Deep Models for Computational Sarcasm Detection & Interpretation

Submitted in partial fulfillment of the requirements

of the degree of

Master of Technology

by

Abhijeet Dubey

(*Roll no.* 16305R006)

Guided By:

Prof. Pushpak Bhattacharyya (IIT Bombay)

Dr. Aditya Joshi (CSIRO, Sydney)

Department of Computer Science & Engineering

Indian Institute of Technology Bombay

2018

Report Approval

This project report entitled **"Deep Models for Computational Sarcasm Detection & Interpretation"**, submitted by Abhijeet Dubey (Roll No. 16305R006), is approved for the award of degree of Master of Technology in Computer Science & Engineering.

Examiner

Prof. Om P. Damani ____________________

Guide

Prof. Pushpak Bhattacharyya (IIT Bombay) ____________________

Date: ...... October 2018

Place: ______________________________

Declaration of Authorship

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature: ....

Abhijeet Dubey

16305R006

Date: ...... October 2018

Abstract

Sarcasm is a form of speech in which the the implied sentiment is the opposite of literal meaning. Sarcasm is a cutting, often ironic remark intended to express contempt or ridicule. Even the current state of the art sentiment analysis system fails when the input is a sarcastic text. Computational sarcasm detection refers to computational techniques to detect sarcasm. In this report we present approaches for sarcasm detection and interpretation.

Research in sarcasm detection spans almost a decade. However, a peculiar form of sarcasm remains unexplored: sarcasm expressed through numbers. The sentence 'Love waking up at 3 am' is sarcastic because of the number. In this report, we evaluate three typical NLP paradigms for the task: (a) a rule-based approach that estimates a decision threshold value of a number that causes the sentence to be sarcastic, (b) a statistical machine learning (ML) approach that uses indicators of such sarcasm, (c) three typical deep learning-based (DL) architectures, CNN, LSTM based and attention networks. The best performing systems are the DL ones with F-score of 0.93 and 0.84 on two datasets of tweets. We also discuss the scope of future work to further enhance the proposed models for detecting sarcasm due to numbers. To the best of our knowledge, this is the first effort on detecting sarcasm involving numbers.

We also present the task of sarcasm interpretation, defined as converting a sarcastic utterance into its non-sarcastic interpretation. We present three approaches for the task: (a) a rule-based approach that considers sarcasm as a form of dropped negation and associate negation words with verbs present in the sarcastic utterance, (b) statistical machine translation-based (SMT) approach that address the sarcasm interpretation task as monolingual machine translation and (c) three deep learning-based (DL) architectures, Encoder-Decoder Network, Attention Network and Pointer Generator Network. We also discuss the scope of future work to further enhance the proposed models for sarcasm interpretation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The organization of the chapter is as follows. First, we introduce computational sarcasm in section 1.1 and the relationship between irony and sarcasm in section 1.2. Then, in section 1.3 we explain the motivation behind computational sarcasm. section 1.4 describes the problem statement that the project is set to tackle. After that, section 1.5 describes, in brief, the proposed solution to the problem statement, and finally, section 1.6 gives a roadmap to the rest of the report.

1.1 Sarcasm

Wikipedia defines sarcasm as a sharp, bitter, or cutting expression or remark; a bitter gibe or taunt. According to The Free Dictionary[1] sarcasm is a cutting, often ironic remark intended to express contempt or ridicule. Sarcastic sentences have a unique property of having a surface sentiment which is different from the actual implied sentiment. The surface sentiment can be positive. For example, *'I love being ignored'*. Here *'love'* is a positive sentiment bearing word and the phrase *'being ignored'* has negative sentiment and hence *'being ignored'* is an undesired situation. In this example the surface sentiment is positive but implied sentiment is negative. The surface sentiment can be negative. For example, *'His performance in Olympics has been terrible anyway'* as a response to the criticism of an Olympic medalist.[2] The surface sentiment can be neutral as well. For

[1] https://www.thefreedictionary.com/
[2] This example is taken from [20]

example, the idiomatic expression *’and I am the Prime Minister of India’* is used to express sarcasm. Since sarcasm implies sentiment, it is important to detect sarcasm for predicting the correct sentiment. Existing social media analysis systems are hampered by their inability to accurately detect and interpret figurative language. This is particularly relevant in domains like the social sciences and politics, in which the use of figurative communication devices such as verbal irony (roughly, sarcasm) is common. Sarcasm is often used by individuals to express opinions on complex matters and regarding specific targets[1].

1.2 Sarcasm vs Irony

Irony is used to convey, usually, the opposite meaning of the actual things you say, but its purpose is not intended to hurt the other person. Sarcasm, while still keeping the ”characteristic” that you mean the opposite of what you say, unlike irony it is used to taunt the other person or to express contempt or ridicule.

According to *The Sarcasm Society*[3] irony is often naturally occurring and unintentional in nature, while sarcasm uses irony to make a remark or observation of a situation, usually ridiculing it in some fashion. For example, the statement *’a fire station caught fire and burned to the ground’* is ironic because the a fire station is a structure or other area set aside for firefighting and the irony is that the entity which is supposed to put off fire got burned down.

1.3 Motivation for Computational Sarcasm

Following the introduction to sarcasm, we now discuss why computational sarcasm is useful. Computational Sarcasm refers to techniques to process sarcasm that is (a) To detect sarcasm, (b) To interpret sarcasm, and (c) To generate sarcasm.

Sarcastic text is challenging for sentiment classification because of it’s ironic nature. In today’s world sarcasm is ubiquitous. Ranging from social networking websites like Facebook, Twitter etc. to product reviews on e-commerce giants like Amazon, Flipkart

[3]http://sarcasmsociety.com/

etc. sarcasm is present everywhere. Sarcastic humor is downright priceless and applicable to many of our own situations.

Sarcasm is hard to detect and interpret especially non-verbal sarcasm. Sarcasm on internet is hard to detect because of the following reasons:

- Speaker's body language is unknown which is a major part of how people communicate with each other.
- Tone of voice makes a huge difference. Words on a computer screen and face to face conversation are very different.
- Every sentence can be a sarcastic one for a particular context.

All these factors make it difficult to detect sarcasm. This is why understanding the actual meaning from a sarcastic utterance is a very interesting and challenging problem. For understanding sarcasm it is important to first identify if a statement is sarcastic or not. The text can then be further be analyzed to understand the original intention of the speaker.

Due to availability of data and inclination of technology giants towards extracting original meaning from sarcastic text, sarcasm detection and interpretation have become hot topics of research nowadays. Since, manual processing of such a large amount of data is not an easy task, certain Statistical, Machine Learning and Deep Learning based approaches can be used for this purpose. These techniques analyze the text and predict if it is sarcastic or not.

1.4 Problem Statements

The first problem statement of the project can be briefly described by the research question:

"What is the role of numbers in automatic sarcasm detection?"

Towards this, we systematically formulate the problem of detecting sarcasm due to numbers. We present a comparison of three NLP paradigms and implement a sarcasm detection suite for detecting sarcasm in text containing numbers.

The second problem statement of the project is motivated by the fact that sarcasm understanding varies across different cultures. Sarcasm annotation extends beyond linguistic expertise, and often involves cultural context. For example, a sentence identified as sarcastic by a native American, can be identified as non-sarcastic by a native Indian. Therefore, interpretation and understanding of sarcasm is very important. The second problem statement of the project can be briefly described as follows:

"Given a sarcastic utterance, can we extract its literal interpretation?"

Towards this, we introduce the task of sarcasm interpretation, defined as the generation of non-sarcastic text conveying the intended message as the original sarcastic one. Since, manual processing of large amount of data is not an easy task, we implement certain rule-based, statistical machine learning-based and deep learning-based techniques for this purpose. These techniques analyze the text and convert sarcastic text into its non-sarcastic interpretation.

1.5 Proposed Solution

In order to answer the above first research question, a solution is proposed where we evaluate three typical NLP paradigms for the task:

1. A rule-based approach that estimates a decision threshold value of a number that causes the sentence to be sarcastic.

2. A statistical machine learning-based approach that uses indicators of such sarcasm.

3. Three typical deep learning-based architectures, CNN, LSTM based and Attention networks.

To the best of our knowledge, this is the first effort on detecting sarcasm involving numbers.

For our second research question, we implement the following models that convert sarcastic utterances into their non-sarcastic interpretation:

1. Statistical Machine Translation-based approach which uses MOSES.

2. Deep learning-based architectures, Encoder-Decoder Network, Attention Network and Pointer Generator Network.

3. Rule-based approach which uses linguistic theories of sarcasm.

1.6 Roadmap of the Report

The organization of the rest of the report is as follows:

- **Chapter 2** describes the problem of sarcasm detection and interpretation in detail.
- **Chapter 3** gives an overview of incongruity and its role for detecting sarcasm.
- **Chapter 4** basically deals with the previous research works that have been done in the field of sarcasm detection and interpretation.
- **Chapter 5** describes approaches for sarcasm detection.
- **Chapter 6** covers the implementation details of our proposed solution for the problem of sarcasm detection due to numbers. It describes various datasets used for the purpose of experimentation. It also presents results from various models. Finally, we conclude by presenting our error analysis.
- **Chapter 7** covers the implementation details of our proposed solution for the problem of sarcasm interpretation. It describes various datasets used for the purpose of experimentation. It also presents results from various models. Finally, we conclude by presenting our error analysis and visualizations.
- **Chapter 8** concludes this report and present future work.

Summary

This chapter introduces the report by explaining computational sarcasm detection and interpretation and the motivation behind it. It then introduces the two problem statements and the brief description of the proposed solution. Finally, the chapter gives a road-map to the rest of the report.

Chapter 2

Sarcasm Detection & Interpretation

Automatic Sarcasm detection is the task of predicting sarcasm in text. This is a crucial step to sentiment analysis, considering prevalence and challenges of sarcasm in sentiment-bearing text.

In this chapter, we motivate our two problem statements defined in chapter 1. The organization of the chapter is as follows. subsection 2.1.1 motivate the problem of detecting sarcasm due to numbers. subsection 2.1.2 motivate the problem of sarcasm interpretation. section 2.2 describes various linguistic types of sarcasm. section 2.3 lists various applications of sarcasm detection. Finally, we conclude this chapter by discussing problems in sarcasm detection in section 2.4.

2.1 Motivation

2.1.1 Numerical Sarcasm Detection

Computational detection of sarcasm has seen a lot of attention from sentiment analysis community in the past few years.[25]. Past approaches for sarcasm detection report features related to sentiment [16], author's historical context [43], and conversational context [26].

Sarcasm is an infrequent phenomenon in sentiment-bearing text[25]. While several approaches to detect sarcasm have been reported [16, 23], they may fall short in case of sarcasm expressed via numbers. Consider the following sentences:

1. *This phone has an awesome battery backup of 38 hours*
2. *This phone has a terrible battery backup of 2 hours*
3. *This phone has an awesome battery backup of 2 hours*

At the time of writing this report, a battery backup of 38 hours is good for phones while a battery backup of 2 hours is bad. Therefore, sentences 1 and 2 are non-sarcastic because the sentiment of the adjectives (‘*awesome*’ and ‘*terrible*’) conforms with the sentiment associated with the corresponding numerical values. On the contrary, the sarcasm in sentence 3 can be understood in terms of incongruity[1] between the word ‘*awesome*’ and ‘*2 hours*’ for the battery life of the phone. The sarcasm in sentence 3 above occurs because of incompatibility/incongruity between ‘*awesome*’ (positive word) and ‘*2 hours*’ (numerical value).

One may argue that sarcasm in text containing numbers is rare and, hence, does not necessitate consideration from the research community. However, the prevalence of such sarcasm speaks otherwise. We consider a set of approximately 100,000 sarcastic tweets. This set has 11,488 tweets containing numbers, amounting to 11.48%. However, a related question is whether or not tweets containing numbers are sarcastic due to the numbers themselves. An analysis of 500 randomly selected sarcastic tweets containing numbers shows that 79.4% of the tweets are sarcastic due to the numbers present in them.

We note here that there is a subtle difference between ‘*sarcasm in text containing numbers*’ and ‘*sarcasm arising due to numbers in text*’. For example, ‘*Having a head ache at 4 am is the best thing to happen*’ contains a number but is sarcastic without the number as such. For the sake of brevity, we assume that the two are the same. This is a reasonable simplification because majority (79.4%) of these tweets are found to be sarcastic due to the numbers present in them, as indicated above.

2.1.2 Sarcasm Interpretation

In today’s world, sarcasm is ubiquitous, ranging from social networking websites like Facebook, Twitter etc. to product reviews on e-commerce giants like Amazon, Flipkart

[1][32] describe the relationship between incongruity and sarcasm.

etc. sarcasm is present everywhere. Sarcastic humor is downright priceless and applicable to many of our own situations.

