UNIVERSITY OF CAMPINAS

INSTITUTE OF COMPUTING

Master's Qualifying Exam

SENTIMENT CLASSIFICATION OF SHORT TEXTS USING ARTIFICIAL NEURAL NETWORKS

Candidate: Tiago Martinho de Barros

Advisor: Prof. Dr. Hélio Pedrini

Co-Advisor: Prof. Dr. Zanoni Dias

October 2019

Contents

Abstract

Nowadays, user-generated content is common and abundant on the Internet. It also has great value because it enables the understanding of people's opinions and views about different matters. In recent elections, for instance, we saw the power that this kind of content has. A classic example of Sentiment Analysis is a company trying to gauge customer satisfaction of their products and services. Ideally, we want to have as much data as possible, however, to process such amount of data, it is impractical to do it manually, so we have computational solutions to process user-generated texts and output their sentiment. However, to be able to do that effectively, computers must understand natural language, which is a long-standing problem in Artificial Intelligence. Fortunately, important advances on this front have recently been made in the form of unsupervised language representation learning methods based on Artificial Neural Networks, that have been showing promising results in many Natural Language Processing tasks. This work aims to investigate such methods and develop a new methodology to tackle the problem of Sentiment Analysis of short texts. This methodology will be evaluated on public datasets and the results will be compared against state-of-the-art methods.

List of Abbreviations

BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
CR	Customer Reviews
CV	Cross-Validation
ELMo	Embeddings from Language Models
GPT	Generative Pre-trained Transformer
LSTM	Long Short-Term Memory
MPQA	Multi-Perspective Question Answering
MR	Movie Reviews
MultiNLI	Multi-genre Natural Language Inference
NLP	Natural Language Processing
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SNS	Social Networking Service
SST	Stanford Sentiment Treebank
STILTs	Supplementary Training on Intermediate Labeled-data Tasks
SVM	Support Vector Machines
ULMFiT	Universal Language Model Fine-Tuning

Chapter 1

Introduction

With the widespread access to the Internet in the last decades, people have gained a new and powerful medium through which make their opinion visible worldwide. Entities, such as companies and government agencies, are frequently interested in knowing what people think in order to make informed decisions.

Usual sources of opinionated texts are Social Networking Services (SNS) and (micro)blogs. Using data from these sources, it is possible to know the opinion of end users about a product or service [29, 40], evaluate the helpfulness of online reviews [8], and build a recommender system [42], to list some examples.

Since early 2000s, Sentiment Analysis has grown to be one of the most active research areas in Natural Language Processing (NLP). It is also widely studied in Data Mining, Web Mining, Text Mining, and Information Retrieval. In fact, it has spread from Computer Science to Management Sciences and Social Sciences such as Marketing, Finance, Political Science, Communications, Health Science, and even History, due to its importance to business and society as a whole [77].

1.1 Problem Description

Sentiment Analysis, also referred to as Opinion Mining [34, 43], is a research field concerned with the computational manipulation of sentiments. In this context, "sentiment" can be defined as the author's attitude, opinion, or emotion expressed on a named entity, event, or abstract concept that is mentioned in a piece of text in natural language [62].

Traditionally, this problem is cast as a classification task, either binary, with a "positive" class and a "negative" class, or as a multi-class classification problem, including, for instance, a "neutral" or "irrelevant" class to accommodate texts that do not have a sentiment associated with (i.e., objective texts). If the method used to solve this problem does not perform automatic feature extraction, this step must be carried out before the classification process can take place. It is possible to use the frequency of occurrence of words or n-grams as weights to show their relative importance, or use binary weights (indicating if a word occurs in a text or not) [37]. Another common practice is to utilize part of speech information, for instance, searching for adjectives, since they often are related to sentiment sentences.

More recently, unsupervised language representation learning methods have become more popular and have been producing some of the best results so far in many NLP tasks. The idea is to produce a one-size-fits-all model that is pre-trained on large amounts of unlabeled data, and then fine-tuned on an individual target task (downstream task), in this case Sentiment Analysis. Pre-training on large amounts of data aims to produce sentence encoders with substantial knowledge of the target language, that can be applied later on the target task. Examples

of this approach are Universal Language Model Fine-Tuning (ULMFiT) [24], Embeddings from Language Models (ELMo) [52], Generative Pre-trained Transformer (GPT) [56], Bidirectional Encoder Representations from Transformers (BERT) [15], and XLNet [75].

There is also the approach to add a supplementary supervised training step between the unsupervised pre-training and the fine-tuning on the target task. One such method is called Supplementary Training on Intermediate Labeled-data Tasks (STILTs) [53], which was applied on BERT, GPT, and ELMo, and improved the performance in some cases, especially when only a small amount of training data is available for the target task. The authors used four intermediate tasks: the Multi-genre Natural Language Inference (MultiNLI) corpus [69], the Stanford Natural Language Inference (SNLI) corpus [5], the Quora Question Pairs (QQP) dataset[1], and a custom fake-sentence-detection task based on the BookCorpus dataset [81]. But it is not clear at this time which intermediate task will produce more improvement (if any) to the performance on the target task.

1.2 Research Questions

This work aims to investigate and answer the following research questions:

- Can unsupervised language representation learning methods produce competitive results when applied to sentiment classification of short texts?
- Can further unsupervised pre-training on in-domain data improve the sentiment classification performance?
- Can supplementary supervised training improve the sentiment classification performance? If it can, how to select the intermediate task that will give the best performance on the target task?
- How much influence the preprocessing of the input text has on the classification performance? And what types of preprocessing tasks give the best results?

1.3 Objectives and Contributions

This work seeks to research and study state-of-the-art Sentiment Analysis techniques, and propose a competitive method to address the problem.

In order to achieve our general objective, some specific objectives have to be satisfied:

- Study of recent works on the subject;
- Dataset preparation;
- Reproduction of the baseline results;
- Extension of the baseline methods based on approaches that achieved successful results on related problems;
- Proposal of an original methodology using neural networks;
- Performance evaluation of the developed model;
- Publication of the results.

[1] `https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs`

This project aims to propose, create, and evaluate a new methodology for Sentiment Analysis of texts based on unsupervised language representation learning methods that produces competitive results regarding efficacy.

1.4 Text Structure

This text is organized into four chapters. In Chapter 1, we presented an overview of the research problem to be addressed in this work, as well as the main goals and contributions expected. In Chapter 2, we introduce the main concepts and briefly describe relevant work available in the literature done in the field. In Chapter 3, we present our proposed methodology, evaluation metrics, and datasets to be used in our experiments. Finally, in Chapter 4, we outline our work plan and activity schedule.

Chapter 2

Background

In this chapter, we present basic concepts related to Sentiment Analysis and to the relevant methods used to solve this problem. Then, we provide a general explanation of some available methods to tackle the problem, which represent baselines and inspiration for the method we are proposing.

2.1 Concepts and Techniques

In this section, we present some relevant concepts and techniques related to Sentiment Analysis, intending to help the comprehension of the problem under investigation in this work.

2.1.1 Sentiment Analysis Tasks

The Sentiment Analysis problem is generally tackled at three possible levels of granularity: document-level sentiment classification, sentence-level sentiment classification, and aspect-level sentiment classification [77]. Document-level sentiment classification categorizes a text as expressing an overall positive or negative opinion. It treats the entire text document as the basic unit of process and considers that the document contain opinions about a single entity (e.g., a movie review). Sentence-level sentiment classification categorizes individual sentences in a document. In this case, it is not reasonable to assume that every sentence contains an opinion. A traditional approach is to first classify a sentence as opinionated/not opinionated, which is called subjectivity classification. Then, the resulting opinionated sentences are classified as expressing positive or negative opinions. Another possibility is to include a third class to accommodate non-opinionated sentences (e.g., a class named "neutral"). Aspect-level sentiment classification is concerned with the extraction of people's opinions expressed on entities and aspects/features of entities, which are also called targets. For example, in the sentence *"the art direction of 'Star Wars: The Force Awakens' was amazing, but the plot was uninteresting, to say the least"*, we have the entity *"Star Wars: The Force Awakens"* and the aspects *"art direction"* and *"plot"*. Aspect-level sentiment classification should classify the sentiment expressed on the art direction of the movie as positive and on the plot as negative.

