2023

Figurative Language Detection using Deep Learning and Contextual Features

Md Saifullah Bin Razali

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Figurative Language Detection using Deep Learning and Contextual Features

Md Saifullah Bin Razali

This thesis is presented as part of the requirements for the conferral of the degree:

Doctor of Philosophy

Supervisor:
Dr. Yang-Wai Chow

Associate supervisor:
Dr. Alfian Bin Abdul Halin

The University of Wollongong
School of Computing and Information Technology

June, 2023

This research has been conducted with the support of an Australian Government Research Training Program Scholarship.

Declaration

I, *Md Saifullah Bin Razali*, declare that this thesis is submitted in partial fulfilment of the requirements for the conferral of the degree *Doctor of Philosophy*, from the University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualifications at any other academic institution.

Md Saifullah Bin Razali

July 3, 2023

Abstract

The size of data shared over the Internet today is gigantic. A big bulk of it comes from postings on social networking sites such as Twitter and Facebook. Some of it also comes from online news sites such as CNN and The Onion. This type of data is very good for data analysis since they are very personalized and specific. For years, researchers in academia and various industries have been analyzing this type of data. The purpose includes product marketing, event monitoring, and trend analysis. The highest usage for this type of analysis is to find out the sentiments of the public about a certain topic or product. This field is called sentiment analysis. The writers of such posts have no obligation to stick to only literal language. They also have the freedom to use figurative language in their publications. Hence, online posts can be categorized into two: Literal and Figurative. Literal posts contain words or sentences that are direct or straight to the point. On the contrary, figurative posts contain words, phrases, or sentences that carry different meanings than usual. This could flip the whole polarity of a given post. Due to this nature, it can jeopardize sentiment analysis works that focus primarily on the polarity of the posts. This makes figurative language one of the biggest problems in sentiment analysis. Hence, detecting it would be crucial and significant. However, the study of figurative language detection is non-trivial. There have been many existing works that tried to execute the task of detecting figurative language correctly, with different methodologies used. The results are impressive but still can be improved. This thesis offers a new way to solve this problem. There are essentially seven commonly used figurative language categories: sarcasm, metaphor, satire, irony, simile, humor, and hyperbole. This thesis focuses on three categories. The thesis aims to understand the contextual meaning behind the three figurative language categories, using a combination of deep learning architecture with manually extracted features and explore the use of well know machine learning classifiers for the detection tasks. In the process, it also aims to describe a descending list of features according to the importance. The deep learning architecture used in this work is Convolutional Neural Network, which is combined with manually extracted features that are carefully chosen based on the literature and understanding of each figurative language. The findings of this work clearly showed improvement in the evaluation metrics when compared to existing works in the same domain. This happens in all of the figurative language categories, proving the framework's possession of quality.

Acknowledgments

This thesis would not be possible without the grace and guidance of Allah SWT and Prophet Muhammad SAW.

Hereby, I would also like to acknowledge the roles of individuals for their invaluable time spent on me:

I would like to express my gratitude towards my supervisory committee; Associate Professor Dr. Casey Chow and Dr Alfian Abdul Halin that have been attending to all my questions and concerns. Thank you for always making sure that the process is going well.

Special thanks to my support circle; my wife, Nor Afizah Mustafa, my family especially my parents, Dr Razali Shahul Hameed, Noriyah Othman, and my parent-in-law, Rohani Abdul Manaf. Also thanks to my friends that have been there with me throughout this phase.

All of you have kept me going.

Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The size of data shared over the Internet today is tremendous. A big part of the whole bulk comes from short-text posts in social networking sites such as Twitter and Facebook. Some of it also comes from online news sites such as Cable News Network (CNN) and The Onion. This type of data is very good for data analysis since they are very personal [1]. For years, researchers in academia and various industries have been analysing this type of data. The purpose includes for product marketing, event monitoring and trend analysis [2, 3]. This field is called sentiment analysis [3].

However, the writer of such posts has no obligation to stick to only literal language. This gives them freedom to also use figurative language (FL). Hence, online posts can be categorized into two: literal or figurative. Literal posts commonly contain traditional words that can be found in a dictionary with no other meaning than what it is intended to. On the contrary, figurative posts may contain words or phrases that carry a different meaning than the standard. This could flip the whole polarity of a given post. Consider the sentence “I am very sad to see a corrupt leader thrown to prison”. If the sentence is taken as literal when it is not supposed to, all the evaluations of algorithms would decrease. This problem would be more severe when there are many instances like this in the dataset.

Due to this nature, it can jeopardize a sentiment analysis work that only focuses on polarity of posts. This makes the work of FL detection a non-trivial task and one of the biggest problems in sentiment analysis [4, 5]. Hence, detecting it would be crucial and significant. There are existing works that try to tackle the problem of FL by specific types. Most of it is using rule-based, lexicon and seeding techniques. All of these ways are very rigorous, and did not even come close to the results yielded by the works using deep learning. However, by being automated, deep learning also suffers from a big problem in that the contextual meaning behind every FL is left out.

In the real world, FL is used a lot in everyday conversations to convey ideas which are difficult to visualize without it [6]. Even with this realization, most researchers in this field are still focusing on literal language [7]. This is because of all the reasons stated previously which also makes FL very implicit and natural [8] and again the task of detecting it is very important.

There are essentially seven commonly used FL categories: sarcasm, irony, simile, metaphor, satire, humor and hyperbole [5]. However, some of these concepts are overlapping or the distinction between one another is not very clear.

For example, sarcasm is sometimes considered the same as irony [9, 10]. However, there is a slight difference. Sarcasm is proven as a contrast between positive sentiment and negative situation [11]. In the case of irony, it is proposed that the speaker of the utterance would pretend to be an unwise person [12]. This theory is then supported by another work [13].

Simile and metaphor have the exact same structure apart from the use of words like "like" and "as" [14]. For example, the sentence "My office is like the Antarctica" would mean that "My office" is very cold since Antarctica is known for being a cold place. A metaphoric instance of the same sentiment would be to use the same sentence but without the word "like".

In the case of humor, it is fully dependent on culture and language for it to be understood [15]. For example, the sentence "Your friend is green" could be perceived as funny in the Malaysian culture, but not anywhere else. Hyperbole, is just a sentence with exaggeration [16]. Consider the sentence "There are millions of them". It would be agreed that this sentence does not carry the literal meaning for the word "millions". Instead, the speaker is just trying to portray high volume.

In comparison to the other FL types mentioned previously, satire is the most unique in terms of data collection. This is because satire does not only happen in short sentences, but also throughout paragraphs or books. It is a critiquing technique used upon a particular scenario or situation. For example, the book Gulliver's Travels is a critique on the writers of travelogues that are persistent on making their travels sound unique in their books [17]. In Gulliver's Travels, the author writes about his encounter with giants and very tiny people [18]. These giants and very tiny people do not actually exist.

All of the explanations of the FL types above bring this work to its main focus. It is to only detect three categories of FL: sarcasm, metaphor, and satire. There are existing works done to detect each of the FL types. The main difference between the works are the methods used. Some works used rule-based techniques and some works used deep learning techniques. There is limited work using the combination of both. All of these existing methods are thoroughly explained in the literature review chapter of this work.

1.2 Research Motivation

Today, a large amount of data is accessible by basically anyone anywhere. This information covers various topics and sentiments. They can be very valuable to decision makers in the industry or academia. They can even solve existing problems in these organizations. However, the data themselves can pose multiple problems. One of the biggest problems is when the data is using FL or having different meanings than what they are supposed to mean. For example, in the case of sentences or words used to convey sarcasm, metaphor or satire to the receiver, a machine can hardly detect the sender's real intention. The task to analyze and come up with a solid verdict is even more challenging.

This research aims to detect the three biggest categories of FL in text: sarcasm, metaphor, and satire. The reason is that when they are detectable, they can be analyzed correctly. The information gathered from the analysis done on well understood data would be much more valuable. Another interesting motivation is to investigate whether the contexts of all these FL categories will bring any points to the sheer detection of them.

1.3 Problem Statement

One of the biggest problems in sentiment analysis is FL detection, due to its ability to flip the polarity of instances. This would jeopardize the evaluations for each algorithm used in experiments. There are many categories of FL and differentiating one from another is not a trivial task. This is due to each of them being contextually unique.

Hence, several research works in the domain of FL detection have been reviewed and summarized as below:

1. Sarcasm detection: It is found that most of recent works use supervised learning approaches to tackle the issues in sarcasm [19–21]. Although some of them use semi-supervised [11, 22–27] or rule-based approaches [28–33], the performance of these works are not as good as the previously mentioned works using supervised learning. This makes the two latter approaches almost obsolete. However, the context of sarcasm in each of the sentences in the dataset is abandoned in a given supervised learning setting.

2. Metaphor detection: Even though this domain has been studied a lot from the perspective of psychology, linguistics, sociology, anthropology, and computational linguistics [34], the main issue with metaphor detection remains unsolved. Metaphor is deemed by most researchers as very challenging to understand [8]. There have been works done on building a metaphor dataset with seeding technique [8, 35]. They collected metaphoric instances manually and used these seeds to create a

larger corpus while preserving the same sentence structure. However, seeding technique is very rigorous. There is no work that has been done to fully automate the detection or collection of metaphor instances. This is partly because there is not much work done on the understanding of contextual meaning behind metaphorical instances.

3. Satire detection: Satire in computational perspective are rare [36]. Apart from being used as an indicator in the works of fake news detection [37–39], there are hardly credible works on satire detection. It is mentioned that the acceptance of satire among the writers and the readers is based on commonality, a lot like sarcasm [40]. The main issue faced in satire is also the same as sarcasm, which is the detection of context and intent for each of the instances in the datasets.

As a result of the thorough reviews done on existing works, the problem statements for this thesis are found and finalized as below:

1. The work of understanding the contextual meaning behind FL instances is rare.
2. The existing works either only use deep learning architectures that abandon the contextual meanings of instances or use manually extracted features through rigorous process. Combinatory works are almost non-existence.
3. No existing exploratory work on using well known Machine Learning classification models in relation to FL detection tasks.
4. No existing literature about the most to the least important features of each FL type.

1.4 Research Objectives

FL detection is an actively studied domain where several approaches and techniques have been used to support the task. However, it is done on each category respectively, i.e. sarcasm detection, metaphor detection and satire detection. In this work, the general objective is to devise the detection approach for three categories of FL. To achieve this general objective, four objectives have been outlined:

1. To propose the most essential and justified contextual-based feature sets for each of the FL category detection tasks (sarcasm detection, metaphor detection, satire detection).
2. To design effective combinations of deep features and manually extracted features.
3. To experiment and compare the outcomes of Machine Learning classification models in relation to FL detection tasks.
4. To experiment and find the most to the least important features of each FL type.

1.5 Research Scope

The research scope is as the following:

1. According to the literature review that is done for this work, there are still many loopholes and issues that needs to be solved in FL detection in text. For this reason, only text modality is covered in this work.

2. Only three FL is chosen for this work: Sarcasm, Metaphor, Satire. This is due to the fact that Sarcasm and Irony are very similar, apart from that Sarcasm is more clearly defined in literature. Metaphor and Simile are also very similar, apart from that Simile is only adding the words “like” and “as” to instances of Metaphor. Hyperbole only means sentences that use exaggeration such as multiple exclamation marks and Humor is very dependent on the culture it is used in. For these reasons this work is not experimenting on Irony, Simile, Hyperbole and Humor.

3. Only data that is available online is used. This is to ease the process of comparing the performance of this work with the existing works in the same domain.

4. There are over 7000 languages that exist in the world today. Each of the languages has their own way of conveying messages from speakers to listeners or writers to readers. Since English is the most spoken language with 1132 million speakers, it is the only language chosen for this work.

1.6 Research Contributions

This thesis has made the following contributions:

1. Feature extraction methods that are based on the contextual justifications of each FLs.

2. A proof that using the combination of a deep learning architecture with carefully crafted contextual features can optimize the performance of natural language processing (NLP) tasks.

3. A thorough comparison of the performances of machine learning classifiers post-experimentation on each FL detection task.

4. A thorough findings of the most to the least important feature sets for every FL category.

1.7 Thesis Organization

This thesis contains seven chapters. Chapter two will describe the literature review in detail. The background of existing research in FL detection: i. sarcasm detection, ii. metaphor detection, iii. satire detection and techniques that have been used will be identified and described in detail. This chapter will also highlight important criteria that an FL detection work should possess. Finally, this chapter also discusses the issues posed by the relevant literature.

Next, chapter three will describe the research methodology that has been used throughout the work. Generally, this chapter will explain all the activities involved in defining the significant components, classification, and evaluation of the framework.

The fourth chapter will discuss a new framework for FL detection using deep learning with contextual features. In this chapter, the focus is on sarcasm detection. The fifth chapter focuses on metaphor detection. The sixth chapter focuses on satire detection.

Finally, the seventh chapter will present the conclusion of this research and future works recommended.

1.8 Summary

This research is an integration of figurative language detection with its contextual features as well as machine learning. It highly contributes to identify the useful methods for the specific task of the detection. This chapter presents the essence of the thesis; issues faced, the motivation for this work and the main contributions. In the next chapters, more details will be given to the techniques, models and the experimental analysis.

1.9 Publications Arising from This Thesis

This thesis managed to produce a list of publications in the specific domain. The publications include:

- Razali, M.S., Halin, A.A., Norowi, N.M. and Doraisamy, S.C., 2017, December. The importance of multimodality in sarcasm detection for sentiment analysis. In 2017 IEEE 15th Student Conference on Research and Development (SCOReD) (pp. 56-60). IEEE.
- Razali, M.S., Halin, A.A., Ye, L., Doraisamy, S. and Norowi, N.M., 2021. Sarcasm detection using deep learning with contextual features. IEEE Access, 9, pp.68609-68618.
- Razali, M.S., Halin, A.A., Chow, Y.W., Norowi, N.M. and Doraisamy, S., 2022. Context-Driven Satire Detection With Deep Learning. IEEE Access, 10, pp.78780-78787.
- Razali, M.S., Halin, A.A., Chow, Y.W., Norowi, N.M. and Doraisamy, S., 2022, December. Deep Learning Metaphor Detection with Emotion-Cognition Association. In 2022 International Conference on Digital Transformation and Intelligence (ICDI) (pp. 8-14). IEEE.

Chapter 2

Literature Review

2.1 Introduction

FL is the use of a language to convey non-literal intentions. There are essentially seven commonly used FL categories: sarcasm, irony, simile, metaphor, satire, humor and hyperbole. However, some of these concepts are overlapping or the distinction between one and another is unclear. For example, sarcasm is sometimes blatantly considered the same as irony [9, 10]. Simile and metaphor have the exact same structure apart from the use of words like "like", "as" and "than" [41]. For example, the sentence "You are like a diamond" would be a simile and "You are a diamond" would be a metaphor. As for the case of humor, it is fully dependent on culture and language for it to be understood [15], making it too vague to be computational processed without the need to be specific on the culture and language. The concept of hyperbole is considered too small in comparison to the other FLs. It is just an exaggeration such as excess exclamation marks or question marks in a sentence. However, it is used in most of the other FLs [42, 43]. Hence, we have decided to focus our work on only three FL concepts: sarcasm, metaphor and satire in their respective automatic detection tasks, using the ways that are explained in the methodology chapter. This work only focuses on online texts in the English language.

This literature review begins with a close look at works on sarcasm detection. Sarcasm is essentially positive sentences with underlying negative intent [11]. There are many existing works on sarcasm detection. Most of them use rule-based approaches [28–30]. For a rule-based approach to work, the authors of the rules first need to work the context of a given FL from multiple point of view. This view is including that of psychology. Only then rules such as "the number of times a positive word is followed by a negative word, and vice versa" [10] and "number of quotes" [27] can be found and used.

In this work, rule-based technique is also used. However, the difference is that we are also focusing on the combination of it with deep learning. Deep learning on the other hand is proven to be quite capable and time efficient in solving NLP tasks, but it is leaving out the contextual meanings. Hence, the focus of leveraging both of these ways to increase the efficiencies, by using both.

Secondly, metaphor is looked at closely. It has been studied a lot in the perspective of psychology, linguistics, sociology, anthropology and computational linguistics [34]. However, the main issue of metaphor detection or also known as metaphor identification remains unsolved. It is the detection of intent behind the metaphorical sentences. One work [8] has worked on building a large metaphor dataset with seeding technique. They collected metaphoric instances manually and used these seeds to create a large dataset while preserving the same sentence structure. The seeding technique is very rigorous. It takes a lot of time and energy to be put on, in order to get the seeds right. Even then, they did not solve the problem of detecting metaphor optimally.

The third FL in this work is satire. Existing works of satire from a computational perspective are rare [44]. Apart from being used as an indicator in the works of fake news detection [38, 39, 45], there are hardly credible works on satire detection. One work mentioned that the acceptance of satire among the writers and the readers is based on commonality [40], a lot like sarcasm.

Therefore, the main issue faced in sarcasm, which is the detection of intent, is also faced in metaphor and satire. The reason why this work is able to see the similarities between the FLs is because it is working on them simultaneously. However, these three FL categories are also very distinct from the contextual viewpoint. This part is discussed in this work extensively and also then utilized in the methodology chapter. At the fundamental level, the FL categories have similarities and also very distinct at the same time. This work utilizes both in its methodology; using rule-based to extract the contextual meanings and deep learning to automatically extract the common patterns.

2.2 Machine Learning

One of the largest parts of this work is deep learning. Deep learning is an architecture created with neural network as the backbone. Since neural network is a part of machine learning, this section offers an overview of machine learning.

2.2.1 Supervised Learning

Machine learning is a field of work that overlaps with many other fields. This includes artificial intelligence, data mining and data analytics. The focus of the field is for a machine to acquire knowledge through experience or training. Most commonly, this means using useful concepts through the analysis of existing data.

There are many types of machine learning. Most commonly, the three types are supervised learning, unsupervised learning and reinforcement learning. Supervised learning describes a class of problem that involves using a model to learn a mapping between input examples and the target variable. Applications in which the training data comprises examples of the input vectors along with their corresponding target vectors are known as supervised learning problems [46].

Models are fit on training data consisting of inputs and outputs and used to make predictions on test sets where only the inputs are provided and the outputs from the model are compared to the withheld target variables and used to estimate the skill of the model.

Learning is a search through the space of possible hypotheses for one that will perform well, even on new examples beyond the training set. To measure the accuracy of a hypothesis we give it a test set of examples that are distinct from the training set [47].

There are two main types of supervised learning; classification that involves predicting a class label and regression that involves predicting a numerical value. Both classification and regression problems may have one or more input variables and input variables may be any data type, such as numerical or categorical.

An example of a classification problem would be the MNIST handwritten digits dataset where the inputs are images of handwritten digits in the form of pixels and the output is a class label for what digit the image represents, which are the numbers 0 to 9 [48]. An example of a regression problem would be the Boston house prices dataset where the inputs are variables that describe a neighborhood and the output is a house price in dollars [49].

Popular examples for supervised machine learning include K-nearest neighbor, logistic regression, discriminant analysis, decision trees and support vector machines which are all going to be used in this work [50].

These algorithms are referred to as supervised because they learn by making predictions given examples of input data, and the models are supervised and corrected via an algorithm to better predict the expected target outputs in the training dataset. The term supervised learning originates from the view of the target being provided by human beings [51].

Some algorithms are specifically designed for classification, such as logistic regression or regression, such as linear regression. Some may be used for both classification and regression, such as neural networks. As mentioned earlier in this section, neural network is a large part of this work. The architecture used is explained in the research methodology chapter.

2.2.2 Unsupervised Learning

Unsupervised learning describes a class of problems that involves using a model to describe or extract relationships in data. Compared to supervised learning, unsupervised learning operates upon only the input data without outputs or target variables. As such, unsupervised learning does not have corrections to the model, as in the case of supervised learning [51].

There are many types of unsupervised learning. However, there two main types that are often used by practitioners. They are clustering that involves finding groups in the data and density estimation that involves summarizing the distribution of data.

An example of a clustering algorithm is k-means where k refers to the number of clusters to discover in the data. An example of a density estimation algorithm is Kernel Density Estimation that involves using small groups of closely related data samples to estimate the distribution for new points in the problem space. The most common unsupervised learning task is clustering: detecting potentially useful clusters of input examples. For example, a taxi agent might gradually develop a concept of "good traffic days" and "bad traffic days" without ever being given labeled examples of each by a human being [47]

Additional unsupervised methods may also be used, such as visualization that involves graphing or plotting data in different ways and projection methods that involves reducing the dimensionality of the data. An example of a visualization technique would be a scatter plot matrix that creates one scatter plot of each pair of variables in the dataset. An example of a projection method would be Principal Component Analysis that involves summarizing a dataset in terms of eigenvalues and eigenvectors, with linear dependencies removed. The goal in such unsupervised learning problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization [47].

Reinforcement Learning

Reinforcement learning describes a class of problems where an agent operates in an environment and must learn to operate using feedback. Reinforcement learning is learning what to do and how to map situations to actions as to maximize a numerical reward signal. The machine is not told which actions to take, but instead must discover which actions yield the most reward by trying them [52].

The use of an environment means that there is no fixed training dataset, rather a goal or set of goals that an agent is required to achieve, actions they may perform, and feedback about performance toward the goal. Some machine learning algorithms do not just experience a fixed dataset. For example, reinforcement learning algorithms interact with an environment, so there is a feedback loop between the learning system and its experiences[51].

It is similar to supervised learning in that the model has some response from which to learn, although the feedback may be delayed and statistically noisy, making it challenging for the agent or model to connect cause and effect. An example of a reinforcement problem is playing a game where the agent has the goal of getting a high score and can make moves in the game and received feedback in terms of punishments or rewards.

In many complex domains, reinforcement learning is the only feasible way to train a program to perform at high levels. For example, in game playing, it is very hard for a human to provide accurate and consistent evaluations of large numbers of positions, which would be needed to train an evaluation function directly from examples. Instead, the program can be told when it has won or lost, and it can use this information to learn an evaluation function that gives reasonably accurate estimates of the probability of winning from any given position [51].

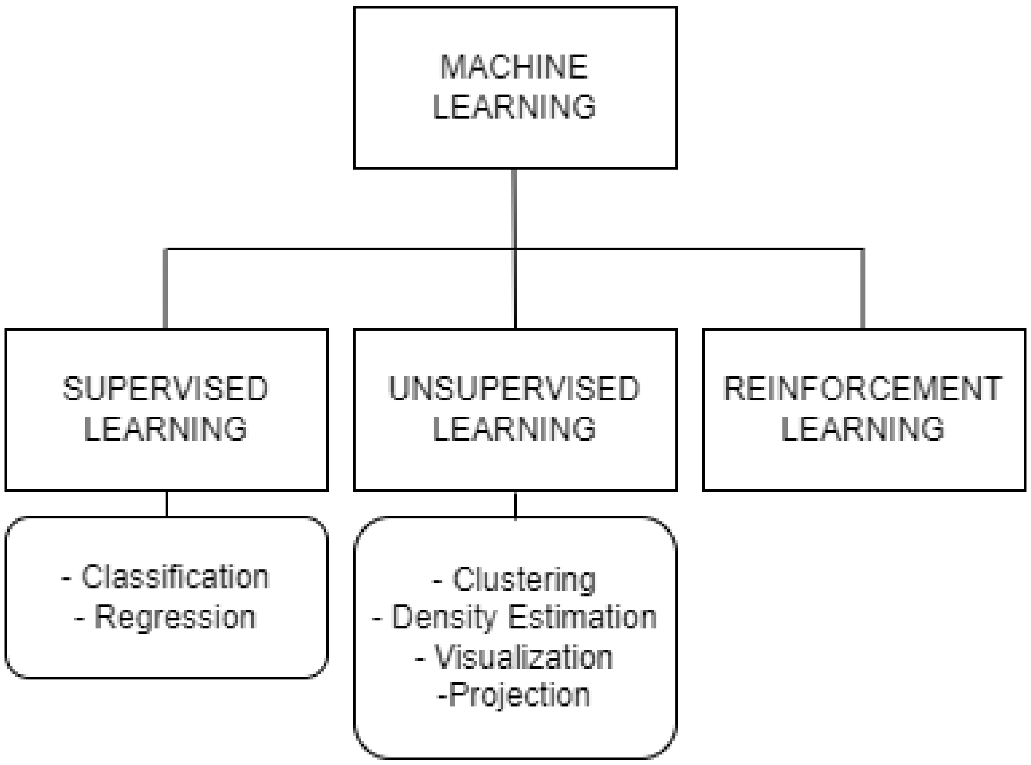


Figure 2.1: Machine Learning Types

2.3 Rule-Based

A rule-based system is applied to systems involving human-crafted or curated rule sets. An example of a rule-based system is the domain-specific expert system that uses rules to make deductions or choices. For example, an expert system might help a doctor choose the correct diagnosis based on a cluster of symptoms, or a gamer to select tactical moves to play a game [53].

Rule-based systems can be used to perform lexical analysis to compile or interpret computer programs, or in NLP [54]. Rule-based programming attempts to derive execution instructions from a starting set of data and rules. This is a more indirect method than that employed by an imperative programming language, which lists execution steps sequentially. A typical rule-based system has four basic components [55]

- A list of rules or rule base
- An inference engine or semantic reasoner
- Temporary working memory
- A user interface

The decision to use rule-based or machine learning or both is mostly depending on the need of a given task. Machine learning is mostly used to handle complex and intensive problems with varied environments, while a rule-based is more suited to situations with lower volumes of data and the rules are fairly simple. Even though this work might seem that it needs machine learning more at first, but it is evident towards the end that rule-based could also be very helpful in performing the tasks at hand.

2.4 Sarcasm

According to the Merriam-Webster dictionary, sarcasm sentences or utterances are usually used to either mock or annoy someone, or for humorous reasons. Sarcasm might be ironic, but it is not compulsory for it to be so [56]. However, it is largely context-dependent [57]. This serves as the first inspiration for this work, to extract and justify the contexts of the FLs and use it to leverage the detection tasks.

According to Riloff et al. [11], sarcasm is also described as a good statement or sentence with hidden malice. Sarcasm is defined by Merriam-Webster as “A sharp and frequently satirical or ironic speech designed to give pain” or “A kind of satirical wit depending for its effect on bitter, caustic, and often ironic language that is typically aimed against an individual.” According to the survey by Liu and Zhang [58], it is one of NLP field’s most difficult problems. Sarcasm can change the meaning of a statement, hence it is important for automated NLP systems to recognize and handle these expressions appropriately. Sarcasm detection was traditionally approached using rule-based algorithms, as in the works of Davidov et al. [22] and Riloff et al. [11]. However, more recent works have gone towards deep learning to detect significant features automatically [19, 59].

2.4.1 History of Sarcasm

Sarcasm originated from a Greek word “Sarkasmós”, which initially means to tear flesh, bite the lip in rage or sneer. All these words rule the violent enigma of physical strength. Though the origin of sarcasm was from Greek language, in the sixteenth century Latin language came up with the word “Sarcasm”.

To give a sarcastic comment one has to be creative. Sarcasm is generally defined as the opposite of what is true in order to make someone look or feel stupid. People can use irony to mock or convey contempt of a person and it can be extremely hurtful. Professionals in psychology and related fields have long looked upon sarcasm negatively [60, 61], particularly noting that sarcasm tends to be a maladaptive coping mechanism for those with unresolved anger or frustrations. Psychologist Clifford N. Lazarus describes sarcasm as “hostility disguised as humor”. While an occasional sarcastic comment may enliven a conversation, Lazarus suggests that too frequent use of sarcasm tends to “overwhelm the emotional flavor of any conversation [62]”.

However, sarcasm can be both demotivating or motivating, depending on the giving and the receiving end [63]. Since almost all of the works in this domain only look at sarcasm from a negative point of view, this work takes the liberty to look closely into the context of positive sarcasm as well. This is done by using positive words from a lexicon. This strategy is also inspired by the work by Riloff et al. [11] even though they have only used positive words to prove their assumption of “Sarcasm as Contrast between a Positive Sentiment and Negative Situation”.

Sarcasm actually has a noble tradition and there are many instances of it in some of Shakespeare's plays [64]. The British parliament has used sarcasm a lot in the past and present. Winston Churchill was considered to be the outright winner in a list of the funniest insults using sarcasm doled out by musicians, actors, politicians and other public figures. He was famous for his quick wit and sharp tongue and had an on-going verbal duel with leading Conservative Lady Astor who was forever chastising him for his cigar smoking and consumption of alcohol. But he was not one to take insults lying down. One memorable exchange had her saying to him "If you were my husband, I'd poison your tea!" to which he replied, "If you were my wife, I'd drink it!". He also said to her on one occasion, "I may be drunk Miss, but in the morning, I will be sober, and you will still be ugly!". Churchill also said to another Labour opponent "I'm not insulting you; I'm describing you!" and "I'm busy right now. Can I ignore you some other time?" [65, 66].

Mahatma Gandhi was also famous for being sarcastic. On a visit to London, a reporter asked him what he thought of Western civilisation, to which Gandhi replied "I think it would be a good idea!" [67]. Ronald Reagan said "I have left orders that I am to be alerted at any time in case of national emergency, unless I'm in a cabinet meeting" [68]. Abraham Lincoln remarked that "It is better to remain silent and be thought a fool, than to speak out and remove all doubt". All of these are proof that sarcasm is used en masse in the cultures of the world [69].

