

Investigations in Computational Sarcasm

Submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
of the
Indian Institute of Technology, Bombay, India
and

Monash University, Australia

by

Joshi Aditya Madhav Manda

Supervisors:

Pushpak Bhattacharyya (IIT Bombay)

Mark J Carman (Monash University)





The course of study for this award was developed jointly by the Indian Institute of Technology, Bombay and Monash University, Australia and given academic recognition by each of them. The programme was administered by The IITB-Monash Research Academy.

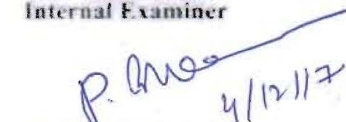
2017


Approval Sheet

The thesis entitled "Investigations in Computational Sarcasm" by Joshi Aditya Madhav Manda
approved for the degree of **Doctor of Philosophy**


Preeti Joshi
External Examiner


Preeti Joshi
Internal Examiner


P. Anand 4/12/17
IITB Supervisor


MARK CARMAN
Monash Supervisor


Mathias Kulkarni
Chairman

Date: 20 November, 2017
Place: Mumbai, India.

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY, INDIA

CERTIFICATE OF COURSE WORK

This is to certify that **Joshi Aditya Madhav Manda** (Roll No. 124054003) was admitted to the candidacy of Ph.D. degree on 02-01-2013, after successfully completing all the courses required for the Ph.D. programme. The details of the course work done are given below.

S.No	Course Code	Course Name	Credits
1	CS712	Topics in Natural Language Processing	6
2	CS726	Advanced Machine Learning	6
3	CSS801	Seminar	4
		Total Credits	16

IIT Bombay

Date:

Dy. Registrar (Academic)

Declaration

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.

Notice 1

Under the Copyright Act 1968, this thesis must be used only under the normal conditions of scholarly fair dealing. In particular no results or conclusions should be extracted from it, nor should it be copied or closely paraphrased in whole or in part without the written consent of the author. Proper written acknowledgement should be made for any assistance obtained from this thesis.

Notice 2

I certify that I have made all reasonable efforts to secure copyright permissions for third-party content included in this thesis and have not knowingly added copyright content to my work without the owner's permission.



Student Name Joshi Aditya Madhav Manda
IITB ID: 124054003
Monash ID: 25103385

To my parents (*Aai* and *Baba*)
for their courage and mine.

Abstract

Sarcasm is verbal irony that is intended to mock or ridicule. Existing sentiment analysis systems show a degraded performance in case of sarcastic text. Hence, computational sarcasm has received attention from the sentiment analysis community. Computational sarcasm refers to computational techniques that deal with sarcastic text. This thesis presents our investigations in computational sarcasm based on the linguistic notion of incongruity. For example, the sentence ‘*I love being ignored*’ is sarcastic because the positive word ‘*love*’ is incongruous with the negative phrase ‘*being ignored*’. These investigations are divided into three parts: understanding the phenomenon of sarcasm, sarcasm detection and sarcasm generation.

To first understand the phenomenon of sarcasm, we consider two components of sarcasm: implied negative sentiment, and presence of a target. To understand how implied negative sentiment plays a role in sarcasm understanding, we present an annotation study which evaluates the quality of a sarcasm-labeled dataset created by non-native annotators. Following this, in order to show how the target of sarcasm is important to understand sarcasm, we first describe an annotation study which highlights the challenges in distinguishing between sarcasm and irony (since irony does not have a target while sarcasm does), and then present a computational approach that extracts the target of a sarcastic text.

We then present our approaches for sarcasm detection. To detect sarcasm, we capture incongruity in two ways: ‘*intra-textual incongruity*’ where we look at the incongruity within the text to be classified (*i.e.*, target text), and the ‘*context incongruity*’ where we incorporate information outside the target text. To detect incongruity within the target text, we present four approaches: (a) A classifier that captures sentiment incongruity using sentiment-based features (as in the case of ‘*I love being ignored*’), (b) A classifier that captures semantic incongruity (as in the case of ‘*A woman needs a man like a fish needs bicycle*’) using word embedding-based features, (c) A topic model that captures sentiment incongruity using sentiment distributions in the text (in order to discover sarcasm-prevalent topics such as work, college, etc.), and (d)

An approach that captures incongruity in the language model using sentence completion. The approaches in (a) and (c) incorporate sentiment incongruity relying on sentiment-bearing words, whereas approach in (b) and (d) tackle other forms of incongruity where sentiment-bearing words may not be present.

On the other hand, to detect sarcasm using contextual incongruity, we describe two approaches: (a) A rule-based approach that uses historical text by an author to detect sarcasm in the text generated by them, and (b) A statistical approach that uses sequence labeling techniques for sarcasm detection in dialogue. The approach in (a) attempts to detect sarcasm that requires author-specific context while that in (b) attempts to detect sarcasm that requires conversation-specific context. Finally, we present a technique for sarcasm generation. In this case, we use a template-based approach to synthesize incongruity and generate a sarcastic response to user input.

Our investigations demonstrate how evidences of incongruity (such as sentiment incongruity, semantic incongruity, etc.) can be modeled using different learning techniques (such as classifiers, topic models, etc.) for sarcasm detection and sarcasm generation. In addition, our findings establish the promise of novel problems like sarcasm target identification and sarcasm versus irony classification, and provide insights for future research in sarcasm detection.

Contents

Abstract	v
List of Tables	xii
List of Figures	xvi
1 Introduction	2
1.1 Sentiment Analysis (SA)	2
1.2 Sarcasm & Computational Sarcasm	3
1.3 Motivation	5
1.3.1 Turing test-completeness	5
1.3.2 Impact on Sentiment Classification	6
1.4 Prevalence of Sarcasm	7
1.4.1 In Popular Culture	7
1.4.2 On the Web	9
1.5 Sarcasm Studies in Linguistics	9
1.6 Incongruity for Sarcasm	13
1.7 Contribution	16
1.8 Thesis Organization	18
1.9 Publications	18
2 Literature Survey	22
2.1 Problem Definition	23
2.2 Datasets	23
2.2.1 Short text	25
2.2.2 Long text (Passages)	27

2.2.3	Other datasets (Dialogues, syntactic patterns, etc.)	27
2.3	Approaches	28
2.3.1	Rule-based Approaches	28
2.3.2	Statistical Approaches	29
2.3.3	Deep Learning-based Approaches	30
2.3.4	Shared Tasks & Benchmark Datasets	32
2.4	Reported Performance	32
2.5	Trends in Sarcasm Detection	32
2.5.1	Pattern discovery	34
2.5.2	Role of Context in Sarcasm Detection	35
2.6	Issues in Sarcasm Detection	36
2.6.1	Issues with Annotation	36
2.6.2	Issues with Sentiment as a Feature	37
2.6.3	Dealing with Dataset Skews	37
2.7	Summary	38
3	Understanding the Phenomenon of Sarcasm	40
3.1	Impact on Cross-Cultural Annotation	40
3.1.1	Motivation	41
3.1.2	Experiment Description	42
3.1.3	Analysis	43
3.2	Sarcasm-versus-irony Classification	48
3.2.1	Motivation	49
3.2.2	Experiment Description	50
3.2.3	Analysis	50
3.3	An Approach for Identification of the Sarcasm Target	53
3.3.1	Sarcasm Target	54
3.3.2	Motivation	55
3.3.3	Architecture	56
3.3.4	Experiment Description	63
3.3.5	Results & Analysis	65
3.4	Summary	69

4	Sarcasm Detection Using Incongruity Within Target Text	72
4.1	Sentiment Incongruity as Features	73
4.1.1	Motivation	73
4.1.2	Sentiment Incongruity-based Features	74
4.1.3	Experiment Setup	76
4.1.4	Results	78
4.1.5	Error Analysis	80
4.2	Semantic Incongruity as Features	81
4.2.1	Motivation	82
4.2.2	Word Embedding-based Features	82
4.2.3	Experiment Setup	84
4.2.4	Results	86
4.2.5	Error Analysis	88
4.3	Sentiment Incongruity Using Topic Model	89
4.3.1	Motivation	90
4.3.2	Model	91
4.3.3	Experiment Setup	94
4.3.4	Results	95
4.3.5	Application to Sarcasm Detection	100
4.4	Language Model Incongruity Using Sentence Completion	102
4.4.1	Motivation	106
4.4.2	Approach	107
4.4.3	Experiment Setup	109
4.4.4	Results	110
4.4.5	Discussion	112
4.4.6	Error Analysis	112
4.5	Comparative Analysis of Approaches	113
4.5.1	Individual Modules	113
4.5.2	Majority-based prediction	115
4.6	Summary	116

5	Sarcasm Detection Using Contextual Incongruity	119
5.1	Contextual Incongruity in a Monologue	119
5.1.1	Motivation	120
5.1.2	Architecture	121
5.1.3	Experiment Setup	124
5.1.4	Results	125
5.1.5	Error Analysis	126
5.2	Contextual Incongruity in Dialogue	127
5.2.1	Motivation	128
5.2.2	Prior Work	128
5.2.3	Architecture	129
5.2.4	Conversational Sarcasm Dataset	131
5.2.5	Paradigm 1: Traditional Models	134
5.2.6	Paradigm 2: Deep Learning-based Models	137
5.2.7	Results	141
5.2.8	Error Analysis	144
5.3	Comparative Analysis of Approaches	147
5.4	Summary	149
6	Sarcasm Generation	151
6.1	Motivation	152
6.2	Architecture	152
6.2.1	Input Analyzer	153
6.2.2	Generator Selector	154
6.2.3	Sarcasm Generator	155
6.3	Evaluation	158
6.3.1	Experiment Details	159
6.3.2	Results	160
6.4	Summary	161
7	Conclusion & Future Work	162
7.1	Summary	162
7.2	Conclusion	165

7.3 Future Work	168
Bibliography	169
Acknowledgments	182

List of Tables

1.1	Degradation in performance of sentiment classification in case of sarcastic text	6
1.2	Relationship between sarcasm, deception, metaphor and irony as given in Gibbs [1994]	14
1.3	An example scenario where incongruity may occur at different levels of difficulty	15
2.1	Summary of sarcasm detection along different parameters	24
2.2	Summary of sarcasm-labeled datasets	25
2.3	Summary of features used for statistical sarcasm classifiers	31
2.4	Summary of reported performance values; Precision/Recall/F-measures and Accuracy values are indicated in percentages	33
3.1	Examples of sentences that the Indian annotators found difficult to annotate; ‘twitter_handle’ are twitter handles suppressed for anonymity	43
3.2	Inter-annotator agreement statistics for Tweet-A; Avg. American1 is as reported in the original paper	44
3.3	Inter-annotator agreement statistics for Discussion-A	45
3.4	Class-wise agreement (%) for pairs of annotators, for both datasets	45
3.5	Impact of non-native annotation on sarcasm classification	46
3.6	Predicting annotator agreement using textual features; Values are averaged over Indian annotators	47
3.7	Confusion matrix, averaged over three annotators	49
3.8	Change in Cohen’s Kappa Values for the three annotators	49
3.9	Average and weighted average values for three configurations of sarcasm/irony/philosophy classification	52

3.10	Average Precision, Recall and F-score values for sarcastic class for the three configurations of sarcasm/irony/philosophy classification	52
3.11	Examples of sarcasm targets	56
3.12	Summary of rules in the rule-based sarcasm target extractor	60
3.13	Features of learning-based sarcasm target extractor	61
3.14	Statistics of sarcasm target datasets	62
3.15	Performance of sarcasm target identification for individual rules for book snippets ; % Increase shows how much the score increases in case of ‘Conditional’ as compared to ‘Overall’	65
3.16	Performance of sarcasm target identification for individual rules for tweets ; % Increase shows how much the score increases in case of ‘Conditional’ as compared to ‘Overall’	66
3.17	Performance of sarcasm target identification for snippets	68
3.18	Performance of sarcasm target identification for tweets	68
3.19	Evaluation of sarcasm target identification for examples with target outside the text (indicated by ‘Outside’ cases)	69
4.1	Sentiment incongruity features for sarcasm detection	77
4.2	Performance on Tweet-A using rule-based algorithm and statistical classifiers using our feature combinations	79
4.3	Performance on Discussion-A using our feature combinations	79
4.4	Comparison of our sarcasm detection approach using sentiment incongruity with two past works, for Tweet-B	79
4.5	Motivational example for sentiment incongruity in sarcastic text	83
4.6	Performance of unigrams versus our semantic similarity-based features using embeddings from word2vec	85
4.7	Performance obtained on augmenting semantic incongruity features to features from four prior works, for four word embeddings; L: Liebrecht et al. (2013), G: González-Ibáñez et al. (2011a), B: Buschmeier et al. (2014) , J: Joshi et al. (2015)	86
4.8	Average gain in F-Scores obtained by using an intersection of the four word embeddings	88
4.9	Average gain in F-scores for the four types of word embeddings	88

4.10	List of variables/distributions in sarcasm topic model	92
4.11	Sentiment-related topics estimated when the sarcasm topic model is learned on only sarcastic tweets	96
4.12	Topics estimated when the sarcasm topic model is learned on full corpus	97
4.13	Sentiment-related topics estimated when the sarcasm topic model is learned on full corpus	99
4.14	Probability of sentiment label for topics discovered by sarcasm topic model . .	100
4.15	Topics estimated when the sarcasm topic model is learned on only sarcastic tweets	101
4.16	Topics estimated when Supervised LDA is learned on full corpus	102
4.17	Top 5 topics per label when Supervised LDA is learned on full corpus	103
4.18	Topics estimated when LDA model is learned on full corpus	104
4.19	Topics containing ‘sarcasm’ or related indicators, when LDA is learned on full corpus; * : ‘sarcasm’ is present in top 10 words, # : ‘sorrynotsorry’ is present in top 10 words	105
4.20	Topics estimated by LDA model on sarcastic corpus	105
4.21	Performance of topic model-based approach for sarcasm detection	106
4.22	Performance of language model incongruity-based approaches on dataset of tweets; T: Optimal threshold, P: Precision, R: Recall, F: F-score	110
4.23	Performance of language model incongruity-based approaches on dataset of discussion forum posts; T: Optimal threshold, P: Precision, R: Recall, F: F-score	111
4.24	Performance of language model incongruity-based technique when the incon- gruous word is known; T: Optimal threshold, P: Precision, R: Recall, F: F-score	113
4.25	Comparison of individual approaches	114
4.26	Comparison of ensemble-based approach for sarcasm detection; ALL indicates S1 + S2 + S3	115
5.1	Averaged Precision, Recall and F-score of the historical incongruity-based ap- proach using SD1, for four configurations of the integrator	125
5.2	Averaged Precision, Recall and F-score of the historical incongruity-based ap- proach using SD2, for four configurations of the integrator	125
5.3	Positive Precision (PP) and Recall (PR) for historical incongruity-based sar- casm detection using SD1 and SD2; OHTB: Only Historical tweet-based	127
5.4	Percentage of sarcastic utterances for six lead characters in the Friends dataset .	132


