(t-1)$, $w(t+1)$, and $w(t+2)$, i.e., 2 words before and 2 words after $w(t)$. Conversely, the CBOW neural architecture predicts a central target word $w(t)$ according to its surrounding words of window size c, i.e., $w(t-c), ..., w(t-1), w(t+1), ..., w(t+c)$.

For both CBOW and Skip-gram models, the process is executed for each word $w(t)$ present in vocabulary V. At the end, after the model is trained, the weights of the neural network are used as the embedding representation of word $w(t)$. As a consequence, words that appear in the same context tend to have similar representations.

Pennington et al. [71] argue that the statistics of the words in a given training corpus are sub utilized by the Skip-gram model [61] since it does not take into account global co-occurrence counts of words. For that reason, they propose a weighted least squares model, namely GloVe (Global Vectors), that leverages global word-word co-occurrence

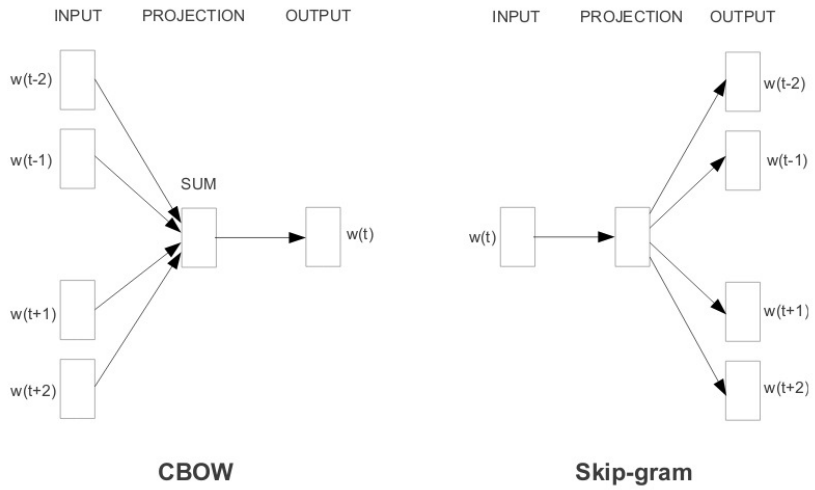

Figure 2.1: CBOW and Skip-gram architectures (Source: Mikolov et al. [59]).

counts in the word embedding training phase. They have trained a 300-dimensional word vector and evaluated the proposed model on the word analogy, word similarity, and named entity recognition tasks, proving that GloVe outperforms the w2v models (CBOW and Skip-gram) by a significant margin.

Most techniques to train word vectors ignore the internal structure of words, making it difficult to learn good representations for morphologically rich languages, which have many different inflected forms for the same word. Thus, Bojanowski et al. [10] have proposed the fastText model, which learn representations for character n-grams as an extension of the Skip-gram model [61]. Later, Mikolov et al. [60] have combined some preprocessing strategies rarely used together to improve the standard fastText model and achieved state-of-the-art results on several tasks.

Reasoning the inefficiency of traditional approaches to train word embeddings for sentiment analysis, some authors have designed solutions to train word vectors specifically for the sentiment analysis task [2, 35, 84, 97]. Tang et al. [84] developed a neural network to learn sentiment-specific word embeddings (SSWE) on a massive corpus of tweets. They used the SSWE word vectors as features in a supervised machine learning strategy and reported comparable results with those achieved by applying the meta-level features proposed in [63].

Felbo et al. [35] took advantage of the vast amount of emoji occurrences on tweets to train models with rich emotional representations by using a transfer learning approach, namely DeepMoji. They have evaluated the DeepMoji model on eight benchmark datasets for the emotion, sarcasm, and sentiment classification tasks and their results outperformed

state-of-the-art results for all assessed datasets, including the results achieved with the SSWE [84] method.

In [97], Xu et al. proposed Emo2Vec, which is a multi-task training framework that incorporates six different emotion-related tasks in the training process, such as sentiment analysis, emotion classification, sarcasm detection, abusive language classification, stress detection, insult classification, and personality recognition. They argue that including the affective information from all those domains may benefit the learning process, thus enabling the creation of a more general embedding emotional space. Compared with the SSWE and DeepMoji models, the Emo2Vec word vectors achieved competitive results. Also, claiming that Emo2Vec is weak on capturing the syntactic and semantic meaning of words, they concatenated Emo2Vec with the pre-trained GloVe [71] vectors for comparison with state-of-the-art results on 14 datasets from distinct domains. In the experimental evaluation, the combination of Emo2Vec with GloVe vectors as input to an LR classifier achieved comparable performance to state-of-the-art results for some datasets.

Discussing the challenges of the emotion classification problem, Agrawal et al. [2] address some limitations of this task by leveraging noisy training data with a large range of emotions to learn emotion-enriched word representations, namely Emotion Word Embeddings (EWE). Instead of tweets, they have explored product reviews, as this type of text may generalize better for other domains. They have evaluated the predictive performance of EWE against state-of-the-art pre-trained word vectors [35, 61, 71, 84] on four datasets from various domains, such as fairy tales, blogs, experiences, and tweets. To this end, they have used LR and SVM as the learning strategies showing that the proposed method outperformed all the other methods with a statistically significant difference.

It has already been acknowledged that achieving suitable and sufficient representations of words depends on the volume of data used to train the word embedding models. Much effort in recent research is mainly focused on scalability issues of existing methods. For that reason, many researchers make the word vectors trained with their architectures available for public use. Those publicly available word vectors are referred to as pre-trained word embeddings.

Table 2.3 presents the characteristics of the pre-trained word embeddings generated by the methods discussed in this section. The $|D|$ and $|V|$ columns refer to the dimension and vocabulary sizes of each pre-trained embedding, respectively. The Type column separates the word embeddings trained for general purpose (generic) from those specially trained for the sentiment analysis and emotion detection tasks (affective). Additionally, under

the Corpus column, we present information about the textual corpora used to train the embeddings.

Table 2.3: Characteristics of the pre-trained word embeddings separated by type and ordered by the number of dimensions ($|D|$ column).

Type	**Embedding**	$\|D\|$	$\|V\|$	**Corpus**
Generic	GloVe-TW [71]	200	1.2M	Twitter (27B tokens)
	GloVe-WP [71]	300	400K	Wikipedia+Gigaword (6B tokens)
	fastText [60]	300	1M	Wikipedia+news+UMBC text corpus (16B tokens)
	w2v-GN [61]	300	3M	Google news (100B tokens)
	w2v-Edin [11]	400	259K	Twitter (10M tweets)
	w2v-Araque [3]	500	57K	Twitter (1.28M tweets)
Affective	SSWE [84]	50	137K	Twitter (10M tweets)
	Emo2Vec [97]	100	1.2M	Twitter (1.9M tweets)
	DeepMoji [35]	256	50K	Twitter (1B tweets)
	EWE [2]	300	183K	Amazon reviews (200K reviews)

As described in Table 2.3, the GloVe-TW and GloVe-WP word vectors [71] were trained on massive text corpora from Twitter and Wikipedia + Gigaword, respectively. The fastText vectors [60] were trained on rich and vast sources of data, including Wikipedia, news from statmt.org, and the UMBC text corpus.

Regarding the word vectors trained with the word2vec tool, w2v-GN is the former one whose construction is detailed in [61]. Bravo-Marquez et al. [11] have used the Skip-gram method implemented in the word2vec tool to train word vectors on a vast corpus of ten million tweets from the Edinburgh Twitter corpus [72]. In [11], they have optimized the parameters for classifying words into emotions and made the pre-trained vectors publicly available (w2v-Edin). More recently, Araque et al. [3] developed a supervised learning system using word vectors as features. The w2v-Araque vectors were trained on a corpus of 1,280,000 tweets with the word2vec tool, and the system was used as a baseline to compare it to other approaches.

Regarding the affective pre-trained vectors, which leverage the sentiment or emotion information during the training phase, the SSWE [84], Emo2Vec [97], and DeepMoji [35] word vectors were trained on tweets, while the EWE [2] representations were trained on product reviews from Amazon. All of them were generated using specific methods for creating word representations to incorporate the sentiment information of texts during the training process.

2.5 Summary

In this chapter, as one of the contributions of this thesis, we presented a literature review of the most common feature representation in the sentiment classification of tweets. We have identified three main sets of features: n-grams, meta-level features, and word embedding-based features. As another contribution, we proposed to group an aggregated rich set of meta-level features from different works into different categories, namely, Microblog, Part-of-speech, Surface, Emoticon and Lexicon-based.

The next chapter presents the experimental evaluation of the feature sets described in this chapter. Besides, we examine the predictive power of the categories of meta-level features, as well as the classification effectiveness of a relevant set of pre-trained word embedding models designed and used in the literature of Twitter sentiment analysis.

Chapter 3

Evaluating Features in Twitter Sentiment Analysis

3.1 Introduction

This chapter presents the computational results obtained by evaluating the different feature sets introduced in Chapter 2, namely n-grams, meta-level features, and word embeddings, respectively. Specifically, we aim at responding to research question RQ1, as discussed in Chapter 1.

Despite the acknowledged use of SVM due to its robustness on large feature spaces [18, 42, 47, 63], we believe that different classification strategies may benefit from the use of an appropriate set of features. For example, an SVM classifier fed with n-grams may be successful in this task, but a successful result may not be obtained if we use the same feature representation with another inducer, such as Random Forest. We investigate this hypothesis by evaluating the predictive power of features from distinct kinds, feeding them to different state-of-the-art learning algorithms.

Besides the individual evaluation of each feature set, we present the evaluation of the categories of meta-features proposed in Chapter 2, as well as an underlying evaluation of pre-trained embedding models adopted in the literature of Twitter Sentiment Analysis.

The remainder of this chapter is organized as follows. We start by describing, in Section 3.2, the experimental protocol we followed. Then, in Section 3.3, we report and discuss the results of a set of experiments to respond to research question RQ1, introduced in Chapter 1. Finally, in Section 3.4, we present a summary of the chapter.

3.2 Experimental Setup

Attending to answer the research question RQ1, presented in Chapter 1, we adopted Weka's [43] implementation of the machine learning algorithms Support Vector Machines (SVM), L2-regularized Logistic Regression (LR), and Random Forests (RF). Regarding SVM and LR, we used the LIBSVM[1] [19] and LIBLINEAR[2] [33] implementations, respectively. We set the regularization parameter to its default value ($C = 1.0$), and we employed the linear kernel for LIBSVM. Furthermore, for RF, we used the default number of trees to be generated (i.e., *number of iterations* = 100).

We used a set of twenty-two datasets in the computational experiments reported in this section. These datasets have been extensively used in the literature of Twitter sentiment analysis. To the best of our knowledge, this is the first study using a significant number of datasets of tweets in the evaluation of different types of features that have been employed in the literature. These datasets are: irony [41], sarcasm [41], aisopos[3], SemEval-Fig[4], sentiment140 [40], person [20], hobbit [54], iphone6 [54], movie [20], sanders[5], Narr [66], archeage [54], Obama-McCain Debate (OMD) [29], Health Care Reform (HCR) [82], STS-gold [76], SentiStrength [85], Target-dependent [31], Vader [46], SemEval13[6], SemEval16[7], SemEval17[8], and SemEval18[9]. Some characteristics of these datasets are presented in Table 3.1, namely their total number of tweets, positive tweets, and negative tweets.

In the experimental evaluation, the predictive performance of the sentiment classification is measured in terms of Accuracy and weighted average F-measure. For each evaluated dataset, the classification accuracy was computed as the ratio between the number of correctly classified tweets and the total number of tweets, after a 10-fold cross-validation. The weighted average F-measure[10], F^{AVG}, was computed as shown in Equation 3.1.

$$F^{AVG} = \frac{(F^P \times \#positive\ tweets) + (F^N \times \#negative\ tweets)}{total\ number\ of\ tweets} \tag{3.1}$$

[1] Available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
[2] Available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear`
[3] `http://grid.ece.ntua.gr`
[4] `http://alt.qcri.org/semeval2015/task11`
[5] `http://www.sananalytics.com/lab/twitter-sentiment`
[6] `https://www.cs.york.ac.uk/semeval-2013/task2.html`
[7] `http://alt.qcri.org/semeval2016/task4/`
[8] `http://alt.qcri.org/semeval2017/task4/`
[9] `https://competitions.codalab.org/competitions/17751`
[10] As computed in Weka environment.

Table 3.1: Characteristics of the datasets of tweets, ordered by size (*#tweets* column).

Dataset	**#tweets**	**#positive**	**#negative**
irony	65	22	43
sarcasm	71	33	38
aisopos	278	159	119
SemEval-Fig	321	47	274
sentiment140	359	182	177
person	439	312	127
hobbit	522	354	168
iphone6	532	371	161
movie	561	460	101
sanders	1,224	570	654
Narr	1,227	739	488
archeage	1,718	724	994
SemEval18	1,859	865	994
OMD	1,906	710	1,196
HCR	1,908	539	1,369
STS-gold	2,034	632	1,402
SentiStrength	2,289	1,340	949
Target-dependent	3,467	1,734	1,733
Vader	4,196	2,897	1,299
SemEval13	4,378	3,183	1,195
SemEval17	6,347	2,375	3,972
SemEval16	12,216	8,893	3,323

In Equation 3.1, F^P is the F-measure for the *positive* class, as follows:

$$F^P = \frac{2 \times precision^P \times recall^P}{precision^P + recall^P}, \tag{3.2}$$

$$\text{where } precision^P = \frac{PP}{PP + PN}, \tag{3.3}$$

$$\text{and } recall^P = \frac{PP}{PP + NP}. \tag{3.4}$$

Analogously, the F-measure for the *negative* class, F^N, is computed as follows:

$$F^N = \frac{2 \times precision^N \times recall^N}{precision^N + recall^N}, \tag{3.5}$$

$$\text{where } precision^N = \frac{NN}{NN + NP}, \tag{3.6}$$

$$\text{and } recall^N = \frac{NN}{NN + PN}. \tag{3.7}$$

In Equations 3.3, 3.4, 3.6, and 3.7, PP, PN, NP, and NN are the cells of the confusion matrix, as shown in Table 3.2.

Table 3.2: Confusion matrix for the polarity classification of tweets.

		Actual class	
		Positive	Negative
Predicted class	Positive	PP	PN
	Negative	NP	NN

Moreover, as suggested by Demšar [28], we ran the Friedman test followed by the Nemenyi post-hoc test to determine whether the differences among the results are statistically significant at a 0.05 significance level. Whenever applicable, we present the results of the statistical tests right below each table of results. For this purpose, we use the symbol $\succ$ to show that some classifier x is significantly better than some classifier y, so that $\{x\} \succ \{y\}$.

3.3 Responding to Research Question RQ1

The experiments conducted in this section aim at responding to research question RQ1, as follows:

– *RQ1. Which group of features is the most effective in Twitter sentiment analysis?*

To answer this question, first, we answer the intermediate question — "*Which classification strategies are the most suitable for each feature set?*" throughout Subsections 3.3.1, 3.3.2, and 3.3.3, by assessing the distinct feature sets we have identified in the literature. Those features include n-grams, meta-features, and word embeddings. Then, after determining the best classifier for each set of features, we perform a comparison among them to determine the most representative one in the Twitter sentiment analysis task. The discussion on this comparison is presented in Subsection 3.3.4.

Besides the comparative evaluation of the feature sets, in Subsection 3.3.2, we present the evaluation of the categories of meta-features introduced in Chapter 2 (Section 2.3), as well as an assessment study of a significant collection of pre-trained embedding models, in Subsection 3.3.3.

3.3.1 Effectiveness of N-gram Features

The n-gram features used in the computational experiments reported in this section are unigrams, bigrams, and trigrams. We do not explore higher-order n-grams trying to minimize the negative effect of the curse of dimensionality. Besides, unigrams, bigrams, and trigrams are the most adopted n-gram features in the literature of sentiment detection in tweets, as we have shown in Table 2.1.

As a preprocessing step, we used the same strategy as done in [63]. Each tweet was tokenized and labeled according to their part-of-speech tag, using the Twitter-specific part-of-speech tag set tool [39]. This tag set consists of twenty-five part-of-speech tags, specifically designed for tweets, that takes into account the different aspects that tweets have as compared to regular text. Next, for each tweet in a given dataset, we replaced URLs by the token *someurl* and user mentions by the token *someuser*. Regarding stopwords removal, we discarded stopwords only as unigrams, since it has been acknowledged that stopwords can affect the polarity of some expressions in higher-order n-grams [82]. Finally, considering that negation words[11] (*shouldn't*, for example) can affect the n-gram-based features, we handled negation by employing the same approach as used by Mohammad et al. [63]. In [63], negated contexts change n-gram-based features. Specifically, they add the tag _NEG on each token into a negated context. In other words, in a negated context, Mohammad et al. concatenate the tag _NEG to every token between the negation word and the first punctuation mark after it. For example, in the sentence "*He isn't a great book writer, but I read his books*", the unigrams *great*, *book*, and *writer* become *great_NEG*, *book_NEG*, and *writer_NEG*, respectively.

After preprocessing all tweets, the feature space is represented by a binary feature vector indicating whether each n-gram existing in the vocabulary occurs in the tweet or not. In that case, the values 0 and 1 represent the absence and presence of each n-gram in the tweet, respectively.

Table 3.3 shows the results of the evaluation of the n-gram features in terms of Accuracy (%) and average F-measure (%), as well as the number of features, i.e., the number of n-grams, extracted for each dataset (#features column). The bold-faced values indicate the best results, and the total number of wins for each classifier is presented in the #wins row. Also, we compute a ranking to make a fair comparison among the results. Precisely, for each dataset, we assign scores from 1.0 to 3.0 for each tested strategy (each column), in

[11] We used the negation words available at `http://sentiment.christopherpotts.net/lingstruc.html#negation`

ascending order of Accuracy or F-measure, where the score 1.0 is assigned to the strategy with higher Accuracy or F-measure. Thus, low score values indicate better results. Finally, we sum the assigned scores for each classifier, as shown in the rank sums row.

As we can observe in Table 3.3, the best results were achieved by SVM in 12 out of the 22 datasets in terms of Accuracy, and in 14 out of the 22 datasets with respect to the F-measure metric. Indeed, SVM has proven its robustness on large feature spaces in Twitter sentiment analysis [63]. The LR classifier achieved the second-best results for both Accuracy and F-measure metrics. Conversely, the worse performance was achieved by the RF classifier. The poor performance of RF may be due to the sparse nature of the data, in which most feature values are zero, increasing the risk of randomly choosing a subset of uninformative features when splitting the data at a decision node in the trees.

Table 3.3: Accuracies and F-measure scores (%) achieved by evaluating the n-gram features using SVM, LR, and RF classifiers, respectively.

Dataset	**#features**	***Accuracy***			***F-measure***								
					average			***positive***			***negative***		
		SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	1.8K	66.2	66.2	66.2	52.7	52.7	52.7	0.0	0.0	0.0	79.6	79.6	79.6
sarcasm	1.8K	50.7	**52.1**	46.5	48.0	**52.2**	46.5	**58.8**	51.4	44.1	38.6	**52.8**	48.6
aisopos	6.5K	**87.8**	87.4	72.7	**87.4**	87.0	68.9	**90.3**	90.0	80.7	**83.5**	82.9	53.1
SemEval-Fig	8.8K	**91.0**	90.0	85.4	**89.8**	88.2	78.6	**60.3**	51.5	0.0	**94.9**	94.4	92.1
sentiment140	7.6K	84.1	**84.4**	83.0	84.1	**84.4**	82.9	85.0	**85.3**	84.3	83.2	**83.4**	81.5
person	10.0K	79.0	**79.5**	71.8	**77.8**	77.3	61.0	86.1	**86.8**	83.4	**57.4**	54.1	6.1
hobbit	8.5K	92.9	**93.3**	87.5	93.0	**93.3**	86.7	94.7	**95.0**	91.5	89.3	**89.7**	76.7
iphone6	9.4K	77.6	**78.0**	77.4	73.6	**73.9**	72.3	86.0	**86.2**	86.1	45.2	**45.6**	40.6
movie	10.2K	**84.1**	83.2	82.0	**80.2**	77.7	73.9	**91.0**	90.7	90.1	**31.0**	19.0	0.0
sanders	23.6K	**83.0**	81.6	73.1	**83.0**	81.6	72.3	**82.4**	80.0	76.5	**83.6**	83.0	68.6
Narr	24.2K	**83.7**	82.6	73.7	**83.7**	82.4	70.5	**86.6**	86.0	81.6	**79.2**	76.8	53.7
archeage	28.2K	**86.3**	85.9	82.8	**86.4**	85.9	82.9	**84.3**	83.3	82.0	**87.9**	87.8	83.6
SemEval18	42.0K	**80.2**	79.2	71.9	**79.9**	78.8	69.4	**76.6**	74.8	58.7	**82.8**	82.3	78.7
OMD	32.1K	81.2	**82.4**	77.5	81.0	**81.9**	77.7	**74.0**	73.6	71.3	85.2	**86.8**	81.5
HCR	40.5K	79.1	**79.5**	76.7	**77.7**	77.3	70.4	**55.9**	52.9	31.0	86.3	**86.9**	86.0
STS-gold	37.4K	**84.0**	83.6	74.6	**83.4**	82.5	68.3	**71.3**	67.9	32.8	88.9	**89.0**	84.4
SentiStrength	49.4K	**73.2**	72.4	64.1	**72.7**	71.5	56.0	**78.5**	78.4	76.2	**64.4**	61.8	27.4
Target-dependent	66.6K	81.4	**82.0**	78.8	81.4	**82.0**	78.7	81.3	**81.8**	79.9	81.5	**82.2**	77.5
Vader	68.4K	**84.8**	83.3	75.5	**83.9**	81.9	69.6	**89.7**	88.9	84.9	**71.0**	66.2	35.6
SemEval13	105.0K	**81.0**	79.9	74.1	**78.9**	76.8	64.5	**88.0**	87.5	84.8	**54.5**	48.1	10.4
SemEval17	127.6K	86.9	**87.1**	84.5	86.8	**87.0**	84.4	**82.1**	81.7	78.7	89.6	**90.1**	87.8
SemEval16	252.1K	**85.8**	85.0	74.0	**85.1**	84.0	64.2	**90.7**	90.3	74.8	**70.2**	67.2	35.9
#wins		**12**	9	0	**14**	7	0	**16**	5	0	**12**	9	0
rank sums		**31.5**	34.5	64.5	**29.5**	36.5	64.5	**28.5**	38.5	63.5	**32.5**	34.5	63.5
		{SVM, LR} $\succ$ {RF}						{SVM, LR} $\succ$ {RF}					

Another point we can highlight is that the n-gram model does not seem to be a good choice for representing the tweets from datasets irony and sarcasm. This can be justified by the few numbers of tweets these datasets contain, that is, 65 and 71, respectively. The n-gram-based features may not be representative enough in the sentiment classification of the tweets from these datasets since the classification is performed based on the vocabulary extracted from the training set, that is, the n-grams themselves. Finally, the Friedman test followed by the Nemenyi post-hoc test detected that both SVM and LR are significantly

better than RF for this kind of feature, but there is no significant difference between them, for both Accuracy and F-measure.

3.3.2 Effectiveness of Meta-level Features

In this section, we present an assessment study of the meta-features in two parts. First, we show and compare the predictive performance of SVM, LR, and RF by using the full set of meta-features, in order to recognize the most appropriate classification algorithm when this type of feature is exploited. Then, we evaluate each category of meta-features to identify the most effective one in the sentiment classification of tweets. The meta-features evaluated in this section are those categorized in Chapter 2 (See Table 2.2 for details).

To determine the polarity of adjectives, nouns, adverbs, and verbs, we used the SentiWordNet sentiment lexicon [5]. Similarly, to identify internet slang and emoticons, we used the internet slang and emoticon lists introduced in [1]. For abbreviations, we adopted the Internet Lingo Dictionary [92], as done in [51].

The results of the first experiment are reported in Table 3.4. We can notice that the RF classifier performed significantly better than SVM and LR in terms of Accuracy, achieving the best results in 16 out of the 22 datasets. Although RF might not be a good choice on sparse feature spaces, it is robust to outliers, noise, and can handle class imbalance [14]. Those characteristics may have led to an improvement in classification accuracy, as compared to SVM and LR.

Regarding the results achieved in terms of F-measure, RF outperformed SVM and LR in 14 out of the 22 datasets. The Friedman and Nemenyi tests indicated that RF performed significantly better than SVM, but there was no statistically significant differences between RF and LR. Nevertheless, the RF classifier obtained the smallest rank sum (see rank sums row), which means that it has achieved the best overall results.

In general, for both Accuracy and F-measure, SVM and LR achieved comparable performances. Although LR performed slightly better than SVM, as shown in the rank sums row, there was no significant difference between them.

3.3.2.1 Categories of Meta-level Features

The second part of the experiments reported in this section consists of determining the most predictive categories of meta-features, following the categorization proposed in Chapter 2 (Section 2.3).

Table 3.4: Accuracies and F-measure scores (%) achieved by evaluating the meta-features using SVM, LR, and RF classifiers, respectively.

Dataset	*Accuracy*			*F-measure*								
				average			*positive*			*negative*		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	76.9	78.5	**81.5**	76.8	77.5	**80.7**	65.1	63.2	**68.4**	82.8	84.8	**87.0**
sarcasm	71.8	69.0	**80.3**	71.8	68.6	**80.1**	69.7	63.3	**77.4**	73.7	73.2	**82.5**
aisopos	**94.2**	93.5	92.8	**94.2**	93.5	92.8	**95.1**	94.5	93.8	**93.0**	92.2	91.4
SemEval-Fig	88.5	90.0	**90.3**	87.7	**88.5**	88.2	**54.3**	**54.3**	50.8	93.4	94.4	**94.6**
sentiment140	85.2	**85.5**	85.0	85.2	**85.5**	85.0	85.5	**86.0**	85.0	**85.0**	**85.0**	84.9
person	82.2	82.5	**83.6**	82.1	82.1	**82.8**	87.7	87.9	**89.0**	**68.3**	67.8	67.6
hobbit	88.9	89.5	**91.6**	88.9	89.4	**91.6**	91.8	92.3	**93.8**	82.9	83.5	**87.0**
iphone6	80.5	81.4	**82.5**	80.0	80.7	**82.2**	86.4	87.3	**87.8**	65.1	65.5	**69.3**
movie	85.6	86.5	**87.0**	84.3	85.1	**85.2**	91.5	92.1	**92.5**	51.5	**53.1**	52.3
sanders	81.0	80.9	**84.8**	81.0	80.9	**84.7**	79.7	79.7	**83.1**	82.1	82.0	**86.2**
Narr	89.6	89.5	**90.3**	89.6	89.5	**90.3**	91.4	91.3	**92.0**	86.9	86.7	**87.7**
archeage	84.6	**85.4**	**85.4**	84.6	**85.4**	85.3	81.6	**82.5**	81.8	86.7	87.6	**87.8**
SemEval18	85.6	85.2	**86.0**	85.6	85.1	**86.0**	84.2	83.7	**84.6**	86.7	86.4	**87.2**
OMD	78.1	78.2	**79.8**	77.8	78.0	**79.2**	69.2	69.3	**69.5**	83.0	83.1	**84.9**
HCR	75.8	76.0	**77.5**	72.9	73.4	**74.7**	42.9	44.5	**46.4**	84.7	84.7	**85.8**
STS-gold	92.2	91.8	**93.1**	92.2	91.8	**93.1**	87.3	86.6	**88.7**	94.4	94.1	**95.0**
SentiStrength	83.2	**83.6**	83.3	83.2	**83.5**	83.3	85.7	**86.2**	85.8	79.5	**79.7**	**79.7**
Target-dependent	**83.3**	82.9	83.1	**83.3**	82.9	83.1	**83.2**	82.8	82.9	**83.3**	82.9	**83.3**
Vader	**93.3**	93.2	93.0	**93.2**	**93.2**	92.9	**95.2**	**95.2**	95.0	**88.7**	**88.7**	88.2
SemEval13	86.4	86.7	**86.9**	86.2	86.4	**86.5**	90.8	91.0	**91.2**	73.7	**74.2**	74.1
SemEval17	86.4	86.3	**86.5**	86.3	86.3	**86.4**	**81.4**	**81.4**	81.3	89.2	89.2	**89.4**
SemEval16	**85.6**	85.3	85.4	**85.3**	85.0	84.9	**90.4**	90.2	90.3	**71.8**	70.9	70.7
#wins	4	3	**16**	4	5	**14**	6	6	**13**	6	5	**15**
rank sums	51.0	49.5	**31.5**	51.5	46.5	**34.0**	49.0	47.0	**36.0**	49.5	47.5	**35.0**
	{RF} $\succ$ {SVM, LR}						{RF} $\succ$ {SVM}					

Table 3.5 and Table 3.6 present the accuracies and F-measure scores achieved by evaluating each category of meta-features (MIC, POS, SUR, EMO, and LEX columns) using the RF classifier, and their comparison with the results achieved by using the set of all meta-features (ALL column). We used the RF classifier since it has achieved better results than SVM and LR in the previous experiment. The best overall results are boldfaced, and the best results among the five categories are underlined. The values right below each category name refer to the number of features in each category.

As we can see in Tables 3.5 and 3.6, the category Lexicon-based (LEX column) achieved the best results among all categories, with the highest number of wins in 21 out of the 22 datasets in terms of Accuracy, and in 20 out of the 22 datasets in terms of average F-measure. In the overall evaluation, regarding Accuracy, the category Lexicon-based outperformed the set of all meta-features (ALL column) in three datasets (sarcasm, hobbit, and STS-gold). Considering the overall results obtained with the F-measure metric, the category Lexicon-based achieved the best results in six out of the 22 datasets (sarcasm, hobbit, iphone6, movie, STS-gold, and SemEval16). Nevertheless, the set of all meta-features achieved the best overall results in 19 out of the 22 datasets in terms of Accuracy, and in 17 out of the 22 datasets in terms of average F-measure. None of the

Table 3.5: Accuracies (%) achieved by evaluating each category of meta-features using an RF classifier.

Dataset	*Accuracy*					
	MIC	**POS**	**SUR**	**EMO**	**LEX**	**ALL**
	10	25	15	10	70	130
irony	64.6	61.5	60.0	66.2	76.9	**81.5**
sarcasm	59.2	54.9	47.9	53.5	**81.7**	80.3
aisopos	61.2	68.7	54.0	91.4	82.4	**92.8**
SemEval-Fig	87.5	87.2	83.5	85.0	87.5	**90.3**
sentiment140	59.1	49.9	53.5	54.9	84.1	**85.0**
person	66.3	69.2	67.4	71.1	83.1	**83.6**
hobbit	66.5	74.7	67.4	70.1	**91.8**	91.6
iphone6	65.4	77.3	73.1	69.5	82.3	**82.5**
movie	80.9	81.6	79.3	82.2	86.6	**87.0**
sanders	60.2	68.2	66.3	57.0	83.6	**84.8**
Narr	62.3	65.5	62.9	62.8	90.0	**90.3**
archeage	71.1	75.7	72.0	65.6	84.1	**85.4**
SemEval18	54.8	59.7	56.4	57.2	85.8	**86.0**
OMD	62.1	65.2	64.8	63.0	77.6	**79.8**
HCR	69.9	73.2	69.3	71.9	76.2	**77.5**
STS-gold	68.0	69.7	66.8	69.1	**93.5**	93.1
SentiStrength	60.6	60.9	59.0	60.2	82.6	**83.3**
Target-dependent	52.1	59.4	56.2	51.1	83.0	**83.1**
Vader	68.7	71.4	66.7	71.0	92.1	**93.0**
SemEval13	71.7	73.5	70.5	73.8	86.1	**86.9**
SemEval17	66.0	70.0	67.5	64.4	86.3	**86.5**
SemEval16	72.6	72.9	70.4	73.1	85.3	**85.4**
#wins (categories)	1	0	0	1	**21**	–
rank sums (categories)	85.5	56.0	92.0	73.0	**23.5**	–
#wins (overall)	0	0	0	0	3	**19**

{LEX} ≻ {MIC, POS, SUR, EMO} *and* {POS} ≻ {MIC, SUR}

Table 3.6: F-measure scores (%) achieved by evaluating each category of meta-features using an RF classifier.

