UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

WORDS, TWEETS AND REVIEWS: LEVERAGING AFFECT KNOWLEDGE BETWEEN MULTIPLE DOMAINS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

CRISTIÁN FELIPE TAMBLAY VEAS

PROFESOR GUÍA:
FELIPE BRAVO MÁRQUEZ

MIEMBROS DE LA COMISIÓN:
CLAUDIO GUTIÉRREZ GALLARDO
JOSÉ BENGURIA DONOSO

SANTIAGO DE CHILE
2020

Resumen

Tres dominios populares de aplicación del análisis de sentimiento son: 1) la clasificación de críticas de películas, 2) la extracción de opiniones en Twitter, y 3) la inferencia de la orientación semántica de las palabras. Los elementos léxicos de estos dominios difieren en su longitud, es decir, las reseñas de películas suelen ser más largas que los tweets y las palabras son obviamente más cortas que los tweets, pero también comparten la propiedad de poder ser anotadas con las mismas categorías de afectos (por ejemplo, positivo, negativo, ira, alegría). Además, los modelos de vanguardia para estos dominios se basan en el mismo enfoque: la formación de modelos de aprendizaje automático supervisado sobre ejemplos anotados manualmente. Este enfoque sufre de un importante problema: los ejemplos anotados son escasos y su obtención requiere mucho tiempo y recursos.

En este trabajo proponemos técnicas de transferencia de afectos entre las palabras, los tweets y las críticas de películas utilizando dos métodos de representación: "static word embeddings" utilizando Word2Vec y "contextualized word embeddings" usando BERT. Empleando estos métodos construimos representaciones compatibles para reseñas de películas, tweets y palabras. Entrenamos y evaluamos modelos supervisados en todas las combinaciones de dominios de origen y destino. Este enfoque es valioso cuando los datos anotados en el dominio de destino son limitados.

Nuestros resultados experimentales muestran que el conocimiento de los afectos puede ser transferido con éxito entre nuestros tres dominios, además, las representaciones contextualizadas tienden a superar a sus homólogas estáticas, y por último, se obtienen mejores resultados de transferencia de aprendizaje cuando el dominio de origen tiene unidades léxicas más largas que el dominio de destino.

Abstract

Three popular application domains of sentiment and emotion analysis are: 1) the automatic rating of movie reviews, 2) extracting opinions and emotions on Twitter, and 3) inferring sentiment and emotion of words. The textual elements of these domains differ in their length i.e., movie reviews are usually longer than tweets and words are obviously shorter than tweets, but also share the property that they can be plausibly annotated according to the same affective categories (e.g., positive, negative, anger, joy). Moreover, state-of-the-art models for these domains are based on the same approach: training supervised machine learning models on manually-annotated examples. This approach suffers from an important bottleneck: manually annotated examples are expensive and time-consuming to obtain and not always available.

In this thesis we propose a method for transferring affective knowledge between words, tweets, and movie reviews using two representation techniques: Word2Vec static embeddings and BERT contextualized embeddings. We build compatible representations for movie reviews, tweets, and words, using these techniques and train and evaluate supervised models on all combinations of source and target domains.

Our experimental results show that affective knowledge can be successfully transferred between our three domains, that contextualized embeddings tend to outperform their static counterparts, and that better transfer learning results are obtained when the source domain has longer textual units that the target domain.

I just have never heard a program speak of love. It is a human emotion.
-No, it is a word. What matters is the connection that the word implies.
Matrix Revolutions (2003)

Agradecimientos

Quisiera agradecer a mi Profesor Guía Dr. Felipe Bravo Marquez por su ayuda, apoyo y consejos para realizar esta memoria. Gracias por darme la oportunidad de aprender tanto sobre este hermoso campo.

Agradezco también a mi familia, a mi padre José Tamblay y a mi madre Marcela Veas por su apoyo incondicional durante todas las épocas de mi vida. Gracias por tenerme tanta paciencia.

A mis hermanos José, Marcela y mis cuñados por su ayuda siempre que los necesité.

A Sophie De Bona, por su compañia y cariño, más aún durante la cuarentena.

A los invaluables amigos que hice durante el DCC y mi estadía en el DIM.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Sentiment analysis attempts to computationally extract people's opinions, emotions and views from natural language texts. A closely related field is affective computing, which focuses on the design of machines capable of recognizing and expressing human emotions [9]. Finally, the field of sentic computing proposes a holistic view of human emotions and natural language that integrates both sentiment analysis and affective computing with other related disciplines such as knowledge representation, linguistics, and psychology [10].

These techniques have been successfully applied in various domains, such as automatic monitoring of public opinion in social media, conducting market research and improving companies' customer service. As a more concrete example, in [25], the authors applied sentiment analysis techniques to tweets related to the 2016 U.S. presidential election, and reported a correlation of 94% with official polls. In [54], the authors predicted movie ratings on RottenTomatoes[1], a website where experts assign ratings to movies, using movie reviews as input data.

A particular property of sentiment and emotions is that they can be found across all types of linguistic units (e.g., words, phrases, sentences, paragraphs, documents) and textual sources (e.g., social media publications, movie reviews, newspapers). In this paper, we study how the affective[2] knowledge between three different domains can be leveraged and ultimately transferred: words, tweets, and movie reviews.

We argue that because of the semantic interaction between words, sentences, and documents, the affective patterns between these three domains are strongly interconnected, something that has been widely studied by the linguistics community as discussed below.

The principle of semantic compositionality claims that the meaning of a sentence is a function only of the meaning of its lexical units, together with how these units are combined [43]. This principle suggests that the meaning of a sentence is determined by the meaning of

[1] www.rottentomatoes.com

[2] In this work we use the term "affect" to encompass both sentiment and emotions.

its individual words as well as the sentence structure. The distributional hypothesis, on the other hand, states that words used in the same contexts tend to have similar meanings [23]. As a consequence, word meanings can be inferred by the contexts in which they occur.

These two linguistic theories propose a conceptual framework of meaning for both words and sentences, which we extend in this work from meaning to affective states such as sentiment and emotions. The relationship between meaning and affect can be argued at both the lexical and sentence level. At the lexical level it can be argued that words with similar meanings (i.e., synonyms) probably express the same sentiment, and similarly, sentences that convey the same meaning using alternative expressions are also very likely to express the same sentiment and the same emotions.

These principles are used in this research to find exploitable patterns in the relationship between words, sentences, and documents. More specifically, in this work we focus on specific sentiment and emotion detection tasks described below.

1.2 Problem Statement

The sentiment analysis tasks that we study are: 1) polarity lexicon induction (PLI) [2], 2) sentence-level sentiment classification (SSC) [40], 3) and document-level sentiment classification (DSC) [3]. The objective of the PLI task is to determine the semantic orientation of a word in a lexicon, which corresponds to classifying whether the word is positive or negative. For example, it classifies the word "happy" as positive and the word "sad" as negative. Meanwhile, the SSC task aims to classify entire sentences as positive or negative. An example of this task would be to classify the tweet "my dog is the best #doglover" as positive[3]. Finally, the DSC task intends to classify entire documents as positive or negative, based on the opinion expressed in them. Movie reviews are a clear example of this task.

We also study two emotion tasks at both word and sentence level[4]. The first task is the detection of word affect intensities (WAI) [39], which consists of associating words with real-valued intensity scores for four basic emotions: anger, fear, sadness, and joy. For example, the word "outraged" has a higher intensity for the emotion anger than the word "agitated". The second task is the detection of sentence-level affect intensities (SAI), which consists of detecting the intensity of emotion felt by the speaker of a tweet [37].

All the above tasks share the property of being addressed by supervised learning algorithms trained on numerical vector representations of their corresponding lexical units and manually annotated labels. However, manually annotating words, tweets, and movie reviews into affective categories can be very time consuming and expensive.

In many practical scenarios, the resources needed for training supervised models (i.e., annotated examples) are not available. A possible solution to this problem, is to adapt models trained from a related domain where training data is available, to the task at hand.

[3]In this work we are making the assumption that tweets are usually formed by a single sentence.

[4]We are not studying emotions at the document level due to the lack of annotated data to experiment with.

In the context of sentiment classification, training a model in the word domain and then applying it in a sentence classification task (or vice versa) has been shown to be useful when training data from the target domain is insufficient [7]. This exercise is commonly known as transfer learning. Transfer learning between two domains refers to the acquisition of knowledge from a source domain and its subsequent application to a target domain. A formal definition of transfer learning is given as follows:

Definition 1.1 (Transfer learning) *Given a source domain $\mathcal{D}_S$ and a learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function f_T in $\mathcal{D}_T$ using the knowledge $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$* [42]

Transfer learning requires that both the source and target domains to be related. As discussed above our three domains: words, tweets, and movie reviews, are related in the sense that they can all be plausibly associated with the same affective categories (e.g, positive, negative, anger, joy).