5.5	Average surface positive and negative scores for the two classes in the Friends dataset	132
5.6	Percentage of sarcastic and non-sarcastic utterances with actions in the Friends dataset	132
5.7	Dataset statistics for the Friends dataset	133
5.8	Dataset-Derived features for sarcasm detection of dialogue	134
5.9	Comparison of sequence labeling techniques with classification techniques, for features reported in dataset-derived features	141
5.10	Class-wise precision/recall values for all techniques using our dataset-derived features	141
5.11	Feature combinations for which different techniques exhibit their best performance for dataset-derived features	142
5.12	Comparison of sequence labeling techniques with classification techniques, for features reported in Gonzalez-Ibanez et al. (2011)	143
5.13	Comparison of sequence labeling techniques with classification techniques, for features reported in Buschmeier et al. (2014)	144
5.14	Comparison of classification versus sequence labeling architecture; Best reported values from the traditional model paradigm have been mentioned for comparison	145
5.15	Class-wise performance of classification and sequence labeling as formulations for the deep learning paradigm	145
5.16	Proportion of utterances of different types of sarcasm that were correctly labeled by sequence labeling but incorrectly labeled by classification techniques	147
5.17	Comparison of individual approaches, ensembles and sequence labeling-based approach for a dataset of conversational transcripts	148
6.1	Sarcasm generator modules in SarcasmBot	155
6.2	Example outputs of SarcasmBot	158
6.3	Multi-annotator agreement statistics for Experiment 1	160
6.4	Evaluation of SarcasmBot: Results of Experiment 1	160
6.5	Evaluation of SarcasmBot: Results of Experiment 2	160
6.6	Percentage of statements in which the evaluators are able to correctly identify SarcasmBot output, in case of Experiment 2	161


List of Figures


1.1	A taxonomy illustrating the scope of this thesis	16
2.1	Trends in Sarcasm Detection Research	34
3.1	Architecture of our sarcasm target identification approach	57
4.1	Plate diagram of sarcasm topic model	93
4.2	Distribution of word-level sentiment labels for tweet-level labels as extracted by sarcasm topic model	98
5.1	Motivating example for use of author’s historical context	120
5.2	Architecture of sarcasm detection that incorporates historical incongruity in monologue	121
5.3	Our hypothesis for sarcasm detection of conversational text	129
5.4	Example from Friends dataset: Part of a scene	131
5.5	Neural network-based architecture for sarcasm detection as a classification task	138
5.6	Neural network-based architecture for sarcasm detection as a sequence labeling task	138
6.1	Architecture of SarcasmBot	153


Approval Sheet

The thesis entitled "Investigations in Computational Sarcasm" by Joshi Aditya Madhav Manda
approved for the degree of Doctor of Philosophy


Praetima Misra
External Examiner


Preethi Joshi
Internal Examiner

 4/12/17
P. Anand
IITB Supervisor


MARK CARMAN
Monash Supervisor


Mahesh Kulkarni
Chairman

Date: 20 November, 2017
Place: Mumbai, India.

Chapter 1

Introduction

The rise of Web 2.0¹ enabled internet users to generate content, which often contained emotion. Considering the value of this content, automatic prediction of sentiment, *i.e.*, sentiment analysis, became a popular area of research in natural language processing. However, as new approaches were reported, sarcasm was observed to be a crucial challenge for sentiment analysis. This led to the work in computational approaches to process sarcasm, *i.e.*, computational sarcasm. This thesis presents our investigations in computational sarcasm of text.

This chapter is organized as follows. We introduce sentiment analysis (SA) in Section 1.1 followed by computational sarcasm in Section 1.2. We then give the motivation behind computational sarcasm in Section 1.3. We discuss prevalence of sarcasm in popular culture and social media in Section 1.4. We describe linguistic theories of sarcasm in Section 1.5 and specifically, the notion of incongruity in Section 1.6. Incongruity forms the foundation of our work. In Section 1.7, we specify our contribution. Finally, the thesis organization is in Section 1.8. The publications resulting from this work are in Section 1.9.

1.1 Sentiment Analysis (SA)

SA is the task of automatically predicting polarity in text. For example, the sentence ‘*The pizza is delicious*’ should be labeled as positive, while the sentence ‘*The pizza tastes awful*’ should be labeled as negative. The value of SA arises from the opportunity to understand preferences of individuals and communities, using user-generated content on the web.

Several challenges to SA are well-known [Pang and Lee, 2008]. The first challenge is

¹https://en.wikipedia.org/wiki/Web_2.0

negation. A negation marker can make sentiment prediction difficult. For example, the word ‘not’ in the sentence ‘*I do not like this phone*’ negates the sentiment of the verb ‘like’ making the sentence negative. However, in the sentence ‘*I do not like this phone but it’s still the best in its category*’, the negation word ‘not’ negates only the portion before the word ‘but’. Scope of a negation marker has been studied in the past [Harabagiu et al., 2006]. The second challenge to SA is domain dependence. Sentiment of words may differ depending on the domain. For example, the word ‘unpredictable’ is positive for a movie review (for example, ‘*The plot of the movie is unpredictable*’) but negative for an automobile review (for example, ‘*The steering of a car is unpredictable*’). Domain-specific sentiment is a long-studied sub-area of SA [Fahrni and Klenner, 2008]. Similarly, polysemous words may carry different sentiment in different contexts. The word ‘deadly’ may occur in the positive sentence ‘*Shane Warne is a deadly spinner*’ and also in the negative sentence ‘*There are deadly snakes in the Amazon forest*’. Learning classifiers that incorporate polysemous nature of words have also been reported in the past [Balamurali et al., 2011]. Another challenge is **thwarted expectations**. An example of thwarting is: ‘*This city is polluted, has really bad traffic problems, and the weather sucks. However, I grew up here and I love the city.*’ The second sentence reverses the sentiment expressed by the first, although in terms of word count, negative words (‘polluted’, ‘bad’ and ‘sucks’) outnumber the positive words (‘love’). An ontology-based approach to detect thwarting has been reported [Ramteke et al., 2013].

Thus, it can be seen that in addition to approaches for sentiment analysis (in general), there have been explorations that focus on specific challenging aspects (in particular) such as polysemy, domain adaptation, etc. Our work is a similar endeavour that focuses on a specific challenge to sentiment analysis.

1.2 Sarcasm & Computational Sarcasm

The focus of this thesis is another crucial challenge to SA: **Sarcasm**. Sarcasm is verbal irony that expresses negative and critical attitudes toward persons or events [Kreuz and Glucksberg, 1989]. So, the sentence ‘*Your shirt would totally suit my 5 year old nephew!*’ is sarcastic because it is ironic, has an implied negative sentiment, and ridicules the listener (referred by ‘your’ in the sentence). A peculiarity of sarcasm relevant to SA is that it may contain indicators of positive sentiment but imply negative sentiment. For example, the sentence ‘*It’s a lot of fun*

to wait around for 2 hours in the sun has a strong positive phrase ‘a lot of fun’ but an implied negative sentiment.

Other definitions of sarcasm enable one to get a holistic view of the concept. Deshpande [2002] defined sarcasm as a deliberate attempt to point out, question or ridicule attitudes and beliefs by the use of words and gestures in ways that run counter to their normal meanings. This definition has multiple components. The first is that sarcasm is ‘deliberate’ which means that it is purposefully intended by the speaker, and not an interpretation of the listener. That sarcasm ‘points out, questions or ridicules’ means that there is an implied negative sentiment. For example, if a teacher catches a student cheating on a test and says, “*Your parents would be really proud today*”, the implied negative sentiment is towards the listener (the student). The last part of the definition, ‘in ways that run counter to their normal meanings’ highlights the relationship of sarcasm with irony. Irony is a situation in which something which was intended to have a particular result has the opposite or a very different result². Thus, sarcasm is a specific case of irony. Similarly, Gibbs [1994] defined sarcastic language as irony that is especially bitter and caustic. A simplified definition of sarcasm is provided in the Oxford Dictionary: ‘the use of irony to mock or convey contempt’. These definitions show that there are two components of sarcasm: (a) ironic nature, and (b) intention to express contempt. This malicious intent underlying sarcasm has been observed by Irish playwright Oscar Wilde, in his quote, ‘Sarcasm is the lowest form of wit, but the highest form of intelligence’³.

Computational sarcasm refers to computational approaches to process sarcasm⁴. Past work in computational sarcasm deals only with sarcasm detection due to its impact on SA. Sarcasm detection is defined as the computational task of predicting if a given text is sarcastic. So, the sentence ‘I love it when my 15 year old son rolls his eyes at me’ should be labeled as sarcastic, while the sentence ‘I love it when my 15 year old son gets me a present’ should be labeled as non-sarcastic. In this thesis, we introduce another novel problem of sarcasm generation. Sarcasm generation is the computational task of generating a sarcastic response to an input sentence. So, in response to the input sentence ‘What do you think of the new movie?’, we would generate a sarcastic response such as ‘The movie was about as exciting as a two hour math class’.

²Source: The Cambridge Dictionary

³<http://www.goodreads.com/quotes/1284197-sarcasm-is-the-lowest-form-of-wit-but-the-highest>

⁴This name is analogous to ‘Computational Linguistics’

1.3 Motivation

Following the introduction to sarcasm and computational sarcasm, we now discuss why computational sarcasm may be useful. We address this question from two perspectives: relationship with the Turing test and the impact of sarcastic text on sentiment classification.

1.3.1 Turing test-completeness

The Turing test states that an artificially intelligent agent must be able to imitate humans as accurately as possible [Turing, 1956]. This means that it should be able to exhibit human properties. Sarcasm understanding and expression are common human phenomena. Hence, sarcasm-aware computers would be a step towards Turing test-completeness. To elaborate, we present observations from two popular chatbots, ALICE⁵ and Assistant.ai⁶. ALICE is a popular chatbot from early days of chat systems, while Assistant.ai is an assistant for Android phones. The responses by the two for the input ‘*You are slow like a snail*’ are as follows⁷:

ALICE: There is a lot of network traffic right now.

Assistant.ai: A good assistant is whatever their boss needs them to be.

ALICE understands the negative sentiment in ‘*slow like a snail*’ and responds with an explanation. The response by Assistant.ai is mildly humorous but coherent to the input sentence.

The input in the previous case was non-sarcastic. We now consider a sarcastic input: ‘*You are fast like a snail*’. The two chatbots produce the following responses:

ALICE: Thank you for telling me I am fast like a snail.

Assistant.ai: A good assistant is whatever their boss needs them to be.

ALICE does not understand the implied negative sentiment and thanks the human as if the input sentence were a compliment. Assistant.ai repeats the response as in the previous case. These examples show that for current chatbots to meet the requirements levied by the Turing test, an understanding of sarcasm and choice of an appropriate response would be crucial.

⁵<http://alice.pandorabots.com/>

⁶<https://assistant.ai/>

⁷All observations quoted in this section were recorded in December 2015.

System	Dataset	Precision (Sarcastic)	Precision (Overall)
MeaningCloud	Dialogues	20.14	49.41
NLTK	Dialogues	38.86	81
MeaningCloud	Tweets	17.58	50.13
NLTK	Tweets	35.17	69

Table 1.1: Degradation in performance of sentiment classification in case of sarcastic text

1.3.2 Impact on Sentiment Classification

Sarcastic text may be challenging for sentiment classification because of its ironic nature *i.e.*, because positive or neutral sentiment words may be present in a sarcastic sentence, despite the intended sentiment being negative. To validate the impact of sarcastic text on sentiment classification, we compare the performance of two sentiment classifiers for sarcastic and non-sarcastic text. We use two popular sentiment classifiers:

1. MeaningCloud⁸: This is a SaaS product that offers text analytics services, including sentiment prediction⁹.
2. Natural Language Toolkit (NLTK) [Bird, 2006]: This is a research library that provides APIs for several NLP tasks, including sentiment prediction.

We use these classifiers to obtain sentiment predictions for two datasets: a dataset of 283 sarcastic dialogues from a TV series, and a dataset of 506 sarcastic tweets from the dataset given by Riloff et al. [2013]. Both these datasets are manually labeled at the sentence level. For a dataset of general (*i.e.*, not specifically sarcastic) text, we use a set of tweets from the Sentiment140¹⁰ corpus, and a set of dialogues from the same TV series manually labeled with positive/negative/neutral sentiment labels.

Table 1.1 shows the precision of sentiment classification for sarcastic and overall text. The precision of MeaningCloud is 49.41% and 50.13% for dialogues and tweets respectively, in case of overall text. In case of sarcastic tweets, the precision falls from 50.13% to 17.58%. The

⁸<https://www.meaningcloud.com/>

⁹<https://en.wikipedia.org/wiki/MeaningCloud>

¹⁰<http://www.sentiment140.com/>

degradation in performance also holds for NLTK¹¹. The overall precision of NLTK on tweets as reported is 69%. However, for sarcastic tweets, the precision is 35.17%. This degradation holds for all other cases as well. The degraded precision shows that existing sentiment analysis systems may not be able to predict the correct sentiment in case of sarcastic text. This highlights that computational approaches that detect sarcasm will help in improving the performance of sentiment classifiers.

1.4 Prevalence of Sarcasm

The previous section shows why sarcasm needs special attention. In this section, we discuss how prevalent sarcasm is. We first describe occurrences of sarcasm in popular culture, followed by sarcasm on the web (which is the focus of datasets used in our approaches).

1.4.1 In Popular Culture

Sarcasm has been observed in popular culture, including creative works. This section gives examples from around the world.

In a book based on the Indian epic ‘*Mahabharata*’ [Verma, 1986], the author describes a sarcastic remark in the volume ‘*Adi Parva*’ (loose translation: ‘*Book of the Beginning*’). Sage Drona is humiliated by his long-time friend, King Drupada. To avenge the humiliation, Drona captures Drupada’s kingdom with the help of his disciples. He then says to Drupada, “*I have not forgotten that in our boyhood, we have been good friends and have studied together under the same teacher. I seek your friendship even today and give you half of your kingdom which is now mine through conquest*”. Verma [1986] state that Drona pretends to be concerned for Drupada but is actually mocking him instead. Such pretense is known to be a device for sarcastic expression [Camp, 2012].

The iconic Marathi musical play ‘*Sangeet Sharada*’¹² (1899) written by playwright Govind Ballal Deval, features a song replete with sarcasm. The song is famous by its opening lines, ‘*Mhaatara ituka na avaghe*’ (He isn’t old after all). The protagonist of the play, Sharada is a young girl about to get married to a much older man, in an arranged marriage. The song has

¹¹The values for NLTK are as reported in <http://www.laurentluce.com/posts/twitter-sentiment-analysis-using-python-and-nltk/>

¹²https://en.wikipedia.org/wiki/Sangeet_Sharada

her friends teasing her about her to-be husband's age. The opening lines of the song, with their translation, are as follows:

Mhatara ituka na avaghe paaun-she vayman
Lagna ajuni lahaan, avaghe paaun-she vayman
(He isn't old - (he is) 25 less than 100, after all!

He is too young for marriage - 25 less than 100, after all!)

Maya Sarabhai, a lead character in the Indian TV series '*Sarabhai versus Sarabhai*¹³' (2005) is famous for her scathing taunts towards her daughter-in-law, Monisha. Her catchphrase '*Don't mind, beta (child). I am only using you as an example*' is sarcastic and evokes humor in the show.