Researchers are also working on other topics of Sentiment Analysis, such as Emotion Analysis [7], Cross-Domain Sentiment Classification [44], Sarcasm Detection [21], and Multilingual Sentiment Analysis [4].

2.1.2 Word Embedding

Many Deep Learning models in NLP need word embedding vectors as input features [11]. Word embedding is a technique for Language Modeling and Feature Learning, which transforms words in a vocabulary to vectors of continuous real numbers (distributional vectors, also called word vectors or word embeddings). This technique follows a hypothesis that words with similar meanings tend to occur in similar contexts. These vectors try to capture the characteristics of the neighbors of a word and the similarity between words. One method for computing this similarity is by using cosine similarity. Figure 2.1 shows an example of word embedding, where numbers were mapped to colors for better visualization.

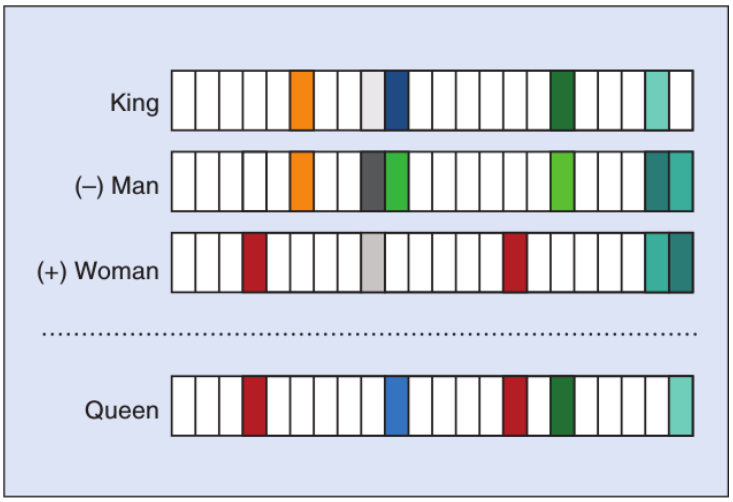

Figure 2.1: An example of word embedding [76].

The process typically involves embedding from a high-dimensional sparse vector space (e.g., one-hot encoding vector space, in which each word takes a dimension) to a lower-dimensional dense vector space. Each dimension of the distributional vector represents a latent feature of a word. These vectors may encode linguistic regularities and patterns. Usually, word embeddings are pre-trained by optimizing an auxiliary objective in a large unlabeled corpus, such as predicting a word based on its context [39], where the learned word vectors can capture general syntactic and semantic information.

The learning of word embeddings can be done using neural networks or matrix factorization. One popular word embedding system is word2vec [39], which is a computationally efficient neural network prediction model that learns word embeddings from text. Another frequently used learning approach is GloVe [51], which is trained on the nonzero entries of a global word-word co-occurrence matrix.

2.1.3 Recurrent Neural Network

In late 1990s, the research community lost interest in neural networks, mainly because they were regarded as only practical for "shallow" neural networks (neural networks with one or two layers), as training a "deep" neural network (neural networks with more layers) was complicated and computationally very expensive [77]. However, in the past 10 years, thanks to the increasing computing power, the availability of huge amounts of training data, and the power and flexibility of learning intermediate representations [3], there has been a resurgence of interest and research in neural networks. They have been employed in many tasks, such as Speech

Recognition [22], Machine Translation [1, 58], and Part of Speech Tagging [11], in general with great success.

One type of neural network that is of great interest for us is the Recurrent Neural Network (RNN) [18]. An RNN is an extension of a conventional feedforward neural network which is able to handle a variable-length sequence input. The connections between neurons in an RNN form a directed cycle, and, unlike feedforward neural networks, RNN can use its internal "memory" to process a sequence of inputs, which makes it particularly suitable for processing sequential information, such as text and audio.

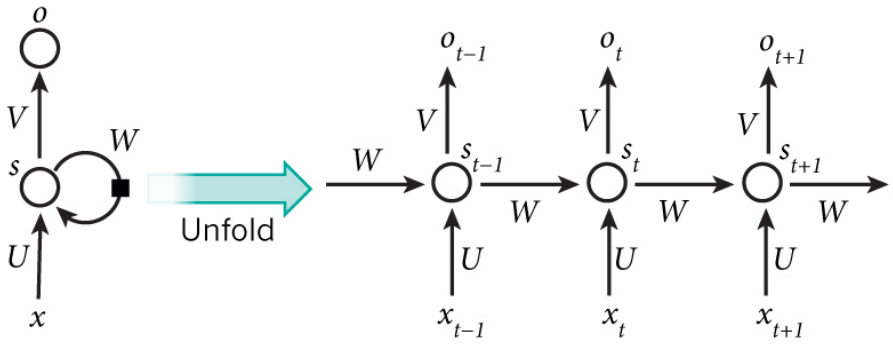

Figure 2.2: An example of a Recurrent Neural Network [33].

RNNs process an input sequence one element at a time, maintaining in their hidden units a "state vector" that implicitly contains information about the history of all the past elements of the sequence. The outputs of the hidden units at different discrete time steps are equivalent to the outputs of different neurons in a deep multi-layer network, in respect of the backpropagation application to train RNNs.

Figure 2.2 illustrates an example of an RNN. The graph in the left-hand side is a folded network with cycles, while the graph in the right-hand side is an unfolded sequence network with three time steps. The number of time steps is determined by the length of input. In Figure 2.2, x_t is the input vector at time step t, s_t is the hidden state at time step t, which is calculated based on the input at the current time step and the previous hidden state, W is the weight matrix used to condition the previous hidden state s_{t-1}, and U is the weight matrix used to condition the input x_t. Given a sequence $x = (x_1, x_2, \cdots, x_T)$, the RNN updates its recurrent hidden state s_t by

$$s_t = \begin{cases} 0, & t = 0 \\ \phi(x_t, s_{t-1}), & \text{otherwise} \end{cases} \tag{2.1}$$

where ϕ is a nonlinear function such as the composition of a logistic sigmoid with an affine transformation [10].

Traditionally, the update of the recurrent hidden state in Equation 2.1 is implemented as

$$s_t = f(Ux_t + Ws_{t-1} + b) \tag{2.2}$$

where f is a smooth, bounded function such as logistic sigmoid function or hyperbolic tangent function (although the utilization of the rectifier is also possible [31]), and b is the bias to be applied.

Term o_t is the output probability distribution over the vocabulary at time step t.

$$o_t = softmax(Vs_t) \tag{2.3}$$

The hidden state s_t is regarded as the memory of the network. It captures information about what happened in all previous time steps. o_t is calculated solely based on the memory s_t at time step t and the corresponding weight matrix V.

The same parameters (matrices U, W, and V) are used at each time step, unlike a feedforward neural network. This means that it performs the same task at each step, just with different inputs. This greatly reduces the total number of parameters needed to learn.

2.1.4 Long Short-Term Memory Network

RNNs, as seen previously, seem to be a wise choice for processing sequential information. Theoretically, it can make use of the information in arbitrarily long sequences, but unfortunately, it has been observed that it is difficult to train RNNs to capture long-term dependencies because of two gradient-related problems: they tend to either vanish (most common case) or explode (rare case, but with severe consequences) [2]. There have been two main approaches by which many researchers have tried to reduce the negative impacts of this issue: devise a better learning algorithm than a simple stochastic gradient descent or design a more sophisticated activation function. An example of the first is the *clipped gradient* [50], by which the norm of the gradient vector is clipped, or using second-order methods, which may be less sensitive to the issue if the second derivatives follow the same growth pattern as the first derivatives (which is not guaranteed to be the case).