However, sarcasm is hard to interpret, especially non-verbal sarcasm. [22] shows that sarcasm may not be understood by people from some cultures. Sarcasm expressed in a native language is difficult to interpret by non native speakers. We describe approaches in this paper that will help in understanding sarcasm. In verbal communication, sarcastic utterances are accompanied by a certain tone of voice and facial expressions (For eg., *rolling of eyes*). However, in textual communication, these cues are absent which makes identification and interpretation of sarcasm very challenging even for humans. Sarcasm on the internet is hard to interpret because of the following reasons:

1. Speaker's body language is unknown which is a major part of how people communicate with each other.

2. Tone of voice makes a huge difference. Words on a computer screen and face to face conversation are very different.

3. Every sentence can be a sarcastic one for a particular context.

All these factors makes it difficult to interpret sarcasm. This is why understanding the actual meaning from a sarcastic utterance is a very interesting and challenging problem. Therefore, to understanding sarcasm, it is important to first convert sarcastic text into its non-sarcastic form. The non-sarcastic form can be analyzed further using existing sentiment analysis systems to understand the original intention of the speaker. Towards this, we introduce the task of sarcasm interpretation, defined as the generation of a non-sarcastic text conveying the intended message as the original sarcastic one. Since, manual processing of large amount of data is not an easy task, certain rule-based, statistical machine learning-based and deep learning-based techniques can be used for this purpose. These techniques analyze the text and convert sarcastic text into its non-sarcastic interpretation.

2.2 Linguistic Types of Sarcasm

Sarcasm can be classified into four main types:

1. **Propositional:** Propositional sarcasm delivers an implication that is the contrary of a proposition that would have been expressed by a sincere utterance. For example, *"James must be a real hit with the ladies."*

2. **Embedded:** This type of sarcasm has an embedded incongruity in the form of words and phrases themselves. Embedded sarcasm is a fairly commonplace and flexible phenomenon. For example, *"Because George has turned out to be such a diplomat, we've decided to transfer him to Payroll, where he'll do less damage."*

3. **Like Prefixed:** Like-prefixed sarcasm targets an entire proposition. This inevitably includes the sentence's focal content, and often only that content. For example, *"Like that's a good idea!"*

4. **Illocutionary:** The scope of this type of sarcasm encompasses not just some element within the uttered sentence, but the entire illocutionary act. This kind of sarcasm involves non-textual clues that indicate an attitude opposite to a sincere utterance. In such cases, non-textual variations play a role. For example, rolling one's eyes while saying *"You sure know a lot."*

Apart from the four main categories of sarcasm defined above, *The Literary Devices*[2] defines seven categories of sarcasm:

1. **Self-Deprecating Sarcasm:** This category of sarcasm expresses an overstated sense of inferiority and worthlessness. For example, "Hey Bob, I'm gonna need you to work overtime this weekend." "Yeah, that's fine. I mean, I was gonna get married this weekend but, you know, it's not a big deal, I'll just skip it. She would've left me anyway."

2. **Brooding Sarcasm:** In this criticism, the speaker utters something polite. However, the tone of his speech has a marked bitterness in it. For example, "Hey Bob, I'm gonna need you to work overtime this weekend." "Looking forward to it. I live to serve."

[2] https://literarydevices.net/sarcasm/

3. **Deadpan Sarcasm:** It is expressed without emotion or laughter making it difficult to judge whether the utterance is joking or mocking. For example, ”Hey Bob, gonna need you to work overtime this weekend.” ”Can’t make it. Got a cult meeting. It’s my turn to kill the goat.”

4. **Polite Sarcasm:** This category of sarcasm is subtle, but just a little too nice. This is a kind of sarcasm that seems genuine at first, but then it slowly dawns on the listener that the speaker was just screwing with him. For example, ”Hey Bob, I’m gonna need you to work overtime this weekend.” ”Ooh, fun! I’ll bring the ice cream!”

5. **Obnoxious Sarcasm:** This category of sarcasm is not really funny or clever, but it gets under your skin. It’s usually spoken in a whiny tone of voice. For example, ”Hey Bob, gonna need you to work overtime.” ”Oh, well that’s just absolutely great. Just what I wanted to do this weekend. Awesome.”

6. **Manic Sarcasm:** This type of sarcasm is delivered in an unnatural happy mood that it makes the speaker look like he has gone crazy. For example, ”Hey Bob, I’m gonna need you to work overtime.” ”God, you are the best boss EVER! Have I ever told you how much I love this job? I wish I could live here! Somebody get me a tent, I never wanna leave!”

7. **Raging Sarcasm:** This category of sarcasm relies heavily on hyperbole and threats of violence. For example, ”Bob. Overtime.” ”Oh, don’t worry! I’ll be there! Want me to shine your shoes while I’m at it?! Hell, I’ll come to your house tonight and wash your goddamn Ferrari! Actually, you know what? Forget it. I’m just gonna go home and blow my brains out.”

2.3 Applications of Sarcasm Detection & Interpretation

There are numerous applications of automatic sarcasm detection and interpretation. Some of them are as follows:

1. Systematic sarcasm detection helps in enhancing the performance of existing sentiment analysis systems.

2. Product reviews on various e-commerce websites are sometimes sarcastic. Automatic sarcasm detection of product reviews will help in gauging customer satisfaction.

3. Automatic sarcasm detection on social media platforms like Twitter and Facebook help in analyzing various situations such as elections.

4. Sarcasm expressed in a native language is difficult to interpret by non native speakers. This is where sarcasm interpretation comes in. We describe approaches in this paper that will help in understanding sarcasm.

2.4 Issues in Sarcasm Detection

In this section, we focus on three recurring design issues that appear in different sarcasm detection works. The first deals with the quality of the **annotation**. The second issue deals with using sentiment as a **feature** for classifcation. Finally, the third issue lies in the context of handling **unbalanced datasets**.

1. Although hashtag-based labeling can provide large-scale supervision, the quality of the dataset may be dubious. This is particularly true in the case of using not to indicate insincere sentiment. [34] show that not is often used to express sarcasm - while the rest of the sentence is not suffcient for identifying the sarcasm. For example, *'Looking forward to going back to school tomorrow not'*. The speaker expresses sarcasm through not. In most reported works that use hashtag-based supervision, the hashtag is removed in the pre-processing step. This reduces the sentence above to *'Looking forward to going back to school tomorrow'* - which may not have a sarcastic interpretation, unless the author's context is incorporated.

2. Several approaches use lexical sentiment as a feature to the sarcasm classifier. It must, however, be noted that these approaches require 'surface polarity': the apparent polarity of a sentence. [33] describe a rule-based approach that predicts a

sentence as sarcastic if a negative phrase occurs in a positive sentence. [5] use a sentiment imbalance feature that is represented by star rating of a review disagreeing with the surface polarity. [40] cascade sarcasm detection and sentiment detection, and observe an improvement of 4% in accuracy for sentiment classification, when sentiment detection is aware of sarcastic nature of text.

3. Sarcasm is an infrequent phenomenon of sentiment expression. This skew also reflects in datasets. [54] use a dataset with a small set of sentences are marked as sarcastic. 12.5% of tweets in the Italian dataset given by [3] are sarcastic. On the other hand, [44] present a balanced dataset of 15k tweets. In some papers, specialized techniques are used to deal with the dataset imbalances. Data imbalance also influences the choice of performance metrics reported. Since AUC is known to be a more reliable indicator of performance than F-score for skewed data, [34] report AUC for balanced as well as skewed datasets, to demonstrate the benefit of their classifier.

Summary

In this chapter we gave an introduction to computational sarcasm detection & interpretation. After that we motivated the two problem statements. Then, we discussed linguistic types of sarcasm. Then, we discussed applications of sarcasm detection & interpretation. Finally, some of the issues in sarcasm detection were outlined.

Chapter 3

Incongruity

The organization of the chapter is as follows. section 3.1 describes incongruity in linguistics and its role in sarcasm. section 3.2 gives details about various types of incongruity.

3.1 Incongruity in Linguistics & Sarcasm

Sarcasm detection is based on well studied linguistic theories. In this chapter, we use one such linguistic theory: **context incongruity**.

Incongruity is defined as *"the state of being not in agreement, as with principles"*[1]. Context incongruity is a necessary condition for sarcasm.

[55] state that verbal irony is a technique of using incongruity to suggest a distinction between reality and expectation. Since sarcasm is a form of verbal irony, incongruity is of interest to sarcasm as well. Incongruity[2] is defined as the state of being incongruous (i.e. lacking in harmony; not in agreement with principles). [33] state that sarcasm/irony is understood because of incongruity.

The incongruity theory states that sarcasm is perceived at the moment of realization of incongruity between a concept involved in a certain situation and the real objects thought to be in some relation to the concept.[3]

Incongruity theories are essentially cognitive, i.e. they are based on some objective characteristics of a sarcastic text or other act (situation, event, picture, etc.). It is as-

[1]Source: The Free Dictionary
[2]Source: The Free Dictionary
[3]Source: Wikipedia

sumed that every such act involves two different lines of thought. These two are mutually incompatible, but also include a certain common part which makes the shift from one to another possible. The recipient begins to process textual or other information reducing it to the most accessible script, and proceeds until the interpretation bounces over a semantic obstacle and fails. Then some instantaneous cognitive work will be done to overcome the contradiction and another interpretation that has so far remained hidden can be found. The renewal of understanding is attended by the emotion of surprise.

Verbal irony serves many communicative purposes. One of these is to highlight disparity between expectations and reality. For instance, sarcastic statements like *"you are so punctual"* often convey failed expectations. The statement expresses what the speaker expected (punctuality), and because the statement is out of keeping with events (the target was actually late) there is incongruity between the speaker's attitude (negative, disappointment) and their actions (a positive statement). This incongruity is a reliable cue to sarcastic intent. Further, the degree of incongruity influences the extent to which ironic intent is perceived. The greater situational disparity led to a higher perception of irony, compared to a situation with less disparity (5 min late versus 50 min late). Thus, there is clearly an effect of degree of disparity on the perception of sarcasm.

We describe incongruity using an example. Imagine the following situation:[4]

Sheila was looking forward to her boyfriend Walter's visit. When Walter arrived, he was in a terrible mood and was snapping and yelling at Sheila and her housemates. Sheila turned to him and said:

- Weak verbal sarcasm: *"Aren't you in an agreeable mood?"* The incongruity between the negative situation and positive sentiment phrase *"agreeable mood"* conveys weak verbal sarcasm to *Walter*.

- Strong verbal sarcasm: *"Aren't you in a magnificent mood?"* The incongruity between the negative situation and strong positive sentiment phrase *"magnificent mood"* conveys strong verbal sarcasm to *Walter*.

- Literal: *"Aren't you in a bad mood?"*

[4]This example is taken from [33]

This example highlights that incongruity is an essential component of sarcasm, and the possibility of incongruity in different degrees is at the heart of sarcasm. Research shows that when there was a high degree of difference between the strong and weak version of statements, the speakers of strongly sarcastic statements were rated to be more condemning, more humorous, and more self-protecting than the speakers of weakly sarcastic statements.

3.2 Types of Incongruity

[33] state that the sarcasm processing time (time taken by humans to understand sarcasm) depends on the degree of context incongruity between the statement and the context. Deriving from this idea, [23] consider two cases of incongruity in sarcasm that are analogous to two degrees of incongruity. They call them explicit incongruity and implicit incongruity, where implicit incongruity demands a higher processing time.[5]

- **Explicit Incongruity:** Explicit incongruity is overtly expressed through sentiment words of both polarities (as in the case of *"I love being ignored"* where there is a positive word *"love"* and a negative word *"ignored"*). The converse is not true as in the case of *"The movie starts slow but the climax is great"*.

- **Implicit Incongruity:** An implicit incongruity is covertly expressed through phrases of implied sentiment, as opposed to opposing polar words. Consider the example *"I love this paper so much that I made a doggy bag out of it"*. There is no explicit incongruity here: the only polar word is *"love"*. However, the clause *"I made a doggy bag out of it"* has an implied sentiment that is incongruous with the polar word *"love"*.

[33] states that a strongly positive statement (the biasing information) presented in a negative situation can make the situation (the target) appear more negative. This forms the core of sarcasm.

[5]Examples of implicit and explicit incongruity are taken from [23].

Summary

This chapter gave an introduction to the theory of incongruity. It described incongruity in linguistics and its role in sarcasm. Then, different types of incongruity were explained to gain some insight.

Chapter 4

Literature Survey

In this chapter, we explore different works that have been done in the field of computational sarcasm detection and interpretation. The various dimensions of work include studying the nature of sarcasm and various approaches in the field of computational sarcasm detection and interpretation. Further, the role of dependency parsing for computational sarcasm interpretation is discussed. Finally, we discuss the role of numbers in sarcasm and models for incorporating numeracy in language models.