Each one of our domains expresses affect in a unique way. First, tweets capture someone's mood and thoughts in a short but emotionally charged message. Then, movie reviews comprehensively reflect why someone liked or disliked a movie. Finally, the field of lexical semantics has widely studied the inherent affect of isolated words, which is also referred to as semantic orientation [19]. Another important difference between our domains is the length of their respective linguistic units: movie reviews are typically longer than tweets, and tweets are composed by words.

The core of our transfer learning proposal is to represent the lexical units of each domain with compatible representations (i.e., numerical vectors residing in the same vector space). Afterwards, a classifier can be trained on labeled instances from a source domain, to be later deployed on instances from a target domain. Any of our three domains (i.e., words, tweets, and movie reviews) can interchangeably play the role of source or target domain.

The most important building block that our approach requires, is a model capable of representing lexical units of different lengths as compatible feature vectors. In this work, we adapt two popular resources in Natural Language Processing (NLP) for this purpose: 1) static word embeddings [35], and 2) contextualized word embeddings [17].

The main difference between them is that while static word embeddings assign a fixed representation to each word, contextualized word embeddings provide a variable representation that depends on both the word and its context.

1.3 Objectives

Specifically, we experiment with all the affective tasks and combinations of source and target domains described above, using the following representation models:

- BERT-Base contextualized word embeddings [17].

- Word2Vec [35] static word embeddings trained on the same dataset as BERT-Base (i.e., Wikipedia + BookCorpus[64]).
- Word2Vec [37] static word embeddings trained over a corpus of tweets.

This work's main contribution is a new framework that allows a transparent comparison of static and contextual word embeddings for various scenarios of affective knowledge transfer between words, tweets, and movie reviews. We argue that this approach can be especially valuable when annotated data in the target domain is scarce. Moreover, we also believe that our proposal can benefit the sentic computing community, as it allows for directly transferring affective knowledge from lexical resources such as SenticNet [11] to other domains.

1.4 Roadmap

The remainder of this article is organized as follows. Chapter 2 presents a background in NLP and language representation relevant to this article and a review of related work in affect analysis. In Chapter 3, we describe the proposed method for transferring affective knowledge. In Chapter 4, we present the experiments conducted to evaluate transfer learning tasks and discuss results. The main findings and conclusions are presented in Chapter 5.

Chapter 2

Background and Related Work

This Chapter aims to show transfer learning and affect analysis related work, but it is necessary to introduce many machine learning concepts first, which will be explained in the following sections.

2.1 Classification

A classifier is a function that maps an input vector of one or more dimensions into a discrete space. This study's first experiment maps lexical representations into the sentiment space with two classes, positive and negative. Classification is the task of finding the classifier that fits the best given the input and the problem's output data. To choose the classification algorithm, we need to focus on the structure of the dataset. In the endless assortment of conceivable datasets, there is a wide range of structures that can happen, and a classification algorithm, regardless of how powerful it is, that is searching for one class of structure may miss dependencies of an alternate kind, paying little heed to how simple those might be [26].

We are trying to find linear dependencies between our attributes by the nature of our sentiment datasets, mapping multi-dimensional spaces to $\{-1, 1\}$, representing negative and positive classes, respectively. What matters here is the weighted sum of the numerical attribute values with appropriately chosen weights. Linear models fit the best to our sentiment annotated datasets.

First experiment will apply the logistic regression[1] algorithm, a particular case of the well-known linear regression. It predicts the occurrence of a binary event applying a sigmoid function

$$\sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{2.1}$$

over the linear regression equation.

Specific metrics are used to compare between classifiers. The following metrics are used

[1]This algorithm has two steps. First, it acts as a regression (explained in the next section), calculating the continuous probability of events. It turns into a classifier by including a decision rule, i.e., classify the event as positive if the probability is over 0.5.

		Actual	
		Positive	Negative
Predicted	Positive	TP	FP
	Negative	FN	TN

Table 2.1: Example of contingency table in a binary classification problem.

throughout this study. Refer to Table 2.1 for all the calculations.

1. **Accuracy** The most common metric is accuracy, which is the number of correct predictions over all the predictions. Its formula is

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.2}$$

2. **Precision** Precision is a metric that shows what proportion of the data that was classified as positive, actually was positive, that is,

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.3}$$

3. **Recall** Recall shows what proportion of the data that actually was positive, was classified as positive, i.e.,

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.4}$$

   There is a trade-off between Precision and Recall metrics. For example, to maximize Recall, an algorithm could classify all its inputs as positive, i.e., $FN = 0 \Rightarrow \text{Recall} = 1$. This is undesirable behavior in a good classifier. To solve this problem, we introduce a metric that captures the Precision and Recall behaviors.

4. **F1-score** F1 score is the harmonic mean between precision and recall, the formula is then

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.5}$$

   Precision, Recall, and, consequently, F1-score depends on which the positive class is. We can do the same calculations assuming that the positive class is the negative one, obtaining one F1-score for each class. The F1-score used in this study is the average of F1-scores, that is,

$$\text{F1} = \frac{\text{F1}_{\text{positive}} + \text{F1}_{\text{negative}}}{2} \tag{2.6}$$

   This is usually known as macro-averaged F1-score.

5. **ROC AUC** To understand the ROC AUC (Receiver Operating Characteristic Area under Curve) metric it is necessary to first introduce the concept of ROC Curve. The ROC curve captures the classifier's performance at different thresholds between the false positive rate and the true positive rate, as shown in Figure 2.1. Then, we compute the area under the ROC curve, obtaining the desired metric. ROC AUC score varies between 0 and 1, with a random classifier yielding 0.5.

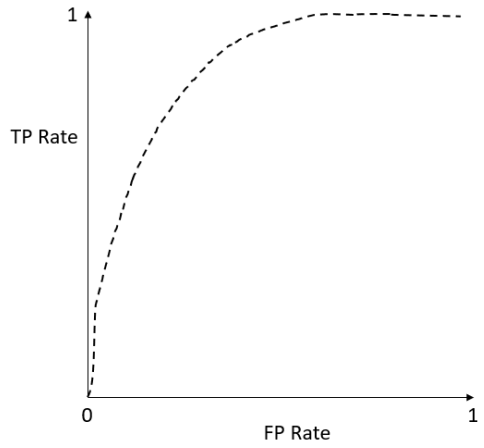

Figure 2.1: ROC curve showing True Positive vs. False Positive rate at different classification thresholds. The area under this curve is known as the ROC AUC score.

6. **Kappa** Cohen's Kappa coefficient is "the proportion of agreement after chance agreement is removed from consideration" [16] which formula is

$$\kappa = \frac{\text{Accuracy} - p_{\text{e}}}{1 - p_{\text{e}}} \tag{2.7}$$

$$\text{where } p_{\text{e}} = \frac{TP + FP}{TP + FP + FN + TN} \cdot \frac{TP + FN}{TP + FP + FN + TN} + \frac{FN + TN}{TP + FP + FN + TN} \cdot \frac{FP + TN}{TP + FP + FN + TN} \tag{2.8}$$

Landis and Koch [28] characterized κ values < 0 as indicating no agreement, 0–0.20 as slight, 0.21–0.40 as fair, 0.41–0.60 as moderate, 0.61–0.80 as substantial, and 0.81–1 as almost perfect agreement.

2.2 Regression

While a classifier solves a discrete space classification problem, the regressions are useful when the label space is continuous. The second experiment maps lexical representations into the emotion intensity space, labeled continuously in [0,1]. This experiment's output is a real number, so the previous section metrics are not useful to measure the regressions' performance.

The regression we are using in this study is a Support Vector Regression [18], a special case of a Support Vector Machine. This regression considers the points that are within a decision boundary line, noted as ε.

The following metrics approach this problem, indicating in diverse manners how exact the prediction was. In our study we compare between the regression's estimated target values, noted as $\hat{y}_{\text{i}}$, and the actual target values, y_{i}. These notations will be used in all the calculations.

1. **Correlation** Pearson correlation coefficient is a measure of the strength of a connection between two variables. Its formula is:

$$R = \frac{\sum_{\mathrm{i}=1}^{N}(\hat{y}_{\mathrm{i}} - \bar{\hat{y}})(y_{\mathrm{i}} - \bar{y})}{\sqrt{\sum_{\mathrm{i}=1}^{N}(\hat{y}_{\mathrm{i}} - \bar{\hat{y}})^2}\sqrt{\sum_{\mathrm{i}=1}^{N}(y_{\mathrm{i}} - \bar{y})^2}} \text{ where } \bar{X} = \frac{1}{N}\sum_{\mathrm{i}=1}^{N} X_{\mathrm{i}} \tag{2.9}$$

2. **MAE** Mean Absolute Error is the average absolute error between the regression prediction and the target value which formula is:

$$\text{MAE } = \frac{1}{N}\sum_{\mathrm{i}=1}^{N} |y_{\mathrm{i}} - \hat{y}_{\mathrm{i}}| \tag{2.10}$$

3. **RMSE** Root Mean Square Error is the square root of the average of squared differences between the regression prediction and the target observation, that is,

$$\text{RMSE } = \sqrt{\frac{1}{N}\sum_{\mathrm{i}=1}^{N}(y_{\mathrm{i}} - \hat{y}_{\mathrm{i}})^2} \tag{2.11}$$

   RMSE value is proportional to the size of the squared error, so larger errors have a disproportionately large effect on RMSE, thus sensitive to outliers.