Similarly, in a famous scene from the Bollywood classic '*Sholay*¹⁴' (1975), Jai meets Basanti's aunt. Jai does not want his best friend to get married to Basanti. So, he pretends to convince Basanti's aunt to agree to the marriage while covertly dissuading her. Jai says to Basanti's aunt, "*(translation) He's a great guy, but who can say no to alcohol? And if he starts gambling after a few drinks, one can't really blame him, can they?*" Thus, Jai uses sarcasm to express contempt towards his friend, so as to avoid the marriage.

Similarly, Chandler is one of the lead characters in a popular American TV show '*Friends*¹⁵'. Chandler's introduction on the Wikipedia page of the show mentions him as a character that uses sarcasm. Another popular TV show '*The Big Bang Theory*¹⁶' has a lead character called Sheldon Cooper who is unable to detect sarcasm and often makes literal interpretations. His inability to understand sarcasm evokes humor in the show.

Sarcasm understanding or the lack of it has received interest from science fiction as well. In an episode of Season 10 of the American TV series, '*The Simpsons*¹⁷', a character named Prof. Frink is seen with a machine labeled '*Sarcasm detector*'. The Professor claims that the machine detects sarcasm. In response, a character seated next to him says, "*Oh, a sarcasm detector, that's a real useful invention*" and the sarcasm detector explodes¹⁸! The American comedy science-fiction show '*Small Wonder*¹⁹' (1985) features VICI, a humanoid robot, as a

¹³https://en.wikipedia.org/wiki/Sarabhai_vs_Sarabhai

¹⁴<https://en.wikipedia.org/wiki/Sholay>

¹⁵<https://en.wikipedia.org/wiki/Friends>

¹⁶https://en.wikipedia.org/wiki/The_Big_Bang_Theory

¹⁷https://en.wikipedia.org/wiki/The_Simpsons

¹⁸http://simpsons.wikia.com/wiki/Sarcasm_Detector

¹⁹[https://en.wikipedia.org/wiki/Small_Wonder_\(TV_series\)](https://en.wikipedia.org/wiki/Small_Wonder_(TV_series))

lead character. Many episodes show how hilarious situations arise because VICI misinterpretes figurative language such as idioms and sarcastic remarks. R2-D2²⁰ is a robot from Star Wars, a Hollywood science-fiction film series. R2-D2 responds with beeps, but have been interpreted by fans as sarcastic responses²¹. This is interesting because without use of words, the robot expresses sarcasm that is understood by the viewers. This points to the fact that sarcasm may be inherent in situations.

1.4.2 On the Web

Since sarcasm is a form of sentiment expression, forums on the web that contain sentiment, also contain sarcasm. These include review websites like `www.rottentomatoes.com`, or social media like Facebook or Twitter.

Twitter²² is a social media platform where users create messages called ‘*tweets*’ that are read by other users. Because sarcasm evokes humor, there exist several twitter handles dedicated to sarcastic tweets. In addition, individual users also write sarcastic tweets to appraise people or situations around them. Twitter users use a technique called a hashtag which is simply a word preceded by the hash (#) symbol²³. Hashtags are a mechanism of adding an index term to a tweet. Therefore, the use of hashtags also enables twitter users to mark their tweets as sarcastic. Hashtags such as #not, #sarcasm, #notserious have been used to express sarcastic intent.

The popular use of hashtags, availability of APIs to download this data and usage permissions for research have been a crucial factor in advancements in sarcasm detection research.

1.5 Sarcasm Studies in Linguistics

Sarcasm is a well-studied phenomenon in linguistics. Grice et al. [1975] described sarcasm as a form of figurative language, *i.e.*, the literal meaning of words does not hold. In case of sarcasm, an interpretation with opposite sentiment is intended. Because of this intended opposite interpretation, sarcasm is a form of irony. Gibbs [1994] defined **three types of irony: verbal, situational and dramatic**. Verbal irony is irony that is expressed in words. For example,

²⁰<https://en.wikipedia.org/wiki/R2-D2>

²¹<http://legionofleia.com/2016/02/r2d2s-beeps-translated-for-full-force-sarcasm/>

²²<http://www.twitter.com/>

²³Hashtags are a mechanism to create reverse indexes. For example, to search a tweet about India, one could search for the hashtag ‘#india’

the sentence ‘*Your paper on automatic grammar correction has many grammatical mistakes*’ is ironic²⁴. Situational irony is irony arising out of a situation. Consider a situation where a scientist discovers a medicine for a disease but succumbs to the disease themselves. The third form of irony is dramatic irony that is used in performances where the audience knows more than the characters and the audience experiences the irony because of this additional knowledge. Several linguistic studies describe various aspects of sarcasm:

1. **Causes of sarcasm:** Campbell and Katz [2012] stated that sarcasm occurs along several dimensions, namely: failed expectation, pragmatic insincerity, negative tension, and presence of a victim. Eisterhold et al. [2006] stated that sarcasm can be understood in terms of the response it elicits. They observe that responses to sarcasm may be laughter, zero response, smile, sarcasm (in return), a change of topic (because the listener was not happy with the caustic sarcasm), literal reply and non-verbal reactions. According to Wilson [2006], sarcasm arises when there is situational disparity between text and a contextual information.
2. **Types of sarcasm:** Camp [2012] show that there are four types of sarcasm:
 - **Propositional:** Such sarcasm appears to be a non-sentiment-bearing proposition but has an implicit sentiment involved (for example, ‘*This phone should have been a paper-weight*’),
 - **Embedded:** This type of sarcasm has an embedded sentiment incongruity in the form of words and phrases themselves (for example, ‘*I love being ignored*’),
 - **Like-prefixed:** A like-phrase provides an implied denial of the argument being made (for example, ‘*Like I care!*’), and
 - **Illocutionary:** This type of sarcasm involves non-textual clues that indicate an attitude different from the sincere interpretation. In such cases, non-textual variations (such as change in pitch) play a role.

Thus, sarcasm is an inherently challenging phenomenon consisting of the types described above.

²⁴A common and distinguishing feature of ironic statements is the humour that results from coincidental situations that appear to have been deliberately constructed to be humorous (self-deprecating, etc.).

3. **Formulations of sarcasm:** Ivanko and Pexman [2003] represent sarcasm as a 6-tuple consisting of $\langle S, H, C, u, p, p' \rangle$ where:

S = Speaker , H = Hearer/Listener

C = Context, u = Utterance

p = Literal Proposition

p' = Intended Proposition

The tuple can be read as '*Speaker S generates an utterance u in Context C meaning proposition p but intending that hearer H understands p'* '. For example, if a teacher says to a student, "*Well, you've done a good job on that assignment, haven't you!*" while the student had not completed the assignment, the student would understand the sarcasm. The 6-tuple representation of this statement is:

S: Teacher, H: Student

C: The student has not completed his/her assignment.

u: "Well, you've done a good job on that assignment, haven't you!"

p: You have done a good job on the assignment.

p': You have done a bad job on the assignment.

A contrary view by Sperber [1984] states that a literal proposition may not always be intended. This can be understood with the help of the sentence '*I love it when I do not forward a chain mail and I die the next day*'. The intention of the speaker is to remind the listener of situations where chain mails do not have any result. It is through this reminder that the speaker's intended ridicule of chain mails is understood. The echoic reminder theory also suggests a similar perspective [Kreuz and Glucksberg, 1989].

4. **Sarcasm as a dropped negation:** Giora [1995] state that irony/sarcasm is a form of negation in which an explicit negation marker is lacking. In other words, when one expresses sarcasm, a negation is intended but a negation word like '*not*' is absent. An interesting implication of this is that a sarcastic sentence can be converted to a non-sarcastic sentence by applying an appropriate negation. For example, the sarcastic sentence '*I love being ignored*' is equivalent to the non-sarcastic sentence '*I do not love being ignored*'.

5. **Sarcasm Understanding:** Gibbs and O'Brien [1991] describe how sarcasm is understood. They state that the violation of truthfulness maxims is a key for a listener to understand sarcasm. For example, '*I love being ignored*' is understood as sarcastic by a listener who believes that being ignored is not a pleasant state to be in. However, '*I love your new shirt!*' may or may not be sarcastic. The sarcasm in this sentence, if any, cannot be understood until the listener assumes the opposite meaning of a literal meaning after they observe that the literal meaning of the text violates truthfulness. To understand sarcasm, if any, in the sentence above, it would be essential to know information that would violate the truthfulness. In case of this new shirt example, if the listener sees that the shirt is stained, the violation of the speaker's truthfulness is likely to convey the sarcasm in the sentence.

Some of these linguistic theories see a close correspondence with advances in automatic sarcasm detection. For example, the additional information that violates truthfulness is what different approaches of sarcasm detection attempt to capture.

Comparison with irony, deception, humor, and metaphor

Sarcasm is related to other forms of figurative language such as irony, deception, metaphor or humor. We now describe the relationship of sarcasm with these concepts:

1. **Sarcasm and irony:** Sarcasm and irony are both forms of figurative language. As discussed earlier, sarcasm has an element of ridicule that is absent in case of irony [Lee and Katz, 1998]. Thus, presence of a target differentiates the two.
2. **Sarcasm and deception:** Both deception and sarcasm do not intend the proposition being literally stated. If a person says '*I love this soup*', they could be speaking the truth (literal proposition), they could be lying (deception) or they could be sarcastic (sarcasm). However, the difference between literal proposition and deception lies in intention of the speaker [Gibbs, 1994] and the shared knowledge between the speaker and the listener [Long and Graesser, 1988]. If the speaker saw a fly floating on the soup and the speaker knows that the listener saw it, the statement above is sarcastic. If the speaker did not know if the listener saw the fly, they could be deceiving the speaker (out of politeness, if the listener had made the soup, for example) by saying '*I love this soup*'.
3. **Sarcasm and humor and metaphor:** Stieger et al. [2011] refer to sarcasm as a form of

aggressive humor. Thus, the peculiarity that makes sarcasm a special case of humor, is the element of mockery or ridicule. All humor may not intend to ridicule.

4. **Relationship between deception, sarcasm, literality and metaphor:** Gibbs [1994] describe the relationship between deception, sarcasm, literality and metaphor using two dimensions:

- **Speaker's intentionality** indicates whether the speaker is truthful or not.
- **Plausibility** of a sentence indicates the likelihood of the proposition in the sentence.

Table 1.2 presents a matrix that compares the four forms of figurative language along the two dimensions above. When the speaker's intentionality is true and the plausibility of the statement high, a statement is literal (For example, '*I ate oats for breakfast*', assuming that the speaker indeed ate oats for breakfast. The assumption makes the plausibility high). When the speaker's intentionality is true but the plausibility is low, a statement is metaphorical (For example, saying '*You are an elephant*' to a human is metaphorical because a human cannot be an elephant, but in this case, is being compared to an elephant.). However, when the speaker's intentionality is false but the statement has high plausibility, a statement qualifies as a deception (For example, a speaker who says '*I think you are a great guy*' without meaning it, is a deception because of the speaker's lack of intentionality). When the intentionality is false and plausibility is low, the statement qualifies as sarcasm. Consider the sentence '*I love the soup*'. In absence of any additional information, it is a plausible situation. But, if the person does not mean it (false intentionality), it is a deception. If the soup tastes bad or has a fly in it, the sentence no longer describes a plausible situation and hence, is a sarcastic statement.

1.6 Incongruity for Sarcasm

Gibbs [1994] 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. Incongruity²⁵ is defined as the state of being incongruous (i.e. lacking in harmony; not in agreement with principles). Ivanko and Pexman [2003] state that sarcasm/irony is understood because of incongruity.

²⁵Source: The Free Dictionary

		Speaker's intentionality	
		True	False
Plausibility	High	Literal <i>'I ate oats for breakfast'</i> . Assumption about intentionality: The speaker indeed ate oats for breakfast	Deception <i>'I think you are a great guy'</i>
	Low	Metaphorical <i>'You are an elephant'</i> . Assumption about plausibility: The sentence was addressed to a human and not an actual elephant	Sarcasm <i>'I love this soup'</i> . Assumption about intentionality: The speaker has not liked the soup. A situation resulting in this dislike for the soup may or may not known to be the listener.

Table 1.2: Relationship between sarcasm, deception, metaphor and irony as given in Gibbs [1994]

We describe incongruity using an example. Assume that Person A makes breakfast for Person B. Three types of incongruity may arise:

1. Person B sees the eggs that Person A made for them, and says *"I love under-cooked eggs for breakfast"*. Based on the information that under-cooked eggs are not desirable, the likelihood of B loving it is low. This incongruity conveys the sarcasm to person A.
2. Person B sees the breakfast, pokes a fork to point out that they are under-cooked and says *"This is exactly what I wanted for breakfast!"*. Person B does not express their negative sentiment through their words, but conveys it through their action.
3. Person B sees the eggs and says, *"This is exactly what I wanted for breakfast"*. This sarcasm is understood only if person A knew that the eggs are under-cooked.

The three situations above and the resulting incongruity are summarized in Table 1.3. In the first situation, the negative sentiment was expressed using sentiment words. In the second situation, the negative sentiment was expressed through an action. In the third situation, the negative sentiment was left to be understood by the listener. In either of the three scenarios, if the eggs

Situation	How is the sarcasm conveyed?	How is the sarcasm understood?
B says, “ <i>I love under-cooked eggs for breakfast</i> ”	Under-cooked eggs for breakfast is an undesirable situation, based on general knowledge.	Incongruity is expressed with the use of sentiment-bearing words. This incongruity can be understood from the text itself.
B says, “ <i>This is exactly what I wanted for breakfast</i> ”	B pokes a fork on the eggs to establish the shared knowledge that the eggs are not cooked well.	The negative sentiment is expressed through an action. This implied sentiment of this action, along with the words, is necessary to understand the incongruity.
B says, “ <i>This is exactly what I wanted for breakfast</i> ”	B does not perform any action to convey that the eggs are not cooked well	The sarcasm can be understood by A only if A has realized that the eggs are under-cooked. This incongruity is the most difficult, among the three, to understand.

Table 1.3: An example scenario where incongruity may occur at different levels of difficulty

were not under-cooked, the incongruity does not hold and hence, the sentences become non-sarcastic. However, if the eggs were under-cooked, the three situations merely differ in how this incongruity was conveyed. 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.

The incorporation of incongruity in order to understand sarcasm is central to this thesis. For example, our sarcasm detection approaches use indicators that capture incongruity by identifying words that are ‘*out-of-place*’ within a sentence. Our sarcasm generation approach emulates incongruity with a set of template-based sarcasm generators. Therefore, the thesis is also organized in chapters based on how the incongruity is handled.