Cultural perspectives on sarcasm vary widely with more than a few cultures and linguistic groups finding it offensive to varying degrees. A Scottish Historian, Thomas Carlyle despised sarcasm. He said, "Sarcasm I now see to be, in general, the language of the devil; for which reason I have long since as good as renounced it" [70]. A Russian Novelist by the name of Fyodor Dostoyevsky said sarcasm is just a cry of pain. In his own words, he described sarcasm as "usually the last refuge of modest and chaste-souled people when the privacy of their soul is coarsely and intrusively invaded" [71]. RFC 1855, a collection of guidelines for Internet communications, includes a warning to be especially careful with sarcasm as it "may not travel well [71]. "Professional translators advised that international business executives "should generally avoid sarcasm in intercultural business conversations and written communications" because of the difficulties in translating sarcasm [72]. All of these are proof that sarcasm detection is crucial and not a trivial task.

In late August 2016, North Korea banned sarcasm against the government. It was reported that the government gave the warnings in mass meetings across the country. Subsequent media reports suggest that North Korea banned sarcasm altogether [73, 74]

Understanding the subtlety of this usage requires second-order interpretation of the speaker's or writer's intentions; different parts of the brain must work together to understand sarcasm. From a medical point of view, this sophisticated understanding can be lacking in some people with certain forms of brain damage, dementia and sometimes autism. Sarcasm detection research can also help doctors distinguish between different types of neurodegenerative diseases, such as frontotemporal dementia and Alzheimer's disease [75].

Sarcasm is also hypothesized to develop as a cognitive and emotional tool that adolescents use in order to test the borders of politeness and truth in conversation. Sarcasm recognition and expression both require the development of understanding forms of language, especially if sarcasm occurs without a cue or signal (e.g. a sarcastic tone or rolling of the eyes). Sarcasm is argued to be more sophisticated than lying because lying is expressed as early as the age of three, but sarcastic expressions take place much later during development [76].

In English, sarcasm is often conveyed with prosodic cues by speaking more slowly and with a lower pitch. Similarly, Dutch uses a lowered pitch; sometimes to such an extent that the expression is reduced to a mere mumble. In Cantonese, sarcasm is indicated by raising the fundamental frequency of one's voice. In Amharic, rising intonation is used to show sarcasm. There are also many other ways that speakers signal sarcastic intentions [77, 78].

Though in the English language there is no standard accepted method to denote irony or sarcasm in written conversation, several forms of punctuation have been proposed before the birth of social media. Among the oldest and frequently used a backwards question mark. More recent signs to denote sarcasm include the snark mark and the use of a tilde [79]. A bracketed exclamation point or question mark as well as scare quotes are also used [80]. All these are proof that from the beginning, users of the English language have realized the importance of understanding sarcasm.

The unavailability of features in sarcastic writings in comparison with sarcasm utterances is treated as another inspiration for this dissertation. Although this work does not go as far as proposing ways to denote sarcasm in English, one of the main focuses of this work is to find the most optimal contextual features from the normal English instances. This is done with adequate justifications for each feature.

2.4.2 Definition of Sarcasm

Sarcasm and irony are usually regarded as the same concept by most researchers. However, there are some works that differentiate sarcasm and irony [4, 5]. Sarcasm is also often referred to as just verbal irony [4]. The problems with these definitions and the reason why this dissertation does not thoroughly investigate the distinction between irony and sarcasm involves the ideas that:

- People can pretend to be insulted when they are not or pretend not to be insulted when they are seriously offended
- An individual may feel ridiculed directly after the comment and then find it humorous or neutral thereafter

- The individual may not feel insulted until years after the comment was expressed and considered

2.4.3 Related work

Sarcastic tweets frequently include hashtag terms like #sarcasm, #sarcastic, and #not [11, 22, 30]. According to these cited series of works, hashtags are the greatest ways to immediately spot sarcasm. The work by Liebrecht et al. [81] utilized the hashtag “#not” to indicate whether or not their tweets were sarcastic. For instance, it would be determined that the tweet “Donald Trump is the best president ever #not” was sarcastic. In order to determine the order of the features, they also used bigrams and trigrams. For instance, the term “One Hundred” is discovered 636 times in their dataset and 836 times for the term “Nineteen Eighty.” This would lower scores if only unigram is used. This is an inspiration taken into our work, which is to use multiple types of N-grams.

These older works also use rules in addition to relying solely on hashtags. For instance, Barbieri et al. used word frequency and rarity as their major features in their modeling work [82]. With additional guidelines for deriving sarcastic word patterns, Bouazizi and Ohtsuki [27] also employed this technique. The guidelines are for counting both the positive and negative words in a tweet as well as the emotionally charged terms, both positive and negative. These are not the only works that use rule-based methods to identify sarcasm. In 2013, Riloff et al. [11], had already utilized classifiers that search for positive verbs in sentences with negative contexts. The verbs and situational terms were put into a lexicon. Then, in their experiment, this lexicon is used to distinguish between sarcastic and regular statements. This work served as a model for other works that employ a similar technique of hashtag-assisted identification, either by using deep learning or manually handcrafted features [4, 19, 30, 83]. For example, the phrases “being sarcastic” as in “I was being sarcastic” or “She was being sarcastic” was utilized as a seed phrase in a more recent work by Shmueli et al [83]. The seed is then used to compile satirical Twitter posts. In the end, a fresh dataset is produced to assist in addressing the problem of sarcasm dataset scarcity. They used the reply tweet feature fed into CNN, BiLSTM and BERT networks. They call this the reactive supervision technique.

In recent developments, deep learning is the main method being applied to detect sarcasm features automatically [19, 59, 83] in contrast to older works which frequently used manually handcrafted methods [11, 22, 30]. This is in line with deep learning’s successful application to solve NLP issues. One of the most well-known deep learning architectures, the Convolutional Neural Network (CNN), was utilized by Poria et al. [19] to automatically extract four feature sets from four datasets. A Support Vector Machine (SVM) classifier is used to combine these feature sets and classify the data.

Another work from Ilić et al. [84] used a deep learning model based on character-level word representations derived by the Embeddings from Language Models (ELMo). ELMo is a representation technique that uses vectors derived from a bidirectional Long Short Term Memory (LSTM) [85].

Another recent work [86] experimented with the notion of leveraging historical knowledge to identify sarcasm using a deep learning Bidirectional Encoder Representations from Transformers (BERT) architecture. They made use of traditional conversational elements including response and last, second last, and third last utterances.

Other works created their features from user historical tweets. This concept was first tested by Bamman and Smith [30] with a careful examination of the key elements in a series of tweets from the same user. The characteristics are depicted in Figure 2.2. It is clear that combining all of the pertinent features will result in maximum accuracy. Each of these pertinent characteristics has its own set of guidelines. For instance, Twitter's audience feature takes advantage of previous conversations between the tweet's author and the person for whom it is intended. This concept was influenced by Kreuz and Caucci's work [87], which asserts that sarcasm is more likely to occur between people who are acquainted. Then, as part of the audience feature, the rank, the frequency of messages, and whether there has been at least one reciprocal message between the author and this other user are also added.

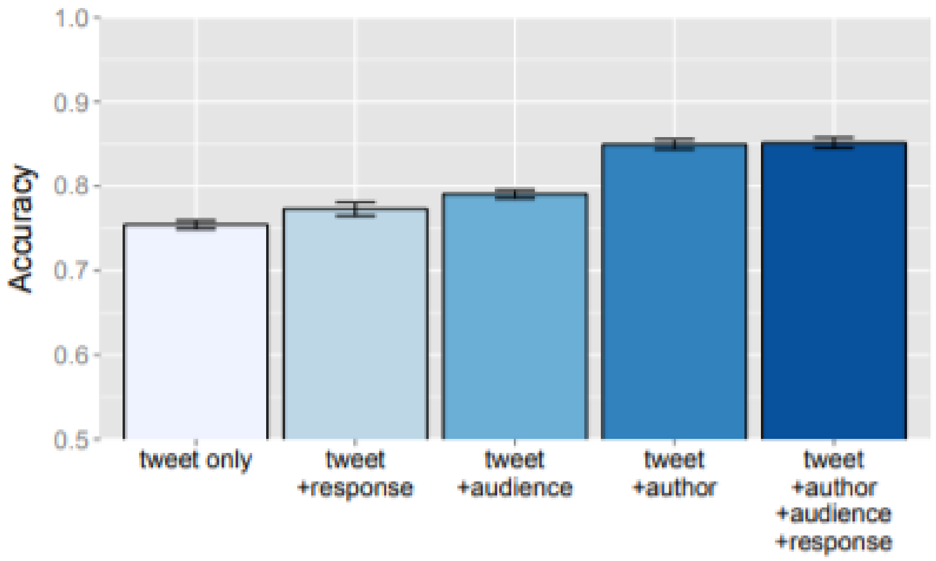


Figure 2.2: Most Relevant Features for Sarcasm Detection according to Bamman & Smith[30]

A few more works, like Amir et al. [88], Ghosh et al. [31], and Rajadesingan et al. [32] also use this technique. For instance, Rajadesingan et al. [32] developed a framework called Sarcasm Classification Using a Behavioral Modeling Approach (SCUBA) that categorizes a user's behavior using an approach similar to that used by Bamman and Smith

[30] (using historical tweets), but with additional factors like the difference in the word count between the users. They also tested whether the performance of sarcasm detection correlates directly with the availability of historical data, as seen in Figure 2.3.

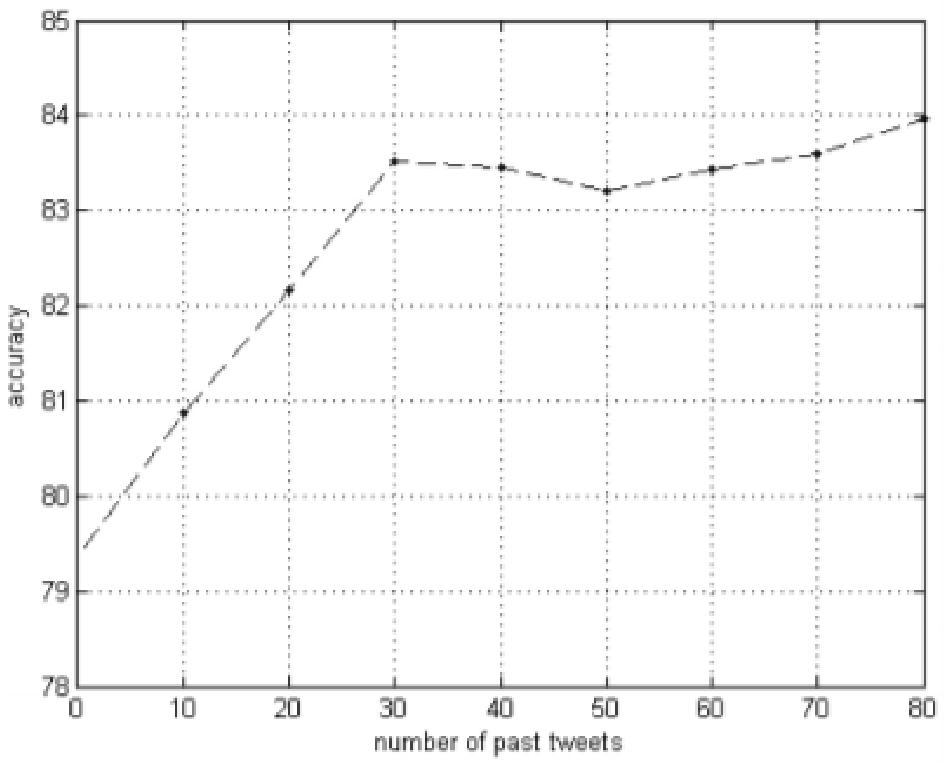


Figure 2.3: Direct Correlation between Availability of Historical Information and Sarcasm Detection Performance according to Rajadesingan et al. [32]

Another study on sarcasm detection looked at the context of particular tweets [10]. Compared to the other research projects employing historical data, this one produced better results. All the other context-based sarcasm detection studies, including from Riloff et al. [11], Bamman and Smith [30], and Poria et al. [19], included historical data in some way during their research. This fact is another one of the inspirations to only use data provided in the dataset for our work.

Exaggeration or hyperbole, interjections like “gee” or “gosh,” and punctuation marks like “?” are all examples of linguistic indicators [87]. They may also be useful indicators of sarcasm. Tsur et al. [23] even assert that words with modifications like “joy! ” or “no!” are components of sarcastic tendencies in Amazon product reviews. All of them have been tested in rule-based systems and confirmed to work.

Another prominent way to detect sarcasm is the use of lexicons [11, 87]. Many works have used lexicons to assist their sarcasm detection methods. For example, Riloff et al. [11] created their own lexicons of positive verbs and negative situations using a novel bootstrapping algorithm. They then use back these lexicons for the main sarcasm detection task. Existing lexicons such as Wordnet [89] have also been used to assist in different sarcasm detection tasks, mainly in the process of word counting [32, 90].

Apart from short-texts such as tweets, existing researchers also work on long-texts such as product reviews and online discussions [91, 92]. Interestingly, features that excelled in these works are in the form of N-Grams and Part-of-speech N-Grams. This gives a strong support to the researchers working with short texts. N-Grams based techniques could yield a good result if used correctly.

Sarcasm could also be temporal [93] in the sense that a sarcastic sentence could be regarded as ill-intentioned in a year but then good-intentioned in another year. However, many researchers believe that sarcasm is first created to be ill-intentioned unlikeness [11, 94]. If it uses any temporal words, it just means that the user is hoping that the situation changes in the future [93].

Table 2.1 below shows an overview of significant existing works by focusing on the methods used. The methods are broken into Manually Handcrafted (MH) and Deep Learning (DL). Since machine learning methods other than deep learning are seen in use only for classification tasks in the first category, it is combined with Manually Handcrafted for the reason that the feature extraction is done so. The results are mostly shown in the best F1-Measure since it is the harmony of precision and recall. However, other metrics are used when F1-Measure is not available. Figure 2.4 shows the corresponding percentage for Table 2.1.

Table 2.1: Methods from Existing Sarcasm Detection Works

Author	Method	Dataset	Result	Other Observation
Kreuz & Caucci (2007)	Regression analysis (MH)	100 instances of Google Book Search Corpus	The amount of variance is 5%	Some textual factors are reliable cues
Davidov et al. (2010)	Mechanical Turk tool (MH)	5.9 million tweets and 66000 product reviews from Amazon	F1-Measure of 0.78 on the product reviews dataset and 0.83 on the Twitter dataset	Psychological and cognitive patterns can benefit NLP systems
Liebrect et al. (2013)	Winnow Classification (MH)	78 thousand Dutch tweets for training, and 3.3 million Dutch tweets posted on a single day for testing	AUC score of 0.79	Hard to distinguish sarcastic tweets from literal tweets in an open setting
Riloff et al. (2013)	A novel bootstrapping algorithm (MH)	175,000 self-scrapped tweets	29% recall and precision of 38%	Identifying contrasting contexts using phrases learned through bootstrapping yields improve recall
Barbieri et al. (2014)	Decision Tree (MH)	10,000 tweets from three popular newspapers (New York Times, The Economist and The Guardian). 50,000 more added using related hashtags	F1-Measure of 0.92	Distinguishing between sarcasm from irony is a challenging issue

Author	Method	Dataset	Result	Other Observation
Ptacek et al. (2014)	Maximum Entropy and SVM (MH)	Dataset 1: English Twitter dataset of 100,000 balanced tweets. Dataset 2: English Twitter dataset of 100,000 imbalanced tweets. Dataset 3: Czech Twitter dataset consisting of 7,000 manually-labeled tweets.	F1-Measure of 0.98	Presented the evaluation of two classifiers with various combinations of feature sets on both the Czech and English datasets
Bamman & Smith (2015)	Logistic regression (MH)	The dataset is evenly balanced at 19,534 tweets	Accuracy of 98%	Most computational approaches to sarcasm detection before this treat it as a purely linguistic matter
Rajadesingan et al. (2015)	Logistic regression (MH)	9104 sarcastic tweets, self-described by users as being sarcastic using hashtags	Accuracy of 83%	Claimed to be first attempt on understanding sarcasm from the behaviour perspective
Kunneman et al. (2015)	Winnow classification (MH)	78,000 self-scrapped Dutch tweets	75% of the test data are really found to be sarcastic	The reliability of the hashtags can actually be questioned since users are not experts in sarcasm

Author	Method	Dataset	Result	Other Observation
Bharti et al. (2015)	Parsing and counting the occurrence of the interjection words (MH)	52,500 self-scrapped tweets	F1-Measure of 0.9	The idea of assisting sentiment analysis through sarcasm detection is born
Joshi et al. (2015)	LibSVM (MH)	Tweet-A (5208 tweets, 4170 sarcastic), Tweet-B (2278 tweets, 506 sarcastic), and Discussion-A (1502 discussion forum posts, 752 sarcastic)	F1-Measure of 0.61	Using context incongruity to detect sarcasm helps to improve the result
Bouazizi & Ohtsuki (2016)	LibSVM (MH)	58,609 self-collected tweets	Accuracy of 83%	Pattern-based features to computational detect sarcasm utterances help improve the result
Amir et al. (2016)	Content and User Embedding + CNN (A modified CNN) (DL)	11,541 self-collected tweets	Accuracy of 87%	Using user-embeddings is proven to be a good way to go
Poria et al. (2016)	CNN-SVM (CNN with SVM classification) (DL)	Dataset 1: 50,000 sarcastic tweets and 50,000 non-sarcastic tweets Dataset 2: 25,000 sarcastic tweets and 75,000 non-sarcastic tweets	F1-Measure of 0.93	Multiple feature sets are proven to provide added advantage

Author	Method	Dataset	Result	Other Observation
Ilic et al. (2018)	ELMo embeddings + LSTM) (DL)	36,835 tweets from the Ptacek corpus, 8,095 tweets from the Riloff corpus and 26,168 from the SemEval-2018 corpus	F1-Measure of 0.87	Using word embedding is proven to be a good way to go
Majumder al. (2019)	CNN+GRU (Gated Recurrent Unit)) (DL)	3,994 self-sampled	F1-Measure of 0.87	This work is proving that sarcasm and sentiment is related
Shmueli al. (2020)	CNN, BiLSTM and BERT (DL)	Dataset 1: 49,804 self-scrapped tweets Dataset 2: 100,000 balanced tweets Dataset 3: 100,000 imbalanced tweets	F1-Measure of 0.86	Using reply tweets is proven to be able to yield a good result
Baruah al. (2020)	BERT, BiLSTM and SVM (DL)	6800 tweets from The Second Workshop on Figurative Language Processing	F1-Measure of 0.74	Including context in the form of the last utterance in a dialog chain slightly improved performance

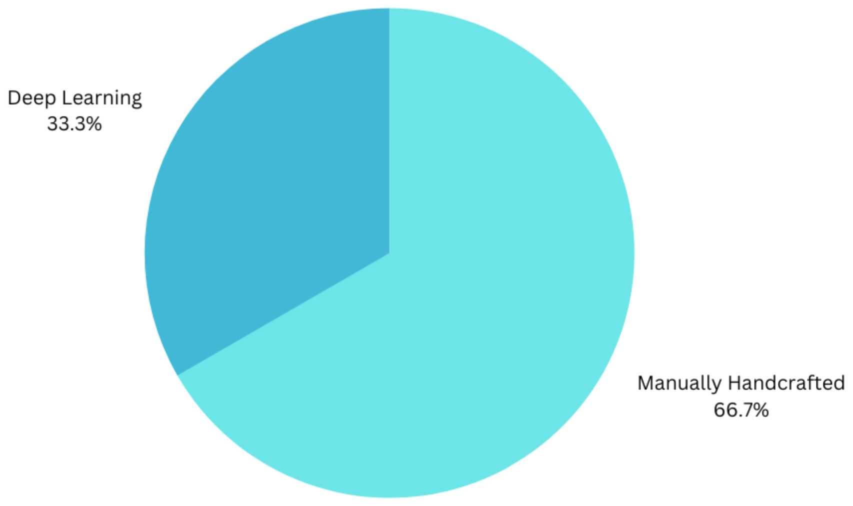


Figure 2.4: Methods from Existing Sarcasm Detection Works in Percentage

From Figure 2.4, it is concluded that only a small percentage of existing works use deep learning for sarcasm detection. It is very clear that even though deep learning is trending particularly in NLP works recently, sarcasm detection is left behind.

There are a few gaps or loopholes that need to be addressed and experimented in sarcasm detection as well as for other FL detection. Firstly, it is crucial to extract and justify the contexts of the FLs and use it to leverage the detection tasks. It is also crucial to look into an alternative perspective. For example, even though sarcasm is mostly perceived as negative, there are works that suggested it could be positive. Hence, this work takes the liberty to look closely into the context of positive sarcasm as well. It is also important to note that the work of sarcasm detection is a non-trivial matter and will contribute largely to the world with all the proof given in this literature review.

2.5 Metaphor

Text data analytics is plagued by the issue of sentences having different meanings than what is normally perceived. One of the reasons behind this is the use of metaphor. Metaphor is a literary concept based on resemblance. It is the substitution of one notion for another in virtue of some resemblance or analogy between them [95]. It is essentially an element of human cognition where resemblance or analogy is used by exploiting commonalities between two concepts. For example, the concept "competition" can be taken as another concept "sickness" or "disease" by exploiting the properties of the first concept. This then can be used in conversations i.e. "beat cancer" or "fight the virus together" [96]. Detecting and handling an FL like metaphor is crucial in an automated NLP system, mainly since they do not tell the literal meaning [58]. Metaphor has been studied a lot from the perspective of psychology, linguistics, sociology, anthropology, and computational linguistics [34]. This work looks into many of these perspectives to come up with the most optimal features in automatically detecting metaphor.

Traditional works in this domain have used manual handcrafted features based on its syntactic or contextual nature [97, 98]. More recent works resorted to deep learning techniques [99, 100], given its high reputation in NLP.

In this work, the whole feature extraction process started with a deep learning architecture extracting a feature set which is called deep features. It is then combined with the feature sets that are handcrafted according to the explanations in the subsequent chapters. Short sentences are used as the main resource of the experiments. In the real world, people usually use features like different tones, facial expressions and words to utter metaphoric sentences. In writings, basically only words are available [101]. Hence, this feature is fully utilized in this work.

2.5.1 History of Metaphor

Metaphor is an FL type that implies a comparison between two unlikely entities, distinguishing itself from simile only by the words "like" or "as" [102]. In this work, simile will not be investigated for the obvious reason that it is very close to metaphor, with the only difference of using those words.

In order to generate a single new creation that possesses the traits of both, a metaphor goes beyond a plausible, though maybe mundane, comparison to an identification or fusion of two objects. Many detractors view the creation of metaphors as a way of thinking that predates or eschews logic.

Although it is used frequently at all linguistic levels and in all kinds of language, metaphor is a vital component of poetry. Many words began as strong pictures. Along with single words, phrases and expressions that were formerly only metaphors abound in ordinary English. For instance, the expression "time flies" is frequently linked to the

Latin term “tempus fugit,” which first appeared in Virgil’s “Georgics” in 29 BC [103]. Tennessee Williams referred to the “Bird” as “Time” in 1959 when he gave his play the title “Sweet Bird of Youth” [104]. The usage of many words as the input for the entire experiment framework in this work is inspired by works like these. In this work, phrases up to three words (3-grams) are employed.

A metaphor in poetry can serve a variety of purposes, from the simple note of a similarity to the summoning of a swarm of associations; it might be an incidental beauty or the main idea and controlling picture of the poem.

The connection of two or more unrelated concepts results in a mixed metaphor, which can have an unintentionally comedic effect due to the writer’s disregard for word meanings or the comparison’s fallacy. As in William Shakespeare’s “Hamlet,” which was published in the Renaissance Era year of 1603, a mixed metaphor can also be employed quite well [105].

A mixed metaphor is a metaphor that is made up of the combination of a few metaphors in a sentence. It is different from the use of many metaphors at the same time where the metaphor instances come one after another. An example of mixed metaphor is a sentence from “Hamlet”: “...take arms against a sea of troubles”. “Sea of troubles” is already a metaphor because trouble is an abstract concept, which is not tangible. Adding “take arms against” in before it makes this sentence a mixed metaphor since taking arms against many troubles is not normal. This is what Shakespeare is actually trying to convey inside the metaphorical sentence. That Hamlet is crazy. This serves the inspiration in this work to use more than one context for the detection of metaphor and other FLs at the same time.

An FL that expressly alludes to one thing while discussing another is known as a metaphor. One of the most frequently used metaphors in English literature would be another example of metaphor. It originates from “As You Like It,” another one of Shakespeare’s masterpieces, which was first published in 1623 [106]:

“All the world’s a stage, And all the men and women merely players, They have their exits and their entrances”.

The world is not literally a stage, and most people are not literally actors and actresses playing parts, so this quote expresses a metaphor. Shakespeare makes the claim that the world is a theatre in order to explain the workings of the world and human conduct. He does this by drawing comparisons between the world and a stage.

A metaphor has two components, according to Richards and Constable [107]: the tenor and the vehicle. The subject to which characteristics are applied is the tenor. The thing whose characteristics are borrowed is the car. In the above illustration, “the world” is compared to a stage and given the characteristics of “the stage,” with “players” serving as the secondary tenor and “men and women” as the secondary vehicle. The phrases “metaphrand,” “metaphier,” “paraphiers,” and “paraphrands” were created by psychologist Julian Jaynes [108]. The terms “tenor” and “vehicle” from the previous piece are comparable to “metaphrand” and “metaphier,” respectively.

For instance, the metaphor “Pat is a tornado” uses “Pat” as the metaphier and “tornado” as the metaphrand. “Tornado” carries paraphiers like power, storm, wind, danger, menace, devastation, etc. as a metaphier. The metaphorical meaning of “tornado” is ambiguous; one person may interpret it to suggest that Pat is very destructive through the analogy of physical and emotional devastation, while a different person may interpret it to mean that Pat can spin out of control. In the latter instance, “spinning motion” has been rephrased as “psychological spin,” proposing a completely new metaphor for emotional irrationality—a possible description of a person but one that is unlikely appropriate to a tornado. This provides us with yet another source of inspiration for our work, where the subjects themselves may serve as the tenors.

It is clear from examination of the dataset that sentences using metaphors frequently have more than one subject or tenor. Additionally, metaphor and emotions go hand in hand.

The use of metaphors is relatively recent in modern European languages; it is, in theory, a post-Renaissance phenomena, claims linguist Anatoly Liberman [109]. The Latin term metaphora, which means “carrying over,” and the Greek word metaphora, which means “transfer (of ownership),” are the sources of the English word metaphor, respectively. As a result, even the word metaphor itself is a metaphor since it is derived from a Greek verb that means to “transfer” or “carry through.” When words, images, ideas, or situations are linked together to form a metaphor, they “carry” meaning from one to the next.

As was already established, similes and metaphors are frequently contrasted. A simile only implies a similarity by using terms like “like” or “as,” whereas a metaphor claims the items in the comparison are identical. Hence, similes are typically viewed as less effective than conventional metaphors [110]. We only pay attention to metaphor in its most traditional definition with all of its contextual linkages for the same reason. As can be seen below, there are numerous other ideas that can be classified as different forms of metaphor.

2.5.2 Types of Metaphor

- Allegory: An elaborate metaphor in which a story highlights a key aspect of the topic.
- Antithesis: A parallel grouping of words, clauses, or sentences used to contrast concepts in rhetoric.
- Catachresis: A mixed metaphor that is employed both on purpose and accidentally (a rhetorical fault).

- Hyperbole: An extreme exaggeration used to make a point.
- Parable: A lengthy metaphor used in narrative form to highlight or impart a moral or spiritual lesson.
- Pun: A verbal trick that gives a sentence many legitimate readings by using different definitions of the same word or its homophones, usually for comedic effect.
- Similitude: A prolonged simile or metaphor that includes a point of comparison, a reality component, and an image component.

2.5.3 Related work

Earlier works in metaphor detection are limited to finding certain syntactic or contextual features. For example, Jang et al. [98] claim that topic transitions between metaphoric sentences are different from literal sentences. They also claim that metaphor is usually expressed emotionally or in a thinking process. Hence, they extracted features based on these claims and trained a support vector machine for metaphor detection.

Klebanov et al. [111] extracted unigrams, part-of-speech, and topic models. They train a logistic regression classifier using this feature set for the detection task. From the results, it was the unigram feature that contributed the most, and topic model is the second most useful feature.

The next work in this group was by Tsvetkov et al. [112], they extracted features such as image words e.g. "torture" and "vengeance" and supersense phrases e.g. ""drinks gasoline". Then, they trained a random forest classifier using these features for the detection task. They also hypothesized that metaphors are more conceptual than lexical. It is also proven in their work that this hypothesis is true when their English-trained model can detect metaphors in other languages such as Spanish, Farsi, and Russian.

Turney et al. [97] hypothesized that metaphorical word usage has direct correlation with the degree of abstractness of the specific word's context. They came up with an algorithm that puts a score of abstractness or concreteness to each word in their lexicon. Then, they used these words to label whether the sentences are literal or not. Logistic regression algorithm is used for the final classification task.

Recently, researchers in the domain have relied more on deep learning methods for metaphor detection, given its recent good reputation in NLP. For example, the work by Do Dinh and Gurevych [113] used a Multilayer Perceptron (MLP) with one hidden layer which is shown in Figure 2.5. A large corpora trained on word embeddings is used as the input. Their best result was 0.56 for F1-measure.