Dataset	*F-measure*																	
	average						*positive*						*negative*					
	MIC	**POS**	**SUR**	**EMO**	**LEX**	**ALL**	**MIC**	**POS**	**SUR**	**EMO**	**LEX**	**ALL**	**MIC**	**POS**	**SUR**	**EMO**	**LEX**	**ALL**
	10	25	15	10	70	130	10	25	15	10	70	130	10	25	15	10	70	130
irony	60.7	56.0	58.9	52.7	76.1	**80.7**	30.3	19.4	35.0	0.0	61.5	**68.4**	76.3	74.7	71.1	79.6	83.5	**87.0**
sarcasm	58.5	54.8	47.4	37.3	**81.7**	80.1	50.8	50.0	39.3	0.0	**80.0**	77.4	65.1	59.0	54.3	69.7	**83.1**	82.5
aisopos	61.2	68.0	53.6	91.2	82.3	**92.8**	65.6	74.6	61.0	92.9	84.7	**93.8**	55.4	59.2	43.9	88.9	79.1	**91.4**
SemEval-Fig	85.7	84.7	78.2	78.5	84.7	**88.2**	42.9	36.9	3.6	0.0	35.5	**50.8**	93.0	92.9	91.0	91.9	93.1	**94.6**
sentiment140	58.6	49.8	53.5	49.9	84.1	**85.0**	54.8	51.9	54.2	34.7	84.5	**85.0**	62.6	47.7	52.7	65.5	83.8	**84.9**
person	58.6	60.9	63.0	59.1	82.4	**82.8**	79.4	81.4	79.4	83.1	88.7	**89.0**	7.5	10.6	22.7	0.0	67.0	**67.6**
hobbit	58.6	72.7	65.9	61.2	**91.8**	91.6	79.1	82.9	77.4	81.7	**93.9**	93.8	15.5	51.1	41.8	17.9	**87.2**	87.0
iphone6	60.6	75.7	72.3	57.2	**82.2**	**82.2**	77.9	84.8	81.5	82.0	87.4	**87.8**	20.7	54.7	51.2	0.0	**70.3**	69.3
movie	73.4	74.0	72.8	75.2	**85.3**	85.2	89.5	89.9	88.4	90.1	92.2	**92.5**	0.0	1.9	1.7	7.4	**54.0**	52.3
sanders	60.2	68.0	66.3	45.2	83.5	**84.7**	57.9	63.6	63.3	15.4	81.8	**83.1**	62.3	71.8	68.9	71.2	85.1	**86.2**
Narr	59.1	64.4	62.2	54.1	90.0	**90.3**	72.9	73.5	70.7	75.7	91.7	**92.0**	38.2	50.8	49.4	21.4	87.3	**87.7**
archeage	70.5	75.3	71.8	59.7	84.0	**85.3**	62.1	68.6	65.6	36.8	80.2	**81.8**	76.6	80.2	76.4	76.4	86.7	**87.8**
SemEval18	49.1	59.4	56.4	50.1	85.8	**86.0**	28.5	53.7	52.6	27.5	84.5	**84.6**	66.9	64.4	59.7	69.7	86.9	**87.2**
OMD	49.8	63.2	63.8	49.2	77.0	**79.2**	5.2	43.5	47.5	2.2	66.5	**69.5**	76.3	74.9	73.5	77.2	83.2	**84.9**
HCR	61.5	67.3	65.3	60.8	73.5	**74.7**	9.5	25.8	26.4	2.9	44.7	**46.4**	82.0	83.7	80.6	83.6	84.8	**85.8**
STS-gold	58.5	65.7	64.7	57.7	**93.5**	93.1	9.5	33.2	36.5	4.8	**89.5**	88.7	80.6	80.4	77.5	81.6	**95.3**	95.0
SentiStrength	60.1	59.7	58.1	47.3	82.6	**83.3**	67.9	69.3	67.3	74.5	85.2	**85.8**	49.0	46.1	45.1	9.0	78.9	**79.7**
Target-dependent	48.6	59.4	56.2	37.6	83.0	**83.1**	62.1	59.2	56.3	8.6	**82.9**	**82.9**	35.2	59.7	56.0	66.6	83.2	**83.3**
Vader	57.4	68.1	63.9	60.8	92.1	**92.9**	81.3	81.3	77.9	82.6	94.4	**95.0**	4.1	38.5	32.7	12.0	86.9	**88.2**
SemEval13	61.4	67.9	67.8	64.0	85.8	**86.5**	83.4	83.9	81.2	84.7	90.6	**91.2**	2.5	25.2	32.2	8.9	72.8	**74.1**
SemEval17	62.9	68.6	66.9	53.9	86.2	**86.4**	40.9	52.9	52.9	14.4	81.2	**81.3**	76.2	78.0	75.2	77.5	89.3	**89.4**
SemEval16	61.5	66.1	68.1	62.2	**85.0**	84.9	84.1	83.8	81.0	84.4	90.2	**90.3**	1.1	18.9	33.7	2.7	**71.0**	70.7
#wins (categories)	1	0	0	1	**20**	–	1	0	0	1	**20**	–	0	0	0	1	**21**	–
rank sums (categories)	83.0	56.5	71.0	95.0	**24.5**	–	78.0	63.0	83.0	81.0	**25.0**	–	84.0	65.0	87.5	70.5	**23.0**	–
#wins (overall)	0	0	0	0	6	**17**	0	0	0	0	4	**19**	0	0	0	0	6	**16**

{LEX} ≻ {MIC, POS, SUR, EMO} *and* {POS, SUR} ≻ {EMO}

other categories, namely Microblog (MIC column), Part-of-speech (POS column), Surface (SUR column), and Emoticon (EMO column) achieved meaningful results.

The Friedman and the Nemenyi tests detected that the category Lexicon-based is significantly better than all other categories, for both Accuracy and F-measure. In terms of Accuracy, the category Part-of-speech is only significantly better than categories Microblog and Surface. Regarding F-measure, both categories Part-of-speech and Surface are significantly better than category Emoticon only.

In general, although using only the features from category Emoticon does not seem to be effective in the sentiment classification of tweets, it is interesting to point out that this category has achieved the best result for dataset aisopos among all categories. Analyzing the tweets from this dataset, we note that about 80% of them contain emoticons. Since the polarity of emoticons are taken into account in the features of this category, the sentiment detection of tweets may benefit from this information. Indeed, analyzing the most informative meta-features for this dataset by ranking all 130 meta-features with the Information Gain (IG) relevance measure, five out of the top 10 most relevant features are *whether the tweet has negative emoticon*, *number of negative emoticons*, *whether the last token is negative emoticon*, *whether the tweet has positive emoticon*, and *number of positive emoticons*, all of them from the category Emoticon. Table 3.7 presents the top 10 features for dataset aisopos.

Table 3.7: Top 10 most relevant meta-features for dataset aisopos.

rank	IG	meta-feature
1	**0.592**	**EMO_hasNegativeEmoticon**
2	**0.592**	**EMO_numberOfNegativeEmoticons**
3	0.452	LEX_balanceScoreSentiment140
4	**0.364**	**EMO_isLastTokenNegativeEmoticon**
5	**0.348**	**EMO_hasPositiveEmoticon**
6	0.286	LEX_totalScoreOfNegativeWordsInSentiment140
7	0.273	LEX_totalScoreOfPositiveWordsInSentiment140
8	0.270	LEX_maximalScoreOfNegativeWordsInSentiment140
9	**0.266**	**EMO_numberOfPositiveEmoticons**
10	0.217	LEX_maximalScoreOfPositiveWordsInSentiment140

Still, regarding the rank generated by the IG measure, nearly all meta-features belonging to category Surface appear at the bottom of the rank for most datasets. Interestingly, for datasets irony and OMD, surface features referring to punctuation are ranked among the top 25 most significant features for dataset irony and among the top 20 features for dataset OMD (among all 130 meta-features), as shown in Tables 3.8 and 3.9, respectively. For dataset irony, the surface features *whether the tweet has exclamation mark* and *number of exclamation mark* are ranked as the 22nd and 23rd most relevant meta-

features. Similarly, for dataset OMD, the features *whether the tweet has question mark* and *number of question mark* are ranked as the 18th and 19th most discriminative features. It is in agreement with recent findings on the irony detection task, which has acknowledged that punctuation marks are useful to identify irony, especially in tweets [34]. Also, it is worth mentioning that dataset OMD, whose tweets are related to a political debate, may contain ironic content due to its nature.

Table 3.8: Top 25 most relevant meta-features for dataset irony.

rank	IG	meta-feature
1	0.241	LEX_maximalScoreOfPositiveWordsInSentiWordNet
2	0.228	LEX_sumOfPolarityOfARVNFromSentiWordNet
3	0.228	LEX_totalScoreOfPositiveWordsInSentiWordNet
4	0.228	LEX_balanceScoreSentiWordNet
5	0.203	LEX_numberOfPositiveAdjectives
6	0.197	LEX_balanceScoreAFINN
7	0.160	LEX_balanceScoreNRCHashtagLexicon
8	0.157	LEX_numberOfNegativeWordsInAFINN
9	0.157	LEX_totalScoreOfNegativeWordsInAFINN
10	0.157	LEX_maximalScoreOfNegativeWordsInAFINN
11	0.146	LEX_totalScoreOfPositiveWordsInAFINN
12	0.146	LEX_maximalScoreOfPositiveWordsInAFINN
13	0.146	LEX_balanceScoreNRCEmotionLexicon
14	0.144	LEX_numberOfNegativeWordsInNRCHashtagLexicon
15	0.144	LEX_maximalScoreOfNegativeWordsInNRCHashtagLexicon
16	0.132	LEX_numberOfPositiveWordsInSentiWordNet
17	0.118	LEX_numberOfNegativeWordsInOpinionFinderLexicon
18	0.114	LEX_numberOfNegativeWordsInBingLiuLexicon
19	0.114	LEX_totalScoreOfNegativeWordsInBingLiuLexicon
20	0.102	LEX_numberOfPositiveNouns
21	0.099	LEX_numberOfPositiveWordsInAFINN
22	**0.079**	**SUR_hasExclamationMark**
23	**0.079**	**SUR_numberOfExclamationMark**
24	0.079	MIC_hasInternetSlang
25	0.079	MIC_numberOfInternetSlang

Table 3.9: Top 20 most relevant meta-features for dataset OMD.

rank	IG	meta-feature
1	0.133	LEX_totalScoreOfNegativeWordsInSentiment140
2	0.111	LEX_balanceScoreBingLiuLexicon
3	0.105	LEX_maximalScoreOfNegativeWordsInSentiment140
4	0.096	LEX_balanceScoreAFINN
5	0.088	LEX_balanceScoreSentiment140
6	0.076	LEX_numberOfNegativeWordsInSentiment140
7	0.073	LEX_balanceScoreNRCHashtagLexicon
8	0.070	LEX_totalScoreOfNegativeWordsInBingLiuLexicon
9	0.070	LEX_numberOfNegativeWordsInBingLiuLexicon
10	0.063	LEX_sumOfPolarityOfAdjAdvVerbNounFromSentiWordNet
11	0.063	LEX_balanceScoreSentiWordNet
12	0.062	LEX_totalScoreOfNegativeWordsInNRCHashtagLexicon
13	0.061	LEX_maximalScoreOfNegativeWordsInNRCHashtagLexicon
14	0.057	LEX_totalScoreOfNegativeWordsInAFINN
15	0.057	LEX_maximalScoreOfNegativeWordsInAFINN
16	0.057	LEX_numberOfNegativeWordsInAFINN
17	0.056	LEX_maximalScoreOfPositiveWordsInAFINN
18	**0.055**	**SUR_hasQuestionMark**
19	**0.055**	**SUR_numberOfQuestionMark**
20	0.052	LEX_totalScoreOfNegativeWordsInSentiWordNet

Furthermore, analyzing the top relevant meta-features for datasets aisopos, irony, and OMD, as shown in Tables 3.7, 3.8, and 3.9, it is interesting to note that, regarding the meta-features from category Lexicon-based, it seems that the sentiment classification of tweets benefits from the use of a diverse set of dictionaries from the literature. As we can observe, features extracted from the seven different lexical resources we used, i.e., Sentiment140, SentiWordNet, AFINN, NRC-emotion, NRC-hashtag, Bing Liu's lexicon, and OpinionFinder, appear at the top of the rank.

In addition to the individual assessment of each category, we also investigate the reverse situation. We analyze how each category of meta-features contributes to the set of all features. Table 3.10 and Table 3.11 show the results of this investigation in terms of Accuracy and F-measure, respectively. The Loss column shows the loss (or gain) in Accuracy and F-measure when one category is removed, as compared to the set of all meta-features (ALL column).

Table 3.10: Accuracies (%) achieved by evaluating different subsets of meta-features using the RF classifier.

Dataset	***Accuracy***										
	ALL	**ALL−MIC**		**ALL−POS**		**ALL−SUR**		**ALL−EMO**		**ALL−LEX**	
		Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss
irony	**81.5**	75.4	−6.1	76.9	−4.6	80.0	−1.5	76.9	−4.6	66.2	−15.3
sarcasm	**80.3**	73.2	−7.1	76.1	−4.2	78.9	−1.4	74.6	−5.7	53.5	−26.8
aisopos	92.8	93.5	+0.7	**93.9**	+1.1	93.2	+0.4	81.7	−11.1	91.0	−1.8
SemEval-Fig	**90.3**	89.7	−0.6	89.4	−0.9	90.0	−0.3	90.0	−0.3	87.9	−2.4
sentiment140	**85.0**	84.7	−0.3	83.0	−2.0	84.4	−0.6	84.4	−0.6	63.8	−21.2
person	**83.6**	**83.6**	–	83.1	−0.5	82.9	−0.7	83.4	−0.2	72.4	−11.2
hobbit	91.6	91.8	+0.2	91.8	+0.2	**92.0**	+0.4	91.0	−0.6	75.3	−16.3
iphone6	82.5	**83.3**	+0.8	83.1	+0.6	82.1	−0.4	81.8	−0.7	79.1	−3.4
movie	**87.0**	86.6	−0.4	86.5	−0.5	**87.0**	–	86.6	−0.4	81.3	−5.7
sanders	84.8	**84.9**	+0.1	84.2	−0.6	84.1	−0.7	83.3	−1.5	71.5	−13.3
Narr	90.3	**90.4**	+0.1	90.1	−0.2	90.1	−0.2	90.1	−0.2	74.2	−16.1
archeage	**85.4**	84.5	−0.9	84.9	−0.5	84.5	−0.9	84.8	−0.6	80.0	−5.4
SemEval18	**86.0**	85.6	−0.4	85.7	−0.3	85.4	−0.6	85.4	−0.6	64.2	−21.8
OMD	**79.8**	79.2	−0.6	79.3	−0.5	78.4	−1.4	79.3	−0.5	68.2	−11.6
HCR	77.5	75.9	−1.6	**77.7**	+0.2	77.1	−0.4	77.4	−0.1	73.5	−4.0
STS-gold	93.1	92.7	−0.4	92.8	−0.3	**93.2**	+0.1	92.8	−0.3	71.5	−21.6
SentiStrength	**83.3**	82.7	−0.6	82.9	−0.4	83.0	−0.3	82.4	−0.9	66.1	−17.2
Target-dependent	**83.1**	**83.1**	–	82.8	−0.3	82.9	−0.2	82.9	−0.2	61.6	−21.5
Vader	93.0	92.7	−0.3	93.1	+0.1	**93.2**	+0.2	92.1	−0.9	74.5	−18.5
SemEval13	**86.9**	**86.9**	–	**86.9**	–	86.6	−0.3	86.6	−0.3	75.1	−11.8
SemEval17	86.5	86.5	–	86.5	–	86.5	–	**86.6**	+0.1	72.8	−13.7
SemEval16	85.4	85.4	–	85.5	+0.1	85.3	−0.1	**85.7**	+0.3	74.3	−11.1
#gains	–	5		6		4		2		0	
#losses	–	12		14		16		20		22	

In Tables 3.10 and 3.11, as we can see in the #gains and #losses rows, removing any category from the set of all meta-features is not beneficial, especially considering the category Lexicon-based (losses in all 22 datasets for both Accuracy and F-measure). In general, the gains achieved by removing the meta-features from some category are not relevant, except for dataset aisopos, whose Accuracy and F-measure values increased up to 1.1% by removing the meta-features from category Part-of-speech (ALL−POS column).

Table 3.11: Average F-measure scores (%) achieved by evaluating different subsets of meta-features using the RF classifier.

Dataset	*F-measure (average)*									
	ALL	**ALL−MIC**		**ALL−POS**		**ALL−SUR**		**ALL−EMO**		**ALL−LEX**	
		Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss
irony	**80.7**	74.7	−6.0	76.1	−4.6	79.3	−1.4	75.6	−5.1	61.9	−18.8
sarcasm	**80.1**	73.0	−7.1	75.9	−4.2	78.7	−1.4	74.3	−5.8	53.1	−27.0
aisopos	92.8	93.5	+0.7	**93.9**	+1.1	93.1	+0.3	81.6	−11.2	90.9	−1.9
SemEval-Fig	**88.2**	87.7	−0.5	87.0	−1.2	88.0	−0.2	88.0	−0.2	85.2	−3.0
sentiment140	**85.0**	84.7	−0.3	83.0	−2.0	84.4	−0.6	84.4	−0.6	63.8	−21.2
person	82.8	**82.9**	+0.1	82.4	−0.4	82.2	−0.6	82.6	−0.2	66.3	−16.5
hobbit	91.6	91.8	+0.2	91.8	+0.2	**92.0**	+0.4	91.0	−0.6	72.8	−18.8
iphone6	82.2	**83.1**	+0.9	82.8	+0.6	81.8	−0.4	81.3	−0.9	77.4	−4.8
movie	85.2	84.8	−0.4	84.9	−0.3	**85.5**	+0.3	85.0	−0.2	73.5	−11.7
sanders	84.7	**84.8**	+0.1	84.2	−0.5	84.0	−0.7	83.3	−1.4	71.4	−13.3
Narr	90.3	**90.4**	+0.1	90.1	−0.2	90.1	−0.2	90.0	−0.3	73.9	−16.4
archeage	**85.3**	84.4	−0.9	84.8	−0.5	84.4	−0.9	84.7	−0.6	79.9	−5.4
SemEval18	**86.0**	85.6	−0.4	85.7	−0.3	85.4	−0.6	85.4	−0.6	63.9	−22.1
OMD	**79.2**	78.5	−0.7	78.7	−0.5	77.8	−1.4	78.7	−0.5	66.2	−13.0
HCR	74.7	73.1	−1.6	**75.2**	+0.5	74.4	−0.3	74.7	–	67.9	−6.8
STS-gold	93.1	92.6	−0.5	92.7	−0.4	**93.2**	+0.1	92.7	−0.4	67.7	−25.4
SentiStrength	**83.3**	82.7	−0.6	82.9	−0.4	83.0	−0.3	82.3	−1.0	65.7	−17.6
Target-dependent	**83.1**	**83.1**	–	82.8	−0.3	82.9	−0.2	82.9	−0.2	61.6	−21.5
Vader	92.9	92.6	−0.3	93.0	+0.1	**93.1**	+0.2	92.0	−0.9	71.9	−21.0
SemEval13	86.5	86.5	–	**86.6**	+0.1	86.3	−0.2	86.2	−0.3	71.2	−15.3
SemEval17	86.4	86.4	–	86.4	–	86.4	–	**86.5**	+0.1	71.7	−14.7
SemEval16	84.9	85.0	+0.1	85.1	+0.2	84.8	−0.1	**85.3**	+0.4	68.9	−16.0
#gains	–	7		7		5		2		0	
#losses	–	12		14		16		19		22	

3.3.3 Effectiveness of Word Embedding-based Features

In this section, we present the evaluation of the word embedding-based features. We used the ten different pre-trained embedding models summarized in Table 2.3 (Chapter 2), aiming at determining the most discriminative embedding model in distinguishing the sentiment expressed in tweets.

We adopted the Weka's AffectiveTweets package [12] for calculating the features from the pre-trained word embeddings. More precisely, for each dataset, we applied the default configuration of the *TweetToEmbeddingFeatureVector* filter to create a representation for each tweet by aggregating the embedding values of the words. In the default configuration of the filter, the aggregation is done by averaging the word vectors. A dummy word-embedding vector formed by zeroes is used for word with no corresponding embedding[12]. Also, as a preprocessing step, we replaced URLs by the token *someurl*, user mentions by the token *someuser*, and we removed stopwords.

An example of how features from pre-trained word embedding models are calculated is shown in Figure 3.1. Considering the tweet $t =$ "*he is a great book writer*" and some pre-trained model E, where each embedded word is represented by a real-valued vector

[12]As stated in `https://affectivetweets.cms.waikato.ac.nz`.

of d dimensions, the feature vector for tweet t is calculated by averaging the embedded values for each word in t, i.e., *great*, *book*, and *writer* (the stopwords *he*, *is*, and *a* are removed). For example, the value of the first feature created for tweet t is 0.133, which corresponds to the average of the embedded values from the first dimension (*dim1*) of the words from t in the embedding model E (i.e., 0.13, 0.15, and 0.12). At the end of the process, a feature vector of size d is created for tweet t.

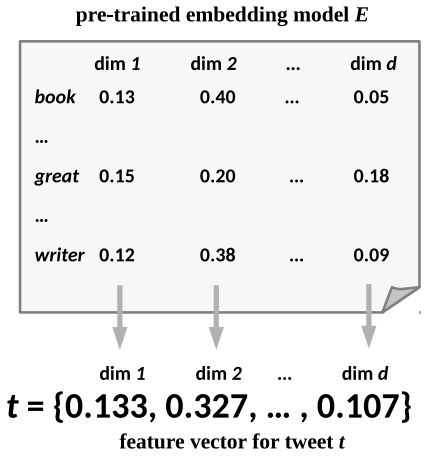

Figure 3.1: Example of how features from pre-trained embedding models are calculated.

We evaluate the word embedding representations in two steps. First, to determine which classification strategy is the most suitable for this type of feature, we evaluate the predictive performance of SVM, LR, and RF by using the features extracted from each of the ten pre-trained word vectors, one at a time and for each algorithm. For space reasons, we only report a summary of the results (refer to Appendix A for the detailed evaluation). Then, after determining the best classification strategy, we compare and analyze the predictive power of the features extracted from each pre-trained model, to identify the most appropriate model in the task of Twitter sentiment analysis.

Tables 3.12 and 3.13 show a summary of the results achieved by evaluating each classification strategy (SVM, LR, and RF columns) on the 22 datasets, and by using as features those calculated from one embedding model at a time (Embedding column), in terms of Accuracy and F-measure, respectively. For each assessed embedding model, we present the number of wins achieved by each classifier (#wins columns), as well as the rank sums, in parentheses. We also show whether the differences among the results are statistically significant (Friedman statistical test and Nemenyi post-hoc test columns).

From Table 3.12, we can notice that LR achieved the best results in nine out of the ten tested pre-trained embedding models, while SVM performed slightly better merely by

using SSWE embeddings, in terms of Accuracy. Moreover, LR outperformed SVM with a statistical difference between them in seven out of the nine wins. Conversely, RF did not achieve meaningful results, with the lowest number of wins.

Table 3.12: Summary of the accuracies achieved by evaluating SVM, RF, and LR classifiers on the 22 datasets of tweets, and by using as features those calculated from each pre-trained word embedding model.

Embedding	***Accuracy***				
	SVM #wins	**LR** #wins	**RF** #wins	**Friedman statistical test**	**Nemenyi post-hoc test**
w2v-GN	5 (41.5)	19 (**26.5**)	1 (64.0)	✔	{SVM, LR} ≻ {RF}
GloVe-WP	3 (46.0)	18 (**26.0**)	1 (60.0)	✔	{LR} ≻ {SVM, RF}
fastText	0 (48.0)	20 (**24.0**)	2 (60.0)	✔	{LR} ≻ {SVM, RF}
EWE	4 (46.0)	17 (**28.0**)	1 (58.0)	✔	{LR} ≻ {SVM, RF}
GloVe-TW	4 (44.5)	20 (**25.5**)	1 (62.0)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
w2v-Araque	2 (49.0)	18 (**26.5**)	3 (56.5)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
w2v-Edin	5 (42.5)	19 (**26.5**)	1 (63.0)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
SSWE	9 (**40.5**)	5 (45.5)	8 (46.0)	✘	not applicable
Emo2Vec	11 (39.5)	8 (**38.0**)	5 (54.5)	✔	{SVM, LR} ≻ {RF}
DeepMoji	5 (42.0)	16 (**28.0**)	1 (62.0)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}

Table 3.13: Summary of the average F-measure scores achieved by evaluating SVM, RF, and LR classifiers on the 22 datasets of tweets, and by using as features those calculated from each pre-trained word embedding model.

Embedding	***F-measure* (*average*)**				
	SVM #wins	**LR** #wins	**RF** #wins	**Friedman statistical test**	**Nemenyi post-hoc test**
w2v-GN	6 (39.5)	16 (**28.5**)	1 (64.0)	✔	{SVM, LR} ≻ {RF}
GloVe-WP	4 (44.5)	18 (**26.5**)	1 (61.0)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
fastText	4 (43.0)	19 (**26.0**)	1 (63.0)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
EWE	6 (41.5)	16 (**28.5**)	1 (62.0)	✔	{SVM, LR} ≻ {RF}
GloVe-TW	5 (41.0)	19 (**26.0**)	0 (65.0)	✔	{SVM, LR} ≻ {RF}
w2v-Araque	2 (46.0)	20 (**24.5**)	1 (61.5)	✔	{SVM} ≻ {RF} {LR} ≻ {SVM, RF}
w2v-Edin	5 (42.0)	18 (**27.0**)	1 (63.0)	✔	{SVM, LR} ≻ {RF}
SSWE	11 (**39.5**)	6 (43.5)	7 (49.0)	✘	not applicable
Emo2Vec	12 (38.0)	9 (**37.0**)	3 (57.0)	✔	{SVM, LR} ≻ {RF}
DeepMoji	8 (38.0)	13 (**31.0**)	1 (63.0)	✔	{SVM, LR} ≻ {RF}

Table 3.13 reports the results in terms of F-measure. LR outperformed SVM and RF

in nine out of the ten pre-trained models. SVM achieved the best overall results only by using SSWE model. The Friedman and Nemenyi tests detected that SVM and LR performed significantly better than RF for all assessed situations, except for the SSWE model. Although the results obtained with the LR classifier are significantly better than SVM only by using GloVe-WP, fastText, and w2v-Araque models, it has achieved the best overall results, as we can observe in the rank sums, presented in parentheses. Then, considering that the LR classifier achieved the best results for both Accuracy and F-measure, we acknowledge LR as the most appropriate classification strategy for the word embedding-based features.

Next, we analyze the performance of the LR classifier fed with the embedding-based features from each pre-trained model, in order to identify which model suits better in detecting the polarity of tweets. The results are presented in Tables 3.14 and 3.15, in terms of Accuracy and F-measure, respectively. The number of dimensions right below each embedding name refers to the number of features calculated from each pre-trained model.

Table 3.14: Comparison among the Accuracies (%) achieved with each pre-trained embedding model by using the LR classifier.

Dataset	***Accuracy***									
	***E*1**	***E*2**	***E*3**	***E*4**	***E*5**	***E*6**	***E*7**	***E*8**	***E*9**	***E*10**
	300	**300**	**300**	**300**	**200**	**500**	**400**	**50**	**100**	**256**
irony	70.8	**76.9**	73.8	69.2	66.2	69.2	75.4	73.8	**76.9**	73.8
sarcasm	67.6	63.4	64.8	69.0	69.0	70.4	56.3	**73.2**	62.0	59.2
aisopos	90.6	86.3	76.6	73.7	76.6	74.8	92.8	91.7	79.9	**94.6**
SemEval-Fig	88.2	86.3	88.2	86.9	87.2	86.0	89.1	87.5	87.5	**89.4**
sentiment140	84.1	85.5	82.5	83.8	83.6	80.2	**87.7**	84.1	84.7	80.8
person	81.3	80.4	83.1	**84.1**	83.1	78.6	81.3	78.6	79.3	80.4
hobbit	91.0	90.4	91.0	92.5	90.0	92.3	92.5	83.1	88.7	**92.7**
iphone6	78.8	77.8	81.2	78.4	**82.1**	78.4	81.6	74.8	78.8	79.7
movie	87.9	87.3	88.2	87.2	86.6	87.0	88.6	88.4	**89.3**	86.5
sanders	80.6	77.4	80.1	79.0	80.6	77.9	**82.9**	77.8	79.1	82.1
Narr	88.0	84.9	86.1	84.5	88.6	85.3	**89.6**	89.5	88.6	89.1
archeage	83.1	82.5	83.9	83.5	85.2	83.2	**87.0**	79.5	81.9	83.5
SemEval18	79.0	79.2	81.2	79.5	81.4	75.3	**82.8**	80.8	80.4	80.0
OMD	81.2	81.9	80.1	78.6	77.2	77.1	**83.3**	77.2	76.4	75.9
HCR	78.8	76.0	77.3	76.9	**79.0**	74.1	78.5	73.6	75.3	75.4
STS-gold	84.6	83.3	85.3	85.1	85.3	86.2	87.5	**87.8**	85.8	**87.8**
SentiStrength	77.0	75.9	78.2	77.9	78.0	76.8	81.2	79.2	**85.4**	79.9
Target-dependent	81.9	80.5	82.5	82.6	**83.1**	81.5	82.5	77.5	81.3	82.0
Vader	87.7	87.1	88.5	88.0	87.4	86.7	**89.3**	87.7	87.1	88.6
SemEval13	83.1	81.8	83.2	82.5	83.1	81.0	83.6	83.2	**88.7**	83.4
SemEval17	86.4	86.6	**88.5**	87.2	87.7	83.1	87.6	80.8	85.3	84.9
SemEval16	84.8	85.2	86.2	85.6	**86.4**	82.7	**86.4**	81.5	84.5	84.3
#wins	0	1	1	1	4	0	**8**	2	4	4
rank sums	122.0	152.5	97.0	129.5	105.5	175.5	**53.0**	138.0	126.5	109.0
rank position	5	9	2	7	3	10	**1**	8	6	4

{fastText, GloVe-TW, Deepmoji} $\succ$ {w2v-Araque}
{w2v-Edin} $\succ$ {w2v-GN, GloVe-WP, EWE, w2v-Araque, SSWE, Emo2Vec}

***E*1**: w2v-GN ***E*2**: GloVe-WP ***E*3**: fastText ***E*4**: EWE ***E*5**: GloVe-TW
***E*6**: w2v-Araque ***E*7**: w2v-Edin ***E*8**: SSWE ***E*9**: Emo2Vec ***E*10**: DeepMoji

Table 3.15: Comparison among the F-measure scores (%) achieved with each pre-trained embedding model by using the LR classifier.