Once we have a way to measure classifiers' and regressions' performance, we need to explain how to reshape lexical information into numerical data. The following section addresses this problem.

2.3 Representations

Representation is the act of transforming the input data into a vector of fixed dimension that portrays the data. Different kinds of data require distinct types of representations. As an example, in computer vision, we can represent images directly with their RGB matrices. In other fields, such as audio processing, we can obtain representations based on frequency and amplitude over time. Unfortunately, for words and sentences, there is no straightforward numerical representation.

Throughout Natural Language Processing history, the way we understand and learn representations of language has evolved. In classical machine learning, the developer had to create the features and representations himself. This exercise required a lot of domain-specific knowledge engineering. These systems were fragile and did not work well outside the tasks for which they were designed.

An example of this is the bag-of-words model [62]. In this model, a sentence is represented as a collection of the individual words it contains. This vector can count the appearance of each word or if it is present or not. This feature extraction method treats sentences as a set of unsorted words, losing some of its meaning. For example, the sentences "I am happy today. Yesterday I was sad" and "I am sad today. Yesterday I was happy" would have the same representation under this model, as shown in Figure 2.2

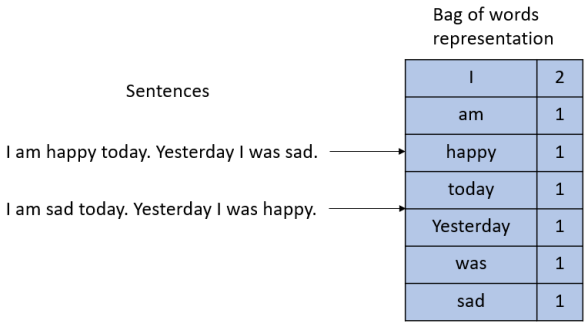

Figure 2.2: Example of two sentences yielding the same bag of words representation.

Word-context matrices take a more sophisticated approach. Rows and columns of the matrix represent the words of the vocabulary. Each element of the matrix holds the number of occurrences of a word that occurs simultaneously with another word in the same context. GloVe model takes this approach [44].

In recent years, there has been a paradigm shift in natural language research. The focus shifted from hand-made rules to unsupervised methods that can learn from the underlying relationships of language. Modern approaches for word and sentence representation employ neural network models discussed below.

2.4 Neural Networks

A neural network is an artificial model inspired by biological neural networks in animals' brains, composed by layers of neurons. The first layer is called the input layer, followed by one or more hidden layers, and finishing in the output layer. A vector representing the data must be supplied to the input layer. Each neuron is given an activation function, a bias, and a set of weights. This function defines the output given a set of inputs, and the weights combined with the bias represent how important each of the previous neurons is. Weights and bias are adjusted by training the network.

In fully connected networks, every neuron is connected to all neurons in the next layer. Figure 2.3 shows an example of a fully connected network. In this example, given$\vec{x} = [x1, x2, x3]$, we can obtain the output $\vec{y} = [y1, y2]$ with the following formula, known as the mathematical representation of a neural network:

$$\vec{y} = g(f(\vec{x}W^1 + \vec{b^1})W^2 + \vec{b^2}) \tag{2.12}$$

This formula is easily extendable to a network with an arbitrary amount of hidden layers. Neural networks can be used for predictive tasks and to learn data representations automatically, usually called representation learning.

Self-supervised learning is a promising alternative where neural networks learn without explicit supervision in a way that helps with downstream performance on tasks of interest. Datasets do not need to be manually labeled by a human. They are labeled by finding and exploiting the relations between distinct inputs. Different representations can entangle and hide the different explanatory factors of variation behind the data [4].

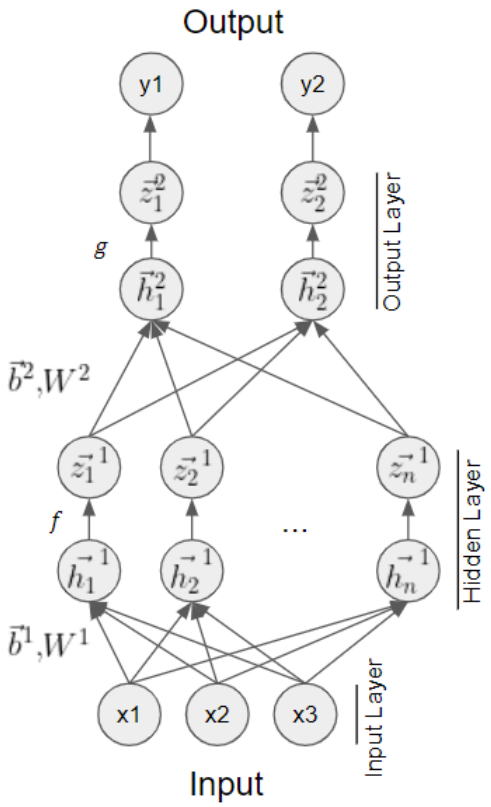

Figure 2.3: Fully connected network with an input of size 3, a hidden layer of size n, and an output layer of size 2. f and g are the activation functions of neurons. W^1 and W^2 represent the weight matrices. $\vec{b}^1$ and $\vec{b}^2$ are the biases of each layer.

After introducing the leading tools, neural networks, classifiers, and regressions, we can discuss how they are applied in Natural Language Processing tasks in the next section.

2.5 Applications in human language

What Natural Language Processing aims for is a way for computers to process and analyze our language and somehow find hidden semantic relationships.

The most straightforward way to turn a word into a vector is to use the one-hot encoding model, that is, a sparse vector as long as the vocabulary, containing a 1 in the word's index. Having this model in mind, the natural extension was the bag of words model that was used for many years, with its first mentions in the 1950s [23]. As computing power kept growing exponentially, word and sentence representation models experienced a drastic evolution during the last decade and focused on obtaining more reliable representations of words and sentences.

We can now process a massive amount of data to train our models, and such unlabeled data is within everyone's reach. For NLP tasks, neural networks focus on learning vector representations of words, called "word embeddings". There are two paradigms of word embeddings, static word embeddings, and contextualized word embeddings.

On the one hand, static word embeddings define an injective map f between words and their respective d-dimensional representation, i.e.,

$$f : w_{\mathrm{i}} \to \mathbb{R}^{\mathrm{d}} \tag{2.13}$$

This means that one word has only one representation. On the other, contextualized word embeddings define an injective map f from a n length sequence of words to their multiple d-dimensional representations, i.e.,

$$f : \{w_1, w_2, ..., w_n\} \to (\mathbb{R}^{\mathrm{d}})^n \tag{2.14}$$

In this paradigm, each word has as many representations as contexts.

A salient property of these two models is their ability to learn from massive amounts of unlabeled corpora, which can be freely obtained from the Web.

Below we define the most relevant architectures that can exploit this property and are based on the previous embedding paradigms:

- **Word2Vec** [35]: This is a two-layer neural network. Training of it takes as input a large corpus and returns a set of feature vectors for words in that corpus. Word2Vec model is used for learning vector representations of words. This model uses the static word embeddings paradigm.
- **Recurrent Neural Network** (RNN): Neural networks become recurrent when the transformation is applied repeatedly to a series of given inputs and produce a series of output vectors. For example, long short-term memory (LSTM) is an RNN architecture

in which the neuron is fed continuously with the current state and can decide whether to add new information or remove it. LSTMs can keep track of arbitrary long-term dependencies relatively well. LSTMs helped develop new text representation models such as ELMo [45], using contextualized word embeddings.

- **Attention**: An attention function is a map f from a query and multiple (key,value) pairs, i.e., f : query $\times$ (keys,values)$^n \rightarrow \mathbb{R}^{\mathrm{d}}$. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a function w : query $\times$ key $\rightarrow$ weight [56].
- **Encoder - Decoder**: An encoder is a stack of several recurrent units or attention mechanisms where each one accepts a single element of the input sequence, collects information for that element, and propagates it forward. The last vector encapsulates all the information about the input sequence. A decoder takes this vector as input; it goes through several other recurrent units or attention mechanisms and gives an output [58].
- **Transformer**: This neural network relies entirely on an attention mechanism to draw global dependencies between input and output. The Transformer follows encoder-decoder overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder. The Transformer allows for significantly more parallelization than RNNs [56].

Word2Vec [35] and GloVe [44] are possibly the most popular static word embeddings models that can be efficiently trained on large corpora. These models can effectively capture semantic and syntactic properties of words, by exploiting their surrounding words in a fixed size window. [8]. In [61], the authors employed these models for sentiment classification yielding results with an accuracy above 85%. Static word embeddings have improved the performance of a wide range of NLP tasks, such as machine translation [65] and text classification [27]. The main limitation of static word embeddings is that they conflate all the different meaning of polysemous words into a single representation.