The organization of the chapter is as follows. The chapter starts with the description of the works that have been done in the field of sarcasm detection and interpretation section 4.1. Then section 4.2 focuses on the datasets for sarcasm detection. section 4.3 explain state of the art deep learning architectures for sarcasm detection. section 4.4 explains the role of dependency parsing in sarcasm interpretation task. Finally, in **??**, we describe numeracy for language models and conclude the chapter.

4.1 Related Work in Sarcasm Detection & Interpretation

Sarcasm and irony detection has been extensively studied in linguistic, psychology and cognitive science [14, 56]. Computational detection of sarcasm has become a popular area of natural language processing research in recent years ([21]). [53] present sarcasm recognition in speech using spectral (average pitch, pitch slope, etc.), prosodic and contextual cues. [6] use simple linguistic features like interjection, changed names, etc. for irony detection. [8] train a sarcasm classifier with syntactic and pattern-based features. [17] states that sarcasm transforms the polarity of an apparently positive or negative utterance into its opposite. [34] showed that sarcasm is often signaled by hyperbole, using intensifiers and exclamations; in contrast, non-hyperbolic sarcastic messages often receive an explicit marker. [48] captures sarcasm as a contrast between a positive sentiment word and a negative situation. [5] analyze the impact of different features for sarcasm/irony classification. [24] show how sarcasm arises because of implicit or explicit incongruity in the sentence. [4] proposed a pattern-based approach to detect sarcasm on Twitter. They proposed four sets of features that cover the different types of sarcasm. As deep learning techniques gain popularity, few deep learning-based architectures for sarcasm detection have also appeared in literature. [10] provide a neural network semantic model for sarcasm detection. Their model is composed of Convolution Neural Network (CNN) followed by a Long Short Term Memory (LSTM) network and finally a Deep Neural Network(DNN). [42] propose a novel method to detect sarcasm using Convolution Neural Networks. They have developed models based on a pre-trained convolutional neural network for extracting sentiment, emotion and personality features for sarcasm detection. [2] proposed a deep-learning-based architecture to automatically learn user embeddings. In their proposed approach they have used this user embeddings to provide *contextual features*, going beyond the lexical and syntactic cues for sarcasm. [60] use a bi-directional gated recurrent neural network followed by a pooling neural network to detect sarcasm. Recently, [12] propose a neural architecture that considers speaker's mood based on most recent prior tweets and model context based on the response of the tweet.

While there is a lot of work in the field of sarcasm detection, the problem of sarcasm interpretation is relatively new. Since this is a relatively new area of research, there is not much work done in this field. In this section, we describe some of previous works that focus on sarcasm interpretation and sarcasm detection. [41] propose models for sarcasm interpretation task. They focus on monolingual machine translation (MT). Their work include MT algorithms and evaluation measures. They introduce an algorithm called Sarcasm SIGN which is a MT based algorithm for sarcasm interpretation task which targets sentiment words, a defining element of textual sarcasm. Other works do not explore deep learning-based and rule-based approaches for sarcasm interpretation task.

4.2 Datasets for Sarcasm Detection

This section[1] describes the datasets used for experiments in sarcasm detection. We divide them into four classes: short text (typically characterized by noise and situations where length is limited by the platform, as in tweets on Twitter), long text (such as discussion forum posts), transcripts (such as transcripts of a TV show or a call center conversation), and other miscellaneous datasets. Short text can contain only one (possibly sarcastic) utterance, whereas long text may contain a sarcastic sentence among other non-sarcastic sentences.

1. Social media makes large-scale user-generated text accessible. However, because of restrictions on text length imposed by some of these platforms, this text tends to be short requiring authors to use abbreviations to fit their statements within the specific limit. Despite this noise, datasets of tweets have been popular for sarcasm detection because of availability of the Twitter API and popularity of Twitter as a medium. For Twitter-based datasets, two approaches to obtain annotations have been used. The first is manual annotation. [49] introduced a dataset of tweets which were manually annotated as either sarcastic or not. [37] studied sarcastic tweets and their impact to sentiment classification. They experimented with around 600 tweets which were marked for subjectivity, sentiment and sarcasm.

   The second technique to create datasets is the use of hashtag-based supervision. Many approaches use hashtags in tweets as indicators of sarcasm, to create labeled datasets. The popularity of this approach (over manual annotation) can be attributed to various factors:

   - Nobody but the author of a tweet can determine with certainty, whether the tweet was intended to be sarcastic or not. A hashtag is a label provided by authors themselves.

[1]This section is heavily based on [20]

- This approach allows rapid creation of large-scale datasets since manual effort is restricted. In order to create such a dataset, tweets containing particular hashtags are labeled as sarcastic.

2. **Long Text:** Reviews and discussion forum posts have also been used as sarcasm-labeled datasets. [36] use the Internet Argument Corpus which marks a dataset of discussion forum posts with multiple labels, one of them being sarcasm. [46] create a dataset of movie reviews, book reviews and news articles marked with sarcasm and sentiment. [45] deal with products that saw a spate of sarcastic reviews all of a sudden. Their dataset consists of 11000 reviews. [9] use a sarcasm-labeled dataset of around 1000 reviews. [5] create a labeled set of 1254 Amazon reviews, out of which 437 are ironic. [54] consider a large dataset of 66000 Amazon reviews. [35] use a dataset of reviews, comments, etc. from multiple sources such as Amazon, Twitter, Netease and Netcena. In these cases, the datasets are manually annotated because markers like hashtags are not available.

3. **Transcripts and Dialogue:** Since sarcasm is a form of verbal irony and is often expressed in the context of a conversation, datasets based on transcripts and dialogue have also been reported. [53] use 131 call center transcripts. Each occurrence of 'yeah right' is marked as sarcastic or not. The goal is to identify which 'yeah right' is sarcastic. Similarly, [44] create a crowdsourced dataset of sentences from an MTV show, Daria. On similar lines, [26] report their results on a manually annotated transcript of the TV Series 'Friends'. Every 'utterance' in a scene is annotated with two labels: sarcastic or not sarcastic.

4. **Miscellaneous datasets:** In addition to the three kinds of datasets above, several other datasets have been reported. [30] use 20 sarcastic and 15 non-sarcastic book excerpts, which are marked by 101 annotators. The goal is to identify lexical indicators of sarcasm. [57] focus on identifying which similes are sarcastic. For example, the simile 'as useful as a chocolate teapot' is to be predicted as sarcastic, while the simile 'as big as a plum' is not8. Hence, they first search the web for the pattern '* as a *'. This results in 20,000 distinct similes which are then annotated. [13] use a crowdsourcing tool to obtain a non-sarcastic version of a sentence if applicable. For

example ‘Who doesn’t love being ignored’ is expected to be corrected to ‘Not many love being ignored’. [39] create a manually labeled dataset of quotes from a website called sarcasmsociety.com. [27] create a similar dataset of quotes from GoodReads, a book recommendation website. However, in this case, they use user-determined tags to assign sarcasm labels.

4.3 State of the Art Deep Learning Architectures for Sarcasm Detection

As architectures based on deep learning techniques gain popularity in Natural Language Programming (NLP) applications, a few such approaches have been reported for automatic sarcasm detection as well. [27] use similarity between word embeddings as features for sarcasm detection. They augment these word embedding-based features with features from four prior works. The inclusion of past features is key because they observe that using the new features alone does not suffice for good performance. [1] present a novel Convolutional Network-based architecture that learns user embeddings in addition to utterance-based embeddings. The authors state that it allows them to learn user-specific context. [11] use a combination of a Convolutional Neural Network, a Recurrent Neural Network (Long Short-Term Memory) followed by a Deep Neural Network. They compare their approach against recursive SVM, and show an improvement for the deep learning architecture. [42] investigate the use of Deep Convolutional Networks for sarcasm detection.

4.4 Numeracy for Language Models: A step towards Numerical Sarcasm Detection

Numeracy is the ability to understand and work with numbers[2]. In [51], authors have explored different strategies for modelling numerals with language models, such as digit-by-digit composition, and propose a novel neural architecture that uses a continuous

[2]This section is heavily based on [51]

probability density function to model numerals from an open vocabulary.

Numeracy and literacy refer to the ability to comprehend, use, and attach meaning to numbers and words, respectively. Language models exhibit literacy by being able to assign higher probabilities to sentences that are both grammatical and realistic, as in this example:

'I eat an apple' (grammatica land realistic)

'An apple eats me' (unrealistic)

'I eats an apple' (ungrammatical)

Likewise, a numerate language model should be able to rank numerical claims based on plausibility:

'John's height is 1.75 metres' (realistic)

'John's height is 999.999 metres' (unrealistic)

In this section, we describe different strategies for modelling numerals, such as memorization, digit-by-digit composition and neural architectures based on continuous probability density functions. We also describe the use of evaluations that adjust for the high out-of-vocabulary rate of numerals and account for their numerical value (magnitude).

The intuition behind modelling numbers is that, humans process quantities through two exact systems (verbal and visual) ad one approximate number system that semantically represents a number on a mental number line.

Softmax based models treats all tokens (words and numerals) alike. There are modifications to softmax based models which use digit based embeddings. These models use two types of embeddings for numerals.

1. Character based embeddings for in-vocabulary numerals. The character set comprises digits (0-9), the decimal point, and an end-of-sequence character.

2. Token embeddings for the remaining vocabulary.

Other variants of softmax based models use hierarchical softmax. The probability of next token is decomposed to that of its class (numeral vs text) and the probability of the exact token (number vs word) from within the class.

Neural network based models include digit based RNN. They use digit-by-digit composition strategy that estimates the probability of the numeral from the probabilities of its digits. This strategy can accommodate an open vocabulary, i.e., it eliminates the need of **UNK** symbol, as the probability is normalized one digit at a time over the much smaller vocabulary of digits.

There are models which use mixture of gaussians (MoG). These models are inspired by the approximate number system and mental line discussed above. The MoG model compute the probability of numerals from a probability density function over real numbers, using a mixture of gaussians for the underlying probability density function.

Summary

This chapter covered the various works that have been done in the field of sarcasm detection and interpretation. First, we discussed the work done in the field of computational sarcasm detection and interpretation. Then, we described various datasets for computational sarcasm detection and interpretation and discussed various state of the art deep learning-based models for computational sarcasm detection and interpretation. Finally, we described models that are capable of modelling numeracy for language models which is a very promising step towards numerical sarcasm detection.

Chapter 5

Approaches for Sarcasm Detection

In general, approaches to sarcasm detection can be classified into: rule-based, statistical and deep learning-based approaches[25].[1]. This chapter describes these approaches in detail. section 5.1 describes various rule-based approaches. Then, in section 5.2 we describe various statistical machine learning-based approaches. section 5.3 describes various deep learning-based approaches. Finally, in section 5.4, we discuss the role of additional context for detecting sarcasm.

5.1 Rule Based Approaches

Rule-based approaches attempt to detect sarcasm using specific set of rules that rely on specific indicators of sarcasm. [37] propose that hashtag sentiment is a key indicator of sarcasm. Usually, twitter users use hashtags to indicate sarcasm, and hence, if the sentiment expressed by a hashtag does not agree with the sentiment expressed in a tweet, then, the tweet is predicted as sarcastic.

[49] present rule based classifiers that look for a positive verb and a negative situation phrase in a sentence. The set of negative situation phrases are extracted using a well-structured, iterative algorithm that begins with a bootstrapped set of positive verbs and iteratively expands both the sets (namely, positive verbs and negative situation phrases). They experiment with different configurations of rules such as restricting the order of the verb and situation phrase.

[1]This chapter is heavily based on [25]

[31] present a two rule based approaches for detecting sarcasm due to numbers. In their first approach, they create two repositories, i.e., sarcastic and non-sarcastic using a training dataset. Their rule based approach predict numerical sarcasm in a test tweet by matching noun phrase list of a test tweet with tweets in two repositories. Then they match the number unit and if it matches they make final prediction in the number present in the test tweet lies within ± 2.58 Std dev of the mean value for that number unit present in the matched sarcastic entry. If the number unit does not match, then, the non-sarcastic tweet repository is consulted for making prediction.

This approach is restricted in terms of phrase matching. Therefore, in their second rule based approach, instead of exact matching, they match the noun phrase list using cosine similarity.