About the other approach, the most famous attempt in this direction resulted in an activation function (or a recurrent unit) called Long Short-Term Memory (LSTM) unit [23]. An LSTM network is a special type of RNN which is capable of learning long-term dependencies.

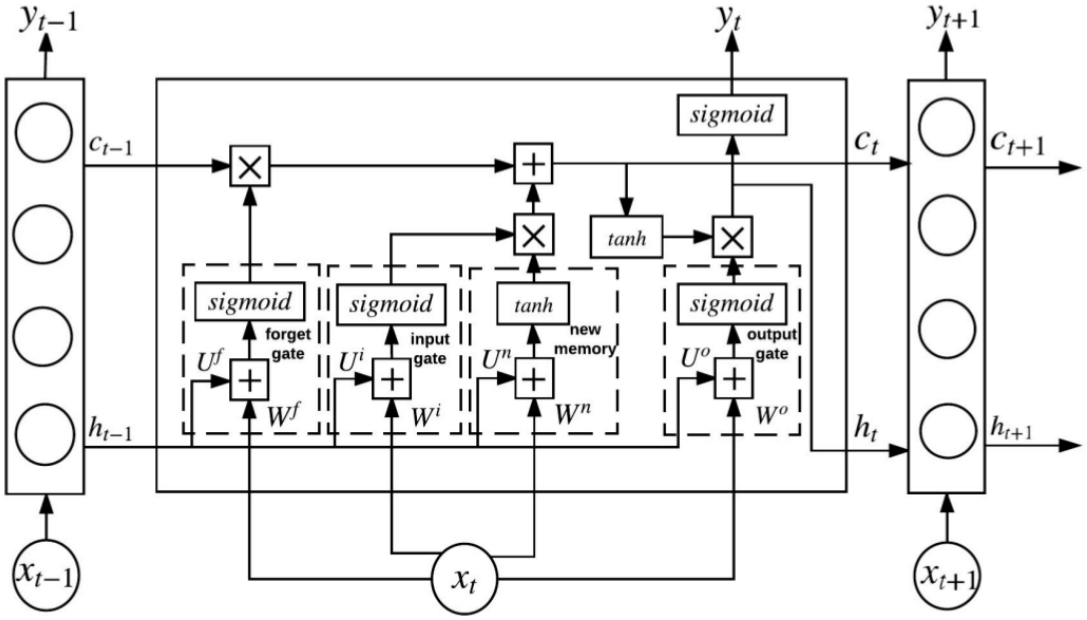

Figure 2.3: An example of a Long Short-Term Memory Network [77].

Figure 2.3 illustrates an example of an LSTM network. It has two states (hidden state h_t and cell state c_t) and three gates (input gate i, output gate o, and forget gate f [19]), which

are sigmoid functions. At time step t, LSTM first decides what information to dump from the cell state. This decision is made by the forget gate. It takes as inputs x_t (current input) and h_{t-1} (output from the previous hidden layer), and outputs a number in the interval $[0, 1]$, with 1 meaning "completely keep" and 0, "completely dump":

$$f_t = \sigma(W^f x_t + U^f h_{t-1} + b^f) \tag{2.4}$$

Then, LSTM decides what new information to store in the cell state. This has two steps: first, the input gate decides which values LSTM will update (Equation 2.5); next, a hyperbolic tangent function creates a vector of new candidate values $\tilde{c}_t$ (Equation 2.6), which will be added to the cell state.

$$i_t = \sigma(W^i x_t + U^i h_{t-1} + b^i) \tag{2.5}$$

$$\tilde{c}_t = \tanh(W^n x_t + U^n h_{t-1} + b^n) \tag{2.6}$$

LSTM then combines these two values to update the cell state:

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \tag{2.7}$$

where $\circ$ is the Hadamard product (element-wise product).

Finally, LSTM decides the output, which is based on the cell state. First, the output gate decides which parts of the cell state to output:

$$o_t = \sigma(W^o x_t + U^o h_{t-1} + b^o) \tag{2.8}$$

Then, LSTM puts the cell state through a g function and multiplies it by the output of the output gate, so that LSTM can decide which parts it wants to output:

$$h_t = o_t \circ g(c_t) \tag{2.9}$$

where g can be either the hyperbolic tangent function or the identity.

2.1.5 Convolutional Neural Network

Convolutional Neural Network (CNN) is a type of neural network originally employed in the field of Computer Vision and mostly applied to images. Its design is inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. The visual cortex contains cells that are responsible for detecting light in small and overlapping sub-regions of the visual field, known as *receptive fields*. These cells act as local filters over the input space. Typically, a CNN consists of multiple convolutional layers, each of which performs a function analogous to that of the cells in the visual cortex.

Figure 2.4 illustrates an example of a CNN. It shows a CNN architecture used for recognizing hand-written digits using the MNIST dataset [32]. The input is a 28 $\times$ 28 $\times$ 1 pixel image (width $\times$ height $\times$ number of channels). In this first stage, the filter (size 5 $\times$ 5 $\times$ 1) is used to scan the image. The filter is also called *kernel*. Each region in the input image that the filter projects on is a receptive field. The filter is actually an array of numbers (called weights or parameters). As the filter is sliding (or *convolving*), it is multiplying its weight values by the original pixel values of the image (Hadamard product). The multiplications are all summed up to a single number, which is a representative of the receptive field. A number is produced for

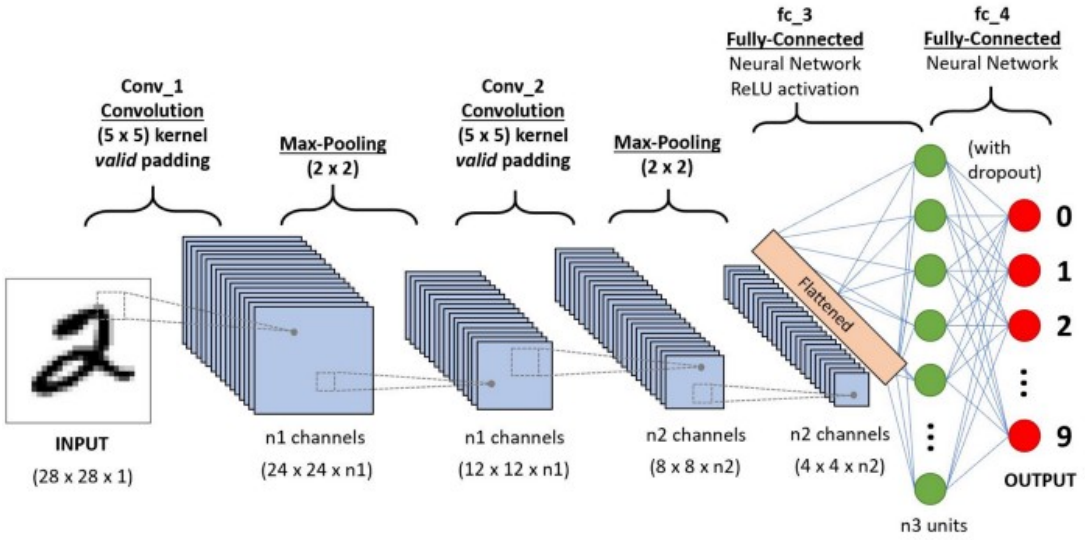

Figure 2.4: An example of a Convolutional Neural Network [57].

every receptive field. After the filter finishes scanning over the image, we get an array (size 24 $\times$ 24 $\times$ 1), which is called *Activation Map* or *Feature Map*. In most cases, different filters are used to scan the input. In Figure 2.4, we apply n_1 kinds of filters and thus have n_1 stacked Feature Maps in the first *convolutional layer*. Following the convolutional layer, a *pooling layer* (sub-sampling) is usually used to progressively reduce the spatial size of the representation, thus reducing the number of features and the computational complexity of the network. For example, after pooling in the first stage, the dimensions of the Feature Maps are reduced to 12 $\times$ 12 $\times$ n_1. While the dimensionality of each Feature Map is reduced, the pooling step retains the most important information. A commonly used sub-sampling operation is the Max-Pooling, with another possibility being the Average-Pooling. Afterwards, the output from the first stage becomes the input to the second stage and new filters are employed. Their sizes are 5 $\times$ 5 $\times$ n_1, where n_1 is the size of the Feature Map of the last layer. After the second stage, the CNN uses a *Fully-Connected* layer and, since this example illustrates a multi-class classification problem, a *softmax* readout layer with the probabilities that the input image represents the digit $i, \forall i \in \mathbb{N} \mid 0 \leq i \leq 9$.