Another work using deep learning architecture is done by Bizzoni and Ghanimifard [114]. They compared two deep learning learning architectures for the task of metaphor detection. One is a bi-LSTM model and another one is modified model based on the

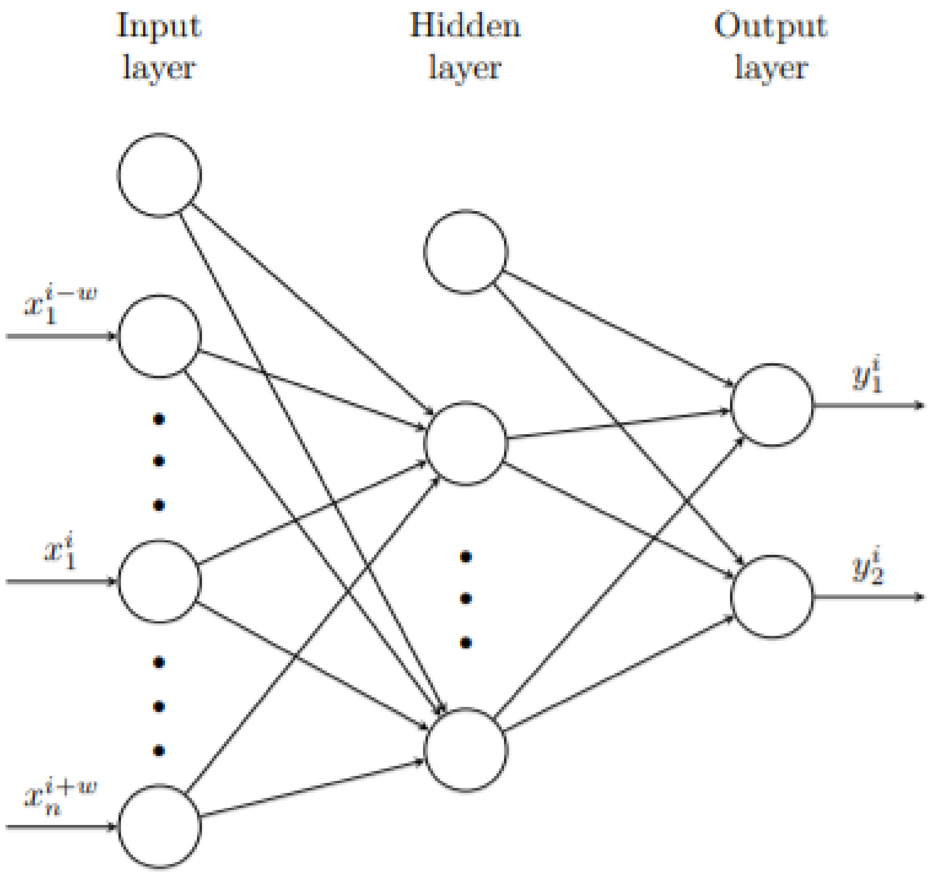


Figure 2.5: MLP architecture used by Do Dinh and Gurevych [113]

previously mentioned work by Do Dinh and Gurevych [113], shown in Figure 2.6. The input is also the same as the work from Do Dinh and Gurevych, which is a large corpora trained on word embeddings. Their best result was 0.75 for F1-measure, which is a slight improvement from the work mentioned previously.

The work by Swarnkar and Singh [115] claims that metaphors exist when the contrast between a word's general meaning and its contextual meaning is found. The input for the architecture is still word embeddings vectors pre-trained on the metaphor dataset used by the two previously mentioned works. The network architecture is based on forward and backward LSTM as shown in Figure 2.7 [116]. Their best result was 0.64 in terms of F1 measure, which is slightly lower than the work by Bizzoni et al [114].

Using lexicons is also regarded as another good way for this task [117]. Lexicons such as Wordnet [89] and Verbnet [118] are used in metaphor detection in the process of extracting features based on word frequency.

Table 2.2 below shows an overview of significant existing works by focusing on the methods used. The methods are broken into Manually Handcrafted (MH) and Deep Learning (DL). Since machine learning methods other than deep learning is seen in use

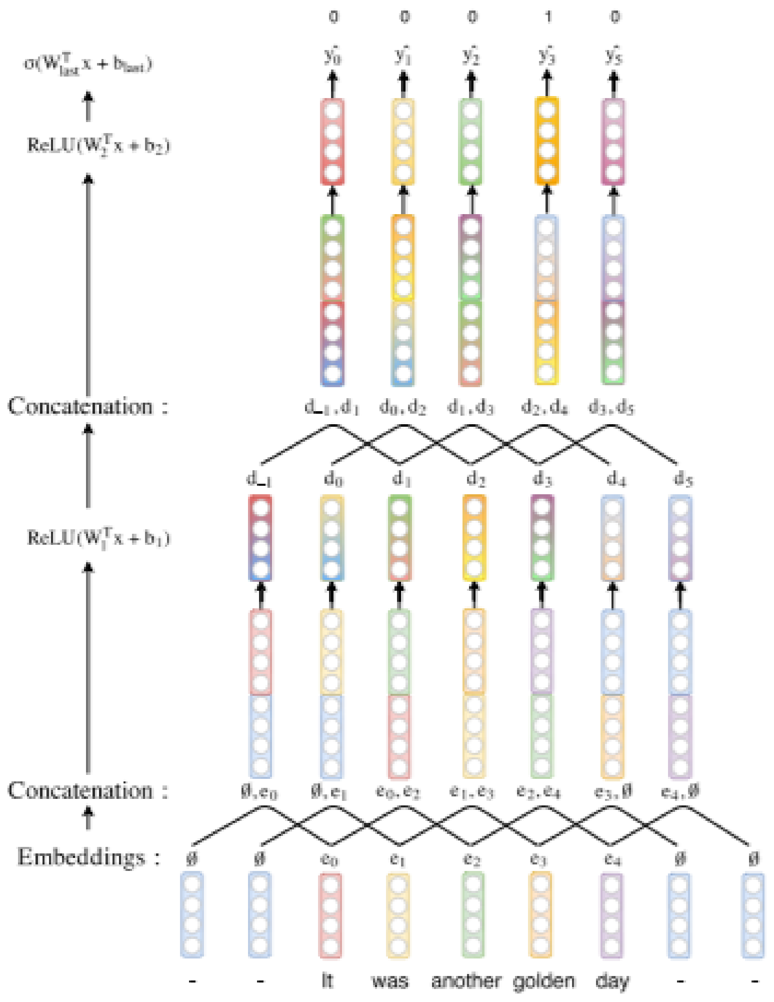

$\sigma(W_{last}^T x + b_{last})$
$ReLU(W_2^T x + b_2)$
Concatenation :
$ReLU(W_1^T x + b_1)$
Concatenation :
Embeddings :
0 0 0 1 0
$\hat{y_0}$ $\hat{y_1}$ $\hat{y_2}$ $\hat{y_3}$ $\hat{y_5}$
d_{-1}, d_1 d_0, d_2 d_1, d_3 d_2, d_4 d_3, d_5
d_{-1} d_0 d_1 d_2 d_3 d_4 d_5
$\emptyset, e_0$ $\emptyset, e_1$ e_0, e_2 e_1, e_3 e_2, e_4 $e_3, \emptyset$ $e_4, \emptyset$
$\emptyset$ $\emptyset$ e_0 e_1 e_2 e_3 e_4 $\emptyset$ $\emptyset$
- - It was another golden day - -

Figure 2.6: Modified MLP architecture used by Bizzoni and Ghanimifard [114]

only for classification tasks in the first category, it is combined with Manually Handcrafted for the reason that the feature extraction is done so. The results are mostly shown in the best F1-Measure since it is the harmony of precision and recall. However, other metrics are used when F1-Measure is not available. Figure 2.8 shows the corresponding percentage for Table 2.2.

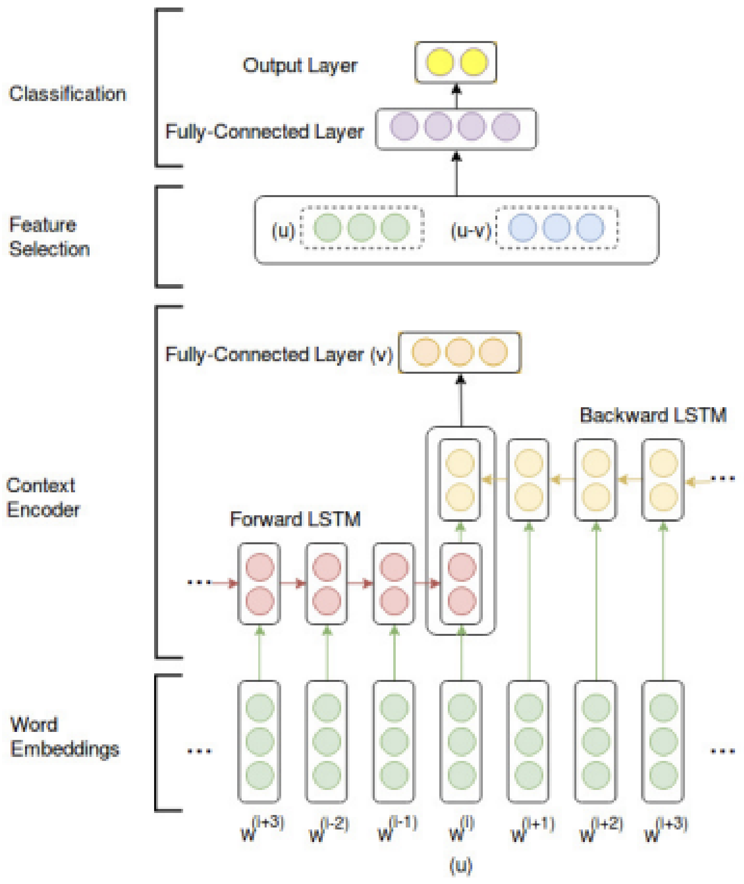


Figure 2.7: Modified LSTM architecture used by Swarnkar and Singh [115]

Table 2.2: Methods from Existing Metaphor Detection Works

Author	Method	Dataset	Result	Other Observation
Turney et al. (2011)	Weka tool (MH)	Corpus of Contemporary American English (COCA)	Accuracy of 79%	Readily generalizes to new words, outside of the training data
Klebanov et al. (2014)	Logistic regression (MH)	VUAmsterdam (117 fragments sampled from genres: Academic, News, Conversation, and Fiction	F1-Measure of 0.67	Inspiration for using unigrams
Tsvetkov et al. (2014)	Scikit-learn toolkit (MH)	TroFi: 3,737 manually annotated English sentences from the Wall Street Journal	F1-Measure of 0.86	Multilingual
Jang et al. (2015)	GibbsC++ toolkit (MH)	1,562,459 messages from discussion forums for an online breast cancer support group	F1-Measure of 0.79	Understanding context of metaphor is key to detecting it
Do Dinh & Gurevych (2016)	MLP (DL)	VUAmsterdam	MAE of 0.16	Created own corpus to help with the task

Author	Method	Dataset	Result	Other Observation
Rei et al. (2017)	Similarity Network (DL)	MOH (647 verb–noun pairs (316 metaphorical and 331 literal)) and Tsetkov (884 literal and 884 metaphorical pairs for training and 100 literal and 100 metaphorical pairs for testing)	F1-Measure of 0.81	Claimed to be the first to use deep learning for the task
Bizzoni & Ghanimifard (2018)	Bi-LSTM and Modified MLP (DL)	Self-annotated: 200 sets	F1-Measure of 0.75	Compared two different deep learning architectures
Swarnkar & Singh (2018)	Modified LSTM (DL)	VUAmsterdam	F1-Measure of 0.64	Used cost-sensitive learning by re-weighting examples and baseline features
Mao et al. (2018)	CBOW and Skip-gram (DL)	Self-scrapped instances from Wikipedia and MOH	F1-Measure of 0.75	Demonstrated that using a candidate word output vector instead of its input vector yields better results

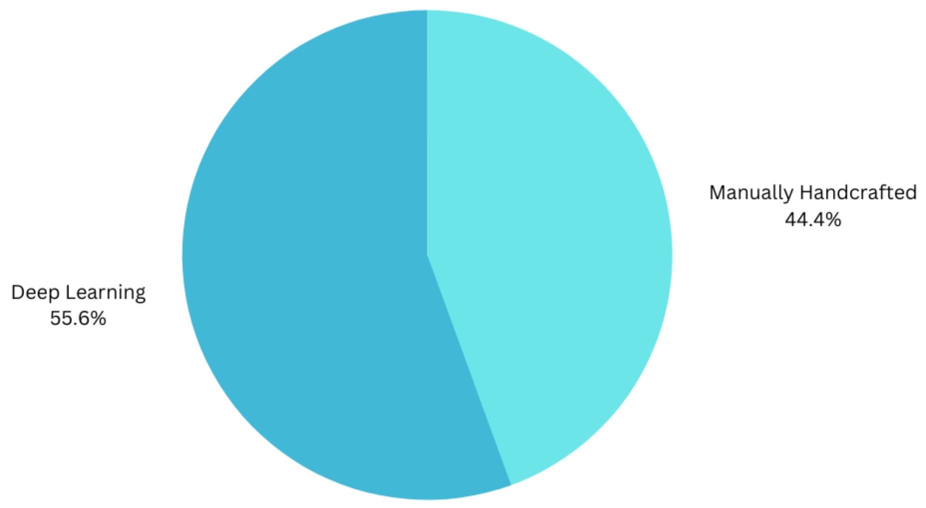


Figure 2.8: Methods from Existing Metaphor Detection Works in Percentage

From Figure 2.8, it is concluded that a considerable amount of percentage from existing works are using deep learning for metaphor detection. It is very clear that even though deep learning is trending particularly in NLP works recently, metaphor detection is left behind.

Apart from the obvious lack of deep learning use in the existing works, this literature review finds multiple ways to run experiments on metaphor detection as well as the other FLs: firstly, input to detection tasks is not restricted to single words. Secondly, more than one context can happen in a single instance, more so in a dataset with paragraph-type instances as used in the satire detection part of this work. Thirdly, subjects can be used as tenors or figuratively (specific only to metaphor). Fourthly, FL is closely related to emotions. Then, FL is multi-lingual and not many work are done on this. Finally, work on metaphor detection can be improved using deep learning.

2.6 Satire

Just like sarcasm detection which can help in sentiment analysis tasks, satire detection can help in fake news detection tasks. Fake news is not a technical term. Rather, it is a new term born from modern journalism. Its vastness includes false stories, factual errors, satire and sometimes just simply a story that a certain person or party does not like. Often, it is difficult to identify because of this vastness of definition [119]. For this problem, this work tries to solve only one part of it: satire. This is also because satire falls under the topic of this work, which is to detect FL. Additionally, satire is not as new as fake news. This makes the pool of existing research on the subject vast and open.

According to the Merriam-Webster dictionary, satire came into English at the beginning of the sixteenth century, and the meaning of the word has not strayed very far from its original sense. The initial uses were primarily applied to poems, and the term now has a broader applicability. Satire has a semantic and etymological overlap with both farce and lampoon [119]. Farce (”a light dramatic composition marked by broadly satirical comedy and improbable plot“) came into English as a synonym for ”forcemeat“, meaning ”finely chopped and highly seasoned meat or fish that is either served alone or used as a stuffing“. Lampoon (”a harsh satire usually directed against an individual“) is thought to come from the French ”lampons“, meaning ”let us guzzle“. The term ”satire“ is believed to trace back to the Latin ”satur“, meaning ”well-fed“, and was used in the phrases like “lanx satura,” meaning “a dish full of many kinds of fruit”. Though these words seem far removed from the definition of satire, they were used by ancient Roman critics and writers to refer to what we know as satire today.

2.6.1 History of Satire

In the seventh century BC, satirical works were already having profound effects on people [120]. Greek poets and authors such as Archilochus and Aristophanes helped build the foundations for all of Western comedy [121]. Their work like poems and plays are often harsh and humorous. They use these works to critique the society and certain individuals. These kind of works would have been considered satire today.

The term ”satire“ is based on the Latin word ”satur“, because Latin authors are responsible for adapting the earlier Greek form of the genre into what we are more familiar with today [122, 123]. In fact, many satires are still categorized as Horatian or Juvenalian to this day by their resemblances to works of the old Roman poets. Those resembling the works of Horace are known to be learned and witty, often using subtly sarcastic wordplay even at the author’s own expense. Those who imitate Juvenal, on the other hand, are typically considered openly harsh in their mockery and ridicule, occasionally even downright dark.

After the Roman satirists, there is no significant satirical works that can be found for about a millennium. As Europe emerged, satirical elements began to reappear throughout medieval manuscripts. One good example of this is the Spanish novel Don Quixote [124], which satirized the overly romanticized lives and literature of the previous centuries and changed views on formal chivalry forever. The revival of European art and literature at this time is called the Renaissance. Even though Shakespeare is the most popular Renaissance writer, he rarely used satire. However, all others were very fond of using it [125].

Modern Satire

Today's audience favors Horatian more than Juvenalian satire. This can clearly be seen by the attention that is received by Horation works such as "Gulliver's Travel" [18]. Juvenalian works such as "Animal Farm" [126] even though also popular, pales in comparison to their Horatian counterparts.

Even though there are still many recent written works on satire, other media are also starting to use it. Satirized news reports like The Onion have become a normalcy [127]. The same goes to satirical movies and drama series. This has started from as early as the days of black-and-white antics of Charlie Chaplin, where he mimicked the attributes of Adolf Hitler [128]. Even animated productions like South Park are taking the chance to entertain and also educate their audiences on the social and artistic issues happening today, using satire [129].

Today satire is used a lot in politics. This includes political cartoons and jokes [130, 131]. Political cartoons appear both in print and online. A common structure for a political cartoon is to have one large panel, with a drawing that mocks the physical features of an elected official or any newsworthy figure. Political jokes are also used a lot by the comedians on TV.

2.6.2 Types of Satire

Satire remains a powerful tool in contemporary culture. Film and television, in particular, have been important vehicles for satire over the past several decades. There are three main types of satire, each serving a different role.

Horatian

Horatian satire is comic and offers light social commentary. It is meant to poke fun at a person or situation in an entertaining way.

"Gulliver's Travels", written in the eighteenth century by Jonathan Swift, is a perfect example of Horatian satire in literature [18]. The work is a spoof of the kind of travelogues that were common at that time. Through his invented narrator, Gulliver, Swift takes aim at travel writers, the English government, and human nature itself. The books which served as the inspiration include ”Jupiter’s Travel“ where the author wrote about his travel around the world on his motorcycle meeting interesting people from his point of view [132]. Another book is ”The Snow Leopard“ which follows the author’s journey through the Himalayas which is somewhat spiritual [133]. ”In Patagonia“ is another book that talks about a new interesting place that is deemed as fascinating by the author [134].

These books which were all written in the eighteenth century, carry one of the most popular narratives at the time, travelogue. It is following the author’s journey through places which are deemed interesting. However, these journeys, places, people and creatures that they met along the way are still existing ones. The author of Gulliver’s Travel took all these as an inspiration to write his satirical book where all the components in it do not really exist i.e. giants and tiny people.

Today, The Onion is regarded as a popular satirical online news site that embodies Horatian satire [127]. It is also one of the sources used for the dataset in this research work.

Juvenalian

Juvenalian satire is dark, rather than comedic. It is meant to speak truth to the authority, without getting into too much trouble.

”Animal Farm“ is a perfect example of Juvenalian satire [135]. It tells a story of a group of animals trying to take control of a farm by toppling down the original controller, which is the owner of the farm. The animals use whatever necessary techniques they could think of, including the use of conspiracies and politics. The novel’s intended target is communism and Stalin-era Soviet Union. Animal Farm is also an allegorical satire: it can be read as a simple tale of farm animals, but it has a deeper political meaning. A modern-day example is the animated television show South Park, which juxtaposes biting satire with juvenile humor [129].

Menippean

Menippean satire casts high judgment on a particular belief, such as racism. It can be comic and light, much like Horatian satire—although it can also be as stinging as Juvenalian satire.

Lewis Carroll’s ”Alice’s Adventures in Wonderland“ is one of the best examples of Menippean satire in literature [136]. It tells the story of how Alice found herself in a different world than what she and everyone she knows is used to. The main characters are also abnormal in comparison to the people the world is used to.

The novel pokes fun at upper-class intellectualism but does it with a distinct sense of humor. The ridicule is there, but it is good-natured in spirit. A modern-day example is the television show Saturday Night Live, which has carried a long tradition of poking fun at elected officials [137].

In the FL detection space, there are not many works focusing on satire as compared to the more popular ones such as sarcasm and metaphor [138]. This is unfortunate seeing it is rich with history and is also very timeless. It can be relevant for all time since it mocks events that are either happening in the past, present or future. This work takes advantage of its rich definitions as stated above, by observation through experiments and also by taking some cues from existing related works. The most discriminatory features are recognized, extracted and used for the final model.

2.6.3 Related work

Earlier works in satire detection are limited to finding linguistic or sentiment features. For example, Rubin et al. [38] claim that satire is when someone intentionally leave cues to reveal deceptiveness. This is for the reason that humans in general find it challenging to recognize deceptions. They say there are three reasons behind this challenge:

- Humans tend to assume the information they receive is true
- Humans are generally gullible
- Humans have confirmation bias (they only see what they want to see)

Hence, they extracted word level features, humor-related features and also opposing script features for the satire detection task. Bi-normal separation (BNS) feature scaling was used as the feature weighting strategy. All of the features were then classified using SVM. The highest result is 0.87 in terms of F1-measure.

Another work in this domain by Burfoot and Baldwin [139] is the main inspiration for the previously mentioned work. The focus of this work was to extract indicative words from the headline, slang and profanity used in an article. They used both binary feature weighting and BNS. Binary feature weighting only allows them to mark each of the instances in the dataset as only having or not having satire. This is an issue for this type of work, as proven in our experiments in the latter chapters. The instances that have more main features should have been weighted more. Satirical clues should have not just give a ”0“ or ”1“ as in its feature extraction process. In this dissertation, binary weighting for main features is changed to frequency weighting. They used SVM for the classification. Their best result was 0.80 in terms of F1-measure. It is important to note that this is slightly lower than its successor above. Binary weighting was omitted entirely by this successor as it did not help with the performance of their final experiments.

Another work in this domain is by Reganti et al. [140]. They opted to use features related to sentiment as well as linguistics. Sentiment features include punctuation, emoticon, slang word, acronym and interjection. An ensemble classifier that include Logistic

Regression (LR), Random Forest (RF) and Decision Tree (DT) was used for this work. Their best yielded result was 0.79 in terms of F1-measure. This is slightly lower than the work by Burfoot and Baldwin [139] even though this work is far more recent. The use of ensemble classifier in their work inspired the investigation of all the machine learning classifiers for this work. This work treats the final classification as an exploratory experiment to find the most optimal strategy.

Other researchers used machine learning algorithms to support their handcrafted features. For example, one work combined psycholinguistic features that were extracted using LIWC with common machine learning algorithms such as Sequential Minimal Optimization, BayesNet and J48 [138]. WEKA tool was used for the machine learning classification purpose. The dataset used in this work contains multiple tweets in Spanish. The whole framework can be seen in Figure 2.9.

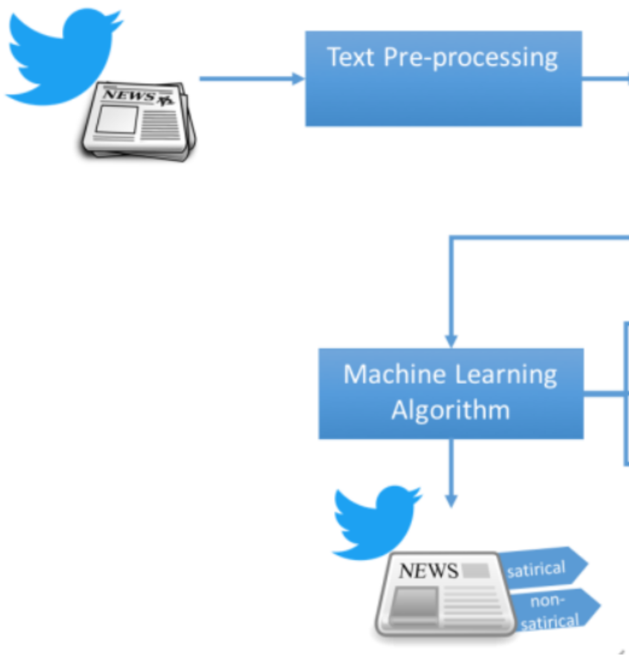

Text Pre-processing
Psycholinguistic Feature Extraction
Machine Learning Algorithm
SMO
J48
BN
NEWS
satirical
non-satirical

Figure 2.9: Framework used by del Pilar Salas-Zárate et al. [138]

Another work that uses a machine learning algorithm as an addition to the handcrafted features is the use of multiple datasets to prove its point [141]. The datasets range from the collection of product reviews and tweets news articles. SVM was used as the classification algorithm to classify the emotional and sentimental features. The architecture can be seen in Figure 2.10.

Machine learning is also used in another work [142], without any need for handcrafted features. They hypothesized that one of the most important features of satire detection is the source, such as the website's name. The architecture of the model is shown in Figure 2.11. The model does not only output a binary classification of satire or not satire, but

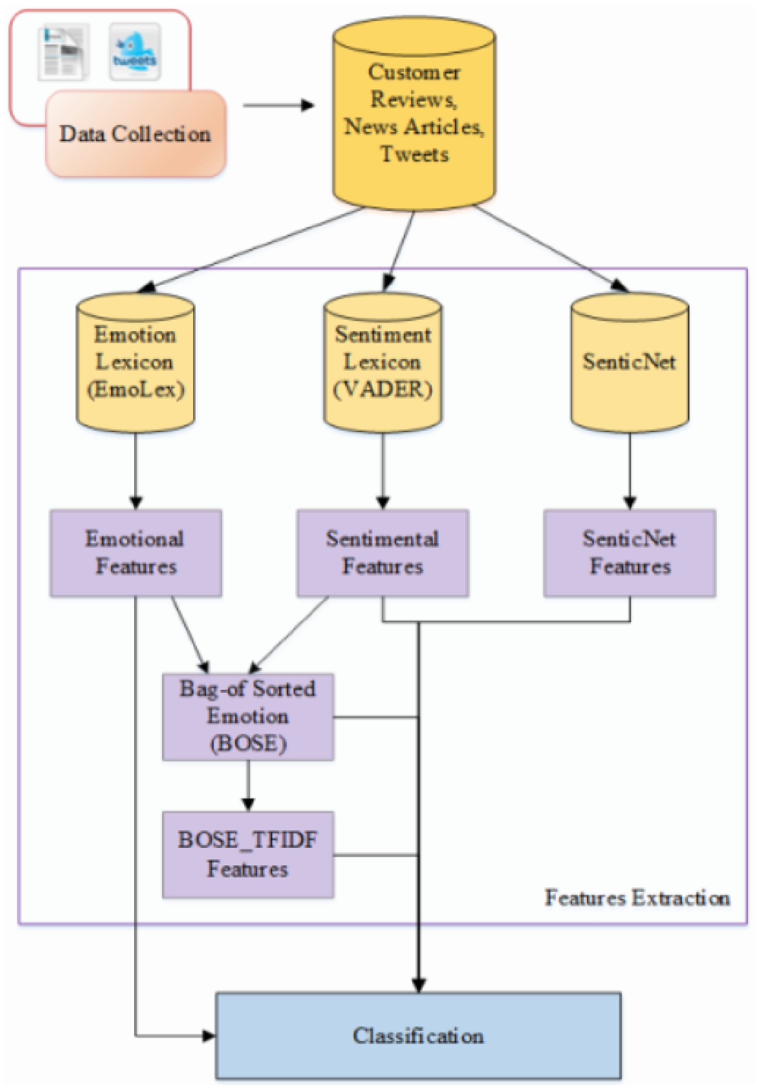


Figure 2.10: Architecture of the Model used by Thu et al. [141]

also a multi-class classification of the sources. This work used bidirectional Long Short-term Memory [143] and self-attention [144] layers for both the satire detection and source detection tasks simultaneously.

Recently, researchers in the domain have relied more on deep learning methods for satire detection, given its recent good reputation in NLP. For example, the work by Zhang et al. [145] used a language model (LM) with 2 LSTM which is shown in Figure 2.12. They used a dataset that was created by another work that is explained next. The best result was 0.90 in terms of F1-measure. This is higher than all of the results yielded by the previous works mentioned previously.