5.2 Statistical Machine Learning-Based Approaches

Most work in statistical sarcasm detection relies on different forms of Support Vector Machines (SVM) (or SVM-Perf[2]). [16] use SVM and Logistic Regression, with the χ^2 test used to identify discriminating features. Others have compared rule-based techniques with a SVM-based classifier. [47] use Naive Bayes and Decision Trees for multiple pairs of labels among irony, humor, politics and education. [58] use SVM-HMM in order to incorporate sequence nature of output labels in a conversation. Similarly, [26] validate that for conversational data, sequence labeling algorithms perform better than classification algorithms. They use SVM-HMM and SEARN as the sequence labeling algorithms. [35] compare several ensemble-based classification approaches including Bagging, Boosting, etc. and show results on five datasets.[3]

5.3 Deep Learning-Based Approaches

Deep learning based approaches are getting popular in various NLP tasks. A few such approaches have been explored for the task of automatic sarcasm detection as well. [28]

[2]http://www.cs.cornell.edu/people/tj/svm_light/svm_perf.html
[3]This section is taken from [25]

use similarity between word embeddings as features for sarcasm detection. They augment these word embedding-based features with features from four prior works. The inclusion of past features is key because they observe that using the new features alone does not suffice for good performance. [1] present a novel Convolutional Network-based architecture that learns user embeddings in addition to utterance-based embeddings. The authors state that it allows them to learn user-specific context. [11] use a combination of a Convolutional Neural Network, a Recurrent Neural Network (Long Short-Term Memory) followed by a Deep Neural Network. They compare their approach against recursive SVM, and show an improvement for the deep learning architecture.

[31] present three deep neural network based architectures for the task of detecting sarcasm due to numbers. A novel Convolutional Network based model. They also present a Recurrent Neural Network (Long Short-Term Memory) followed by a fully connected layer. The third model (CNN-LSTM-FF) is a hybrid deep network which is a combination of Convolutional Neural Network followed by Long Short-Term Memory layer followed by affine layer. Their models outperform state of the art systems for the task of detecting sarcasm due to numbers.

5.4 Going Beyond Target Sentence

Considering only text utterance for detecting if it is sarcastic or not is not sufficient. Additional context is very crucial because every sentence can be sarcastic given the right context. For example, consider the tweet (ignoring the sarcasm hashtag for a moment) *"@BernieSanders and obamas are doing a great job #sarcasm"*[4]. Without knowing author's political leaning, it would be difficult to conclude whether the remark was intended sarcastically or not. The intuition is that different speakers tend to employ sarcasm regarding different subjects and, thus, sarcasm detection models ought to encode such speaker information.

[4] This example is taken from [1]

5.4.1 Role of Context in Sarcasm Detection

Recent trend in sarcasm detection is the use of context. The term context here refers to any information beyond the text to be predicted, and beyond common knowledge. For example, the sentence ‘I love solving math problems all weekend’ may not be sarcastic to a student who loves math, but may be sarcastic to many others. This example requires context outside of the text to be classified. Incorporating context for sarcasm detection task have shown promising results and several recent approaches have looked at ways of incorporating context. [25] describes three types of contexts that have been reported in the past:

1. **Author Specific Context:** This type of context refers to additional information about the author of the target text. For example, the statement *”Nicki Minaj... I love her”*[5] may be an exaggeration. In order to understand sarcasm therein, information specific to the author who wrote the text is useful. Past works have considered historical tweets by the author to model context. Named entity phrases in the target tweet are searched for in the timeline of the author in order to gather the true sentiment of the author. This historical sentiment is then used to predict whether the author is likely to be sarcastic, given the sentiment expressed towards the entity in the target tweet.

2. **Conversational Context:** This type of context refers to the text in the conversation of which the target text is a part. Consider a simple exclamation *”Yeah right, I can see that!”*. This may or may not be sarcastic. If the preceding sentence is *”I don’t feel bad about my low grades at all”*, the sarcasm in the exclamation can be inferred. [58] use sequence labeling to capture conversational context. For a sequence of tweets in a conversation, they estimate the most probable sequence of three labels: happy, sad and sarcastic, for the last tweet in the sequence. A similar approach is used in [26] to predict sarcasm in every text unit in a sequence of utterances in a scene.

3. **Topical Context:** This context follows the intuition that some topics are more

[5]This example is taken from [25]

likely to evoke sarcasm more commonly than others. For example, a tweet related to politics is more likely to evoke sarcasm than a tweet about weather. [12] show that the mood exhibited by a speaker over tweets leading up to a new post is as useful a cue for sarcasm as the topical context of the post itself.

5.4.2 Recent Approaches for incorporating context for Sarcasm Detection

[1] introduces a deep neural network that incorporates content as well as context for sarcasm detection. The intuition is that different speakers will tend to employ sarcasm regarding different subjects and, thus, sarcasm detection models ought to encode such speaker information. They propose a novel approach to sarcasm detection on social media that does not require extensive feature engineering to incorporate additional context. They develop a neural model that learns to represent and exploit embeddings of both content and context. The goal is to learn representations (vectors) that encode latent aspects of users and capture homophily, by projecting similar users into nearby regions of the embedding space. The intuition is to capture relations between users and the content they produce by optimizing the conditional probability of texts, given their authors. Given a sentence S = $w_1, ..., w_N$ where w_i denote a word drawn from a vocabulary V, they aim to maximize the following probability:

$$P(S|user_j) = \sum_{w_i \in S} \log P(w_i|u_j) + \sum_{w_i \in S} \sum_{w_k \in C(w_i)} \log P(w_i|e_k) \tag{5.1}$$

where,

C(w_i) denote the set of words in a pre-specified window around w_i, $e_k \in R^d$ and $u_j \in R^d$ denote the embeddings of the word k and user j, respectively.

This objective function encodes the notion that the occurrence of a word w, depends both on the author of S and it's neighbouring words. They have named this model as CUE-CNN (Content and User Embedding Convolutional Neural Network) and the complete model architecture is illustrated in Figure 5.1 [6]

[6] This figure is taken from [1]

The learned user embeddings are visualized in 2 dimensions using t-SNE and is illustrated in Figure 5.2

Summary

In this chapter, we discussed various approaches pertaining to three typical paradigms of NLP. We discussed rule-based approaches, statistical machine learning-based approaches and deep learning-based approaches for sarcasm detection. We also discussed the role of additional context in sarcasm detection. Finally, we looked at state of the art deep learning-based models that incorporate additional context for the task of sarcasm detection.

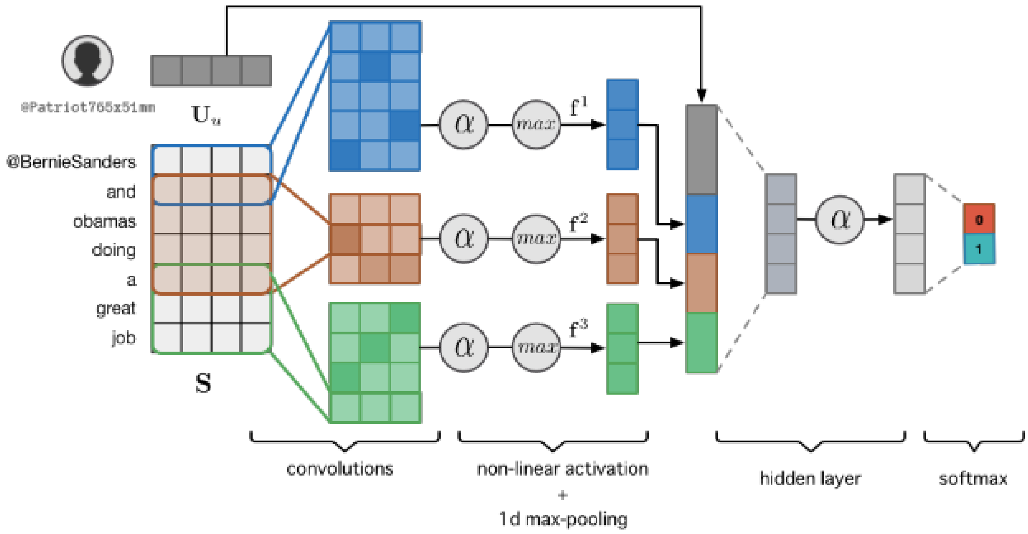

Figure 5.1: Illustration of the CUE-CNN model for sarcasm detection. The model learns to represent and exploit embeddings of both content and users in social media.

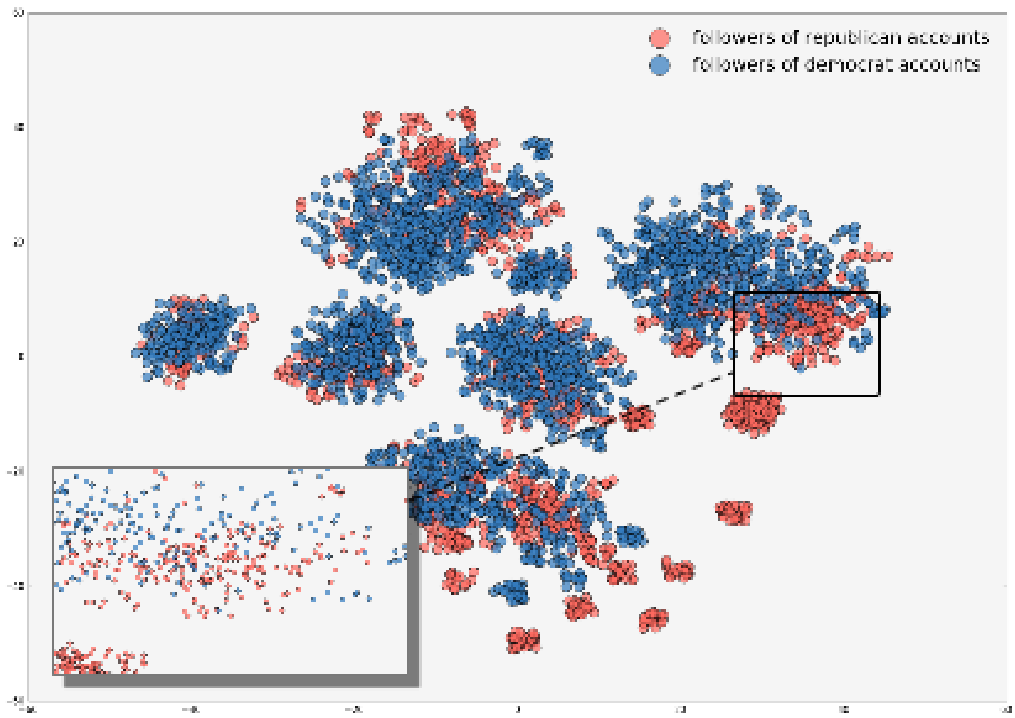

Figure 5.2: Users colored according to the politicians they follow on Twitter: the blue circles represent users that follow at least one of the democrats accounts; the red circles represents users that follow at least one of the republican accounts. We can see that the users with a similar political leaning tend to have similar vectors.

Chapter 6

Detecting Sarcasm in Text Containing Numbers: A Comparison of Three NLP Paradigms

This chapter is based on the paper we submitted in AAAI 2019 [1].

Sarcasm is a peculiar challenge to sentiment analysis because it uses verbal irony to express contempt or ridicule, thereby, potentially confusing typical sentiment classifiers. Several approaches for sarcasm detection have been reported in the recent past [21]. The approaches are general in nature, in that they apply to all sentences irrespective of specific properties. In this paper, we focus on a peculiar form of sarcasm: sarcasm expressed through numbers. In other words, the goal of this chapter is the classification task where a tweet containing one or more numbers is classified as sarcastic or not. For example, the sentence '*Having 2 hours to write a paper is fun*' is sarcastic. The numeral plays a key role in conveying the sarcasm.

Therefore, in this chapter, we focus on different approaches for detection of sarcasm in text containing numbers. We have introduced the task in chapter 2. In this chapter, we identify its challenges, introduce a labeled dataset and devise three approaches for the task. Our approaches are based on three prevalent paradigms of NLP: rule-based, statistical machine learning-based[2] and deep learning-based approaches. Then, we present results

[1]https://aaai.org/Conferences/AAAI-19/

[2]This refers to statistical approaches that do not rely on deep learning.

and error analysis.

The three approaches show typical trends. The rule-based approach uses a training dataset to compute optimal numerical thresholds which determine the sarcastic nature of the tweet, thereby relying on an ad-hoc incorporation of human judgment. The statistical machine learning-based approach uses features specific to the task, as a result of which human-supplied features are used for classification. Finally, the deep learning-based approaches use three typical networks: Convolution Neural Networks (CNN), Long-short Term Memory (LSTM) networks and Attention Networks. Thus, feature engineering is relegated to data, thereby removing all human involvement specific to the task.

The contribution of the paper we submitted in AAAI 2019 is as follows:

1. The paper details out the purpose and challenges of the problem. We believe that this paper will motivate future research.

2. We introduce a dataset of tweets containing numbers which is labeled with sarcasm.

3. Finally, our results spanning three prevalent paradigms serve as a strong baseline for future work in detecting sarcasm arising due to numbers.

The rest of the chapter is organized as follows. First, we present our motivation, and describe various approaches in detail. Then, we outline the experimental setup and present the results of our experiments. Then, we present our error analysis and discuss the related work. Finally, we conclude the paper and discuss future work.