Convolutional layers in CNNs play the role of a feature extractor, which extracts local features as they restrict the receptive fields of the hidden layers to be local. It means that CNNs have a special spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. Such a characteristic is useful for classification in NLP, in which we expect to find strong local clues regarding class membership, but these clues can appear in different places in the input. For example, in a document classification task, a single key phrase (or an *n-gram*) can help in determining the topic of the document. We would like to learn those sequences of words that are good indicators of the topic, not necessarily caring where they appear in the document. Convolutional and pooling layers allow a CNN to learn to find such local indicators, regardless of their positions [20].

2.2 Related Work

Until the first years of this millennium, texts were traditionally classified by topic. However, in 2002, two seminal papers were published, that put Sentiment Analysis on the map of NLP research fields. The first one is the work of Pang et al. [48], in which they apply Machine Learning methods (Naïve Bayes, Maximum Entropy Classification, and Support Vector Machines (SVM)) to perform sentiment classification of movie reviews. The other paper is by Turney [63], in which he presents an unsupervised learning algorithm for classifying customer reviews as recommended or not recommended using the average semantic orientation of the phrases in the review that contain adjectives or adverbs.

Turney and Littman [64] introduced, in 2003, a method for inferring the semantic orientation of a word (how much positive or negative) from its statistical association with a set of positive and negative paradigm words. The authors evaluated two methods: Pointwise Mutual Information and Latent Semantic Analysis.

In 2004, Hu and Liu [25] mined and summarized the features of products on which customers have expressed their opinions and whether the opinions were positive or negative. This publication produced a dataset that is used by many works in the field. In the same year, Pang and Lee [45] proposed a novel Machine Learning method that applies text categorization techniques to just the subjective portions of the document. They presented a sentence-level graph-based formulation relying on finding minimum cuts to decide if a sentence is subjective (contains an opinion) or objective (does not contain an opinion), then, considering only the subjective sentences, a standard Machine Learning classifier is used to determine the document polarity (positive or negative).

Pang and Lee [46] considered, in 2005, the problem of classifying the sentiment of texts with respect to a multi-point scale (e.g., one to five "stars"). They also released a dataset of movie reviews that proved to be very popular amongst researchers.

In 2008, Pang and Lee [47] published a comprehensive and influential survey on Sentiment Analysis, presenting a panorama of the field at the time, covering new challenges, promising approaches, issues regarding privacy, manipulation, and economic impact of the development of opinion-oriented information-access services, and providing a discussion of available resources, benchmark datasets, and evaluation campaigns.

In 2011, Taboada et al. [59] introduced the Semantic Orientation CALculator (SO-CAL): a lexicon-based approach to extracting sentiment from text. It used sentiment dictionaries with annotations of polarity and strength of semantic orientation. The authors also described the process of dictionary creation.

Liu [34] also published an important survey, in 2012. Liu discusses the different formulations (e.g., cross-domain sentiment classification and aspect-based sentiment analysis) and approaches (e.g., dictionary-based approach and corpus-based approach) for Sentiment Analysis, as well as the problems that usually arise. Some other, related tasks are also discussed, for instance, opinion spam detection.

In the realm of unsupervised learning, Mikolov et al. [38] introduced, in 2013, the Skip-Gram model to generate word vectors by training a neural network to predict words that usually occur nearby a given word. In 2014, Le and Mikolov [30] proposed the Paragraph Vector method, that applies the idea of word embedding to variable-length pieces of text, ranging from sentences to whole documents. Kiros et al. [28] abstracted the Skip-Gram model to the sentence level. Instead of using a word to predict its surrounding context, the authors encoded a sentence to predict the sentences around it. The resulting sentence encoder model, called Skip-Thoughts, was published in 2015.

Dai and Le [14] first proposed, in 2015, the supervised fine-tuning step after the unsupervised pre-training, such as predicting adjacent sentences. The basic idea is to use the parameters obtained from the pre-training as a starting point for the supervised training model. In 2018, Peters et al. [52] introduced Embeddings from Language Models (ELMo), which employs contextualized word embedding to produce different word vectors for the same word if the context/meaning is different, using bi-directional LSTMs trained on a *language modeling* objective. In the same year, Howard and Ruder [24] introduced the Universal Language Model Fine-Tuning (ULMFiT), building upon the pre-training-followed-by-fine-tuning concept and addressing issues of over-fitting and catastrophic forgetting. Radford et al. [56] proposed the Generative Pre-trained Transformer (GPT), that combines the unsupervised pre-training with Transformers [65], as opposed to LSTMs. Devlin et al. [15] introduced a method called Bidirectional Encoder Representations from Transformers (BERT), which also employs Transformers but has a different training objective: *masked language modeling*, in which words in a sentence are randomly erased and replaced with a special token ("masked") with some small probability. Then, a Transformer is used to generate a prediction for the masked word based on the unmasked words surrounding it, both to the left and right. Finally, in 2019, Yang et al. [75] proposed the XLNet, a generalized auto-regressive pre-training method that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order.

2.2.1 Summary of Results

In this section, we present a summary of the results of the relevant related work. The adopted evaluation metric is accuracy, as it is the most widely used metric in the field. We define accuracy as:

$$\text{Accuracy} = \frac{\sum_{i=1}^{I}(TP_i + TN_i)}{\sum_{i=1}^{I}(TP_i + FP_i + TN_i + FN_i)} \tag{2.10}$$

where TP_i is 1 if the i-th instance is positive and the predicted class is positive as well, otherwise it is 0; TN_i is 1 if the i-th instance is negative and the predicted class is negative as well, otherwise it is 0; FP_i is 1 if the i-th instance is negative and the predicted class is positive, otherwise it is 0; and FN_i is 1 if the i-th instance is positive and the predicted class is negative, otherwise it is 0. I is the number of instances.

We collected and organized results for four datasets: Movie Reviews (MR) (Table 2.1), Customer Reviews (CR) (Table 2.2), Multi-Perspective Question Answering (MPQA) Opinion Corpus (Table 2.3), and Yelp (Table 2.4). Those datasets are described in Section 3.3.

Tables 2.1 to 2.4 are organized according to the non-increasing order of the accuracy values. Column "Method" shows the approach utilized, according to the respective publication. Since most of these datasets do not provide an explicit split of train/validation/test sets, column "Split" shows how the datasets were split between these sets (percentage), in that order. "10-fold CV" means 10-fold cross-validation. Some methods were pre-trained using other data than these datasets; in these cases, there is a symbol in the "Split" column and a description of the data they were pre-trained on is available in the caption. The values reported in the "Split" column always refer to the dataset presented in the table.