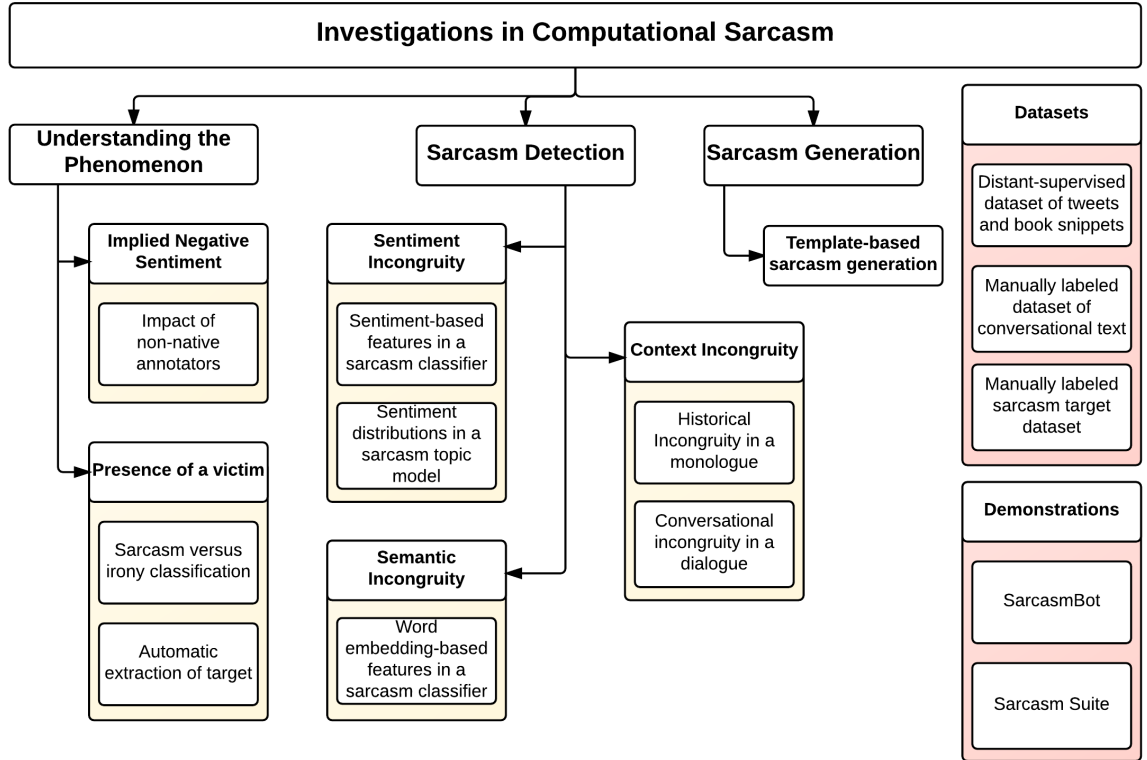


Figure 1.1: A taxonomy illustrating the scope of this thesis

1.7 Contribution

Figure 1.1 shows a taxonomy of the contribution of this thesis. Our investigations in computational sarcasm span three areas: (a) Understanding the phenomenon of sarcasm, (b) Sarcasm detection, and (c) Sarcasm generation. Specific contributions are as follows:

1. Approaches for sarcasm detection based on the notion of incongruity. The incongruity is explored at different levels of difficulty: sentiment incongruity (when sentiment words are clear indicators of incongruity), semantic incongruity (where semantic distance between components of a sentence are indicators of incongruity), etc.
2. To the best of our knowledge, a topic model-based approach to sarcasm detection has not been explored in the past.
3. Introduction of novel research problems apart from sarcasm detection, namely, sarcasm target identification and sarcasm generation. These result in following works:
 - (a) A first-of-its-kind approach to identify the target of sarcasm in a sarcastic sentence.
 - (b) A first-of-its-kind, open source natural language generation system that responds sarcastically to user input.
4. Creation of sarcasm-annotated datasets: (a) transcripts from TV series ‘Friends’ using manual annotation, (b) book snippets using distant supervision, and (c) tweets using distant supervision based on hashtags.
5. Sarcasm annotation study to understand implications of cultural differences on sarcasm annotation and sarcasm classification.

To the best of our knowledge, all of the above contributions are first-of-their-kind. In addition to these contributions, this work also resulted in ‘*Sarcasm Suite*’ which is a deployment of our sarcasm detection and generation modules. Sarcasm Suite was selected for demonstration presentation at Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)²⁶.

The code for projects reported in this thesis, apart from sarcasm generation, is at: <https://github.com/adityajo/ComputationalSarcasm/> The code for sarcasm generation is at: <https://github.com/adityajo/sarcasmbot/>

²⁶The demonstration is at: <http://www.cfilt.iitb.ac.in/sarcasmsuite/>. It uses server-side scripting in PHP, Bootstrap CSS Framework²⁷ for the front end, and jQuery-based AJAX requests that query the modules.

1.8 Thesis Organization

This thesis is organized as follows. Chapter 2 presents a survey of past work in computational sarcasm. Chapter 3 describes our investigations in understanding the phenomenon of sarcasm. We describe two annotation studies: one to understand the impact of non-native annotators on sarcasm annotation, and another to validate the difficulty of sarcasm versus irony classification. Following that, we discuss an introductory computational approach to identify target of sarcasm. Chapters 4 and 5 present our approaches to sarcasm detection. We divide our approaches in two groups: sarcasm detection approaches based on incongruity within the target text (*i.e.*, the text to be classified), and approaches based on context incongruity outside target text. Chapter 4 presents approaches corresponding to the former, while Chapter 5 describes approaches for the latter. Chapter 6 describes our work in sarcasm generation. Chapter 7 concludes the thesis and discusses future work.

1.9 Publications

The publications resulting from this research are:

Related to Computational Sarcasm

1. Aditya Joshi, Pushpak Bhattacharyya, Mark J Carman, ‘Automatic Sarcasm Detection: A Survey’, Vol. 50, No. 5, Article 73, ACM Computing Surveys, 2017.
2. Aditya Joshi, Samarth Agrawal, Pushpak Bhattacharyya, Mark J Carman, ‘Expect the unexpected: Harnessing Sentence Completion for Sarcasm Detection’, PACLING 2017, Yangon, Myanmar, August 2017.
3. Aditya Joshi, Diptesh Kanojia, Pushpak Bhattacharyya, Mark Carman, ‘Sarcasm Suite: A browser-based engine for sarcasm detection and generation’, AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, California, February 2017.
4. Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya and Mark Carman, ‘Are Word Embedding-based Features Useful for Sarcasm Detection?’, Conference on Empirical Methods on Natural Language Processing (EMNLP) 2016, Austin, Texas, November 2016.

5. Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, Mark Carman, Meghna Singh, Jaya Saraswati and Rajita Shukla, ‘How Challenging is Sarcasm versus Irony Classification?: A Study With a Dataset from English Literature’, Workshop of the Australasian Language Technology Association (ALTA) 2016, Melbourne, Australia, December 2016.
6. Aditya Joshi, Prayas Jain, Pushpak Bhattacharyya, Mark Carman, ‘Who would have thought of that!: A Novel Hierarchical Topic Model for Extraction of Sarcasm-prevalent Topics and Sarcasm Detection’, Extra-Propositional Aspects of Meaning (ExProM) in Computational Linguistics Conference (COLING) 2016, Osaka, Japan, December 2016.
7. Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya and Mark Carman, ‘Harnessing Sequence Labeling for Sarcasm Detection in Dialogue from TV Series Friends’, Conference on Natural Language Learning (CONLL) 2016, Berlin, Germany, August 2016.
8. Aditya Joshi, Vinita Sharma, Pushpak Bhattacharyya, ‘Harnessing context incongruity for sarcasm detection’, Conference of the Association for Computational Linguistics (ACL) 2015, Beijing, China, July 2015.
9. Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, Jaya Saraswati and Rajita Shukla, ‘How Do Cultural Differences Impact the Quality of Sarcasm Annotation?: A Case Study of Indian Annotators and American Text’, 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities at Conference of the Association for Computational Linguistics (ACL) 2016, Berlin, Germany, August 2016.
10. Anupam Khattri, Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, ‘Your sentiment precedes you: Using an author’s historical tweets to predict sarcasm’, 6th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA) at Conference on Empirical Methods on Natural Language Processing (EMNLP) 2015, Lisbon, Portugal, September 2015.
11. Aditya Joshi, Anoop Kunchukuttan, Pushpak Bhattacharyya, Mark Carman, ‘Sarcasm-Bot: An open-source sarcasm-generation module for chatbots’, WISDOM at KDD 2015, Sydney, Australia, August 2015.

Other Related Publications:

1. Samarth Agrawal, Aditya Joshi, Joe Cheri Ross, Pushpak Bhattacharyya, Harshawardhan Wabgaonkar, 'Are Word Embedding and Dialogue Act Class-based Features Useful for Coreference Resolution in Dialogue?', PACLING 2017, Yangon, Myanmar, August 2017.
2. Aditya Joshi, Abhijit Mishra, Balamurali AR, Pushpak Bhattacharyya, Mark Carman, 'A computational approach for automatic prediction of drunk-texting', Conference of the Association for Computational Linguistics (ACL) 2015, Beijing, China, July 2015.
3. Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, 'Political Issue Extraction Model: A Novel Hierarchical Topic Model That Uses Tweets By Political And Non-Political Authors', 7th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA) at NAACL 2016, San Diego, USA, June 2016.
4. Aditya Joshi, Vaibhav Tripathi, Ravindra Soni, Pushpak Bhattacharyya, Mark Carman, 'EmoGram: An Open-Source Time Sequence-based Emotion Tracker and its innovative applications', Workshop on Knowledge Extraction from Text (KET) at AAAI Conference on Artificial Intelligence (AAAI) 2016, Phoenix, USA, February 2016.
5. Diptesh Kanojia, Aditya Joshi, Pushpak Bhattacharyya and Mark James Carman, 'That'll do fine!: A coarse lexical resource for English-Hindi MT, using polylingual topic models', LREC 2016, Portoro, Slovenia, May 2016.
6. Aditya Joshi, Abhijit Mishra, Nivvedan Senthamilselvan and Pushpak Bhattacharyya, 'Measuring Sentiment Annotation Complexity of Text', Conference of the Association for Computational Linguistics (ACL) 2014, Baltimore, USA, June 2014.
7. Abhijit Mishra, Aditya Joshi and Pushpak Bhattacharyya, 'A cognitive study of subjectivity extraction in sentiment annotation', 5th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA) at Conference of the Association for Computational Linguistics (ACL) 2014, Baltimore, USA, June 2014.

The contents of the thesis have also been accepted for publication as a monograph as follows:

1. Aditya Joshi, Pushpak Bhattacharyya, Mark J Carman, 'Investigations in Computational Sarcasm', Cognitive Systems Monographs, Springer Nature, Singapore. (Awaiting publication)

arXiv upload:

1. Aditya Joshi, Pushpak Bhattacharyya, Mark Carman, Automatic Sarcasm Detection: A Survey, arXiv preprint arXiv:1602.03426 (2016).
2. Aditya Joshi, Pranav Goel, Pushpak Bhattacharyya, and Mark Carman, ‘Automatic Identification of Sarcasm Target: An Introductory Approach’, arXiv preprint arXiv:1610.07091 (2016).

Chapter 2

Literature Survey

Sarcasm, being a peculiar form of sentiment expression¹, has been a commonly quoted challenge for sentiment analysis, leading to developments in computational sarcasm. Despite this, all past work in computational sarcasm pertains to sarcasm detection. Starting with the earliest known work [Tepperman et al., 2006] which deals with sarcasm detection in speech, the area has seen wide interest from the text analytics community. Sarcasm detection from text has now extended to different data forms and techniques. In this chapter, we survey past work in sarcasm detection²³.

In Table 2.1, we present a matrix that summarizes past work in sarcasm detection across various parameters. These parameters are datasets, approaches, annotation strategies, features and contextual information used. The tables provide a bird's-eye view of past work in sarcasm detection. The remaining chapter describes each of these parameters in detail. The chapter is organized as follows. Section 2.1 presents different problem definitions for sarcasm detection. Sections 2.2, 2.3 and 2.4 describe datasets, approaches and performance values reported, respectively. Section 2.5 highlights trends underlying sarcasm detection, while Section 2.6 discusses recurring issues in sarcasm detection research. Section 2.7 summarizes the chapter.

¹Liebrecht et al. [2013] state that “*detecting sarcasm is like a needle in a haystack*”

²Wallace [2013] give a comprehensive discussion of linguistic challenges of computational irony. The paper focuses on linguistic theories and possible applications of these theories for sarcasm detection. We delve into the computational work.

³A version of this chapter is on arXiv (including the work presented in this thesis) at: <http://www.arxiv.org/abs/1602.03426>. The literature survey was published in ACM Computing Surveys, Vol. 50, Issue 5, Art. 73, 2017.

2.1 Problem Definition

We first look at how the problem of automatic sarcasm detection has been defined in past work. The most common formulation for sarcasm detection is a **classification** task. Given a piece of text, the goal is to predict whether or not it is sarcastic. Past work varies in terms of output labels. For example, trying to understand/characterize the relationship between sarcasm, irony and humor, Barbieri et al. [2014b] consider labels for the classifier as: politics, humor, irony and sarcasm. Reyes et al. [2013] use a similar formulation but learn two-class classifiers for all pairs of these four labels. Apart from labels, there have been variations in the data units that will be annotated. Tepperman et al. [2006] is an early paper in sarcasm detection that looks at occurrences of a common sarcastic phrase ‘*yeah right*’ and classifies each occurrence of ‘*yeah right*’ as sarcastic or not. Veale and Hao [2010] annotate similes such as ‘*as excited as a patient before a surgery*’ with sarcastic or non-sarcastic labels.

Other formulations for sarcasm detection have also been reported. Ghosh et al. [2015b] model sarcasm detection as a sense disambiguation task. They state that a word may have a literal sense and a sarcastic sense. They detect sarcasm by identifying the sense in which a word is used.

A note on languages

Most research in sarcasm detection deals with datasets in English. However, some research in the following languages has also been reported: Chinese [Liu et al., 2014], Italian [Barbieri et al., 2014a], Czech [Ptáček et al., 2014], Dutch [Liebrecht et al., 2013], Greek [Charalampakis et al., 2016], Indonesian [Lunando and Purwarianti, 2013] and Hindi [Desai and Dave, 2016].

2.2 Datasets

This section describes the datasets used to evaluate sarcasm detection. We divide them into three 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) and other datasets. A short, sarcastic text would have a sharp sarcastic remark while a long, sarcastic text may contain either a build-up or unrelated statements. The third category of datasets, ‘other datasets’ have peculiar structures (for example, conversations) and hence, cannot be categorized as either short or long. Table 2.2 lists past works in which each of the three types were used.