6.1 Motivation

Sarcasm is an infrequent phenomenon in sentiment-bearing text[21]. While several approaches to detect sarcasm have been reported [17, 24], they may fall short in case of sarcasm expressed via numbers. Consider the following sentences:

1. *This phone has an awesome battery backup of 38 hours*

2. *This phone has a terrible battery backup of 2 hours*

3. *This phone has an awesome battery backup of 2 hours*

At the time of writing this report, a battery backup of 38 hours is good for phones while a battery backup of 2 hours is bad. Therefore, sentences 1 and 2 are non-sarcastic because the sentiment of the adjectives (‘*awesome*’ and ‘*terrible*’) conforms with the sentiment associated with the corresponding numerical values. On the contrary, the sarcasm in sentence 3 can be understood in terms of incongruity[3] between the word ‘*awesome*’ and ‘*2 hours*’ for the battery life of the phone. The sarcasm in sentence 3 above occurs because of incompatibility/incongruity between ‘*awesome*’ (positive word) and ‘*2 hours*’ (numerical value).

One may argue that sarcasm in text containing numbers is rare and, hence, does not necessitate consideration from the research community. However, the prevalence of such sarcasm speaks otherwise. We consider a set of approximately 100,000 sarcastic tweets. This set has 11,488 tweets containing numbers, amounting to 11.48%. However, a related question is whether or not tweets containing numbers are sarcastic due to the numbers themselves. An analysis of 500 randomly selected sarcastic tweets containing numbers shows that 79.4% of the tweets are sarcastic due to the numbers present in them.

We note here that there is a subtle difference between ‘*sarcasm in text containing numbers*’ and ‘*sarcasm arising due to numbers in text*’. For example, ‘*Having a head ache at 4 am is the best thing to happen*’ contains a number but is sarcastic without the number as such. For the sake of brevity, we assume that the two are the same. This is a reasonable simplification because majority (79.4%) of these tweets are found to be sarcastic due to the numbers present in them, as indicated above.

6.2 Approaches

As stated in the Introduction section, we describe three approaches for the problem: rule-based (relying on heuristic-based evidences), statistical machine learning-based (referring to traditional feature engineering-based approaches) and deep learning-based (which learns representations using neural networks without relying on human-engineered features).

[3] [18] describe the relationship between incongruity and sarcasm.

6.2.1 Rule-based Approaches

In this section, we describe the two rule-based approaches. The approaches have a common training phase and distinct test phases. In general, the approaches consider noun phrases in the tweet as candidate contexts, and determine the optimal threshold of a numerical measure for each context. The optimal threshold is determined using a three-step algorithm.

In the **training phase** that is common to both the approaches, two repositories, namely sarcastic repository and non-sarcastic repository, each containing an entry corresponding to sarcastic and non-sarcastic tweets respectively, are created. Each entry in the repository corresponds to a tweet, stored in the form:

- **Tweet Index No.**: A numerical identifier for a tweet in the dataset.
- **Noun Phrase list**: A noun phrase is a phrase which has a noun as its head. Noun phrases in the tweet are extracted using the Stanford constituency parser[4].
- **Numerical Value:** The numerical term in the tweet.
- **Number Unit:** The number unit in the tweet. The word in the tweet following the word whose part-of-speech (POS) tag is CD (Cardinal Number) is selected as number unit. Examples of number units are minutes, hour, days, years etc.

In addition to tweet entries, both sarcastic and non-sarcastic repositories also maintain two dictionaries. The details of which are as follows:

- **Dictionary of Mean values:** Each entry is a key value pair where key is the number unit and value is the average of all the numbers corresponding to that number unit.
- **Dictionary of Standard Deviation:** Each entry is a key value pair where key is the number unit and value is the standard deviation of all the numbers corresponding to that number unit.

[4]https://nlp.stanford.edu/software/srparser.html

For example, assume that the 14th instance in the dataset is the sarcastic tweet *'This phone has an awesome battery backup of 2 hours'*. This tweet contains two noun phrases: '*phone*' and '*awesome battery backup*'. The words in noun phrases in this tweet are *'phone', 'awesome', 'battery', 'backup'*. Given these entities, the tweet representation is: *(14, ['phone', 'awesome', 'battery', 'backup'], 'hours')*. Since the tweet is sarcastic, it is stored in the sarcastic repository.

In case of **the test phase for Approach 1**, a test tweet is classified as sarcastic or non-sarcastic based on the following steps:

1. **Noun Phrase List Computation**: The noun phrase list is computed in a manner similar to the training dataset.

2. **Sarcastic Repository Consultation**: The words in the noun phrase list of test tweet are first looked up in the noun phrase list of entries in the sarcastic repository. The entry with the highest word overlap is chosen. We call this the most similar entry. If the number unit in the most similar entry is same as that in the test tweet, we use dictionary of mean values and dictionary of standard deviation to check whether the number present in the test tweet lies within ± 2.58 standard deviation[5] of the mean of numerical values for that number unit. If it does, the tweet is predicted as sarcastic. If it does not, it is predicted as non-sarcastic.

3. **Non-sarcastic Repository Consultation**: If the number unit in the most similar entry is not the same as that in the test tweet, we consult the non-sarcastic repository and choose the most similar entry from the repository. If the number unit in the most similar entry is same as that in the test tweet, we use dictionary of mean values and dictionary of standard deviation to check whether the number present in the test tweet lies within ± 2.58 standard deviation of the mean of numerical values for that number unit. If it does, the tweet is predicted as non-sarcastic and if it does not, it is predicted as sarcastic.

4. **Fall-back label assignment**: If no match is found, the tweet is predicted as non-sarcastic.

[5] ± 2.58 indicates the 99% Confidence Interval

Rule-based Approach 1 is restrictive because, during the computation of the most similar entry, words in the noun phrase list of test tweet must exactly match with words in the noun phrase list of entries in the repository. Therefore, **the test phase of Approach 2** relaxes the constraint by using *cosine similarity* to compute the most similar entry. A test tweet is classified as sarcastic or non-sarcastic based on the following steps:

1. **Computation of Test Vectors & Repository Vectors**: The noun phrase list is computed in a manner similar to approach 1. A 'test vector' is computed from the list by adding up corresponding word vectors[6] and normalizing it by the length of the noun phrase list. Similarly, noun phrase list of each entry in the two repositories are converted to '*repository vectors*' (referred to as '*sarcastic repository vectors*' and '*non-sarcastic repository vectors*').

2. **Sarcastic Repository Consultation**: Instead of matching words as in approach 1, the most similar entry from the sarcastic repository is the entry whose sarcastic repository vector has the highest cosine similarity with the test vector. The prediction based on most similar entry is the same as in case of approach 1: the numerical value is compared with the range defined by certain standard deviations of the mean value.

3. **Non-sarcastic Repository Consultation**: The most similar entry from the non-sarcastic repository is determined in the same manner as the sarcastic repository consultation. The corresponding predictions are also analogous.

6.2.2 Statistical Machine Learning-based Approaches

In this section, we describe various features used by statistical machine learning-based approaches:

- **Sentiment-based features (S)**: These features include number of positive words, number of negative words[7], number of highly emotional positive words, number of

[6]These are 200-dimensional vectors learned using Word2Vec from a tweet corpus containing 6 million tweet words downloaded using Twitter-API (https://dev.twitter.com).

[7]Positive and negative words are selected using SentiWordNet.

highly emotional negative words. (Positive/Negative word is said to be highly emotional if it is an adjective, adverb or verb.)

- **Emoticon-based features (E)**: These features include positive emoticon, negative emoticon, contrast between word, i.e, a boolean feature that will be one if both positive and negative words are present in the tweet, contrast between emoji, i.e, it will take the value as one when either positive word and negative emoji is present or negative word and positive emoji is present in the tweet.

- **Punctuation-based features (P)**: These features include number of exclamation marks, number of dots, number of question mark, number of capital letter words and number of single quotations.

- **Numerical value (NV)**: This feature is the numerical value in the tweet.

- **Numerical unit (NU)**: This feature is a one-hot representation of the of unit present in the tweet.

- **Sentence Vector (T)**: We learn word vectors of different dimensions (25-D, 50-D, 100-D, 150-D, 200-D, 250-D and 300-D), of tweet words using word2vec [38] tool on a large corpora of 6 million tweets[8]. To generate sentence vectors, word vectors are summed for each dimension and normalized by the length of the tweet. Finally, we obtain the sentence vectors of different dimensions.

6.2.3 Deep Learning-based Approaches

In this section, we describe three deep learning-based approaches.

[8]Downloaded by crawling twitter using Twitter API (https://dev.twitter.com)

CNN-FF Model

The architecture of the CNN-FF model is shown in Figure 6.1. The embedding matrix $E \in \mathbb{R}^{|V| \times d}$ uses $|V|$ as the vocabulary size and d as the word vector dimension. For the input tweet, we obtain an input matrix $I \in \mathbb{R}^{|S| \times d}$ where $|S|$ is the length of the tweet. I_i is the d-dimensional vector for i-th word in the tweet in the input matrix. Let k be the length of the filter, and the vector $f \in \mathbb{R}^{|k| \times d}$ is a filter for the convolution operation. For each position p in the input matrix I, there is a window w_p of k consecutive words, denoted as:

$$w_p = [I_p, I_{p+1}, ..., I_{p+k-1}] \tag{6.1}$$

A filter f convolves with the window vectors (k-grams) at each position to generate a feature map $c \in \mathbb{R}^{|S|-k+1}$, each element c_p of the feature map for window vector w_p is produced as follows:

$$c_p = func(w_p \circ f + b) \tag{6.2}$$

where $\circ$ is element-wise multiplication, $b \in \mathbb{R}$ is a bias term and $func$ is a nonlinear transformation function that can be sigmoid, hyperbolic tangent, etc. We apply max-over-time pooling over the obtained feature map. We use multiple filters of different sizes and output from each filter is concatenated to get the final feature vector. This feature vector acts as input to the fully-connected layer. We train the entire model by minimizing the binary cross-entropy loss over a mini-batch (of size e) of training examples.

$$E(y, \widehat{y}) = \sum_{i=1}^{e} y_i \log(\widehat{y_i}) \tag{6.3}$$

CNN-LSTM-FF Model

The input matrix representation for the input tweet is obtained from the embedding matrix E in a similar way as in CNN-FF model. Filters of size $5Xd$, where d is the word vector dimension, slides over the input matrix I of the tweet in order to extract the features. We pass the output of the convolutional network through a pooling layer and use max-pooling with size 4. All the filters are of same dimension and after performing

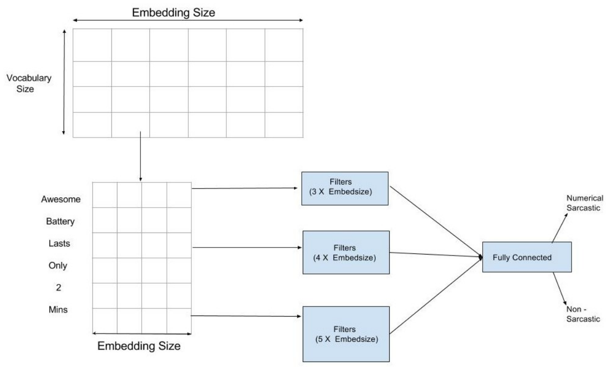

Figure 6.1: CNN followed by Fully Connected Layer for Numerical Sarcasm Detection

pooling operation over their outputs, we obtain a concatenated feature matrix denoted as:

$$C = [c_1; c_2; ....c_n]$$

where n in c_n denotes the total number of filters used in the architecture. In feature matrix $C \in \mathrm{I\!R}^{l \times n}$, each $c_i \in \mathrm{I\!R}^l$, where l is the dimension obtained after pooling operation. Let $x_j \in \mathrm{I\!R}^{1 \times n}$ is vector obtained from matrix C. Vector x_j is the input for the LSTM cell at j^{th} timestep and the LSTM cell runs for l timesteps taking different input obtained from matrix C, at each timestep. At the end of l^{th} timestep, output from the LSTM cell acts as input for the fully connected layer. We train this model by minimizing binary cross-entropy loss.

Attention Network

Figure 7.2 shows the architecture of our attention network. It consists of two main parts: a word encoder and a word level attention layer. The two main components of our attention network are described below:

1. **Word Encoder:** Given an input tweet of length T with words w_t, where $t \in [1, T]$. We convert each word w_t to its vector represention x_t using the embedding matrix E. Then, we use a bidirectional LSTM to get annotations of words by summarizing information from both directions. The bidirectional LSTM contains the forward LSTM $\overrightarrow{f}$, which reads the tweet from w_1 to w_T and a backward LSTM $\overleftarrow{f}$, which

reads the tweet from w_T to w_1:

$$x_t = E^T w_t, t \in [1, T] \tag{6.4}$$

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(x_t), t \in [1, T] \tag{6.5}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(x_t), t \in [T, 1] \tag{6.6}$$

The annotation for a given word w_t is finally obtained by concatenating the forward hidden state $\overrightarrow{h_t}$ and backward hidden state $\overleftarrow{h_t}$, i.e., $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$, which summarizes the information of the whole tweet centered around w_t.