Another work using deep learning architecture was done by Yang et al. [146]. They combined encoders such as character-level encoder using CNN and word-level encoder using gated recurrent unit (GRU) with features such as linguistic, psycholinguistic, writing style and structural features. Their overall proposed model is a 4-level hierarchical neural network model as shown in Figure 2.13. Their best result was 0.91 in terms of

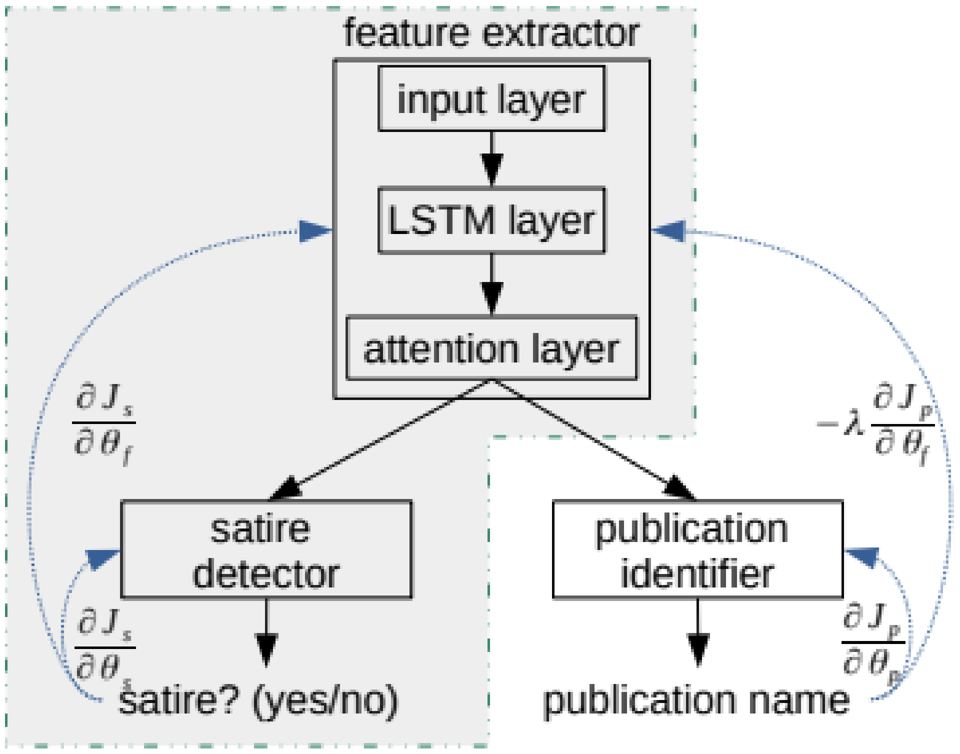

feature extractor
input layer
LSTM layer
attention layer
$\frac{\partial J_s}{\partial \theta_f}$
$-\lambda \frac{\partial J_p}{\partial \theta_f}$
satire detector
publication identifier
$\frac{\partial J_s}{\partial \theta_s}$
$\frac{\partial J_p}{\partial \theta_p}$
satire? (yes/no)
publication name

Figure 2.11: Architecture for the Model used by McHardy et al. [142]

F1-measure, which is slightly higher than the more recent work mentioned above.

Table 2.3 below shows an overview of significant existing works by focusing on the methods used. The methods are broken into Manually Handcrafted (MH) and Deep Learning (DL). Since machine learning methods other than deep learning are seen in use only for classification tasks in the first category, it is combined with Manually Handcrafted for the reason that the feature extraction is done so. The results shown in the best F1-Measure since it is the harmony of precision and recall. Figure 2.14 shows the corresponding percentage for Table 2.3.

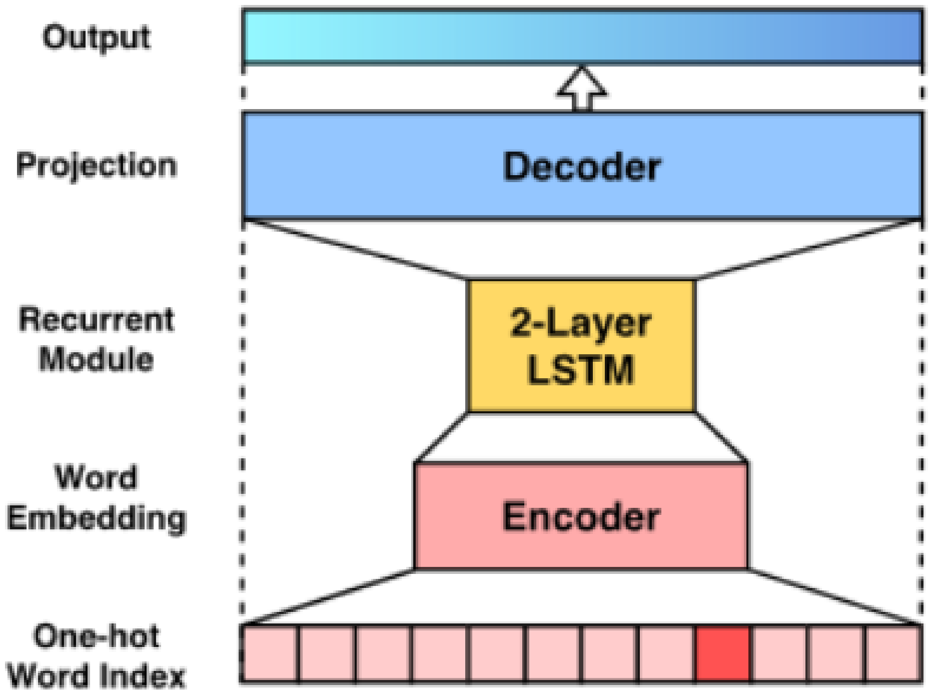


Figure 2.12: Language Model used by Zhang et al. [145]

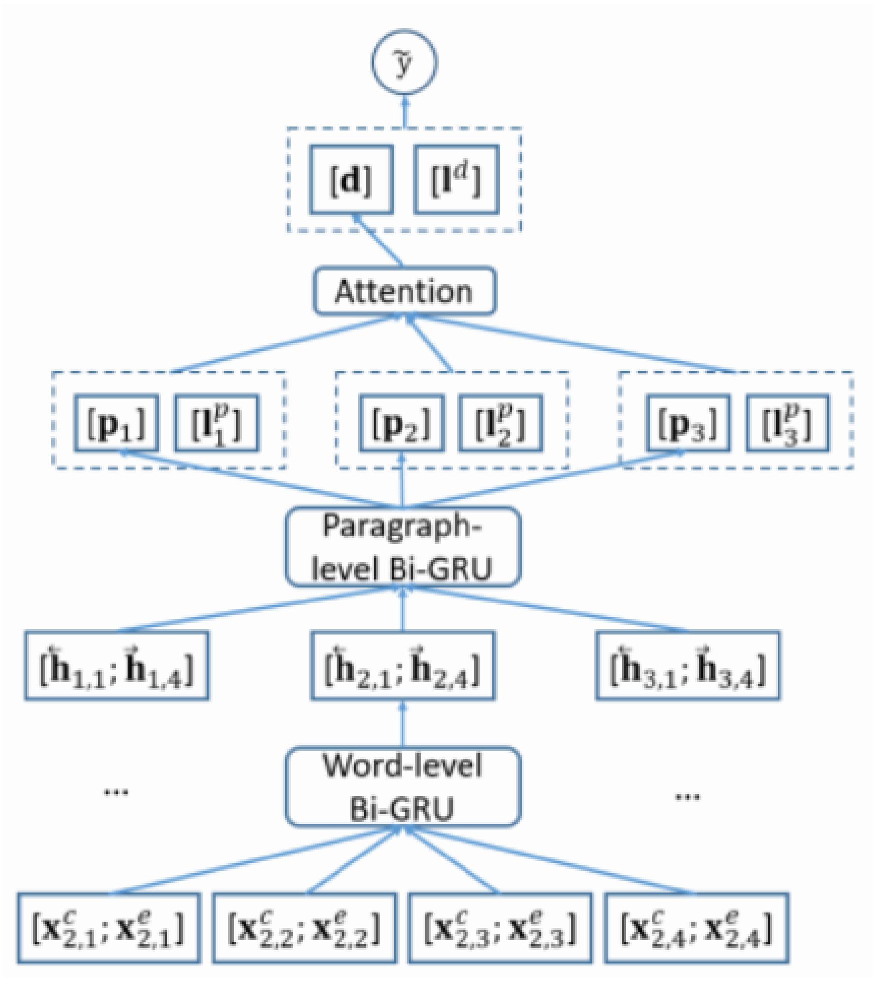


Figure 2.13: 4-level Hierarchical Neural Network Model used by Yang et al. [146]

Table 2.3: Methods from Existing Satire Detection Works

Author	Method	Dataset	Result	Other Observation
Burfoot & Baldwin (2009)	SVM (MH)	4000 newswire documents and 233 satire news articles from English Gigaword Corpus	F1-Measure of 0.79	Claimed to be the first work to detect if a news is true or satirical
Barbieri et al. (2015)	SVM (MH)	11,064 tweets from 4 Twitter acconts: El Mundo Today, El Jueves, El Mundo, and El Pais	F1-Measure of 0.85	Detection for Spanish language
Barbieri et al. (2015)	SVM (MH)	33,192 Tweets from 12 Twitter accounts	F1-Measure of 0.76	Detection for English, Spanish, and Italian languages
Rubin et al. (2016)	Scikit-learn and TFIDF (MH)	360 news articles from The Onion, The Beaverton, The Toronto Star, and The New York Times	F1-Measure of 0.87	Using satire to detect misleading news
Reganti et al. (2016)	Logistic Regression (MH)	4233 newswire documents, 1,254 Amazon product reviews, and 6000 tweets	F1-Measure of 0.79	Detecting satire on three different platforms: New articles, product reviews and tweets

Author	Method	Dataset	Result	Other Observation
Frain & Wuben (2016)	BOW (MH)	3411 news articles self-scrapped from 15 news outlet	F1-Measure of 0.93	Created a language resource for satire in news articles: SatiricLR
Goldwasser & Zhang (2016)	Novel Narrative Representation Graph (MH)	12, 146 self collected news articles from CNN, The Onion and Burfoot	F1-Measure of 0.8	Claimed to detect satire using common-sense
Yang et al. (2018)	CNN and GRU (DL)	110,806 news articles gathered from 14 news outlets	F1-Measure of 0.91	Tried a few machine learning models until finally decided to propose model
Zhang et al. (2020)	LSTM (MH)	Yang	F1-Measure of 0.93	Incorporated a feature set derived from imagination called Imaginative Content

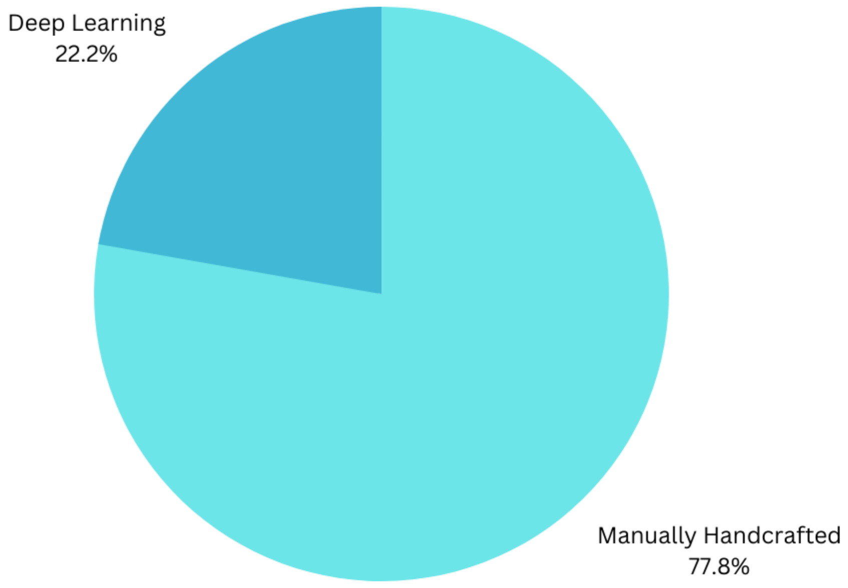


Figure 2.14: Methods from Existing Satire Detection Works in Percentage

Figure 2.14 shows the corresponding percentage for Table 2.3. From this figure, it is concluded that only a small percentage of existing works are using deep learning for satire detection. It is very clear that even though deep learning is trending particularly in NLP works recently, satire detection is still left behind. It is also found that satire types are very clearly defined in existing literary works. Hence, it can be used for feature extraction.

2.7 Summary

As a summary of the literature review, the three FLs are both similar and also very distinct in context. The similarities include that the context itself must be understood first before any experiment can be run on each of them. It is also understood that there can be more than one context in an FL sentence. The second is that the detection of each of them makes a substantial contribution since it is largely used around the world. Third is that FL detection is not limited to word-by-word processing. This is because most of the FLs use phrases instead of just words. This is also rarely looked into in the existing works. All the FLs are also found to be very closely related to human emotions. An obvious finding is that not many existing works on FL detection have used deep learning.

There are also smaller uncharted territories that were found in this literature review process such as there has been no work on the detection of positive sarcasm, even though it has been discussed in fields other than computation. Another one is that subjects can be used as tenors or figuratively as in the case of metaphors. The final one is that for some of these FLs, the smaller categories or types are very clearly defined. This can be seen in the case of satire above. Hence, these types can also be used for feature extraction.

Chapter 3

Research Methodology

3.1 Introduction

This chapter presents the overall methodology with detailed framework for the proposed figurative language detection with machine learning in a text setting. By considering the contexts in all of the available figurative languages, the proposed approach will and only pick up the essential signals and improve the performance.

3.2 Figurative Language Detection Framework

The proposed method is shown in Figure 3.1. The whole process consists of Data Acquisition, Data Preprocessing, Figurative Language Type Detection, Classification and Evaluation. Several lexicons are also used to assist the processes. The lexicons are also going to be different from one FL detector to another, apart from the stopwords lexicon which is used to remove the stopwords from all of the datasets. The details of each part are given below.

3.2.1 Data Acquisition

All the datasets are split into 80 percent training and 20 percent testing sets. This is a common split in terms of training and testing for machine learning experiments [147]. The datasets used in this work can either be downloaded from a website or provided by the original authors of the works that this work is comparing. These datasets are explained in their respective detection chapters.

3.2.2 Data Preprocessing

Data preprocessing is a crucial step to remove all the noises from datasets [25]. It is the process of eliminating the components that will not give any value to the experiments.

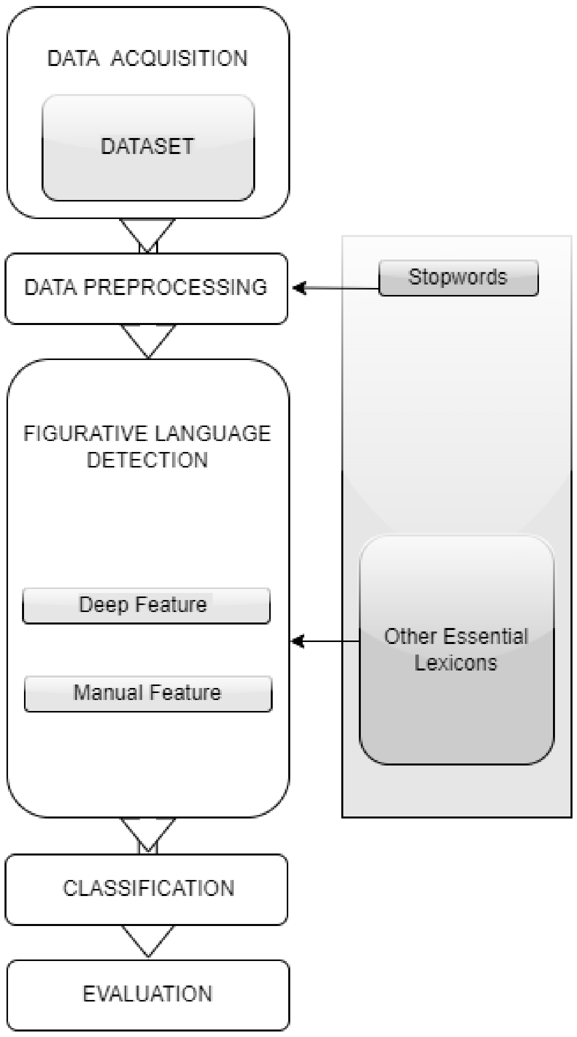


Figure 3.1: Overall framework of the work

This process is aided by the Matlab tool. Four preprocessing techniques have been chosen to fulfill this:

Firstly, all the text are changed to their lowercase counterparts. If they are already in lowercase, nothing is changed. This is to guarantee that each and every instance of the same word is taken as one word instead of multiple words.

Secondly, all the stopwords are omitted. Stopwords are words that do not carry any meaning. For example, the words ”but“, ”and“, ”he“, ”she“ and ”it“. This is to guarantee that no word that is without any meaning is included in the process.

Then, all punctuations are omitted. Many of the existing works in this domain use punctuation as a feature [10, 19, 27]. However, since this work is focusing more on the context of the FLs, punctuation is omitted.

Finally, lemmatization is performed on all the words. Lemmatization is the process of changing the words into their general root form e.g. ”building“ to ”build“. This is for the same reason as making all the words turn to the lowercase counterparts. It is to guarantee that each same words are taken as one word.

3.2.3 Deep Feature

Existing researchers in this domain are starting to use deep learning as their main strategy after its recent successes in NLP. Deep learning is now known to have the strength to automatically detect good features for NLP tasks. These features are called deep features. Deep learning has revolutionized the way NLP is analyzed and processed. As shown in the literature review, most existing works in this domain have benefited greatly from two architectures of deep learning. This due to their high performance with less need of engineered features. The two deep learning architectures are Convolutional Neural Network (CNN) [148] and Recurrent Neural Network (RNN) [149].

In this work, CNN is chosen to be the architecture. It is hierarchical based [148] whilst RNN is sequential based [149]. A work by Vu et al. [150] found that CNN extracts the most informative N-grams in comparison to RNN. A few other existing works also support CNN over RNN for classification of sentences [151, 152]. Dauphin et al. [153] further argue that a fine-tuned CNN can also model long context dependency, getting new state-of-the-art in language modelling above all RNN competitors. Another existing work working on sentence classification created a very basic CNN architecture which managed to yield good performance [154]. The architecture used here is shown in Figure 3.2. These are only for layers in this architecture: input layer, convolutional layer, maxpooling layer and fully connected layer with dropout and softmax classification.

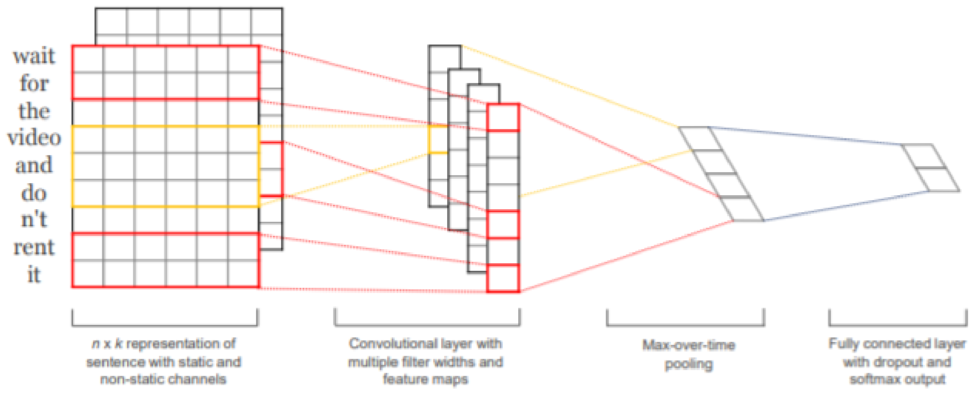

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 3.2: CNN architecture used by Kim [154]

In addition, another work [155] achieved better performance of CNN than RNN for answer selection, another NLP task. To fairly work the capability of different deep learning architectures, the following guideline is created:

- Always train from scratch.

- Always train using a basic setup without complex techniques such as batch normalization.
- Search for optimal hyperparameters for each task and each model separately.
- Investigate the basic architecture and utilization of each model.

A similar architecture was previously proposed in another work [156] and also managed to yield a good result. This work has used three out of four of the guidelines above in order to achieve good performance. Batch normalization is used since it is not as complex now as it was then.

All the layers used in this work have specific functions. Batch normalization can reparametrize the underlying optimization problem to make it more stable [157]. Dropout prevents co-adaptation of hidden units by randomly dropping out a proportion of hidden units during forward-backpropagation. Maxpooling is applied over the feature map and takes the maximum value as the feature corresponding to this particular filter. The idea is to capture the most important feature for each feature map. This pooling scheme naturally deals with variable sentence lengths. According to another existing work that performs an empirical evaluation on the effect of varying hyperparameters in CNN architectures, max-pooling always beats average pooling [158]. Finally, the activation function ReLU is used for its simplicity. It outputs 0 when the variable is below 0, and a linear function when the variable is equal or above 0 [159].

In short, the CNN architecture in this work consists of input representation, mandatory convolutional layers, batch normalization layers, ReLUs and dropout layers which are repeated three times over three graph lines for optimality. The three lines are ended by one maxpooling layer for each, before they are joined again by a depth concatenation layer. Fully connected layer at the end is added to extract the features.

In this work, the CNN architecture is proposed to extract the first feature set. This architecture is not too simple and not too complicated. The extracted set consists of about 10 deep features for each FL type. 10 is chosen as the number of features to match the approximate number of manual features from each FL type. This is depicted in the CNN part from Figure 3.4 and further shown in detail in Figure 3.5. The specific CNN setting for each of the FL types are thoroughly explained in the subsequent chapters.

As shown in Figure 3.4, the deep features extractor (CNN) is a large part of the overall architecture of all the FL Detectors. It is also assisted by the word-embedding technique FastText as a way to convert tweet sentences into feature vectors as input. The details are described below.

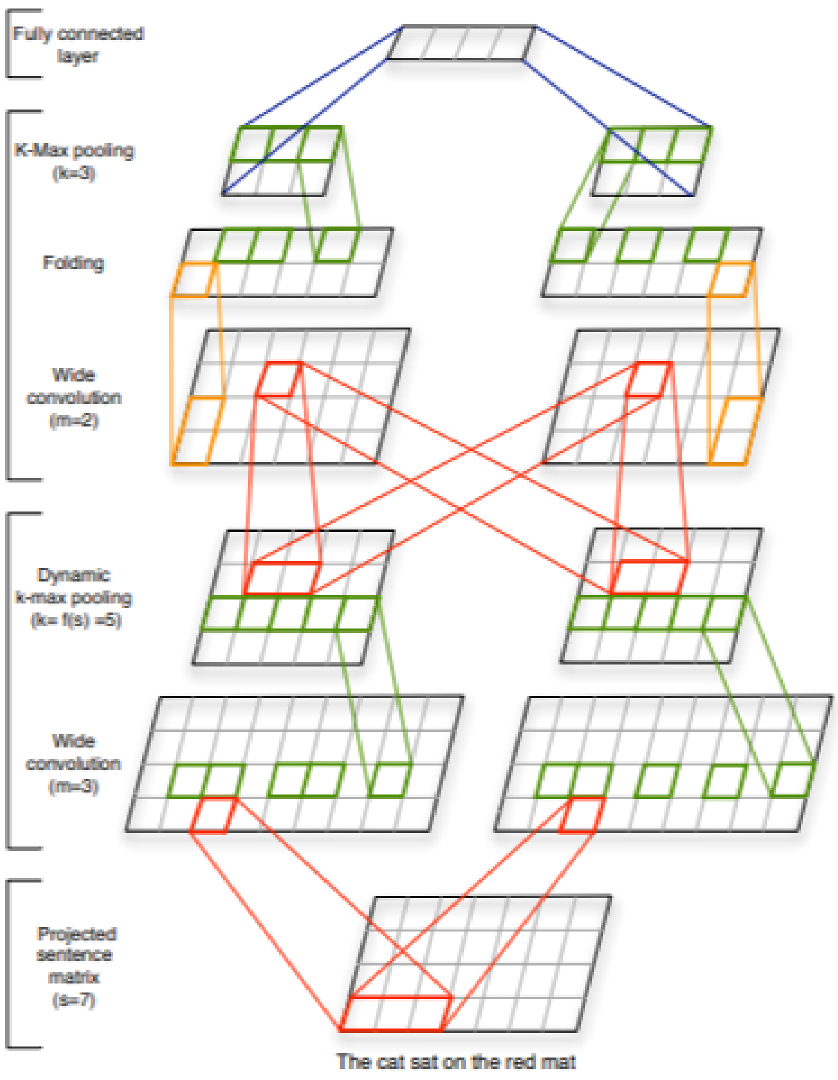

Fully connected layer
K-Max pooling (k=3)
Folding
Wide convolution (m=2)
Dynamic k-max pooling (k= f(s) =5)
Wide convolution (m=3)
Projected sentence matrix (s=7)
The cat sat on the red mat

Figure 3.3: CNN architecture used by Kalchbrenner et al. [156]

FastText

We have decided to use FastText [160] as our word embedding technique instead of the commonly used Word2Vec [161]. This is because FastText breaks every word into N-grams instead of using individual words as with Word2Vec. Then, the words are fed into a neural network.

FastText feeds the words into a neural network in the form of unigram, bigram and trigram. For example, the word “human” is broken into “hum”, “uma” and “man” as a trigram for the word ”human“. The vectors for ”human“ will be the total of all the broken N-grams. Artificial Neural Network (ANN) is used as the training method.

The output is a word-embedding vector for all the broken N-grams in the training dataset. Hence, FastText gives a better representation even for the rare and misspelled words. This makes it very effective for social networks analysis. Since the same word

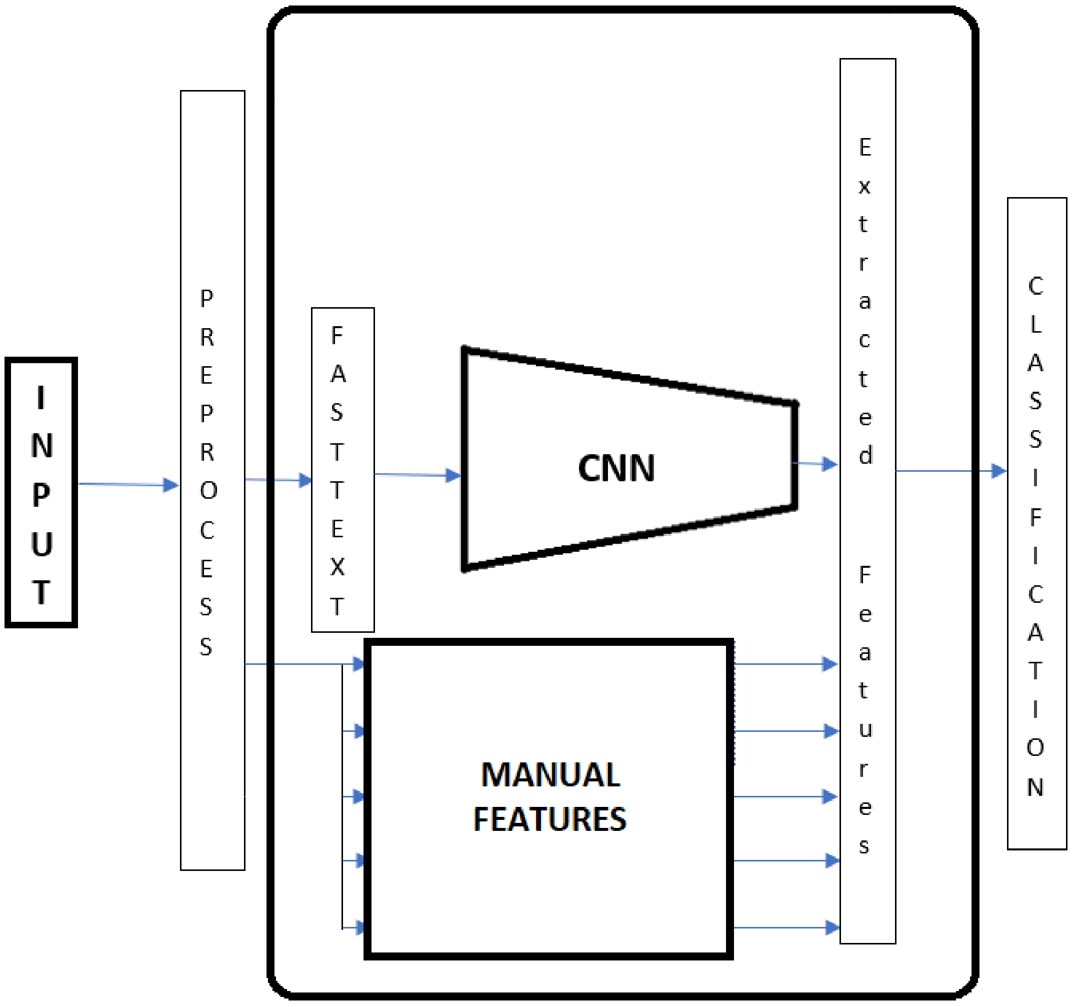


Figure 3.4: Overall architecture of the work

embedding technique is used for the subsequent FL detection, this part is not repeated in the subsequent chapters.

Convolutional Neural Network(CNN)

For the purpose of extracting the deep features, a CNN architecture as shown in Figure 3.5 is used. This is the detail of the CNN part from the overall architecture shown in Figure 3.4. This same architecture is used in all the FL detection in the subsequent chapters for the deep feature extraction part, with slightly different settings. Hence, Figures 3.4 and 3.5 are not repeated exactly anywhere else in this work. The variations are explained for each FL type detection.