Over the last two years, novel contextualizers such as ELMo, BERT, [17] and XLNet [**?**] have dramatically improved performance for many (NLP) tasks, including sentiment analysis. ELMo and its predecessors extract context-sensitive features from left-to-right and right-to-left text representation models. This model advanced the state-of-the-art for several NLP benchmarks [45]. With the release of the Transformer architecture [**?**], LSTM-based neural networks, such as ELMo, started to fall behind mainly because Transformers can deal efficiently with long-term dependencies [55]. BERT and XLNet are examples of deep learning architectures that use the Transformer architecture [56]. These architectures can concurrently process all inputs of a sequence, leading to faster training times. Because of this, novel Transformer-based architectures quickly gained more attention than LSTM-based ones. They can exploit the advantages of the new graphics processing units (GPUs) and be trained over massive datasets. The standard way to use these models consists of a pre-training phase, in which the model is built in a self-supervision scheme over a large corpus, and then a fine-tuning phase in which the model is adapted to the target task where labeled examples are available. Alternatively, a pre-trained contextualizer can be used as a feature extractor without the need for fine-tuning[48].

As discussed so far, the idea of pre-training a neural network model from a large corpus to

obtain language representations is now a standard practice in NLP. The success of BERT led to the development of many other models based on it, many of which have reported state-of-the-art results in various NLP tasks. Examples of these are DistilBERT [51], RoBERTa [30], ERNIE [63], BETO [13] and ELECTRA [15]. This success is the motivation for our work, and to the best of our knowledge, there are no studies on using BERT to transfer affective knowledge across multiple domains.

2.6 Related work

In this section, we review works on transfer learning, sentiment analysis and sentic computing that are relevant to this article.

As explained in the introduction section, transfer learning refers to the acquisition of knowledge from one source domain and its application to a related target domain. Prior work on transfer learning focuses on domain adaptation, i.e., training a classifier in one domain, e.g., internet blogs, and deploying it in a domain where a different terminology is used, e.g., newspapers. [21].

A transfer learning framework for transferring sentiment knowledge between words and tweets based on the aggregation of instances has been proposed in [7]. This work provides the foundations of our work, representing tweets and words using compatible representations. Words are represented as the collection of tweets in which they occur and sentences are the centroid of each word representation. However, this thesis does not make use of any novel deep learning architecture, which is the main goal of our study. Moreover, our work goes beyond sentiment, analyzing also the intensities of emotions.

WordNet [36] is an English database created in the 90s that links nouns, verbs, adjectives, and adverbs to form sets of synonyms called synsets. These synonyms are, in turn, connected by semantic relationships that determine the definitions of words. This resource is particularly useful for handling polysemic words, that is, words with multiple meanings. In [19], a polarity classifier was trained using the WordNet database on a set of positive and negative labeled words. For each unknown word, related terms in which the polarity is known are retrieved (e.g., synonyms, antonyms) and used to classify the unknown polarities by assuming that synonyms must have the same polarity and antonyms the opposite. The resulting expanded lexicon is employed to determine the polarity of sentences by adding up the polarity of their words. This process can be considered as a transfer learning from the domain of words to sentences.

There was another approach in [24], where sentiment annotated documents such as Amazon reviews help transferring knowledge to aspect-level sentiment classification. This study applies LSTM neural network architecture combined with annotated datasets.

In [24], an approach is adopted where sentiment annotated Amazon reviews help transfer knowledge to the aspect-level sentiment classification task using an LSTM neural network architecture.

An algorithm based on the joint regularization of a bipartite graph of labeled and unlabeled

nodes was proposed in [52]. The nodes correspond to documents and words, and sentiment labels are propagated from the labeled nodes to the unlabeled one using regularized least squares.

A framework for incorporating word knowledge to text sentiment classification was proposed in [33] using a generative Naïve Bayes model. In this thesis, the authors refine the knowledge of a sentiment lexicon with annotated blogs. Then, they proceed to classify movie reviews, political blogs, and IBM products reviews.

A recursive neural tensor network capable of learning the sentiment of lexical units of different granularities such as words, phrases (including negated expressions), and sentences was proposed in [53]. An unsupervised learning approach for Twitter sentiment analysis using three domain-independent sentiment lexical resources has been developed in [14]. This study has shown that lexicons can infer the sentiment of tweets by averaging the sentiment values of their lexical units.

We next present other works that fall within the sentic computing paradigm.

A relevant problem in sentiment analysis is contextual polarity ambiguity. This problem is tackled in [59] by performing word polarity disambiguation using Bayesian model with opinion-level features. In [31], the authors proposed a novel neural network architecture and two extensions of the traditional LSTM architecture for the task of targeted aspect-based sentiment analysis, i.e., recognizing aspect categories and assigning their polarity. The work of [47] introduces a novel paradigm for concept-level sentiment analysis that combines the Hourglass of Emotions [12], common-sense computing, and deep learning techniques. In [1], the authors propose a stacked ensemble method for predicting real-valued intensities of emotion and sentiment. The method combines the outputs of various deep learning and standard feature-based models using a feedforward network. Finally, [49] applied a convolutional neural network for sentiment classification of Hindi movie reviews that outperformed many other machine learning baselines.

Chapter 3

Proposed Methodology

This Chapter starts by introducing our transfer learning method, followed by a description of how to represent words and sentences using static word embeddings and ends with another description of how to achieve the same goal using contextualized word embeddings.

3.1 Method

This section introduces a new method for transferring affective knowledge between words, tweets, and movie reviews based on two paradigms of language representation. As stated in Chapter 1, the principle of semantic compositionality states that the meaning of a sentence depends on its lexical elements together with the form in which they are composed. On the other hand, the distributional hypothesis suggests that word meanings can be inferred by the contexts in which words occur.

These two semantic theories lay the foundation of our study, giving us the tools to jump from the word domain to the sentence or document domains and vice versa. Our method is inspired by previous work to transfer sentiment knowledge word and tweets [7]. The method is illustrated in Figure 3.1. One domain will act as the source domain $\mathcal{D}_S$ and the other as the target domain $\mathcal{D}_T$. The transfer learning procedure is described in the following steps:

1. Set the language representation model, choosing between static or contextualized word embeddings.
2. Represent both the training instances from the source domain $\mathcal{D}_S$ and the testing instances from the target domain $\mathcal{D}_T$ with the chosen representation model to obtain compatible k-dimensional vectors, using aggregation functions explained in the next sections.
3. Train a predictive model on the training instances represented according to previous step.
4. Apply the resulting model to the corresponding testing instances from $\mathcal{D}_T$.

As an example, let us assume a scenario of transferring sentiment knowledge between words to tweets. We have the following annotated words {“hate”: negative, “love”: positive}

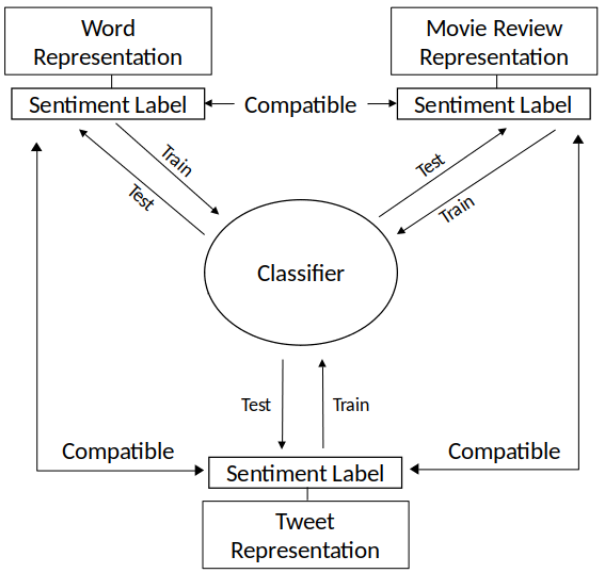

Figure 3.1: Method for transferring sentiment knowledge between words, tweets and movie reviews

and the following annotated tweets {"I detest this movie": negative, "I like how he performed": positive}. First, we represent both datasets with the chosen paradigm, obtaining compatible representations. Then, we train a classifier on the word's dataset and predict the sentiment of the target tweets, using the corresponding representations in both cases. Finally, we evaluate the overall performance of the classifier on the target domain. We expect that after obtaining sentiment knowledge from the word domain, the classifier will be able to successfully classify the sentiment of tweets as a result of the semantic relationships between these two domains. The same could be done in the other direction, training on the tweet domain and testing on the word domain.

It is worth pointing out that the classifier will not be further adjusted (i.e., by tuning hyper-parameters) using data from the target domain, since we are interested in evaluating the model's transfer learning capabilities in a scenario where there is no training data in the target domain.

Having introduced the fundamental ideas of our methodology, we are now in a position to describe how words, sentences, and documents are represented using our two aforementioned paradigms.