2. **Word Level Attention:** We claim that numbers play a crucial role while predicting sarcasm. Hence, we introduce attention mechanism to extract information which is important to the overall meaning of the tweet. Our attention architecture is similar to the attention model introduced in [59].

$$u_t = tanh(W_w^T h_t + b_w) \tag{6.7}$$

$$\alpha_t = \frac{exp(u_t^T u_w)}{\sum_t exp(u_t^T u_w)} \tag{6.8}$$

$$s_i = \sum_t \alpha_t h_t \tag{6.9}$$

$$p = softmax(W_c^T s_i + b_c) \tag{6.10}$$

First, the word annotation h_t is multiplied with $W_w \in \mathbb{R}^{2d \times T}$ and added to $b_w \in \mathbb{R}^{T \times 1}$, which is fed into *tanh* layer to get u_t as its hidden representation. Then, we calculate the similarity of u_t with a word level context vector u_w to measure the importance of the words and get a normalized importance weight α_t using softmax function. The word level context vector u_w is randomly initialized and jointly learned during the training process. Finally, this representation is aggregated to form a tweet vector s_i, which is multiplied with $W_c \in \mathbb{R}^{2d \times 2}$ and added to $b_c \in \mathbb{R}^{2 \times 1}$ to generate p, which is used for classification. We train this model similar to the previous architectures, i.e., by minimizing binary cross-entropy loss.

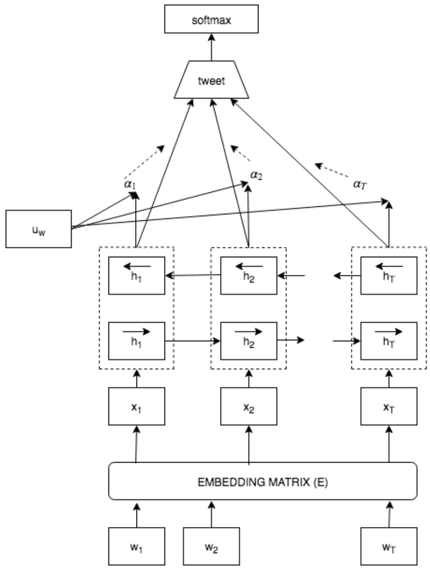

Figure 6.2: Attention Network for Numerical Sarcasm Detection

Dataset	**Sarcastic**	**Non-Sarcastic**
Dataset 0	100000	250000
	Numeric Sarcastic	**Non-Sarcastic**
Dataset 1	11024	49925
Dataset 2	11024	11024

Table 6.1: Statistics of Datasets.

6.3 Experimental Setup

We create three datasets containing tweets as follows. We download tweets containing hashtags #sarcasm, #sarcastic, #BeingSarcastic as sarcastic, and those with #nonsarcasm, #notsarcastic as the non-sarcastic, using the Twitter-API[9]. We call this Dataset 0. We eliminate duplicate tweets and retweets. We also remove URLs, usernames and other Non-ASCII characters from remaining tweets. Then, we retain only those tweets which contain numerical values. We remove irrelevant tweets, like the ones which contain alphabet or special character adjacent to a number like *Model34d, 4s,* < 3 (heart smiley) etc. We call this Dataset 1. This dataset contains more non-sarcastic tweets than sarcastic tweets. Since sarcasm is a rare phenomenon, Dataset 1 also simulates this distribution.

We also create a balanced dataset from Dataset 1, by selecting equal number of sarcastic and non-sarcastic tweets from Dataset 1. We call this Dataset 2. Table 6.1 shows

[9]https://dev.twitter.com

statistics of our datasets.

As baselines, we re-implement the work reported by [5], [17], [34] and [24]. We train classifiers for the features introduced by these approaches, using SVM^{perf} by [19] with RBF kernel. We compare their performance against our approaches, and report the average 5-fold cross-validation values in the next section.

For our statistical machine learning-based approaches, we use the following classifiers:

- **SVM** with RBF kernel and c[10] $= 1.0$ using grid-search
- **Random-forest** with $number\ of\ estimators = 10$
- **KNN** with neighbors $K = 3$

We implement these classifiers using scikit[11]. Features used are described in Statistical Machine Learning-based Approaches section.

For the deep learning-based approaches, we pad shorter tweets by a special pad character. We initialize the embedding matrix E by pre-trained tweet word embeddings of different dimensions (25-D, 50-D, 100-D, 150-D, 200-D, 250-D and 300-D) learned on a large corpora of 6 million tweets[12]. For CNN-FF Model, we use 128 filters each of size 3, 4 and 5, i.e., a total 128×3 filters. We use a dropout probability of 0.5. We train our network using mini-batch gradient descent. In the pooling layer, we perform max-over time pooling over the output from each filter.

For CNN-LSTM-FF Model, we use 64 filters, each with the same dimension, i.e., $5 \times Embeddingsize$. We investigate the results with different size embeddings and dropout probability of 0.25 for this architecture.

Approach	**Dataset-0**	**Dataset-1**	**Dataset-2**
[5]	0.69	0.19	0.16
[17]	0.68	0.17	0.15
[34]	0.67	0.21	0.17
[24]	**0.72**	**0.27**	**0.25**

Table 6.2: F-score degradation of four previous approaches on datasets containing numbers (Dataset 1 and Dataset 2).

[10] c is a regularization parameter
[11] http://scikit-learn.org/stable/modules
[12] Extracted by crawling twitter using Twitter API (https://dev.twitter.com)

Approaches	Dataset 1			Dataset 2		
	P	R	F	P	R	F
Rule-Based Approaches						
Approach-1	0.81	0.82	0.81	0.81	**0.83**	0.82
Approach-2	0.78	0.79	0.78	0.78	0.81	0.79
Statistical Machine-Learning-based Approaches						
SVM	0.82	0.83	0.82	**0.87**	0.82	**0.84**
KNN	0.77	0.79	0.78	0.84	0.70	0.76
Random Forest	0.83	0.85	0.84	0.85	0.80	0.82
Deep Learning-based Approaches						
CNN-FF	**0.93**	**0.93**	**0.93**	0.65	0.65	0.65
CNN-LSTM-FF	0.92	0.92	0.92	0.62	0.63	0.62
Attention Model	0.91	0.92	0.91	0.62	0.62	0.62

Table 6.3: Comparison of Classification Results of different approaches on **Dataset 1** and **Dataset 2**.

6.4 Results

In this section, we evaluate our approaches to detect sarcasm in numerical portions of text.

We first empirically motivate why sarcasm detection of numbers is necessary. Table 6.2 shows the degradation in performance of four previous approaches on Dataset 1 and Dataset 2. We observe that among the four previous approaches, features from [24] perform the best and obtain an F-score of **0.27** and **0.25** on Dataset 1 and Dataset 2 respectively. There is a **degradation of 62.5% and 65.3%** on Dataset 1 and Dataset 2 respectively. This degradation clearly shows that the past approaches are not able to capture the sarcasm that arise due to numbers in text. This shows that there is a need to develop a system that is capable of capturing sarcasm arising due to numbers.

We now look at results of our approaches for detecting sarcasm arising due to numbers. Table **??** compares the three approaches. We observe that, among the rule based approaches, Rule-based Approach 1 performs better with an F-score of **0.81** on Dataset 1 and **0.82** on Dataset 2. Among the three statistical machine learning-based approaches, Random Forest performs best on Dataset 1 and SVM performs best on Dataset 2. Random Forest obtains an F-score of **0.84** and **0.82** on Dataset 1 and Dataset 2 respectively. SVM obtains an F-score of **0.82** and **0.84** on Dataset 1 and Dataset 2 respectively. KNN obtains an F-score of **0.78** and **0.76** on Dataset 1 and Dataset 2 respectively.

Among the deep learning based approaches, CNN-FF network performs best with the embedding size of 250-D and obtains an F-score of **0.93** and **0.65** on Dataset 1 and Dataset 2 respectively. CNN-LSTM-FF network obtains the best F-score of **0.92** and

0.62 with embedding size of 200-D on Dataset 1 and Dataset 2 respectively. Attention Network obtains the best F-score of **0.91** and **0.62** with embedding size of 100-D on Dataset 1 and Dataset 2 respectively.

Among the three sets of approaches, the best F-score of **0.93** is obtained by CNN-FF Model on Dataset 1. The best F-score of **0.84** is obtained by SVM on Dataset 2. This is a **significant improvement of 66% and 59%** in F-score on Dataset 1 and Dataset 2 respectively against the best performing past approach of [24].

F-scores of all our deep learning-based approaches are very close. To check if they are statistically significant, we perform **Kolmogorov-Smirnov test** (KS test)[13] of significance on the results generated by 100 runs of each deep learning-based approach. After performing the KS test, we find that the **p-value**[14] (probability value) is **less than 0.05**. This indicates the strong evidence against the null hypothesis, and proves that the results obtained from our deep learning-based approaches are statistically significant.

6.5 Error Analysis

A qualitative analysis of errors results in six categories:

1. **Sarcasm not due to numbers**: Sarcastic sentences which contain a number but the sarcasm is not due to the number. This error category is obvious since it is beyond our scope to distinguish between the two. An example of this type is '*phelps will be the mvp for 2014 lmao phelpshaterhere*'

2. **Numbers highlighting sarcasm**: An interesting type of error is related to the previous. Although the sarcasm is not due to the numerical value, the number highlights the sarcastic property of the sentence, as in '*day 2 of having an adorable puppy n he already chewed up my macbook charger*'. The fact that the incident happened on the 2nd day strengthens the sarcastic expression in the sentence.

[13]https://en.wikipedia.org/wiki/Kolmogorov-Smirnov_test

[14]The p-value is used in the context of null hypothesis testing in order to quantify the idea of statistical significance of evidence.

3. **Interplay of numbers**: Multiple numerical entities may result in sarcasm as in the case of '*wow..from 30$ to 25$... significant discount!*'. Our approaches are not designed to take this into account.

4. **Unseen situations**: Since numeric sarcasm is associated with situations present in the tweet, situations unseen in the training set result in errors in sarcasm detection. An example of such a tweet is '*yay it's 3 am & i'm bored with no one to talk to*'.

5. **'Special' numbers**: These include numeric tokens that should not have been considered as tokens at all. This includes the use of '2' and '4' in place of 'to' and 'for' in noisy text such as tweets.

6. **Additional context required**: These are examples where the sarcasm is understood if additional context is available. For example, '*i get to work with the worlds mos (sic) exciting person at 9 to make my day better*'.

To understand the proportion of errors made by each of our approaches, we randomly sample fifty instances each for three configurations: (A) Examples where the rule-based approach fails to detect sarcasm but machine learning-based approach detects it, (B) Examples where the machine learning-based approach fails to detect sarcasm but deep learning-based approach detects it, and (C) Examples where none of the approaches detect the sarcasm. Table 6.4 shows the proportion of errors for each of the three configurations. The ad-hoc nature of the rule-based approach reflects in percentage values in (A). Similarly, the interplay of numbers and number as a sarcasm highlight appear as useful pointers for future work.

Summary

This chapter described approaches to handle a special case of sarcasm: sarcasm expressed using numbers. Approaches pertaining to three paradigms of NLP were presented: (a) rule-based that capture ranges for sarcastic/non-sarcastic, (b) statistical machine learning-based that combine standard features with number-related features to make decision, and (c) deep learning-based (CNN, LSTM and attention-based networks). We first showed

	(A)	(B)	(C)
Sarcasm not due to numbers	34	32	10
Number as a sarcasm highlight	12	22	20
Interplay of numbers	4	12	12
Unseen situations	32	14	18
'Special' numbers	12	12	30
No incongruity within text	6	8	10

Table 6.4: Percentage of errors across different error categories for three configurations; (A): Rule-based approach goes wrong but statistical approach is correct, (B): Statistical approach goes wrong but deep learning-based approach is correct, (C) : All three approaches go wrong

that past works in sarcasm detection did not perform well for text containing numbers. We then compared our approaches with four baselines. The best performance was obtained with the deep learning-based model on Dataset 1, with the F-score standing at 0.93. SVM performed best on Dataset 2, with the F-score standing at 0.84.

Our error analysis points to specific numerical sarcasm challenges, thus creating immediate future tasks. Long term future work consists in tackling irony in general, humour and humble bragging (*'Oh my life is miserable: I have to sign 500 autographs a day'*) all of which have their genesis in incongruity. The utility of our work lies in the fact that our system is a crucial link in a pipeline for sarcasm detection, wherein a tweet labeled as non-sarcastic and containing a number gets a final check with respect to it being sarcastic.