The CNN architecture is shown in a top-down manner starting from the start (top) to the finish (bottom) node. A 1-gram (or unigram) is a one-word sequence. For example, the sentence ”I love Artificial Intelligence“ would be broken into: “I”, “love”, “Artificial”, and “Intelligence”. A 2-gram (or bigram) is a two-word sequence of words, for example, “I love”, “love Artificial”, and “Artificial Intelligence”. And a 3-gram (or trigram) is a

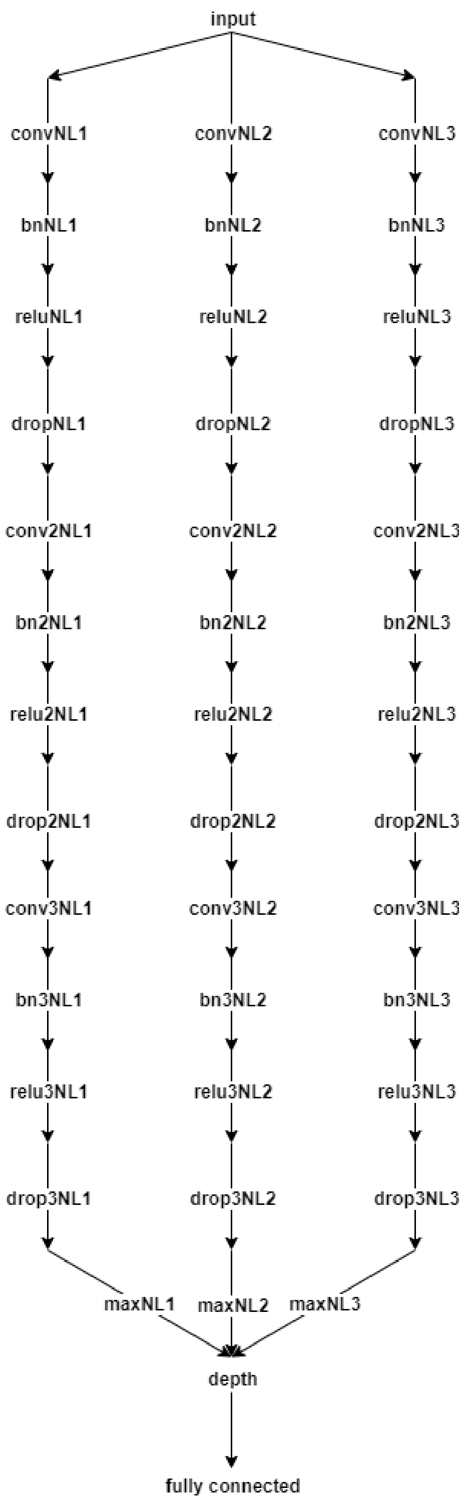


Figure 3.5: CNN architecture used in the FL Detectors

three-word sequence of words, for example, “I love Artificial”, and ”love Artificial Intelligence”. The architecture uses up until only trigrams because FL usually uses phrases up until only three-word sequence [4, 5]. The followings are the abbreviations and the respective expanded terms for the CNN:

- NL: N-gram Length.
- conv: Convolutional Layer
- bn: Batch Normalization Layer
- reLu: Rectified Linear Activation Unit Layer
- drop: Dropout Layer
- max: Max Pooling Layer
- depth: Depth Concatenation

The focus of this CNN architecture is the same as any other CNN architecture, which is the convolutional layers. This is because this layer produces the feature maps. Then, batch normalization layers are used to re-center and re-scale the input data to improve the speed, performance, and stability. Then, the activation functions are included to inspect the input data. Then, dropout layers are used to increase the generalization, validation accuracy and to avoid overfitting. The minimum 0.2 dropout is used in this case. Finally, the max pooling layer is included for the final vote.

The initial input vector size is set to the specific need of the FL type. These are the vectors created by FastText earlier in the process. These vectors are created according to N-gram lengths.

Then, these vectors go through three graph architectures. Each of these graph architectures begins to run concurrently until they meet again at the concatenation layer. A fully connected layer is the last layer. This is where the deep features are extracted for further process. In the subsequent process, these deep features are combined with the manual features. Then, the combined features go through the machine learning classification.

3.2.4 Manual Features

The essential Manual Features part in Figure 3.4 is where all the contextual features for each of the FL types are engineered. The engineering processes are all explained in the respective subsequent chapters for each FL type under the subsection “Manual Features”.

3.2.5 Machine Learning Classifiers

Five classification algorithms are utilized for comparison, including the following: Support vector machine, K-Nearest Neighbor, Logistic Regression, Decision Tree, and Discriminant Analysis. These are all well-known statistical, pattern recognition, and machine learning techniques.

Support Vector Machine

The support vector machine (SVM) is a binary classification technique that seeks out the best hyperplane to divide a set of data into two groups, negative and positive [162]. The radial basis function (RBF) will serve as the kernel in this work. The function known as an RBF only depends on the separation between the input and another fixed point. Among all the binary classification kernels, this has produced the best results. From earlier research by Poria et al. [19], it has also been demonstrated that this CNN-SVM approach is highly useful for text categorization using deep learning.

K-Nearest Neighbour

The K-nearest neighbor (KNN) algorithm divides outputs into groups by considering the nodes that are close to them [163]. To make decisions, this algorithm uses a threshold value called "k". The default setting of k=10 is utilized for this work, as it is the standard value used in applied machine learning to evaluate models.

Logistic Regression

In this work, a binary classification logistic regression is employed. It only models the likelihood that output will occur in response to an input. A cutoff value is selected as a binary classifier, and the classification is dependent on whether the probability of the inputs is more than the cutoff, which will be put in one class, or lower than the cutoff, which will be put in the other class [164]. The default cutoff value of 0.5 is employed for this work.

Decision Tree

A technique known as a decision tree (DT) uses nodes to represent tests that are performed on an attribute. Additionally, it makes use of branches to represent test results and leaf nodes to indicate labels or conclusions drawn after totaling up all experimental variables. The routes that the tree takes from its root to its leaf nodes serve as a representation of the experiment's rules [165].

Linear Discriminant Analysis

A linear discriminant analysis (LDA) models the variations among the provided data classes. Only continuous measurements of the variables being used will allow it to function. In cases where groups are known in advance, it can also be used. A score on one or more quantitative measures, as well as a score on group measures, are required for each case [166]. LDA is essentially an algorithm for classifying or grouping instances of the same kind into distinct groups.

3.3 Evaluation

The work goes on to the experiments when the features have been extracted. F1-measure, precision, recall, and accuracy are the measures employed to assess the technique. The mathematical equations (3.1) through (3.4) below represent the formulas used for F1-measure, precision, recall, and accuracy, respectively. The corresponding chapters additionally include presentations of the experiments' learning curves.

F1-measure:

$$F1 = 2.\frac{precision\,.\,recall}{precision + recall} \tag{3.1}$$

Precision:

$$precision = \frac{TP}{TP + FP} \tag{3.2}$$

Recall:

$$recall = \frac{TP}{TP + FN} \tag{3.3}$$

Accuracy:

$$accuracy = \frac{Correct\ predictions}{Total\ predictions} \tag{3.4}$$

The accuracy of a classification method is typically used to assess its performance, as shown in equation (3.4) [167]. Although the accuracy might be extremely good, a detection algorithm might not benefit significantly from it. As an illustration, a president of a country might write one statement that sounds normal but is actually sarcastic. Even though it's only one sentence, this error could have unintended or disastrous consequences. Despite the fact that a mistake would have grave consequences, the system's accuracy might nevertheless be very high. Whether the error is due to the algorithm or not is not immediately apparent.

Due to the above reason, the majority of this study uses simply precision, recall, and F1-measure. The equation that indicates precision is equation (3.2). In essence, it is the total number of true positive and false positive instances divided by the number of true positive instances. In this manner, the ratio of actual sarcastic instances to all sarcastic instances anticipated by the system may be calculated. Low precision indicates that the system does a poor job of predicting actual sarcastic statements.

The equation that denotes recall is equation (3.3). In essence, it is the ratio of the number of true positive cases to the sum of true positive and false negative examples. In this scenario, it would be known what portion of the sarcastic instance was genuinely sarcastic. Low recall indicates that the system may have mistaken some sarcastic sentences for normal ones.

Equation F1-measure is denoted in equation (3.1). Precision and recall are balanced using this equation, especially when the dataset is unbalanced. A dataset of sarcasm, for instance, would have more examples of normal speech than sarcastic. This has a clear connection to reality because people rarely utilize sarcasm in conversation as opposed to their regular counterparts.

3.4 Hardware Requirement

A machine running 64-bit Windows 10 and equipped with an Intel(R) Core(TM) i7 8th Gen and NVIDIA(R) GeForce(R) GTX is the setting used to conduct the experiments. Mathworks Matlab 2019a is the application utilized.

3.5 Summary

This work delves into the performance of the pairing of deep features and the contextual explanations of each feature set that is carefully constructed. It provides a sense of how FL detection functions in practice. The proposed methods in the subsequent chapters have shown significant improvement in comparison to the results of other works in the same domain which mainly focus on using just deep learning or just manually handcrafted methods. For future work, the process of finding the contextual features through the correct understanding of FL types could be expanded.

Chapter 4

Sarcasm Detection

4.1 Introduction

This chapter focuses on ways to understand and detect sarcasm from a computational perspective in conjunction with its context. The subsequent two chapters are focused on sarcasm's siblings, namely, metaphor and satire.

4.2 Proposed Method

The collection of data is the first step in this work's comprehensive framework, which concludes with evaluation. Figure 4.1 depicts the entire procedure from above.

The feature extraction techniques listed in the related works section are insufficient to identify all potential sarcastic tweets. In everyday speech, sarcasm frequently uses unusual tones [62, 168] or exaggerations [169]. These are converted into specific wording, symbols, or even the way they are written in a written instance. These are the types of features that are extracted and utilized in this work.

4.2.1 Data Acquisition

The dataset created and publicly shared by [170] was utilized in this research. It comprises actual tweets that were used as instances of sarcasm by using the hashtag "#sarcasm". This dataset may be accessed at https://liks.fav.zcu.cz/sarcasm. In contrast to sarcastic ones, normal occurrences are more likely to occur in tweets and in real life [170]. In light of this, this dataset represents an uneven distribution of 780,000 English tweets (130,000 sarcastic and 650,000 non-sarcastic).

The publicly available dataset is originally divided into two sets, with the ratio of Train to Test being 80:20. The Train set is then divided once more, this time in an 80:20 ratio, into Train and Validation. The entire dataset is, in essence, divided into three sets: train, test, and validation sets, in the ratio of 64:16:20. The 780,000 English tweets in

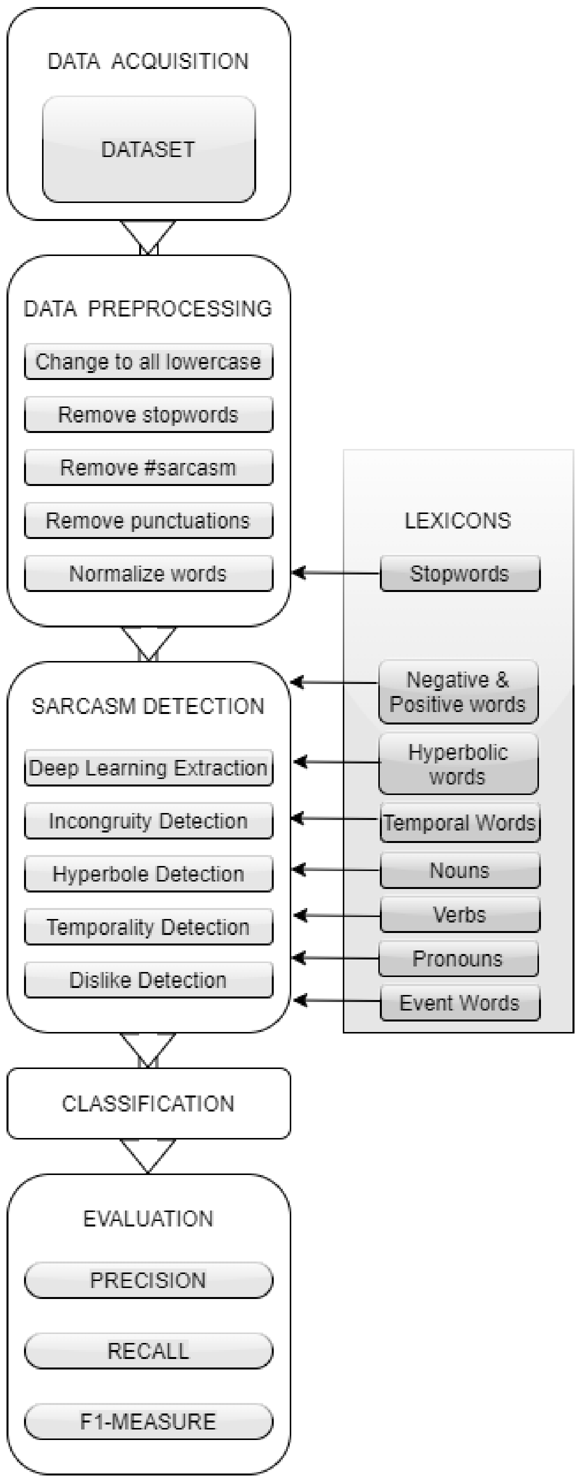


Figure 4.1: Overall framework of Sarcasm Detection

this dataset, 83 percent of which are normal examples, and the others are sarcastic. Thus, there would be 499,200 examples in the training set (414,336 normal and 84,864 sarcastic), 124,800 instances in the validation set (103,504 normal and 21,296 sarcastic), and 150,000 instances in the testing set (124,500 normal and 25,500 sarcastic).

4.2.2 Data Preprocessing

Any NLP project must include preprocessing, to quote Ghosh et al. [25]. This is done to prevent the preprocessed portion from biasing and adding weight to the tests. Five different preprocessing approaches were employed for this goal.

The dataset's text is first transformed to lowercase. The stopwords are then all eliminated. Prior to eliminating any existing stopwords for this activity, the stopword lexicon is utilized to identify all of them. Any instances of "#sarcasm" are then eliminated from the document. The elimination of any punctuation marks comes next. All terms are finally changed to their root form. In contrast to the "stem" form, the "lemma" style is chosen for this task since it is more all-encompassing.

4.2.3 Sarcasm Detection

Deep Feature

Given its strong reputation in NLP, deep learning is increasingly being used in recent studies on the detection of sarcasm [19, 59, 83, 86]). The main benefit of deep learning is its capacity to automatically acquire the best features for a given task, as noted by Poria et al. [19] and Majumder et al. [59].

In this work, ten deep features are extracted using the CNN architecture described in the previous chapter. The deep features extractor (CNN), which is a significant component of the Sarcasm Detector's overall design, is depicted in Figure 4.2. The word-embedding method FastText, which is also covered in the previous chapter, also helps.

The breakdown for the CNN architecture for sarcasm detection part is as below:

1. An input layer of size 1 X 100 X N where N is the number of instances from the dataset. Vectors of embedded-words are used as the initial input.
2. Then the layers between the input and the concatenation is introduced:
    - One convolutional layer with 200 neurons to receive and filter size 1 X 100 X N where N is the number of instances from the dataset. The stride is [1 1].
    - Two convolutional layers with 200 neurons to receive and filter size 1 X 100 X 200. The stride is [1 1].
    - Three batch normalization with 200 channels.
    - Three ReLU activation layers.
    - Three dropout layers with 20 percent dropout.
    - A max pooling layer with stride [1 1].
3. A depth concatenation layer to concatenate all the last max pooling layers.

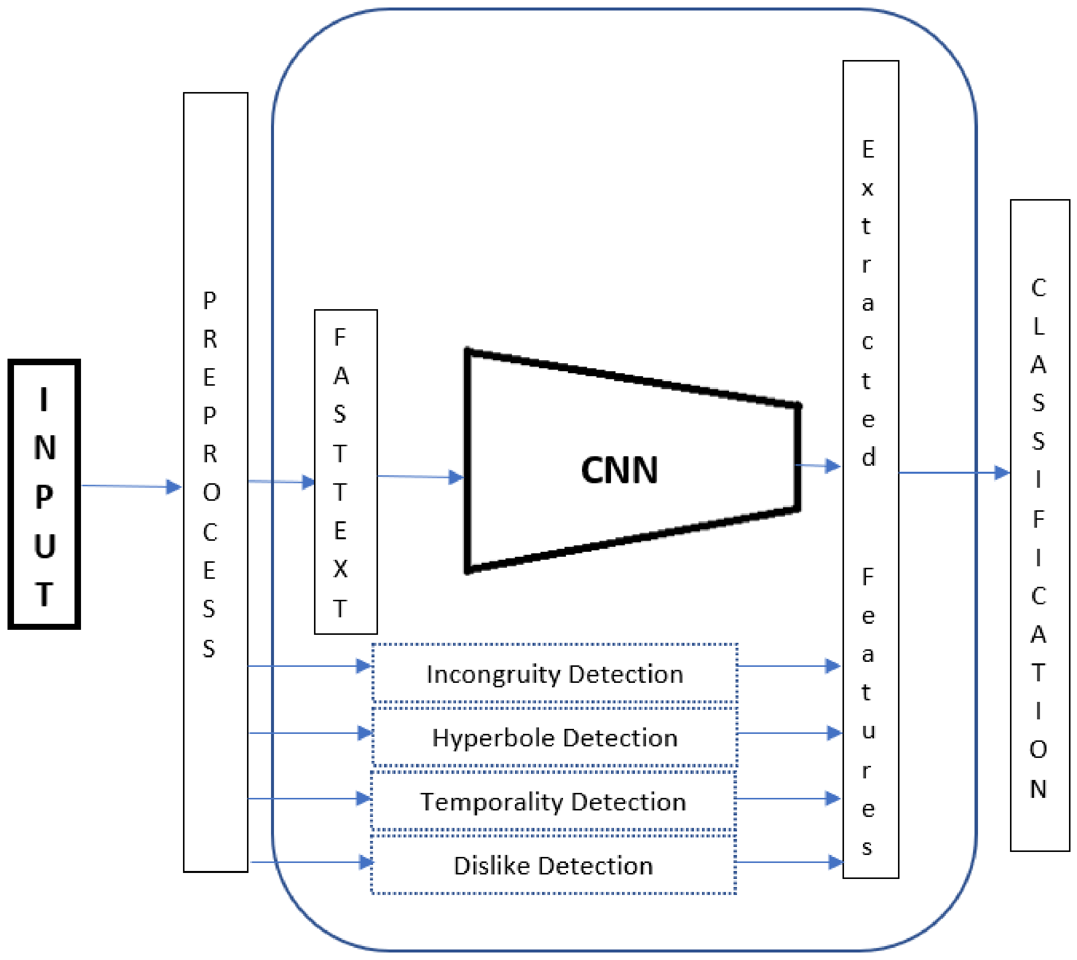


Figure 4.2: Overall architecture of the Sarcasm Detector

4. A fully connected layer with ten neurons.

The convolutional layers utilized to create the feature maps are the main emphasis of this design. The batch normalization layers come next, which re-center and re-scale the incoming data to increase speed, performance, and stability. The scanning of input data is done using the activation functions. In order to avoid overfitting, improve validation accuracy, and boost generalizing power, the minimum 0.2 dropout layers are also used. The final vote is then conducted using the max pooling layer.

The word vectors produced with FastText serve as the starting inputs. [1 100] is the vector size set. Then, these vectors are divided into three groups: group one (N-Gram Length-1, or unigram), group two (N-Gram Length-2, or bigram), and group three (N-Gram Length-3 or normally known as trigram).

The word vectors are divided into their corresponding N-Gram Length groups before being sent into the third layer of the three-graph design. Before being joined at the concatenation layer, the three graph architectures are executed concurrently. The ten neurons in this completely linked layer are retrieved and used as our deep features. The manual features are then added to this feature set before classification.

Figure 4.3 is showing the bird's eye view of all the feature extraction processes. The features are simplified into binary or frequency as shown in the square boxes in the figure as a result of the extraction processes in the squircle boxes above them. The formulas for each of the manually extracted feature sets are given in the mathematical equations (4.1)-(4.11) below. The abbreviations are given next to the title of the feature sets. The notation "f" stands for frequency and "b" stands for binary (either 0 for absence or 1 for presence):

Congruity (C):

$$C1 = \mathrm{f}(positivenegativephrase) + \mathrm{f}(negativepositivephrase) \tag{4.1}$$

$$C2 = \mathrm{f}(positiveword) + \mathrm{f}(negativeword) \tag{4.2}$$

Hyperbole (H):

$$H1 = \mathrm{f}(exaggeratedword) \tag{4.3}$$

$$H2 = \mathrm{f}(positiveword) + \mathrm{f}(negativeword) \tag{4.4}$$

$$H3 = \mathrm{b}(2.exaggeratedword) \tag{4.5}$$

Temporal (T):

$$T1 = \mathrm{b}(temporalword) \tag{4.6}$$

$$T2 = \mathrm{b}(nounword \sim temporalword) \tag{4.7}$$

Dislike (D):

$$D1 = \mathrm{b}(selfpronounword >= 2) \tag{4.8}$$

$$D2 = \mathrm{b}(selfpronounword \cup eventword) \tag{4.9}$$

$$D3 = \mathrm{f}(verbword) \tag{4.10}$$

$$D4 = \mathrm{f}(eventword) \tag{4.11}$$

Incongruity Feature

A sentence that does not make sense in the context is incongruous. The amount of incongruity between the sentence and the context, as mentioned by Ivanko and Pexman [171], affects how long it takes to grasp a sarcastic sentence. This was reiterated by Campbell and Katz [172], that claimed using context incongruity in sarcasm is a need. A similar concept was also developed by Ramteke et al. in their work [173], except they refer to it as thwarting rather than incongruity. Another piece of work refers to this concept as "a contrast," and they developed their own system to identify it [11]. Therefore, in the attempt to extract some incongruity features, subsets of concepts from both publications by Ramteke et al. [173] and Riloff et al. [11] are employed.

The frequency of a negative term appearing after a positive phrase and vice versa is tallied for the context incongruity feature set. This type of feature is also known as an oxymoron. Another feature is the overall quantity of words, both positive and negative. Every instance of the tweets receives this treatment. The next subsection's description of the positive and negative word lexicons is employed in this approach. Positive examples of sarcasm are included in this fashion. Here, the combined features of the two sets are extracted.

Hyperbole Feature

A reliable linguistic indicator for sarcasm detection has always been hyperbole [169]. Sarcasm and exaggeration are two FLs that can go hand in hand. According to Bharti et al. [26], hyperbole can make it easier to identify sarcasm. This is supported by additional works from Kunneman et al. [169] and Bharti et al. [174] in support of the claim that sarcastic expressions frequently employ exaggeration, also known as hyperbole. For instance, the phrase "Great! I love Mondays!" could be sarcastic as well as hyperbolic because it is an overstated statement. After all, people generally detest Mondays. As a result, a few hyperbolic features are retrieved in order to assess their value in sarcasm detection.

The total number of hyperbolic words used in each tweet is tallied. From a lexicon that is described in the next subsection, hyperbolic words are extracted. Additionally, this lexicon is used to check for the presence of two or more hyperbolic words in a tweet. If so, it is included as another feature. The binary form of the second hyperbolic feature is also used. Here, the combined features of the two sets are extracted.

Temporality Feature

Temporality is another characteristic of sarcasm [93]. When Donald Trump won the US election in 2016, the statement "You think just like Donald Trump today" would have a different meaning than when he lost the election in 2020.

Each tweet's temporal words are counted as a separate collection of binary attributes. By paying attention to instances where the word that comes three words before or after the temporal word is a noun, another feature set is additionally added. This trait is also binary in nature. We used the lexicons of temporal words and nouns described in the subsection below for both of these properties. Here, the combined features of the two sets are extracted.

Dislike Feature

Sarcasm is frequently used to indicate that a person has a negative attitude about another person or an event [94]. According to one of the most defining characteristics of sarcasm—that it rarely occurs between strangers [168]—some elements are retrieved to model hate. These characteristics are further classified into the following groups: i. In the same tweet, a person addresses a known individual. ii. A reference in the same tweet to a well-known event.

Another sign that a bad emotion is likely to be experienced in real life is online participation. This was clarified by Chenand Boves [175] in two straightforward definitions that are utilized as two more feature groups: i. The more tweets there are about anything, the more we may trust that it represents the active disapproval of the public. ii. Online talking points with a lot of activity are typically more likely to make people feel bad.

Here, four dislike features are added. First, a tweet must contain both a self-pronoun and another self-pronoun in it. The second is when a self-pronoun appears in the same tweet as an event word like "affair," "incident," or "episode." These two properties both exist in binary form. Every verb that appears throughout the entire dataset is recorded and used as a feature using the count occurrence. Verbs serve as clear indicators of a person's behavior [11]. The use of event terms is done using three lexicons which are explained in the subsection below.

4.2.4 Lexicons

In order to extract all the manual features, some lexicons are used. These lexicons are directly correlated with the manual feature engineering in the subsection above based on its order.

Stopwords There are 225 stopwords used in this work. Stopwords are words that do not carry any meaning to the whole structure of a given sentence, e.g. "this", "that", "he" and "she". The stopwords lexicon offered in the application is utilized in the experiments. This application is explained in the Hardware Requirement section in Chapter 3.

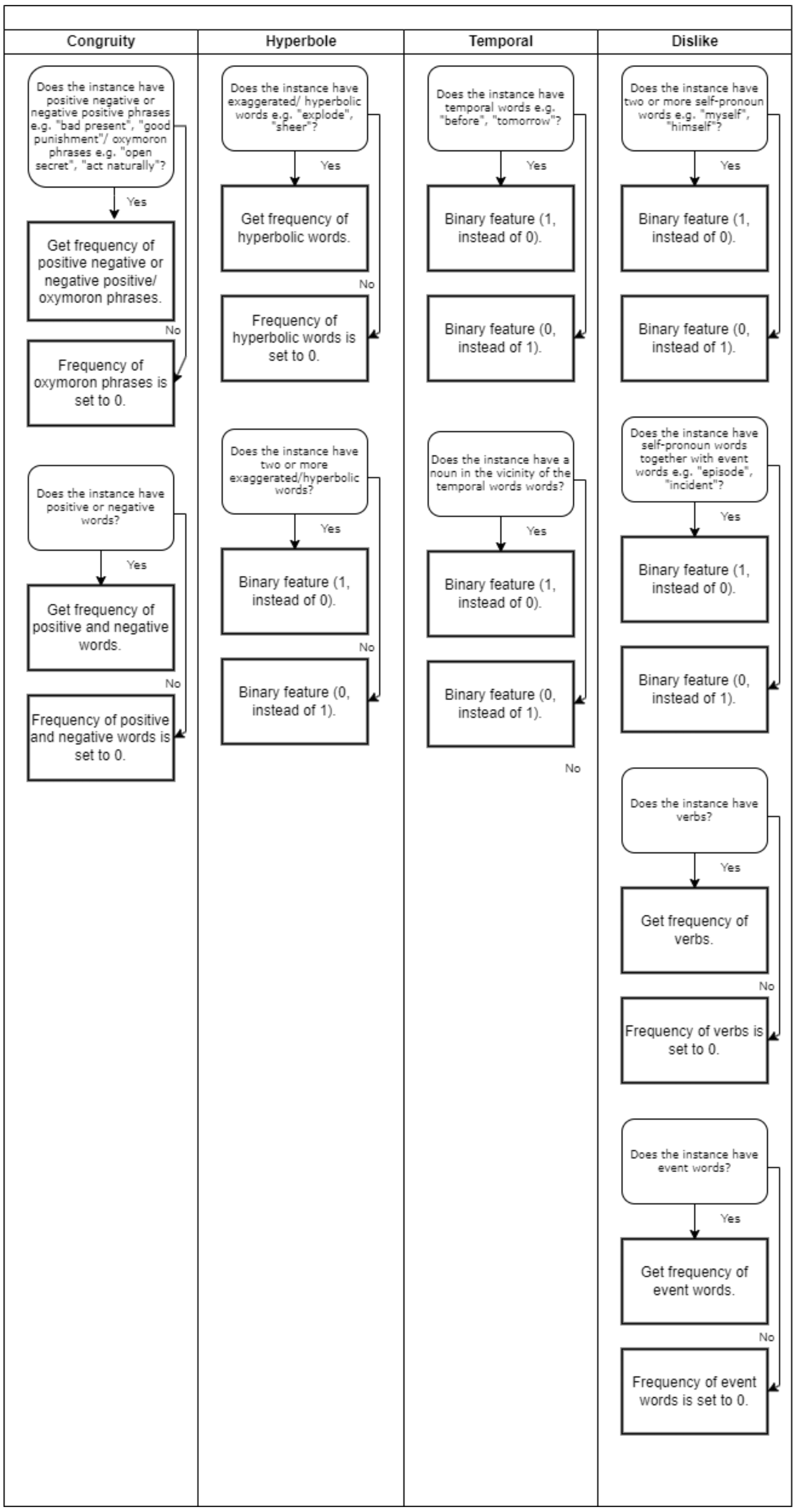

Congruity
Hyperbole
Temporal
Dislike
Does the instance have positive negative or negative positive phrases e.g. "bad present", "good punishment"/ oxymoron phrases e.g. "open secret", "act naturally"?
Yes
Get frequency of positive negative or negative positive/ oxymoron phrases.
No
Frequency of oxymoron phrases is set to 0.
Does the instance have positive or negative words?
Yes
Get frequency of positive and negative words.
No
Frequency of positive and negative words is set to 0.
Does the instance have exaggerated/ hyperbolic words e.g. "explode", "sheer"?
Yes
Get frequency of hyperbolic words.
No
Frequency of hyperbolic words is set to 0.
Does the instance have two or more exaggerated/hyperbolic words?
Yes
Binary feature (1, instead of 0).
No
Binary feature (0, instead of 1).
Does the instance have temporal words e.g. "before", "tomorrow"?
Yes
Binary feature (1, instead of 0).
Binary feature (0, instead of 1).
Does the instance have a noun in the vicinity of the temporal words words?
Yes
Binary feature (1, instead of 0).
Binary feature (0, instead of 1).
No
Does the instance have two or more self-pronoun words e.g. "myself", "himself"?
Yes
Binary feature (1, instead of 0).
Binary feature (0, instead of 1).
Does the instance have self-pronoun words together with event words e.g. "episode", "incident"?
Yes
Binary feature (1, instead of 0).
Binary feature (0, instead of 1).
Does the instance have verbs?
Yes
Get frequency of verbs.
No
Frequency of verbs is set to 0.
Does the instance have event words?
Yes
Get frequency of event words.
No
Frequency of event words is set to 0.

Figure 4.3: Manually Extracted Features for Sarcasm Detection

Positive and Negative Words The positive and negative words lexicon was downloaded from an existing resource [176] that is working sentiment analysis. It consists of 6800 words.