	Datasets			Approach			Annotation			Features					Context		
	Short Text	Long Text	Other	Rule-based	Semi-superv.	Superv	Manual	Distant	Other	Unigram	Sentiment	Pragmatic	Patterns	Other	Author	Conversation	Other
Kreuz and Caucchi [2007]			✓				✓			✓							
Tsur et al. [2010]		✓			✓		✓			✓			✓				
Davidov et al. [2010a]		✓			✓		✓			✓			✓				
Veale and Hao [2010]			✓	✓					✓	✓							
González-Ibáñez et al. [2011]	✓					✓		✓		✓	✓	✓					
Reyes et al. [2012]	✓					✓		✓		✓	✓	✓		✓			
Reyes and Rosso [2012]		✓				✓	✓			✓	✓			✓			
Filatova [2012]		✓					✓										
Riloff et al. [2013]	✓				✓		✓			✓		✓	✓				
Lukin and Walker [2013]		✓			✓		✓			✓			✓				
Liebrecht et al. [2013]	✓					✓		✓			✓		✓		✓		
Reyes et al. [2013]	✓					✓		✓		✓	✓	✓		✓			
Reyes and Rosso [2014]	✓	✓				✓		✓		✓	✓	✓		✓			
Rakov and Rosenberg [2013]			✓			✓	✓			✓				✓			
Barbieri et al. [2014b]	✓					✓		✓			✓			✓			
Maynard and Greenwood [2014]	✓			✓			✓			✓	✓			✓			
Wallace et al. [2014]		✓					✓										
Buschmeier et al. [2014]		✓				✓			✓	✓	✓	✓			✓		
Barbieri et al. [2014a]	✓					✓			✓	✓			✓				
Rajadesingan et al. [2015]	✓					✓		✓		✓	✓			✓	✓	✓	
Bamman and Smith [2015]	✓					✓		✓		✓	✓	✓		✓	✓	✓	✓
Wallace et al. [2015]		✓				✓	✓			✓	✓			✓		✓	✓
Ghosh et al. [2015a]	✓			✓	✓	✓		✓		✓		✓					
Hernández-Farías et al. [2015]	✓					✓		✓		✓	✓	✓		✓			
Wang et al. [2015]	✓					✓			✓	✓						✓	
Ghosh et al. [2015b]			✓				✓			✓				✓			
Liu et al. [2014]	✓	✓				✓		✓		✓	✓	✓		✓			
Bharti et al. [2015]	✓				✓			✓		✓	✓		✓				
Fersini et al. [2015]	✓					✓		✓		✓		✓		✓			
Bouazizi and Ohtsuki [2015a]	✓					✓	✓			✓		✓		✓			
Muresan et al. [2016]	✓					✓		✓		✓	✓	✓					
Mishra et al. [2016]	✓	✓	✓	✓			✓			✓	✓	✓		✓			
Abercrombie and Hovy [2016]	✓					✓	✓			✓					✓	✓	✓
Silvio et al. [2016]	✓					✓		✓							✓		
Ghosh and Veale [2016]	✓					✓	✓										
Bouazizi and Ohtsuki [2015b]	✓				✓			✓		✓		✓	✓				
Farías et al. [2016]	✓					✓	✓	✓		✓	✓		✓	✓			

Table 2.1: Summary of sarcasm detection along different parameters

2.2.1 Short text

Social media makes large-scale user-generated text accessible. However, because of restrictions imposed by some of these platforms on text length, this text tends to be short, requiring authors to fit their statements within the specified limit. This results in noise in the form of shortened words, dropped subjects, etc. For example, ‘*I love your shirt...*’ may be shortened to ‘*I luv ur shirt...*’. Also, since social media is accessed real-time, there may be additional noise in the form of mis-spellings, etc. Despite this noise, the short length also necessitates that this text be succinct and hence, sharp in sarcasm. Therefore, datasets of tweets have been popular for sarcasm detection. An important reason for this is also the availability of the Twitter API. For Twitter-based datasets, two approaches to obtain annotations have been used. The first approach is manual annotation. Riloff et al. [2013] introduced a dataset of tweets, manually annotated as sarcastic or not. Maynard and Greenwood [2014] experimented with around 600 tweets which were marked for subjectivity, sentiment and sarcasm. The second technique to create datasets

Text form	Related Work
Tweets	<p>Manual: [Maynard and Greenwood, 2014, Ptáček et al., 2014, Mishra et al., 2016, Abercrombie and Hovy, 2016]</p> <p>Hashtag-based: [Davidov et al., 2010a, González-Ibáñez et al., 2011, Reyes et al., 2012, 2013, Barbieri et al., 2014a, Ghosh et al., 2015a, Bharti et al., 2015, Liebrecht et al., 2013, Bouazizi and Ohtsuki, 2015a, Wang et al., 2015, Barbieri et al., 2014b, Baman and Smith, 2015, Fersini et al., 2015, Rajadesingan et al., 2015, Abercrombie and Hovy, 2016]</p>
Reddits	[Wallace et al., 2014, 2015]
Long text (Reviews, etc.)	[Lukin and Walker, 2013, Reyes and Rosso, 2014, 2012, Buschmeier et al., 2014, Liu et al., 2014, Filatova, 2012]
Other datasets	[Tepperman et al., 2006, Kreuz and Caucci, 2007, Veale and Hao, 2010, Rakov and Rosenberg, 2013, Ghosh et al., 2015b, Abercrombie and Hovy, 2016]

Table 2.2: Summary of sarcasm-labeled datasets

is the use of hashtag-based supervision. Many approaches create datasets using 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:

1. Only the author of a tweet can determine if it was sarcastic. A hashtag is a label provided by authors themselves,
2. The 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. Davidov et al. [2010a] use a dataset of tweets, which are labeled based on sarcasm-indicative hashtags such as #sarcasm, #sarcastic, #not, etc⁴. González-Ibáñez et al. [2011], Reyes et al. [2012] also use hashtag-based supervision for tweets. However, they retain examples where it occurs at the end of a tweet but discard tweets where the hashtag is a part of the running text. For example, the tweet *‘#sarcasm is popular among teens’* is eliminated because it is a tweet about sarcasm and not a sarcastic tweet. Reyes et al. [2013] present a large dataset of 40000 tweets labeled as sarcastic or not, again using hashtags. Ghosh et al. [2015a] divide their hashtag-annotated dataset of tweets into three parts: 1000 trial, 4000 development and 8000 test tweets. Liebrecht et al. [2013] use *‘#not’* to obtain tweets. Barbieri et al. [2014b] create a dataset using hashtag-based supervision based on hashtags indicated by multiple labels: politics, sarcasm, humor and irony. Several other works use hashtag-based supervision in this way to create labeled datasets [Barbieri et al., 2014a, Bharti et al., 2015, Bouazizi and Ohtsuki, 2015a, Abercrombie and Hovy, 2016].

However, datasets of tweets pose several challenges. Twitter API requires you to search for a keyword in a tweet. However, the author of a text uses a hashtag to indicate that they are being sarcastic. So, obtaining sarcastic tweets in this way is easy. However, the collection of non-sarcastic tweets using hashtag-based supervision is difficult. Tweets containing *‘#notsarcastic’* may be downloaded as non-sarcastic but are unlikely to have the desired statistical properties of general non-sarcastic text. In some cases, authors have assumed that tweets not containing the sarcastic hashtag are not sarcastic [Bamman and Smith, 2015]. Hashtag-based supervision may lead to a degradation in the quality of training data for reasons such as incorrect use of a sarcasm-indicative hashtag. To ensure quality, Fersini et al. [2015] obtain the initial label based on the presence of hashtags following which the labels are verified by manual annotation.

⁴An exhaustive list is not possible since topical hashtags may be created. For example, #pappu in Indian tweets are likely to be sarcastic towards a certain politician. The word *‘pappu’* by itself means a young boy.

Twitter also provides access to additional context such as past tweets by the same author. Hence, in order to predict sarcasm, supplementary datasets⁵ have also been used for sarcasm detection. Rajadesingan et al. [2015] use a dataset of tweets, labeled by hashtag-based supervision, along with a historical context of 80 tweets per author.

Like supplementary datasets, supplementary annotation (*i.e.*, annotation apart from sarcasm/ non-sarcasm) has also been explored. Mishra et al. [2016] capture cognitive features based on eye-tracking. They employ annotators who are asked to determine the sentiment of a text. While the annotators read the text, their eye movements are recorded by an eye-tracker. This eye-tracking information serves as supplementary annotation.

Other social media text used for sarcasm-labeled datasets includes posts from Reddits. Wallace et al. [2014] create a corpus of Reddit posts of 10K sentences, from 6 Reddit topics. Wallace et al. [2015] present a dataset of 5625 Reddit comments.

2.2.2 Long text (Passages)

Reviews and discussion forum posts have also been used for creation of sarcasm-labeled datasets. Lukin and Walker [2013] present Internet Argument Corpus that marks a dataset of discussion forum posts with multiple labels, one of them being sarcasm. Reyes and Rosso [2014] create a dataset of movie reviews, book reviews and news articles marked with sarcasm and sentiment. Reyes and Rosso [2012] deal with products that receive substantial negative reviews. Their dataset consists of 11,000 reviews. Filatova [2012] use a sarcasm-labeled dataset of around 1,000 reviews. Buschmeier et al. [2014] create a labeled set of 1,254 Amazon reviews, out of which 437 are ironic. Tsur et al. [2010] consider a large dataset of 66,000 Amazon reviews. Liu et al. [2014] use a dataset of reviews from multiple sources such as Amazon, Twitter, Netease and Netcena. In case of long text, the datasets are manually annotated when markers like hashtags are not available.

2.2.3 Other datasets (Dialogues, syntactic patterns, etc.)

Tepperman et al. [2006] use 131 call center transcripts where each occurrence of ‘*yeah right*’ is marked as sarcastic or not. The goal of their work then is to identify which ‘*yeah right*’ is

⁵‘*Supplementary*’ datasets refer to text that does not need to be annotated but that will contribute to the judgment of the sarcasm detector

sarcastic. Kreuz and Caucchi [2007] use 20 sarcastic, and 15 non-sarcastic excerpts, which are marked by 101 annotators. Veale and Hao [2010] search the web for the pattern ‘* as a *’, and create a dataset of 20,000 distinct similes which are labeled as sarcastic or not. Rakov and Rosenberg [2013] create a crowdsourced dataset of sentences from an MTV show, Daria.

Ghosh et al. [2015b] 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*’. Mishra et al. [2016] create a manually labeled dataset of quotes from a website called sarcasmsociety.com.

2.3 Approaches

In this section, we describe approaches for sarcasm detection. In general, approaches to sarcasm detection can be classified into: rule-based, statistical and deep learning-based approaches. We also describe shared tasks on sarcasm detection from different conferences.

2.3.1 Rule-based Approaches

Rule-based approaches attempt to identify sarcasm through rules based on indicators of sarcasm. Veale and Hao [2010] identify sarcasm in similes using Google searches, in order to determine how likely a simile is. They present a 9-step approach where at each step, a simile is predicted to be sarcastic using heuristic-based checks (such as lexical similarity between the two components of the simile. For example, the similarity between ‘useful’ and ‘chocolate teapot’ is used to determine if ‘as useful as a chocolate teapot’ is a sarcastic simile) and the number of search results (such as comparing the number of search results returned by ‘*as useful as a chocolate teapot*’ and ‘*about as useful as a chocolate teapot*’). Maynard and Greenwood [2014] propose that hashtag sentiment is a key indicator of sarcasm. Hence, in their case, if the sentiment expressed by a hashtag does not agree with the remaining tweet, the tweet is predicted as sarcastic. They use a hashtag tokenizer to split hashtags made of concatenated words. Bharti et al. [2015] present two rule-based classifiers. The first uses a parser-based lexicon generation algorithm that creates parse trees of sentences and identifies situation phrases that bear sentiment. If a negative phrase occurs in a positive sentence, it is predicted as sarcastic. The second algorithm aims to capture hyperbolic sarcasm (*i.e.*, sarcasm involving exaggeration or hyperbole) by using interjections (such as ‘*wow*’) and intensifiers (such as ‘*very*’) that occur

together. Riloff et al. [2013] 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 the situation phrase.

2.3.2 Statistical Approaches

Statistical approaches to sarcasm detection vary in terms of features and learning algorithms. We look at the two in forthcoming subsections.

Features

In this subsection, we investigate the set of features that have been reported for statistical sarcasm detection. Most approaches use bag-of-words as features. However, in addition to these, several other sets of features have been reported. Table 2.3 summarizes these features. In this subsection, we focus on features related to the text to be classified. Contextual features (*i.e.*, features that use information beyond the text to be classified) are described in a latter subsection.

Tsur et al. [2010] design pattern-based features that indicate the presence of discriminative patterns as extracted from a large sarcasm-labeled corpus. To prevent overfitting of patterns, these pattern-based features take real values based on three situations: exact match, partial overlap and no match. González-Ibáñez et al. [2011] use sentiment lexicon-based features and pragmatic features like emoticons and user mentions. Similarly, Farías et al. [2016] use features derived from multiple affective lexicons such as AFINN, SentiWordnet, General Inquirer, etc. In addition, they also use features based on semantic similarity, emoticons, counter-factuality, etc. Reyes et al. [2012] introduce features related to ambiguity, unexpectedness, emotional scenario, etc. Ambiguity features cover structural, morpho-syntactic, semantic ambiguity, while unexpectedness features measure semantic relatedness. Riloff et al. [2013] use a set of patterns, specifically positive verbs and negative situation phrases, as features for a classifier (in addition to a rule-based classifier). Liebrecht et al. [2013] introduce bigrams and trigrams as features. Reyes et al. [2013] explore skip-gram and character n-gram-based features. Maynard and Greenwood [2014] include seven sets of features such as maximum/minimum/gap of inten-

sity of adjectives and adverbs, max/min/average number of synonyms and synsets for words in the target text, etc. Apart from a subset of these, Barbieri et al. [2014a] use frequency and rarity of words as indicators. Buschmeier et al. [2014] incorporate ellipsis, hyperbole and imbalance in their set of features.

Rajadesingan et al. [2015] use features based on extensions of words, number of flips, readability in addition to contextual features. Hernández-Farías et al. [2015] present features that measure semantic relatedness between words using Wordnet-based similarity. Liu et al. [2014] introduce POS sequences and semantic imbalance as features. Since they also experiment with Chinese datasets, they use language-typical features like use of homophony, use of honorifics, etc. Mishra et al. [2016] design a set of gaze-based features such as average fixation duration, regression count, skip count, etc. based on annotations from their eye-tracking experiments. In addition, they also use complex gaze-based features based on saliency graphs which connect words in a sentence with edges representing saccade between the words. In general, emoticons, punctuation marks, unigrams, or phrase-based features are common for sarcasm detection.

Learning Algorithms

Most work in sarcasm detection relies on different forms of SVM [Tepperman et al., 2006, Kreuz and Caucci, 2007, Tsur et al., 2010, Davidov et al., 2010a, Reyes and Rosso, 2012]. González-Ibáñez et al. [2011] use SVM with SMO and logistic regression. Chi-squared test is used to identify discriminating features. Riloff et al. [2013] compare rule-based techniques with a SVM-based classifier. Liebrecht et al. [2013] use balanced winnow algorithm in order to determine high-ranking features. Reyes et al. [2013] use Naïve Bayes and decision trees. Bamman and Smith [2015] use binary logistic regression. Wang et al. [2015] use SVM-HMM in order to incorporate sequence nature of output labels in a conversation. Liu et al. [2014] compare several ensemble-based classification approaches such as bagging and boosting. In general, SVM-based classification is popular for sarcasm detection.

2.3.3 Deep Learning-based Approaches

As architectures based on deep learning techniques gain popularity, such approaches have been reported for automatic sarcasm detection as well. Silvio et al. [2016] present a novel convolu-

tional network-based architecture that learns user embeddings in addition to utterance-based embeddings. Ghosh and Veale [2016] use a combination of convolutional neural network, LSTM followed by a DNN. They compare their approach against recursive SVM, and show an improvement in case of deep learning architecture.