Chapter 7

Converting Sarcastic Utterances into their Non-Sarcastic Interpretation

This chapter is based on the paper we submitted in CODS-COMAD 2019 [1].

Sarcasm is a form of speech in which the the implied sentiment is the opposite of literal meaning. In this chapter, we present the task of sarcasm interpretation, defined as converting a sarcastic utterance into its non-sarcastic interpretation. We present three approaches for the task: (a) a rule-based approach that considers sarcasm as a form of dropped negation and associate negation words with verbs present in the sarcastic utterance, (b) statistical machine translation-based (SMT) approach that address the sarcasm interpretation task as monolingual machine translation and (c) three deep learning-based (DL) architectures, Encoder-Decoder Network, Attention Network and Pointer Generator Network. We also discuss the scope of future work to further enhance the proposed models for sarcasm interpretation.

Sarcasm is a peculiar challenge to sentiment analysis because it uses verbal irony to express contempt or ridicule, thereby, potentially confusing typical sentiment classifiers. Several approaches for sarcasm detection have been reported in the recent past [20]. In this chapter, we focus on approaches which convert sarcastic utterances into their non-sarcastic interpretation. In other words, the goal of this chapter is the task where a sarcastic tweet is converted to its non-sarcastic interpretation. For example, the sentence

[1] http://cods-comad.in/2019/index.html

'I love being ignored' is sarcastic and one of it's possible intended interpretation can be *'I hate being ignored'*. The incongruity between the positive sentiment word *'love'* and negative situation *'being ignored'* plays a crucial role in identifying sarcasm.

In this chapter, we present the task of interpretation of sarcastic utterances. Towards this, we first introduce the task, identify its challenges, and devise three approaches for the task. Our approaches are based on three prevalent paradigms of NLP: statistical machine translation-based, deep learning-based approaches and rule-based approach.

The contribution of the chapter is as follows:

1. We systematically investigate sarcasm interpretation and details out the purpose and challenges of the problem. We believe that this paper will motivate future research.
2. We extend the problem introduced in [41] and present a suite of approaches for sarcasm interpretation task.
3. We present a novel rule-based algorithm for the task. We also explore state of the art deep learning-based and statistical machine translation-based architectures for the task of sarcasm interpretation.
4. Finally, our results spanning three prevalent paradigms serve as a strong baseline for future work in interpreting sarcasm.

7.1 Motivation

In today's world, sarcasm is ubiquitous, ranging from social networking websites like Facebook, Twitter etc. to product reviews on e-commerce giants like Amazon, Flipkart etc. sarcasm is present everywhere. Sarcastic humor is downright priceless and applicable to many of our own situations.

However, sarcasm is hard to interpret, especially non-verbal sarcasm. [22] shows that sarcasm may not be understood by people from some cultures. Sarcasm expressed in a native language is difficult to interpret by non native speakers. We describe approaches in this paper that will help in understanding sarcasm. In verbal communication, sarcastic utterances are accompanied by a certain tone of voice and facial expressions (For eg., *rolling of eyes*). However, in textual communication, these cues are absent which makes identification and interpretation of sarcasm very challenging even for humans. Sarcasm on the internet is hard to interpret because of the following reasons:

1. Speaker's body language is unknown which is a major part of how people communicate with each other.

2. Tone of voice makes a huge difference. Words on a computer screen and face to face conversation are very different.

3. Every sentence can be a sarcastic one for a particular context.

All these factors makes it difficult to interpret sarcasm. This is why understanding the actual meaning from a sarcastic utterance is a very interesting and challenging problem. Therefore, to understanding sarcasm, it is important to first convert sarcastic text into its non-sarcastic form. The non-sarcastic form can be analyzed further using existing sentiment analysis systems to understand the original intention of the speaker. Towards this, we introduce the task of sarcasm interpretation, defined as the generation of non-sarcastic text conveying the intended message as the original sarcastic one. Since, manual processing of large amount of data is not an easy task, certain rule-based, statistical machine learning-based and deep learning-based techniques can be used for this purpose. These techniques analyze the text and convert sarcastic text into its non-sarcastic interpretation.

7.2 Approaches

In this section, we describe three approaches for the problem: statistical machine translation-based, deep learning-based and rule-based.

7.2.1 Dataset Details

The authors of [41] collected a data set, first of its kind, of sarcastic tweets and their non-sarcastic (literal) interpretations. The data set is publicly available and we have used this dataset for our experiments. The data set consists of tweets marked with the content tag #sarcasm, posted between January and June of 2016. The tweets in this data set are plain text without any images and URLs. There are 3000 sarcastic tweets containing text only, where the average sarcastic tweet length is 13.87 utterances. For each sarcastic tweet there are five literal interpretations present in the data set. This results in a monolingual parallel corpora with 15000 sentences. We call this Dataset 1. From Dataset 1, we create another dataset by selecting short tweets (up to 10 words) of the form *'i am ...'*, *'i m ...'*, *'he is ...'*, *'he s ...'*, *'you are ...'*, *'she is ...'*, *'she s ...'*, *'you are ...'*, *'you re...'*, *'we are ...'*, *'we re ...'*, *'they are ...' 'they re ...'* and their literal interpretation. We call this Dataset 2.

Example from parallel sarcastic corpus is shown below.

- Sarcastic Tweet: *What a great way to end my night.*
- Literal Interpretations:
  1. *Such a bad ending to my night.*
  2. *Oh what a great way to ruin my night.*
  3. *What a horrible way to end a night.*
  4. *Not a good way to end the night.*
  5. *Well that wasn't the night I was hoping for.*

7.2.2 Statistical Machine Translation-Based Approach

In this approach, we model the task of sarcasm interpretation as a monolingual machine translation task. We use Moses [29] which is a tool for phrase-based MT. To build the phrase table, phrases of up to 8 words are used. We use SRILM[2] with kneser-ney discounting[3] for language modeling and this language model is trained on the non-sarcastic tweet interpretations (the target side of the parallel corpus).

The corpus is divided into training, development and test sets of sizes 12000, 1500 and 1500 respectively. Then, we train Moses on the training set and tune the model parameters on the development set.

[2]http://www.speech.sri.com/projects/srilm/
[3]https://en.wikipedia.org/wiki/Kneser-Ney_smoothing

7.2.3 Deep Learning-Based Approaches

- **Encoder Decoder Network:** [52] is used as a reference for implementing this model. We use Gated Recurrent Units [7] for both encoder and decoder. Figure 7.1 illustrate this model.[4]

- **Attention Network:** Figure 7.2 shows the architecture of our Attention Network. Attention Network produce exceptional results for short tweets, but when the length of tweets get longer, we observe two key problems with our attention network. They are as follows:

  1. Problem 1: The summaries sometimes reproduce factual details inaccurately. For example, for the sarcastic sentence *Dhoni scored 127, his days are long gone*, the output of attention network is *Dhoni scored 111 his days are not long gone*. This is especially common for rare or out-of-vocabulary words such as 127.

  2. Problem 2: The summaries sometimes repeat themselves. For example, for the sarcastic sentence *It's 4 in the morning and I'm having a head ache... great way to start my day*, the output of attention network is *morning head ache are ache are ache are...*

  We handle these issues with Pointer Generator Network, which we explain next.

- **Pointer Generator Network:** [50] introduce pointer generator network for text summarization. Since, ours is a monolingual machine translation task, apart from generating a translation from scratch, copying from original source sentence also plays a key role. Pointer generator network handle above mentioned problems as follows:

  1. Pointer Generator Network handle problem 1 by choosing to copy words from the source via pointing, while retaining the ability to generate words from the fixed vocabulary.

[4] This figure is taken from [52]

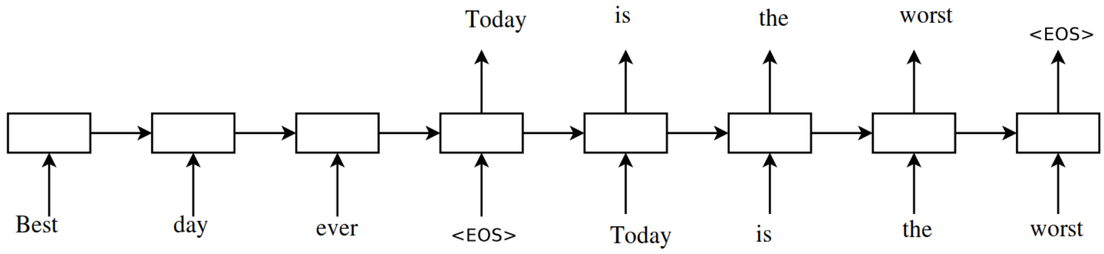

Figure 7.1: Encoder Decoder Network

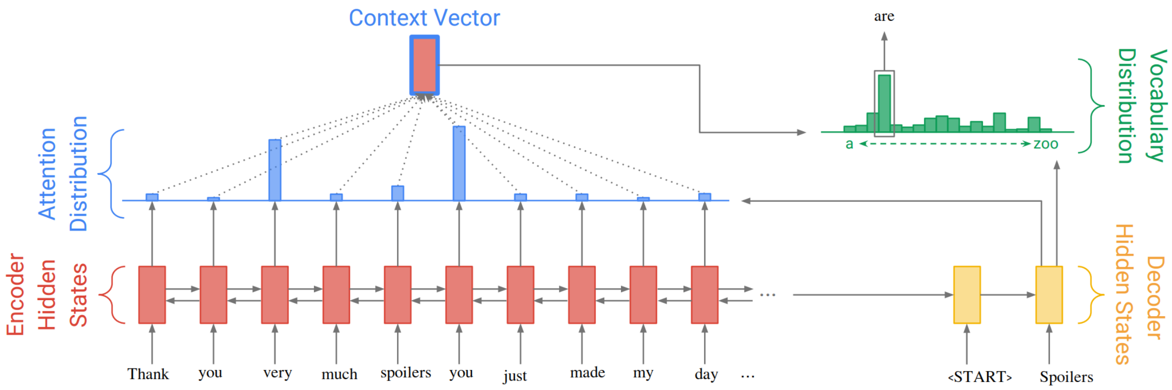

Figure 7.2: Attention Network

2. Pointer Generator Network handle problem 2 by using a technique called coverage. The idea is to penalize the network for attending to same parts again.

7.2.4 Rule-Based Approach

Our rule-based approach is based on a linguistic theory that considers sarcasm as a form of dropped negation [15]. This approach is centered around finding the right place to insert a negation word. In other words, when one expresses sarcasm, a negation is intended, despite the lack of a negation word like 'not'. For example, the literal interpretation of the sarcastic sentence *'Being awake at 4 am with a head-ache is fun'* is the non-sarcastic sentence *'Being awake at 4 am with a head-ache is not fun'.* This results in the possibility that many sarcastic sentences could be converted to non-sarcastic by simply applying an appropriate negation.

In this approach, we associate negation words with verbs present in the sarcastic utterance. We utilize a list of negation words for converting each sarcastic sentence into its non-sarcastic interpretation. The key idea is to associate a negation word with the verb. For each such combination between a negation word and a verb, we calculate perplexity[5]. For calculating perplexity, we create a bigram language model using words in the training corpus. Output sentence is the one with lowest perplexity. algorithm 1 illustrates this process.

[5] https://en.wikipedia.org/wiki/Perplexity

Algorithm 1: Rule Based Approach

```
Input: Sarcastic statement
Output: Non Sarcastic interpretation
Sarcasm Interpretation(s):
negation_words = ['do-not', 'does-not', 'did-not', 'not', 'never', 'neither', 'nor']
/* corpus is entire training corpus */
/* word_tokenize will return tokens from text */
corpus_tokens = word_tokenize(corpus)
lm = create_language_model(tokens)
pp ⟵ ∞
sentence_tokens = word_tokenize(s) tagged = pos_tag(tokens)
for each word in s do
    if word is a verb then
        for each negation_word do
            new_sentence ⟵ sentence after placing negation word before verb
            if pp <get_getperplexity(new_sentence) then
                pp = get_perplexity(new_sentence)
                output_sentence = new_sentence
            new_sentence ⟵ sentence after placing negation word after verb
            if pp <get_getperplexity(new_sentence) then
                pp = get_perplexity(new_sentence)
                output_sentence = new_sentence
return output_sentence
```

7.3 Results

In this section, we evaluate our approaches for converting sarcastic text into their non-sarcastic interpretation. Table 7.1 compares results of three approaches on Dataset 1 and Dataset 2. We observe that, among the deep learning-based approaches, encoder-decoder network perform better on average with BLEU score of **53.60** and **78.28**, METEOR score of **30.46** and **44.62**, and ROUGE-L score of **54.60** and **80.89** on Dataset 1 and Dataset 2 respectively. We also observe that our statistical machine translation-based approach which used MOSES outperforms other approaches on Dataset 1 and obtains a BLEU score of **67.96**, METEOR score of **34.43** and ROUGE-L score of **68.81**. On Dataset 2, we observe that MOSES obtains the best BLEU score of **78.57** whereas our attention network obtains the best METEOR score of **45.02** and our Encoder-Decoder network obtains the best ROUGE-L score of **80.89**.