Table 2.4 is a little more involved because there are different versions of the Yelp dataset. The "2013 - Version 1" is composed of 78,966 randomly selected user reviews from the 2013

Authors	Year	Method	Split (%)	Accuracy (%)
Radford et al. [55]	2017	Byte mLSTM	10-fold CV†	86.9
Wang et al. [66]	2017	WCCNN	80/0/20	83.8
Zhao et al. [80]	2015	AdaSent	10-fold CV	83.1
Du et al. [17]	2019	BGRU-Capsule	90/0/10	82.5
Zhang et al. [79]	2019	3W-CNN	10-fold CV	82.3
Yang et al. [73]	2019	Capsule-CNN	80/10/10	82.3
Qian et al. [54]	2016	LR-Bi-LSTM	81/9/10	82.1
Yang et al. [73]	2019	Capsule-LSTM	80/10/10	81.7
Kim [27]	2014	CNN-non-static	10-fold CV	81.5
Conneau et al. [13]	2017	Bi-LSTM-Max	5-fold CV‡	81.1
Chen et al. [9]	2019	NIM-CNN	80/10/10	80.1
Wang and Manning [67]	2012	NBSVM-bi	10-fold CV	79.4
Nakagawa et al. [41]	2010	Tree-CRF	10-fold CV	77.3

Table 2.1: Comparative summary for the MR dataset. † pre-trained on an Amazon product review dataset [36]. ‡ pre-trained on the Stanford Natural Language Inference (SNLI) [5] and Multi-Genre Natural Language Inference (MultiNLI) [69] corpora.

Authors	Year	Method	Split (%)	Accuracy (%)
Radford et al. [55]	2017	Byte mLSTM	10-fold CV†	91.4
Conneau et al. [13]	2017	Bi-LSTM-Max	5-fold CV‡	86.3
Zhao et al. [80]	2015	AdaSent	10-fold CV	86.3
Zhang et al. [79]	2019	3W-CNN	10-fold CV	85.8
Yang et al. [73]	2019	Capsule-CNN	80/10/10	85.1
Kim [27]	2014	CNN-multichannel	10-fold CV	85.0
Yang et al. [73]	2019	Capsule-LSTM	80/10/10	84.9
Wang and Manning [67]	2012	NBSVM-bi	10-fold CV	81.8
Nakagawa et al. [41]	2010	Tree-CRF	10-fold CV	81.4

Table 2.2: Comparative summary for the CR dataset. † pre-trained on an Amazon product review dataset [36]. ‡ pre-trained on the Stanford Natural Language Inference (SNLI) [5] and Multi-Genre Natural Language Inference (MultiNLI) [69] corpora.

Authors	Year	Method	Split (%)	Accuracy (%)
Zhao et al. [80]	2015	AdaSent	10-fold CV	93.3
Zhang et al. [79]	2019	3W-CNN	10-fold CV	90.3
Conneau et al. [13]	2017	Bi-LSTM-Max	5-fold CV‡	90.2
Kim [27]	2014	CNN-static	10-fold CV	89.6
Radford et al. [55]	2017	Byte mLSTM	10-fold CV†	88.5
Wang and Manning [67]	2012	SVM-bi	10-fold CV	86.7
Nakagawa et al. [41]	2010	Tree-CRF	10-fold CV	86.1

Table 2.3: Comparative summary for the MPQA dataset. † pre-trained on an Amazon product review dataset [36]. ‡ pre-trained on the Stanford Natural Language Inference (SNLI) [5] and Multi-Genre Natural Language Inference (MultiNLI) [69] corpora.

2013 - Version 1 [60] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Wu et al. [70]	2018	HUAPA	80/10/10	68.3
Xu et al. [72]	2016	B-CLSTM	80/10/10	59.8
Tang et al. [60]	2015	UPNN (full)	80/10/10	59.6
Xu et al. [72]	2016	CLSTM	80/10/10	59.4
Tang et al. [60]	2015	UPNN (no UP)	80/10/10	57.7
2013 - Version 2 [61] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Yang et al. [74]	2016	HN-ATT	80/10/10	68.2
Tang et al. [61]	2015	LSTM-GRNN	80/10/10	65.1
2014 - Version 1 [60] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Wu et al. [70]	2018	HUAPA	80/10/10	68.6
Xu et al. [72]	2016	B-CLSTM	80/10/10	61.9
Tang et al. [60]	2015	UPNN (full)	80/10/10	60.8
Xu et al. [72]	2016	CLSTM	80/10/10	59.2
Tang et al. [60]	2015	UPNN (no UP)	80/10/10	58.5
2014 - Version 2 [61] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Yang et al. [74]	2016	HN-ATT	80/10/10	70.5
Tang et al. [61]	2015	LSTM-GRNN	80/10/10	67.1
2015 - Version 1 [61] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Yang et al. [74]	2016	HN-ATT	80/10/10	71.0
Tang et al. [61]	2015	LSTM-GRNN	80/10/10	67.6
2015 - Version 2 [78] - 5 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Yang et al. [75]	2019	XLNet	93/0/7†	72.2
Xie et al. [71]	2019	BERT	93/0/7‡	70.7
Howard and Ruder [24]	2018	ULMFiT	93/0/7	70.0
Johnson and Zhang [26]	2017	DPCNN	93/0/7	69.4
Zhang et al. [78]	2015	CharCNN	93/0/7	62.1
2015 - Version 2 [78] - 2 Classes				
Authors	**Year**	**Method**	**Split (%)**	**Accuracy (%)**
Yang et al. [75]	2019	XLNet	94/0/6†	98.5
Xie et al. [71]	2019	BERT	94/0/6‡	98.1
Howard and Ruder [24]	2018	ULMFiT	94/0/6	97.8
Johnson and Zhang [26]	2017	DPCNN	94/0/6	97.4
Zhang et al. [78]	2015	CharCNN	94/0/6	95.1

Table 2.4: Comparative summary for the Yelp dataset. † trained on BookCorpus [81], English Wikipedia, Giga5 [49], ClueWeb 2012-B (extended from [6]), and Common Crawl [12]. ‡ trained on BookCorpus [81] and English Wikipedia.

version of the Yelp dataset[1], "2013 - Version 2" is composed of 335,018 randomly selected user reviews, "2014 - Version 1" is composed of 231,163 randomly selected user reviews from the 2014 version of the Yelp dataset, "2014 - Version 2" is composed of 1,125,457 randomly selected user reviews, and "2015 - Version 1" is composed of 1,569,264 randomly selected user reviews from the 2015 version of the Yelp dataset.

The previous versions are all for classification in 5 classes. However, the "2015 - Version 2" comes in two types: 5 classes ("Yelp reviews full star dataset") and 2 classes ("Yelp reviews polarity dataset"). The Yelp reviews full star dataset was constructed by randomly taking 130,000 training samples and 10,000 testing samples for each review star from 1 to 5. In total, there are 650,000 training samples and 50,000 testing samples. The Yelp reviews polarity dataset was constructed by considering stars 1 and 2 as negative, and 4 and 5 as positive. For each polarity, 280,000 training samples and 19,000 testing samples were taken randomly. In total, there are 560,000 training samples and 38,000 testing samples.

[1] `https://www.yelp.com/dataset`

Chapter 3

Material and Methods

This chapter describes the methodology proposed in this work, as well as evaluation metrics, datasets and computational resources that will be used in the experiments during the development of the project.

3.1 Methodology

In this section, we describe the steps of the proposed methodology for Sentiment Analysis. There are six main steps in the process:

- Unsupervised language representation training (pre-training);
- Further unsupervised pre-training on in-domain data;
- Supplementary supervised training;
- Preprocessing of the input text;
- Fine-tuning on sentiment data;
- Assessing the classification performance.

An overview of the methodology is presented in Figure 3.1, where red arrows represent training phase and blue arrows represent testing phase. Each step is explained in more details in the subsequent sections.