	Salient Features
[Tsur et al., 2010]	Sarcastic patterns, Punctuations
[González-Ibáñez et al., 2011]	User mentions, emoticons, unigrams, sentiment-lexicon-based features
[Reyes et al., 2012]	Ambiguity-based, semantic relatedness
[Reyes and Rosso, 2012]	N-grams, POS N-grams
[Riloff et al., 2013]	Sarcastic patterns (Positive verbs, negative phrases)
[Liebrecht et al., 2013]	N-grams, emotion marks, intensifiers
[Reyes et al., 2013]	Skip-grams, Polarity skip-grams
[Barbieri et al., 2014b]	Synonyms, Ambiguity, Written-spoken gap
[Buschmeier et al., 2014]	Interjection, ellipsis, hyperbole, imbalance-based
[Barbieri et al., 2014a]	Freq. of rarest words, max/min/avg # synsets, max/min/avg # synonyms
[Rajadesingan et al., 2015]	Readability, sentiment flips, etc.
[Hernández-Farías et al., 2015]	Length, capitalization, semantic similarity
[Liu et al., 2014]	POS sequences, Semantic imbalance. Chinese-specific features such as homophones, use of honorifics
[Ptáček et al., 2014]	Word shape, Pointedness, etc.
[Mishra et al., 2016]	Cognitive features derived from eye-tracking experiments
[Bouazizi and Ohtsuki, 2015b]	Pattern-based features along with word-based, syntactic, punctuation-based and sentiment-related features
[Farías et al., 2016]	Affect-based features derived from multiple emotion lexicons

Table 2.3: Summary of features used for statistical sarcasm classifiers

2.3.4 Shared Tasks & Benchmark Datasets

Shared tasks allow comparative evaluation of multiple approaches on a common dataset. Ghosh et al. [2015a] describe a shared task from SemEval-2015 that deals with sentiment analysis of figurative language. The organizers provided a dataset of ironic and metaphorical statements labeled as positive, negative and neutral. The submitted systems were expected to correctly identify the sentiment polarity in case of figurative expressions like irony. The teams that participated in the shared task used affective resources, character n-grams, etc. The winning team used “*four lexica, one that was automatically generated and three that were manually crafted. (sic)*”. The second shared task was a data science contest organized as a part of PAKDD 2016 ⁶. The dataset consisted of Reddit comments labeled as either sarcastic or non-sarcastic.

2.4 Reported Performance

Table 2.4 summarizes reported values from past works. The values may not be directly comparable because they are based on incomparable datasets, techniques or metrics. However, the table does provide a ballpark estimate of current performance of sarcasm detection. González-Ibáñez et al. [2011] show that unigram-based features outperform the use of a subset of words as derived from a sentiment lexicon. They compare the accuracy of the sarcasm classifier with the human ability to detect sarcasm. Reyes and Rosso [2012] report sentiment-based features as their top discriminating features. The Logistic Regression classifier in Rakov and Rosenberg [2013] results in an accuracy of 81.5%. Rajadesingan et al. [2015] show that historical features along with sentiment flip-based features are the most discriminating features, and result in an accuracy of 83.46%. Farías et al. [2016] compare their features against several past reported works and show that their approach outperforms them with a F-score of 0.82.

2.5 Trends in Sarcasm Detection

In the previous sections, we looked at the datasets, approaches and performances of past work in sarcasm detection. In this section, we detail interesting trends observed in sarcasm detection research. These trends are represented in Figure 2.1. As seen in the figure, there have

⁶<http://www.parrotanalytics.com/pacific-asia-knowledge-discovery-and-data-mining-conference-2016-contest/>

	Details	Reported Performance
[Tepperman et al., 2006]	Conversation transcripts	F: 0.70, Acc: 87
[Davidov et al., 2010a]	Tweets	F: 0.545 Acc: 89.6
[González-Ibáñez et al., 2011]	Tweets	A: 75.89
[Reyes et al., 2012]	Irony vs general	A: 70.12, F: 0.65
[Reyes and Rosso, 2012]	Reviews	F: 0.891, P: 88.3, R: 89.9
[Riloff et al., 2013]	Tweets	F: 0.51, P: 44, R: 62
[Lukin and Walker, 2013]	Discussion forum posts	F: 0.69, P: 75, R: 62
[Liebrecht et al., 2013]	Tweets	AUC: 0.76
[Reyes et al., 2013]	Irony vs humor	F: 0.76
[Silvio et al., 2016]	Tweets	Acc: 87.2
[Rakov and Rosenberg, 2013]	Speech data	Acc: 81.57
[Muresan et al., 2016]	Reviews	F: 0.757
[Rajadesingan et al., 2015]	Tweets	Acc: 83.46, AUC: 0.83
[Bamman and Smith, 2015]	Tweets	Acc: 85.1
[Ghosh et al., 2015a]	Tweets	Cosine: 0.758, MSE: 2.117
[Fersini et al., 2015]	Tweets	F: 0.8359, Acc: 94.17
[Wang et al., 2015]	Tweets	Macro-F: 0.6913
[Abercrombie and Hovy, 2016]	Tweets	AUC: 0.6
[Buschmeier et al., 2014]	Reviews	F: 0.713
[Hernández-Farías et al., 2015]	Irony vs politics	F: 0.81
[Farías et al., 2016]	Tweets	F: 0.82

Table 2.4: Summary of reported performance values; Precision/Recall/F-measures and Accuracy values are indicated in percentages

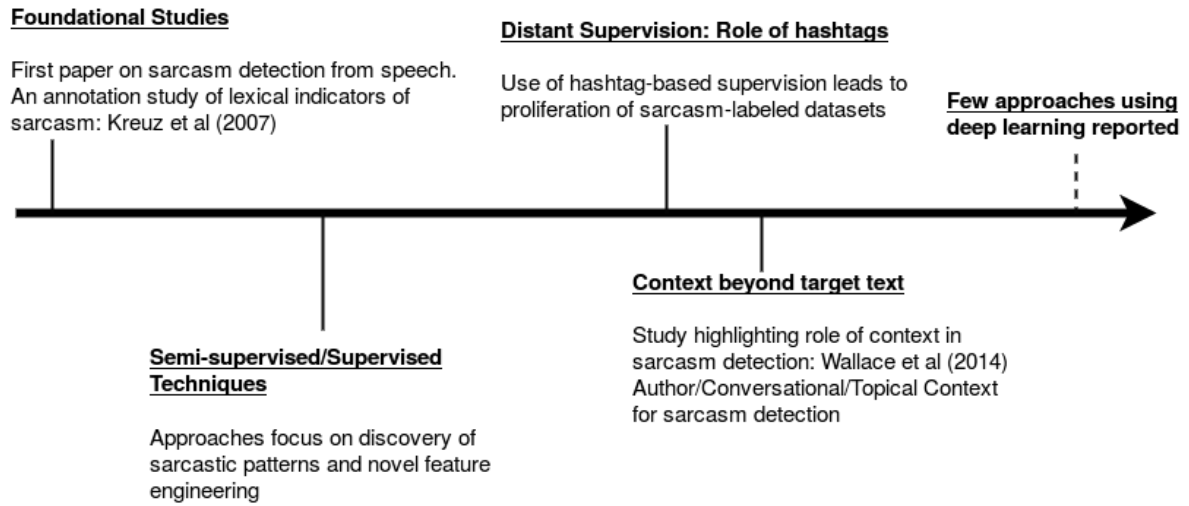


Figure 2.1: Trends in Sarcasm Detection Research

been around four key milestones. Following fundamental studies, supervised/semi-supervised sarcasm classification approaches were explored. These approaches focused on using specific patterns or novel features. Then, as Twitter emerged as a viable source of data, hashtag-based supervision became popular. Recently, there is an emerging trend to use context beyond the text being classified.

In the rest of this section, we describe in detail two of these trends: (a) the discovery of sarcastic patterns, and use of these patterns as features, and (b) the use of contextual information *i.e.*, information beyond the target text for sarcasm detection.

2.5.1 Pattern discovery

Since sarcasm may contain implicit sentiment phrases (for example ‘*I love being awake with a tooth ache at 4am*’), discovering sarcastic patterns was an early trend in sarcasm detection. Several approaches dealt with extracting patterns that are indicative of sarcasm, or carry implied sentiment. These patterns may then be used as features for a statistical classifier, or as a knowledge base for a rule-based classifier. Tsur et al. [2010] extract sarcastic patterns from a seed set of labeled sentences. They first select words that either occur more frequently than an upper threshold or less frequently than a lower threshold. They identify a large set of candidate patterns from among these extracted words. The patterns which occur prominently in either class are then selected. Ptáček et al. [2014] and Bouazizi and Ohtsuki [2015b] use a similar approach for Czech and English tweets respectively.

Lukin and Walker [2013] begin with a seed set of nastiness and sarcasm patterns, created using Amazon Mechanical Turk. They train a high precision sarcastic post classifier, followed by a high precision non-sarcastic post classifier. These classifiers are then used to generate a large labeled dataset. This automatically labeled dataset is then used to train the final sarcasm classifier.

2.5.2 Role of Context in Sarcasm Detection

An emerging trend in sarcasm detection is the use of context. The term context here refers to any information beyond the text to be predicted. 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. In this section, we describe approaches that use contextual information to detect sarcasm. We refer to the textual unit to be classified as the ‘*target text*’. As we will see, this context may be incorporated in a variety of ways - in general, using either supplementary data or supplementary information from the source platform providing the data. Wallace et al. [2014] is the first annotation study that highlighted the need for context in sarcasm detection. The annotators mark Reddit comments with sarcasm labels. During this annotation, annotators often request for additional context in the form of Reddit comments in the thread. The authors also present a transition matrix that shows how many times authors change their labels after this conversational context is displayed to them.

Following this observation regarding the promise of context for sarcasm detection, several recent approaches have looked at ways of incorporating it. The contexts that have been reported are of three types:

1. **Author-specific context:** This type of context refers to textual footprint of the author of the target text. Rajadesingan et al. [2015] incorporate context using the author’s past tweets. The features deal with various dimensions. They use features about author’s familiarity with twitter (in terms of use of hashtags), familiarity with language (in terms of words and structures), and familiarity with sarcasm. Bamman and Smith [2015] consider the author context in features such as historical salient terms, historical topic, profile information, historical sentiment (how likely is he/she to be negative), etc. Silvio et al. [2016] capture author-specific embeddings using a neural network based architecture.

The author-specific embeddings are extracted based on 1000 past tweets posted by the author.

2. **Conversational context:** This type of context refers to text in the conversation of which the target text is a part. Bamman and Smith [2015] capture conversational context using pair-wise similarity features between the target tweet and the previous tweet in the conversation. In addition, they also use audience-based features *i.e.*, the features of the tweet author who responded to the target tweet. Wallace et al. [2015] look at comments in the thread structure to obtain context for sarcasm detection. To do so, they use the subreddit name, and noun phrases from the thread to which the target post belongs. Wang et al. [2015] 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.
3. **Topical context:** This context follows the intuition that some topics are likely to evoke sarcasm more commonly than others. Wang et al. [2015] use topical context. To predict sarcasm in a tweet, they download tweets containing a hashtag in the tweet. Then, based on timestamps, they create a sequence of these tweets and use sequence labeling to detect sarcasm in the target tweet (the last in the sequence).

2.6 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 annotated data. The second issue deals with using sentiment as a feature for classification. Finally, the third issue lies in the context of handling unbalanced datasets.

2.6.1 Issues with Annotation

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 the hashtag #not to indicate insincere sentiment. Liebrecht et al. [2013] show that #not can be used to express sarcasm - while the rest of the sentence is not sufficient for identifying 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 author’s context is incorporated. Thus, hashtag-based supervision may be ambiguous in some cases. To mitigate this problem, a new trend is to validate on multiple datasets - some annotated manually while others annotated through hashtags [Ghosh and Veale, 2016, Bouazizi and Ohtsuki, 2015b]. Ghosh and Veale [2016] train their deep learning-based model using a large dataset of hashtag-annotated tweets, but use a test set of manually annotated tweets.

Even in case of manually annotated datasets, the quality of annotation is a concern. Since sarcasm is a subjective phenomenon, the inter-annotator agreement values reported in past work are diverse. Tsur et al. [2010] indicate an agreement of 0.34 while the value in case of Fersini et al. [2015] is 0.79 and 0.81 for Riloff et al. [2013].

2.6.2 Issues with Sentiment as a Feature

Sarcasm is considered a challenge for sentiment analysis. Since sarcastic sentences may not be correctly classified, many papers deliberate if sentiment can be used as a feature for sarcasm detection. Several approaches use predicted sentiment as a feature in the sarcasm classifier. It must, however, be noted that these approaches rely on ‘*surface polarity*’ - the apparent polarity of a sentence. Bharti et al. [2015] describe a rule-based approach that predicts a sentence as sarcastic if a negative phrase occurs in a positive sentence. In a statistical classifier, surface polarity may be used as a feature [Reyes et al., 2012, Rajadesingan et al., 2015, Bamman and Smith, 2015]. Reyes et al. [2013] capture polarity in terms of two emotion dimensions: activation and pleasantness. Buschmeier et al. [2014] use a sentiment imbalance feature that is represented by star rating of a review disagreeing with the surface polarity. Bouazizi and Ohtsuki [2015a] cascade sarcasm detection and sentiment detection, and observe an improvement of 4% in accuracy for sentiment classification when the sentiment detection is aware of the sarcastic nature of text.

2.6.3 Dealing with Dataset Skews

Sarcasm is an infrequent phenomenon in sentiment expression. This skew also reflects in imbalanced datasets. Tsur et al. [2010] use a dataset with a small set of sentences are marked as

sarcastic. Only 12.5% of tweets in the Italian dataset given by Barbieri et al. [2014a] are sarcastic. On the other hand, Rakov and Rosenberg [2013] present a balanced dataset of 15k tweets. In some papers, specialized techniques are used to deal with imbalanced data. For example, Liu et al. [2014] present a multi-strategy ensemble learning approach. Similarly, in order to deal with sparse features and imbalanced data, Wallace et al. [2015] introduce an LSS-regularization strategy. They use a sparsifying L1 regularizer over contextual features and L2-norm for bag-of-word features. 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 on skewed data, Liebrecht et al. [2013] report AUC for balanced as well as skewed datasets, when demonstrating the benefit of their approach. Another methodology to ascertain the benefit of a given approach despite the imbalanced data is by Abercrombie and Hovy [2016]. They compare performance of sarcasm classification for many datasets of different data imbalances.

2.7 Summary

This chapter presents a literature survey of research in computational sarcasm, specifically, automatic sarcasm detection. We observed three milestones in the history of sarcasm detection research: semi-supervised pattern extraction to identify implicit sentiment, use of hashtag-based supervision to create large-scale annotated datasets, and use of context beyond target text. We summarize datasets, approaches and performance values that have been reported. We described that rule-based approaches capture evidences of sarcasm in the form of rules such as sentiment of hashtag not matching sentiment of rest of the tweet. Statistical approaches use features like sentiment changes, unigrams, bigrams, and discriminative phrases. To incorporate context, additional features specific to the author, the conversation and the topic have been explored in the past. We also highlight three issues in sarcasm detection: quality of sarcasm annotation through manual or distant supervised datasets, the relationship between sarcasm and sentiment as a feature, and data skew in case of sarcasm-labeled datasets. The table that compares all past papers along dimensions such as approach, annotation approach, features, etc. allows us to understand the current state-of-art in sarcasm detection research.