Dataset	***F-measure***																													
	average										***positive***										***negative***									
	***E*1**	***E*2**	***E*3**	***E*4**	***E*5**	***E*6**	***E*7**	***E*8**	***E*9**	***E*10**	***E*1**	***E*2**	***E*3**	***E*4**	***E*5**	***E*6**	***E*7**	***E*8**	***E*9**	***E*10**	***E*1**	***E*2**	***E*3**	***E*4**	***E*5**	***E*6**	***E*7**	***E*8**	***E*9**	***E*10**
	300	**300**	**300**	**300**	**200**	**500**	**400**	**50**	**100**	**256**	**300**	**300**	**300**	**300**	**200**	**500**	**400**	**50**	**100**	**256**	**300**	**300**	**300**	**300**	**200**	**500**	**400**	**50**	**100**	**256**
irony	69.1	**75.6**	71.7	65.3	63.8	67.8	74.2	70.1	74.4	72.4	48.6	**59.5**	51.4	37.5	38.9	47.4	57.9	45.2	54.5	54.1	79.6	83.9	82.1	79.6	76.6	78.3	82.6	82.8	**84.5**	81.7
sarcasm	67.3	63.4	64.4	69.0	68.4	70.1	56.1	**73.2**	61.3	58.1	62.3	61.8	59.0	66.7	62.1	65.6	50.8	**70.8**	54.2	49.1	71.6	64.9	69.1	71.1	73.8	74.1	60.8	**75.3**	67.5	65.9
aisopos	90.6	86.2	76.5	73.3	76.4	74.8	92.8	91.7	79.8	**94.6**	92.0	88.5	80.0	78.5	80.2	78.3	93.9	92.8	82.7	**95.3**	88.7	83.2	71.9	66.4	71.4	70.1	91.3	90.2	75.9	**93.6**
SemEval-Fig	86.6	83.7	85.7	84.5	84.4	84.4	87.7	84.4	84.9	**87.8**	47.2	33.3	40.6	36.4	34.9	40.0	**52.1**	33.3	37.5	51.4	93.3	92.4	93.4	92.7	92.9	92.1	93.8	93.1	93.1	**94.1**
sentiment140	84.1	85.5	82.4	83.8	83.6	80.2	**87.7**	84.1	84.7	80.8	84.3	85.8	82.9	84.2	83.9	80.4	**87.9**	84.3	85.2	81.5	83.9	85.2	81.9	83.5	83.2	80.0	**87.6**	83.9	84.1	80.0
person	80.5	79.5	82.6	**83.5**	82.5	77.9	80.9	77.3	78.5	79.8	87.5	86.9	88.6	**89.2**	88.6	85.4	87.2	85.8	86.0	86.7	63.4	61.3	67.8	**69.6**	67.5	59.5	65.3	56.5	59.9	62.6
hobbit	90.9	90.3	90.8	92.5	89.9	92.3	92.5	82.7	88.6	**92.7**	93.5	93.1	93.6	94.6	92.8	94.4	94.5	88.1	91.8	**94.7**	85.5	84.4	85.1	88.0	83.9	87.9	88.5	71.2	81.7	**88.6**
iphone6	78.2	76.8	80.8	77.6	**81.6**	78.1	81.3	73.0	78.1	79.0	85.3	84.9	86.9	85.2	**87.7**	84.8	87.1	83.2	85.3	86.0	61.7	57.9	66.7	59.9	67.6	62.8	**67.8**	49.6	61.4	62.8
movie	86.3	85.5	86.5	85.4	84.6	85.5	87.3	86.9	**88.3**	84.6	93.0	92.7	93.2	92.6	92.3	92.4	93.3	93.3	**93.7**	92.2	55.8	53.0	56.0	52.6	49.7	54.1	60.0	57.5	**63.4**	50.0
sanders	80.6	77.3	80.1	79.0	80.6	78.0	**82.9**	77.7	79.1	82.1	78.6	74.9	78.5	77.6	79.0	76.5	**81.4**	75.7	77.5	80.6	82.3	79.4	81.4	80.3	82.0	79.2	**84.2**	79.5	80.5	83.4
Narr	88.0	84.8	86.1	84.4	88.6	85.3	**89.6**	89.4	88.5	89.1	90.2	87.7	88.6	87.4	90.6	88.0	**91.5**	91.4	90.7	91.0	84.6	80.5	82.3	80.0	85.6	81.3	**86.7**	86.4	85.3	86.2
archeage	83.0	82.5	83.8	83.5	85.2	83.2	**87.0**	79.3	81.8	83.4	79.3	78.8	80.6	80.4	82.1	79.7	**84.5**	74.6	77.6	79.7	85.7	85.1	86.2	85.8	87.4	85.7	**88.9**	82.7	84.8	86.1
SemEval18	79.0	79.2	81.2	79.5	81.4	75.2	**82.8**	80.8	80.4	80.0	77.0	77.4	79.5	77.5	79.5	72.6	**81.3**	78.9	78.4	77.7	80.7	80.8	82.7	81.2	83.0	77.5	**84.1**	82.4	82.1	81.9
OMD	80.9	81.6	79.8	78.3	76.9	76.7	**83.1**	76.5	75.8	75.4	72.8	74.1	71.7	69.3	67.3	67.2	**76.1**	65.7	64.9	64.9	85.6	86.1	84.6	83.6	82.5	82.4	**87.2**	83.0	82.2	81.7
HCR	77.6	74.2	75.7	75.4	**77.7**	71.6	77.3	68.3	71.5	72.5	**56.3**	48.4	52.0	51.2	56.1	41.8	55.6	28.8	38.5	42.2	86.0	84.4	85.1	84.9	**86.2**	83.3	85.8	83.8	84.5	84.4
STS-gold	84.3	82.8	85.0	84.8	85.0	86.0	87.3	**87.6**	85.5	87.5	73.5	70.7	74.8	74.3	74.6	76.6	78.8	**79.2**	75.4	79.0	89.1	88.3	89.6	89.5	89.7	90.2	91.1	91.3	90.0	**91.4**
SentiStrength	76.8	75.8	78.0	77.8	77.9	76.7	81.2	79.1	**85.4**	79.8	80.8	79.9	81.9	81.6	81.6	80.6	84.1	82.7	**87.7**	83.3	71.2	70.0	72.6	72.5	72.7	71.3	77.0	73.9	**82.1**	75.0
Target-dependent	81.9	80.5	82.5	82.6	**83.1**	81.5	82.5	77.5	81.3	82.0	81.9	80.5	82.4	82.5	**83.1**	81.2	82.5	77.3	81.1	81.7	81.8	80.5	82.5	82.7	**83.1**	81.8	82.5	77.7	81.5	82.3
Vader	87.4	86.8	88.3	87.7	87.1	86.4	**89.2**	87.3	86.8	88.3	91.4	91.0	91.9	91.5	91.1	90.6	**92.5**	91.4	91.0	92.0	78.6	77.6	80.1	79.2	78.0	76.9	**81.8**	78.2	77.4	80.1
SemEval13	82.5	81.0	82.6	81.7	82.5	80.3	83.1	82.6	**88.5**	82.9	88.8	88.0	88.9	88.4	88.8	87.4	89.0	88.9	**92.3**	89.0	65.7	62.4	66.0	63.8	65.5	61.4	67.5	65.6	**78.2**	66.6
SemEval17	86.3	86.6	**88.4**	87.2	87.6	82.8	87.5	80.4	85.2	84.7	81.4	81.7	**84.3**	82.7	83.2	75.8	83.1	72.2	79.6	78.7	89.3	89.4	**90.9**	89.9	90.3	87.0	90.2	85.3	88.5	88.3
SemEval16	84.3	84.7	85.8	85.0	86.0	82.1	**86.1**	80.6	84.0	83.8	89.9	90.2	90.8	90.4	**90.9**	88.5	**90.9**	87.9	89.7	89.6	69.3	70.2	72.4	70.7	72.9	64.9	**73.2**	61.2	68.5	68.2
#wins	0	1	1	1	3	0	**8**	2	3	3	1	1	1	1	3	0	**9**	2	3	2	0	0	1	1	2	0	**9**	1	4	4
rank sums	120.5	154.0	97.0	128.5	107.5	167.5	**52.0**	143.5	129.0	109.0	117.5	150.0	98.5	125.0	107.5	170.5	**51.0**	146.0	130.5	112.5	121.0	156.5	97.0	132.5	108.0	172.0	**51.5**	137.0	125.5	109.0
rank position	5	9	2	6	3	10	**1**	8	7	4	5	9	2	6	3	10	**1**	8	7	4	5	9	2	7	3	10	**1**	8	6	4

{fastText} $\succ$ {w2v-Araque} *and* {w2v-Edin} $\succ$ {w2v-GN, GloVe-WP, EWE, w2v-Araque, SSWE, Emo2Vec}

***E*1**: w2v-GN
***E*2**: GloVe-WP
***E*3**: fastText
***E*4**: EWE
***E*5**: GloVe-TW
***E*6**: w2v-Araque
***E*7**: w2v-Edin
***E*8**: SSWE
***E*9**: Emo2Vec
***E*10**: DeepMoji

As we can see in Tables 3.14 and 3.15, the w2v-Edin model achieved the best performance in eight out of the 22 datasets, for both Accuracy and average F-measure, and was ranked first in the overall evaluation (rank position row). Although the w2v-Edin model did not leverage any sentiment information during its construction, as enlightened by its authors in [11], its training parameters were optimized for the emotion detection task on tweets, which may have benefited the sentiment classification of tweets. The fastText model achieved the second-best results, followed by GloVe-TW, DeepMoji, and w2v-GN, for both Accuracy and F-measure.

The Friedman test detected a significant difference among the results. The Nemenyi test showed that the results achieved by the w2v-Edin embeddings are significantly better than w2v-GN, GloVe-WP, EWE, w2v-Araque, SSWE, and Emo2Vec, for both Accuracy and average F-measure. In terms of Accuracy, fastText, GloVe-TW, and DeepMoji results are significantly better the w2v-Araque, which is the model that achieved the worse overall performance. Regarding average F-measure, only the results obtained with the fastText model are significantly better than the w2v-Araque one.

Among the affective embeddings (DeepMoji, Emo2Vec, EWE, and SSWE), the SSWE model achieved the worse performance for both Accuracy and F-measure. Surprisingly, the generic embeddings w2v-Edin, fastText, and GloVe-TW outperformed all affective embeddings. One possible reason is the number of words embedded in the pre-trained models, i.e., the vocabulary size of each pre-trained word vector. Indeed, as shown in Table 2.3, the vocabulary sizes of the fastText and GloVe-TW generic embeddings (1M and 1.2M, respectively) are much larger than the DeepMoji, EWE, and SSWE affective ones (50K, 183K, and 137K, respectively). Although the number of words embedded in the Emo2Vec affective model is as large as in the GloVe-TW generic one (1.2M), Emo2Vec may have performed poorly considering that it is weak on capturing syntactic and semantic meaning of words, as stated by its authors in [2].

To investigate whether the vocabulary size may influence in the results, Table 3.16 presents a coverage analysis of the pre-trained models for the five best-ranked embeddings (w2v-Edin, fastText, GloVe-TW, DeepMoji, and w2v-GN). More specifically, for each dataset, we show the fraction of words found in a given pre-trained model. The information below each model name refers to their vocabulary size. We also show, in parentheses, the rank assigned for each model. We can observe that the w2v-Edin model, which achieved the best overall results, has the highest coverage for all datasets, except for Semeval13. Also, fastText and GloVe-TW, whose vocabulary sizes are much larger than

the DeepMoji one, have the second and third highest coverage, followed by DeepMoji. The w2v-GN model has the lowest coverage, even though it is the model with the largest vocabulary size (3M). Since this model was trained on a corpus of Google news articles, it may not have generalized well to short, noisy texts, such as tweets.

Table 3.16: Coverage analysis (%) of the pre-trained word vectors vocabulary for the five best ranked embeddings.

Dataset	**w2v-GN** $\lvert V\rvert$ = 3M	**fastText** $\lvert V\rvert$ = 1M	**GloVe-TW** $\lvert V\rvert$ = 1.2M	**w2v-Edin** $\lvert V\rvert$ = 259K	**DeepMoji** $\lvert V\rvert$ = 50K
irony	71.33 (5.0)	78.05 (3.0)	78.23 (2.0)	**82.48** (1.0)	75.40 (4.0)
sarcasm	72.27 (5.0)	76.76 (2.0)	75.98 (3.0)	**81.64** (1.0)	74.22 (4.0)
aisopos	71.12 (5.0)	76.31 (3.0)	77.81 (2.0)	**82.67** (1.0)	75.02 (4.0)
SemEval-Fig	70.31 (5.0)	74.24 (4.0)	74.40 (2.0)	**81.17** (1.0)	74.34 (3.0)
sentiment140	75.69 (5.0)	80.80 (2.0)	80.45 (3.0)	**86.49** (1.0)	76.92 (4.0)
person	74.81 (5.0)	81.53 (2.0)	80.66 (3.0)	**86.65** (1.0)	77.51 (4.0)
hobbit	67.33 (5.0)	74.29 (2.0)	73.29 (3.0)	**77.27** (1.0)	69.25 (4.0)
iphone6	63.88 (5.0)	66.29 (3.0)	67.04 (2.0)	**73.27** (1.0)	65.16 (4.0)
movie	80.52 (5.0)	84.25 (3.0)	85.26 (2.0)	**91.59** (1.0)	82.36 (4.0)
sanders	61.77 (5.0)	66.01 (2.0)	65.99 (3.0)	**75.04** (1.0)	62.18 (4.0)
Narr	72.76 (5.0)	79.60 (3.0)	81.76 (2.0)	**88.43** (1.0)	78.73 (4.0)
archeage	61.71 (5.0)	70.45 (2.0)	69.12 (3.0)	**74.51** (1.0)	63.41 (4.0)
SemEval18	51.99 (5.0)	60.76 (3.0)	61.74 (2.0)	**68.15** (1.0)	59.24 (4.0)
OMD	72.15 (5.0)	85.04 (2.0)	82.84 (3.0)	**86.95** (1.0)	75.94 (4.0)
HCR	52.13 (5.0)	63.82 (2.0)	62.19 (3.0)	**70.26** (1.0)	55.47 (4.0)
STS-gold	63.64 (5.0)	73.36 (3.0)	73.82 (2.0)	**79.53** (1.0)	69.30 (4.0)
SentiStrength	54.31 (5.0)	64.04 (3.0)	66.01 (2.0)	**71.81** (1.0)	60.50 (4.0)
Target-dependent	65.57 (5.0)	79.81 (3.0)	82.98 (2.0)	**84.75** (1.0)	73.85 (4.0)
Vader	66.79 (5.0)	82.07 (3.0)	83.26 (2.0)	**88.93** (1.0)	75.32 (4.0)
SemEval13	**80.60** (1.0)	62.01 (4.0)	65.58 (3.0)	70.81 (2.0)	57.70 (5.0)
SemEval17	38.13 (5.0)	50.23 (2.0)	49.80 (3.0)	**54.19** (1.0)	42.85 (4.0)
SemEval16	38.67 (5.0)	51.92 (3.0)	53.23 (2.0)	**57.40** (1.0)	45.52 (4.0)
rank sums	106.0	59.0	54.0	**23.0**	88.0

3.3.4 Overall Analysis of Features Effectiveness

In the previous sections (3.3.1, 3.3.2, and 3.3.3), we have identified the best classifiers for each feature set proposed and adopted in state-of-the-art works in Twitter sentiment analysis, i.e., n-grams, meta-features, and word embedding-based features. In this section, we present an overall analysis of those different sets of features. More specifically, we aim at effectively responding to research question RQ1 — "*Which group of features is the most effective in Twitter sentiment analysis?*", by performing a comparison among the following classifiers: RF fed with meta-features, SVM with n-grams, and LR with embedding-based features calculated from the w2v-Edin model.

Table 3.17 and Table 3.18 report the comparison among the aforementioned classifiers (meta-features, n-grams, and w2v-Edin columns), in terms of Accuracy and F-measure, respectively. The best results achieved by each classifier are in bold-face type.

We can see that the RF classifier fed with meta-features achieved the highest accuracies in 13 out of the 22 datasets, for both Accuracy and average F-measure, followed by the embedding-based features provided by w2v-Edin word vectors. In general, the n-gram features achieved worse predictive performance.

Note that the n-gram features outperformed meta-features and w2v-Edin only for datasets SemEval-Fig, hobbit, and HCR. The tweets from SemEval-Fig and HCR are regarded as belonging to challenging domains, such as metaphor languages and health campaigns, respectively. For that reason, the n-grams may have succeeded in capturing more context from the specific language used in these datasets. Indeed, by analyzing the most relevant among all features for dataset HCR, we observed that the unigram "#tcot", which means "*top conservatives on Twitter*", appears at the top of the ranking as the most important feature. Since this term is very context-sensitive, the n-gram classifier may have benefited from this kind of information.

Regarding meta-features, we can observe that applying them on tweets from datasets irony and sarcasm led to a significant gain in accuracy as compared to n-grams and embedding-based features. As mentioned before, ironic and sarcastic tweets usually contain signals, such as punctuation marks, that may help determine the sentiment expressed in them.

It is worth mentioning that the number of meta-features is much smaller than the number of n-grams. As shown in Table 3.3 (#features column), the number of n-grams varies from 1.8K to 252.1K (datasets irony and SemEval16, respectively), while an increased predictive performance was achieved by using only a small set of 130 meta-features. Similarly, the number of meta-features is smaller than the number of the features extracted from the w2v-Edin pre-trained model, i.e., 400 features, as shown in Table 2.3 ($|D|$ column).

Another advantage of meta-features over word embedding representations is the fact that meta-features can be easily interpreted. For example, by applying relevance measures, such as IG, to determine the most predictive meta-features, we can figure out the kind of information that may be useful in distinguishing the positive tweets from the negative ones for some specific domain. On the other hand, the features calculated from pre-trained embedding models, i.e., real values corresponding to distinct dimensions or aspects of words, are hard to explain.

Also, unlike meta-features, pre-trained embedding models are language-dependent. In general, the word vectors are trained on huge text corpora containing documents from the same language. Otherwise, it is not possible to capture semantic and syntactic relations

Table 3.17: Comparison among the Accuracies (%) of the best classifiers under the individual evaluation of each feature set.

Dataset	***Accuracy***		
	meta-features	***n*-grams**	**w2v-Edin**
	RF	**SVM**	**LR**
irony	**81.5**	66.2	75.4
sarcasm	**80.3**	50.7	56.3
aisopos	**92.8**	87.8	**92.8**
SemEval-Fig	90.3	**91.0**	89.1
sentiment140	85.0	84.1	**87.7**
person	**83.6**	79.0	81.3
hobbit	91.6	**92.9**	92.5
iphone6	**82.5**	77.6	81.6
movie	87.0	84.1	**88.6**
sanders	**84.8**	83.0	82.9
Narr	**90.3**	83.7	89.6
archeage	85.4	86.3	**87.0**
SemEval18	**86.0**	80.2	82.8
OMD	79.8	81.2	**83.3**
HCR	77.5	**79.1**	78.5
STS-gold	**93.1**	84.0	87.5
SentiStrength	**83.3**	73.2	81.2
Target-dependent	**83.1**	81.4	82.5
Vader	**93.0**	84.8	89.3
SemEval13	**86.9**	81.0	83.6
SemEval17	86.5	86.9	**87.6**
SemEval16	85.4	85.8	**86.4**
#wins	**13**	3	7
rank sums	**37.5**	55.0	39.5

Table 3.18: Comparison among the F-measure scores (%) of the best classifiers under the individual evaluation of each feature set.

Dataset	***F-measure***								
	average			***positive***			***negative***		
	meta-features	***n*-grams**	**w2v-Edin**	**meta-features**	***n*-grams**	**w2v-Edin**	**meta-features**	***n*-grams**	**w2v-Edin**
	RF	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**
irony	**80.7**	52.7	74.2	**68.4**	0.0	57.9	**87.0**	79.6	82.6
sarcasm	**80.1**	48.0	56.1	**77.4**	58.8	50.8	**82.5**	38.6	60.8
aisopos	**92.8**	87.4	**92.8**	93.8	90.3	**93.9**	**91.4**	83.5	91.3
SemEval-Fig	88.2	**89.8**	87.7	50.8	**60.3**	52.1	94.6	**94.9**	93.8
sentiment140	85.0	84.1	**87.7**	85.0	85.0	**87.9**	84.9	83.2	**87.6**
person	**82.8**	77.8	80.9	**89.0**	86.1	87.2	**67.6**	57.4	65.3
hobbit	91.6	**93.0**	92.5	93.8	**94.7**	94.5	87.0	**89.3**	88.5
iphone6	**82.2**	73.6	81.3	**87.8**	86.0	87.1	**69.3**	45.2	67.8
movie	85.2	80.2	**87.3**	92.5	91.0	**93.3**	52.3	31.0	**60.0**
sanders	**84.7**	83.0	82.9	**83.1**	82.4	81.4	**86.2**	83.6	84.2
Narr	**90.3**	83.7	89.6	**92.0**	86.6	91.5	**87.7**	79.2	86.7
archeage	85.3	86.4	**87.0**	81.8	84.3	**84.5**	87.8	87.9	**88.9**
SemEval18	**86.0**	79.9	82.8	**84.6**	76.6	81.3	**87.2**	82.8	84.1
OMD	79.2	81.0	**83.1**	69.5	74.0	**76.1**	84.9	85.2	**87.2**
HCR	74.7	**77.7**	77.3	46.4	**55.9**	55.6	85.8	**86.3**	85.8
STS-gold	**93.1**	83.4	87.3	**88.7**	71.3	78.8	**95.0**	88.9	91.1
SentiStrength	**83.3**	72.7	81.2	**85.8**	78.5	84.1	**79.7**	64.4	77.0
Target-dependent	**83.1**	81.4	82.5	**82.9**	81.3	82.5	**83.3**	81.5	82.5
Vader	**92.9**	83.9	89.2	**95.0**	89.7	92.5	**88.2**	71.0	81.8
SemEval13	**86.5**	78.9	83.1	**91.2**	88.0	89.0	**74.1**	54.5	67.5
SemEval17	86.4	86.8	**87.5**	81.3	82.1	**83.1**	89.4	89.6	**90.2**
SemEval16	84.9	85.1	**86.1**	90.3	90.7	**90.9**	70.7	70.2	**73.2**
#wins	**13**	3	7	**12**	3	7	**13**	3	6
rank sums	**37.5**	55.0	39.5	39.5	53.5	**39.0**	**35.5**	57.0	39.5

between words. Meta-features, in turn, can be used disregarding language limitations, except for the lexicon-based meta-features, which rely on sentiment lexicons from specific languages. Nevertheless, it is possible to use lexicons generated for any language, whether it is available. Thus, it is not necessarily a limitation of meta-features. Indeed, Sousa et al. [81] successfully used a subset of meta-features we have examined and categorized in our previous work [18] to identify relevant tweets in preventing mosquito-borne diseases, such as the Zika virus, in Portuguese tweets. Therefore, meta-features are not only language-independent but can also be easily applied in cross-domain problems.

3.4 Summary

In this chapter, as one of the contributions of this thesis, we presented an experimental evaluation of the importance of the feature sets described in Chapter 2, in the polarity classification of tweets from distinct domains. Those features include n-grams, meta-features, and word embedding-based features. We used twenty-two datasets of tweets in the series of experiments conducted in this study. To the best of our knowledge, this is the first work that evaluates different types of features for a significant number of datasets of tweets. Besides, we also presented an assessment study of the categories of meta-features we proposed in Chapter 2

Attending to respond to research question RQ1 — "*Which group of features is the most effective in Twitter sentiment analysis?*", our experiments showed that a concise yet rich set of 130 meta-features aggregated from the literature achieved the best overall results compared to n-grams and word embedding-based features. It may be evidence that meta-features play an important role in Twitter sentiment analysis.

As another contribution, we evaluated the categories of meta-features proposed in Chapter 2 to identify the most suitable one in determining the polarity of tweets. We showed that lexicon-based features, i.e., features that rely on sentiment lexicons and lists of words, have the most predictive power. Nevertheless, when considering the set of all meta-features, we showed that by removing the features from all categories, one at a time, the classification performance drop considerably for all datasets. Then, we believe that meta-features from distinct categories may complement each other in this task.

Finally, we presented an underlying evaluation of a significant collection of ten pre-trained word embedding models adopted in the literature of Twitter sentiment analysis. We evaluated six generic models and four affective ones. Interestingly, the generic w2v-

Edin model, trained on a huge corpus of 10M tweets, achieved the best overall results over all other generic and affective models.

In the next chapter, we combine the feature sets evaluated in this chapter, considering that features from different types might complement each other, leading to an improvement in detecting the polarity of tweets. We investigate two strategies for combination, such as feature concatenation and ensemble learning methods.

Chapter 4

Combining Features in Twitter Sentiment Analysis

4.1 Introduction

Several studies in the literature of Twitter sentiment analysis adopt stand-alone classifiers using, as features, n-grams, different sets of meta-features, or word embedding-based features, as discussed in Chapter 2. In this context, arguing that the combination of classifiers has not been properly investigated in the literature, da Silva et al. [25] is one of the earliest works that effectively exploit ensemble approaches in the sentiment detection task on Twitter data.

In [25], da Silva et al. show that a classifier ensemble formed by Multinomial Naive Bayes (MNB), Support Vector Machines (SVM), Random Forest (RF), and Logistic Regression (LR) can improve the classification accuracy on four sentiment datasets used in the investigation, when combined in a majority voting strategy. The diversity in the classifier ensemble is addressed by varying only the base learners, all of them using the same feature representation, such as bag-of-words, the number of positive and negative words, and the number of positive and negative emoticons.

Prusal et al. [73] evaluate seven base classifiers combined with either bagging or boosting ensemble strategies on the sentiment classification of tweets, using only unigrams as features. In bagging, different training partitions are sampled from the original training dataset (with replacement), and a single base learner is trained on each partition. Boosting, on the other hand, iteratively creates the base classifiers, where in each iteration a classifier is trained based on the misclassified instances from the previous iterations. At the end of the process, both ensemble techniques aggregate the resulting classifiers by

averaging the posterior probabilities of each model in the ensemble. In [73], they show that using ensemble strategies such as bagging and boosting can benefit the sentiment classification of tweets, particularly on high dimensional datasets.

In [37], Fersini et al. propose a Bayesian Ensemble Learning approach based on Bayesian Model Averaging (BMA), which uses a greedy backward elimination strategy to select the optimal set of base classifiers. The base candidate classifiers that integrate the search space are a dictionary-based approach (DIC), NB, SVM, Maximum Entropy (ME), and Conditional Random Fields (CRF). The feature space used for learning is the bag-of-words model, except for the DIC approach, which relies on the polarities of words in sentiment lexicons. Interestingly, although the DIC approach presents the lower individual predictive performance on the datasets used in the experimental evaluation, the optimal ensemble provided by BMA always include DIC as one of the base classifiers for all datasets.

Lately, Fersini et al. [38] point out that not only words are important features in detecting the sentiment polarity of tweets, but also some strong signals can help to discriminate the positive messages from the negative ones. In this context, in [38], the combination of the bag-of-words representation of tweets with adjectives, pragmatic particles (emoticons, initialisms for emphatic expressions, and onomatopoeic expressions), and expressive lengthening are investigated independently and as part of an ensemble learning strategy. More precisely, the bag-of-words vectors representing each tweet are expanded with five new features: the number of positive and negative adjectives, the number of positive and negative pragmatic particles and the expressive lengthening of a tweet. In the experimental investigation, they show that using the bag-of-words model expanded with all those expressive signals on an ensemble learning framework (BMA [37]) can lead to a significant improvement in terms of accuracy.

The combination of distinct preprocessing techniques with well-established classification algorithms has been investigated by Lochter et al. [54]. In [54], they propose an ensemble system that performs a grid search to select the best combination between text processing techniques and different classification methods, such as Naive Bayes (NB), SVM, LR, k-Nearest Neighbors (k-NN), and Decision Trees (DT). In [54], they evaluate the predictive power of the ensemble system on nine datasets of tweets. Once their goal is to detect the best combination of text preprocessing techniques and classifiers, they have used a small fixed set of features for each assessed learning method, such as unigrams and the count of positive and negative terms in each tweet.

Emadi and Rahgozar [32] have recently proposed a classifier ensemble approach which combines supervised and unsupervised methods in Twitter sentiment classification. To this end, three supervised machine learning algorithms, such as SVM, NB, and ME are used as base classifiers, each of them fed with unigrams, bigrams, and a combination of both. In addition to those classifiers, an unsupervised NLP-based method is used. The classifiers are chosen based on diversity measures in order to select methods that complement each other. Once the diverse set of classifiers is identified, i.e., classifiers with sufficient diversity, a learning fusion method is applied to assign a polarity orientation for each tweet. In [32], the Choquet Fuzzy Integral (CFI) method is used as a meta-learning strategy, which combines the decision of each classifier.

Araque et al. [3] investigate different combinations of features via ensemble learning and through feature concatenation. They evaluate and compare the predictive performance of these combinations against a supervised baseline model using as features word embeddings trained on a corpus of 1.28M tweets. For the ensemble model, they use as base classifiers six different state-of-the-art sentiment methods [27, 40, 49, 55, 57, 65], each one trained with various yet simple features (e.g., n-grams, POS features, polarity values of words, etc.), in addition to classifiers trained with generic and affective word embeddings, i.e., word vectors trained for general purpose and for the sentiment analysis task, respectively.

Different from ensemble learning methods, which combine the strength of classifiers and features at prediction time, feature concatenation, or feature ensemble, consists in combining different sets of features into a unified set as a preprocessing step to the classification process. Aiming at evaluating the combination of several types of features, Araque et al. [3] have proposed three feature concatenation models. The first one, denoted by M_{SG}, combines a small set of meta-features and generic word embedding vectors. The second type, M_{GA}, combines generic and affective word vectors. Finally, the third, M_{SGA}, consists in the combination of the features included in the first and second models, i.e., meta-features, generic and affective word vectors. In the experimental evaluation, both the ensemble model and the feature concatenation model M_{SG} achieved the best results, with no significant statistical difference between them.

Agarwal et al. [1] have proposed a rich set of meta-features and divided them into three categories, $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{B}$, which represents features whose value is a positive integer (e.g., #hashtags, #positive words, etc.), features whose value is a real number (e.g., polarity score of words in some lexicon), and features whose value is a boolean (e.g., presence of

capitalized text), respectively. Besides, they have adopted unigrams as a baseline. In the experimental evaluation of the proposed set of features, features were added incrementally to the baseline unigram model and they show that the best result is achieved by using all meta-features in combination with the unigrams, through feature concatenation.