Hyperbolic Words This dataset is a subset from that of another work [177] that works on hyperbole. It consists of 710 instances. This lexicon is used in the hyperbole-related features extraction process.

Temporal Words Temporal or transition words are words that deal with time. We downloaded a list of these words from a prominent website providing it [178]. It consists of 52 instances. This lexicon is used in the process of extracting the temporal-related features.

Nouns This lexicon was downloaded from a website that has the most comprehensive list of nouns [179]. This list contains 1500 instances. This lexicon is used in the process of extracting the features for temporal-related features.

Pronouns There are many types of pronouns. That includes personal, objective, subjective, possessive, demonstrative and many more. We collected all the instances for the purpose of this work. They are downloaded from a prominent website [180]. It has about 300 instances.

Verbs This lexicon is downloaded from a website with the most comprehensive list of verbs [181]. This list contains about 600 instances. Then another list was downloaded from another website [182] containing the most comprehensive list of irregular verbs. This list contains about 300 instances. These lexicons are used in the process of extracting the features for dislike-related features, same as the last two lexicons below.

Event Words A list of synonyms for the word “event” is downloaded from an online dictionary for the purpose of having an event words lexicon. It consists of the words: “affair”, “circumstance”, “episode”, “hap”, “happening”, “incident”, “occasion”, “occurrence”, and “thing”.

4.2.5 Machine Learning Classifiers

Five machine learning algorithms are used for the purpose of classification in this work. First is the support vector machine (SVM) [162]. The kernel that is chosen to be used in this work is radial basis function (RBF). All the other kernels are also tried in the experiment but RBF has proven itself to be the best. Second is the K-nearest neighbor (KNN) [163]. The default threshold value k=10 is used in this work. Third is the logistic regression (LR) [164]. The default threshold value of 0.5 is used in this work. Next is the decision tree (DT) [165], which is using all the default parameters. Finally, the linear discriminant analysis (LDA) [166], also uses all the default parameters.

4.3 Results and Discussion

In this section, the results produced by all the experiments are given using the evaluation metrics explained in the previous chapter. This is followed by an explanation of how each component in the experiments is used to fulfill the objective of the work.

4.3.1 Classification Results

This subsection consists of performance results for every classification algorithm in the experiment using all the feature sets combined. It is shown in terms of Precision, Recall, F1-measure, and Accuracy in Table 4.1.

Table 4.1: Performance Comparison for Classification Algorithms using all the Feature Sets combined for Sarcasm Detection

Classification Algorithm	Precision	Recall	F1	Accuracy
SVM	0.92	0.92	0.92	92%
KNN	0.91	0.91	0.91	91%
LR	**0.95**	**0.94**	**0.94**	**94%**
DT	0.92	0.92	0.92	92%
LDA	0.92	0.91	0.92	91%

SVM and DT are both good classifiers for this task, with high accuracies and F1-measures. However, the performance of Logistic Regression are the highest. For the rest of the chapter, the results used are those from the classifier Logistic Regression. The most optimum learning curve for this is also shown in Figure 4.4 below.

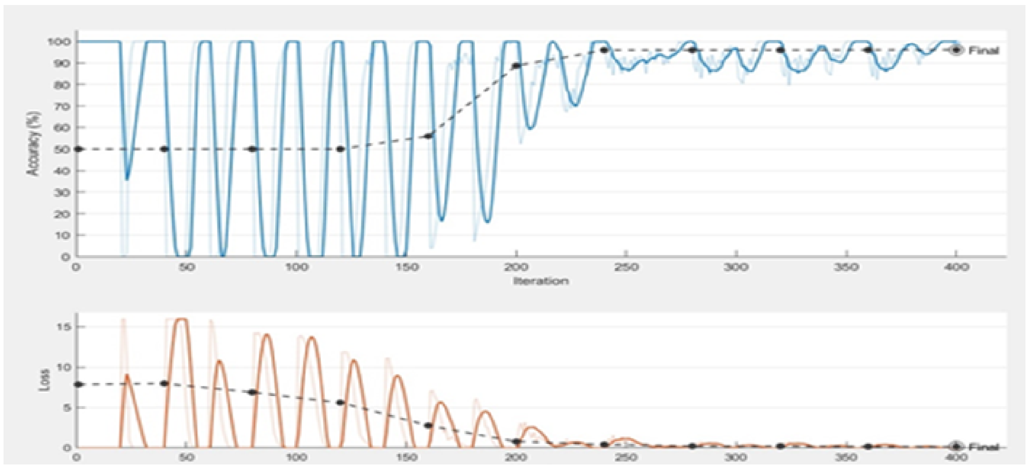


Figure 4.4: Learning Curve using LR for Sarcasm Detection

4.3.2 Performance Comparison with Existing Works

For comparison, two recent works in the same domain that used the same dataset as in our experiment are chosen. The results are compiled and shown in Table 4.2.

Table 4.2: Performance Comparison with Existing Sarcasm Detection Works

Method	Precision	Recall	F1	Accuracy
Ilić et al.[84]	0.87	0.87	0.87	88%
Shmueli et al.[83]	0.87	0.91	0.86	87%
Proposed Method	**0.95**	**0.94**	**0.94**	**94%**

The proposed method showed significantly better performance in comparison to the others across all metrics used. This supports our claim that manually extracted contextual feature sets are useful for this task. The performance for each of the feature sets is also given.

4.3.3 Performance Comparison among Feature Sets

This subsection consists of performance results for each of the feature sets. It is shown in terms of F1-measure as it is the harmony between precision and recall.

Figure 4.5 shows the performance of each of the feature sets used in the work. The performance of the deep feature set is obviously the highest in comparison to the others shown below, with 0.72. The performance for the incongruity and hyperbolic feature sets are comparable to the highest, with 0.7 and 0.69 respectively.

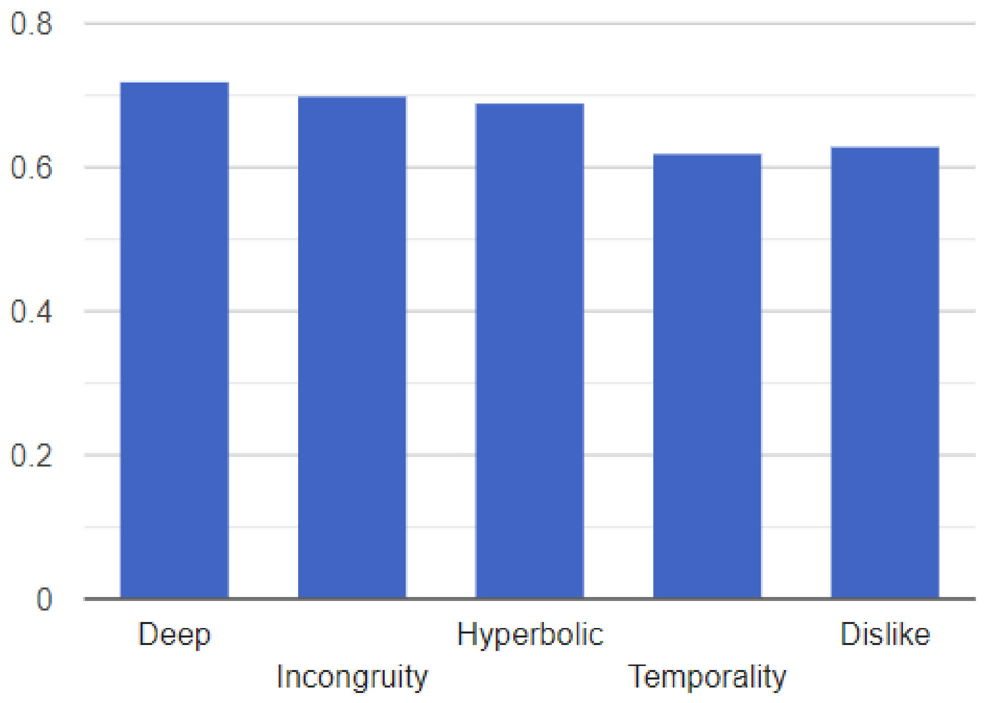


Figure 4.5: Performance of Each Feature Set for Sarcasm Detection

Compared to the other three, temporality and dislike feature sets show low performances with 0.62 and 0.63 respectively. According to observation, the reason for this to happen is that these features have low presence in the data set due to the use of slang and short forms. This phenomenon makes it more challenging for sarcasm to be found using temporality and dislike features. However, the F1-measure given by these feature sets is more than chance (equivalent to 0.5), showing the importance of such features for the task of sarcasm detection [183]. Although the performance is not good as stand-alone features, they have higher added value when correlated with other features. This tells us that even though the features are least important in comparison to other features due to the lack of frequency, the contextual prowess cannot be taken lightly.

4.4 Conclusion

The results of our experiments provide some valuable insights into the usages of different features of tweets. The three feature sets that are found to be the highest in the results are actually expected, especially the deep features. The reason being is that from the literature review section, it is evident deep learning can be a very powerful tool in detecting textual features. However, as mentioned in the results section of this chapter the handcrafted features are also significant.

Hence, all the features are exploited to build a framework that is proven to be useful to detect sarcasm in the end. The method also significantly improves the F1-measure from the existing work using the same dataset. This work also demonstrates the generality of a deep learning architecture, where the features extracted from it can be used to be combined effectively with other features. For future work, a few datasets will be used to further generalize the comparisons. The process of extracting and gathering meaningful features could also be expanded further.

Chapter 5

Metaphor Detection

5.1 Introduction

As stated in the previous chapters, there are many types of FL. Metaphor is one that requires careful work and consideration, since it is a big concept. This work has studied it from the perspective of linguistics and psychology. From the literature review, it is found that there are a few features that have not been exploited for the task of detecting it in a computational setting.

This chapter focus on ways to understand and detect metaphor from a computational perspective together with its context. The subsequent chapter is focused on satire.

5.2 Proposed Method

The whole proposed method is shown in Figure 5.1. As mentioned earlier, the focus of this work is on using a deep learning architecture to create deep features that are then combined with justified handcrafted features.

As shown in Figure 5.1, the whole process consists of Data Acquisition, Data Preprocessing, Metaphor Detection (the largest and most important part of the work), Classification (using multiple machine learning algorithms) and Evaluation. A few lexicons are also used to assist the processes. The detail of each part is given.

5.2.1 Data Acquisition

The dataset used for the features extraction is a manually annotated one, comprising 1600 balanced instances of metaphoric and normal English sentences. It is created and shared by Mohammad et al. [184] and also known as the MOH dataset. It can be downloaded from the website: http://saifmohammad.com/WebPages/ metaphor.html. Consider the example "He absorbed the costs for the accident". The authors of the dataset believe that

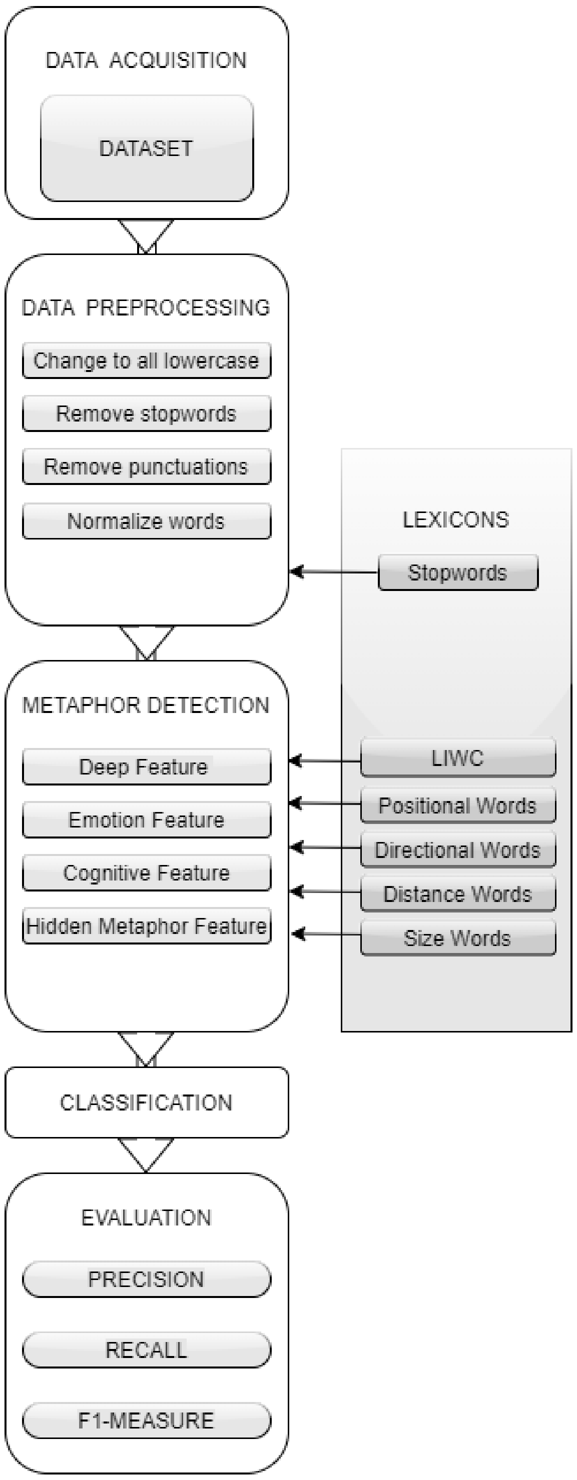


Figure 5.1: Overall framework of Metaphor Detection

the keyword for this instance that made it metaphoric is the word “absorbed” since the person in the sentence does not literally absorb the subject.

5.2.2 Data Preprocessing

Five preprocessing techniques that have been used from the previous chapter in the data preprocessing subtopic are chosen again to fulfill this detection task.

5.2.3 Metaphor Detection

Deep Feature

Existing researchers in this domain are starting to use deep learning as their main strategy after recent successes in NLP [113–115]. Deep learning is now known to have the strength to automatically detect reliable features for NLP tasks. These features are called deep features.

The CNN architecture used here is the same one as explained in the previous chapters. It is proposed to extract the set containing ten deep features. Overall architecture of Metaphor Detection method is shown in Figure 5.2 below.

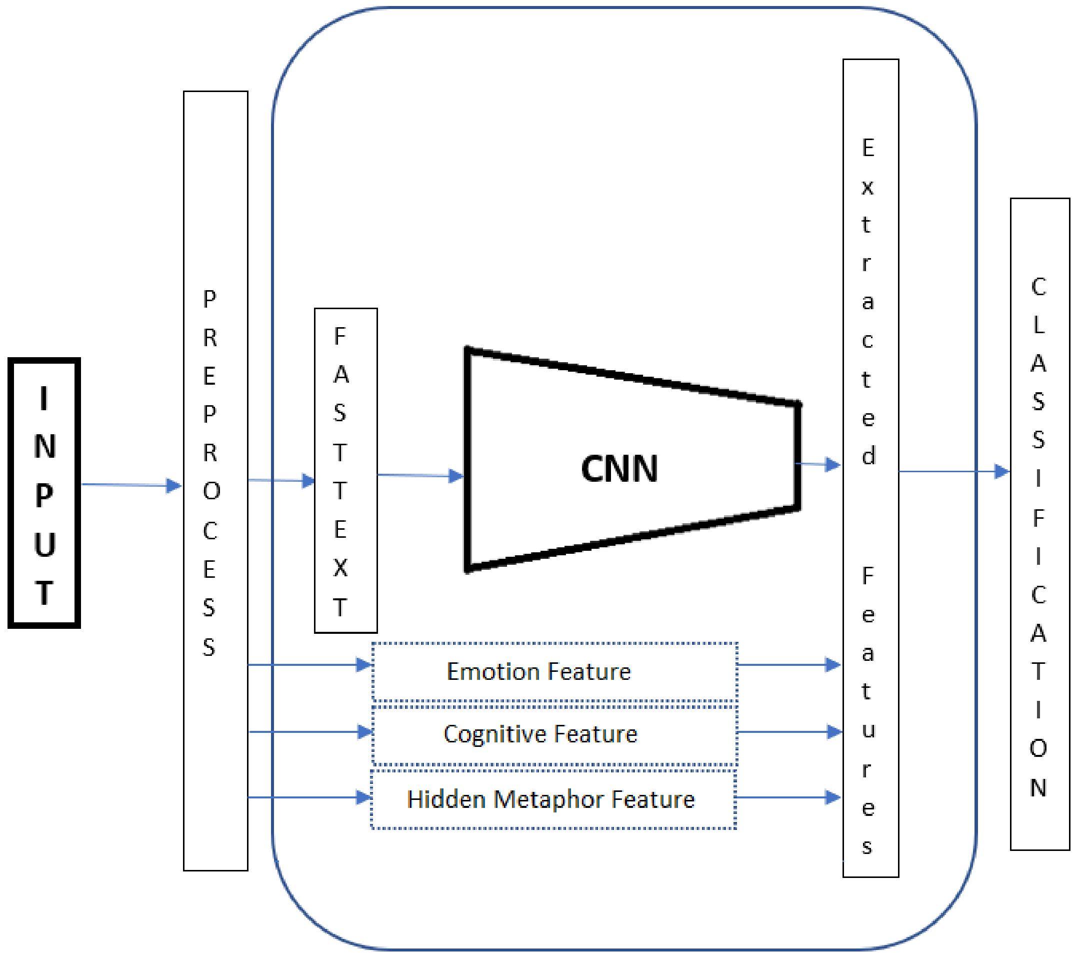


Figure 5.2: Overall architecture of Metaphor Detection

After the deep features are extracted using the CNN architecture, these deep features are combined with the manual features. Then, the combined features go through the machine learning classification. The manual features are explained below:

Emotion Feature

Metaphor is a medium for emotion [184]. It is common knowledge that metaphors are usually used to describe difficult emotions [185]. Some researchers would go as far as create a new lexicon to prove this [28] and some even combined emotion with other features such as cognition to detect metaphor [101, 186].

For emotion feature extraction, Linguistic Inquiry Word Count (LIWC) [187] is used. It is a tool used to count specific words and has been used multiple times in NLP works. It is used mainly for the process of counting words, instead of relying on other methods such as seeding. This is also the reason that it is decided lexicons will be used for all the manually handcrafted contextual features extraction process. For the purpose of emotion feature extraction, categories closely related to emotion are chosen from LIWC. Each category becomes one feature set. These include the below:

1. Positive emotion, e.g. “passion”, “agree”
2. Negative emotion, e.g. “agony”, “annoy”
3. Anxiety, e.g. “embarrass”, “avoid”
4. Anger, e.g. “assault”, “offend”
5. Sadness, e.g. “despair”, “grim”.

Cognitive Feature

Cognition is closely related to emotion in the way that there would be no emotion without any cognitive activity [101, 186]. Hence, cognitive feature is also included in this work.

For cognitive feature extraction, the same LIWC [187] tool is used. For the purpose of cognitive feature extraction, categories closely related to cognition are chosen from LIWC. Each category becomes one feature set. These include the below:

1. Affect, e.g. “ache”, “like”
2. Insight, e.g. “believe”, “aware”
3. Certainty, e.g. “never”, “true”
4. Cognitive, e.g. “if”, “could”

Hidden Metaphor

Upon observation of the metaphor datasets, this work hypothesized that metaphoric words have a few hidden characteristics. These include the below:

1. Time is related to position, e.g. "Never turn around to your past" and "I am looking forward to seeing you again". In these cases, the words "around" and "forward" which are generally used for the space-context, are used in the time-context.

2. Emotion is related to distance, e.g. "You are always near to my heart" and "I am always close to you". In these cases, the words "near" and "close" are used as a way to explain emotional-closeness rather than the general distance-closeness.

3. Emotion is related to direction, e.g. "I always look up to you" and "You look down". Instead of saying "happy" or "sad" in these cases, the words "up" and "down" are used to replace them.

4. Importance is related to size, e.g. "You are a big man now" and "Just a little effort". The words "big" and "little" in these cases are used to show the importance of the subjects.

Hidden metaphor categories include positional words (i.e. "above", "after", "around"), directional words (i.e. "behind", "below", "beside"), distance words (i.e. "nigh", "near", "elevation") and size words (i.e. "beefy", "immeasurable", "short"). For the purpose of extracting these features, other lexicons are used [188–191]. Again, these features are in the form of word counts. Each occurrence of the words in each sentence are counted according to its category. Each category becomes one feature.

In total, thirteen manual features are extracted. Figure 5.3 is showing the bird's eye view of all the feature extraction processes. The features are simplified into frequency as shown in the square boxes in the figure as a result of the extraction processes depicted in the squircle boxes above them. The formulas for each of the manually extracted feature sets are given in the mathematical equations (5.1)-(5.6) below. The abbreviations are given next to the title of the feature sets. The notation "f" stands for frequency and "b" stands for binary (either 0 for absence or 1 for presence):

Emotion (E):

$$E1 = \mathrm{f}(emotionalwords) \tag{5.1}$$

Emotion (Co):

$$Co = \mathrm{f}(cognitivewords) \tag{5.2}$$

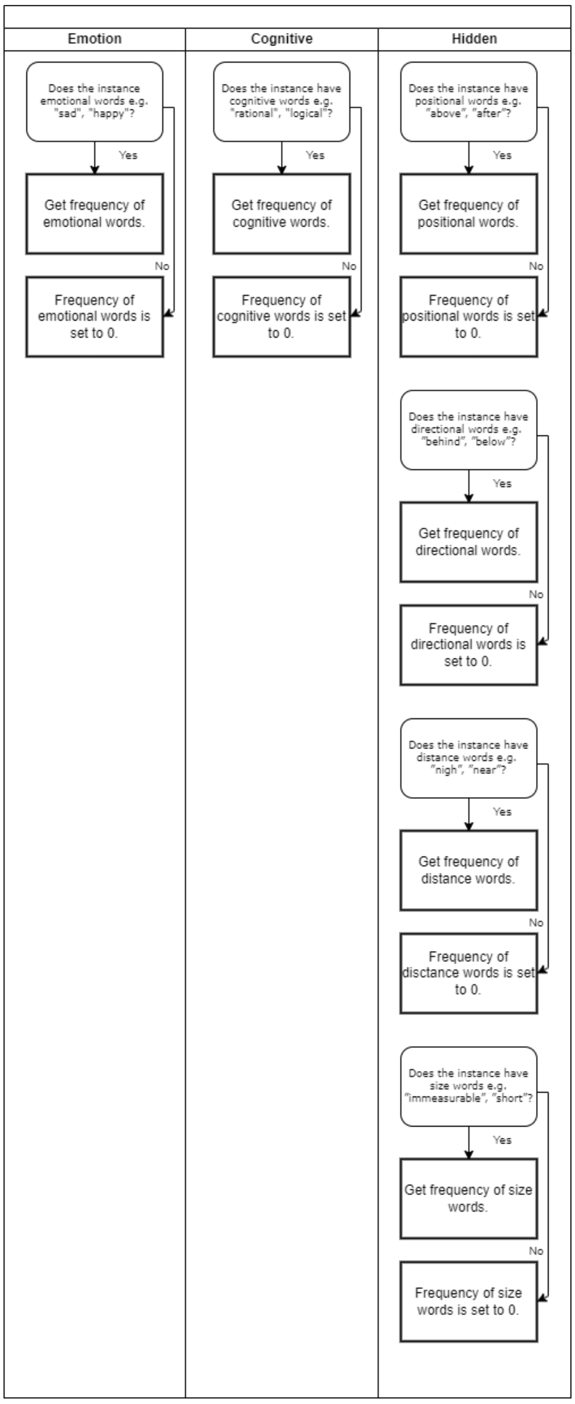
Hidden (Hi):

$$Hi1 = \mathrm{f}(positionalwords) \tag{5.3}$$

$$Hi2 = \mathrm{f}(directionalwords) \tag{5.4}$$

$$Hi2 = \mathrm{f}(distancewords) \tag{5.5}$$

$$Hi2 = \mathrm{f}(sizewords) \tag{5.6}$$



Emotion
Cognitive
Hidden
Does the instance emotional words e.g. "sad", "happy"?
Yes
Get frequency of emotional words.
No
Frequency of emotional words is set to 0.
Does the instance have cognitive words e.g. "rational", "logical"?
Yes
Get frequency of cognitive words.
No
Frequency of cognitive words is set to 0.
Does the instance have positional words e.g. "above", "after"?
Yes
Get frequency of positional words.
No
Frequency of positional words is set to 0.
Does the instance have directional words e.g. "behind", "below"?
Yes
Get frequency of directional words.
No
Frequency of directional words is set to 0.
Does the instance have distance words e.g. "nigh", "near"?
Yes
Get frequency of distance words.
No
Frequency of disctance words is set to 0.
Does the instance have size words e.g. "immeasurable", "short"?
Yes
Get frequency of size words.
No
Frequency of size words is set to 0.

Figure 5.3: Manually Extracted Features for Metaphor Detection

5.2.4 Lexicons

Some lexicons are used to extract the manually handcrafted features. The lexicons are as followed:

Stopwords The stopwords lexicon offered in the application is utilized in the experiments

LIWC Specific categories are selected from this lexicon [187] for the detection task.

Hidden Words Hidden words for hidden metaphor features is downloaded from existing resources [188–191]. They contain the categories described above.

5.2.5 Machine Learning Classifiers

Five machine learning algorithms used in the previous chapter for sarcasm detection are used again for the purpose of classification. This time it is for metaphor detection.

5.3 Results and Discussion

All the results yielded by the experiments in this work are shown in this section. The way each element of this work is used is also explained.

5.3.1 Classification Results

Every result yielded from all classification algorithms used in the experiments are shown in Table 5.1. These are the results of all the feature sets combined. The metrics used are Precision, Recall, F1-measure and accuracy.

Table 5.1: Performance Comparison for Classification Algorithms using all the Feature Sets combined for Metaphor Detection

Classifier	Precision	Recall	F1	Accuracy
SVM	**0.83**	**0.84**	**0.83**	**83%**
KNN	0.80	0.80	0.80	80%
LR	0.81	0.81	0.81	81%
DT	0.81	0.81	0.81	81%
LDA	0.81	0.80	0.81	81%

LR, DT and LDA are all good classifiers for this task, with high F1-measures, precision and recall. KNN also does not fall too far below. However, the performance of SVM is highest in this work. Hence, for the rest of this chapter only SVM results are used to represent this work. The most optimum learning curve for this is also shown in Figure 5.4 below.

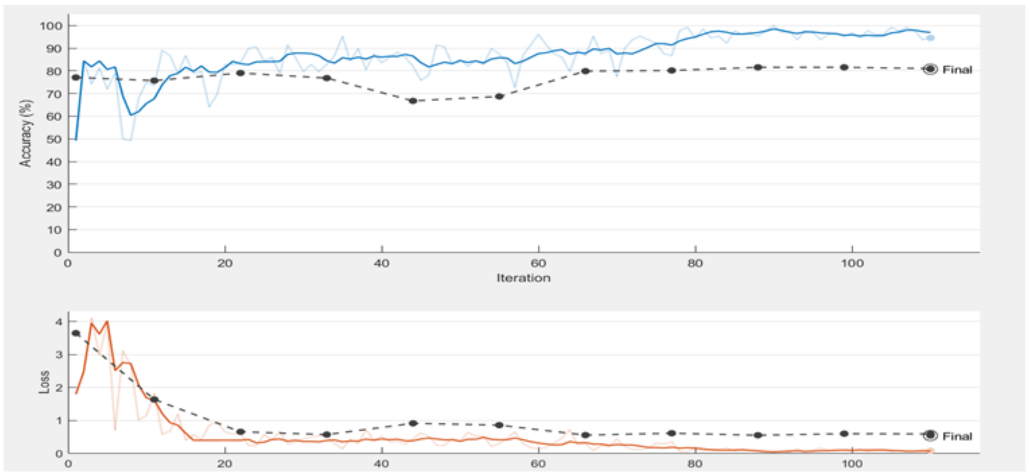


Figure 5.4: Learning Curve using SVM for Metaphor Detection

5.3.2 Performance Comparison with Existing Works

Table 5.2: Performance Comparison with Existing Metaphor Detection Works

Method	Precision	Recall	F1	Accuracy
Shutova et al.[8]	0.67	0.76	0.71	72%
Mao et al.[192]	0.68	0.82	0.74	75%
Rei et al.[193]	0.74	0.76	0.74	74%
Proposed Method	**0.84**	**0.83**	**0.83**	**83%**

Three existing works attempting the same task and using the same dataset are chosen. The results are compiled together with the results of this work and shown in Table 5.2. The first existing work [192] have proposed an unsupervised learning method that detects metaphors at word-level without any preprocessing. The next work [8] uses visual and textual features for metaphor detection. Wikipedia is also used as their source for training textual features and visual embeddings. The final work [193] is also the first that uses deep learning for this task. They have used the combination of vector space mapping weighted cosine as their main technique.

In comparison with all the other existing works in the same domain stated, the method proposed in this thesis shows better performance in all metrics used. This confirms our submission that manually handcrafted contextual features are useful for this specific task.

5.3.3 Performance Comparison among Feature Sets

The results of each of the feature set is shown in this subsection as separate entities, as a way of comparison and additional information. They are shown in Figure 5.5 in terms of F1-measure.

The performance of the deep feature set is shown to be the single set that has the highest performance, with 0.75. The emotion and cognitive feature sets are almost the same to each other with 0.67 and 0.68 respectively. The hidden metaphor set shows a slightly lower performance than the other three with 0.61. According to observation, the reason for this to happen is that this feature is rarely used. However, the F1-measure given by these feature sets is more than chance (equivalent to 0.5), showing the importance of such features for the task [183]. Even though it is the least important as compared to the others as a stand-alone feature, it is obvious that it has some merit contextually and when used in a combination.