To the best of our knowledge, ours is the first work which keeps a linguistic notion in focus and builds a set of approaches that tackle this notion from different perspectives. The linguistic notion in our case is incongruity, and the rest of the thesis describes how we harness incongruity

in different ways for different problems in computational sarcasm.

Chapter 3

Understanding the Phenomenon of Sarcasm

In the thesis so far, we have presented past work related to sarcasm in linguistics and computational linguistics. This chapter presents our exploration in understanding the phenomenon of sarcasm. We focus on three studies directed towards understanding the phenomenon of sarcasm. These studies deal with two components of sarcasm: implied negative sentiment and presence of a target. The first study focuses on how sarcasm is understood by an annotator, in context of their cultural background. The second study deals with influence of sarcasm target on sarcasm understanding. We consider a specific case of sarcasm-versus-irony classification because a target is absent in irony unlike sarcasm. The third study explores sarcasm target a step further by devising a computational approach to identify sarcasm targets of a piece of sarcastic text.

The rest of the chapter is organized as follows. Section 3.1 describes the impact of cultural backgrounds of annotators on sarcasm annotation and sarcasm classification, while Section 3.2 presents an annotation study on sarcasm-versus-irony classification. Finally, Section 3.3 describes an approach for sarcasm target identification.

3.1 Impact on Cross-Cultural Annotation

To understand the phenomenon of sarcasm, we first focus on one aspect of sarcasm: implied negative sentiment. To study how implied negative sentiment in sarcasm is understood by humans, we investigate a situation in which human annotators may differ in terms of their understanding of this implied negative sentiment.

This situation is an example of cross-cultural dependencies where the reader being from a different cultural background to the speaker may result in sarcasm not being understood. Linguistic studies concerning cross-cultural dependencies of sarcasm have been reported [Boers, 2003, Thomas, Tannen, 1984, Rockwell and Theriot, 2001, Bouton, 1988, Haiman, 1998, Dress et al., 2008]. We look at cross-cultural annotation of sarcasm in text, and its impact on supervised sarcasm classification. The value of our study also lies in understanding the quality of sarcasm-annotated datasets when non-native annotators are employed to create them.

We consider the case of annotators of Indian origin who label datasets consisting of discussion forums/tweets from the US. These datasets were earlier annotated by American annotators. It may be argued that since crowd-sourcing is often used for sourcing annotations, a large pool of annotators makes up for cultural differences amongst the annotators. However, a fundamental study like ours that performs a micro-analysis of culture combinations is likely to be useful for a variety of reasons such as judging the quality of new datasets, or deciding among annotators. Balancing the linguistic and computational perspectives, we present our findings in two ways: (a) degradation in quality of sarcasm annotation by non-native annotators, and (b) impact of this quality on sarcasm classification.

The work described in this section was presented at the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities at ACL 2016¹.

3.1.1 Motivation

When annotators are hired, factors such as language competence are considered. However, while tasks like sense annotation or part-of-speech labeling require linguistic expertise, sarcasm annotation extends beyond linguistic expertise, and often involves cultural context. Tannen [1984] describes how a guest thanking the host for a meal may be perceived as polite in some cultures, but sarcastic in some others. In the latter case, the idea is that if a host has invited a guest, it is expected that they will be hospitable towards the guest. If the guest thanks the host, it is perceived that the guest is subtly hinting at inadequacies/concerns with the hospitality.

Due to the popularity of crowd-sourcing, the cultural background of annotators in published papers is often not known. Keeping these constraints in mind, a study of non-native annotation, and its effect on the corresponding NLP task assumes importance. Similar studies have been reported for related tasks. Hupont et al. [2014] deal with result of cultural differences

¹<http://aclweb.org/anthology/W/W16/W16-2111.pdf>

on annotation of images with emotions. Das and Bandyopadhyay [2011] describe multi-cultural observations during creation of an emotion lexicon. For example, they state that the word ‘blue’ may be associated with sadness in some cultures but with evil in others. Wiebe et al. [1999] show how some annotators may have individual biases towards a certain subjective label, and devise a method to obtain bias-corrected tags. Cohn and Specia [2013] consider annotator biases for the task of assigning quality scores to machine translation output.

3.1.2 Experiment Description

In this section, we describe our annotation experiments in terms of the datasets, the annotators and the experiment methodology.

Datasets

We use two sarcasm-labeled datasets that have been reported in past work. The first dataset called **Tweet-A** was introduced by Riloff et al. [2013], and consists of 2,278 manually labeled tweets, out of which 506 are sarcastic. We refer to these original annotations as **American1**. The second dataset is **Discussion-A** and was introduced by Walker et al. [2012]. It consists of 5,854 discussion forum posts, out of which 742 are sarcastic. This dataset was created using Amazon Mechanical Turk, with IP addresses of Turk workers limited to USA during the experiment². We call these annotations **American2**.

Our Annotators

Our annotators are two female professional linguists of Indian origin with more than 8,000 hours of experience each in annotating English documents for tasks such as sentiment analysis, word sense disambiguation, etc.³ Both have similar hours of experience. They are both 50+ years old and follow only international news that would expose them to American culture. We refer to these annotators as Indian1 and Indian2. The choice of ‘Indian’ annotators was made bearing in mind the difference between American and Indian cultures, and our access to Indian annotators.

²We acknowledge the possibility that some of these annotators were not physically located within USA, despite IP, due to VPN or similar infrastructure. We also note that workers may not have had English as their first language.

³This description highlights that they have strong linguistic expertise.

Example	Remarks
1 I have the worlds best neighbors!	The annotators were not sure if this was intended to be sarcastic. Additional context would have been helpful.
2 @twitter_handle West Ham with Carlton Cole and Carroll up front. Going to be some free flowing football this season then	Annotators were not familiar with these players. Hence, they were unable to determine the underlying sentiment.
3 And, I'm sure that Terri Schiavo was fully aware of all that Bush and radical right-wing religionists did for her and appreciates what they did.	Indian annotators did not know about Terri Schiavo (A woman in an 'irreversible persistent vegetative state' whose 'right to die' was questioned) ⁴ , and had to look up her story on the internet.
4 Love going to work and being sent home after two hours	The Indian annotators were unaware of the context of the long commute and hence, could not detect the sarcasm.
5 @twitter_handle Suns out and I'm working,#yay	The annotators were not sure if a sunny day is pleasant - considering temperatures in India.
6 'So how are those gun free zones working out for you?'	With inadequate knowledge about gun free zones, the annotators were doubtful about sarcasm in the target sentence.

Table 3.1: Examples of sentences that the Indian annotators found difficult to annotate; 'twitter_handle' are twitter handles suppressed for anonymity

Experiments

The annotation experiment is conducted as follows. Our annotators read a unit of text and determine whether it is sarcastic or not. The experiment is conducted in sessions of 50 textual units, and the annotators can pause at any point throughout a session. This results in two datasets where each textual unit has three annotations as follows: (A) Tweet-A has three annotations: American1, Indian1, Indian2, (B) Discussion-A has three annotations: American2, Indian1, Indian2. As stated earlier, the American annotations are taken from past work.

3.1.3 Analysis

We now analyze these datasets from three perspectives: (a) the difficulties during creation and the impact on quality, (b) the degradation in annotation quality, (c) the impact of quality degra-

Annotator Pair	κ	Agreement (%)
Avg. American1	0.81	-
Indian1 & Indian2	0.686	85.82
Indian1 & American1	0.524	80.05
Indian2 & American1	0.508	79.98

Table 3.2: Inter-annotator agreement statistics for Tweet-A; Avg. American1 is as reported in the original paper

dation on sarcasm classification, and (d) the prediction of disagreement between native and non-native annotators.

What difficulties do our Indian annotators face?

Table 3.1 shows examples on which our Indian annotators faced difficulty in determining the sarcasm label. We describe experiences from the experiments in two parts:

1. **Situations in which they were unsure of the label:** These include sources of confusion for our annotators, but may or may not have led to incorrect labels.

Data bias: There are more non-sarcastic texts in the dataset than sarcastic ones. Despite that, the annotators experienced suspicion about every sentence that they had to mark as sarcastic or non-sarcastic. This resulted in confusion as in the case of example 1 in Table 3.1.

Unfamiliar words: The annotators consult a dictionary for jargon like ‘abiogenesis’ or unfamiliar words like ‘happenstance’. For urban slang, they look up the urban dictionary website⁵. Hashtags and emoticons are key clues that the annotators use to detect sarcasm. For example, ‘*No, my roommate play (sic) out of tune Zeppelin songs right outside my door isnt annoying. Not at all #sigh*’. They also state that they could understand the role of these hashtags after a few occurrences.

2. **Situations in which their labels did not match their American counterparts:**

Unknown context about named entities Consider examples 2 and 3 in Table 3.1. In some cases, annotators were unfamiliar with popular figures in sports and politics, and also their associated sentiment.

⁵<http://www.urbandictionary.com/>

Unknown context about situations: Example 4 is a case of Indian annotators marking a text as non-sarcastic, while their American counterparts marked it as sarcastic.

Unclear understanding of socio-political situations: The tweet in example 5 was labeled as non-sarcastic by Indian annotators. Similarly, example 6 appears to be a non-sarcastic question. However, based on their perception about gun shooting incidents in USA, they were unsure if this statement was indeed non-sarcastic.

Annotator Pair	κ	Agreement (%)
Indian1 & Indian2	0.700	92.58
Indian1 & American2	0.569	89.81
Indian2 & American2	0.288	83.33

Table 3.3: Inter-annotator agreement statistics for Discussion-A

How do cross-cultural difficulties affect the quality of annotation?

We now compare the quality of non-native annotation using inter-annotator agreement metrics. Table 3.2 shows statistics for Tweet-A dataset. The Kappa coefficient⁶ as reported in the original paper is 0.81. The corresponding value between Indian1 and Indian2 is 0.686. The values for Discussion-A are shown in Table 3.3. For Discussion-A, the Kappa coefficient between the two Indian annotators is 0.700, while that between Indian1/2 and American annotators is 0.569 and 0.288 respectively. The average values for American annotators are not available

⁶https://en.wikipedia.org/wiki/Cohen's_kappa

Annotator Pair	Sarcastic	Non-sarc
Tweet-A		
Indian1 & American1	84.78	77.71
Indian2 & American1	79.24	80.24
Discussion-A		
Indian1 & American2	67.24	93
Indian2 & American2	40.91	89.5

Table 3.4: Class-wise agreement (%) for pairs of annotators, for both datasets

Training Label Source	Test Label Source	Accuracy (%)	Precision (%)	Recall (%)	F-Score (%)	AUC
Tweet-A						
American	American	80.5	71.5	69.2	70.27	0.816
Indian	American	74.14	65.78	68.61	65.28	0.771
Discussion-A						
American	American	83.9	61.5	59.05	59.97	0.734
Indian	American	79.42	58.28	56.77	56.36	0.669

Table 3.5: Impact of non-native annotation on sarcasm classification

in the original paper, and hence not mentioned. This shows that the inter-annotator agreement between our annotators is higher than their individual agreement with the American annotators. The Kappa values are lower in the case of tweets than discussion forum posts.

The agreement (%) indicates the percentage overlap between a pair of labels. This agreement is high between Indian annotators in case of Tweet-A (85.82%), and Discussion-A (92.58%), and comparable with American annotators.

Table 3.4 shows the percentage agreement separately for the two classes, with American labels as reference labels. In case of Tweet-A, our annotators agree more with American annotators on sarcastic than non-sarcastic tweets. This means that in the case of short text such as tweets, it is the non-sarcastic tweets that cause disagreement. This highlights the fuzziness of sarcastic expressions. On the contrary, in case of long text such as discussion forum posts, sarcastic tweets cause disagreement for our annotators because sarcasm may be in a short portion of a long discussion forum post.

How do these difficulties affect sarcasm classification?

We now evaluate if difficulties in sarcasm annotation have an impact on sarcasm classification. To do so, we use LibSVM by Chang and Lin [2011b] with a linear kernel to train a sarcasm classifier that predicts if a given text is sarcastic or not. We use unigrams as features, and report five-fold cross-validation. Table 3.5 shows various performance metrics for Discussion-A and Tweet-A, namely, Accuracy, Precision, Recall, F-score and Area Under Curve (AUC). These values are averaged across the Indian annotators, for the respective configuration of training

Dataset	Accuracy (%)	AUC
Tweet-A	67.10	0.56
Discussion-A	75.71	0.59

Table 3.6: Predicting annotator agreement using textual features; Values are averaged over Indian annotators

labels⁷. For Tweet-A, using the dataset annotated by American annotators as training labels, leads to an AUC of 0.816. When labels given by the Indian annotators are used to train the classifier, the corresponding value is 0.771. Similar trends are observed in case of other metrics, and also for Discussion-A. However, degradations for both Tweet-A and Discussion-A are not statistically significant for the 95% confidence interval. Thus, although our Indian annotators face difficulties during annotation resulting in partial agreement in labels, it seems that annotations from these annotators did not lead to significant degradation to what the sarcasm annotation will eventually be used for, *i.e.*, sarcasm classification. The two-tailed p-values for Tweet-A and Discussion-A are 0.221 and 0.480 respectively.

Can disagreements be predicted?

We now explore if we can predict, solely using properties of text, whether our Indian annotators will disagree with their American counterparts. This goal is helpful so as to choose between annotators for a given piece of text. For example, if it can be known beforehand that a text is likely to result in a disagreement between native and non-native annotators, its annotation can be obtained from a native annotator alone. With this goal, we train an SVM-based classifier that predicts (dis)agreement. In the training dataset, the agreement label is assigned using our datasets that had multiple annotations. We use three sets of features: (a) POS, (b) Named entities, (c) Unigrams (a & b are obtained from NLTK [Bird, 2006]). Table 3.6 shows the performance for 3-fold cross-validation, averaged over the two annotators as in the previous case. We obtain an AUC of 0.56 for Tweet-A, and 0.59 for Discussion-A. The high accuracy and AUC values show that words and lexical features (such as named entities and part-of-speech tags) can effectively predict disagreements between native and non-native annotators.

⁷This means that the experiment in case of Indian annotators as training labels consisted of two runs, one for each annotator.

3.2 Sarcasm-versus-irony Classification

In the previous section, we describe how implied negative sentiment in sarcasm may not be understood by annotators of a cultural background different from the source text. In this section, we aim to understand the second component of sarcasm: presence of a target of ridicule. As stated in Chapter 1, sarcasm has a target of ridicule, which is absent in the case of irony [Lee and Katz, 1998]. For example, the sentence ‘*He invented a new cure for a heart disease but later, died of the same disease himself*’ is ironic but not sarcastic. On the other hand, sarcasm is always contemptuous or ridiculing as in the case of ‘*I didn’t make it big in Hollywood because I don’t write bad enough*’. This sentence is contemptuous towards Hollywood. Since sarcasm has a target that is being ridiculed, it is crucial that sarcasm be distinguished from mere irony. Towards this end, we compare sarcasm-versus-irony classification with sarcasm-versus-philosophy classification. ‘Philosophy’ here refers to general philosophical statements and is chosen due to availability of data (discussed below). In case of sarcasm-versus-irony classification, the two classes are similar (where sarcasm is a hurtful/contemptuous form of irony). In case of sarcasm-versus-philosophy classification, the two classes are likely to be diverse.