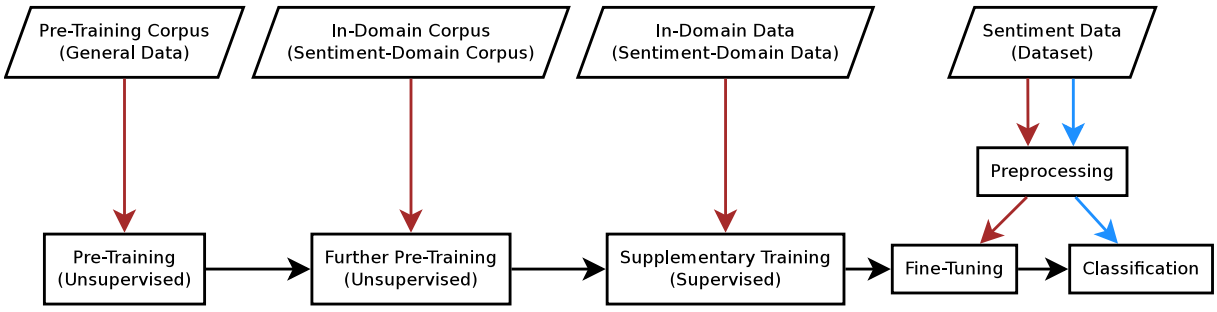

Figure 3.1: Proposed methodology for Sentiment Analysis.

3.1.1 Unsupervised Language Representation Training

The objective of this step is to produce sentence encoders with substantial knowledge of the target language. In order to achieve this, a large corpus is needed, which means it is com-

putationally expensive to train the model. Fortunately, there are pre-trained models available [15, 24, 52, 75]. One of the main advantages of this approach is that, by using unsupervised training, the process is scalable beyond only the subsets and domains of data that can be cleaned and labeled given resource, time, or other constraints.

3.1.2 Further Unsupervised Pre-Training on In-Domain Corpus

The model coming out of the previous step has broad knowledge about the language it was pre-trained on. This step aims to enhance the model's knowledge of sentences that express sentiment (in-domain corpus) by further pre-training it on such data. This is an optional step; we want to verify if it improves the performance of the final model.

3.1.3 Supplementary Supervised Training

After the model was pre-trained, we can try and improve the knowledge stored in the sentence encoders by performing supplementary supervised training on a dataset different from the one to be used later in fine-tuning. Intuitively, using a closely-related dataset should produce the best results, but the interaction between the supplementary training and the fine-tuning process needs a systematic and comprehensive study.

3.1.4 Preprocessing of the Input Text

When fine-tuning the model, we must preprocess the sentiment data in order to be usable. It can be as simple as just tokenizing the input text, or ignoring numbers, converting the text to lower case, filtering punctuation out, and so on. More involved preprocessing includes removing stop words, stemming (transforming a word into its root form, e.g., "playing" $\rightarrow$ "play"), and normalization (transforming text into a canonical form, e.g., "b4" $\rightarrow$ "before"). Different preprocessing schemes lead to (potentially) different classification results. We intend to discover the best scheme for our problem.

3.1.5 Fine-Tuning on Sentiment Data

In this step, we fine-tune the model to perform the target task, i.e., sentiment classification. This is the final step before the model can be effectively used. We prepare training and validation data using subsets of the datasets, and with this data we fine-tune the model. It is a supervised training process on the target dataset with the objective of making the final adjustments to the model using the same kind of data that it will encounter in the test phase.

3.1.6 Assessing the Classification Performance

This is the final step of the pipeline. After all the (pre-)training and fine-tuning is performed, classification performance is evaluated on the test data. The same preprocessing scheme utilized in Section 3.1.4 must be used here as well. The model will process the input text and output its sentiment: positive or negative.

3.2 Evaluation Metrics

To evaluate the performance of the proposed model we will use four widely employed evaluation metrics: Accuracy, Precision, Recall, and F1 score. In this section, we describe them briefly.

3.2.1 Accuracy

The most common performance metric used in Sentiment Analysis is accuracy, which is defined in Equation 2.10, in Section 2.2.1.

3.2.2 Precision

Precision is the proportion of positive identifications that were actually correct. We define Precision as:

$$\text{Precision} = \frac{\sum_{i=1}^{I}(TP_i)}{\sum_{i=1}^{I}(TP_i + FP_i)} \tag{3.1}$$

The meanings of i, I, TP_i, and FP_i are explained in Section 2.2.1. Values range from 0 (worst) to 1 (best).

3.2.3 Recall

Recall is the proportion of actual positives that were identified correctly. It is also known as Sensitivity. We define Recall as:

$$\text{Recall} = \frac{\sum_{i=1}^{I}(TP_i)}{\sum_{i=1}^{I}(TP_i + FN_i)} \tag{3.2}$$

The meanings of i, I, TP_i, and FN_i are explained in Section 2.2.1. Values range from 0 (worst) to 1 (best).

3.2.4 F1 score

F_1 score is the harmonic mean of Precision and Recall.

$$\text{F}_1 \text{ score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

Values range from 0 (worst) to 1 (best).

3.3 Datasets

In this section, the publicly available datasets to be used in order to validate our results are presented and briefly described. All datasets have two classes (positive and negative):

- **Movie Reviews (MR)**[1]: Organized by Pang and Lee [46] (Cornell University) and released in 2005, this dataset is widely used by many researchers in the Sentiment Analysis field. This short movie reviews dataset was built using data from the review-aggregation website Rotten Tomatoes and contains one sentence per review. The objective is to classify each

[1] http://www.cs.cornell.edu/people/pabo/movie-review-data

movie review into either positive or negative. The instances were labeled automatically, using the Tomatometer Ranking: reviews marked with "fresh" are assumed to be positive, and reviews marked with "rotten" are assumed to be negative.

- **Customer Reviews (CR)**[2]: This dataset is composed of three parts released separately. The first part was organized by Hu and Liu [25] (University of Illinois at Chicago), released in 2004, and it contains reviews for 5 products from Amazon.com and from CNET. The second part was organized by Ding et al. [16], released in 2008, and it contains reviews for 9 products from Amazon.com. And, finally, the third part was organized by Liu et al. [35], released in 2015, and it contains reviews for 3 products from Amazon.com. However, most of the related work used only the first two parts, that combined result in a dataset with customer reviews for 14 products, such as digital cameras, MP3 players, and cellular phones. For this reason, we will be using only the first two parts.
- **Multi-Perspective Question Answering (MPQA) Opinion Corpus**[3]: Wiebe, Wilson (University of Pittsburgh), and Cardie (Cornell University) [68] organized this dataset and released it in 2005. The relevant part for this work is the opinion polarity detection subtask of the dataset. In the MPQA corpus, sentiment polarities are attached not to sentences but to expressions (sub-sentences), and the common practice is to regard the expressions as sentences and classify the polarities.
- **Yelp Reviews Polarity**[4]: This dataset is a subset of the 2015 version of the Yelp dataset. It consists of randomly selected user reviews, and it was organized by Zhang et al. [78] (New York University). Since the original Yelp data has five classes (stars), to make this dataset binary, reviews with 1 or 2 stars were considered as negative, and reviews with 4 or 5 stars as positive, ignoring reviews with 3 stars.

Table 3.1 summarizes the main characteristics of these datasets. The third column refers to the number (N) of instances in each dataset, the fourth and fifth columns show the number of instances that have positive (N^+) and negative (N^-) sentiment, respectively, the sixth column is the average number of words (w) per instance, and the last column refers to the vocabulary size (V).

Dataset	Year	N	N^+	N^-	w	V
MR	2005	10662	5331	5331	21.01	18324
CR	2008	3746	2385	1361	18.38	5476
MPQA	2005	10514	3177	7337	3.04	5924
Yelp	2015	598000	299000	299000	134.04	214908

Table 3.1: Comparative summary of datasets for sentiment classification.

3.4 Baseline Results

In order to understand better the datasets and to have some results generated by ourselves, we performed some experiments on the datasets, using some well-known Machine Learning classifiers: Support Vector Machines (SVM), Naïve Bayes, Logistic Regression, and Random Forest. The results are shown in Table 3.2, organized according to the non-increasing order of the accuracy values.