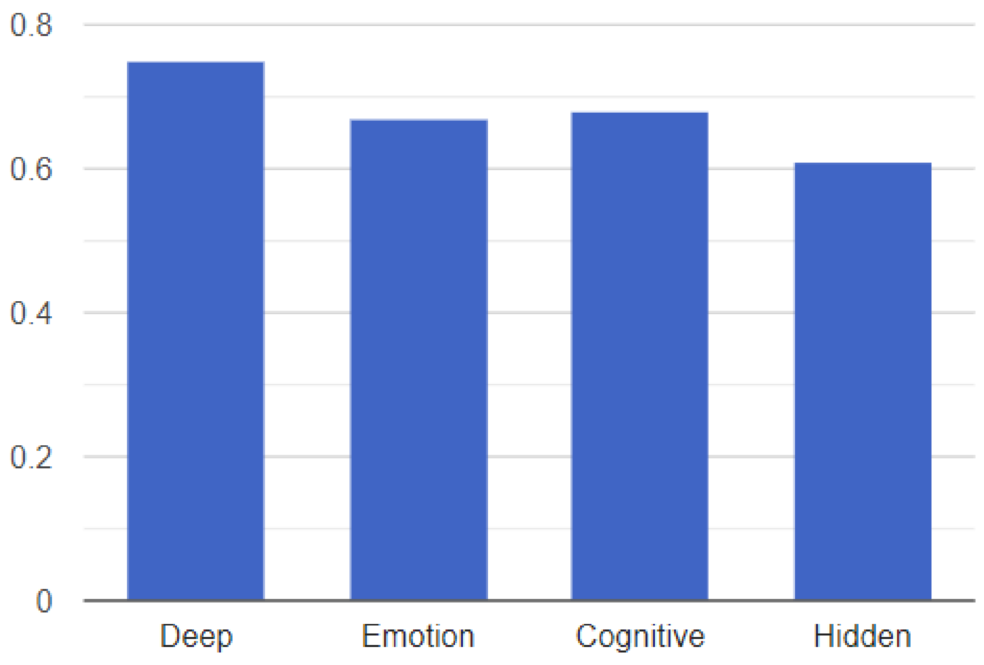


Figure 5.5: Performance of Each Feature Set for Metaphor Detection

5.4 Conclusion

This work dives into the contextual justifications of each feature sets that are manually handcrafted. It gives a sense of how metaphor works in the real world. These sets are then combined with the automatically extracted set using a deep learning architecture. This process offers a perfect metaphor of how the mechanics of machines can work well with humans.

As expected from the literature review, the feature set gathered from the deep learning architecture performs the best, followed by the other feature sets. Even though the other sets did not perform as well, but it is proven to be significant since they did help in increasing the overall performance. The overall proposed method is shown to significantly improve the results in comparison to other works in the same domain which mainly focuses on using just deep learning or manually handcrafted methods. For future work, the process of finding the contextual features through the right understanding of metaphor can be expanded.

Chapter 6

Satire Detection

6.1 Introduction

Satire is an FL type that is often overlooked by researchers in this domain even though it is a big concept. Hence, there is not very much reference on satire detection techniques. This might be because it does not usually happen in short-texts. This makes it more challenging to process. This is also the reason why dataset used for this part is different from the others (long text).

This chapter focus on the task of understanding and detecting satire. All the contexts are explained in great detail.

6.2 Proposed Method

The whole proposed method is shown in Figure 6.1. As mentioned earlier, the focus of this work is on using a deep learning architecture to create deep features that are then combined with justified handcrafted features.

As shown in Figure 6.1, the whole process consists of Data Acquisition, Data Preprocessing, Satire Detection (the largest and most important part of the work), Classification (using multiple machine learning algorithms), and Evaluation. A few lexicons are also used to assist the processes. The detail of each part is given below.

6.2.1 Data Acquisition

The news dataset used for the features extraction was created and shared by Yang et al. [146], through e-mail communication. The articles were crawled from self-proclaimed satirical news providers, i.e. The Onion and real news providers, i.e. CNN. All the other features in the dataset were removed apart from the source (news providers), which is used as the label as well as the news contents as the input. The whole dataset comprises

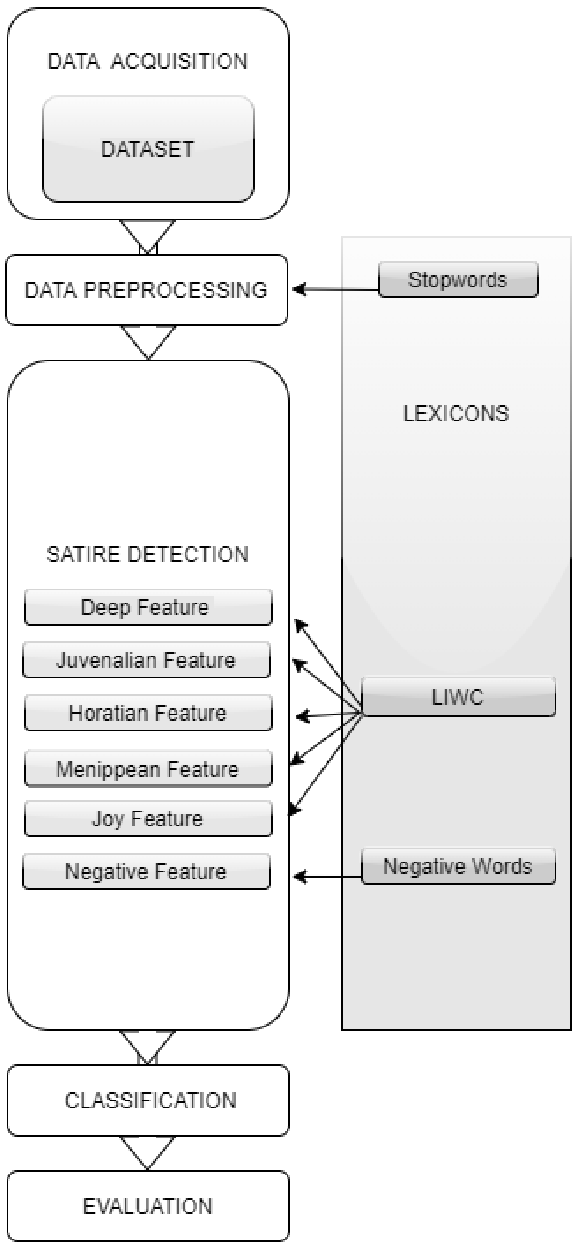


Figure 6.1: Overall framework of Satire Detection

about 16000 satirical news data and 160000 real news data. This number directly imitates the real world where satirical instances rarely happen in comparison to real instances.

6.2.2 Data Preprocessing

Five preprocessing techniques that have been used from the previous two chapters in the data preprocessing subtopic are chosen again to fulfill this detection task.

6.2.3 Satire Detection

Deep Feature

Existing researchers in this domain are starting to use deep learning [145, 146] as their main strategy after its recent successes in NLP. Deep learning is now known to have the strength to automatically detect good features for NLP tasks. These features are called deep features.

In this work, a CNN architecture is proposed to extract the first set of the work. It is also the biggest and most important part of the overall architecture, as shown in Figure 6.2. The extracted set consists of ten deep features. The CNN architecture is the same as the one used in the previous chapters. The only difference in the architecture is that all the initial input vector size is set to [1 500] as satire is about paragraphs rather than sentences.

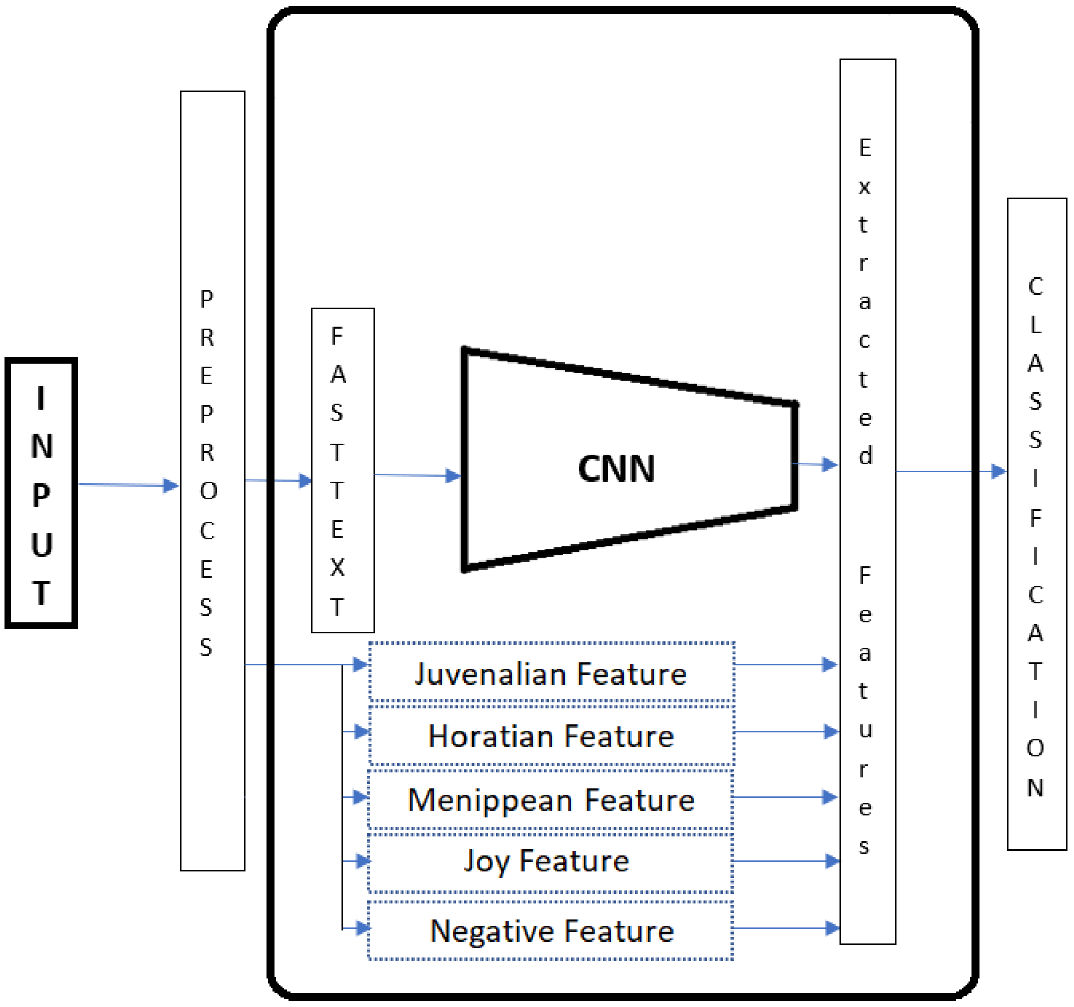


Figure 6.2: Overall architecture of Satire Detection

As mentioned in the literature review section, there are three very distinct types of satire: Juvenalian, Horatian and Menippean [194–196]. However, manual feature extraction for this work goes beyond by looking into existing work in the same domain.

Juvenalian Feature

The Juvenalian type is when satire is used in a comedic manner to make the situation lighter. Hence, the comedic category is chosen from the LIWC tool lexicon to extract features in the form of frequency. This is the first feature set.

Horatian Feature

The Horatian type of satire is dark, which is quite the opposite of the Juvenalian mentioned above. For this purpose, dark category is chosen from the LIWC tool lexicon to extract features in the form of frequency. This is the second feature set

Menippean Feature

There is a third type where satire is used to mock culture, belief system, political system, or human characteristic, which is called the Menippean type [197, 198]. This type is used a lot in works of arts [135, 199]. Here the feature set is created using the frequency of culture, belief, politic and character words from the respective categories in the LIWC tool lexicon. There are four feature sets created here, making it six so far.

Joy Feature

The seventh set of manual feature is based on a work that claims satire is also used for the amusement of oneself [200]. This means that it contains the element of joy. This feature is then created by observing the frequency of joy words by referring to its respective category in the LIWC tool lexicon.

Negative Feature

The eighth and final set of manual feature is based on a work that claims satire is nothing but a form of trope that is used to make fun of something or someone [140]. Making fun of anything in its nature is an ill-intentioned act. Hence, this set is made from the absence or presence of negative words in the form of frequency.

In total, eight manual features are extracted. Figure 6.3 is showing the bird's eye view of all the feature extraction processes. The features are simplified into frequency as shown in the square boxes in the figure as a result of the extraction processes in the squircle boxes above them. The formulas for each of the manually extracted feature sets are given in the mathematical notations (6.1)-(6.8) below. The abbreviations are given next to the title of the feature sets. The notation "f" stands for frequency and "b" stands for binary (either 0 for absence or 1 for presence):

Juvenalian (J):

$$J1 = \mathrm{f}(comedicwords) \tag{6.1}$$

Horatian (Ho):

$$Ho1 = \mathrm{f}(darkwords) \tag{6.2}$$

Menippean (M):

$$M1 = \mathrm{f}(culturalwords) \tag{6.3}$$

$$M2 = \mathrm{f}(beliefwords) \tag{6.4}$$

$$M3 = \mathrm{f}(politicalwords) \tag{6.5}$$

$$M4 = \mathrm{f}(characterwords) \tag{6.6}$$

Joy (Jo):

$$J1 = \mathrm{f}(joywords) \tag{6.7}$$

Negative (N):

$$N1 = \mathrm{f}(negativewords) \tag{6.8}$$

6.2.4 Lexicons

Some lexicons are used to extract the manually handcrafted features. The lexicons used are as followed:

Stopwords The stopwords lexicon offered in the application is utilized in the experiments

LIWC Specific categories are selected from this lexicon [187] for the detection task.

Negative Words Negative words lexicon is downloaded from an existing resource [176] that is working sentiment analysis. It consists of 4700 words.

6.2.5 Machine Learning Classifiers

Five machine learning algorithms that are used for the purpose of classification in sarcasm detection and metaphor detection in the previous chapters are also used here. This time is for satire detection.

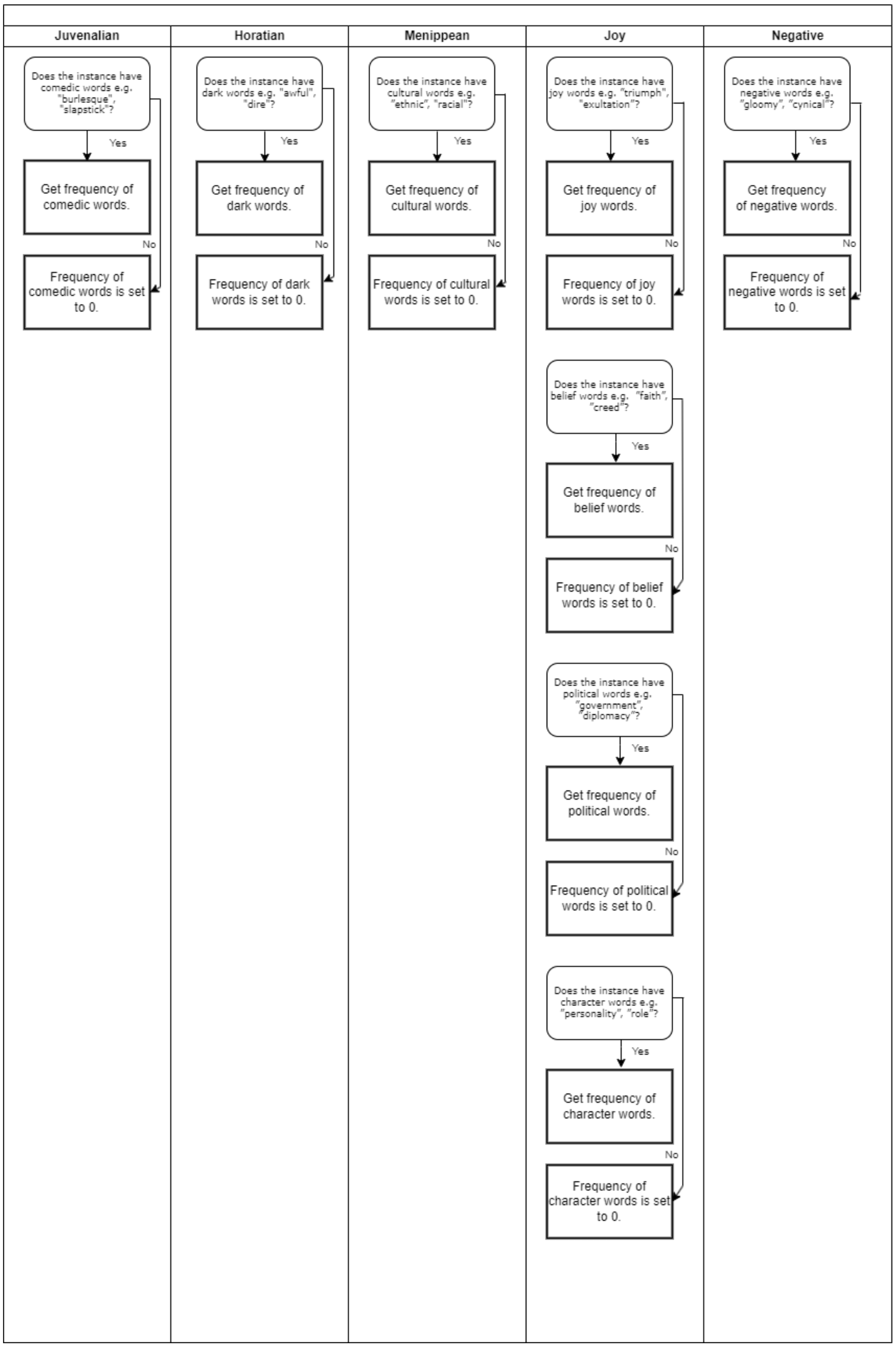

Juvenalian
Horatian
Menippean
Joy
Negative
Does the instance have comedic words e.g. "burlesque", "slapstick"?
Yes
Get frequency of comedic words.
No
Frequency of comedic words is set to 0.
Does the instance have dark words e.g. "awful", "dire"?
Get frequency of dark words.
Frequency of dark words is set to 0.
Does the instance have cultural words e.g. "ethnic", "racial"?
Get frequency of cultural words.
Frequency of cultural words is set to 0.
Does the instance have joy words e.g. "triumph", "exultation"?
Get frequency of joy words.
Frequency of joy words is set to 0.
Does the instance have belief words e.g. "faith", "creed"?
Get frequency of belief words.
Frequency of belief words is set to 0.
Does the instance have political words e.g. "government", "diplomacy"?
Get frequency of political words.
Frequency of political words is set to 0.
Does the instance have character words e.g. "personality", "role"?
Get frequency of character words.
Frequency of character words is set to 0.
Does the instance have negative words e.g. "gloomy", "cynical"?
Get frequency of negative words.
Frequency of negative words is set to 0.

Figure 6.3: Manually Extracted Features for Satire Detection

6.3 Results and Discussion

All the results yielded by the experiments in this work are shown in this section. The way each element of this work is used is also explained.

6.3.1 Classification Results

Every result yielded from all classification algorithms used in the experiments are shown in this subsection. These are the results of all the feature sets combined. The metrics used are Precision, Recall, F1-measure and Accuracy.

Table 6.1: Performance Comparison for Classification Algorithms using all the Feature Sets combined for Satire Detection

Classifier	Precision	Recall	F1	Accuracy
SVM	0.91	0.90	0.91	91%
KNN	0.86	0.86	0.86	86%
LR	**0.93**	**0.92**	**0.92**	**92%**
DT	0.90	0.90	0.90	90%
LDA	0.91	0.90	0.91	91%

SVM, DT and LDA are all good classifiers for this task, with high F1-measures, precision and recall. KNN also does not fall too far below. However, the performance of LR is highest in this work. Hence, from here on only LR results are used in this chapter to represent this work. The most optimum learning curve for this is also shown in Figure 6.4 below.

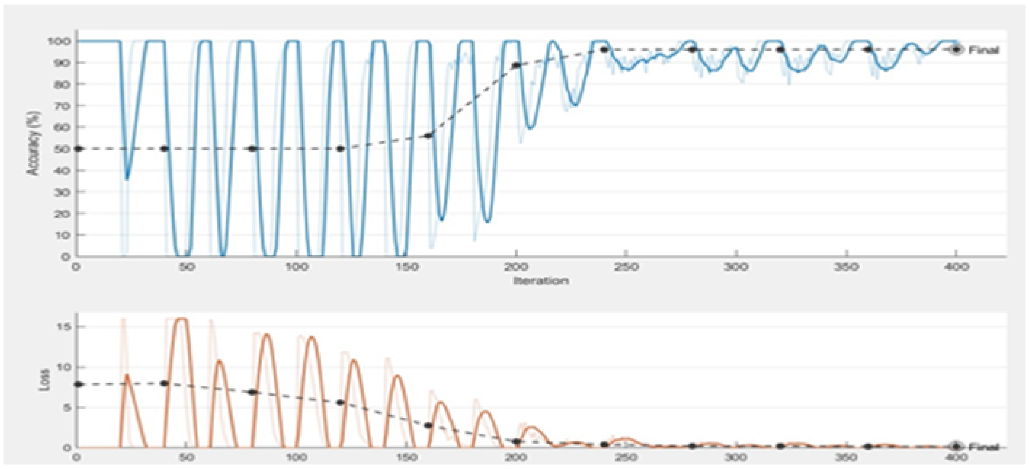


Figure 6.4: Learning Curve using LR for Satire Detection

6.3.2 Performance Comparison with Existing Works

Three existing works in the same domain and experimented using the same dataset are chosen. The results are compiled together with the results of this work and shown in Table 6.2. The original comparison is made and shown in the work by Yang et al. [146] with the additional from Zhang et al. [145].

In comparison with all the other existing works in the same domain stated, the method proposed in this paper shows better performance in all metrics used. This confirms our submission that manually handcrafted contextual features are useful for this specific task.

Table 6.2: Performance Comparison with Existing Satire Detection Works

Method	Precision	Recall	F1	Accuracy
Rubin et al.[38]	0.93	0.83	0.88	85%
Yang et al.[146]	0.94	0.90	0.91	92%
Zhang et al.[145]	0.93	0.87	0.90	91%
Proposed Method	**0.93**	**0.92**	**0.92**	**92%**

6.3.3 Performance comparison among Feature Sets

The results of each of the feature set is shown in this subsection as separate entities, as a way of comparison and additional information. They are shown in Figure 6.5 in terms of F1-measure.

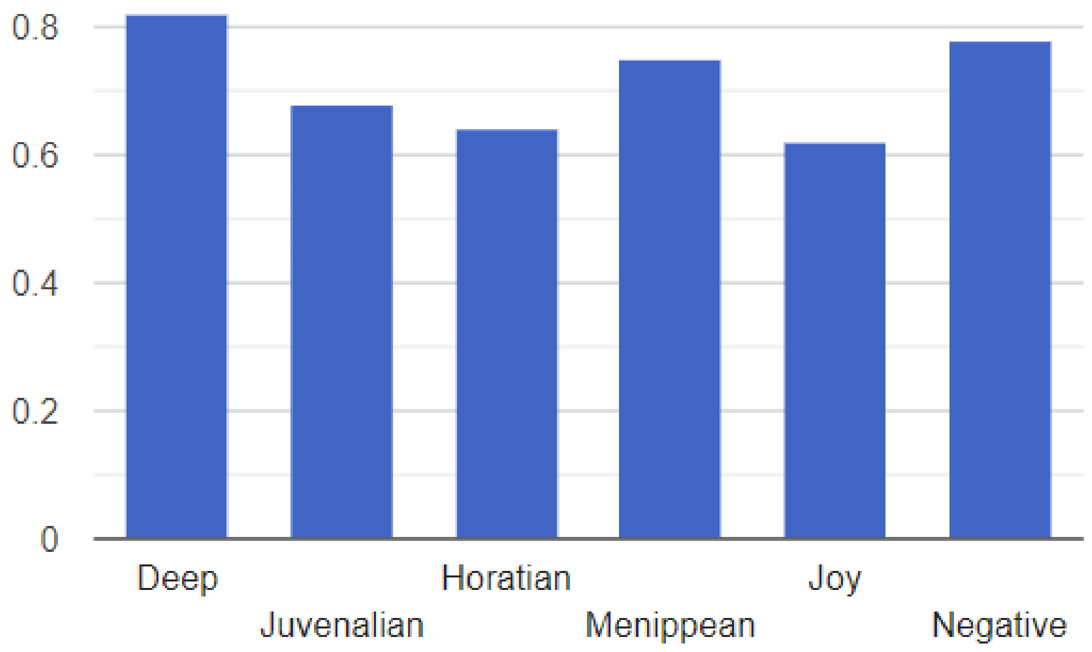


Figure 6.5: Performance of Each Feature Set for Satire Detection

The performance of the deep feature set shows to be the single set that has the highest performance, with 0.82. The Menippean and negative feature sets are almost the same to each other with 0.75 and 0.78 respectively. The Juvenalian, Horatian and joy feature sets show lower performance than the others with 0.68, 0.64 and 0.62 respectively. Through observation, this phenomenon happens because the instances that used these types of satire are rare. However, the F1-measure given by these feature sets is more than chance (equivalent to 0.5), showing the importance of such features for the task [183]. Even though it is the least important as compared to the others as a stand-alone feature, it is obvious that it has some contextual merit and when used in a combination.

6.4 Conclusion

This work dives into the contextual justifications of each feature sets that are manually handcrafted. It gives a sense of how satire works in the real world. These sets are then combined with the automatically extracted set using a deep learning architecture. Even though each of the manually handcrafted feature sets are proven not to be as effective as the one set extracted from the deep learning architecture, they are still significant. This is evident by the increased performance when all of them are combined- the deep feature set and the manually handcrafted feature sets.

The proposed method is shown to significantly improve in results as compared to other works in the same domain which mainly focus on using just deep learning or manually handcrafted methods. For future work, the process of finding the contextual features through the right understanding of satire can be expanded.

Chapter 7

Conclusion and Future Works

7.1 Conclusion

This chapter presents the conclusion of the whole thesis and describes future work that can be undertaken. In this thesis, the issue of finding the most optimal features to detect three types of figurative language is investigated. This was done with the emphasis on the way to integrate the context of the figurative language type with deep learning. The main objective is to develop the sarcasm, metaphor and satire detection methods using deep learning with contextual features. In this thesis, an improvement of the existing detection methods was found.

The proposed models outperform the existing models in all the metrics used: F1-measure, precision and recall. The results of the experiments demonstrated the ability of the proposed model to make use of the manual features to assist the deep features using carefully justified contexts based on the figurative language nature. Then, available machine learning algorithms were used in an exploratory way by differentiating the performance of one to another. Furthermore, this research offers a deeper look into each of the feature set contribution in each of the detection models. Experimental results have shown that the proposed model proved that combining contextual features with deep features is a good strategy.

The results show that the proposed techniques can enhance the detection performance in all metrics used in the work. In summary, the following can be considered as the main contribution of this thesis:

1. The extraction of the most essential and justified contextual-based feature sets for each of the figurative language category detection tasks (sarcasm detection, metaphor detection, satire detection).

2. The design of effective combinations of deep features and manually extracted features.

3. The experimentation and comparison of outcomes for Machine Learning classification models in relation to FL detection tasks.

4. The discovery of the most to the least important features for each FL type.

7.2 Future Works

Based on the observations and results of this work, some of the related and possible future works are:

1. More datasets can be used to further generalize the results.

2. Novel datasets can be created to further generalize the results.

3. Data augmentation techniques such as oversampling and undersampling could be discussed as possible approaches to overcome dataset limitations.

4. The process of extracting and gathering meaningful features can be expanded.

5. The detection methods can be extended to other figurative language categories or NLP problems.

6. The detection methods can experiment with the combination of figurative and literal languages.

7. The contextual features can experiment with other deep learning techniques

8. The whole framework can experiment with computational complexity.

Even though the datasets used in this research work were made of tweets and news articles; the proposed methods can be applied to extract the features and experiments on other types of texts. This is as long as the texts are having the mentioned FL types.

First of all, a researcher who is interested in pursuing research in the same direction should know where or how to get the data. This is mentioned in each FL detection chapters under the subsection Data Acquisition. When the data is successfully accessed, the researcher should observe the data, make sense of it and simply make some assumptions. Some of these assumptions are going to be the base of the experiments.

Then, the running of the experiments would require the researcher to understand existing methods that were used previously. After the understanding is the time to try out some new ways.

These new ways will become the proposed method and main contributions of the research, provided that they can uplift the domain with more effective ways. This is what this work is all about. Contributing to the domain, not only in the improved results but also in new ways of achieving the results.

Bibliography

(1) A. Ghosh and T. Veale, “Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal”, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, 482–491.

(2) J. Aquino, “Transforming social media data into predictive analytics”, *CRM Magazine*, 2012, **16**, 38–42.

(3) B. Liu, “Sentiment analysis and opinion mining”, *Synthesis lectures on human language technologies*, 2012, **5**, 1–167.

(4) A. Joshi, P. Bhattacharyya and M. J. Carman, “Automatic sarcasm detection: A survey”, *ACM Computing Surveys (CSUR)*, 2017, **50**, 1–22.

(5) M. Abulaish, A. Kamal and M. J. Zaki, “A survey of figurative language and its computational detection in online social networks”, *ACM Transactions on the Web (TWEB)*, 2020, **14**, 1–52.

(6) R. M. Roberts and R. J. Kreuz, “Why do people use figurative language?”, *Psychological science*, 1994, **5**, 159–163.