In [84], Tang et al. explore the combination of the SSWE embeddings and the state-of-the-art meta-features proposed in [63] through feature concatenation, which has improved the predictive performance from 84.98% to 86.58%. In order to obtain rich sources of information, Vo and Zhang [89] use as features a combination of word vectors trained with two different embedding learning approaches, Google's word2vec [61] and SSWE [84]. To this end, they have trained the embeddings with a large-scale corpus of 5M unlabeled tweets and show that the combination of generic and affective word vectors can benefit the sentiment classification of tweets. Xu et al. [97] investigate the performance of the proposed affective embedding learning system, Emo2vec, by combining the word vectors obtained with their approach and Stanford's GloVe vectors designed in [71], trying to make the feature representation more accurate, since Emo2vec is weak on capturing syntactic and semantic meaning. Table 4.1 presents a summary of the combination methods discussed in this section.

In this chapter, we exploit two distinct strategies for combining the strength of features in Twitter sentiment analysis — feature concatenation and ensemble learning. While the former one consists in concatenating different sets of features into a unique feature vector before the classification process, the last one combines the decisions of a diverse set of classifiers at prediction time.

Diversity is a key point in designing ensemble techniques [15]. Despite the application of ensemble methods in Twitter sentiment classification, as shown in Table 4.1, most works [25, 32, 37, 38, 54] use the same feature representation varying only the classification algorithms, i.e., they use heterogeneous classifiers as base learners [98], except for Araque et al. [3] and Prusa et al. [73]. While Araque et al. [3] adopt as base learners state-of-the-art classifiers from the literature [27, 40, 49, 55, 57, 65], Prusa et al. [73] use the same learning algorithm on different representations of the training data by using bagging and boosting techniques, i.e., they use homogeneous classifiers to form the ensemble [98].

In the experiments conducted in Chapter 3, we showed that different classification strategies benefit from the use of an appropriate set of features. Then, in this chapter, rather than using homogeneous or heterogeneous classifiers to form the ensembles, we address the diversity issue by exploiting a hybrid approach to ensemble learning. Specifi-

Table 4.1: Summary of combination strategies on Twitter sentiment classification, separated by classifier ensemble and feature concatenation approaches.

Reference	Strategy	Feature Representation
	CLASSIFIER ENSEMBLE APPROACHES	
da Silva et al. [25]	Majority voting (MNB, SVM, RF, LR)	bag-of-words + #(+/-) emoticons + #(+/-) words
Fersini et al. [37]	BMA (Dictionary-based approach, NB, SVM, ME, CRF)	bag-of-words
Prusa et al. [73]	Bagging and Boosting (k-NN, DT, SVM, LR Multilayer Perceptron, RBF)	bag-of-words
Fersini et al. [38]	Majority voting and BMA (MNB, SVM, DT, Bayesian Networks)	bag-of-words + #(+/-) adjectives + #(+/-) pragmatic particles + expressive lengthening
Lochter et al. [54]	Weighted majority voting (NB, SVM, LR, k-NN, DT)	unigrams + #(+/-) terms
Araque et al. [3]	Majority voting and Stacking	
	Sentiment140 [40]	unigrams + bigrams + POS
	Stanford CoreNLP [57]	word embeddings
	Sentiment WSD [49]	polarity of words in SentiWordNet
	Vivekn [65]	n-grams
	pattern.en [27]	POS + polarity and subjectivity scores of words + WordNet vocabulary information
	TextBlob [55]	unigrams
	M_G (LR)	generic word embeddings
	M_{GA} (LR)	generic + affective embeddings
	M_{SG} (LR)	generic embeddings + #(+/-) words + #neutral words + #exclamation marks + #question marks + #hashtags + #words in all caps + #elongated words
Emadi and Rahgozar [32]	Choquet Fuzzy Integral (SVM, NB, ME, NLP-based approach)	unigrams + bigrams
Reference	**Strategy**	**Feature Representation**
	FEATURE CONCATENATION APPROACHES	
Agarwal et al. [1]	Support Vector Machines	unigrams + $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{B}$ features [1]
Araque et al. [3]	M_{GA} (LR)	features from M_{GA}
	M_{SG} (LR)	features from M_{SG}
	M_{SGA} (LR)	features from M_{GA} and M_{SG}
Tang et al. [84]	Logistic Regression	affective embeddings + meta-features from [63]
Vo and Zhang [89]	Logistic Regression	generic + affective embeddings
Xu et al. [97]	Logistic Regression	generic + affective embeddings

cally, we use as base learners distinct classification strategies, each one using diverse and disjoint sets of features as input. The base learners used in this investigation are the best classifiers for each feature set identified in Chapter 3, i.e., SVM with n-grams, RF with meta-features, and LR with embedding-based features.

The remainder of this chapter is organized as follows. In Section 4.2, we describe two distinct strategies for combining features, such as feature concatenation and ensemble learning methods. Section 4.3 report the results of the experiments conducted by combining the feature sets investigated in this thesis. Then, in Section 4.4, we present some concluding remarks of this chapter.

4.2 Strategies for Combining Features

This section describes two strategies for combining features, such as feature concatenation and ensemble of classifiers. The simple feature concatenation approach is presented in Subsection 4.2.1, and Subsection 4.2.2 introduces the fundamentals of ensemble learning techniques.

4.2.1 Feature Concatenation

One of the most straightforward methods to combine the strength of different types of features is through feature concatenation. Feature concatenation aims at combining distinct sets of features, represented as feature vectors, into a unified feature set, as a preprocessing step that precedes the classification process, trying to make the feature space more informative to the classifier. Hence, a classifier trained on this combined feature vector may achieve improved predictive performance than stand-alone classifiers that learns from each individual feature set.

An example of feature concatenation is depicted in Figure 4.1. Let $f1$ and $f2$ be two feature vectors of sizes n and m, respectively. As a preprocessing step, $f1$ and $f2$ are concatenated $(f1 + f2)$, generating a resulting feature vector of size $n + m$, which can be used to generate some classifier k.

4.2.2 Ensemble Learning

Another approach to combine the discriminative power of different sets of features is through ensemble classification methods. Ensemble methods are learning algorithms that create a set of classifiers, also called base classifiers or base learners, which are used to classify new instances by combining their decisions in some way [30].

Dietterich [30] points out three intuitive fundamental reasons for building and using an ensemble of classifiers — *statistical*, *computational*, and *representational*, which are, in

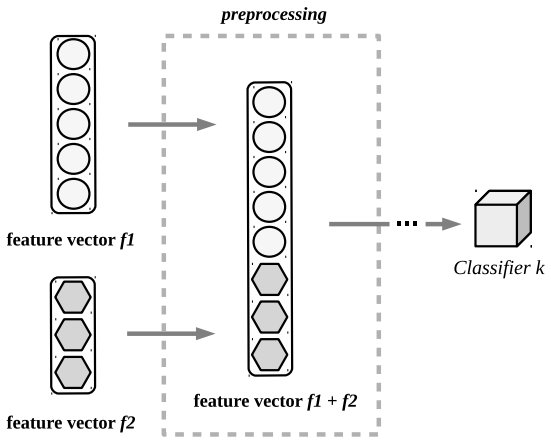

Figure 4.1: Feature concatenation approach.

general, the reasons why learning algorithms fail.

For some learning problem, a classifier can be seen as a hypothesis in a search space $\mathcal{H}$ of hypothesis. The statistical issue emerge when we do not have enough training data. In that case, the learning algorithm can find many hypothesis in $\mathcal{H}$ with equally predictive performance over the limited training data available. Then, combining the predictive power of those hypotesis may give a better aproximation of the unknown true hypothesis rather than selecting one of them.

The second reason is computational. Learning algorithms that apply some kind of local search may get stuck in local optima, hence resulting in a sub-optimal hypothesis. Even if there exists a better unknown hypothesis, it is computationally infeasible for the learning algorithm to find it. In this context, ensembles might be able to approximate to the unknown true hypothesis by combining many sub-optimal hypothesis.

The third and last reason is representational. Considering the set $\mathcal{H}$ of possible hypothesis a learning algorithm can achieve, sometimes, for many learning problems, a learning algorithm cannot find the unknown true hypothesis due to limitations in representing the problem. Then, combining the strength of the hypothesis from $\mathcal{H}$ might enlarge the space of representable solutions.

According to Woods et al. [96], there are two basic approaches to combine the decisions of the base classifiers that form an ensemble, such as classifier selection and classifier fusion. Classifier selection approaches try to identify which base classifier is most likely to be correct in predicting the class for a given instance. In that case, only the prediction of the selected classifier is considered as the final prediction. In classifier fusion strategies, which are the strategies adopted in the experiments conducted in this chapter, the base

classifiers are used in parallel and their predictions are combined according to some rule.

The most popular combination rule is the majority voting one. In the majority voting procedure, each base classifier votes for a class according to its prediction. The class that receives the majority of votes is selected as the final prediction of the ensemble. Figure 4.2 illustrates the majority voting procedure.

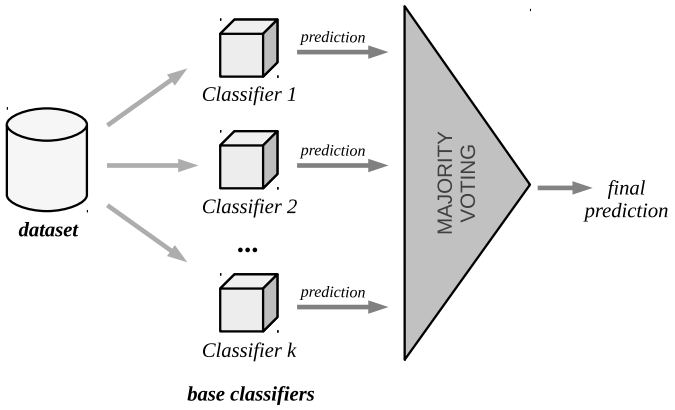

Figure 4.2: Overview of the majority voting procedure.

In many ensemble learning applications, it is very common to use the average of class probability distributions of the base learners as the combination rule. Figure 4.3 presents an example of the average of probabilities rule. Given an instance, each base learner outputs a probability distribution as an estimate of that instance belonging to each class. Then, the output probabilities of each class are averaged, and the class with the higher average of probabilities is taken as the final prediction for that instance.

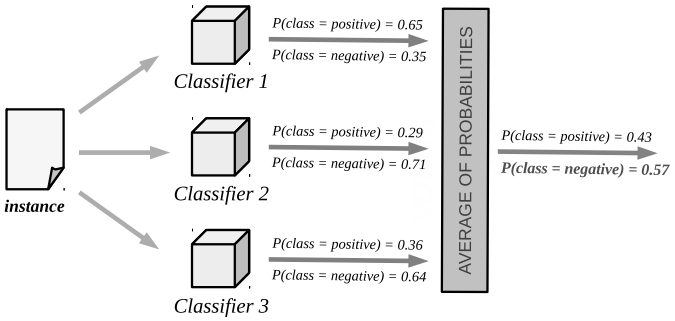

Figure 4.3: Classifier ensemble by the average of probabilites combination rule.

Another way of combining the decisions of base learners is through meta-learning strategies. Stacking, or stacked generalization [95], is an ensemble technique that uses the predictions made by the base learners as inputs for a meta-learning task, as shown in Figure 4.4. First, the base classifiers, also referred to as level-0 models, are trained on the

original feature space, or level-0 data, and their predictions are used as new data (level-1 data) for another learning problem. Then, in the second stage, a meta-learning algorithm, or level-1 generalizer, is trained on the level-1 data to solve this new learning problem [86].

More formally, as defined in [86], given a dataset $\mathcal{S} = \{(x_n, y_n), n = 1, ..., N\}$, or *level-0 data*, where x_n is the n-th instance and y_n is drawn from a discrete set of M classes $\{c_1,...,c_m\}$, randomly split the data into J equal parts $\mathcal{S}_1, ..., \mathcal{S}_J$. Let $\mathcal{S}_j$ and $\mathcal{S}^{-j} = \mathcal{S} - \mathcal{S}_j$ be the test and training sets for the j-th fold of a J-fold cross-validation. Given K learning algorithms, for $k = 1, ..., K$, invoke the k-th algorithm on the training set $\mathcal{S}^{-j}$ to induce a model $\mathcal{M}_k^{-j}$. Those are the *level-0 models*.

For each instance x_n in $\mathcal{S}_j$, the test set for the j-th cross-validation fold, let z_{kn} be the prediction of the k-th model $\mathcal{M}_k^{-j}$ on x_n. At the end of the cross-validation process, let $\hat{\mathcal{S}} = \{(z_{1n},...,z_{Kn}, y_n), n = 1, ..., N\}$ denote the data constructed with the predictions of the K models. This is the *level-1 data*. Then, using some learning algorithm, which is called the *level-1 generalizer*, a model $\hat{\mathcal{M}}$ is derived from the level-1 data. Lastly, to complete the training process, models $\mathcal{M}_k$, $k = 1, ..., K$, are derived using all data in $\mathcal{S}$.

In the classification process, given a new instance, models $\mathcal{M}_k$, $k = 1, ..., K$, produce a vector $(z_1,...,z_K)$, which is input to the level-1 model $\hat{\mathcal{M}}$. Finally, the output of $\hat{\mathcal{M}}$ is considered as the final prediction for that instance.

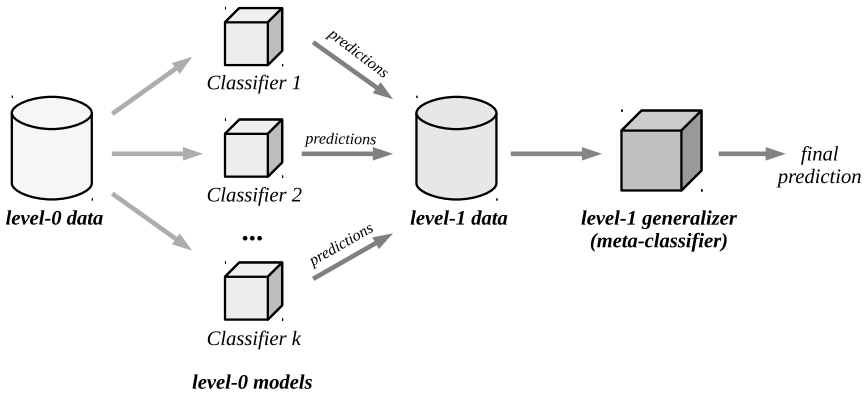

Figure 4.4: Overview of the stacking ensemble strategy.

4.3 Experimental Evaluation

In this section, we explore the combinations of the feature sets evaluated in the previous chapter. Specifically, we address the research questions RQ2 and RQ3 introduced in Chapter 1.

First, after evaluating the individual performance of each feature set (Chapter 3), we examine how they complement each other in the polarity detection task on Twitter by using a simple feature concatenation approach. We report the results of this evaluation in Subsection 4.3.1. Then, in Subsection 4.3.2, we use and evaluate the best individual classifiers as base learners of two distinct ensemble learning techniques, by averaging the class probability distributions of base learners, and by stacking [95], which is a meta-learning technique that uses the probability distributions of base learners as meta-features for a new learning problem.

In the experimental evaluation, we adopted the same experimental settings described in Chapter 3 (Section 3.2). More specifically, we used the twenty-two datasets of tweets presented in Table 3.1. Also, we measured the predictive performance of the strategies for combining features in terms of Accuracy and weighted average F-measure (Equation 3.1), after a 10-fold cross-validation.

4.3.1 Responding to Research Question RQ2

This section presents the results achieved by combining n-grams, meta-features, and embedding-based features through feature concatenation, i.e., by concatenating each feature set into a unique feature vector. Specifically, we address the research question RQ2, as follows:

– RQ2. Can the concatenation of the different features proposed in the literature boost the classification performance in Twitter sentiment analysis?

To respond to this question, we first evaluate each possible combination of feature sets with one classification algorithm at a time (SVM, RF, and LR) to identify the most suitable classifier for each combined situation. More precisely, as a preprocessing step to the classification process, we combine meta-features, n-grams, and w2v-Edin embedding-based features through feature concatenation. As a result, the four following feature vectors are generated: meta-features + n-grams, meta-features + w2v-Edin, n-grams + w2v-Edin, and meta-features + n-grams + w2v-Edin. The results of those evaluations are reported in Table 4.2.

As we can see in Table 4.2, the best results when combining n-grams with any other feature set, i.e., meta-features or embedding-based features (groups $c1$ and $c3$, respectively), were achieved by using SVM, for both Accuracy and F-measure. It may be due to the higher number of n-grams, considering that SVM performed better under the in-

Table 4.2: Accuracies and F-measure scores (%) achieved by evaluating the combination of meta-features, n-grams, and w2v-Edin embeddings through feature concatenation.

Dataset	meta-features + n-grams ($c1$)						meta-features + w2v-Edin ($c2$)					
	Accuracy			*F-measure* (*avg*)			*Accuracy*			*F-measure* (*avg*)		
	SVM	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM	LR	RF
irony	**80.0**	73.8	73.8	**79.3**	71.7	70.1	80.0	**81.5**	70.8	79.3	**80.3**	65.4
sarcasm	**76.1**	70.4	**76.1**	**75.9**	70.1	**75.9**	71.8	73.2	**76.1**	71.8	73.0	**75.9**
aisopos	92.8	**93.5**	89.6	92.7	**93.5**	89.4	92.4	93.5	**93.9**	92.4	93.5	**93.9**
SemEval-Fig	**91.9**	91.3	85.4	**91.1**	90.0	78.6	89.1	**90.7**	85.7	88.9	**90.1**	79.4
sentiment140	**90.0**	88.3	87.5	**90.0**	88.3	87.5	86.6	**89.1**	87.7	86.6	**89.1**	87.7
person	**85.0**	84.1	80.2	**84.7**	83.5	77.0	81.5	**85.4**	82.9	81.6	**85.3**	81.3
hobbit	93.3	**93.5**	91.8	93.3	**93.5**	91.6	91.2	90.8	**92.1**	91.2	90.8	**92.1**
iphone6	**81.8**	80.6	80.5	**81.1**	79.7	78.1	80.6	**82.5**	81.8	80.6	**82.2**	80.3
movie	**88.2**	87.3	84.0	**87.1**	85.0	78.5	88.8	**89.1**	85.0	**88.7**	88.5	80.3
sanders	**87.3**	86.7	85.6	**87.3**	86.7	85.6	84.2	**85.0**	84.7	84.1	**85.0**	84.7
Narr	**90.1**	89.8	89.0	**90.1**	89.8	88.8	89.2	**90.5**	90.4	89.2	**90.5**	90.3
archeage	**89.2**	88.8	86.6	**89.2**	88.8	86.4	86.6	**87.8**	86.6	86.6	**87.8**	86.5
SemEval18	**86.7**	85.9	85.2	**86.6**	85.9	85.1	84.5	**86.5**	84.8	84.5	**86.5**	84.7
OMD	85.0	**85.5**	82.1	85.0	**85.3**	81.2	81.5	**84.1**	83.0	81.4	**84.0**	82.2
HCR	80.9	**81.0**	76.5	**80.0**	79.8	70.2	77.7	**79.0**	76.8	77.2	**78.2**	72.0
STS-gold	**92.3**	91.6	89.6	**92.2**	91.5	89.2	90.5	**91.7**	91.1	90.5	**91.6**	90.9
SentiStrength	**83.2**	82.8	80.7	**83.2**	82.7	80.2	81.4	**83.1**	82.8	81.4	**83.1**	82.7
Target-dependent	84.9	**85.5**	83.8	84.9	**85.5**	83.8	84.0	**84.8**	84.3	84.0	**84.8**	84.3
Vader	93.2	**93.4**	91.6	93.1	**93.3**	91.4	93.4	**93.6**	92.5	93.4	**93.6**	92.3
SemEval13	**88.4**	88.0	81.5	**88.2**	87.6	78.4	86.3	**87.3**	85.7	86.2	**87.1**	84.8
SemEval17	90.0	**90.1**	87.8	**90.0**	**90.0**	87.6	89.4	**89.9**	88.3	89.4	**89.9**	88.2
SemEval16	**88.6**	88.3	81.0	**88.4**	88.0	77.4	**87.6**	87.5	85.7	**87.4**	87.3	84.9
#wins	**15**	7	1	**17**	6	1	1	**18**	3	2	**17**	3
rank sums	**29.5**	38.5	64.0	**28.0**	39.5	64.5	55.5	**27.5**	49.0	52.0	**28.0**	52.0
	{SVM, LR} ≻ {RF}			{SVM, LR} ≻ {RF}			{LR} ≻ {SVM, RF}			{LR} ≻ {SVM, RF}		

Dataset	n-grams + w2v-Edin ($c3$)						meta-features + n-grams + w2v-Edin ($c4$)					
	Accuracy			*F-measure* (*avg*)			*Accuracy*			*F-measure* (*avg*)		
	SVM	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM	LR	RF
irony	**73.8**	70.8	69.2	**70.1**	64.0	59.3	**78.5**	75.4	72.3	**76.4**	71.4	65.1
sarcasm	56.3	62.0	**70.4**	55.9	61.3	**69.7**	69.0	70.4	**71.8**	68.4	69.9	**71.2**
aisopos	**93.5**	**93.5**	88.1	**93.5**	**93.5**	87.9	93.5	**94.6**	89.9	93.5	**94.6**	89.8
SemEval-Fig	**91.3**	89.4	85.4	**90.4**	87.4	78.6	**92.2**	**92.2**	85.4	**91.5**	91.2	78.6
sentiment140	88.0	**88.6**	82.7	88.0	**88.6**	82.7	**90.5**	89.7	86.6	**90.5**	89.7	86.6
person	82.7	**83.1**	73.3	82.0	**82.1**	65.3	85.4	**86.1**	76.3	85.2	**85.7**	70.7
hobbit	**92.9**	92.1	77.8	**92.9**	92.2	73.9	92.5	**93.1**	89.3	92.5	**93.1**	88.8
iphone6	**81.6**	81.0	76.7	**80.7**	79.9	71.4	83.3	**83.5**	77.4	83.0	**83.1**	73.2
movie	**87.7**	86.8	82.0	**85.7**	83.6	73.9	**89.7**	88.2	82.2	**88.9**	86.4	74.3
sanders	**86.3**	86.2	79.0	**86.3**	86.2	78.8	87.5	**88.1**	84.2	87.5	**88.1**	84.1
Narr	88.5	**88.6**	81.4	88.5	**88.6**	80.5	90.5	**90.6**	86.8	90.5	**90.6**	86.5
archeage	89.1	**89.4**	85.0	89.1	**89.4**	84.6	89.6	**90.3**	86.6	89.6	**90.3**	86.2
SemEval18	**84.8**	84.3	76.6	**84.8**	84.2	76.2	**86.5**	86.0	82.6	**86.5**	86.0	82.4
OMD	84.3	**85.1**	77.1	84.1	**84.8**	74.4	85.6	**86.0**	79.2	85.5	**85.8**	77.1
HCR	80.5	**81.3**	73.8	79.5	**80.0**	64.8	81.4	**81.7**	74.0	80.7	80.7	65.3
STS-gold	89.0	**89.5**	74.5	88.8	**89.2**	68.0	**92.3**	92.2	82.0	**92.3**	92.1	79.7
SentiStrength	**81.7**	81.5	70.6	**81.6**	81.4	68.0	**84.3**	84.2	78.8	**84.3**	84.2	77.9
Target-dependent	83.4	**83.7**	78.1	83.4	**83.7**	78.1	85.0	**85.5**	83.6	85.0	**85.5**	83.6
Vader	**89.8**	89.6	75.1	**89.5**	89.3	69.0	93.4	**93.9**	85.2	93.4	**93.8**	83.7
SemEval13	**85.2**	**85.2**	74.1	**84.8**	84.6	64.4	88.1	**88.6**	76.5	88.0	**88.4**	69.6
SemEval17	**89.8**	**89.8**	82.9	**89.8**	89.7	82.1	90.6	**90.8**	86.7	90.6	**90.8**	86.2
SemEval16	**88.3**	88.1	79.7	**88.1**	87.8	79.3	**88.9**	**88.9**	86.8	88.8	88.7	86.6
#wins	**13**	11	1	**13**	9	1	8	**15**	1	7	**12**	1
rank sums	**33.5**	34.5	64.0	**32.5**	35.5	64.0	38.0	**30.0**	64.0	36.5	**31.5**	64.0
	{SVM, LR} ≻ {RF}			{SVM, LR} ≻ {RF}			{SVM, LR} ≻ {RF}			{SVM, LR} ≻ {RF}		

dividual evaluation of the n-gram features, as we have shown in Chapter 3 (Table 3.3). Regarding the combination of meta-features with word embedding features (group $c2$), LR outperformed SVM and RF by a significant margin, for both Accuracy and F-measure. At last, the combination of all feature sets into a unique feature vector was most benefited by using LR (group $c4$). Indeed, the LR algorithm achieved comparable performance to

SVM for the n-gram features (Table 3.3) and the second-best results in the meta-features evaluation (Table 3.4). In both evaluations, LR was ranked as the second-best classifier. Then, combining all feature vectors may have made LR excel on SVM and RF.

After identifying the best classifiers for each concatenated feature vector, we perform a comparative evaluation of their predictive performances. Those results are reported in Tables 4.3 and 4.4. We can notice that the sentiment classification of tweets benefits from the concatenation of all feature sets, i.e., meta-features + n-grams + w2v-Edin (last column), achieving the best overall results in 15 out of the 22 datasets, for both Accuracy and average F-measure. The second-best results were achieved by meta-features + n-grams (fifth column), followed by meta-features + w2v-Edin (sixth column). The least accurate results were achieved by n-grams + w2v-Edin (seventh column). Moreover, all four concatenated feature sets outperformed the three individual classifiers (meta-features, n-grams, and w2v-Edin columns), as shown in the rank sums row.

Table 4.3: Accuracies (%) achieved by combining different feature sets through feature concatenation.

Dataset	***Accuracy***						
	meta-features	***n*-grams**	**w2v-Edin**	**Feature concatenation**			
	■	□	●	■+□	■+●	□+●	■+□+●
	RF	**SVM**	**LR**	**SVM**	**LR**	**SVM**	**LR**
irony	**81.5**	66.2	75.4	80.0	**81.5**	73.8	75.4
sarcasm	**80.3**	50.7	56.3	76.1	73.2	56.3	70.4
aisopos	92.8	87.8	92.8	92.8	93.5	93.5	**94.6**
SemEval-Fig	90.3	91.0	89.1	91.9	90.7	91.3	**92.2**
sentiment140	85.0	84.1	87.7	**90.0**	89.1	88.0	89.7
person	83.6	79.0	81.3	85.0	85.4	82.7	**86.1**
hobbit	91.6	92.9	92.5	**93.3**	90.8	92.9	93.1
iphone6	82.5	77.6	81.6	81.8	82.5	81.6	**83.5**
movie	87.0	84.1	88.6	88.2	**89.1**	87.7	88.2
sanders	84.8	83.0	82.9	87.3	85.0	86.3	**88.1**
Narr	90.3	83.7	89.6	90.1	90.5	88.5	**90.6**
archeage	85.4	86.3	87.0	89.2	87.8	89.1	**90.3**
SemEval18	86.0	80.2	82.8	**86.7**	86.5	84.8	86.0
OMD	79.8	81.2	83.3	85.0	84.1	84.3	**86.0**
HCR	77.5	79.1	78.5	80.9	79.0	80.5	**81.7**
STS-gold	**93.1**	84.0	87.5	92.3	91.7	89.0	92.2
SentiStrength	83.3	73.2	81.2	83.2	83.1	81.7	**84.2**
Target-dependent	83.1	81.4	82.5	84.9	84.8	83.4	**85.5**
Vader	93.0	84.8	89.3	93.2	93.6	89.8	**93.9**
SemEval13	86.9	81.0	83.6	88.4	87.3	85.2	**88.6**
SemEval17	86.5	86.9	87.6	90.0	89.9	89.8	**90.8**
SemEval16	85.4	85.8	86.4	88.6	87.5	88.3	**88.9**
#wins	3	0	0	3	2	0	**15**
rank sums	100.0	139.5	119.0	53.0	71.5	94.0	**37.5**

{meta-features+n-grams+w2v-Edin} $\succ$ {meta-features, n-grams, w2v-Edin, n-grams+w2v-Edin}
{meta-features+n-grams} $\succ$ {meta-features, n-grams, w2v-Edin}
{meta-features+w2v-Edin} $\succ$ {n-grams, w2v-Edin}
{n-grams+w2v-Edin} $\succ$ {n-grams}

Interestingly, concerning the combinations of pairs of feature sets (fifth, sixth and seventh columns), only the concatenation provided by meta-features + n-grams performed significantly better than all individual classifiers (meta-features, n-grams, and w2v-Edin columns), for both Accuracy and F-measure.

Table 4.4: F-measure scores (%) achieved by combining different feature sets through feature concatenation.