3.2 Static word embeddings

As discussed in Chapter 2, static word embeddings define an injective mapping between words and vectors:

$$f : w_{\mathrm{i}} \rightarrow \mathbb{R}^{\mathrm{d}}. \tag{3.1}$$

This means that a word has one and only one representation. These static word embeddings are obtained by training neural networks on large corpora. There are some hyper-parameters that need to be adjusted such as the context's size and the embeddings vectors' size. The

context size is usually known as the window size and refers to the number words surrounding each target word that are considered during the learning phase. The embeddings vectors' size is usually set between 100 to 300 dimensions. Very low dimensional embeddings are usually incapable of capturing rich semantic information. On the other hand, there is no evidence of significant gains by increasing the dimensionality after a certain point.

Static word embeddings need an aggregation function to pass from word embeddings to sentence embeddings. Usually, this aggregation function is a linear map over the individual word embeddings in a sentence. We will use th|e "average" aggregation function. Out-of-vocabulary words do not have a representation in some static word embeddings models, so they are removed. Formally, given a collection $\{w_1, ..., w_n\} \in \Sigma^+$ of words in a sentence where Σ are the symbols of the language. First we remove out-of-vocabulary words, resulting in a collection $\{w_1, ..., w_k\}$. Then we obtain the embedding as follows:

$$\text{Embedding}(\{w_1, ..., w_n\}) = \frac{1}{k} \sum_{\text{i}=1}^{k} \text{Embedding}(w_{\text{i}}) \tag{3.2}$$

If $\frac{k}{n} < 0.8$, i.e. more than 20% of tokens are unknown, the sentence is discarded. Words do not need an aggregation function as they are represented passing them through the model.

3.3 Contextualized word embeddings

As shown in Chapter 2, contextualized word embeddings define an injective mapping between a sequence of tokens to a sequence of vectors:

$$f : \{w_1, w_2, ..., w_n\} \to (\mathbb{R}^{\text{d}})^n. \tag{3.3}$$

In this paradigm, each word has as many representations as contexts where the word occurs.

More specifically, contextualizers receive a sequence of known tokens as input. These tokens are obtained by a special tokenizer, which maps words to ids in the model's dictionary. As an example, BERT uses WordPiece tokenizer in which most frequent combinations of the symbols in the vocabulary are iteratively added to the vocabulary. This is useful for treating rare and out of vocabulary words, e.g., splitting the word *strawberry* into *straw* and *#berry* that are more common. By the nature of contextualizers, sentences do not need any special treatment. They are passed directly through the tokenizer and then fed into the model. It is essential to add that BERT supports up to 512 tokens, so any sequence above that is truncated. In the case of words, we consider them a single word sentence and then use the same procedure. Impressively, this way to represent words yields good results.

There are many ways to obtain words or sentence representations using contextualizers. The most common approach is to average or sum the model's output layer. In BERT, we can also use the output of the first token, known as the CLS token, which is usually employed for text classification tasks. These aggregation methods are used to represent both sentences and word because words can also have multiple tokens under BERT's tokenization scheme.

Chapter 4

Experiments

In this Chapter, we report transfer learning experimental results. We divide it into five sections. The first section describes the datasets and lexicons used. Then, we present the representation models used in section 4.2. Next, sections 4.3 and 4.4 report sentiment experiments and discussions. Finally, in section 4.5 and 4.6 we describe and discuss our emotion experiments.

4.1 Data

Our sentiment experiments require three annotated resources for training and evaluation purposes, each of which coming from one of our domains: documents, sentences (or tweets), and words.

The document domain dataset is a collection of $50,000$ English reviews from the Internet Movie Database (IMDB) dataset [32]. This dataset consists of an equal number of positive and negative reviews, considering only the highly polarized ones. We use the official split of this dataset, that is, 50% training and 50% test.

The second dataset is the SemEval-2014 Task 9 [50] corpus, consisting of $5,232$ positive and $2,067$ negative tweets annotated by Amazon Mechanical Turk. We split the dataset stratified and randomly into 50% training and 50% testing for evaluating the transfer learning tasks.

We consider the *metaLex* dataset [6] as the word domain lexicon. This resource is built from the combination of four existing lexicons: *MPQA* [57], *Bing Liu* [29], *Afinn* [41], and *NRC-emotion lexicon* [38] resulting in $17,271$ positive, neutral and negative words. We kept only the positive and negative words and discarded the ones with conflicting polarities by different lexicons. This results in a collection of $10,183$ annotated words. We split this lexicon using random and stratified partitions of 67% training and 33% testing instances. The main properties of the three datasets are summarized in Table 4.1.

Dataset	NRC	WASSA
Anger Instances (Train)	994	941
Anger Instances (Test)	489	760
Fear Instances (Train)	1,183	1,257
Fear Instances (Test)	582	995
Sadness Instances (Train)	870	860
Sadness Instances (Test)	429	673
Joy Instances (Train)	849	897
Joy Instances (Test)	418	714

Table 4.2: Emotion datasets properties. Words in NRC lexicon may appear in multiple emotions.

Dataset	metaLex	IMDB	SemEval 2014
Positive Instances (Train)	2,525	12,500	2,642
Negative Instances (Train)	4,295	12,500	1,007
Positive Instances (Test)	1,244	12,500	2,590
Negative Instances (Test)	2,116	12,500	1,060

Table 4.1: Sentiment datasets properties.

We also evaluate the task of transferring four emotion intensities (anger, fear, sadness, and joy) between words and tweets. The annotated tweets are taken from the WASSA-2017 Shared Task on Emotion Intensity [37]. This dataset is composed of $7,097$ tweets and is divided into 4 separate datasets, each for a different emotion: *anger*, *fear*, *sadness*, and *joy*. Each tweet receives a real-valued score that determines the strength of the corresponding emotion in a range between 0 and 1. These tweets were annotated using the best-worst scaling technique. We use the official training and testing of this dataset as shown in Table 4.2.

In the matter of emotion in words, we chose the NRC Affect Intensity Lexicon v0.5 (NRC-AIL) [39]. This lexicon contains intensity scores for four basic emotions: *anger*, *fear*, *sadness*, and *joy*, rated between 0 and 1, as in WASSA, making both datasets compatible. This lexicon was also built using the best-worst scaling technique. In this lexicon, each of the $4,192$ words may have multiple emotions. To deal with this, we divided the dataset into each emotion first and then we did a random 67% train and 33% test split. NRC lexicon partitions are specified in Table 4.2. We introduce the representation models used in the experiments in the next section.

4.2 Representation models

As discussed in Chapter 3, two paradigms are used to represent words and sentences: static and contextualized word embeddings.

Regarding the contextualized word embeddings we use the BERT-base model in our experiments. This model has 12 layers (transformer blocks), 12 attention heads, and 110 million

parameters and was trained over the union of two text corpora, The English Wikipedia and BookCorpus [64],

Regarding the static word embeddings, we use two Word2Vec [35] models, which we refer to as General purpose embeddings and Edinburgh embeddings.

We trained General purpose embeddings on the same data as BERT-base (English Wikipedia + BookCorpus), setting the embedding dimension to 300 and a window size of 15. The rationale behind this selection of corpora is to later compare these embeddings' performance over the BERT-base model, which is also trained on those datasets.

We also use Edinburgh embeddings, a Word2Vec model trained over the Edinburgh dataset [46] consisting of 10 million tweets. The hyper-parameters of this model were calibrated on an emotion classification task [5] and correspond to a window size of 5 and 400 dimensions.

Having presented the representation models and the datasets, we analyze the results for the sentiment experiments on the next section.

4.3 Sentiment Experiments

In this section, we report the results of sentiment classification experiments using static and contextualized word embeddings. We study the effect of training a logistic regression classifier on different domains and then testing on words, tweets, and movie reviews, all of them labeled by positive or negative sentiment.

We use Weka[1] [22] for the classification tasks and a logistic regression included in the LibLINEAR [20] package. All parameters are at their default values except the SVMType, where we chose an L2-regularized logistic regression.

To measure the performance of this binary classifier, we use three metrics: ROC AUC score, macro-averaged F1 score, and Cohen's Kappa score. Table 4.3 shows the results of the embeddings trained over BookCorpus and the English Wikipedia, and Table 4.4 shows the results using Edinburgh embeddings.