[2] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
[3] http://mpqa.cs.pitt.edu/corpora/mpqa_corpus
[4] http://goo.gl/JyCnZq

The training protocol was:

- Randomly separate 10% of the dataset for testing;
- With the remaining 90%, perform Grid Search with stratified 10-fold cross-validation to determine the best values for the hyperparameters;
- Using the best values for the hyperparameters, test the model in the 10% of the data set aside in the beginning. The reported accuracy values are the ones from this step.

Movie Reviews (MR)	
Method	**Accuracy (%)**
Random Forest	78.7
SVM with RBF kernel	78.7
Logistic Regression	78.4
Linear SVM	78.0
Bernoulli Naïve Bayes	77.6
Multinomial Naïve Bayes	77.0
Customer Reviews (CR)	
Method	**Accuracy (%)**
Logistic Regression	80.5
Multinomial Naïve Bayes	79.5
Random Forest	79.2
SVM with RBF kernel	78.9
Linear SVM	78.7
Bernoulli Naïve Bayes	77.6
Multi-Perspective Question Answering (MPQA)	
Method	**Accuracy (%)**
SVM with RBF kernel	91.2
Logistic Regression	90.0
Linear SVM	89.4
Random Forest	88.4
Bernoulli Naïve Bayes	87.2
Multinomial Naïve Bayes	86.8

Table 3.2: Results obtained by using some baseline methods with the evaluated datasets.

3.5 Computational Resources

The methodology proposed in this work will be implemented using the Python programming language due to the availability of mature and well-documented libraries for text manipulation, Machine Learning, numerical computation, and graph plotting, such as NumPy, SciPy, scikit-learn, PyTorch, TensorFlow, Keras, and Matplotlib.

The experiments will be conducted in the Visual Informatics Laboratory of the Institute of Computing. The computers are equipped with 3.5 GHz Intel i7-3770 processors, 32 GB of RAM memory, and NVidia GeForce GTX TITAN Black graphics cards, with 2880 CUDA cores and default memory DDR5 of 6 GB and 7 Gbps clock, running Linux operating system.

Chapter 4

Work Plan and Schedule

Our research plan consists of the activities listed below:

1. Literature review;
2. Preparation of the datasets;
3. Choice of initial neural network as a starting point;
4. Definition of the methodology;
5. Tests with the new model;
6. Analysis and evaluation of results;
7. Documentation and publication of results;
8. Dissertation writing.

The estimated activity schedule of our 24-month research is presented in Table 4.1.

Activities	1^{st} year				2^{nd} year			
	1	2	3	4	1	2	3	4
Stage 1								
Literature review	•	•	•	•	•	•		
Data preparation		•	•	•				
Stage 2								
Selection of baseline neural network			•	•	•			
Development of the new model				•	•	•		
Experiments on the proposed methodology				•	•	•		
Stage 3								
Methodology refinement					•	•		
Result analysis					•	•	•	
Stage 4								
Result publication						•	•	•
Dissertation writing						•	•	•

Table 4.1: Activity list for two-year Master's degree divided into trimesters.

References

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *Computing Research Repository, abs/1409.0473*, pages 1–15, 2014.

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[3] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[4] E. Boiy and M.-F. Moens. A Machine Learning Approach to Sentiment Analysis in Multilingual Web Texts. *Information Retrieval*, 12(5):526–558, 2009.

[5] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A Large Annotated Corpus for Learning Natural Language Inference. In *20th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 632–642. Association for Computational Linguistics (ACL), 2015.

[6] J. Callan, M. Hoy, C. Yoo, and L. Zhao. ClueWeb09 Data Set, 2009.

[7] E. Cambria, B. Schuller, Y. Xia, and C. Havasi. New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems*, 28(2):15–21, 2013.

[8] Q. Cao, W. Duan, and Q. Gan. Exploring Determinants of Voting for the "helpfulness" of Online User Reviews: A text Mining Approach. *Decision Support Systems*, 50(2):511–521, 2011.

[9] X. Chen, Y. Rao, H. Xie, F. L. Wang, Y. Zhao, and J. Yin. Sentiment Classification Using Negative and Intensive Sentiment Supplement Information. *Data Science and Engineering*, 4(2):109–118, 2019.

[10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *Computing Research Repository, abs/1412.3555*, pages 1–9, 2014.

[11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12 (Aug):2493–2537, 2011.

[12] Common Crawl. Common Crawl. `https://commoncrawl.org`, 2007. Accessed: 2019-09-07.

[13] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *Computing Research Repository, abs/1705.02364*, pages 1–12, 2017.

[14] A. M. Dai and Q. V. Le. Semi-Supervised Sequence Learning. In *29th Conference on Neural Information Processing Systems (NIPS)*, pages 3079–3087. Neural Information Processing Systems (NIPS) Foundation, 2015.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *20th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT): Long and Short Papers - Volume 1*, pages 4171–4186. Association for Computational Linguistics (ACL), 2019.

[16] X. Ding, B. Liu, and P. S. Yu. A Holistic Lexicon-Based Approach to Opinion Mining. In *1st International Conference on Web Search and Data Mining (WSDM)*, pages 231–239. Association for Computing Machinery (ACM), 2008.

[17] Y. Du, X. Zhao, M. He, and W. Guo. A Novel Capsule Based Hybrid Neural Network for Sentiment Classification. *IEEE Access*, 7:39321–39328, 2019.

[18] J. L. Elman. Finding Structure in Time. *Cognitive Science*, 14(2):179–211, 1990.

[19] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to Forget: Continual Prediction with LSTM. In *9th International Conference on Artificial Neural Networks (ICANN)*, volume 2, pages 850–855. Institution of Electrical Engineers (IEE), 1999.

[20] Y. Goldberg. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[21] R. González-Ibáñez, S. Muresan, and N. Wacholder. Identifying Sarcasm in Twitter: A Closer Look. In *49th Annual Meeting of the Association for Computational Linguistics (ACL): Human Language Technologies (HLT): Short Papers - Volume 2*, pages 581–586. Association for Computational Linguistics (ACL), 2011.

[22] A. Graves, A.-r. Mohamed, and G. Hinton. Speech Recognition with Deep Recurrent Neural Networks. In *38th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649. Institute of Electrical and Electronics Engineers (IEEE), 2013.

[23] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, 1997.

[24] J. Howard and S. Ruder. Universal Language Model Fine-Tuning for Text Classification. In *56th Annual Meeting of the Association for Computational Linguistics (ACL): Long Papers - Volume 1*, pages 328–339. Association for Computational Linguistics (ACL), 2018.

[25] M. Hu and B. Liu. Mining and Summarizing Customer Reviews. In *10th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–177. Association for Computing Machinery (ACM), 2004.

[26] R. Johnson and T. Zhang. Deep Pyramid Convolutional Neural Networks for Text Categorization. In *55th Annual Meeting of the Association for Computational Linguistics (ACL): Long Papers - Volume 1*, pages 562–570. Association for Computational Linguistics (ACL), 2017.

[27] Y. Kim. Convolutional Neural Networks for Sentence Classification. *Computing Research Repository, abs/1408.5882*, pages 1–6, 2014.

[28] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-Thought Vectors. In *29th Conference on Neural Information Processing Systems (NIPS)*, pages 3294–3302. Neural Information Processing Systems (NIPS) Foundation, 2015.

[29] N. Kobayashi, K. Inui, and Y. Matsumoto. Opinion Mining from Web Documents: Extraction and Structurization. *Information and Media Technologies*, 2(1):326–337, 2007.

[30] Q. Le and T. Mikolov. Distributed Representations of Sentences and Documents. In *31st International Conference on Machine Learning (ICML)*, pages 1188–1196. International Conference on Machine Learning (ICML), 2014.

[31] Q. V. Le, N. Jaitly, and G. E. Hinton. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. *Computing Research Repository, abs/1504.00941*, pages 1–9, 2015.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[33] Y. LeCun, Y. Bengio, and G. Hinton. Deep Learning. *Nature*, 521(7553):436–444, 2015.