Dataset	*F-measure*																						
	average							*positive*							*negative*								
	meta-features ■	***n*-grams** □	**w2v-Edin** ●	**Feature concatenation** ■+□	■+●	□+●■+□+●		**meta-features** ■	***n*-grams** □	**w2v-Edin** ●	**Feature concatenation** ■+□	■+●	□+●■+□+●		**meta-features** ■	***n*-grams** □	**w2v-Edin** ●	**Feature concatenation** ■+□	■+●	□+●■+□+●	
	RF	**SVM**	**LR**	**SVM**	**LR**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**SVM**	**LR**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**SVM**	**LR**	**SVM**	**LR**
irony	**80.7**	52.7	74.2	79.3	80.3	70.1	71.4	**68.4**	0.0	57.9	66.7	66.7	45.2	46.7	87.0	79.6	82.6	85.7	**87.2**	82.8	84.0
sarcasm	**80.1**	48.0	56.1	75.9	73.0	55.9	69.9	**77.4**	58.8	50.8	73.0	68.9	49.2	64.4	**82.5**	38.6	60.8	78.5	76.5	61.7	74.7
aisopos	92.8	87.4	92.8	92.7	93.5	93.5	**94.6**	93.8	90.3	93.9	93.9	94.4	94.5	**95.4**	91.4	83.5	91.3	91.2	92.4	92.1	**93.5**
SemEval-Fig	88.2	89.8	87.7	91.1	90.1	90.4	**91.2**	50.8	60.3	52.1	**65.8**	63.4	63.2	**65.8**	94.6	94.9	93.8	95.4	94.6	95.1	**95.6**
sentiment140	85.0	84.1	87.7	**90.0**	89.1	88.0	89.7	85.0	85.0	87.9	**90.1**	89.3	88.4	89.9	84.9	83.2	87.6	**89.8**	89.0	87.6	89.5
person	82.8	77.8	80.9	84.7	85.3	82.0	**85.7**	89.0	86.1	87.2	89.7	89.8	88.3	**90.5**	67.6	57.4	65.3	72.5	**74.2**	66.4	74.0
hobbit	91.6	93.0	92.5	**93.3**	90.8	92.9	93.1	93.8	94.7	94.5	**95.1**	93.2	94.8	94.9	87.0	89.3	88.5	**89.6**	85.7	89.0	89.3
iphone6	82.2	73.6	81.3	81.1	82.2	80.7	**83.1**	87.8	86.0	87.1	87.5	87.7	87.5	**88.5**	69.3	45.2	67.8	66.4	69.5	65.0	**70.7**
movie	85.2	80.2	87.3	87.1	**88.5**	85.7	86.4	92.5	91.0	93.3	93.1	**93.5**	92.9	93.2	52.3	31.0	60.0	59.8	**65.5**	53.1	55.4
sanders	84.7	83.0	82.9	87.3	85.0	86.3	**88.1**	83.1	82.4	81.4	86.3	83.8	85.2	**87.0**	86.2	83.6	84.2	88.1	86.0	87.2	**89.0**
Narr	90.3	83.7	89.6	90.1	90.5	88.5	**90.6**	92.0	86.6	91.5	91.8	**92.2**	90.4	**92.2**	87.7	79.2	86.7	87.5	88.0	85.6	**88.2**
archeage	85.3	86.4	87.0	89.2	87.8	89.1	**90.3**	81.8	84.3	84.5	87.3	85.4	87.0	**88.4**	87.8	87.9	88.9	90.7	89.5	90.6	**91.7**
SemEval18	86.0	79.9	82.8	**86.6**	86.5	84.8	86.0	84.6	76.6	81.3	**85.4**	85.2	83.1	84.5	87.2	82.8	84.1	**87.7**	87.6	86.3	87.3
OMD	79.2	81.0	83.1	85.0	84.0	84.1	**85.8**	69.5	74.0	76.1	79.4	77.8	77.9	**80.0**	84.9	85.2	87.2	88.3	87.6	87.8	**89.2**
HCR	74.7	77.7	77.3	80.0	78.2	79.5	**80.7**	46.4	55.9	55.6	61.7	58.6	60.4	**62.4**	85.8	86.3	85.8	87.3	85.9	87.0	**87.9**
STS-gold	**93.1**	83.4	87.3	92.2	91.6	88.8	92.1	**88.7**	71.3	78.8	87.2	86.4	81.3	87.1	**95.0**	88.9	91.1	94.5	94.0	92.2	94.4
SentiStrength	83.3	72.7	81.2	83.2	83.1	81.6	**84.2**	85.8	78.5	84.1	85.9	85.7	84.7	**86.7**	79.7	64.4	77.0	79.3	79.4	77.2	**80.6**
Target-dependent	83.1	81.4	82.5	84.9	84.8	83.4	**85.5**	82.9	81.3	82.5	84.9	84.8	83.5	**85.5**	83.3	81.5	82.5	84.9	84.7	83.3	**85.5**
Vader	92.9	83.9	89.2	93.1	93.6	89.5	**93.8**	95.0	89.7	92.5	95.1	95.4	92.8	**95.6**	88.2	71.0	81.8	88.7	89.6	82.3	**89.9**
SemEval13	86.5	78.9	83.1	88.2	87.1	84.8	**88.4**	91.2	88.0	89.0	92.2	91.4	90.2	**92.3**	74.1	54.5	67.5	77.6	75.7	70.3	**77.8**
SemEval17	86.4	86.8	87.5	90.0	89.9	89.8	**90.8**	81.3	82.1	83.1	86.5	86.4	86.2	**87.5**	89.4	89.6	90.2	92.1	92.0	91.9	**92.7**
SemEval16	84.9	85.1	86.1	88.4	87.3	88.1	**88.7**	90.3	90.7	90.9	92.3	91.5	92.2	**92.5**	70.7	70.2	73.2	78.0	75.9	77.0	**78.4**
#wins	3	0	0	3	1	0	**15**	3	0	0	4	2	0	**15**	2	0	0	3	3	0	**14**
rank sums	99.5	141.0	116.5	55.0	70.0	95.5	**38.5**	102.5	139.5	118.5	52.0	70.0	94.5	**39.0**	97.5	140.5	119.0	56.0	69.5	95.0	**38.5**

{meta-features+*n*-grams+w2v-Edin} ≻ {meta-features, *n*-grams, w2v-Edin, *n*-grams+w2v-Edin}
{meta-features+*n*-grams} ≻ {meta-features, *n*-grams, w2v-Edin}
{meta-features+w2v-Edin} ≻ {*n*-grams, w2v-Edin}
{*n*-grams+w2v-Edin} ≻ {*n*-grams}

It is also worth mentioning that the combination of all feature sets is significantly better than all individual classifiers and than n-grams + w2v-Edin. On the other hand, the results achieved by concatenating all feature sets are not significantly better than the meta-features + n-grams and the meta-features + w2v-Edin classifiers. Moreover, the RF classifier fed with meta-features only (meta-features column) achieved an overall performance comparable to the SVM classifier fed with n-grams + w2v-Edin, as we can see in the rank sums row (100.0 and 94.0 in terms of Accuracy, and 99.5 and 95.5 in terms of average F-measure, respectively). Those facts emphasize the predictive power of meta-features and their importance in the context of Twitter sentiment analysis.

4.3.2 Responding to Research Question RQ3

In this section, we present the predictive performance achieved by combining all individual classifiers as base learners of two distinct ensemble strategies. More precisely, we aim at responding to research question RQ3, as follows:

– RQ3. Can the sentiment classification of tweets benefit from the use of ensemble classification strategies having the best classifiers for each feature set as base learners?

For this purpose, we use the best classifiers under the individual evaluation of each feature set, i.e., RF with meta-features, SVM with n-grams, and LR with embedding-based features from w2v-Edin model, as base learners of two ensemble learning strategies formed by: (i) the average of class probabilities and (ii) stacking, as described in Section 4.2.

For the stacking ensemble strategy, we adopted the best individual classifiers for each feature set as the level-0 models to construct the level-1 data, after a 5-fold cross-validation. Furthermore, as suggested in [86], considering that the output of each level-0 model is a set of class probabilities, we used these class probabilities values along with the predicted class value to form the level-1 data, rather than using only the final prediction of each level-0 model. Besides, we used the LR algorithm as the level-1 generalizer.

Tables 4.5 and 4.6 summarize the results (Accuracy and F-measure, respectively). As we can observe, both ensemble strategies (Ensemble column) effectively outperformed all individual classifiers, except for datasets irony and sarcasm. It may be due to the poor performance of the n-gram features on both datasets (66.2% and 50.7% in terms of Accuracy, and 52.7% and 48.0% in terms of average F-measure, respectively). For dataset sarcasm, not only the n-grams performed poorly but also the embedding-based features (56.3% in terms of Accuracy, and 56.1% in terms of average F-measure, respectively).

Also, the stacking strategy (stacking column) achieved the best overall results in 13 out of the 22 datasets, for both Accuracy and average F-measure. Lastly, the Friedman and Nemenyi tests detected that both ensemble strategies are significantly better than all individual classifiers, but there is no significant difference between them.

Table 4.5: Accuracies (%) achieved by combining different feature sets as base learners of ensemble strategies.

Dataset	***Accuracy***				
	meta-features	***n*-grams**	**w2v-Edin**	**Ensemble** avg. prob.	stacking
	RF	**SVM**	**LR**		**LR**
irony	**81.5**	66.2	75.4	75.4	76.9
sarcasm	**80.3**	50.7	56.3	73.2	78.9
aisopos	92.8	87.8	92.8	**93.5**	93.2
SemEval-Fig	90.3	91.0	89.1	**91.9**	**91.9**
sentiment140	85.0	84.1	87.7	**90.8**	89.4
person	83.6	79.0	81.3	84.3	**85.4**
hobbit	91.6	92.9	92.5	**93.1**	92.7
iphone6	82.5	77.6	81.6	83.5	**86.1**
movie	87.0	84.1	88.6	87.5	**89.8**
sanders	84.8	83.0	82.9	86.8	**87.2**
Narr	90.3	83.7	89.6	90.5	**91.0**
archeage	85.4	86.3	87.0	**90.0**	89.7
SemEval18	86.0	80.2	82.8	**87.4**	87.3
OMD	79.8	81.2	83.3	**86.5**	85.9
HCR	77.5	79.1	78.5	**81.5**	**81.5**
STS-gold	93.1	84.0	87.5	91.9	**93.2**
SentiStrength	83.3	73.2	81.2	83.5	**84.2**
Target-dependent	83.1	81.4	82.5	**85.7**	**85.7**
Vader	93.0	84.8	89.3	93.2	**94.2**
SemEval13	86.9	81.0	83.6	87.7	**88.7**
SemEval17	86.5	86.9	87.6	**91.1**	91.0
SemEval16	85.4	85.8	86.4	88.5	**89.1**
#wins	2	0	0	10	**13**
rank sums	76.5	98.0	82.0	40.0	**33.5**

{Ensemble − avg. prob.} $\succ$ {meta-features, *n*-grams, w2v-Edin}
{Ensemble − stacking} $\succ$ {meta-features, *n*-grams, w2v-Edin}

As stated by Dietterich [30], for the predictive performance of an ensemble of classifiers to be better than its base learners, they must be accurate and diverse, i.e., they should make good but different decisions. In this context, we present an analysis of the correlation among the predictions made by each classifier that makes up the ensembles. Precisely, we computed the Pearson correlation coefficient between the outputs (predictions) of each pair of classifiers. The Pearson coefficient ranges from -1 to $+1$, where a value less (greater) than zero indicates a negative (positive) association between the outputs. In that case, for any pair of classifiers, the closer to zero the Pearson coefficient is, the more different (diverse) are the decisions made by them. Table 4.7 shows the Pearson correlation matrices for distinct datasets regarding the predictions made by each classifier.

Table 4.6: F-measure scores (%) achieved by combining different feature sets as base learners of ensemble strategies.

Dataset	*F-measure*														
	average					*positive*					*negative*				
	meta-features	***n*-grams**	**w2v-Edin**	**Ensemble** avg. prob.	**Ensemble** stacking	**meta-features**	***n*-grams**	**w2v-Edin**	**Ensemble** avg. prob.	**Ensemble** stacking	**meta-features**	***n*-grams**	**w2v-Edin**	**Ensemble** avg. prob.	**Ensemble** stacking
	RF	**SVM**	**LR**		**LR**	**RF**	**SVM**	**LR**		**LR**	**RF**	**SVM**	**LR**		**LR**
irony	**80.7**	52.7	74.2	71.4	75.0	**68.4**	0.0	57.9	46.7	57.1	**87.0**	79.6	82.6	84.0	84.2
sarcasm	**80.1**	48.0	56.1	72.8	78.8	**77.4**	58.8	50.8	67.8	76.2	**82.5**	38.6	60.8	77.1	81.0
aisopos	92.8	87.4	92.8	**93.5**	93.1	93.8	90.3	93.9	**94.5**	94.1	91.4	83.5	91.3	**92.2**	91.8
SemEval-Fig	88.2	89.8	87.7	91.1	**91.2**	50.8	60.3	52.1	65.8	**66.7**	94.6	94.9	93.8	**95.4**	**95.4**
sentiment140	85.0	84.1	87.7	**90.8**	89.4	85.0	85.0	87.9	**90.9**	89.7	84.9	83.2	87.6	**90.8**	89.1
person	82.8	77.8	80.9	83.6	**85.1**	89.0	86.1	87.2	89.4	**90.0**	67.6	57.4	65.3	69.3	**72.9**
hobbit	91.6	93.0	92.5	**93.1**	92.7	93.8	94.7	94.5	**94.9**	94.6	87.0	89.3	88.5	**89.4**	88.8
iphone6	82.2	73.6	81.3	82.8	**85.8**	87.8	86.0	87.1	88.7	**90.3**	69.3	45.2	67.8	69.0	**75.5**
movie	85.2	80.2	87.3	85.2	**89.1**	92.5	91.0	93.3	92.9	**94.0**	52.3	31.0	60.0	50.0	**66.7**
sanders	84.7	83.0	82.9	86.8	**87.2**	83.1	82.4	81.4	85.6	**86.1**	86.2	83.6	84.2	87.9	**88.1**
Narr	90.3	83.7	89.6	90.5	**90.9**	92.0	86.6	91.5	92.2	**92.5**	87.7	79.2	86.7	88.0	**88.6**
archeage	85.3	86.4	87.0	**90.0**	89.7	81.8	84.3	84.5	**87.9**	87.8	87.8	87.9	88.9	**91.4**	91.1
SemEval18	86.0	79.9	82.8	**87.3**	87.2	84.6	76.6	81.3	86.0	**86.1**	87.2	82.8	84.1	**88.5**	88.2
OMD	79.2	81.0	83.1	**86.2**	85.7	69.5	74.0	76.1	**80.0**	**80.0**	84.9	85.2	87.2	**89.8**	89.1
HCR	74.7	77.7	77.3	79.4	**80.6**	46.4	55.9	55.6	57.2	**62.5**	85.8	86.3	85.8	**88.2**	87.7
STS-gold	**93.1**	83.4	87.3	91.8	**93.1**	88.7	71.3	78.8	86.3	**88.8**	95.0	88.9	91.1	94.2	**95.1**
SentiStrength	83.3	72.7	81.2	83.5	**84.2**	85.8	78.5	84.1	86.2	**86.6**	79.7	64.4	77.0	79.6	**80.8**
Target-dependent	83.1	81.4	82.5	**85.7**	**85.7**	82.9	81.3	82.5	85.5	**85.6**	83.3	81.5	82.5	**85.8**	**85.8**
Vader	92.9	83.9	89.2	93.1	**94.2**	95.0	89.7	92.5	95.2	**95.8**	88.2	71.0	81.8	88.4	**90.5**
SemEval13	86.5	78.9	83.1	87.2	**88.6**	91.2	88.0	89.0	91.8	**92.3**	74.1	54.5	67.5	74.7	**78.6**
SemEval17	86.4	86.8	87.5	**91.1**	90.9	81.3	82.1	83.1	87.6	**87.7**	89.4	89.6	90.2	**93.1**	92.9
SemEval16	84.9	85.1	86.1	88.2	**88.9**	90.3	90.7	90.9	92.4	**92.6**	70.7	70.2	73.2	76.9	**79.0**
#wins	3	0	0	8	**13**	2	0	0	5	**16**	2	0	0	10	**12**
rank sums	75.5	98.0	81.5	42.0	**33.0**	78.5	96.5	80.0	44.5	**30.5**	71.5	100.0	82.5	42.0	**34.0**

{Ensemble − avg. prob.} ≻ {meta-features, *n*-grams, w2v-Edin}
{Ensemble − stacking} ≻ {meta-features, *n*-grams, w2v-Edin}

Table 4.7: Pearson correlation matrices for the predictions made on distinct datasets by using the meta-features (RF), n-grams (SVM), and w2v-Edin (LR) classifiers.

	aisopos		**SemEval-Fig**		**sentiment140**		**person**	
	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**
meta-features	0.7997	0.8655	0.6666	0.4567	0.5522	0.6994	0.3817	0.5815
***n*-grams**	–	0.7803	–	0.6233	–	0.6508	–	0.5233

	hobbit		**iphone6**		**movie**		**sanders**	
	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**
meta-features	0.8580	0.8127	0.4577	0.5926	0.2655	0.5316	0.5967	0.6614
***n*-grams**	–	0.8796	–	0.4690	–	0.3217	–	0.6118

	Narr		**archeage**		**SemEval18**		**OMD**	
	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**
meta-features	0.6418	0.7939	0.6460	0.6738	0.5725	0.6847	0.4479	0.5506
***n*-grams**	–	0.6258	–	0.7222	–	0.5371	–	0.6204

	HCR		**STS-gold**		**SentiStrength**		**Target-dependent**	
	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**
meta-features	0.4235	0.4065	0.6193	0.7404	0.4576	0.6247	0.5961	0.6511
***n*-grams**	–	0.4650	–	0.5638	–	0.4507	–	0.6741

	Vader		**SemEval13**		**SemEval17**		**SemEval16**	
	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**	***n*-grams**	**w2v-Edin**
meta-features	0.6232	0.7321	0.4360	0.5951	0.6337	0.6929	0.5426	0.6366
***n*-grams**	–	0.5999	–	0.3978	–	0.6823	–	0.6104

We can note that, in general, the predictions made by the base classifiers are sufficiently uncorrelated, leading to improved predictive performance of the ensemble strategies for most datasets. For example, analyzing the correlation matrix for dataset HCR, we see that the correlations between the predictions of each pair of classifiers are sufficiently low. Besides, as shown in Tables 4.5 and 4.6, each classifier has achieved competitive results for this dataset (77.5, 79.1, and 78.5% in terms of Accuracy, and 74.7, 77.7, and 77.3% in terms of average F-measure). As a result, the predictive performance achieved by ensembling them with the stacking technique effectively outperformed the best individual classifier up to 2.4 and 2.9% in terms of Accuracy and average F-measure, respectively.

Similarly, for dataset OMD, we can observe that the low correlations between the predictions made by the base learners, along with their fair accuracies (79.8, 81.2, and 83.3% in terms of Accuracy, and 79.2, 81.0, and 83.1% in terms of average F-measure, respectively), may lead to improved ensemble performance, i.e., 86.5 and 86.2% for the avg. prob. ensemble in terms of Accuracy and average F-measure, respectively. As compared to the best base model (83.3 and 83.1% in terms of Accuracy and average F-

measure, respectively), this represents a gain in Accuracy of 3.2%, and of 3.1% in terms of average F-measure. We can see a similar effect on datasets person, iphone6, SemEval18, and SemEval13.

Interestingly, for dataset STS-gold, it is possible to see that although the correlation coefficients between meta-features and n-grams, and between n-grams and w2v-Edin base classifiers are moderately low (0.6193 and 0.5638, respectively), the n-gram classifier does not seem to be as accurate as the meta-features one. More specifically, while the meta-features classifier achieved a classification accuracy of 93.1% for both Accuracy and average F-measure, the n-gram classifier achieved 84.0 and 83.4% only, in terms of Accuracy and average F-measure. It may be that for this reason, the ensemble strategies did not achieve meaningful results for dataset STS-gold. As can be seen in Table 4.5, the Accuracy achieved by the best ensemble classifier is 93.2% (stacking), which represents a gain of only 0.1% over the best base classifier (meta-features). Regarding average F-measure, the best ensemble classifier (stacking) and the best individual classifier (meta-features) achieved the same result (i.e., 93.1%). We can see a similar effect on dataset Narr.

For dataset hobbit, even though all individual classifiers have achieved very high and competitive results (91.6, 92.9, and 92.5% in terms of Accuracy, and 91.6, 93.0, and 92.5 in terms of average F-measure), the correlation coefficients between any pair of classifiers are greater than 0.8, which means that their predictions are very similar to each other. Hence, there is no sufficient diversity among the base classifiers that make up the ensembles. This may have led the ensemble strategies to achieve rather comparable performances to the best individual base learner, i.e., 93.1% (avg. prob.) and 92.7% (stacking), for both Accuracy and average F-measure, against 92.9 and 93.0% (n-gram classifier), in terms of Accuracy and average F-measure, respectively.

To get a sense of how diversity is relevant when choosing the base learners of an ensemble model, we show that the predictive performance of the ensemble can be improved if we select different base classifiers by leveraging the Pearson coefficients between their predictions. For example, regarding dataset hobbit, Table 4.8 shows that the predictive performances of the ensembles are improved up to 1.0% for both Accuracy and average F-measure, by switching from the w2v-Edin classifier to the fastText one. Indeed, as shown in Table 4.9, analyzing the correlation coefficients among the base classifiers of this new ensemble model (0.8580, 0.7464, and 0.7661), we notice that they are lower than the coefficients of the base learners that form the original ensemble model, i.e., the meta-features, n-grams, and w2v-Edin classifiers (0.8580, 0.8127, and 0.8796).

Table 4.8: Results achieved by combining different feature sets as base classifiers of an ensemble strategy for dataset hobbit.

Accuracy							
meta-features	***n*-grams**	**w2v-Edin**	**fastText**	■ □ ●		■ □ ○	
■	□	●	○	**avg. prob.**	**stacking**	**avg. prob.**	**stacking**
RF	**SVM**	**LR**	**LR**		**LR**		**LR**
91.6	92.9	92.5	91.0	93.1	92.7	**94.1**	**93.7**

F-measure **(*average*)**							
meta-features	***n*-grams**	**w2v-Edin**	**fastText**	■ □ ●		■ □ ○	
■	□	●	○	**avg. prob.**	**stacking**	**avg. prob.**	**stacking**
RF	**SVM**	**LR**	**LR**		**LR**		**LR**
91.6	93.0	92.5	90.8	93.1	92.7	**94.1**	**93.7**

Table 4.9: Pearson correlation matrix for the predictions made on dataset hobbit by using the meta-features (RF), *n*-grams (SVM), and fastText (LR) classifiers.

	***n*-grams**	**fastText**
meta-features	0.8580	0.7464
***n*-grams**	–	0.7661

The result of the previous experiment gives us evidence that selecting the most accurate classifiers as members of an ensemble model does not ensure higher predictive performances. To confirm this hypothesis, we performed an experiment to test whether the predictive performance of an ensemble is improved when replacing the least accurate base classifier for a more accurate one. The result is presented in Table 4.10.

As we can observe in Table 4.10, regarding dataset SemEval18, switching the least accurate classifier, i.e., the *n*-gram classifier, to the fastText one, which is the second-best embedding-based classifier, the predictive performance of the best ensemble (avg. prob.) drops from 87.4 to 85.7% in terms of Accuracy, and from 87.3 to 85.7% in terms of average F-measure. We can see a similar effect regarding the stacking ensemble. Analyzing the correlation coefficients among the base classifiers of this new ensemble, as shown in Table 4.11, we can see that their predictions are much more correlated than the predictions of the base classifiers from the original ensemble (i.e., 0.5725, 0.6847, 0.5371).

4.3.3 Comparing Combination Methods

In this section, we perform a comparison between the combination methods exploited in this study, such as feature concatenation and ensemble of classifiers.

Table 4.10: Results achieved by combining different feature sets as base classifiers of an ensemble strategy for dataset SemEval18.

Accuracy							
meta-features	***n*-grams**	**w2v-Edin**	**fastText**	■ □ ●		■ ● ○	
■	□	●	○	**avg. prob.**	**stacking**	**avg. prob.**	**stacking**
RF	**SVM**	**LR**	**LR**		**LR**		**LR**
86.0	80.2	82.8	81.2	**87.4**	**87.3**	85.7	86.6

***F-measure* (*average*)**							
meta-features	***n*-grams**	**w2v-Edin**	**fastText**	■ □ ●		■ ● ○	
■	□	●	○	**avg. prob.**	**stacking**	**avg. prob.**	**stacking**
RF	**SVM**	**LR**	**LR**		**LR**		**LR**
86.0	79.9	82.8	81.2	**87.3**	**87.2**	85.7	86.5

Table 4.11: Pearson Correlation matrix for the predictions made on dataset SemEval18 by using the meta-features (RF), w2v-Edin (LR), and fastText (LR) classifiers.

	w2v-Edin	**fastText**
meta-features	0.6847	0.6397
w2v-Edin	–	0.6796

Tables 4.12 and 4.13 presents the comparison between the feature concatenation method, the ensemble strategy formed by the average of probabilities rule, and the stacking ensemble method, in terms of Accuracy and F-measure, respectively. We can see that the stacking ensemble strategy outperformed the other two methods in 11 out of the 22 datasets in terms of Accuracy, and in 12 out of the 22 datasets in terms of average F-measure. Nevertheless, regarding the overall analysis, the feature concatenation method achieved a comparable performance to stacking, as shown in the rank sums row.

Disregarding datasets irony and sarcasm, which the best performances were achieved by using the RF with meta-features classifier, it seems that smaller datasets, such as aisopos, SemEval-Fig, person, hobbit, and sanders have benefited from the feature concatenation approach, whilst larger datasets, such as STS-gold, Target-dependent, Vader, SemEval13, SemEval17, and SemEval16, have achieved higher predictive performances by using the ensemble strategies.

Regarding the differences among the results achieved by the combination methods, the Friedman test did not detect any significant statistical difference among them.

Table 4.12: Comparison among the results achieved by evaluating distinct strategies for combination in terms of Accuracy (%).

	Accuracy		
	Feature concat.	**Ensemble learning**	**Ensemble learning**
Dataset	meta-features + *n*-grams + w2v-Edin	meta-features (RF) *n*-grams (SVM) w2v-Edin (LR)	meta-features (RF) *n*-grams (SVM) w2v-Edin (LR)
	LR	**(avg. prob.)**	**LR (stacking)**
irony	75.4	75.4	**76.9**
sarcasm	70.4	73.2	**78.9**
aisopos	**94.6**	93.5	93.2
SemEval-Fig	**92.2**	91.9	91.9
sentiment140	89.7	**90.8**	89.4
person	**86.1**	84.3	85.4
hobbit	**93.1**	**93.1**	92.7
iphone6	83.5	83.5	**86.1**
movie	88.2	87.5	**89.8**
sanders	**88.1**	86.8	87.2
Narr	90.6	90.5	**91.0**
archeage	**90.3**	90.0	89.7
SemEval18	86.0	**87.4**	87.3
OMD	86.0	**86.5**	85.9
HCR	**81.7**	81.5	81.5
STS-gold	92.2	91.9	**93.2**
SentiStrength	**84.2**	83.5	**84.2**
Target-dependent	85.5	**85.7**	**85.7**
Vader	93.9	93.2	**94.2**
SemEval13	88.6	87.7	**88.7**
SemEval17	90.8	**91.1**	91.0
SemEval16	88.9	88.5	**89.1**
#wins	8	6	**11**
rank sums	42.0	50.0	**40.0**

4.4 Summary

In this chapter, we performed a comparative evaluation between two distinct strategies for combining the predictive power of distinct feature sets, such as feature concatenation and ensemble learning. The experimental evaluation conducted in this chapter addressed the research questions RQ2 and RQ3, discussed in Chapter 1.

To respond to research question RQ2 — "*Can the concatenation of the different features proposed in the literature boost the classification performance in Twitter sentiment analysis?*", we used a simple feature concatenation approach to combine those distinct feature sets into a unique feature vector, as a preprocessing step to the classification process. Our results showed that the sentiment classification of tweets benefits from the combination of all feature sets (meta-features + *n*-grams + embedding-based features). Another interesting finding is that, regarding the concatenation of pairs of feature sets, only the combination provided by meta-features + *n*-grams performed significantly better than all individual feature sets.

Table 4.13: Comparison among the results achieved by evaluating distinct strategies for combination in terms of F-measure (%).

Dataset	*F-measure*								
	average			*positive*			*negative*		
	Feature concat.	**Ensemble learning**	**Ensemble learning**	**Feature concat.**	**Ensemble learning**	**Ensemble learning**	**Feature concat.**	**Ensemble learning**	**Ensemble learning**
	meta-features + n-grams + w2v-Edin	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features + n-grams + w2v-Edin	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features + n-grams + w2v-Edin	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features (RF) n-grams (SVM) w2v-Edin (LR)
	LR	**(avg. prob.)**	**LR (stacking)**	**LR**	**(avg. prob.)**	**LR (stacking)**	**LR**	**(avg. prob.)**	**LR (stacking)**
irony	71.4	71.4	**75.0**	46.7	46.7	**57.1**	84.0	84.0	**84.2**
sarcasm	69.9	72.8	**78.8**	64.4	67.8	**76.2**	74.7	77.1	**81.0**
aisopos	**94.6**	93.5	93.1	**95.4**	94.5	94.1	**93.5**	92.2	91.8
SemEval-Fig	**91.2**	91.1	**91.2**	65.8	65.8	**66.7**	**95.6**	95.4	95.4
sentiment140	89.7	**90.8**	89.4	89.9	**90.9**	89.7	89.5	**90.8**	89.1
person	**85.7**	83.6	85.1	**90.5**	89.4	90.0	**74.0**	69.3	72.9
hobbit	**93.1**	**93.1**	92.7	**94.9**	**94.9**	94.6	89.3	**89.4**	88.8
iphone6	83.1	82.8	**85.8**	88.5	88.7	**90.3**	70.7	69.0	**75.5**
movie	86.4	85.2	**89.1**	93.2	92.9	**94.0**	55.4	50.0	**66.7**
sanders	**88.1**	86.8	87.2	**87.0**	85.6	86.1	**89.0**	87.9	88.1
Narr	90.6	90.5	**90.9**	92.2	92.2	**92.5**	88.2	88.0	**88.6**
archeage	**90.3**	90.0	89.7	**88.4**	87.9	87.8	**91.7**	91.4	91.1
SemEval18	86.0	**87.3**	87.2	84.5	86.0	**86.1**	87.3	**88.5**	88.2
OMD	85.8	**86.2**	85.7	80.0	80.0	80.0	89.2	**89.8**	89.1
HCR	**80.7**	79.4	80.6	62.4	57.2	**62.5**	87.9	**88.2**	87.7
STS-gold	92.1	91.8	**93.1**	87.1	86.3	**88.8**	94.4	94.2	**95.1**
SentiStrength	**84.2**	83.5	**84.2**	**86.7**	86.2	86.6	80.6	79.6	**80.8**
Target-dependent	85.5	**85.7**	**85.7**	85.5	85.5	**85.6**	85.5	**85.8**	**85.8**
Vader	93.8	93.1	**94.2**	95.6	95.2	**95.8**	89.9	88.4	**90.5**
SemEval13	88.4	87.2	**88.6**	**92.3**	91.8	**92.3**	77.8	74.7	**78.6**
SemEval17	90.8	**91.1**	90.9	87.5	87.6	**87.7**	92.7	**93.1**	92.9
SemEval16	88.7	88.2	**88.9**	92.5	92.4	**92.6**	78.4	76.9	**79.0**
#wins	8	6	**12**	7	2	**14**	5	7	**11**
rank sums	42.0	50.5	**39.5**	43.5	53.0	**34.0**	43.5	48.5	**40.0**

We also aimed at addressing the research question RQ3 — "*Can the sentiment classification of tweets benefit from the use of ensemble classification strategies having the best classifiers for each feature set as base learners?*". To this end, we used the best classifiers under the individual evaluation of each feature set, identified in Chapter 3, as base learners of two ensemble learning strategies formed by the average of probabilities combination rule and stacking. We showed that, although both ensemble strategies performed significantly better than all individual classifiers, the stacking ensemble strategy achieved the best overall results.

In the next chapter, concerning the data sparsity problem in supervised machine learning applications, in which tweets are usually represented by terms from a vocabulary of uncommon and infrequent words, we present an enrichment approach to Twitter sentiment analysis. The proposed approach uses the prior polarity information conveyed by emoticons, as well as the synonymy relation among words in existing lexicon resources to increase the knowledge of the naturally sparse Twitter data.

Chapter 5

An Enrichment Approach to Twitter Sentiment Analysis

5.1 Introduction

An important factor that can affect the overall performance of machine learning classifiers is related to data sparsity, especially when dealing with short texts [75]. Regarding supervised learning approaches, the textual data are often represented by its vocabulary, that is, the different words that appear in the corresponding corpus.

As described in Chapter 2, the most common representation of textual data is the bag-of-words or unigram model, in which each word of a document is considered as a feature. In general, the feature space is represented by a binary feature vector indicating whether each word of the vocabulary occurs in the document or not. In that case, the values 0 and 1 represent the absence and presence of each word in the document, respectively.

An accurate classification may be even more difficult to achieve if the document size is limited to a small number of characters, as in tweets. The 140-character limit in tweets leads Twitter users to refer to the same concept with a large variety of short and irregular forms, resulting in the low frequency of words and, as a consequence, in the data sparsity problem [58]. As most values in the training feature vector of tweets is zero, it prevents the classifier to correctly learn how to assign a sentiment class for unseen tweets.