[1] https://www.cs.waikato.ac.nz/ml/weka/

		Test domain			
		Word	IMDB	SEval	
Train domain	Word	0.949	0.784	0.769	AUC
		0.881	0.603	0.684	F1
		0.7602	0.2703	0.3677	Kappa
	IMDB	0.799	0.915	0.715	AUC
		0.735	0.836	0.645	F1
		0.469	0.6712	0.2929	Kappa
	SEval	0.870	0.794	0.848	AUC
		0.791	0.740	0.737	F1
		0.5824	0.4294	0.4756	Kappa

Table 4.3: General purpose static embeddings ROC AUC, F1 and Kappa scores from multiple train and test domains

		Test domain			
		Word	IMDB	SEval	
Train domain	Word	0.866	0.785	0.824	AUC
		0.782	0.336	0.359	F1
		0.5672	0.0025	0.0782	Kappa
	IMDB	0.773	0.857	0.828	AUC
		0.701	0.776	0.726	F1
		0.4008	0.5526	0.4553	Kappa
	SEval	0.741	0.763	0.856	AUC
		0.631	0.590	0.559	F1
		0.2821	0.2451	0.1928	Kappa

Table 4.4: Edinburgh static embeddings ROC AUC, F1 and Kappa scores from multiple train and test domains

In both Table 4.3 and Table 4.4, the first column specifies the training domain of the logistic regression for the General Purpose embeddings (GP) and Edinburgh embeddings (Edin). The following columns specify the target domain on which the logistic regression was tested. As an example, General purpose embeddings obtain an AUC score of 0.949, a macro-averaged F1 score of 0.881 and a Kappa score of 0.7602 when trained and tested over the word domain.

Table 4.5 shows the results for both the average (AVG) and CLS BERT representations. The first column specifies the training domain, and each consecutive column specifies a testing domain along with the corresponding representation method used in the experiment. For instance, BERT obtained an AUC score of 0.925, a macro-averaged F1 score of 0.848 and a Kappa score of 0.6958 when trained and tested on the word domain using the CLS representation.

		Test domain						
		Word CLS	Word AVG	IMDB CLS	IMDB AVG	SEval CLS	SEval AVG	
Train domain	Word CLS	0.925	0.905	0.543	0.612	0.480	0.638	AUC
		0.848	0.818	0.483	0.561	0.260	0.589	F1
		0.6958	0.6368	0.0483	0.1678	-0.0047	0.1957	Kappa
	Word AVG	0.577	0.959	0.500	0.648	0.543	0.807	AUC
		0.543	0.892	0.486	0.572	0.237	0.717	F1
		0.0895	0.7826	-0.0044	0.1995	-0.0008	0.4353	Kappa
	IMDB CLS	0.646	0.913	0.843	0.816	0.617	0.831	AUC
		0.529	0.802	0.770	0.738	0.572	0.725	F1
		0.1586	0.6104	0.5389	0.4787	0.1429	0.4582	Kappa
	IMDB AVG	0.549	0.718	0.483	0.924	0.507	0.759	AUC
		0.275	0.633	0.334	0.845	0.515	0.679	F1
		0.002	0.2718	0.0005	0.69	0.0397	0.3581	Kappa
	SEval CLS	0.642	0.631	0.509	0.551	0.827	0.561	AUC
		0.566	0.584	0.338	0.486	0.706	0.514	F1
		0.1869	0.1681	0.0032	0.0414	0.4131	0.079	Kappa
	SEval AVG	0.683	0.899	0.512	0.794	0.555	0.930	AUC
		0.271	0.810	0.458	0.669	0.226	0.836	F1
		0.0003	0.6199	0.0082	0.3573	-0.0003	0.6717	Kappa

Table 4.5: BERT contextualized embeddings ROC AUC, F1 and Kappa scores from multiple train and test domains, varying between CLS and AVG representations.

We summarize these results in Table 4.6. This table shows the winner representation configuration for each task (notice that, except for the results of the diagonal, all results correspond to transfer learning tasks).

		Test domain			
		Word	IMDB	SEval	
Train domain	Word	0.959 AVG	0.785 Edin	0.824 Edin	AUC
		0.892 AVG	0.603 GP	0.717 AVG	F1
		0.7826 AVG	0.2703 GP	0.4353 AVG	Kappa
	IMDB	0.913 CLS/AVG	0.924 AVG	0.831 AVG	AUC
		0.802 CLS/AVG	0.845 AVG	0.752 AVG	F1
		0.6104 CLS/AVG	0.69 AVG	0.4582 AVG	Kappa
	SEval	0.899 AVG	0.794 GP=AVG	0.930 AVG	AUC
		0.810 AVG	0.740 GP	0.836 AVG	F1
		0.6199 AVG	0.4294 GP	0.6717 AVG	Kappa

Table 4.6: Winner configuration from all models ROC AUC, F1, and Kappa scores from multiple train and test domains. AVG represents BERT model with Average to Average embeddings meanwhile CLS/AVG represents BERT with CLS to Average embeddings.

4.4 Sentiment Discussion

We will start by discussing results of static word embeddings. We should recall that a good classifier aims to maximize AUC, F1, and Kappa scores. As we can see, static word embeddings excel at representing words for training and testing over the word domain.

Both static embeddings exhibit relatively high AUC scores, i.e., Edinburgh embeddings have better performance at tweet to tweet sentiment classification task and General purpose embeddings have better performance at movie review to movie review sentiment classification task. From word to movie reviews, both models have a high AUC score. This result indicates that both embeddings have the same ability to extract knowledge from words even though they were trained on different datasets.

Edinburgh embeddings have a better AUC score when transferring from words to tweets. This result is expected as this model was trained over 10 million tweets and has unique tokens for user mentions and URLs, so the test domain has a more reliable representation. Word to tweet transfer using static embeddings are in line with previous AUC score results obtained by [7]. Meanwhile from larger domains to word, General purpose embeddings have better results.

Almost every high AUC score in General purpose embeddings comes along with high macro-averaged F1 and Kappa scores. Interestingly, some Edinburgh embeddings transfer tasks have very low F1 and Kappa scores, having at the same time high AUC scores. A possible explanation for this is that the decision threshold can be shifted when training and testing in different domains.

Secondly, we analyze the results with BERT contextualized word embeddings. Our first experiments with BERT used the CLS representation in both the training and testing domains, but some results were inconsistent, as they had negative Kappa score and very low F1 score, such as Word CLS to Tweet CLS. This motivated the evaluation of different ways to represent training and testing domain using BERT. Best results from word domain to word domain are obtained by averaging BERT output layer. This is valid in all other cases when the source and target domains are the same.

A surprising result is that most of the best results are obtained by averaging the last hidden layer of BERT in both training and testing domains, except in movie reviews. Representing movie reviews with the CLS token at the train domain has outstanding results when testing at a different domain represented by the average representation. One possible explanation is that the CLS token somehow manages to synthesize information better when the domain is longer.

Transferring from the word domain to the movie reviews domain is not as good as the opposite. This behavior appears to be recurring when transferring from smaller to longer lexical units. Based on that, we claim that BERT is better extracting contextualized sentiment information from extensive domains and applying this knowledge to smaller domains.

Comparing both of these embeddings paradigms, we can assert that BERT contextualized word embeddings outperform the static word embedding models for training and testing over

the same domain. This is also true when transferring from a larger to a smaller domain. Word2Vec is a worthy choice when transferring from a smaller to a larger domain. Training and testing on the same domains are an upper bound for all the transfer learning tasks.

Having discussed the sentiment experiments, we proceed with the emotion experiments.

4.5 Emotion Experiments

In this section, we report the results of emotion intensity detection using static and contextualized word embeddings. General purpose embeddings are out of this experiment because previous sections show that Edinburgh has a better performance at representing tweets. The same for BERT's CLS representation, its average representation had consistent results in previous experiments for both train and test domains. The main difference with previous tasks is that in this case we focus on a regression task. Our dataset covers four different emotions: *anger*, *fear*, *sadness*, and *joy*, and each emotion is continuously rated between 0 and 1 rather than in a discrete space. Weka along the LibLINEAR[20] package was used to train a support vector machine regression model, setting the SVMType to L2-regularized L2-loss support vector regression (dual) with the regularization parameter C set to 1.

To measure the performance of our regression models, we use three metrics: Pearson Correlation (COR), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). Classification performances of *anger*, *fear*, *sadness*, and *joy* emotions for both BERT and Edinburgh embeddings are shown in Tables 4.7, 4.8, 4.9, and 4.10. As in the previous experiments, the first column specifies the training domain of the support vector machine regression. Each column afterward specifies at which domain the support vector machine regression was tested along with the representation method used.