[34] B. Liu. Sentiment Analysis and Opinion Mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

[35] Q. Liu, Z. Gao, B. Liu, and Y. Zhang. Automated Rule Selection for Aspect Extraction in Opinion Mining. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1291–1297. International Joint Conferences on Artificial Intelligence (IJCAI), 2015.

[36] J. McAuley, R. Pandey, and J. Leskovec. Inferring Networks of Substitutable and Complementary Products. In *21th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794. Association for Computing Machinery (ACM), 2015.

[37] Y. Mejova and P. Srinivasan. Exploring Feature Definition and Selection for Sentiment Classifiers. In *5th International Conference on Weblogs and Social Media (ICWSM)*, pages 546–549. Association for the Advancement of Artificial Intelligence (AAAI), 2011.

[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *Computing Research Repository, abs/1301.3781*, pages 1–15, 2013.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *27th Conference on Neural Information Processing Systems (NIPS)*, pages 3111–3119. Neural Information Processing Systems (NIPS) Foundation, 2013.

[40] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. Mining Product Reputations on the Web. In *8th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 341–349. Association for Computing Machinery (ACM), 2002.

[41] T. Nakagawa, K. Inui, and S. Kurohashi. Dependency Tree-Based Sentiment Classification using CRFs with Hidden Variables. In *11th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 786–794. Association for Computational Linguistics (ACL), 2010.

[42] M. Nakatsuji and Y. Fujiwara. Linked Taxonomies to Capture Users' Subjective Assessments of Items to Facilitate Accurate Collaborative Filtering. *Artificial Intelligence*, 207:52–68, 2014.

[43] T. Nasukawa and J. Yi. Sentiment Analysis: Capturing Favorability Using Natural Language Processing. In *2nd International Conference on Knowledge Capture (K-CAP)*, pages 70–77. Association for Computing Machinery (ACM), 2003.

[44] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen. Cross-Domain Sentiment Classification via Spectral Feature Alignment. In *19th International Conference on World Wide Web (WWW)*, pages 751–760. Association for Computing Machinery (ACM), 2010.

[45] B. Pang and L. Lee. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 271–278. Association for Computational Linguistics (ACL), 2004.

[46] B. Pang and L. Lee. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124. Association for Computational Linguistics (ACL), 2005.

[47] B. Pang and L. Lee. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135, 2008.

[48] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs Up? Sentiment Classification using Machine Learning Techniques. In *7th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, volume 10, pages 79–86. Association for Computational Linguistics (ACL), 2002.

[49] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda. English Gigaword 5th Edition. Technical report, Linguistic Data Consortium, 2011.

[50] R. Pascanu, T. Mikolov, and Y. Bengio. Understanding the *Exploding Gradient* Problem. *Computing Research Repository, abs/1211.5063*, 2:1–11, 2012.

[51] J. Pennington, R. Socher, and C. Manning. GloVe: Global Vectors for Word Representation. In *19th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics (ACL), 2014.

[52] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep Contextualized Word Representations. In *19th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT): Long Papers - Volume 1*, pages 2227–2237. Association for Computational Linguistics (ACL), 2018.

[53] J. Phang, T. Févry, and S. R. Bowman. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *Computing Research Repository, abs/1811.01088*, pages 1–12, 2018.

[54] Q. Qian, M. Huang, J. Lei, and X. Zhu. Linguistically Regularized LSTM for Sentiment Classification. *Computing Research Repository, abs/1611.03949*, pages 1–11, 2016.

[55] A. Radford, R. Jozefowicz, and I. Sutskever. Learning to Generate Reviews and Discovering Sentiment. *Computing Research Repository, abs/1704.01444*, pages 1–9, 2017.

[56] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving Language Understanding by Generative Pre-Training. Technical report, OpenAI, 2018.

[57] S. Saha. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. `https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53`, 2018. Accessed: 2019-08-11.

[58] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. In *28th Conference on Neural Information Processing Systems (NIPS)*, volume 27, pages 3104–3112. Neural Information Processing Systems (NIPS) Foundation, 2014.

[59] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, 37(2):267–307, 2011.

[60] D. Tang, B. Qin, and T. Liu. Learning Semantic Representations of Users and Products for Document Level Sentiment Classification. In *53rd Annual Meeting of the Association for Computational Linguistics (ACL) and 7th International Joint Conference on Natural Language Processing (IJCNLP): Long Papers - Volume 1*, pages 1014–1023. Association for Computational Linguistics (ACL), 2015.

[61] D. Tang, B. Qin, and T. Liu. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. In *20th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1422–1432. Association for Computational Linguistics (ACL), 2015.

[62] M. Tsytsarau and T. Palpanas. Survey on Mining Subjective Data on the Web. *Data Mining and Knowledge Discovery*, 24(3):478–514, 2012.

[63] P. D. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424. Association for Computational Linguistics (ACL), 2002.

[64] P. D. Turney and M. L. Littman. Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Transactions on Information Systems*, 21(4):315–346, 2003.

[65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is All You Need. In *31th Conference on Neural Information Processing Systems (NIPS)*, pages 5998–6008. Neural Information Processing Systems (NIPS) Foundation, 2017.

[66] J. Wang, Z. Wang, D. Zhang, and J. Yan. Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. In *26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2915–2921. International Joint Conferences on Artificial Intelligence (IJCAI), 2017.

[67] S. Wang and C. D. Manning. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In *50th Annual Meeting of the Association for Computational Linguistics (ACL): Short papers - Volume 2*, pages 90–94. Association for Computational Linguistics (ACL), 2012.

[68] J. Wiebe, T. Wilson, and C. Cardie. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, 39(2-3):165–210, 2005.

[69] A. Williams, N. Nangia, and S. Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *19th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT): Long Papers - Volume 1*, pages 1112–1122. Association for Computational Linguistics (ACL), 2018.

[70] Z. Wu, X.-Y. Dai, C. Yin, S. Huang, and J. Chen. Improving Review Representations with User Attention and Product Attention for Sentiment Classification. In *32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 5989–5996. Association for the Advancement of Artificial Intelligence (AAAI), 2018.

[71] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le. Unsupervised Data Augmentation for Consistency Training. *Computing Research Repository, abs/1904.12848*, pages 1–20, 2019.

[72] J. Xu, D. Chen, X. Qiu, and X. Huang. Cached Long Short-Term Memory Neural Networks for Document-Level Sentiment Classification. In *21st Conference on Empirical Methods in*

Natural Language Processing (EMNLP), pages 1660–1669. Association for Computational Linguistics (ACL), 2016.

[73] M. Yang, W. Zhao, L. Chen, Q. Qu, Z. Zhao, and Y. Shen. Investigating the Transferring Capability of Capsule Networks for Text Classification. *Neural Networks*, 118:247–261, 2019.

[74] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical Attention Networks for Document Classification. In *17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1480–1489. Association for Computational Linguistics (ACL), 2016.

[75] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Computing Research Repository, abs/1906.08237*, pages 1–18, 2019.

[76] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine*, 13(3):55–75, 2018.

[77] L. Zhang, S. Wang, and B. Liu. Deep Learning for Sentiment Analysis: A Survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.

[78] X. Zhang, J. Zhao, and Y. LeCun. Character-Level Convolutional Networks for Text Classification. In *29th Conference on Neural Information Processing Systems (NIPS)*, pages 649–657. Neural Information Processing Systems (NIPS) Foundation, 2015.

[79] Y. Zhang, Z. Zhang, D. Miao, and J. Wang. Three-Way Enhanced Convolutional Neural Networks for Sentence-Level Sentiment Classification. *Information Sciences*, 477:55–64, 2019.

[80] H. Zhao, Z. Lu, and P. Poupart. Self-Adaptive Hierarchical Sentence Model. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4069–4076. International Joint Conferences on Artificial Intelligence (IJCAI), 2015.

[81] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *15th International Conference on Computer Vision (ICCV)*, pages 19–27. Institute of Electrical and Electronics Engineers (IEEE), 2015.