In this context, in order to address the research question RQ4, introduced in Chapter 1, we propose an enrichment approach to Twitter sentiment analysis, which uses the semantic relationships between words in existing lexicon resources, intending to increase the inherent knowledge of the sparse data to be classified.

The remainder of this chapter is organized as follows. In Section 5.2, we describe the enrichment approach proposed in this thesis. The experimental evaluation of the proposed approach is presented in Section 5.3. Then, in Section 5.4, we present a summary of the chapter.

5.2 Description of the Proposed Approach

The enrichment approach proposed in this thesis consists of two complementary methods. More specifically, we propose to enrich the sparse representation of tweets by exploiting: (i) the synonymy relation among words, and (ii) the prior polarity information conveyed by emoticons. These methods are described in Subsections 5.2.1 and 5.2.2, respectively.

5.2.1 Synonymy Relation Among Words

The enrichment approach aims at enriching the natural sparse representation of tweets, by incorporating into them more useful terms that could capture sentiment clues, attempting to make the tweets more informative to the classifier. For this purpose, we take advantage of the existing knowledge represented by the vocabulary of the data, in particular the semantic relations between the large number of terms in this vocabulary, defined in some lexicon resource.

In order to investigate the feasibility of the first enrichment method, we use WordNet [36] as the lexicon resource. WordNet is a well-referenced lexicon database for the English language, which encodes distinct types of semantic relations, such as synonym relations of different part-of-speech categories (nouns, verbs, adjectives, and adverbs), the super-subordinate relations of nouns, also known as "is a" relation (hyperonymy or hyponymy), antonym relations of adjectives, and hierarchical relations of verbs. Although WordNet encodes many kinds of semantic relations, we focus on the synonymy relations as the semantic information used to enrich the sparse representation of tweets.

In WordNet, each entry refers to a set of terms belonging to the same part-of-speech and having the same semantic meaning. This set of terms is referred to as synset, i.e., a set of synonyms, since all terms belonging to the same synset are regarded as synonyms. Moreover, the meaning of a particular synset is referred to as sense. For example, Table 5.1 presents all synsets and senses for the term *cold* in WordNet. As we can see, the term *cold* can be employed in 16 distinct contexts, presenting 13 different semantic meanings as adjective, and 3 meanings as noun. We can also notice that, as adjective, the term *cold*

can be used as synonym of the terms *moth-eaten*, *dusty*, *stale*, *frigid*, *insensate*, *inhuman*, and *cold-blooded*, depending on the context. Similary, as noun, the term *cold* can be used as synonym of *common cold*, *low temperature*, *frigidness*, *frigidity*, and *coldness*.

Table 5.1: Synsets of the word *cold* in WordNet.

#	**POS**	**Synset**	**Sense**
1	ADJ	cold	*having a low or inadequate temperature or feeling a sensation of coldness*
2	ADJ	cold	*extended meanings; especially of psychological coldness; without human warmth or emotion*
3	ADJ	cold	*having lost freshness through passage of time*
4	ADJ	cold	(*color*) *giving no sensation of warmth*
5	ADJ	cold	*marked by errorless familiarity*
6	ADJ	cold moth-eaten dusty stale	*lacking originality or spontaneity*; *no longer new*
7	ADJ	cold	*so intense as to be almost uncontrollable*
8	ADJ	cold frigid	*sexually unresponsive*
9	ADJ	cold insensate inhuman cold-blooded	*without compunction or human feeling*
10	ADJ	cold	*feeling or showing no enthusiasm*
11	ADJ	cold	*unconscious from a blow or shock or intoxication*
12	ADJ	cold	*of a seeker; far from the object sought*
13	ADJ	cold	*lacking the warmth of life*
14	NOUN	cold common cold	*a mild viral infection involving the nose and respiratory passages*
15	NOUN	cold low temperature frigidness frigidity coldness	*the absence of heat*
16	NOUN	cold coldness	*the sensation produced by low temperatures*

Based on the observation that a particular term can be employed with different parts-of-speech, in the proposed approach, we only consider two terms as synonyms if they are from the same part-of-speech category. This criterion helps avoiding to treat terms such as *good* and *beneficial* as synonyms, for example, if the former is being employed as noun.

The general idea behind this first enrichment method is as follows. Given a tweet, we aim at augmenting its feature representation by leveraging the synonymy relations among its original terms and the existing terms in the corresponding vocabulary, as an attempt to enrich its semantic meaning, and making it more informative to the classifier.

A brief example of this method is shown in Figure 5.1, considering the tweet "*I am very irritated tonight*". As a preprocessing step, each unique term that appears in the tweets of the corpus are retrieved, in order to generate the vocabulary of the referred corpus. As we can observe, before the enrichment, only the original terms *tonight*, *irritated*, and *very* of the corresponding tweet are included on its feature vector. Since the terms *I* and *am* are regarded as stopwords, they are not extracted as features in the vocabulary construction step. Then, after the enrichment process, the terms *really* and *annoyed* of the vocabulary are also incorporated as features, since these terms are synonyms of the original terms *very* and *irritated* from the tweet, respectively.

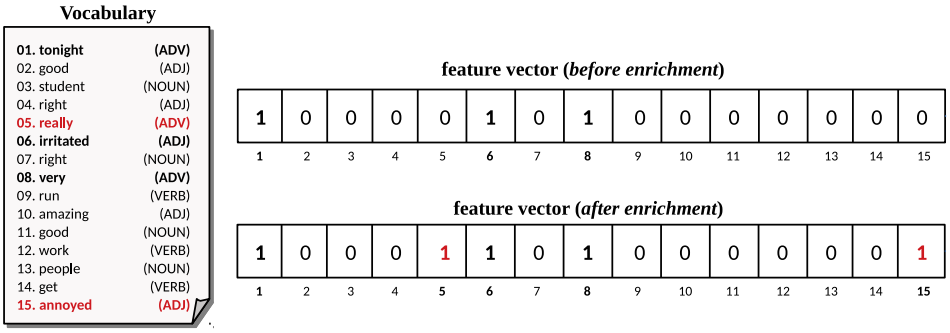

Figure 5.1: Example of the enrichment through the synonymy relation among words.

5.2.2 Prior Polarity Information of Emoticons

As stated by Go et al. [40], obtaining manually labeled data to train classifiers is time-consuming. Then, they proposed to use tweets with emoticons to train classifiers, relying on the prior polarity information conveyed by those emoticons. For example, if a tweet contains :), it is taken as a positive tweet. In [40], this type of label is called noisy label, and they show that it is an effective way to obtain labeled data to train sentiment classifiers.

In this context, we also take advantage of the presence of emoticons in tweets to enrich their sparse representation. More specifically, if a tweet contains a positive (negative) emoticon, we enrich its vector representation with positive (negative) words existing in the vocabulary. To this end, rather than relying on large sentiment lexicons to obtain the polarity of each word in the vocabulary, which can be computationally expensive,

we use only a small fixed set of positive and negative words, proposed by Turney and Littman [87].

In [87], Turney and Littman have successfully proposed to infer the semantic orientation (i.e., polarity) of words in a given corpus from their statistical association with a fixed set of positive and negative paradigm words. Precisely, the semantic orientation of a given word is calculated from the strength of its association with a set of positive words, minus the strength of its association with a set of negative words. For this purpose, they established a set of seven positive and seven negative paradigm words, P^P and P^N, respectively, carefully chosen from their lack of sensitivity to context, i.e., they are positive or negative in almost all contexts. These paradigm words are:

$$P^P = \{good,\ nice,\ excellent,\ positive,\ fortunate,\ correct,\ superior\} \text{ and}$$
$$P^N = \{bad,\ nasty,\ poor,\ negative,\ unfortunate,\ wrong,\ inferior\}.$$

It is worth mentioning that these sets of words consist of opposing pairs of words.

The main idea behind this second enrichment method is as follows. Given a tweet, if it contains a positive (negative) emoticon, we enrich its representation by adding to it any positive (negative) words from P^P (P^N), existing in the vocabulary. In this work, we used the list of positive and negative emoticons compiled by Agarwal et al. [1].

An example of this method is illustrated in Figure 5.2. Given the tweet "*The people at my work are amazing :)*", before the enrichment, only the original terms *people*, *work*, and *amazing* of that tweet are added to its feature vector (the words *The*, *at*, *my*, and *are* are stopwords and hence removed). Then, after the enrichment process, the presence of the positive emoticon *:)* make the positive paradigm words from P^P, *good* and *excellent* of the existing vocabulary be also added as features of that tweet.

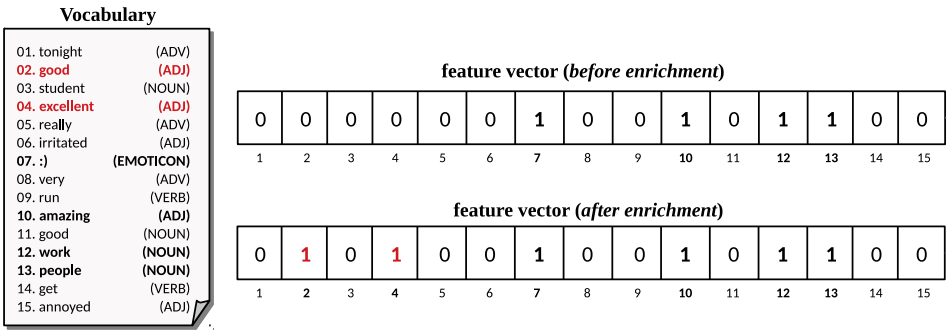

Figure 5.2: Example of the enrichment by leveraging the prior polarity information of emoticons.

5.2.3 Formal Definition

The enrichment approach proposed in this thesis consists of the combination of the two enrichment methods described in Subsections 5.2.1 and 5.2.2. More formally, the general enrichment approach can be described as follows.

1. *Let $D = \{T_1, ..., T_n\}$ be a dataset consisting of n tweets.*

2. *Let $V = \{F_1, ..., F_m\}$ be a set of m unique terms or features, extracted from D, namely its vocabulary.*

3. *Let $T = \{W_1, ..., W_k\}$ be a set of k terms that represents the tweet T, where $T \in D$ and $T \subset V$.*

4. *Let P^P and P^N be a set of positive and negative paradigm words, respectively.*

5. *Let E^P and E^N be a set of positive and negative emoticons, respectively.*

6. *Given a tweet T_i ($1 \leq i \leq n$), for each term $W_j \in T_i$:*

    (a) *The synonyms of W_j are retrieved from vocabulary V, by using WordNet.*

    (b) *Let $S = \{S_1, ..., S_q\}$ be the set of q synonyms of term W_j, where $S \subset V$.*

    (c) *For each synonym S_z ($1 \leq z \leq q$), if $S_z \notin T_i$ and if both W_j and S_z have the same part-of-speech, then S_z is incorporated as a feature of T_i.*

    (d) *If $W_j \in E^P$, $\forall p^P \in P^P$, if $p^P \subset V$ and $p^P \notin T_i$, then p^P is incorporated as a feature of T_i.*

    (e) *If $W_j \in E^N$, $\forall p^N \in P^N$, if $p^N \subset V$ and $p^N \notin T_i$, then p^N is incorporated as a feature of T_i.*

In line 6 of the formal definition of the enrichment approach, items (a), (b), and (c) describe the enrichment through the synonymy relation among words, while items (d) and (e) describe the enrichment via the prior polarity information conveyed by emoticons.

5.3 Experimental Evaluation

This section presents the experiments conducted to evaluate the enrichment approach proposed in this thesis. First, in Subsection 5.3.1, we report the computational results of the proposed approach, as well as a discussion of the results. Then, in Subsection 5.3.2, we

report the results of an extra experiment performed to investigate whether the enriched feature representation of tweets proposed in this chapter can improve the classification effectiveness when combined with the other feature sets evaluated in Chapter 3.

5.3.1 Responding to Research Question RQ4

The computational experiments conducted in this section aim at addressing the last research question, RQ4, as follows:

– RQ4. Is it possible to use semantically related terms to enrich the sparse representation of tweets and boost the predictive performance of the n-gram-based features?

To answer this question, we used the set of sixteen datasets of tweets showed in Table 5.2. We applied the same experimental settings as described in Section 3.2 (Chapter 3), except for the features used. In the experiments reported in this section, we use only bag-of-words (unigrams) as features. For this purpose, as a preprocessing step, each tweet is tokenized and each term from a tweet is tagged with their respective part-of-speech, using the Twitter-specific part-of-speech tagset tool[1] [39]. It is important to mention that the proposed enrichment approach is also applied as a preprocessing step, after the tokenization and part-of-speech tagging steps.

Moreover, according to the results presented in Chapter 3 (Section 3.3.1), in which the SVM classifier achieved the best results for the n-gram features, we also use SVM in the experiments reported in this section.

Table 5.2: Characteristics of the datasets of tweets, ordered by size (*#tweets* column).

Dataset	**#tweets**	**#positive**	**#negative**
irony	65	22	43
sarcasm	71	33	38
aisopos	278	159	119
SemEval-Fig	321	47	274
sentiment140	359	182	177
person	439	312	127
movie	561	460	101
sanders	1,224	570	654
Narr	1,227	739	488
OMD	1,906	710	1,196
HCR	1,908	539	1,369
STS-gold	2,034	632	1,402
SentiStrength	2,289	1,340	949
Target-dependent	3,467	1,734	1,733
Vader	4,196	2,897	1,299
SemEval13	4,378	3,183	1,195

[1] `http://www.ark.cs.cmu.edu/TweetNLP`

In order to evaluate the performance of the proposed enrichment approach in the sentiment classification of tweets, we first analyze the effectiveness of the first method, i.e., the enrichment through the synonymy relation among words, considering words from all part-of-speech categories. More specifically, as a preprocessing step, we enrich the feature vector representation of each tweet with the synonyms of its terms, in which each term can be an adjective, an adverb, a verb, or a noun.

Table 5.3 shows the results achieved by using the enrichment approach (Enrichment column) in the classification, and its comparison with the default unigram model (Default column), in which only the original terms of tweets are used in the classification. As we can notice, incorporating the synonyms of terms from all part-of-speech categories did not entail any gain, as compared to the default unigram model, in which it has outperformed our approach in 13 out of the 16 datasets in terms of Accuracy, and in 11 out of the 16 datasets in terms of F-measure.

Table 5.3: Comparison between the Accuracies and F-measure scores (%) achieved by the default unigram model and the unigram model enriched with synonyms of all part-of-speech categories.

Synonymy relation among words				
Dataset	***Accuracy***		***F-measure***	
	Default	**Enrichment All part-of-speech**	**Default**	**Enrichment All part-of-speech**
irony	**66.2**	64.6	59.1	**60.7**
sarcasm	**54.9**	53.5	**54.5**	52.9
aisopos	**91.4**	90.6	**91.2**	90.5
SemEval-Fig	**91.0**	90.3	**90.5**	89.9
sentiment140	**82.7**	79.9	**82.7**	79.9
person	**77.7**	77.4	77.1	77.1
movie	85.6	85.6	83.9	**84.5**
sanders	**82.1**	79.7	**82.1**	79.7
Narr	**83.1**	81.3	**83.2**	81.3
OMD	**80.7**	79.9	**80.7**	79.8
HCR	**76.4**	75.8	**76.1**	75.6
STS-gold	**84.2**	83.5	**84.0**	83.4
SentiStrength	73.1	**74.3**	73.0	**74.2**
Target-dependent	**79.8**	79.4	**79.8**	79.4
Vader	**86.4**	85.5	**86.1**	85.2
SemEval13	81.1	81.1	80.6	80.6
#wins	**13**	1	**11**	3

In the next set of experiments, we analyze how each part-of-speech category may individually contribute to the enrichment approach. Specifically, for each part-of-speech category, we only enrich the tweets with synonyms of terms of that specific part-of-speech, considering one part-of-speech category at a time. Tables 5.4 and 5.5 present the results of this evaluation, in terms of Accuracy and F-measure, respectively.

Table 5.4: Comparison among the Accuracies (%) achieved by the default unigram model (Default) and the unigram model enriched with synonyms of each part-of-speech category at a time.

Synonymy relation among words					
	Accuracy				
Dataset	**Default**	**Enrichment**			
		Adjectives	**Adverbs**	**Nouns**	**Verbs**
irony	66.2	**67.7**	66.2	**67.7**	64.6
sarcasm	54.9	53.5	54.9	**56.3**	50.7
aisopos	91.4	91.0	91.4	**91.7**	91.0
SemEval-Fig	91.0	**91.3**	91.0	91.0	90.3
sentiment140	**82.7**	81.9	82.2	82.2	82.5
person	77.7	77.4	77.4	**78.4**	77.4
movie	**85.6**	85.2	84.8	85.4	83.8
sanders	**82.1**	81.9	81.7	81.4	81.0
Narr	**83.1**	83.0	82.5	82.8	81.7
OMD	80.7	80.6	**80.8**	79.0	80.4
HCR	**76.4**	**76.4**	76.3	**76.4**	75.4
STS-gold	84.2	**84.9**	84.4	84.2	83.8
SentiStrength	73.1	**74.3**	74.0	73.7	73.6
Target-dependent	79.8	**81.0**	80.3	79.7	79.8
Vader	86.4	**86.5**	85.7	85.5	85.3
SemEval13	81.1	**82.0**	81.3	81.0	81.1
#wins	5	**8**	1	5	0
rank sums	38.0	**36.0**	45.0	46.5	70.0

Table 5.5: Comparison among the F-measure scores (%) achieved by the default unigram model (Default) and the unigram model enriched with synonyms of each part-of-speech category at a time.

Synonymy relation among words					
	F-measure				
Dataset	**Default**	**Enrichment**			
		Adjectives	**Adverbs**	**Nouns**	**Verbs**
irony	59.1	**63.0**	60.6	**63.0**	59.5
sarcasm	54.5	53.2	54.7	**56.2**	50.1
sisopos	91.2	90.9	91.3	**91.6**	90.9
SemEval-Fig	90.5	**90.6**	90.5	90.5	89.9
sentiment140	**82.7**	81.9	82.2	82.2	82.4
person	77.1	76.9	76.8	**77.7**	77.0
movie	**83.9**	83.8	83.3	**83.9**	82.5
sanders	**82.1**	81.9	81.7	81.4	81.1
Narr	**83.2**	83.0	82.5	82.8	81.7
OMD	80.7	80.6	**80.8**	79.0	80.4
HCR	76.1	**76.2**	76.0	**76.2**	75.2
STS-gold	84.0	**84.7**	84.2	84.0	83.6
SentiStrength	73.0	**74.1**	73.8	73.6	73.5
Target-dependent	79.8	**81.0**	80.3	79.7	79.7
Vader	86.1	**86.2**	85.4	85.1	84.9
SemEval13	80.6	**81.4**	80.6	80.4	80.5
#wins	4	**8**	1	6	0
rank sums	41.0	**36.5**	45.5	45.5	70.0

As we can observe in Tables 5.4 and 5.5, the best results were achieved by incorporating synonyms of adjectives (Adjectives column) in the classification, as shown in the #wins and rank sums rows. Using only synonyms of adjectives achieved better results in eight out of the 16 datasets for both Accuracy and F-measure. In terms of Accuracy, the default unigram model (Default column) achieved better performance in five out of the 16 datasets, as well as incorporating synonyms of nouns (Nouns column). With respect to the F-measure metric, the default unigram model achieved better performance in four out of the 16 datasets. Interestingly, although incorporating the synonyms of adverbs (Adverbs column) has only outperformed the other methods for dataset OMD, it has achieved the third-best results in the overall evaluation, as shown in the rank sums row, for both Accuracy and F-measure. Conversely, the worse performance was achieved when the synonyms of verbs (Verbs column) are used to enrich the tweets. The synonyms of verbs may be the ones that misled the classification the most, due to the lack of context and because of the large number of synonyms the terms from this part-of-speech category may have.

Indeed, analyzing the misclassified tweet from dataset SemEval13 — "*I need to get ready for the 6th district Chicago Police Rally Against Violence*", we observed that the presence of the term *get*, which is employed as a verb, caused the augmentation of another 25 verbs in the feature vector representation of that tweet. Since this verb appears in 36 different synsets in WordNet, which means that it has 36 distinct senses or meanings, it is possible that the addition of many of these 25 synonyms has prevented the classifier from correctly assign a sentiment class to that tweet, due to the lack of context.

The fact that using all senses of words might insert noise in the classification, as we could observe in the previous experiment, motivates our next experiment. Specifically, we examine whether the sentiment classification of tweets benefits from using only the most popular meanings of words when retrieving their synonyms.

To confirm this hypothesis, we enrich the feature represention of tweets with the synonyms of the first two synsets retrieved from WordNet only, trying to deviate from the lack of context issue. In WordNet, although there is no specific order among the word senses, the most common uses of a word in the English language are listed above all the others. Moreover, considering that incorporating the synonyms of adjectives achieved the best overall results in the previous experiment, we consider only adjectives to hereafter. The results of this experiment are presented in Table 5.6. The best overall results are boldfaced, and the best results between the enrichment methods (all senses and 1st-2nd senses columns) are underlined.

Table 5.6: Comparison among the Accuracies and F-measure scores (%) achieved by the default unigram model, the unigram model enriched with synonyms of adjectives using all senses, and the unigram model enriched with synonyms of adjectives using the first and second senses only.

Synonymy relation among words						
Dataset	***Accuracy***			***F-measure***		
	Default	**Enrichment (adjectives)**		**Default**	**Enrichment (adjectives)**	
		all senses	**1st-2nd senses**		**all senses**	**1st-2nd senses**
irony	66.2	**67.7**	66.2	59.1	**63.0**	59.1
sarcasm	54.9	53.5	**56.3**	54.5	53.2	**56.0**
aisopos	**91.4**	91.0	91.0	**91.2**	90.9	90.9
SemEval-Fig	91.0	91.3	**91.6**	90.5	90.6	**91.1**
sentiment140	**82.7**	81.9	**82.7**	**82.7**	81.9	**82.7**
person	**77.7**	77.4	77.4	**77.1**	76.9	76.9
movie	**85.6**	85.2	85.4	83.9	83.8	**84.3**
sanders	**82.1**	81.9	**82.1**	**82.1**	81.9	**82.1**
Narr	**83.1**	83.0	82.5	**83.2**	83.0	82.5
OMD	**80.7**	80.6	**80.7**	**80.7**	80.6	**80.7**
HCR	**76.4**	**76.4**	76.2	76.1	**76.2**	76.0
STS-gold	84.2	84.9	**85.1**	84.0	84.7	**84.9**
SentiStrength	73.1	74.3	**74.5**	73.0	74.1	**74.4**
Target-dependent	79.8	**81.0**	80.6	79.8	**81.0**	80.6
Vader	86.4	86.5	**86.7**	86.1	86.2	**86.4**
SemEval13	81.1	**82.0**	81.8	80.6	**81.4**	81.3
#wins (enrichment)	–	5	**9**	–	5	**9**
#wins	**8**	4	**8**	6	4	**9**
rank sums	32.5	34.5	**29.0**	34.0	34.0	**28.0**

As we can observe in Table 5.6, adding the synonyms of the first two senses of adjectives outperformed the situation where the synonyms of all senses are incorporated. The use of the first and second senses (1st-2nd senses column) achieved the best results in nine out of the 16 datasets for both Accuracy and F-measure, as shown in the #wins (enrichment) row. Besides, incorporating the synonyms of the first two senses achieved the best overall results, as shown in the rank sums row.

Next, we analyze the effectiveness of the second enrichment method by leveraging the prior polarity of emoticons, as described in Subsection 5.2.2. The results of this evaluation are presented in Table 5.7. As we can see, this enrichment method (Enrichment column) achieved the best overall results in seven out of the 16 datasets of tweets in terms of Accuracy, and in nine out of the 16 datasets of tweets in terms of F-measure.

Finally, we present the results of the enrichment approach by combining the two enrichment methods described in Subsections 5.2.1 and 5.2.2. Specifically, we use the general enrichment approach as formally defined in Subsection 5.2.3. The results are reported in Table 5.8. Moreover, we ran a paired t-test to determine whether the differences between the results are statistically significant at a 0.05 significance level.

As we can observe in Table 5.8, in fact, it seems that the two enrichment methods

Table 5.7: Comparison among the Accuracies and F-measure scores (%) achieved by the default unigram model and the unigram model enriched using the prior polarity conveyed by emoticons.

Prior polarity of emoticons				
Dataset	***Accuracy***		***F-measure***	
	Default	**Enrichment**	**Default**	**Enrichment**
irony	66.2	66.2	59.1	59.1
sarcasm	54.9	54.9	54.5	**54.7**
aisopos	91.4	91.4	91.2	**91.3**
SemEval-Fig	91.0	91.0	90.5	90.5
sentiment140	82.7	**83.6**	82.7	**83.5**
person	77.7	**79.5**	77.1	**78.9**
movie	85.6	**85.7**	83.9	**84.1**
sanders	**82.1**	81.8	**82.1**	81.8
Narr	**83.1**	82.3	**83.2**	82.4
OMD	**80.7**	80.6	**80.7**	80.6
HCR	76.4	76.4	76.1	76.1
STS-gold	84.2	**85.0**	84.0	**84.9**
SentiStrength	73.1	**73.7**	73.0	**73.5**
Target-dependent	79.8	**80.5**	79.8	**80.5**
Vader	**86.4**	86.1	**86.1**	85.7
SemEval13	81.1	**82.0**	80.6	**81.3**
#wins	4	**7**	4	**9**

complement each other. The default unigram model outperformed the enrichment approach for datasets Narr and HCR only for both Accuracy and F-measure. On the other hand, the enrichment approach performed significantly better than the default unigram model in 12 out of the 16 datasets in terms of Accuracy, and in 13 out of the 16 datasets in terms of F-measure ($p < 0.05$, according to the paired t-test).

Table 5.8: Comparison among the Accuracies and the F-measure scores (%) achieved by the default unigram model and the unigram model enriched with synonyms of adjectives (first and second senses) and using the prior polarity conveyed by emoticons.

General enrichment approach				
Dataset	***Accuracy***		***F-measure***	
	Default	**Enrichment**	**Default**	**Enrichment**
irony	66.2	66.2	59.1	59.1
sarcasm	54.9	**57.7**	54.5	**57.3**
aisopos	91.4	**91.7**	91.2	**91.6**
SemEval-Fig	91.0	**91.6**	90.5	**91.1**
sentiment140	82.7	**83.6**	82.7	**83.6**
person	77.7	**79.0**	77.1	**78.6**
movie	85.6	**86.6**	83.9	**85.5**
sanders	82.1	**82.2**	82.1	**82.2**
Narr	**83.1**	82.3	**83.2**	82.4
OMD	80.7	80.7	80.7	**80.8**
HCR	**76.4**	76.1	**76.1**	75.9
STS-gold	84.2	**85.0**	84.0	**84.8**
SentiStrength	73.1	**74.5**	73.0	**74.4**
Target-dependent	79.8	**80.7**	79.8	**80.7**
Vader	86.4	**86.6**	86.1	**86.3**
SemEval13	81.1	**82.2**	80.6	**81.6**
#wins	2	**12**	2	**13**

{Enrichment} $\succ$ {Default}

Considering that data sparsity is an important aspect that can affect the overall performance of a classifier [75], we present an analysis of the effectiveness of the enrichment approach related to the number of new elements added in the vector representation of tweets of the assessed datasets. Furthermore, we present the sparsity degree of the datasets before and after the enrichment process.

The sparsity degree of a given dataset can be computed as follows [75]. Given a matrix $T \in \mathbb{R}^{n \times m}$, where n is the number of tweets and m is the number of unique terms in the vocabulary (i.e., the size of vocabulary), and each element $e_{i,j} \in T$ can be either 0 or 1, i.e., the term j does not occur in tweet i or the term j occurs in tweet i, respectively. The matrix T will be mostly populated by *zero* elements, due to the sparse nature of the n-gram representation. The sparsity degree S_d of T is the ratio between the number of zero elements and the total number of elements in T ($S_d \in [0, 1]$) [56], as shown in Equation 5.1, where z_i is the number of zero elements in tweet i.

$$S_d = \frac{\sum_i^n z_i}{n \times m} \tag{5.1}$$

Table 5.9 presents an analysis of the sparsity degree of the datasets before and after the enrichment (S_d before and S_d after columns, respectively). It is important to mention that we do not intend to reduce the sparsity degree of the assessed datasets, but to increase the inherent knowledge of the data to be classified by adding new information that may help the classifier. To this end, we show the loss in the number of zero elements (zero elements (loss) column) after the enrichment process (which is the same as the gain in the number of *one* elements).

In fact, as expected, the enrichment approach produces a very small decrement in the sparsity degree of the datasets. However, considering the losses in the number of zero elements, we can point out some insights about the results achieved by the enrichment approach reported in Table 5.8.

From Table 5.9, we can observe that the losses in the number of zero elements vary from 0.004 to 0.1%. Interestingly, for small datasets (in number of tweets), the losses in the number of zero elements are lower than the high-dimensional datasets. While the losses in the number of zero elements for the smallest datasets vary from 0.01 to 0.1%, for the high-dimensional ones the losses vary from 0.004 to 0.01%.

Regarding dataset aisopos, which is one of the smallest datasets, even though it presents the higher number of loss in zero elements (0.1%), the enrichment approach

Table 5.9: Sparsity degree and losses in the number of zero elements of assessed datasets.

Dataset	S_d before	S_d after	zero elements (loss) %	abs
irony	0.98037	0.98022	0.01	4
sarcasm	0.98056	0.98031	0.02	7
aisopos	0.99314	0.99208	0.1	347
SemEval-Fig	0.99401	0.99391	0.01	52
sentiment140	0.99443	0.99415	0.03	146
person	0.99433	0.99409	0.02	186
movie	0.99432	0.99373	0.06	519
sanders	0.99743	0.99725	0.02	741
Narr	0.99805	0.99783	0.02	1.064
OMD	0.99810	0.99804	0.006	473
HCR	0.99792	0.99788	0.004	414
STS-gold	0.99845	0.99837	0.008	799
SentiStrength	0.99877	0.99863	0.01	2.357
Target-dependent	0.99888	0.99880	0.008	2.244
Vader	0.99923	0.99911	0.01	4.583
SemEval13	0.99915	0.99908	0.008	4.333

achieved a marginally gain of 0.3% and 0.4% in terms of Accuracy and F-measure, respectively. Conversely, high-dimensional datasets such as STS-gold, SentiStrength, Target-dependent, Vader and SemEval13, in which the losses in the number of zero elements are among the lowest, the addition of new information on these datasets seems to be effective in increasing the classification effectiveness of the unigram features.

Moreover, we can see that for dataset HCR, in which the default unigram model outperformed the enrichment approach for both Accuracy and F-measure, and for dataset OMD, in which there was a tie between the results in terms of Accuracy, the losses in the number of zero elements are the lowest ones (0.004 and 0.006%, respectively). These datasets belong to a political domain, in which their tweets may contain many ironic content. In this type of domain, adjectives and emoticons are usually used to convey irony, such that their polarities are reversed, hence hindering the classifier to make correct decisions. Moreover, we observed that only a small fraction of tweets from datasets HCR and OMD contain emoticons (0.5 and 1.0%, respectively).