		Test Domain				
		Word		Tweet		
		Edin	BERT	Edin	BERT	
Train domain	Word	0.4626	0.6254	0.2526	0.1716	COR
		0.1494	0.136	0.1484	0.2115	MAE
		0.1883	0.1724	0.1863	0.2664	RMSE
	Tweet	0.0938	0.2941	0.4838	0.5816	COR
		0.2003	0.1864	0.1264	0.1215	MAE
		0.2451	0.2296	0.154	0.1516	RMSE

Table 4.7: Anger transfer learning between word and tweet domains using BERT and Edinburgh (Edin) representations. Metrics shown are Pearson Correlation (COR), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE)

		Test Domain				
		Word		Tweet		
		Edin	BERT	Edin	BERT	
Train domain	Word	0.5053	0.6621	0.2937	0.3032	COR
		0.1429	0.1267	0.1746	0.2138	MAE
		0.1784	0.1631	0.2163	0.2665	RMSE
	Tweet	0.3423	0.4714	0.5435	0.6374	COR
		0.2056	0.1743	0.1439	0.1332	MAE
		0.2574	0.2184	0.1754	0.1674	RMSE

Table 4.8: Fear transfer learning between word and tweet domains using BERT and Edinburgh (Edin) representations. Metrics shown are Pearson Correlation (COR), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE)

		Test Domain				
		Word		Tweet		
		Edin	BERT	Edin	BERT	
Train domain	Word	0.5156	0.677	0.2812	0.4176	COR
		0.1491	0.125	0.2025	0.2931	MAE
		0.1809	0.157	0.2501	0.3582	RMSE
	Tweet	0.3446	0.4644	0.6013	0.6886	COR
		0.1865	0.2453	0.1454	0.1222	MAE
		0.2313	0.2915	0.1751	0.1535	RMSE

Table 4.9: Sadness transfer learning between word and tweet domains using BERT and Edinburgh (Edin) representations. Metrics shown are Pearson Correlation (COR), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE)

		Test Domain				
		Word		Tweet		
		Edin	BERT	Edin	BERT	
Train domain	Word	0.5966	0.5653	0.3587	0.3373	COR
		0.1359	0.1478	0.1789	0.3322	MAE
		0.1675	0.1873	0.2148	0.3912	RMSE
	Tweet	0.4001	0.403	0.5708	0.6146	COR
		0.2256	0.1995	0.1534	0.1421	MAE
		0.2705	0.2502	0.1864	0.1828	RMSE

Table 4.10: Joy transfer learning between word and tweet domains using BERT and Edinburgh (Edin) representations. Metrics shown are Pearson Correlation (COR), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE)

4.6 Emotion Discussion

It is important to note that the metrics are different in these tasks than in previous sections. While the previous experiment's classifier sought to maximize all metrics, a better predictive model in this exercise maximizes the correlation while minimizing the MAE and RMSE. Interestingly, within Edinburgh embeddings results, some emotions have better transfer performance than others. When source and target domains are the same, correlations for *joy* are as high as 0.5966 and 0.5708 against values of 0.4626 and 0.4838 obtained for *anger*. This may be caused by Edinburgh embeddings not being able to distinguish *anger* with the same accuracy as *joy* from the embedding space. Another unexpected result occurs when transferring from tweet to words using Edinburgh embeddings. Somehow, these embeddings fail at transferring *anger*, obtaining a correlation of 0.0938, opposed to 0.4001 obtained at *joy*. This may be caused by *anger* having a more complex set of tweets and words.

In relation to emotion experiments using BERT, the worst results are obtained when transferring from domains with shorter lexical units than those of the target domain, similarly to what is reported in the sentiment experiments. This is expected, as BERT excels at using attention mechanisms and extracting contextual information from larger contexts. This model performs well in general, except when transferring *anger* from tweets to words. This emotion seems to be difficult to capture for both representation models.

If we compare both representation approaches, we can conclude that each approach obtains similar results to those from the sentiment experiments, with BERT extracting better knowledge from domains with longer lexical units to shorter ones. BERT also dominates the diagonal, so it achieves better predictions within the same domains. Meanwhile, Edinburgh embeddings has a slightly better performance than BERT when transferring from words to tweets. This behavior seems to be repeated in the other emotions: BERT dominating all transfer learning experiments excepting the word to tweet task, in which the Edinburgh embeddings present competitive results.

These correlations are in line with previous results obtained in [37]. We must remark that we are using BERT's base model and a fine-tuned version could get even higher correlation results, probably dominating every emotion transfer learning task.

4.7 Qualitative Analysis

In this section we will discuss the various reasons why the classifier performed poorly in some transfer learning tasks.

The first possible reason is that each dataset has a different set of human annotators, so the label assigned to the different lexical units may be different, susceptible to subjectivity. These annotators also had done their work at different time spans, so word intensity and meanings might be different [34]. As an example, the word "terrorism" could be more intense at tweets closer to terrorist attacks.

Another feasible reason can be inferred by observing the wrong predictions of the classifier. At sentiment classification, both models failed at the following 4 tweets of the first 20.

Actual	Predicted	Tweet
positive	negative	Gas by my house hit $3.39!!!! I'm going to Chapel Hill on Sat. :)
positive	negative	with J Davlar 11th. Main rivals are team Poland. Hopefully we an make it a successful end to a tough week of training tomorrow.
positive	negative	Never start working on your dreams and goals tomorrow..... tomorrow never comes....if it means anything to U, ACT NOW! #getafterit
positive	negative	Looks like Andy the Android may have had a little too much fun yesterday. http://t.co/7ZDEfzEC

Table 4.11: Examples of classification errors of both models in tweet prediction, trained on words.

These 4 positive tweets contain some words with strong negative sentiment, such as tough, rivals, and never. We can infer that the presence of specific keywords could generate confusing representations for the classifier.

Finally, at the emotion classification task, we note that anger was difficult to classify from tweets to words, so we extracted the 10 worst predictions from both models, which we analyze below.

Actual	Predicted	Error	Word
0.818	0.293	-0.525	terrorist
0.562	0.037	-0.525	torpedo
0.939	0.41	-0.529	terrorize
0.844	0.308	-0.536	tumultuous
0.621	0.084	-0.537	theft
0.859	0.313	-0.546	tirade
0.851	0.295	-0.556	terrorism
0.825	0.26	-0.565	ruinous
0.544	-0.028	-0.572	robbery
0.862	0.227	-0.635	smite

Table 4.12: Examples of classification errors of BERT AVG representation at word prediction, trained on tweets.

Actual	Predicted	Error	Word
0.125	0.653	0.528	wireless
0.844	0.315	-0.529	tumultuous
0.781	0.251	-0.53	sinister
0.885	0.353	-0.532	wrath
0.03	0.57	0.54	waffle
0.219	0.765	0.546	unhelpful
0.152	0.715	0.563	underpaid
0.814	0.235	-0.579	savage
0.219	0.821	0.602	whiny
0.328	0.95	0.622	spammers

Table 4.13: Examples of classification errors of Word2Vec Edinburgh representation at word prediction, trained on tweets.

In BERT's case, we noticed that 3 related words were misclassified: "terrorist", "terrorize", and "terrorism". When examining the training set, we noticed that none of these words appear in any tweet, so the error may be due to a lack of training data. This lack of data may mean that no tweet has a close representation of these words and the closest ones have a lower intensity than expected.

In Edinburgh's case, we didn't notice such a noticeable pattern as BERT's. So we took a random word and analyzed the training tweets that contained it. The following tweets contain mentions of "unhelpful".

Training tweets	Anger
@ThomsonCares Sam- yes we have! Not helpful at all! We need this sorting ASAP! You keep promising stuff that doesnt happen!!!! #fuming	0.771
@lynnew69 then he said talking about wills uncontrollable animals when moving to another link. These comments do not help! #fuming	0.75
Zero help from @ups customer service. Just pushing the buck back and forth and promising callbacks that don't happen. #anger #loathing	0.854

Table 4.14: Tweets with the presence of words related to "unhelpful"

With this we observe that the predicted value of unhelpful is 0.765, while the value delivered by the human annotators is 0.219. Tweets referring to unhelpful have an intensity greater than 0.7, hence the presence of words within a more intense context such as anger may interfere with the predictions.

Chapter 5

Conclusions and Further Research

This thesis has presented a novel method for leveraging affect knowledge between three different domains: movie reviews, tweets, and words. Our methods exploit the fact that despite the apparent differences between these domains, the sentiment and emotion label space is shared across them and that in many cases, training data is not available for the target domain in which affect intends to be analyzed.

Our method's rationale is that both static and contextual word embeddings can be aggregated to represent textual units of different lengths, such as movie reviews, tweets, and words as compatible vectors (i.e., textual elements from different domains reside in the same feature space). Consequently, a classifier trained with data from one source domain can easily be applied to data from a different target domain.

Our results indicate that, in general, affective knowledge can be transferred between one domain to another using our method. However, classification performance can vary significantly depending on the choice of source-target domain pair and the representation method. Word2Vec tends to produce more stable results than BERT and performs relatively well in many transfer learning tasks. Word2Vec vectors trained on Twitter data work significantly better than General purpose embeddings for tweets' sentiment classification. Concerning BERT, we observe that BERT-derived representations can outperform WordVec in many tasks. However, these results exhibit more variability depending on the aggregation approach used.

Another remarkable result is that, in many cases, the transfer classification results show high scores for the area under the ROC curve (AUC) metric and low scores for F1 and Kappa. This anomaly suggests that the decision boundary gets shifted when moving from one sentiment domain to another. This problem could be mitigated by adjusting the decision threshold on the target domain.

The emotion experiments results suggest that *anger* classification is more challenging than *joy*, *sadness*, and *fear* on transfer learning tasks between word and tweet domains. Both embeddings paradigms have their performance notoriously decreased even when training and testing over the same domain. This decretion could be caused by hard to decipher expressions

such as irony and negation present in *anger* emotion.