(7) T. Chakrabarty, Y. Choi and V. Shwartz, “It’s not Rocket Science: Interpreting Figurative Language in Narratives”, *Transactions of the Association for Computational Linguistics*, 2022, **10**, 589–606.

(8) E. Shutova, L. Sun and A. Korhonen, “Metaphor identification using verb and noun clustering”, *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, 2010, 1002–1010.

(9) E. Filatova, “Irony and Sarcasm: Corpus Generation and Analysis Using Crowdsourcing.”, *Lrec*, 2012, 392–398.

(10) A. Joshi, V. Sharma and P. Bhattacharyya, “Harnessing context incongruity for sarcasm detection”, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2015, 757–762.

(11) E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert and R. Huang, “Sarcasm as contrast between a positive sentiment and negative situation”, *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, 704–714.

(12) H. H. Clark and R. J. Gerrig, “On the pretense theory of irony.”, 1984.

(13) S. Kumon-Nakamura, S. Glucksberg and M. Brown, “How about another piece of pie: The allusional pretense theory of discourse irony”, *Irony in language and thought*, 2007, 57–96.

(14) A. Qadir, E. Riloff and M. A. Walker, *Automatically inferring implicit properties in similes*, 2016.

(15) H. Driessen, “Humor, anthropology of”, *Wright, JD (ed.), International encyclopedia of the social & behavioral sciences (2nd ed.)*, 2015, 416–419.

(16) M. McCarthy and R. Carter, ““There’s millions of them”: hyperbole in everyday conversation”, *Journal of pragmatics*, 2004, **36**, 149–184.

(17) G. Orwell, “Politics vs. Literature: an examination of Gulliver’s Travels”, *Fair Liberty was all his Cry*, 1967, 166–185.

(18) J. Swift, H. Davis and H. Williams, *Gulliver’s travels: 1726*, Blackwell, 1959.

(19) S. Poria, E. Cambria, D. Hazarika and P. Vij, “A deeper look into sarcastic tweets using deep convolutional neural networks”, *arXiv preprint arXiv:1610.08815*, 2016.

(20) R. Misra and P. Arora, “Sarcasm detection using hybrid neural network”, *arXiv preprint arXiv:1908.07414*, 2019.

(21) C. Yin, Y. Chen and W. Zuo, “Multi-Task Deep Neural Networks for Joint Sarcasm Detection and Sentiment Analysis”, *Pattern Recognition and Image Analysis*, 2021, **31**, 103–108.

(22) D. Davidov, O. Tsur and A. Rappoport, “Semi-supervised recognition of sarcasm in Twitter and Amazon”, *Proceedings of the fourteenth conference on computational natural language learning*, 2010, 107–116.

(23) O. Tsur, D. Davidov and A. Rappoport, “ICWSM—a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews”, *fourth international AAAI conference on weblogs and social media*, 2010.

(24) S. Lukin and M. Walker, “Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue”, *arXiv preprint arXiv:1708.08572*, 2017.

(25) A. Ghosh and T. Veale, “Fracking sarcasm using neural network”, *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, 2016, 161–169.

(26) S. K. Bharti, K. S. Babu and S. K. Jena, “Parsing-based sarcasm sentiment recognition in twitter data”, *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2015, 1373–1380.

(27) M. Bouazizi and T. O. Ohtsuki, “A pattern-based approach for sarcasm detection on twitter”, *IEEE Access*, 2016, **4**, 5477–5488.

(28) T. Veale, *Exploding the creativity myth: The computational foundations of linguistic creativity*, A&C Black, 2012.

(29) D. G. Maynard and M. A. Greenwood, “Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis”, *Lrec 2014 proceedings*, 2014.

(30) D. Bamman and N. Smith, “Contextualized sarcasm detection on twitter”, *proceedings of the international AAAI conference on web and social media*, 2015, **9**, 574–577.

(31) A. Ghosh, G. Li, T. Veale, P. Rosso, E. Shutova, J. Barnden and A. Reyes, “Semeval-2015 task 11: Sentiment analysis of figurative language in twitter”, *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, 2015, 470–478.

(32) A. Rajadesingan, R. Zafarani and H. Liu, “Sarcasm detection on twitter: A behavioral modeling approach”, *Proceedings of the eighth ACM international conference on web search and data mining*, 2015, 97–106.

(33) R. Schifanella, P. de Juan, J. Tetreault and L. Cao, “Detecting sarcasm in multimodal social platforms”, *Proceedings of the 24th ACM international conference on Multimedia*, 2016, 1136–1145.

(34) M. Mohler, D. Bracewell, M. Tomlinson and D. Hinote, “Semantic signatures for example-based linguistic metaphor detection”, *Proceedings of the First Workshop on Metaphor in NLP*, 2013, 27–35.

(35) Z. J. Mason, “CorMet: a computational, corpus-based conventional metaphor extraction system”, *Computational linguistics*, 2004, **30**, 23–44.

(36) M. Abulaish, N. Kumari, M. Fazil and B. Singh, “A graph-theoretic embedding-based approach for rumor detection in twitter”, *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, 466–470.

(37) F. Barbieri, F. Ronzano and H. Saggion, “Is this tweet satirical? A computational approach for satire detection in Spanish”, *Procesamiento del Lenguaje Natural*, 2015, 135–142.

(38) V. L. Rubin, N. Conroy, Y. Chen and S. Cornwell, “Fake news or truth? using satirical cues to detect potentially misleading news”, *Proceedings of the second workshop on computational approaches to deception detection*, 2016, 7–17.

(39) G. Guibon, L. Ermakova, H. Seffih, A. Firsov and G. Le Noé-Bienvenu, “Multi-lingual Fake News Detection with Satire”, *CICLing: International Conference on Computational Linguistics and Intelligent Text Processing*, 2019.

(40) N. Frye, “The nature of satire”, *University of Toronto Quarterly*, 1944, **14**, 75–89.

(41) M. C. Beardsley, “The metaphorical twist”, *Philosophy and phenomenological research*, 1962, 293–307.

(42) L. Cano Mora, “At the risk of exaggerating: how do listeners react to hyperbole?”, 2003.

(43) N. Aljadaan, “Understanding Hyperbole”, *Arab World English Journal (October, 2018), Theses ID*, 2018, **212**.

(44) M. Abulaish and A. Kamal, “Self-deprecating sarcasm detection: an amalgamation of rule-based and machine learning approach”, *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, 2018, 574–579.

(45) F. Barbieri, F. Ronzano and H. Saggion, “Do we criticise (and laugh) in the same way? Automatic detection of multi-lingual satirical news in Twitter”, *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

(46) C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, Springer, 2006, vol. 4.

(47) S. Russel and P. Norvig, “Artificial intelligence: A modern approach, 2003”, *EUA: Prentice Hall*, **178**.

(48) G. Cohen, S. Afshar, J. Tapson and A. Van Schaik, “EMNIST: Extending MNIST to handwritten letters”, *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, 2921–2926.

(49) A. Al Bataineh and D. Kaur, “A comparative study of different curve fitting algorithms in artificial neural network using housing dataset”, *NAECON 2018-IEEE National Aerospace and Electronics Conference*, 2018, 174–178.

(50) J. Franklin, “The elements of statistical learning: data mining, inference and prediction”, *The Mathematical Intelligencer*, 2005, **27**, 83–85.

(51) I. Goodfellow, Y. Bengio and A. Courville, *Deep learning*, MIT press, 2016.

(52) M. A. Wiering and M. Van Otterlo, “Reinforcement learning”, *Adaptation, learning, and optimization*, 2012, **12**.

(53) C. Grosan and A. Abraham, *Intelligent systems*, Springer, 2011.

(54) C. Sin-Wai, “The Routledge encyclopedia of translation technology”, *Abingdon: Routledge*, 2015.

(55) F. Cabitza and B. Dal Seno, “DJess-A Knowledge-Sharing Middleware to Deploy Distributed Inference Systems.”, *WEC (2)*, 2005, 66–69.

(56) P. A. Rockwell, *Sarcasm and other mixed messages: The ambiguous ways people use language*, Edwin Mellen Press, 2006.

(57) J. D. Campbell, “Investigating The Necessary Components of a Sarcastic Context”, 2012.

(58) B. Liu and L. Zhang, “A survey of opinion mining and sentiment analysis”, *Mining text data*, 2012, 415–463.

(59) N. Majumder, S. Poria, H. Peng, N. Chhaya, E. Cambria and A. Gelbukh, “Sentiment and sarcasm classification with multitask learning”, *IEEE Intelligent Systems*, 2019, **34**, 38–43.

(60) P. K. Kirkpatrick, “2016: The 22nd Conference of the Australasian Humour Studies Network (AHSN)”, *26th Australasian Humour Studies Network Conference*, 2016.

(61) H. Cason, “Methods of preventing and eliminating annoyances.”, *The Journal of Abnormal and Social Psychology*, 1930, **25**, 40.

(62) S. Attardo, J. Eisterhold, J. Hay and I. Poggi, “Multimodal markers of irony and sarcasm”, *Humor*, 2003, **16**, 243–260.

(63) R. González-Ibánez, S. Muresan and N. Wacholder, “Identifying sarcasm in twitter: a closer look”, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, 581–586.

(64) R. Simpson, *The School of Shakespeare...* JW Bouton, 1878, vol. 2.

(65) R. Expectans, “MR. WINSTON CHURCHILL AND DEMOCRACY.”, *Westminster review, Jan. 1852-Jan. 1914*, 1906, **165**, 15–21.

(66) S. Housley, “The Evolution of Parliament.”, *Strand magazine: an illustrated monthly*, 1896, **11**, 104–112.

(67) N. Ramakrishnan, *Reading Gandhi in the Twenty-First Century*, Springer, 2013.

(68) R. Reagan, *Public Papers of the Presidents of the United States: Ronald Reagan, 1984*, Best Books on, 1986.

(69) A. Lincoln, *The collected works of Abraham Lincoln*, Wildside Press LLC, 2008, vol. 2.

(70) T. Carlyle, “Note on the Text”, *Sartor Resartus*, 2020, XCV–CXLIV.

(71) J. Kruger, N. Epley, J. Parker and Z.-W. Ng, “Egocentrism over e-mail: Can we communicate as well as we think?”, *Journal of personality and social psychology*, 2005, **89**, 925.

(72) S. Joy, “Lost in translation: Emotion and expression through technology”, 2009.

(73) B. Klingner, “Respond Cautiously to North Korean Engagement Offers.”, *Heritage Foundation*, 2015.

(74) A. Panda, *Kim Jong Un and the bomb: Survival and deterrence in North Korea*, Oxford University Press, 2020.

(75) S. G. Shamay-Tsoory, R. Tomer and J. Aharon-Peretz, “The neuroanatomical basis of understanding sarcasm and its relationship to social cognition.”, *Neuropsychology*, 2005, **19**, 288.

(76) W. Brant, “Critique of sarcastic reason: the epistemology of the cognitive neurological ability called “theory-of-mind” and deceptive reasoning”, 2012.

(77) H. S. Cheang and M. D. Pell, “Acoustic markers of sarcasm in Cantonese and English”, *The Journal of the Acoustical Society of America*, 2009, **126**, 1394–1405.

(78) W. Leslau, *Reference grammar of Amharic*, Otto Harrassowitz Verlag, 1995.

(79) K. Houston, *Shady characters: The secret life of punctuation, symbols, and other typographical marks*, WW Norton & Company, 2013.

(80) S. Wilcock, Ph.D. Thesis, University of Johannesburg, 2013.

(81) C. Liebrecht, F. Kunneman and A. van Den Bosch, “The perfect solution for detecting sarcasm in tweets# not”, 2013.

(82) F. Barbieri, H. Saggion and F. Ronzano, “Modelling sarcasm in twitter, a novel approach”, *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2014, 50–58.

(83) B. Shmueli, L.-W. Ku and S. Ray, “Reactive Supervision: A New Method for Collecting Sarcasm Data”, *arXiv preprint arXiv:2009.13080*, 2020.

(84) S. Ilić, E. Marrese-Taylor, J. A. Balazs and Y. Matsuo, “Deep contextualized word representations for detecting sarcasm and irony”, *arXiv preprint arXiv:1809.09795*, 2018.

(85) M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, “Deep contextualized word representations”, *arXiv preprint arXiv:1802.05365*, 2018.

(86) A. Baruah, K. Das, F. Barbhuiya and K. Dey, “Context-aware sarcasm detection using BERT”, *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, 83–87.

(87) R. Kreuz and G. Caucci, “Lexical influences on the perception of sarcasm”, *Proceedings of the Workshop on computational approaches to Figurative Language*, 2007, 1–4.

(88) S. Amir, B. C. Wallace, H. Lyu and P. C. M. J. Silva, “Modelling context with user embeddings for sarcasm detection in social media”, *arXiv preprint arXiv:1607.00976*, 2016.

(89) C. Fellbaum, “WordNet”, *Theory and applications of ontology: computer applications*, 2010, 231–243.

(90) A. Joshi, S. Agrawal, P. Bhattacharyya and M. J. Carman, “Expect the unexpected: Harnessing sentence completion for sarcasm detection”, *International Conference of the Pacific Association for Computational Linguistics*, 2017, 275–287.

(91) A. Reyes, P. Rosso and D. Buscaldi, “From humor recognition to irony detection: The figurative language of social media”, *Data & Knowledge Engineering*, 2012, **74**, 1–12.

(92) D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann and R. Mihalcea, “Cascade: Contextual sarcasm detection in online discussion forums”, *arXiv preprint arXiv:1805.06413*, 2018.

(93) M. Ebrahimi, A. H. Yazdavar and A. Sheth, “Challenges of sentiment analysis for dynamic events”, *IEEE Intelligent Systems*, 2017, **32**, 70–75.

(94) L. Peled and R. Reichart, “Sarcasm sign: Interpreting sarcasm with sentiment based monolingual machine translation”, *arXiv preprint arXiv:1704.06836*, 2017.

(95) A. D. Hepburn, *Manual of English rhetoric*, Wilson, Hinkle & Company, 1875.

(96) O. Zayed, J. P. McCrae and P. Buitelaar, “Phrase-level metaphor identification using distributed representations of word meaning”, *Proceedings of the Workshop on Figurative Language Processing*, 2018, 81–90.

(97) P. Turney, Y. Neuman, D. Assaf and Y. Cohen, “Literal and metaphorical sense identification through concrete and abstract context”, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011, 680–690.

(98) H. Jang, S. Moon, Y. Jo and C. Rose, “Metaphor detection in discourse”, *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2015, 384–392.

(99) C. Su, F. Fukumoto, X. Huang, J. Li, R. Wang and Z. Chen, “DeepMet: A Reading Comprehension Paradigm for Token-level Metaphor Detection”, *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, 30–39.

(100) A. T. Rivera, A. Oliver, S. Climent and M. Coll-Florit, “Neural metaphor detection with a residual bilstm-crf model”, *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, 197–203.

(101) H. Jang, Y. Jo, Q. Shen, M. Miller, S. Moon and C. Rose, “Metaphor detection with topic transition, emotion and cognition in context”, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, 216–225.

(102) G. Sam and H. Catrinel, “On the relation between metaphor and simile: When comparison fails”, *Mind & Language*, 2006, **21**, 360–378.

(103) W. Liebeschuetz, “Beast and man in the third book of Virgil’s Georgics”, *Greece & Rome*, 1965, **12**, 64–77.

(104) P. L. Hays, “Tennessee Williams’ Use of Myth in” Sweet Bird of Youth””, *Educational Theatre Journal*, 1966, 255–258.

(105) M. Mack, “The world of Hamlet”, *Yale Review*, 1952, **41**, 23.

(106) W. Shakespeare, *The complete works of William Shakespeare*, Wordsworth Editions, 2007.

(107) I. A. Richards and J. Constable, *The philosophy of rhetoric*, Routledge, 2018.

(108) J. Jaynes, *The origin of consciousness in the breakdown of the bicameral mind*, Houghton Mifflin Harcourt, 2000.

(109) A. Liberman, *Word origins... and how we know them: Etymology for everyone*, OUP USA, 2009.

(110) T. McArthur, J. Lam-McArthur and L. Fontaine, *Oxford companion to the English language*, Oxford University Press, 2018.

(111) B. B. Klebanov, B. Leong, M. Heilman and M. Flor, “Different texts, same metaphors: Unigrams and beyond”, *Proceedings of the Second Workshop on Metaphor in NLP*, 2014, 11–17.

(112) Y. Tsvetkov, L. Boytsov, A. Gershman, E. Nyberg and C. Dyer, “Metaphor detection with cross-lingual model transfer”, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, 248–258.

(113) E.-L. Do Dinh and I. Gurevych, “Token-level metaphor detection using neural networks”, *Proceedings of the Fourth Workshop on Metaphor in NLP*, 2016, 28–33.

(114) Y. Bizzoni and M. Ghanimifard, “Bigrams and BiLSTMs two neural networks for sequential metaphor detection”, *Proceedings of the Workshop on Figurative Language Processing*, 2018, 91–101.

(115) K. Swarnkar and A. K. Singh, “Di-LSTM contrast: A deep neural network for metaphor detection”, *Proceedings of the Workshop on Figurative Language Processing*, 2018, 115–120.

(116) S. Hochreiter and J. Schmidhuber, “LSTM can solve hard long time lag problems”, *Advances in neural information processing systems*, 1997, 473–479.

(117) Y. Wilks, A. Dalton, J. Allen and L. Galescu, “Automatic metaphor detection using large-scale lexical resources and conventional metaphor extraction”, *Proceedings of the First Workshop on Metaphor in NLP*, 2013, 36–44.

(118) K. K. Schuler, *VerbNet: A broad-coverage, comprehensive verb lexicon*, University of Pennsylvania, 2005.

(119) J. Golbeck, M. Mauriello, B. Auxier, K. H. Bhanushali, C. Bonk, M. A. Bouzaghrane, C. Buntain, R. Chanduka, P. Cheakalos, J. B. Everett et al., “Fake news vs satire: A dataset and analysis”, *Proceedings of the 10th ACM Conference on Web Science*, 2018, 17–21.

(120) F. V. Bogel, *The difference satire makes*, Cornell University Press, 2019.

(121) Z. P. Biles, “Intertextual biography in the rivalry of Cratinus and Aristophanes”, *American journal of philology*, 2002, **123**, 169–204.

(122) B. L. Ullman, “Satura and satire”, *Classical Philology*, 1913, **8**, 172–194.

(123) L. R. Shero, “The Cena in Roman Satire”, *Classical Philology*, 1923, **18**, 126–143.

(124) M. De Cervantes, *Don Quixote*, Lulu. com, 2016.

(125) D. Musgrave, *Grotesque Anatomies: Menippean Satire Since the Renaissance*, Cambridge Scholars Publishing, 2014.

(126) G. Orwell and A. Heath, *Animal farm and 1984*, Houghton Mifflin Harcourt, 2003.

(127) J. Fife, “Peeling The Onion: Satire and the complexity of audience response”, *Rhetoric Review*, 2016, **35**, 322–334.

(128) N. Diehl, “Satire, analogy, and moral philosophy”, *The Journal of Aesthetics and Art Criticism*, 2013, **71**, 311–321.

(129) J. Thorogood, "Satire and geopolitics: Vulgarity, ambiguity and the body grotesque in South Park", *Geopolitics*, 2016, **21**, 215–235.

(130) J. C. Baumgartner and B. Lockerbie, "Maybe it is more than a joke: Satire, mobilization, and political participation", *Social Science Quarterly*, 2018, **99**, 1060–1074.

(131) A. Bigi, K. Plangger, M. Bonera and C. L. Campbell, "When satire is serious: how political cartoons impact a country's brand", *Journal of Public Affairs*, 2011, **11**, 148–155.

(132) T. Simon, *Jupiter's travels*, Penguin UK, 2007.

(133) P. Matthiessen, *The snow leopard*, Penguin, 2008.

(134) B. Chatwin, *In Patagonia*, Random House, 2012.

(135) G. Orwell, *The Orwell reader: Fiction, essays, and reportage*, Houghton Mifflin Harcourt, 1984.

(136) L. Carrol, *Alice in Wonderland and Through the Looking-glass*, Lulu. com, 1939.

(137) A. Day and E. Thompson, "Live from New York, it's the fake news! Saturday Night Live and the (non) politics of parody", *Popular Communication*, 2012, **10**, 170–182.

(138) M. del Pilar Salas-Zárate, M. A. Paredes-Valverde, M. Á. Rodriguez-Garcıa, R. Valencia-Garcıa and G. Alor-Hernández, "Automatic detection of satire in Twitter: A psycholinguistic-based approach", *Knowledge-Based Systems*, 2017, **128**, 20–33.

(139) C. Burfoot and T. Baldwin, "Automatic satire detection: Are you having a laugh?", *Proceedings of the ACL-IJCNLP 2009 conference short papers*, 2009, 161–164.

(140) A. N. Reganti, T. Maheshwari, U. Kumar, A. Das and R. Bajpai, "Modeling satire in English text for automatic detection", *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, 2016, 970–977.

(141) P. P. Thu and N. New, "Implementation of emotional features on satire detection", *2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2017, 149–154.

(142) R. McHardy, H. Adel and R. Klinger, "Adversarial training for satire detection: Controlling for confounding variables", *arXiv preprint arXiv:1902.11145*, 2019.

(143) S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, 1997, **9**, 1735–1780.

(144) Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou and Y. Bengio, "A structured self-attentive sentence embedding", *arXiv preprint arXiv:1703.03130*, 2017.

(145) Y. Zhang, F. Yang, Y. Zhang, E. Dragut and A. Mukherjee, "Birds of a feather flock together: Satirical news detection via language model differentiation", *arXiv preprint arXiv:2007.02164*, 2020.

(146) F. Yang, A. Mukherjee and E. Dragut, "Satirical news detection and analysis using attention mechanism and linguistic features", *arXiv preprint arXiv:1709.01189*, 2017.

(147) J. Tan, J. Yang, S. Wu, G. Chen and J. Zhao, "A critical look at the current train/test split in machine learning", *arXiv preprint arXiv:2106.04525*, 2021.

(148) Y. LeCun, Y. Bengio et al., "Convolutional networks for images, speech, and time series", *The handbook of brain theory and neural networks*, 1995, **3361**, 1995.

(149) J. L. Elman, "Finding structure in time", *Cognitive science*, 1990, **14**, 179–211.

(150) N. T. Vu, H. Adel, P. Gupta and H. Schütze, "Combining recurrent and convolutional neural networks for relation classification", *arXiv preprint arXiv:1605.07333*, 2016.

(151) W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang and J. Suh, "The value of semantic parse labeling for knowledge base question answering", *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, 201–206.

(152) H. Adel and H. Schütze, "Exploring different dimensions of attention for uncertainty detection", *arXiv preprint arXiv:1612.06549*, 2016.

(153) Y. N. Dauphin, A. Fan, M. Auli and D. Grangier, "Language modeling with gated convolutional networks", *International conference on machine learning*, 2017, 933–941.

(154) Y. Kim, "Convolutional Neural Networks for Sentence Classification", 2014.

(155) W. Yin, H. Schütze, B. Xiang and B. Zhou, "Abcnn: Attention-based convolutional neural network for modeling sentence pairs", *Transactions of the Association for Computational Linguistics*, 2016, **4**, 259–272.

(156) N. Kalchbrenner, E. Grefenstette and P. Blunsom, "A convolutional neural network for modelling sentences", *arXiv preprint arXiv:1404.2188*, 2014.

(157) S. Santurkar, D. Tsipras, A. Ilyas and A. Madry, "How does batch normalization help optimization?", *Proceedings of the 32nd international conference on neural information processing systems*, 2018, 2488–2498.

(158) Y. Zhang and B. Wallace, “A Sensitivity Analysis of (and Practitioners’ Guide to) Convolutional Neural Networks for Sentence Classification”, 2016.

(159) A. F. Agarap, “Deep learning using rectified linear units (relu)”, *arXiv preprint arXiv:1803.08375*, 2018.

(160) P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, “Enriching word vectors with subword information”, *Transactions of the Association for Computational Linguistics*, 2017, **5**, 135–146.

(161) T. Mikolov, K. Chen, G. Corrado and J. Dean, “Efficient estimation of word representations in vector space”, *arXiv preprint arXiv:1301.3781*, 2013.

(162) V. Vapnik and R. Izmailov, “Knowledge transfer in SVM and neural networks”, *Annals of Mathematics and Artificial Intelligence*, 2017, **81**, 3–19.

(163) L. E. Peterson, “K-nearest neighbor”, *Scholarpedia*, 2009, **4**, 1883.

(164) A. Christmann, “Least median of weighted squares in logistic regression with large strata”, *Biometrika*, 1994, **81**, 413–417.

(165) B. Kamiński, M. Jakubczyk and P. Szufel, “A framework for sensitivity analysis of decision trees”, *Central European journal of operations research*, 2018, **26**, 135–159.

(166) G. D. Garson, “Discriminant function analysis. Statnotes: topics in multivariate analysis”, *Retrieved March*, 2008, **29**, 2010.

(167) J. Huang and C. X. Ling, “Using AUC and accuracy in evaluating learning algorithms”, *IEEE Transactions on knowledge and Data Engineering*, 2005, **17**, 299–310.

(168) P. Rockwell, “Empathy and the expression and recognition of sarcasm by close relations or strangers”, *Perceptual and motor skills*, 2003, **97**, 251–256.

(169) F. Kunneman, C. Liebrecht, M. Van Mulken and A. Van den Bosch, “Signaling sarcasm: From hyperbole to hashtag”, *Information Processing & Management*, 2015, **51**, 500–509.

(170) T. Ptáček, I. Habernal and J. Hong, “Sarcasm detection on czech and english twitter”, *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, 213–223.

(171) S. L. Ivanko and P. M. Pexman, “Context incongruity and irony processing”, *Discourse Processes*, 2003, **35**, 241–279.

(172) J. D. Campbell and A. N. Katz, “Are there necessary conditions for inducing a sense of sarcastic irony?”, *Discourse Processes*, 2012, **49**, 459–480.

(173) A. Ramteke, A. Malu, P. Bhattacharyya and J. S. Nath, "Detecting turnarounds in sentiment analysis: Thwarting", *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2013, 860–865.

(174) S. K. Bharti, B. Vachha, R. Pradhan, K. S. Babu and S. K. Jena, "Sarcastic sentiment detection in tweets streamed in real time: a big data approach", *Digital Communications and Networks*, 2016, **2**, 108–121.

(175) A. Chen and L. Boves, "What's in a word: Sounding sarcastic in British English", *Journal of the International Phonetic Association*, 2018, **48**, 57–76.

(176) B. Liu, M. Hu and J. Cheng, "Opinion observer: analyzing and comparing opinions on the web", *Proceedings of the 14th international conference on World Wide Web*, 2005, 342–351.

(177) E. Troiano, C. Strapparava, G. Özbal and S. S. Tekiroğlu, "A computational exploration of exaggeration", *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, 3296–3304.

(178) yourdictionary.com, "Temporal Words", 2021, Accessed: 2021-02-21.

(179) talkenglish.com, "Noun", 2021, Accessed: 2021-02-21.

(180) really-learn-english.com, "Pronoun", 2021, Accessed: 2021-02-21.

(181) englishclub.com, "Verb", 2021, Accessed: 2021-02-21.

(182) englishclub.com, "IVerb", 2021, Accessed: 2021-02-21.

(183) Z. Zhang and L. Luo, "Hate speech detection: A solved problem? the challenging case of long tail on twitter", *Semantic Web*, 2019, **10**, 925–945.

(184) S. Mohammad, E. Shutova and P. Turney, "Metaphor as a medium for emotion: An empirical study", *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, 2016, 23–33.

(185) B. P. Meier and M. D. Robinson, "The metaphorical representation of affect", *Metaphor and symbol*, 2005, **20**, 239–257.

(186) Z. Kozareva, "Multilingual affect polarity and valence prediction in metaphor-rich texts", *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, 682–691.

(187) J. W. Pennebaker, M. E. Francis and R. J. Booth, "Linguistic inquiry and word count: LIWC 2001", *Mahway: Lawrence Erlbaum Associates*, 2001, **71**, 2001.

(188) sightwordsgame.com, "Positional Words", 2021, Accessed: 2021-05-25.

(189) sightwordsgame.com, "Directional Words", 2021, Accessed: 2021-05-25.

(190) relatedwords.org, "Distance Words", 2021, Accessed: 2021-05-25.

(191) sightwordsgame.com, “Size Words”, 2021, Accessed: 2021-05-25.

(192) R. Mao, C. Lin and F. Guerin, “Word embedding and wordnet based metaphor identification and interpretation”, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, 1222–1231.

(193) M. Rei, L. Bulat, D. Kiela and E. Shutova, “Grasping the finer point: A supervised similarity network for metaphor detection”, *arXiv preprint arXiv:1709.00575*, 2017.

(194) C. A. Knight, *The literature of satire*, Cambridge University Press, 2004.

(195) C. Sanders, *The scope of satire*, Scott, Foresman, 1971.

(196) T. Erwin, “Menippean Satire Reconsidered: From Antiquity to the Eighteenth Century”, 2008.

(197) M. K. Booker, *Flann O’Brien, Bakhtin, and Menippean Satire*, Syracuse University Press, 1995.

(198) H. D. Weinbrot, *Menippean satire reconsidered: From antiquity to the eighteenth century*, JHU Press, 2005.

(199) H. M. Littlefield, “The wizard of Oz: Parable on populism”, *American Quarterly*, 1964, **16**, 47–58.

(200) C. Condren, “Satire and definition”, *Humor*, 2012, **25**, 375–399.