5.3.2 Further Analysis of the Enrichment Approach Effectiveness

To further investigate the classification effectiveness of the enrichment approach proposed in this chapter, we evaluate its predictive performance in an overall context, by combining the enriched unigram model with the other feature sets exploited in this thesis, i.e., n-grams, meta-features, and word embedding-based features.

Specifically, first, we concatenated the enriched unigram features with bigrams and trigrams extracted from each dataset to form the n-gram feature representation. Next, we trained an SVM classifier with this enriched n-gram representation. Then, considering that the ensemble learning strategy by stacking achieved the best overall results among the strategies for combination evaluated in Chapter 4, we use this ensemble technique to combine the strength of the different feature sets.

Table 5.10 reports the results of this investigation. The Stacking 1 column presents the results achieved by the best ensemble of classifiers (stacking), as presented in Table 4.12 (Ensemble learning column), and the Stacking 2 column presents the results obtained by switching the default n-gram classifier to the enriched one. As we can observe, the ensemble by using the enriched n-gram representation (Stacking 2 column) achieved the best overall results in nine out of the 16 datasets for both Accuracy and F-measure. This may be evidence that the enrichment approach can effectively contribute to improve the classification effectiveness in Twitter sentiment analysis.

Table 5.10: Accuracies and F-measure scores (%) achieved by combining the enriched n-gram representation with different feature sets as base learners of an ensemble strategy.

Dataset	***Accuracy***		***F-measure***	
	Stacking 1	**Stacking 2**	**Stacking 1**	**Stacking 2**
	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features (RF) enriched n-grams (SVM) w2v-Edin (LR)	meta-features (RF) n-grams (SVM) w2v-Edin (LR)	meta-features (RF) enriched n-grams (SVM) w2v-Edin (LR)
irony	**76.9**	75.4	**75.0**	73.0
sarcasm	78.9	78.9	78.8	78.8
aisopos	93.2	**93.9**	93.1	**93.9**
SemEval-Fig	91.9	**92.2**	91.2	**91.6**
sentiment140	**89.4**	89.1	**89.4**	89.1
person	**85.4**	84.5	**85.1**	84.1
movie	89.8	**90.6**	89.1	**89.9**
sanders	87.2	**87.4**	87.2	**87.4**
Narr	91.0	**91.1**	90.9	**91.1**
OMD	85.9	85.9	85.7	85.7
HCR	81.5	**81.6**	80.6	**80.7**
STS-gold	93.2	**93.3**	93.1	**93.2**
SentiStrength	**84.2**	83.9	**84.2**	83.9
Target-dependent	**85.7**	85.5	**85.7**	85.5
Vader	94.2	**94.3**	94.2	**94.3**
SemEval13	88.7	**89.0**	88.6	**88.9**
#wins	5	**9**	5	**9**

5.4 Summary

In this chapter, we addressed the research question RQ4 — "*Is it possible to use semantically related terms to enrich the sparse representation of tweets and boost the predictive*

performance of the n-gram-based features?". We proposed an enrichment approach to Twitter sentiment analysis, which uses lexicon resources and semantically related terms from the vocabulary to increase the knowledge of the naturally sparse Twitter data. To this end, we exploited the synonymy relation among words using WordNet as the lexicon resource, as well as the prior polarity information conveyed by emoticons, using a fixed-set of positive and negative words to enrich the sentiment representation of tweets.

In the experimental evaluation performed in this chapter, we showed that the sentiment classification of tweets benefits from enriching the feature representation of tweets with synonyms from the first and second senses (meanings) of adjectives in WordNet, as well as using the polarity of emoticons to guide the enrichment. Moreover, we showed that the enrichment approach can contribute to improve the classification effectivess in Twitter sentiment analysis in a broader context, by combining the enriched set of n-grams with meta-features and word embedding-based features.

The next chapter presents the concluding remarks of this thesis and directions for future research.

Chapter 6

Conclusions and Future Work

In this thesis, we presented a thoughtful evaluation of the distinct kinds of features employed in state-of-the-art works in Twitter sentiment analysis. The rich feature space exploited in this thesis includes features extracted from the basic n-gram language model to more sophisticated features such as meta-features and word embeddings. Besides the individual evaluation of each feature set, we also investigated the effect of combining them through feature concatenation and via ensemble learning strategies, considering that features from different sets can complement each other.

The meta-features examined in this work were collected from a large set of studies in the literature. Although these studies have proposed different meta-features, we filled the existing gap of aggregating and evaluating the predictive power of those meta-features designed in the literature over the years. Moreover, as an extension of our previous study [18], we categorized this rich set of meta-features to examine the effectiveness of different types of meta-features in discerning the positive tweets from the negative ones. Also, regarding the vast number of publicly available pre-trained embeddings, we conducted experiments to identify the most suitable one for detecting the sentiment expressed in tweets.

Based on the results obtained with the experiments conducted in this thesis, we can draw the following conclusions:

1) For each feature set studied in this work, we could see that an appropriate choice of a supervised learning algorithm can boost the classification effectiveness, on a large collection of 22 datasets of tweets. Specifically, for most situations, we showed that n-grams, meta-features, and embedding-based features could achieve significantly better results when fed to SVM, RF, and LR, respectively.

2) When evaluating the categories of meta-features proposed in this study, we could observe that the features from the Lexicon-based category are the most relevant ones in the task of Twitter sentiment analysis. The features from this category explore the content of tweets by relying on existing sentiment lexicons. In this work, we exploited seven different sentiment lexicons and lists of words. Since each lexicon comprises different words, we believe that they could effectively complement each other in representing the tweets. Nevertheless, we encourage the use of the set of all meta-features, considering that they can achieve even more improved results.

3) When compared to n-grams and word embedding-based features, the rich set of meta-features exploited in this study achieved better results. Also, we noticed that the sentiment classification of tweets benefits from the combination of all feature sets through feature concatenation. Conversely, the least accurate results were achieved by combining n-grams with the embedding-based features. On the other hand, regarding the combination of pairs of feature sets, we could see that only the combination provided by meta-features + n-grams performed statistically better than all individual classifiers (i.e., the classifiers generated for each group of features). For that reason, we believe that meta-features and n-grams can effectively complement each other in the sentiment classification of tweets.

4) We also showed that combining the individual classifiers via an ensemble technique can achieve overall best performances than a simple feature concatenation approach. Furthermore, we could see that the classification effectiveness of an ensemble of classifiers can be improved whether the diversity among the base classifiers is leveraged. Specifically, we showed that for an ensemble of classifiers to succeed in classifying the polarity of tweets better than its base learners, not only the predictive performance of the base learners has to be leveraged, but also the diversity among them. Those findings are in agreement with the literature of ensemble learning techniques [15, 30].

5) Concerning the data sparsity issue in text classification problems, provided by the largely used n-gram representation, we proposed an enrichment approach that uses semantically related terms from the vocabulary, taking advantage of the prior polarity information conveyed by emoticons and the synonymy relation among words in lexicon resources. Our goal was to augment the n-gram representation of tweets with information that may help the classifier to assign a sentiment label to unseen tweets correctly. Our results showed that the proposed approach outperformed the default bag-of-words model in 12 out of the 16 assessed datasets of tweets. Moreover,

we showed that high-dimensional datasets benefited the most from the addition of new information in the vector representation of tweets.

6) Finally, we evidenced that the enrichment approach proposed in this thesis can contribute to the task of Twitter sentiment analysis in an overall context. Specifically, we showed that combining the strength of the enriched set of n-gram features with meta-features and word embedding-based features using an ensemble learning strategy, the classification effectiveness in Twitter sentiment analysis can be indeed improved.

For future work, we plan to investigate more specific types of embedding models, such as Tweet2Vec [90], which is a method for generating general-purpose representation of tweets, using a character-level neural architecture. Also, we plan to examine whether fine-tuning methods initialized with pre-trained embedding models can improve the sentiment classification effectiveness on the target datasets used in this study.

Other aspect to be considered as future work is the application of feature selection methods on the word embedding-based features derived from the pre-trained models used in this thesis, as well as on the combined feature vectors derived from the concatenation of n-grams, meta-features and embedding-based features.

Regarding the enrichment approach proposed in this thesis, we plan to extend our study to use all 22 datasets of tweets in order to guarantee the effectiveness of our approach. Specifically, in the experiments conducted in Chapter 5, we used a subset of 16 datasets and we intend to include the six datasets that were left out from the experimental evaluation of the proposed enrichment approach. Those datasets are: *hobbit*, *iphone6*, *archeage*, *SemEval18*, *Semeval17*, and *SemEval16*.

We also believe that by applying fine hyper-parameter tuning on the classification algorithms used in the experimental evaluation, i.e., SVM, LR, and RF, the results could be further improved.

Finally, it is important to highlight some limitations of the study presented in this thesis. The study conduct in this thesis evaluate different feature sets and strategies for combining them for English tweets only. The features calculated from the pre-trained word embedding models used in the experimental evaluation are trained on English text corpus. Similarly, the meta-features from the category Lexicon-based are extracted from English sentiment lexicons and lists of words. Nevertheless, it is possible to use pre-trained embedding models and sentiment lexicons generated for any language, if available.

References

[1] Agarwal, A.; Xie, B.; Vovsha, I.; Rambow, O.; Passonneau, R. Sentiment analysis of Twitter data. In *Proceedings of the Workshop on Languages in Social Media* (2011), Association for Computational Linguistics, pp. 30–38.

[2] Agrawal, A.; An, A.; Papagelis, M. Learning emotion-enriched word representations. In *Proceedings of the 27th International Conference on Computational Linguistics* (2018), pp. 950–961.

[3] Araque, O.; Corcuera-Platas, I.; Sanchez-Rada, J. F.; Iglesias, C. A. Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications 77* (2017), 236–246.

[4] Arif, M. H.; Li, J.; Iqbal, M.; Liu, K. Sentiment analysis and spam detection in short informal text using learning classifier systems. *Soft Computing 22*, 21 (Nov 2018), 7281–7291.

[5] Baccianella, S.; Esuli, A.; Sebastiani, F. SentiWordNet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th International Conference on Language Resources and Evaluation* (2010), pp. 2200–2204.

[6] Barbosa, L.; Feng, J. Robust sentiment detection on Twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (2010), Association for Computational Linguistics, pp. 36–44.

[7] Bengio, Y.; Ducharme, R.; Vincent, P.; Janvin, C. A neural probabilistic language model. *J. Mach. Learn. Res. 3* (Mar. 2003), 1137–1155.

[8] Bermingham, A.; Smeaton, A. Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (2010), Association for Computing Machinery, pp. 1833–1836.

[9] Bifet, A.; Frank, E. Sentiment knowledge discovery in Twitter streaming data. In *Proceedings of the 13th International Conference on Discovery Science* (2010), Springer-Verlag, pp. 1–15.

[10] Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics 5* (2017), 135–146.

[11] Bravo-Marquez, F.; Frank, E.; Mohammad, S. M.; Pfahringer, B. Determining word-emotion associations from tweets by multi-label classification. In *2016*

IEEE/WIC/ACM International Conference on Web Intelligence (WI) (Oct 2016), pp. 536–539.

[12] Bravo-Marquez, F.; Frank, E.; Pfahringer, B.; Mohammad, S. M. Affectivetweets: a weka package for analyzing affect in tweets. *Journal of Machine Learning Research 20*, 92 (2019), 1–6.

[13] Bravo-Marquez, F.; Mendoza, M.; Poblete, B. Meta-level sentiment models for big social data analysis. *Knowledge-Based Systems 69* (2014), 86–99.

[14] Breiman, L. Random forests. *Machine Learning 45*, 1 (Oct 2001), 5–32.

[15] Brown, G.; Wyatt, J.; Harris, R.; Yao, X. Diversity creation methods: a survey and categorisation. *Information Fusion 6*, 1 (2005), 5 – 20.

[16] Buscaldi, D.; Hernandez-Farias, I. Sentiment analysis on microblogs for natural disasters management: A study on the 2014 genoa floodings. In *Proceedings of the 24th International Conference on World Wide Web* (2015), pp. 1185–1188.

[17] Canuto, S.; Gonçalves, M.; Benevenuto, F. Exploiting new sentiment-based meta-level features for effective sentiment analysis. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining* (2016), Association for Computing Machinery, pp. 53–62.

[18] Carvalho, J.; Plastino, A. An assessment study of feature and meta-level features in twitter sentiment analysis. In *Proceedings of the 22nd European Conference on Artificial Intelligence* (2016), IOS Press, pp. 769–777.

[19] Chang, C.; Lin, C. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2* (2011), 1–27.

[20] Chen, L.; Wang, W.; Nagarajan, M.; Wang, S.; Sheth, A. Extracting diverse sentiment expressions with target-dependent polarity from Twitter. In *Proceedings of the 6th International AAAI Conference on Weblogs and Social Media* (2012), pp. 50–57.

[21] Chikersal, P.; Poria, S.; Cambria, E. Sentu: sentiment analysis of tweets by combining a rule-based classifier with supervised learning. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (2015), pp. 647–651.

[22] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res. 12* (Nov. 2011), 2493–2537.

[23] Cozza, V.; Petrocchi, M. mib at semeval-2016 task 4a: exploiting lexicon based features for sentiment analysis in twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)* (2016), pp. 133–138.

[24] da Silva, N.; Colleta, L.; Hruschka, E.; Hruschka Jr., E. Using unsupervised information to improve semi-supervised tweet sentiment classification. *Information Sciences 355* (2016), 348–365.

[25] da Silva, N.; Hruschka, E.; Hruschka Jr., E. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems 66* (2014), 170–179.

[26] Davidov, D.; Tsur, O.; Rappoport, A. Enhanced sentiment learning using Twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (2010), Association for Computational Linguistics, pp. 241–249.

[27] De Smedt, T.; Daelemans, W. Pattern for python. *Journal of Machine Learning Research 13* (2012), 2063–2067.

[28] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res. 7* (Dec. 2006), 1–30.

[29] Diakopoulos, N.; Shamma, D. Characterizing debate performance via aggregated Twitter sentiment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2010), Association for Computing Machinery, pp. 1195–1198.

[30] Dietterich, T. G. Ensemble methods in machine learning. In *Multiple Classifier Systems* (Berlin, Heidelberg, 2000), Springer Berlin Heidelberg, pp. 1–15.

[31] Dong, L.; Wei, F.; Tan, C.; Tang, D.; Zhou, M.; Xu, K. Adaptive recursive neural network for target-dependent Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: short papers* (2014), Association for Computational Linguistics, pp. 49–54.

[32] Emadi, M.; Rahgozar, M. Twitter sentiment analysis using fuzzy integral classifier fusion. *Journal of Information Science 0*, 0 (2019), 1–17.

[33] Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; Lin, C.-J. Liblinear: A library for large linear classification. *Journal of Machine Learning Research 9* (June 2008), 1871–1874.

[34] Farias, D. H.; Rosso, P. Chapter 7 - irony, sarcasm, and sentiment analysis. In *Sentiment Analysis in Social Networks*, F. A. Pozzi, E. Fersini, E. Messina, and B. Liu, Eds. Morgan Kaufmann, Boston, 2017, pp. 113 – 128.

[35] Felbo, B.; Mislove, A.; Søgaard, A.; Rahwan, I.; Lehmann, S. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524* (2017).

[36] Fellbaum, C. *WordNet*. Wiley Online Library, 1998.

[37] Fersini, E.; Messina, E.; Pozzi, F. Sentiment analysis: Bayesian ensemble learning. *Decision Support Systems 68* (2014), 26 – 38.

[38] Fersini, E.; Messina, E.; Pozzi, F. Expressive signals in social media languages to improve polarity detection. *Information Processing & Management 52*, 1 (2016), 20 – 35.

[39] Gimpel, K.; Schneider, N.; O'Connor, B.; Das, D.; Mills, D.; Eisenstein, J.; Heilman, M.; Yogatama, D.; Flanigan, J.; Smith, N. Part-of-speech tagging for Twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers* (2011), Association for Computational Linguistics, pp. 42–47.

[40] Go, A.; Bhayani, R.; Huang, L. Twitter sentiment classification using distant supervision. Tech. Rep. CS224N, Stanford, 2009.

[41] Gonçalves, P.; Dalip, D.; Reis, J.; Messias, J.; Ribeiro, F.; Melo, P.; Gonçalves, M.; Benevenuto, F. Caracterizando e detectando sarcasmo e ironia no Twitter. In *Proceedings of the Brazilian Workshop on Social Network Analysis and Mining* (2015).

[42] Hagen, M.; Potthast, M.; Büchner, M.; Stein, B. Twitter sentiment detection via ensemble classification using averaged confidence scores. In *Proceedings of the 37th European Conference on IR Research* (2015), Springer, pp. 741–754.

[43] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I. The weka data mining software: an update. *SIGKDD Explorations Newsletter 11*, 1 (2009), 10–18.

[44] Hamdan, H. Sentisys at semeval-2016 task 4: Feature-based system for sentiment analysis in twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)* (2016), pp. 190–197.

[45] Hamdan, H.; Bellot, P.; Bechet, F. Lsislif: Crf and logistic regression for opinion target extraction and sentiment polarity analysis. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)* (2015), pp. 753–758.

[46] Hutto, C.; Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the 8th International AAAI Conference on Weblogs and Social Media* (2014).

[47] Jabreel, M.; Moreno, A. Sitaka at semeval-2017 task 4: sentiment analysis in twitter based on a rich set of features. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (2017), pp. 694–699.

[48] Jiang, L.; Yu, M.; Zhou, M.; Liu, X.; Zhao, T. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the ACL: Human Language Technologies* (2011), Association for Computational Linguistics, pp. 151–160.

[49] Kathuria, P. Sentiment classification using WSD, Maximum Entropy and Naive Bayes classifiers. `https://github.com/kevincobain2000/sentiment_classifier`. Accessed: 2019-08-30.

[50] Khuc, V.; Shivade, C.; Ramnath, R.; Ramanathan, J. Towards building large-scale distributed systems for Twitter sentiment analysis. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (2012), Association for Computing Machinery, pp. 459–464.

[51] Kouloumpis, E.; Wilson, T.; Moore, J. Twitter sentiment analysis: The good the bad and the omg! In *Proceedings of the 5th International AAAI Conference on Web and Social Media* (2011), pp. 538–541.

[52] Liu, B. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies 5*, 1 (2012), 1–167.

[53] Liu, B. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge University Press, 2015.

[54] Lochter, J. V.; Zanetti, R. F.; Reller, D.; Almeida, T. A. Short text opinion detection using ensemble of classifiers and semantic indexing. *Expert Systems with Applications 62* (2016), 243 – 249.

[55] Loria, S. Textblob: Simplified text processing. `https://textblob.readthedocs.io/en/dev/index.html`. Accessed: 2019-08-30.

[56] Makrehchi, M.; Kamel, M. S. Automatic extraction of domain-specific stopwords from labeled documents. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval* (Berlin, Heidelberg, 2008), ECIR'08, Springer-Verlag, pp. 222–233.

[57] Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; McClosky, D. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (Baltimore, Maryland, June 2014), Association for Computational Linguistics, pp. 55–60.

[58] Martínez-Cámara, E.; Martín-Valdivia, M.; Ureña-López, L.; Montejo-Ráez, A. Sentiment analysis in twitter. *Natural Language Engineering 20*, 1 (001 2014), 1–28.

[59] Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[60] Mikolov, T.; Grave, E.; Bojanowski, P.; Puhrsch, C.; Joulin, A. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)* (2018).

[61] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (2013), NIPS'13, pp. 3111–3119.

[62] Miranda-Jiménez, S.; Graff, M.; Tellez, E. S.; Moctezuma, D. Ingeotec at semeval 2017 task 4: A b4msa ensemble based on genetic programming for twitter sentiment analysis. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (2017), pp. 771–776.

[63] Mohammad, S.; Kiritchenko, S.; Zhu, X. Nrc-canada: building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the 7th International Workshop on Semantic Evaluation Exercises* (Atlanta, Georgia, USA, 2013).

[64] Mohammad, S.; Turney, P. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence 29*, 3 (2013), 436–465.

[65] Narayanan, V.; Arora, I.; Bhatia, A. Fast and accurate sentiment classification using an enhanced naive bayes model. In *Intelligent Data Engineering and Automated Learning – IDEAL 2013* (Berlin, Heidelberg, 2013), Springer Berlin Heidelberg, pp. 194–201.

[66] Narr, S.; Hulfenhaus, M.; Albayrak, S. Language-independent Twitter sentiment analysis. In *Proceedings of the Workshop on Knowledge Discovery, Data Mining and Machine Learning* (2012).

[67] Nielsen, F. Å. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. *CoRR abs/1103.2903* (2011).

[68] Pak, A.; Paroubek, P. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the 7th International Conference on Language Resources and Evaluation* (2010), pp. 1320–1326.

[69] Pang, B.; Lee, L.; Vaithyanathan, S. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical Methods in Natural Language Processing* (2002), Association for Computational Linguistics, pp. 79–86.

[70] Park, J. H.; Xu, P.; Fung, P. Plusemo2vec at semeval-2018 task 1: Exploiting emotion knowledge from emoji and# hashtags. In *Proceedings of the 12th international workshop on semantic evaluation (SemEval-2018)* (2018), pp. 264–272.

[71] Pennington, J.; Socher, R.; Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 1532–1543.

[72] Petrović, S.; Osborne, M.; Lavrenko, V. The edinburgh twitter corpus. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media* (Stroudsburg, PA, USA, 2010), Association for Computational Linguistics, pp. 25–26.

[73] Prusa, J.; Khoshgoftaar, T. M.; Dittman, D. J. Using ensemble learners to improve classifier performance on tweet sentiment data. In *2015 IEEE International Conference on Information Reuse and Integration* (2015), pp. 252–257.

[74] Reyes, A.; Rosso, P.; Veale, T. A multidimensional approach for detecting irony in twitter. *Lang. Resour. Eval. 47*, 1 (2013), 239–268.

[75] Saif, H. *Semantic Sentiment Analysis of Microblogs*. PhD thesis, The Open University, June 2015.

[76] Saif, H.; Fernandez, M.; He, Y.; Alani, H. Evaluation datasets for Twitter sentiment analysis: a survey and a new dataset, the STS-Gold. In *Proceedings of the 1st Workshop on Emotion and Sentiment in Social and Expressive Media* (2013).

[77] Saif, H.; Fernández, M.; He, Y.; Alani, H. On stopwords, filtering and data sparsity for sentiment analysis of Twitter. In *Proceedings of the 9th International Conference on Language Resources and Evaluation* (2014), European Language Resources Association, pp. 810–817.

[78] Saif, H.; He, Y.; Alani, H. Alleviating data sparsity for Twitter sentiment analysis. In *Proceedings of the 2nd Workshop on Making Sense of Microposts* (2012), CEUR-WS, pp. 2–9.

[79] Salton, G.; Wong, A.; Yang, C. S. A vector space model for automatic indexing. *Commun. ACM 18*, 11 (Nov. 1975), 613–620.

[80] Siddiqua, U. A.; Ahsan, T.; Chy, A. N. Combining a rule-based classifier with ensemble of feature sets and machine learning techniques for sentiment analysis on microblog. In *2016 19th International Conference on Computer and Information Technology (ICCIT)* (2016), pp. 304–309.

[81] Sousa, L.; de Mello, R.; Cedrim, D.; Garcia, A.; Missier, P.; UchÃ´a, A.; Oliveira, A.; Romanovsky, A. Vazadengue: An information system for preventing and combating mosquito-borne diseases with social networks. *Information Systems 75* (2018), 26 – 42.

[82] Speriosu, M.; Sudan, N.; Upadhyay, S.; Baldridge, J. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the 1st Workshop on Unsupervised Learning in NLP* (2011), Association for Computational Linguistics, pp. 53–63.

[83] Tang, D.; Wei, F.; Qin, B.; Liu, T.; Zhou, M. Coooolll: A deep learning system for twitter sentiment classification. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)* (Dublin, Ireland, Aug. 2014), Association for Computational Linguistics, pp. 208–212.

[84] Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; Qin, B. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Baltimore, Maryland, June 2014), Association for Computational Linguistics, pp. 1555–1565.

[85] Thelwall, M.; Buckley, K.; Paltoglou, G. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology 63*, 1 (2012), 163–173.

[86] Ting, K. M.; Witten, I. H. Issues in stacked generalization. *Journal of artificial intelligence research 10* (1999), 271–289.

[87] Turney, P. D.; Littman, M. L. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems 21*, 4 (Oct. 2003), 315–346.

[88] Vo, D.; Zhang, Y. Don't count, predict! an automatic approach to learning sentiment lexicons for short text. In *Proceedings of the 54th Annual Meeting of*

the Association for Computational Linguistics (2016), Association for Computing Machinery.

[89] Vo, D.-T.; Zhang, Y. Target-dependent twitter sentiment classification with rich automatic features. In *Proceedings of the 24th International Conference on Artificial Intelligence* (2015), IJCAI'15, AAAI Press, pp. 1347–1353.

[90] Vosoughi, S.; Vijayaraghavan, P.; Roy, D. Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2016), SIGIR '16, ACM, pp. 1041–1044.

[91] Wang, H.; Can, D.; Kazemzadeh, A.; Bar, F.; Narayanan, S. A system for real-time Twitter sentiment analysis of 2012 US presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations* (2012), Association for Computational Linguistics, pp. 115–120.

[92] Wasden, L. Internet lingo dictionary: A parents' guide to codes used in chat rooms, instant messaging, text messaging, and blogs. Tech. rep., Office of the Attorney General, 2010.

[93] Wiegand, M.; Balahur, A.; Roth, B.; Klakow, D.; Montoyo, A. A survey on the role of negation in sentiment analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing* (2010), Association for Computational Linguistics, pp. 60–68.

[94] Wilson, T.; Wiebe, J.; Hoffmann, P. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing* (2005), Association for Computational Linguistics, pp. 347–354.

[95] Wolpert, D. H. Stacked generalization. *Neural Networks 5*, 2 (1992), 241 – 259.

[96] Woods, K.; Kegelmeyer, W. P.; Bowyer, K. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 4 (April 1997), 405–410.

[97] Xu, P.; Madotto, A.; Wu, C.; Park, J. H.; Fung, P. Emo2vec: Learning generalized emotion representation by multi-task training. In *Proceedings of the EMNLP WASSA Workshop* (2018).

[98] Zhang, C.-X.; Duin, R. P. An experimental study of one- and two-level classifier fusion for different sample sizes. *Pattern Recognition Letters 32*, 14 (2011), 1756 – 1767.

[99] Zhang, L.; Ghosh, R.; Dekhil, M.; Hsu, M.; Liu, B. Combining lexicon-based and learning-based methods for Twitter sentiment analysis. Tech. Rep. HPL-2011-89, HP Laboratories, 2011.