The qualitative analysis on the different datasets allowed us to identify which are the main reasons for the classification errors on the representations obtained by both models. These errors could be mitigated by using the same human annotators and larger datasets, but the main objective of this thesis is to transfer affective knowledge when training data is scarce.

Finally, as a general trend, we observe that affective knowledge can be easier transferred from longer to shorter domains (e.g., movie reviews to tweets or tweets to words) than the opposite way. We attribute this to the fact that the training domain is richer in contextual sentiment information in those cases. The main contribution of this thesis is a new method to leverage affective labels between diverse domains. This approach can be especially useful for practitioners who lack the resources for creating annotated data for their target domain.

We envision several avenues of future work. First, we plan to explore our method with other affect labels, such as the Hourglass of Emotions[12] and hate speech. Second, we will study how to incorporate other recently developed contextualized models such as XLNet [60], RoBERTa [30], and ERNIE [63] into our method. Finally, we plan to conduct qualitative analysis to understand the limitations of our approach. We plan to determine which types of emotion patterns (e.g., negation, irony) are not captured by our method.

Bibliography

[1] M. S. Akhtar, A. Ekbal, and E. Cambria. How intense are you? predicting intensities of emotions and sentiments using stacked ensemble [application notes]. *IEEE Computational Intelligence Magazine*, 15(1):64–75, 2020.

[2] Silvio Amir, Ramón Astudillo, Wang Ling, Bruno Martins, Mario J Silva, and Isabel Trancoso. Inesc-id: A regression model for large scale twitter sentiment lexicon induction. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 613–618, 2015.

[3] Salima Behdenna, Fatiha Barigou, and Ghalem Belalem. Sentiment analysis at document level. In Aynur Unal, Malaya Nayak, Durgesh Kumar Mishra, Dharm Singh, and Amit Joshi, editors, *Smart Trends in Information Technology and Computer Communications*, pages 159–168, Singapore, 2016. Springer Singapore.

[4] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538, 2012.

[5] Felipe Bravo-Marquez, Eibe Frank, Saif M. Mohammad, and Bernhard Pfahringer. Determining word-emotion associations from tweets by multi-label classification. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2016, Omaha, NE, USA, October 13-16, 2016*, pages 536–539. IEEE Computer Society, 2016.

[6] Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. From unlabelled tweets to twitter-specific opinion words. In Ricardo Baeza-Yates, Mounia Lalmas, Alistair Moffat, and Berthier A. Ribeiro-Neto, editors, *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, pages 743–746. ACM, 2015.

[7] Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. Transferring sentiment knowledge between words and tweets. *Web Intelligence*, 16(4):203–220, 2018.

[8] José Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *J. Artif. Intell. Res.*, 63:743–788, 2018.

[9] Erik Cambria. Affective computing and sentiment analysis. *IEEE intelligent systems*, 31(2):102–107, 2016.

[10] Erik Cambria and Amir Hussain. Sentic computing. *Cognitive Computation*, 7(2):183–185, 2015.

[11] Erik Cambria, Yang Li, Frank Z Xing, Soujanya Poria, and Kenneth Kwok. Senticnet 6: Ensemble application of symbolic and subsymbolic ai for sentiment analysis. *CIKM'20, Oct 20-24*, 2020.

[12] Erik Cambria, Andrew Livingstone, and Amir Hussain. The hourglass of emotions. In *Proceedings of the 2011 International Conference on Cognitive Behavioural Systems*, COST'11, page 144–157, Berlin, Heidelberg, 2011. Springer-Verlag.

[13] José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. Spanish pre-trained bert model and evaluation data. In *Practical ML for Developing Countries Workshop@ ICLR 2020*, 2020.

[14] Rafeeque Pandara Chalil, Sendhilkumar Selvaraju, and G. S. Mahalakshmi. Twitter sentiment analysis for large-scale data: An unsupervised approach. *Cogn. Comput.*, 7(2):254–262, 2015.

[15] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[16] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[18] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alexander J. Smola, and Vladimir Vapnik. Support vector regression machines. In Michael Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, pages 155–161. MIT Press, 1996.

[19] Andrea Esuli and Fabrizio Sebastiani. Determining the semantic orientation of terms through gloss classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 617–624. ACM, 2005.

[20] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June 2008.

[21] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale

sentiment classification: A deep learning approach. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 513–520. Omnipress, 2011.

[22] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[23] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[24] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. Exploiting document knowledge for aspect-level sentiment classification. *CoRR*, abs/1806.04346, 2018.

[25] B. Joyce and J. Deng. Sentiment analysis of tweets for the 2016 us presidential election. In *2017 IEEE MIT Undergraduate Research Technology Conference (URTC)*, pages 1–4, 2017.

[26] Morgan Kaufmann. *Data mining practical machine learning tools and techniques*. Elsevier, 2005.

[27] Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL, 2014.

[28] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[29] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[30] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[31] Yukun Ma, Haiyun Peng, Tahir Khan, Erik Cambria, and Amir Hussain. Sentic LSTM: a hybrid network for targeted aspect-based sentiment analysis. *Cogn. Comput.*, 10(4):639–650, 2018.

[32] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics, 2011.

[33] Prem Melville, Wojciech Gryc, and Richard D. Lawrence. Sentiment analysis of blogs

by combining lexical knowledge with text classification. In John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki, editors, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 1275–1284. ACM, 2009.

[34] Rada Mihalcea and Vivi Nastase. Word epoch disambiguation: Finding how words change over time. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*, pages 259–263. The Association for Computer Linguistics, 2012.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013.

[36] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.

[37] Saif Mohammad and Felipe Bravo-Marquez. WASSA-2017 shared task on emotion intensity. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 34–49, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[38] Saif Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon. *Comput. Intell.*, 29(3):436–465, 2013.

[39] Saif M. Mohammad. Word affect intensities. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018)*, Miyazaki, Japan, 2018.

[40] Huy Nguyen and Minh-Le Nguyen. A deep neural architecture for sentence-level sentiment classification in twitter social networking. In Kôiti Hasida and Win Pa Pa, editors, *Computational Linguistics - 15th International Conference of the Pacific Association for Computational Linguistics, PACLING 2017, Yangon, Myanmar, August 16-18, 2017, Revised Selected Papers*, volume 781 of *Communications in Computer and Information Science*, pages 15–27. Springer, 2017.

[41] Finn Årup Nielsen. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In Matthew Rowe, Milan Stankovic, Aba-Sah Dadzie, and Mariann Hardey, editors, *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages, Heraklion, Crete, Greece, May 30, 2011*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98. CEUR-WS.org, 2011.

[42] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[43] Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1):11–24,

1994.

[44] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

[45] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018.

[46] Saša Petrović, Miles Osborne, and Victor Lavrenko. The Edinburgh twitter corpus. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, pages 25–26, Los Angeles, California, USA, June 2010. Association for Computational Linguistics.

[47] Soujanya Poria, Erik Cambria, Grégoire Winterstein, and Guang-Bin Huang. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowl. Based Syst.*, 69:45–63, 2014.

[48] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[49] Sujata Rani and Parteek Kumar. Deep learning based sentiment analysis using convolution neural network. *Arabian Journal for Science and Engineering*, 44(4):3305–3314, 2019.

[50] Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. Semeval-2014 task 9: Sentiment analysis in twitter. In Preslav Nakov and Torsten Zesch, editors, *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 73–80. The Association for Computer Linguistics, 2014.

[51] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.

[52] Vikas Sindhwani and Prem Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pages 1025–1030. IEEE Computer Society, 2008.

[53] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL, 2013.

[54] Suhariyanto, A. Firmanto, and R. Sarno. Prediction of movie sentiment based on reviews and score on rotten tomatoes using sentiwordnet. In *2018 International Seminar on Application for Technology of Information and Communication*, pages 202–206, 2018.

[55] Trieu H. Trinh, Andrew M. Dai, Thang Luong, and Quoc V. Le. Learning longer-term dependencies in rnns with auxiliary losses. *CoRR*, abs/1803.00144, 2018.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[57] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*, pages 347–354. The Association for Computational Linguistics, 2005.

[58] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. *CoRR*, abs/2001.08317, 2020.

[59] Yunqing Xia, Erik Cambria, Amir Hussain, and Huan Zhao. Word polarity disambiguation using bayesian model and opinion-level features. *Cognitive Computation*, 7, 06 2014.

[60] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764, 2019.

[61] Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. Refining word embeddings for sentiment analysis. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 534–539. Association for Computational Linguistics, 2017.

[62] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statis-

tical framework. *Int. J. Machine Learning & Cybernetics*, 1(1-4):43–52, 2010.

[63] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy, July 2019. Association for Computational Linguistics.

[64] Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724, 2015.

[65] Will Y. Zou, Richard Socher, Daniel M. Cer, and Christopher D. Manning. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1393–1398. ACL, 2013.