Visualization of attention weights for example outputs from our Attention Network is shown in Figure 7.3.

Approaches	Dataset 1			Dataset 2		
	BLEU	METEOR	ROUGE-L	BLEU	METEOR	ROUGE-L
Rule-Based Approach	50.98	23.57	50.91	63.41	41.27	66.97
Statistical Machine-Translation-based Approach (Moses)	**67.96**	**34.43**	**68.81**	**78.57**	43.21	79.91
Encoder-Decoder Network	53.60	30.46	54.60	78.28	44.62	**80.89**
Attention Network	25.77	15.32	29.22	74.19	**45.02**	78.49
Pointer Generator Network	29.22	16.71	31.27	74.21	44.89	79.08

Table 7.1: Comparison of Results of different approaches on **Dataset 1** and **Dataset 2**.

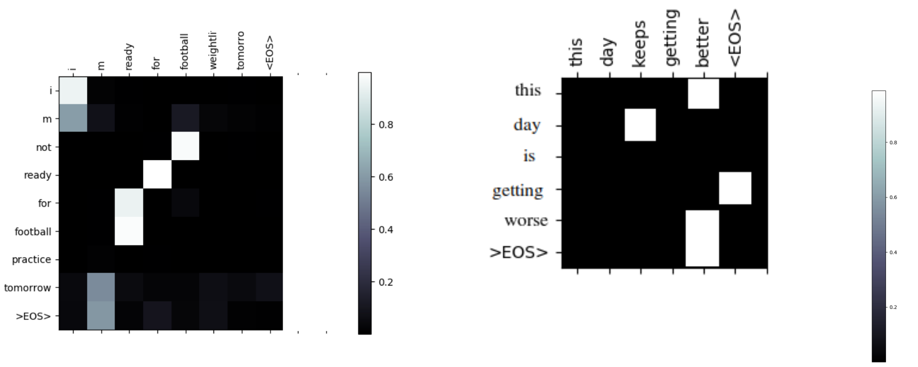

Figure 7.3: Visualization of Attention Weights for example outputs.

Summary

This chapter described approaches that can interpret sarcasm. Approaches pertaining to three paradigms of NLP were presented: (a) rule-based, (b) statistical machine translation-based, and (c) deep learning-based (Encoder-Decoder network, Attention network and Pointer Generator network).

Chapter 8

Conclusions and Future Work

Sarcasm detection research has grown significantly in last few years. First, we have seen how sarcastic utterances poses problems for existing state of the art sentiment analysis systems. Then, we described two problems (1) the problem of automatic sarcasm detection in text containing numbers and (2) the problem of sarcasm interpretation. We described the notion of incongruity and how context incongruity and sarcasm are related to each other.

In this report, we described approaches to handle a special case of sarcasm: sarcasm expressed using numbers. Approaches pertaining to three paradigms of NLP were presented: (a) rule-based that capture ranges for sarcastic/non-sarcastic, (b) statistical machine learning-based that combine standard features with number-related features to make decision, and (c) deep learning-based (CNN, LSTM and attention-based networks). We first showed that past works in sarcasm detection did not perform well for text containing numbers. We then compared our approaches with four baselines. The best performance was obtained with the deep learning-based model on Dataset 1, with the F-score standing at *0.93*. SVM performed best on Dataset 2, with the F-score standing at *0.84*.

Our error analysis points to specific numerical sarcasm challenges, thus creating immediate future tasks. Long term future work consists in tackling irony in general, humour and humble bragging (*'Oh my life is miserable: I have to sign 500 autographs a day'*) all of which have their genesis in incongruity. The utility of our work lies in the fact that our system is a crucial link in a pipeline for sarcasm detection, wherein a tweet labeled as non-sarcastic and containing a number gets a final check with respect to it being sarcastic.

For sarcasm interpretation task, we presented a description of experiments, observations and analyses in the endeavour to implement a system that is capable of interpreting and converting sarcastic statements to non sarcastic statements. We explained why this problem would be difficult to solve and how we can use machine translation-based, deep learning-based and rule based approaches to interpret sarcasm.

First we focused on interpreting sarcasm by posing it as a monolingual machine translation problem. We trained a phrase based MT system using Moses for this task. Then, we devised a novel rule based approach which interprets sarcasm by associating negation words with verbs in the sentence. Finally, we explored three deep learning-based architectures. We then compared the results of three approaches and presented sample results.

Our error analysis points to specific sarcasm interpretation challenges. There are a few caveats where there is a lot of scope for improvement. In our rule-based approach, a negation word is associated with various verbs in the sentence and the final word-verb combination is selected based on perplexity. Future work involves analyzing dependency based parse trees of sarcastic tweets to get more insight on proper placement of negation words. We also aim to incorporate more negation words and take adjectives into account while associating negation words.

We strongly believe that, given a larger parallel corpus of sarcastic tweets and their non-sarcastic interpretations, deep learning-based architectures will perform significantly better than other approaches. The data set used for this project is rather small for training deep learning-based architectures, thus creating immediate future tasks. We aim to gather more sarcastic tweets using Twitter API and create a larger parallel corpora of sarcastic tweets and their non-sarcastic interpretations.

In addition to BLEU score, METEOR score and ROUGE-L score we look forward to incorporate more evaluation metrics such as word error rate and human level evaluation to evaluate our models more thoroughly.

Bibliography

[1] Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *CoRR* abs/1607.00976.

[2] Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976* .

[3] Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. 2014. Italian irony detection in twitter: a first approach. In *The First Italian Conference on Computational Linguistics CLiC-it*. page 28.

[4] Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access* 4:5477–5488.

[5] Konstantin Buschmeier, Philipp Cimiano, and Roman Klinger. 2014. An impact analysis of features in a classification approach to irony detection in product reviews. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. pages 42–49.

[6] Paula Carvalho, Luís Sarmento, Mário J Silva, and Eugénio De Oliveira. 2009. Clues for detecting irony in user-generated contents: oh...!! it's so easy;-. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*. ACM, pages 53–56.

[7] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger

Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR* abs/1406.1078.

[8] Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the fourteenth conference on computational natural language learning*. Association for Computational Linguistics, pages 107–116.

[9] Elena Filatova. ???? Irony and sarcasm: Corpus generation and analysis using crowdsourcing. Citeseer.

[10] Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of NAACL-HLT*. pages 161–169.

[11] Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *WASSA@NAACL-HLT*.

[12] Aniruddha Ghosh and Tony Veale. 2017. Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal. In *EMNLP*.

[13] Debanjan Ghosh, Weiwei Guo, and Smaranda Muresan. 2015. Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 1003–1012.

[14] Raymond W Gibbs. 1986. On the psycholinguistics of sarcasm. *Journal of Experimental Psychology: General* 115(1):3.

[15] Rachel Giora. 1995. On irony and negation. *Discourse Processes* 19(2):239–264.

[16] Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '11, pages 581–586.

[17] Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*. Association for Computational Linguistics, pages 581–586.

[18] Stacey L Ivanko and Penny M Pexman. 2003. Context incongruity and irony processing. *Discourse Processes* 35(3):241–279.

[19] Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 217–226.

[20] Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Automatic sarcasm detection: A survey. *CoRR* abs/1602.03426.

[21] Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Automatic sarcasm detection: A survey. *arXiv preprint arXiv:1602.03426* .

[22] Aditya Joshi, Pushpak Bhattacharyya, Mark James Carman, Jaya Saraswati, and Rajita Shukla. 2016. How do cultural differences impact the quality of sarcasm annotation?: A case study of indian annotators and american text. In *LaTeCH@ACL*.

[23] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *ACL*.

[24] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *ACL (2)*. pages 757–762.

[25] Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Harnessing sequence labeling for sarcasm detection in dialogue from tv series 'friends'. In Yoav Goldberg and Stefan Riezler, editors, *CoNLL*. ACL, pages 146–155.

[26] Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Harnessing sequence labeling for sarcasm detection in dialogue from tv series 'friends'. In *CoNLL*.

[27] Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? *arXiv preprint arXiv:1610.00883* .

[28] Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark James Carman. 2016. Are word embedding-based features useful for sarcasm detection? *CoRR* abs/1610.00883.

[29] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '07, pages 177–180.

[30] Roger J Kreuz and Gina M Caucci. 2007. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on computational approaches to Figurative Language*. Association for Computational Linguistics, pages 1–4.

[31] Lakshya Kumar, Arpan Somani, and Pushpak Bhattacharyya. 2017. "having 2 hours to write a paper is fun!": Detecting sarcasm in numerical portions of text. *CoRR* abs/1709.01950.

[32] Stacey L. Ivanko and Penny Pexman. 2003. Context incongruity and irony processing 35:241–279.

[33] Stacey L. Ivanko and Penny Pexman. 2003. Context incongruity and irony processing 35:241–279.

[34] CC Liebrecht, FA Kunneman, and APJ van Den Bosch. 2013. The perfect solution for detecting sarcasm in tweets# not .

[35] Peng Liu, Wei Chen, Gaoyan Ou, Tengjiao Wang, Dongqing Yang, and Kai Lei. 2014. Sarcasm detection in social media based on imbalanced classification. In Feifei Li, Guoliang Li, Seung-won Hwang, Bin Yao, and Zhenjie Zhang, editors, *Web-Age Information Management*. Springer International Publishing, Cham, pages 459–471.

[36] Stephanie Lukin and Marilyn Walker. 2017. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. *arXiv preprint arXiv:1708.08572* .

[37] Diana Maynard and Mark A. Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *LREC*.

[38] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality .

[39] Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2017. Harnessing cognitive features for sarcasm detection. *arXiv preprint arXiv:1701.05574* .

[40] Sana Parveen, Avinash Surnar, and Sunil Sonawane. ???? Opinion mining in twitter: How to make use of sarcasm to enhance sentiment analysis: A .

[41] Lotem Peled and Roi Reichart. 2017. Sarcasm SIGN: interpreting sarcasm with sentiment based monolingual machine translation. *CoRR* abs/1704.06836.

[42] Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. *arXiv preprint arXiv:1610.08815* .

[43] Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *WSDM 2015 - Proceedings of the 8th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, Inc, pages 97–106.

[44] Rachel Rakov and Andrew Rosenberg. 2013. ” sure, i did the right thing”: a system for sarcasm detection in speech.

[45] Antonio Reyes and Paolo Rosso. 2012. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems* 53(4):754–760.

[46] Antonio Reyes and Paolo Rosso. 2014. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems* 40(3):595–614.

[47] Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Lang. Resour. Eval.* 47(1):239–268.

[48] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *EMNLP*. volume 13, pages 704–714.

[49] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. Association for Computational Linguistics (ACL), pages 704–714.

[50] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *CoRR* abs/1704.04368.

[51] Georgios P. Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. *CoRR* abs/1805.08154.

[52] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR* abs/1409.3215.

[53] Joseph Tepperman, David Traum, and Shrikanth Narayanan. 2006. " yeah right": Sarcasm recognition for spoken dialogue systems. In *Ninth International Conference on Spoken Language Processing*.

[54] Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. Icwsm-a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews.

[55] Mark Turner. 1995. Raymond w. gibbs, jr.,the poetics of mind: Figurative thought, language, and understanding. *Pragmatics and Cognitionpragmatics and Cognition* 3(1):181–187.

[56] Akira Utsumi. 2000. Verbal irony as implicit display of ironic environment: Distinguishing ironic utterances from nonirony. *Journal of Pragmatics* 32(12):1777–1806.

[57] Tony Veale and Yanfen Hao. ???? Detecting ironic intent in creative comparisons.

[58] Zelin Wang, Zhijian Wu, Ruimin Wang, and Yafeng Ren. 2015. Twitter sarcasm detection exploiting a context-based model. In *Proceedings, Part I, of the 16th International Conference on Web Information Systems Engineering — WISE 2015 - Volume 9418*. Springer-Verlag New York, Inc., New York, NY, USA, pages 77–91.

[59] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*.

[60] Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of the 26th International Conference on Computational Linguistics*. pages 2449–2460.

Acknowledgements

I would like to thank my guide **Prof. Pushpak Bhattacharyya** and my co-guide **Dr. Aditya Joshi** for always helping me and pointing me in the right direction whenever I was stuck.

I would also like to thank the CSE department in particular for inculcating a strong work ethic and providing high-quality education.

Signature: ....

Abhijeet Dubey

16305R006

Date: ...... October 2018