APPENDIX A – Detailed Experimental Results: Chapter 3

A.1 Effectiveness of Word Embedding-based Features

Table A.1: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the w2v-GN pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*w2v-GN pre-trained model*											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	66.2	70.8	**75.4**	65.7	69.1	**70.3**	47.6	**48.6**	42.9	75.0	**79.6**	84.3
sarcasm	62.0	**67.6**	59.2	61.6	**67.3**	58.7	55.7	**62.3**	52.5	66.7	**71.6**	64.2
aisopos	88.8	**90.6**	85.6	88.8	**90.6**	85.3	90.3	**92.0**	88.2	86.9	**88.7**	81.5
SemEval-Fig	86.9	**88.2**	85.4	86.3	**86.6**	78.6	**50.0**	47.2	0.0	92.5	**93.3**	92.1
sentiment140	**85.0**	84.1	78.3	**85.0**	84.1	78.3	**85.2**	84.3	78.8	**84.7**	83.9	77.7
person	**81.3**	**81.3**	74.0	**81.3**	80.5	67.4	86.9	**87.5**	84.3	**67.5**	63.4	26.0
hobbit	90.8	**91.0**	82.4	90.8	**90.9**	80.7	93.2	**93.5**	88.2	**85.6**	85.5	64.9
iphone6	77.4	**78.8**	76.1	77.4	**78.2**	70.4	83.8	**85.3**	**85.3**	**62.7**	61.7	36.2
movie	87.3	**87.9**	82.2	**86.7**	86.3	74.3	92.5	**93.0**	90.2	**60.3**	55.8	2.0
sanders	80.2	**80.6**	79.5	80.2	**80.6**	79.2	78.3	**78.6**	75.9	81.9	**82.3**	82.2
Narr	86.7	**88.0**	79.6	86.7	**88.0**	78.7	89.1	**90.2**	84.6	83.1	**84.6**	69.8
archeage	**84.5**	83.1	81.0	**84.5**	83.0	80.4	**81.3**	79.3	74.4	**86.8**	85.7	84.9
SemEval18	78.4	**79.0**	76.9	78.4	**79.0**	76.7	76.6	**77.0**	72.9	80.0	**80.7**	79.9
OMD	80.0	**81.2**	76.9	79.7	**80.9**	74.8	71.6	**72.8**	59.7	84.6	**85.6**	83.8
HCR	76.8	**78.8**	74.6	75.9	**77.6**	67.4	54.0	**56.3**	23.2	84.5	**86.0**	84.8
STS-gold	83.1	**84.6**	76.6	82.9	**84.3**	72.6	71.6	**73.5**	44.7	88.0	**89.1**	85.2
SentiStrength	76.6	**77.0**	71.2	76.5	**76.8**	69.5	80.3	**80.8**	78.2	**71.2**	**71.2**	57.3
Target-dependent	**81.9**	**81.9**	78.5	**81.9**	**81.9**	78.5	**82.0**	81.9	78.9	**81.9**	81.8	78.2
Vader	**87.7**	**87.7**	80.6	**87.5**	87.4	77.9	91.3	**91.4**	87.5	**79.0**	78.6	56.4
SemEval13	81.9	**83.1**	75.8	81.5	**82.5**	68.5	87.9	**88.8**	85.7	64.3	**65.7**	22.9
SemEval17	86.0	**86.4**	82.4	86.0	**86.3**	81.7	81.0	**81.4**	72.9	88.9	**89.3**	87.0
SemEval16	84.6	**84.8**	79.6	84.1	**84.3**	75.5	89.8	**89.9**	87.5	69.0	**69.3**	43.2
#wins	5	**19**	1	6	**16**	1	4	**18**	1	9	**14**	0
rank sums	41.5	**26.5**	64.0	39.5	**28.5**	64.0						

Table A.2: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the GloVe-WP pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*GloVe-WP pre-trained model*											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	**78.5**	76.9	64.6	**77.5**	75.6	54.4	**63.2**	59.5	8.0	**84.8**	83.9	78.1
sarcasm	54.9	63.4	**67.6**	54.8	63.4	**66.8**	50.0	**61.8**	59.6	59.0	64.9	**72.9**
aisopos	84.5	**86.3**	86.0	84.5	**86.2**	85.7	86.6	88.5	**88.6**	81.7	**83.2**	81.7
SemEval-Fig	83.8	**86.3**	85.7	83.3	**83.7**	79.4	**40.9**	33.3	4.2	90.6	**92.4**	92.3
sentiment140	80.5	**85.5**	80.8	80.5	**85.5**	80.8	80.8	**85.8**	81.3	80.2	**85.2**	80.2
person	78.6	**80.4**	76.5	78.4	**79.5**	72.2	85.1	**86.9**	85.4	**61.8**	61.3	39.8
hobbit	89.5	**90.4**	83.5	89.4	**90.3**	82.2	92.3	**93.1**	88.9	83.4	**84.4**	67.9
iphone6	**78.8**	77.8	76.9	**78.5**	76.8	72.7	85.1	84.9	**85.5**	**63.2**	57.9	43.3
movie	85.4	**87.3**	82.2	84.7	**85.5**	74.3	91.3	**92.7**	90.2	**54.4**	53.0	2.0
sanders	75.7	**77.4**	76.7	75.6	**77.3**	76.5	73.6	**74.9**	72.9	77.4	79.4	**79.6**
Narr	84.4	**84.9**	79.2	84.4	**84.8**	78.3	87.1	**87.7**	84.3	80.3	**80.5**	69.2
archeage	82.2	**82.5**	82.0	82.2	**82.5**	81.7	78.7	**78.8**	76.5	84.8	85.1	**85.4**
SemEval18	78.0	**79.2**	77.8	78.0	**79.2**	77.6	76.4	**77.4**	74.4	79.4	**80.8**	80.4
OMD	80.7	**81.9**	79.3	80.5	**81.6**	77.6	72.6	**74.1**	64.2	85.1	**86.1**	85.5
HCR	75.3	**76.0**	73.5	73.6	**74.2**	65.2	47.9	**48.4**	16.8	83.8	**84.4**	84.3
STS-gold	82.2	**83.3**	76.6	81.9	**82.8**	72.9	69.9	**70.7**	46.0	87.4	**88.3**	85.1
SentiStrength	74.7	**75.9**	70.8	74.7	**75.8**	69.0	78.8	**79.9**	78.1	68.8	**70.0**	56.1
Target-dependent	**80.6**	80.5	78.0	**80.6**	80.5	78.0	**80.5**	**80.5**	78.1	**80.8**	80.5	77.8
Vader	86.8	**87.1**	79.7	86.6	**86.8**	76.6	90.7	**91.0**	87.0	77.5	**77.6**	53.4
SemEval13	81.3	**81.8**	75.8	80.5	**81.0**	68.7	87.7	**88.0**	85.6	61.5	**62.4**	23.8
SemEval17	86.3	**86.6**	82.8	86.3	**86.6**	82.2	81.5	**81.7**	74.0	89.2	**89.4**	87.1
SemEval16	85.1	**85.2**	80.3	**84.7**	**84.7**	76.9	90.1	**90.2**	87.9	70.1	**70.2**	47.8
#wins	3	**18**	1	4	**18**	1	3	**18**	2	5	**14**	3
rank sums	46.0	**26.0**	60.0	44.5	**26.5**	61.0						

Table A.3: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the fastText pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*fastText pre-trained model*											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	67.7	**73.8**	67.7	67.5	**71.7**	60.2	51.2	**51.4**	22.2	75.9	**82.1**	79.6
sarcasm	49.3	64.8	**66.2**	49.2	64.4	**65.7**	43.8	59.0	**60.0**	53.8	69.1	**70.7**
aisopos	71.2	**76.6**	69.4	71.2	**76.5**	68.0	75.2	**80.0**	76.3	65.8	**71.9**	56.9
SemEval-Fig	86.0	**88.2**	85.7	85.0	**85.7**	79.4	**44.4**	40.6	4.2	92.0	**93.4**	92.3
sentiment140	78.8	**82.5**	79.9	78.8	**82.4**	79.9	79.0	**82.9**	81.0	78.7	**81.9**	78.8
person	79.0	**83.1**	75.2	79.0	**82.6**	69.6	85.3	**88.6**	84.8	63.5	**67.8**	32.3
hobbit	89.1	**91.0**	81.6	89.0	**90.8**	79.6	92.1	**93.6**	87.8	82.6	**85.1**	62.2
iphone6	76.9	**81.2**	76.9	76.8	**80.8**	72.4	83.5	**86.9**	85.5	61.2	**66.7**	42.3
movie	88.1	**88.2**	82.2	**87.3**	86.5	74.3	92.9	**93.2**	90.2	**61.7**	56.0	2.0
sanders	79.2	**80.1**	77.7	79.2	**80.1**	77.4	78.2	**78.5**	73.5	80.1	**81.4**	80.7
Narr	86.0	**86.1**	79.1	86.0	**86.1**	78.2	88.4	**88.6**	84.3	**82.3**	**82.3**	68.9
archeage	83.8	83.9	**84.2**	**83.8**	**83.8**	83.9	**80.8**	80.6	79.6	86.0	86.2	**87.0**
SemEval18	79.9	**81.2**	76.5	79.9	**81.2**	76.2	78.2	**79.5**	72.2	81.4	**82.7**	79.6
OMD	79.4	**80.1**	76.1	79.2	**79.8**	74.0	71.0	**71.7**	58.3	84.0	**84.6**	83.3
HCR	77.1	**77.3**	74.4	**76.3**	75.7	66.6	**54.9**	52.0	20.5	84.7	**85.1**	84.7
STS-gold	85.0	**85.3**	77.0	84.9	**85.0**	73.3	**75.4**	74.8	46.7	89.2	**89.6**	85.3
SentiStrength	77.2	**78.2**	72.0	77.1	**78.0**	70.5	80.8	**81.9**	78.8	71.9	**72.6**	58.8
Target-dependent	82.2	**82.5**	79.8	82.2	**82.5**	79.8	82.2	**82.4**	79.9	82.2	**82.5**	79.7
Vader	88.1	**88.5**	80.2	88.0	**88.3**	77.4	91.6	**91.9**	87.3	79.9	**80.1**	55.3
SemEval13	82.8	**83.2**	75.9	82.3	**82.6**	68.6	88.5	**88.9**	85.8	65.7	**66.0**	22.9
SemEval17	88.2	**88.5**	85.0	88.2	**88.4**	84.7	84.1	**84.3**	77.9	90.7	**90.9**	88.7
SemEval16	86.1	**86.2**	80.4	**85.8**	**85.8**	76.9	90.7	**90.8**	88.0	**72.5**	72.4	47.3
#wins	0	**20**	2	4	**19**	1	4	**17**	1	3	**18**	2
rank sums	48.0	**24.0**	60.0	43.0	**26.0**	63.0						

Table A.4: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the EWE pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*EWE pre-trained model*											
	Accuracy			*F-measure*								
				average			*positive*			*negative*		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	67.7	**69.2**	67.7	**66.5**	65.3	58.3	**46.2**	37.5	16.0	76.9	79.6	**80.0**
sarcasm	67.6	69.0	**71.8**	67.6	69.0	**71.5**	65.7	**66.7**	**66.7**	69.3	71.1	**75.6**
aisopos	70.9	**73.7**	73.0	70.8	**73.3**	71.9	74.8	78.5	**78.9**	65.5	**66.4**	62.7
SemEval-Fig	83.8	**86.9**	85.4	83.3	**84.5**	78.6	**40.9**	36.4	0.0	90.6	**92.7**	92.1
sentiment140	81.9	**83.8**	82.5	81.9	**83.8**	82.4	81.9	**84.2**	83.0	81.9	**83.5**	81.8
person	78.4	**84.1**	79.3	78.3	**83.5**	76.3	84.8	**89.2**	86.9	62.2	**69.6**	50.3
hobbit	**93.5**	92.5	89.8	**93.4**	92.5	89.5	**95.3**	94.6	92.9	**89.6**	88.0	82.4
iphone6	**80.1**	78.4	78.8	**79.7**	77.6	75.3	86.1	85.2	**86.6**	**65.1**	59.9	49.3
movie	**87.3**	87.2	82.4	**86.5**	85.4	74.7	92.5	**92.6**	90.3	**59.4**	52.6	3.9
sanders	**80.0**	79.0	78.7	**80.0**	79.0	78.5	**78.5**	77.6	75.3	**81.3**	80.3	**81.3**
Narr	83.5	**84.5**	79.1	83.4	**84.4**	78.1	86.4	**87.4**	84.2	78.9	**80.0**	68.9
archeage	82.9	**83.5**	82.9	83.0	**83.5**	82.8	79.9	**80.4**	78.5	85.2	85.8	**85.9**
SemEval18	78.2	**79.5**	77.7	78.1	**79.5**	77.5	76.2	**77.5**	74.1	79.8	**81.2**	80.4
OMD	78.3	**78.6**	75.7	78.0	**78.3**	73.5	69.2	**69.3**	57.5	83.3	**83.6**	82.9
HCR	76.6	**76.9**	74.8	75.3	**75.4**	67.2	**52.1**	51.2	22.1	84.5	84.9	**85.0**
STS-gold	85.0	**85.1**	80.0	**84.8**	**84.8**	77.3	**75.1**	74.3	55.6	89.2	**89.5**	87.1
SentiStrength	77.2	**77.9**	72.7	77.1	**77.8**	71.5	80.8	**81.6**	79.0	71.9	**72.5**	60.9
Target-dependent	81.5	**82.6**	80.0	81.5	**82.6**	80.0	81.4	**82.5**	80.2	81.7	**82.7**	79.7
Vader	87.3	**88.0**	81.5	87.1	**87.7**	79.1	91.0	**91.5**	88.0	78.3	**79.2**	59.3
SemEval13	82.1	**82.5**	76.6	81.4	**81.7**	70.7	88.2	**88.4**	86.0	63.5	**63.8**	30.0
SemEval17	86.8	**87.2**	85.3	86.8	**87.2**	85.1	82.3	**82.7**	78.8	89.5	**89.9**	88.8
SemEval16	85.4	**85.6**	81.5	84.9	**85.0**	78.9	90.3	**90.4**	88.5	70.3	**70.7**	53.3
#wins	4	**17**	1	6	**16**	1	6	**14**	3	4	**14**	5
rank sums	46.0	**28.0**	58.0	41.5	**28.5**	62.0						

Table A.5: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the GloVe-TW pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*GloVe-TW pre-trained model*											
	Accuracy			*F-measure*								
				average			*positive*			*negative*		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	60.0	**66.2**	63.1	60.0	**63.8**	51.2	**40.9**	38.9	0.0	69.8	76.6	**77.4**
sarcasm	64.8	**69.0**	64.8	64.6	**68.4**	64.2	60.3	**62.1**	57.6	68.4	**73.8**	69.9
aisopos	74.5	**76.6**	74.1	74.2	**76.4**	73.1	78.5	**80.2**	79.7	68.4	**71.4**	64.4
SemEval-Fig	84.4	**87.2**	84.7	83.5	**84.4**	78.3	**39.0**	34.9	0.0	91.1	**92.9**	91.7
sentiment140	81.1	**83.6**	80.8	81.1	**83.6**	80.8	81.5	**83.9**	81.2	80.6	**83.2**	80.3
person	81.1	**83.1**	78.4	80.8	**82.5**	75.1	86.9	**88.6**	86.4	65.8	**67.5**	47.5
hobbit	88.3	**90.0**	83.7	88.2	**89.9**	82.6	91.5	**92.8**	88.9	81.5	**83.9**	69.3
iphone6	79.9	**82.1**	79.3	79.7	**81.6**	76.5	85.8	**87.7**	86.7	65.6	**67.6**	53.0
movie	86.5	**86.6**	82.2	**85.2**	84.6	74.6	92.1	**92.3**	90.2	**54.2**	49.7	3.8
sanders	**80.6**	**80.6**	79.2	**80.6**	**80.6**	79.0	**79.1**	79.0	76.3	81.8	**82.0**	81.4
Narr	86.6	**88.6**	84.8	86.6	**88.6**	84.6	88.9	**90.6**	87.8	83.1	**85.6**	79.6
archeage	84.5	**85.2**	**85.2**	84.5	**85.2**	85.0	81.8	**82.1**	81.2	86.6	87.4	**87.8**
SemEval18	**81.5**	81.4	78.4	**81.5**	81.4	78.2	**79.7**	79.5	75.0	**83.0**	**83.0**	81.1
OMD	76.8	**77.2**	76.0	76.4	**76.9**	74.1	66.3	**67.3**	59.3	82.3	82.5	**83.0**
HCR	77.9	**79.0**	76.6	76.9	**77.7**	70.8	55.5	**56.1**	32.6	85.3	**86.2**	85.9
STS-gold	84.9	**85.3**	80.0	84.7	**85.0**	77.8	**74.8**	74.6	57.6	89.2	**89.7**	86.9
SentiStrength	77.9	**78.0**	74.7	77.8	**77.9**	74.1	81.4	**81.6**	79.9	**72.7**	**72.7**	65.8
Target-dependent	82.8	**83.1**	80.6	82.8	**83.1**	80.6	82.7	**83.1**	80.8	82.9	**83.1**	80.4
Vader	87.2	**87.4**	82.9	87.0	**87.1**	81.2	91.0	**91.1**	88.7	**78.1**	78.0	64.5
SemEval13	**83.1**	**83.1**	78.7	82.4	**82.5**	74.6	**88.8**	**88.8**	86.9	65.4	**65.5**	41.8
SemEval17	87.6	**87.7**	85.7	**87.6**	**87.6**	85.4	**83.2**	**83.2**	79.3	90.2	**90.3**	89.0
SemEval16	**86.6**	86.4	82.6	**86.2**	86.0	80.5	**91.0**	90.9	89.0	**73.2**	72.9	57.9
#wins	4	**20**	1	5	**19**	0	8	**16**	0	5	**16**	3
rank sums	44.5	**25.5**	62.0	41.0	**26.0**	65.0						

Table A.6: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the w2v-Araque pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*w2v-Araque pre-trained model*											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	67.7	**69.2**	66.2	67.5	**67.8**	55.2	**51.2**	47.4	8.3	75.9	78.3	**79.2**
sarcasm	63.4	**70.4**	66.2	63.3	**70.1**	65.7	59.4	**65.6**	60.0	66.7	**74.1**	70.7
aisopos	68.7	**74.8**	73.4	68.8	**74.8**	72.4	72.4	78.3	**79.0**	63.9	**70.1**	63.7
SemEval-Fig	82.6	**86.0**	85.4	82.4	**84.4**	78.6	39.1	**40.0**	0.0	89.8	**92.1**	**92.1**
sentiment140	74.4	**80.2**	**80.2**	74.4	**80.2**	**80.2**	74.9	80.4	**80.5**	73.9	**80.0**	79.9
person	77.2	**78.6**	72.9	77.0	**77.9**	65.8	84.2	**85.4**	83.6	59.3	**59.5**	22.2
hobbit	91.2	**92.3**	89.3	91.2	**92.3**	88.9	93.5	**94.4**	92.4	86.1	**87.9**	81.6
iphone6	77.4	78.4	**78.8**	77.4	**78.1**	75.6	83.8	84.8	**86.5**	62.7	**62.8**	50.7
movie	84.0	**87.0**	82.4	83.4	**85.5**	74.7	90.4	**92.4**	90.3	51.6	**54.1**	3.9
sanders	75.7	**77.9**	77.4	75.7	**78.0**	77.1	74.0	**76.5**	73.6	77.3	79.2	**80.2**
Narr	84.1	**85.3**	80.5	84.1	**85.3**	80.0	86.9	**88.0**	84.8	79.8	**81.3**	72.8
archeage	83.1	**83.2**	81.5	83.0	**83.2**	81.2	79.6	**79.7**	75.9	85.5	**85.7**	85.1
SemEval18	73.6	**75.3**	73.5	73.5	**75.2**	73.1	70.9	**72.6**	68.1	75.8	**77.5**	77.4
OMD	74.5	**77.1**	73.9	74.2	**76.7**	71.3	63.8	**67.2**	53.5	80.3	**82.4**	81.9
HCR	73.0	74.1	**74.3**	71.3	**71.6**	66.6	**43.2**	41.8	20.7	82.3	83.3	**84.7**
STS-gold	**86.4**	86.2	80.1	**86.3**	86.0	78.0	**77.7**	76.6	58.3	**90.3**	90.2	86.9
SentiStrength	75.7	**76.8**	73.0	75.6	**76.7**	71.9	79.6	**80.6**	79.3	69.9	**71.3**	61.3
Target-dependent	80.4	**81.5**	77.3	80.4	**81.5**	77.3	79.9	**81.2**	77.2	80.9	**81.8**	77.4
Vader	86.0	**86.7**	80.1	85.8	**86.4**	77.3	90.1	**90.6**	87.3	76.3	**76.9**	55.0
SemEval13	80.4	**81.0**	76.4	79.8	**80.3**	70.1	86.9	**87.4**	85.9	61.2	**61.4**	28.2
SemEval17	82.7	**83.1**	79.6	82.4	**82.8**	78.5	75.4	**75.8**	67.5	86.7	**87.0**	85.2
SemEval16	**82.8**	82.7	78.1	**82.2**	82.1	73.8	**88.5**	**88.5**	86.7	**65.2**	64.9	39.3
#wins	2	**18**	3	2	**20**	1	4	**16**	3	2	**17**	4
rank sums	49.0	**26.5**	56.5	46.0	**24.5**	61.5						

Table A.7: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the w2v-Edin pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*w2v-Edin pre-trained model*											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	**75.4**	**75.4**	66.2	**74.7**	74.2	55.2	**60.0**	57.9	8.3	82.2	**82.6**	79.2
sarcasm	53.5	56.3	**67.6**	53.5	56.1	**67.0**	49.2	50.8	**61.0**	57.1	60.8	**72.3**
aisopos	**93.2**	92.8	91.0	**93.2**	92.8	90.9	**94.0**	93.9	92.5	**92.1**	91.3	88.8
SemEval-Fig	87.5	**89.1**	85.4	87.0	**87.7**	78.6	**53.5**	52.1	0.0	92.8	**93.8**	92.1
sentiment140	82.7	**87.7**	85.5	82.7	**87.7**	85.5	83.4	**87.9**	85.6	82.0	**87.6**	85.4
person	78.8	**81.3**	77.0	78.7	**80.9**	73.8	85.2	**87.2**	85.4	62.7	**65.3**	45.4
hobbit	90.6	**92.5**	83.3	90.7	**92.5**	82.1	93.0	**94.5**	88.7	85.7	**88.5**	68.1
iphone6	81.2	**81.6**	78.2	81.1	**81.3**	74.5	86.7	**87.1**	86.3	**68.2**	67.8	47.3
movie	**89.5**	88.6	82.2	**89.0**	87.3	74.3	**93.7**	93.3	90.2	**67.4**	60.0	2.0
sanders	82.2	**82.9**	79.6	82.2	**82.9**	79.5	80.5	**81.4**	76.9	83.6	**84.2**	81.7
Narr	89.1	**89.6**	85.3	89.0	**89.6**	85.1	91.1	**91.5**	88.3	86.0	**86.7**	80.4
archeage	85.8	**87.0**	84.9	85.8	**87.0**	84.7	83.0	**84.5**	80.8	87.8	**88.9**	87.5
SemEval18	81.4	**82.8**	78.9	81.4	**82.8**	78.7	79.6	**81.3**	75.8	82.9	**84.1**	81.3
OMD	82.3	**83.3**	79.5	82.1	**83.1**	77.9	75.2	**76.1**	65.2	86.3	**87.2**	85.5
HCR	77.4	**78.5**	74.9	76.6	**77.3**	68.3	55.5	**55.6**	26.2	84.9	**85.8**	84.9
STS-gold	87.0	**87.5**	80.1	86.8	**87.3**	77.9	78.2	**78.8**	57.6	90.7	**91.1**	87.0
SentiStrength	79.3	**81.2**	74.7	79.3	**81.2**	74.1	82.4	**84.1**	79.7	74.9	**77.0**	66.3
Target-dependent	81.4	**82.5**	78.7	81.4	**82.5**	78.7	81.4	**82.5**	79.0	81.5	**82.5**	78.5
Vader	**89.3**	**89.3**	82.4	**89.2**	**89.2**	80.5	92.4	**92.5**	88.4	**82.2**	81.8	62.7
SemEval13	83.1	**83.6**	77.8	82.7	**83.1**	73.0	88.6	**89.0**	86.5	66.8	**67.5**	37.2
SemEval17	86.9	**87.6**	84.1	86.9	**87.5**	83.8	82.2	**83.1**	77.0	89.6	**90.2**	87.9
SemEval16	**86.4**	**86.4**	81.5	**86.1**	**86.1**	79.1	**90.9**	**90.9**	88.4	**73.3**	73.2	54.3
#wins	5	**19**	1	5	**18**	1	5	**17**	1	5	**16**	1
rank sums	42.5	**26.5**	63.0	42.0	**27.0**	63.0						

Table A.8: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the SSWE pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*SSWE pre-trained model*											
	Accuracy			*F-measure*								
				average			*positive*			*negative*		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	**78.5**	73.8	75.4	**77.5**	70.1	73.0	**63.2**	45.2	52.9	**84.8**	82.8	83.3
sarcasm	**74.6**	73.2	73.2	**74.6**	73.2	73.0	**71.9**	70.8	68.9	**76.9**	75.3	76.5
aisopos	**93.2**	91.7	92.4	**93.1**	91.7	92.4	**94.1**	92.8	93.5	**91.8**	90.2	91.1
SemEval-Fig	**88.2**	87.5	87.5	**86.2**	84.4	83.3	**44.1**	33.3	25.9	**93.4**	93.1	93.2
sentiment140	83.6	**84.1**	83.3	83.6	**84.1**	83.3	83.7	**84.3**	83.5	83.4	**83.9**	83.1
person	**79.3**	78.6	78.6	**78.1**	77.3	76.4	**86.2**	85.8	**86.2**	**58.1**	56.5	52.5
hobbit	84.5	83.1	**84.7**	**84.2**	82.7	**84.2**	88.9	88.1	**89.2**	**74.4**	71.2	73.5
iphone6	75.0	74.8	**79.3**	73.6	73.0	**78.1**	83.1	83.2	**86.1**	51.6	49.6	**59.9**
movie	**89.5**	88.4	86.3	**88.1**	86.9	83.5	**93.9**	93.3	92.2	**61.4**	57.5	43.8
sanders	78.0	77.8	**79.2**	78.0	77.7	**79.2**	76.2	75.7	**77.2**	79.6	79.5	**81.0**
Narr	89.4	**89.5**	89.1	**89.4**	**89.4**	89.0	91.3	**91.4**	91.0	**86.4**	**86.4**	86.0
archeage	79.3	79.5	**81.6**	79.2	79.3	**81.3**	74.6	74.6	**76.3**	82.5	82.7	**85.0**
SemEval18	**81.4**	80.8	80.2	**81.4**	80.8	80.1	**79.5**	78.9	78.1	**83.1**	82.4	82.0
OMD	76.8	77.2	**77.5**	75.9	76.5	**76.6**	64.3	**65.7**	64.9	82.8	83.0	**83.5**
HCR	71.8	73.6	**74.6**	59.9	68.3	**69.7**	0.0	28.8	**32.4**	83.6	83.8	**84.3**
STS-gold	87.0	**87.8**	86.6	86.9	**87.6**	86.4	78.5	**79.2**	77.1	90.7	**91.3**	90.6
SentiStrength	**79.3**	79.2	78.6	**79.2**	79.1	78.5	**82.9**	82.7	82.2	**74.0**	73.9	73.4
Target-dependent	77.4	77.5	**77.7**	77.4	77.5	**77.7**	77.0	**77.3**	**77.3**	77.8	77.7	**78.1**
Vader	87.5	**87.7**	86.7	87.2	**87.3**	86.4	91.3	**91.4**	90.7	**78.2**	**78.2**	76.9
SemEval13	82.9	83.2	**83.3**	82.2	**82.6**	82.4	88.7	88.9	**89.0**	65.1	**65.6**	64.7
SemEval17	80.7	**80.8**	80.7	80.3	**80.4**	80.3	71.9	**72.2**	71.9	85.3	85.3	85.3
SemEval16	**81.7**	81.5	81.4	**80.7**	80.6	80.1	**88.0**	87.9	87.9	**61.2**	**61.2**	59.3
#wins	**9**	5	8	**11**	6	7	**9**	7	8	**12**	6	6
rank sums	**40.5**	45.5	46.0	**39.5**	43.5	49.0						

Table A.9: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the Emo2Vec pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	*Emo2Vec pre-trained model*											
	Accuracy			*F-measure*								
				average			*positive*			*negative*		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	**80.0**	76.9	73.8	**79.3**	74.4	70.1	**66.7**	54.5	45.2	**85.7**	84.5	82.8
sarcasm	60.6	62.0	**67.6**	60.0	61.3	**67.4**	53.3	54.2	**63.5**	65.9	67.5	**70.9**
aisopos	75.9	**79.9**	78.8	75.8	**79.8**	78.8	79.3	**82.7**	81.5	71.2	**75.9**	75.1
SemEval-Fig	**87.9**	87.5	85.7	**86.7**	84.9	80.4	**49.4**	37.5	11.5	**93.1**	**93.1**	92.2
sentiment140	**85.0**	84.7	83.8	**85.0**	84.7	83.8	**85.4**	85.2	84.2	**84.5**	84.1	83.4
person	78.1	79.3	**80.0**	77.5	78.5	**78.9**	85.1	86.0	**86.6**	58.6	59.9	**60.0**
hobbit	**90.6**	88.7	87.4	**90.5**	88.6	87.2	**93.2**	91.8	90.9	**85.0**	81.7	79.5
iphone6	79.9	78.8	**80.3**	**79.5**	78.1	79.2	86.0	85.3	**86.7**	**64.5**	61.4	61.8
movie	88.9	**89.3**	87.2	87.9	**88.3**	85.5	93.5	**93.7**	92.6	62.2	**63.4**	53.2
sanders	**79.4**	79.1	78.8	**79.4**	79.1	78.7	**77.7**	77.5	76.2	80.9	80.5	**81.0**
Narr	88.0	**88.6**	88.0	88.0	**88.5**	88.0	90.2	**90.7**	90.2	84.7	**85.3**	84.6
archeage	**82.0**	81.9	81.8	**81.9**	81.8	81.6	**77.8**	77.6	76.8	84.9	84.8	**85.0**
SemEval18	80.3	**80.4**	79.1	80.3	**80.4**	79.0	78.3	**78.4**	77.0	82.0	**82.1**	80.8
OMD	76.3	**76.4**	75.5	75.6	**75.8**	74.1	64.5	**64.9**	60.3	**82.3**	82.2	**82.3**
HCR	75.0	75.3	**75.4**	70.0	**71.5**	70.0	32.7	**38.5**	31.9	84.6	84.5	**85.0**
STS-gold	85.4	85.8	**85.9**	85.2	85.5	**85.6**	75.2	**75.4**	**75.4**	89.7	90.0	**90.2**
SentiStrength	85.2	**85.4**	83.6	85.2	**85.4**	83.6	87.5	**87.7**	86.2	81.8	**82.1**	79.8
Target-dependent	**81.5**	81.3	79.5	**81.5**	81.3	79.5	**81.2**	81.1	79.1	**81.8**	81.5	79.9
Vader	**87.1**	**87.1**	86.4	**86.8**	**86.8**	86.0	**91.0**	**91.0**	90.5	77.3	**77.4**	76.1
SemEval13	**88.8**	88.7	87.7	**88.7**	88.5	87.4	**92.5**	92.3	91.8	**78.5**	78.2	75.5
SemEval17	**85.4**	85.3	84.4	**85.3**	85.2	84.2	**79.9**	79.6	77.9	**88.5**	**88.5**	88.0
SemEval16	**84.5**	**84.5**	84.0	**84.0**	**84.0**	83.2	**89.7**	**89.7**	89.5	**68.7**	68.5	66.1
#wins	**11**	8	5	**12**	9	3	**11**	10	4	**10**	8	7
rank sums	39.5	**38.0**	54.5	38.0	**37.0**	57.0						

Table A.10: Accuracies and F-measure scores (%) achieved by evaluating the features derived from the DeepMoji pre-trained model using SVM, LR, and RF classifiers, respectively.

Dataset	***DeepMoji pre-trained model***											
	Accuracy			***F-measure***								
				average			***positive***			***negative***		
	SVM	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**	**SVM**	**LR**	**RF**
irony	69.2	**73.8**	70.8	68.8	**72.4**	64.0	52.4	**54.1**	29.6	77.3	**81.7**	81.6
sarcasm	57.7	59.2	**67.6**	57.4	58.1	**66.8**	51.6	49.1	**59.6**	62.5	65.9	**72.9**
aisopos	92.1	**94.6**	89.6	92.1	**94.6**	89.4	93.1	**95.3**	91.4	90.7	**93.6**	86.8
SemEval-Fig	88.2	**89.4**	85.7	**87.9**	87.8	79.4	**57.8**	51.4	4.2	93.1	**94.1**	92.3
sentiment140	78.0	**80.8**	79.4	78.0	**80.8**	79.4	78.1	**81.5**	80.2	77.9	**80.0**	78.5
person	**81.3**	80.4	74.5	**81.2**	79.8	67.5	**86.9**	86.7	84.6	**67.2**	62.6	25.3
hobbit	92.0	**92.7**	88.1	91.9	**92.7**	87.7	94.1	**94.7**	91.6	87.3	**88.6**	79.5
iphone6	79.5	**79.7**	78.9	**79.3**	79.0	76.0	85.6	86.0	**86.5**	**64.7**	62.8	51.7
movie	85.9	**86.5**	82.4	**85.0**	84.6	74.7	91.7	**92.2**	90.3	**54.3**	50.0	3.9
sanders	81.3	**82.1**	78.8	81.3	**82.1**	78.6	79.8	**80.6**	75.6	82.6	**83.4**	81.2
Narr	88.1	**89.1**	85.7	88.1	**89.1**	85.5	90.2	**91.0**	88.6	84.8	**86.2**	80.8
archeage	83.4	**83.5**	81.9	83.3	**83.4**	81.5	79.6	**79.7**	76.0	85.9	**86.1**	85.5
SemEval18	79.8	**80.0**	77.4	79.7	**80.0**	77.1	**77.7**	**77.7**	73.5	81.5	**81.9**	80.2
OMD	75.8	**75.9**	73.0	75.3	**75.4**	70.4	**64.9**	**64.9**	52.4	81.5	**81.7**	81.1
HCR	75.2	**75.4**	74.8	72.4	**72.5**	68.5	**42.3**	42.2	27.2	84.2	84.4	**84.8**
STS-gold	86.7	**87.8**	82.0	86.5	**87.5**	80.2	77.6	**79.0**	62.5	90.6	**91.4**	88.2
SentiStrength	79.8	**79.9**	76.3	79.7	**79.8**	75.5	83.1	**83.3**	81.5	74.8	**75.0**	67.1
Target-dependent	**82.1**	82.0	78.6	**82.1**	82.0	78.6	**81.8**	81.7	77.8	**82.3**	**82.3**	79.4
Vader	**89.1**	88.6	82.9	**89.0**	88.3	81.1	**92.3**	92.0	88.8	**81.5**	80.1	63.8
SemEval13	**83.5**	83.4	77.9	**83.1**	82.9	72.7	88.9	**89.0**	86.6	**67.5**	66.6	35.4
SemEval17	84.6	**84.9**	80.3	84.4	**84.7**	79.5	78.3	**78.7**	69.3	88.0	**88.3**	85.5
SemEval16	**84.5**	84.3	78.9	**84.0**	83.8	74.8	**89.7**	89.6	87.1	**68.9**	68.2	42.0
#wins	5	**16**	1	8	**13**	1	8	**14**	2	7	**14**	2
rank sums	42.0	**28.0**	62.0	38.0	**31.0**	63.0						