The sarcasm-versus-irony classification problem has been reported in past work. Wang [2013] present a linguistic analysis to understand differences between sarcasm and irony. According to them, aggressiveness is the distinguishing factor between the two. Sulis et al. [2016] present a set of classifiers that distinguish between sarcasm and irony. The novelty of our study as compared to these works is that we look at sarcasm-versus-irony classification from two perspectives: the human perspective, and the computational perspective. Specifically, we first describe agreement statistics and challenges faced by human annotators to distinguish between sarcasm and irony. Then, to validate computational challenges of the task, we compare sarcasm-versus-irony classification with sarcasm-versus-philosophy classification. The work described in this section was presented at the Australasian Language Technology Association (ALTA) Workshop 2016 ⁸.

⁸<https://www.aclweb.org/anthology/U/U16/U16-1.pdf>

		Original Labels		
		Sarcasm	Philosophy	Irony
Annotators	Sarcasm	24.33	17	7
	Philosophy	6.66	209.66	17.33
	Irony	3.33	18.66	13
	Cannot say	12.66	25	11.66

Table 3.7: Confusion matrix, averaged over three annotators

	All Three	Sarcasm-Irony	Sarcasm-Philosophy
A1	0.532	0.624	0.654
A2	0.479	0.537	0.615
A3	0.323	0.451	0.578

Table 3.8: Change in Cohen’s Kappa Values for the three annotators

3.2.1 Motivation

*Goodreads*⁹ is a book recommendation website. The website has a section containing quotes from books added by the users of the website. These quotes are accompanied with labels such as philosophy, experience, crying, etc. These labels have been assigned by the users who upload these quotes. We download a set of 4,306 quotes with three labels: philosophy, irony and sarcasm. The class-wise distribution is: (a) Sarcasm: 753, (b) Irony: 677, (c) Philosophy: 2,876. We ensure that quotes marked with one of the three labels are not marked with another label. Some examples of the three classes as they appear in our dataset are:

1. **Sarcasm:** *A woman needs a man like a fish needs a bicycle.*
2. **Irony:** *You can put anything into words, except your own life.*
3. **Philosophy:** *The best way to transform a society is to empower the women of that society.*

The first quote is sarcastic towards a man, and implies that women do not need men. The victim of sarcasm in this case is ‘a man’. Meanwhile, the second quote is ironic because the speaker thinks that a person’s own life cannot be put into words but everything else can be. It does not

⁹www.goodreads.com

express contempt or ridicule towards life or any other entity, and is thus not sarcastic. Finally, the last quote is a philosophical quote about a method for transforming society.

It is interesting to note that a sarcastic quote can be converted to a philosophical quote by word replacement. For example, converting the first quote to ‘*A woman needs a man like a fish needs water*’ makes it non-sarcastic (and arguably philosophical). Similarly, a philosophical quote can be converted to a sarcastic quote by word replacement. For example, converting the third (*i.e.*, philosophical) quote to ‘*The best way to transform a society is to empower the criminals of that society*’ makes it sarcastic.

3.2.2 Experiment Description

Three annotators with annotation experience of over 8,000 hours each participate in our annotation experiment. We refer to them as A1, A2, and A3. For a subset of quotes as described in the previous subsection¹⁰, we obtain exactly one label out of four labels: sarcasm, irony, philosophy and ‘cannot say’. The last label ‘cannot say’ is a fall-back label that indicates that the annotator could not determine a label possibly due to insufficient context or ambiguity. The three annotators annotate the dataset separately. The annotators are provided definitions of the three classes as from the Free Dictionary. In addition to these definitions, the annotators are instructed that a statement ‘*about*’ sarcasm/irony/philosophy (for example, ‘*People use sarcasm when they are tired*’) must not be marked as sarcastic/ironic/philosophical.

3.2.3 Analysis

We analyze our results from two perspectives: the human and the computational point of view, in the forthcoming sections.

The Human Perspective

This section describes difficulties that human annotators face in sarcasm-versus-irony classification. We show the confusion matrix, where each cell is averaged over values for the three annotators in Table 3.7. The rows indicate labels assigned by the annotators, whereas the columns indicate the gold label *i.e.*, the label as extracted from the source website. The values are counts averaged over three annotators. Sarcasm and irony have nearly equal (12.66 and

¹⁰This subset was selected randomly.

11.66) average values for being confused with ‘*Cannot say*’. Annotators marked ironic text as philosophical nearly three times as likely as marking sarcastic text as philosophical. This may be because the intent to ridicule in sarcasm is more pronounced than in irony which may be subtle.

Table 3.8 compares the Cohen’s Kappa values for the three annotators with the gold label. In case of annotator A1, the agreement of the annotator with the gold labels for the three-label task is 0.532. The agreement of A1 with the gold labels for the sarcasm-versus-irony task is 0.624. The corresponding value for sarcasm-versus-philosophy task is higher: 0.654. This trend holds for the two other annotators as well. In general, an annotator agrees more often with the gold label in case of sarcasm-versus-philosophy classification, as compared to sarcasm-versus-irony classification. This shows that sarcasm-versus-irony classification proves to be a more difficult task than sarcasm-versus-philosophy classification.

These are some example cases where our annotators did not agree with the gold label, for each of the two pairs.

1. **Confusion between sarcasm and irony:** Consider the example ‘... *And I wondered if we had disappointed God so much, that he wrote us off as pets, just alive to entertain.*’ Annotator A1 labeled this quote as sarcastic whereas the gold label was ironic. The annotator felt that the quote was a self-deprecating post where the speaker was being sarcastic towards themselves.
2. **Confusion between sarcasm and philosophy:** Consider another example ‘*Business people - Your business - is your greatest prejudice: it ties you to your locality, to the company you keep, to the inclinations you feel. Diligent in business - but indolent in spirit, content with your inadequacy, and with the cloak of duty hung over this contentment: that is how you live, that is how you want your children to live!*’. This example was labeled as philosophical according to the gold labels. However, Annotator A2 labeled it as sarcastic towards business people. Although the quote expresses contempt towards business people, it does not use sarcastic devices to express this contempt.

The Computational Perspective

In this section, we describe our results from training automatic classifiers to perform two classification tasks: sarcasm-versus-irony and sarcasm-versus-philosophy.

	Precision (%)	Recall (%)	F-Score (%)
sarcasm-versus-philosophy (imbalanced)			
Average	85	84.8	84.6
Weighted Average	76.5	77.7	77
sarcasm-versus-philosophy (balanced)			
Average	80.2	80	80
Weighted Average	80.2	80.1	80.1
sarcasm-versus-irony			
Average	65.4	65.4	65.4
Weighted Average	65.2	65.3	65.2

Table 3.9: Average and weighted average values for three configurations of sarcasm/irony/philosophy classification

	Precision (%)	Recall (%)	F-Score (%)
sarcasm-versus-philosophy (imbalanced)	62.2	65.8	63.8
sarcasm-versus-philosophy (balanced)	80.2	80.2	80.2
sarcasm-versus-irony	67.2	66.6	67.2

Table 3.10: Average Precision, Recall and F-score values for sarcastic class for the three configurations of sarcasm/irony/philosophy classification

We use LibSVM¹¹ to train our classifier. We use default parameters, and report results from five-fold cross-validation. We use following features: (a) Unigrams, (b) Pragmatic features: Capitalization, emoticons, punctuation marks, (c) Implicit sentiment phrases, and (d) Explicit sentiment features: # positive and negative words, largest positive/negative subsequences, lexical polarity. These features are described in detail in a forthcoming chapter. These features have shown promise for sarcasm versus non-sarcasm detection - while we use them for sarcasm-versus-philosophy and sarcasm-versus-irony classification. In other words, the intention is to understand if features designed for sarcastic-versus-non-sarcastic classification perform well for sarcasm-versus-irony classification.

We consider three classification tasks: (a) sarcasm-versus-irony, (b) sarcasm-versus-philosophy

¹¹<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

(imbalanced), and (c) sarcasm-versus-philosophy (balanced). The third configuration neutralizes the effects of data skew on performance of classification since it is known that performance on skewed datasets may not be reliable [Akbari et al., 2004]. In that case, we undersample from philosophy class by randomly eliminating some training instances (as done in Tang et al. [2009]) to ensure that there are equal number of instances from both classes in all training and test sets.

Table 3.9 shows the average and weighted average Precision, Recall and F-score values for three sets of experiments. Weighted average indicates that the average is computed by weighting according the class imbalance. On the other hand, average indicates that class imbalance is not taken into consideration.

The average F-score for sarcasm-versus-philosophy is 84.6%. In the class-balanced configuration as well, the F-score reduces to 80%. This F-score is 15% higher than that for sarcasm-versus-irony, where it is 65.4%. Also, the weighted average is 77% in case of sarcasm-versus-philosophy and 80.1% in case of sarcasm-versus-philosophy (class-balanced). The value is 12% higher than that for sarcasm-versus-irony, where it is 65.2%. The trend is common for both precision and recall. To understand how the three configurations compare, it is also important to compare their performance for the sarcasm class. Therefore, we show the average values for the three configurations for the sarcasm class in Table 3.10. For the class-balanced sarcasm-versus-philosophy configuration, the F-score for sarcasm class is 80.2%. The corresponding value for sarcasm-versus-irony is 67.2%. Our observations show that sarcasm-versus-irony is a challenging task for a supervised classifier, as compared to sarcasm-versus-philosophy. This points to the fact that for sarcasm-versus-irony classification, a new set of features are necessary.

3.3 An Approach for Identification of the Sarcasm Target

The previous section deals with sarcasm versus irony classification where the difference between a sarcastic and an ironic statement can often be determined by the presence or absence of a target of ridicule. In this section, we explore a novel, related problem: sarcasm target identification (*i.e.*, extracting the target of ridicule from a sarcastic sentence). We present an introductory approach for sarcasm target identification. While several approaches have been reported for sarcasm detection [Tsur et al., 2010, Davidov et al., 2010a, González-Ibáñez et al., 2011, Joshi et al., 2015], no past work, to the best of our knowledge, has attempted to identify

this crucial component of sarcasm: the target of ridicule [Campbell and Katz, 2012].

The rest of the section is organized as follows. We first define the sarcasm target in Section 3.3.1. Section 3.3.2 formulates the problem, while Section 3.3.3 describes our architecture in detail. The experiment setup is in Section 3.3.4. The results are in Section 3.3.5.

We define sarcasm target identification as the task of determining the target of ridicule (*i.e.*, sarcasm target) for a sarcastic text. This sarcasm target can be either explicitly mentioned in the text (by a subset of words in the sentence) or be implied but not mentioned directly (in which case we introduce a fall-back label ‘Outside’). We present an introductory approach that takes a sarcastic text as the input and returns its sarcasm target (either a subset of words in the sentence or the term ‘Outside’). Our hybrid approach employs two extractors: a rule-based extractor and a learning-based extractor (that uses a word-level classifier for every word in the sentence, to predict if the word will constitute the sarcasm target). We consider two versions of our approach: Hybrid OR (where prediction by the two extractors is OR-ed) and Hybrid AND (where prediction by the two extractors is AND-ed). Since this is the first work in sarcasm target detection, no past work exists to be used as a baseline. Hence, we devise two baselines to validate the strength of our work. The first is a simple, intuitive baseline to show if our approach holds value¹². We use a technique reported for sentiment target identification as the second baseline. Sarcasm target identification will be useful for aspect-based sentiment analysis so that the negative sentiment expressed in the sarcastic text can be attributed to the correct aspect.

3.3.1 Sarcasm Target

Sarcasm is intended to express contempt or ridicule [Campbell and Katz, 2012]. The target of this contempt or ridicule is the *sarcasm target*. For the purpose of this task, a sarcasm target is either (a) a set of words in the sarcastic text which constitute the target of ridicule, or (b) a fall-back label ‘Other’ if none of the words in the text can be identified as the target. For example, in the sentence ‘*A woman needs a man like a fish needs a bicycle*’, the sarcasm target is ‘*man*’. The focus of the current approach is to restrict to the set of words in the text. For example, the sarcasm target of the sentence ‘*His shirt would totally suit my 5 year old nephew*’ is ‘*his*’ or ‘*his shirt*’. ‘His shirt’ is being ridiculed in the sentence, and as an implication, the person being

¹²In absence of past work, simple techniques have been considered as baselines in sentiment analysis [Tan et al., 2011, Pang and Lee, 2005]

referred to by ‘his’ is being ridicule. This example shows that a sarcasm target can be a person, a thing, an action, etc. There is no restriction on the class of sarcasm targets except that it is either (a) or (b) above.

3.3.2 Motivation

Sarcasm target identification is related to sentiment target identification. Sentiment target identification deals with identifying the target (which could be an entity indicated by a noun phrase) towards which sentiment is expressed in a sentence. Qiu et al. [2011] present an approach to extract opinion words and targets collectively from a dataset. Aspect identification for sentiment has also been studied. This deals with extracting aspects of an entity (for example, color, weight, or battery life of a cell phone). Probabilistic topic models have been commonly used for the same. Titov and McDonald [2008] present a probabilistic topic model that jointly estimates sentiment and aspect for sentiment summarization. Lu et al. [2011] perform multi-aspect sentiment analysis using a topic model. Other topic model-based approaches to aspect extraction have been reported [Mukherjee and Liu, 2012].

Now consider the sarcastic sentence ‘*My cell phone has an awesome battery that lasts 20 minutes*’. This sentence ridicules the battery life of the cell phone. Aspect-based sentiment analysis needs to identify that the sentence ridicules the battery of the phone and hence, expresses negative sentiment towards the aspect ‘*battery*’. Sarcasm target identification, thus, enables aspect-based sentiment analysis to attribute the negative sentiment to the correct target aspect. In case of the sentence ‘*Can’t wait to go to class today*’, the word ‘*class*’ is the sarcasm target. Every sarcastic text has at least one sarcasm target (by definition of sarcasm), and the notion of sarcasm target is applicable for only sarcastic text (*i.e.*, non-sarcastic text does not have a sarcasm target). In case the target of ridicule is not present among these words, a fall-back label ‘Outside’ is expected. Examples of sarcasm targets are given in Table 3.11. Some challenges of sarcasm target identification are:

- **Presence of multiple candidate phrases:** Consider the sentence ‘*This phone heats up so much that I strongly recommend chefs around the world to use it as a cook-top*’. In this sentence, the words ‘*chefs*’, ‘*cook-top*’ and ‘*phone*’ are candidate phrases. However, only the ‘*phone*’ is being ridiculed in this sentence.
- **Multiple sarcasm targets:** The sarcastic sentence ‘*You are as good at coding as he is at*

cooking’ ridicules both ‘*you*’ and ‘*he*’, and hence, both are sarcasm targets.

- **Absence of a sarcasm target word (the ‘Outside’ case):** Consider the situation where a student is caught copying in a test, and the teacher says, ‘*Your parents must be so proud today!*’. No specific word in the sentence is the sarcasm target. The target here is the student. We refer to such cases as the ‘Outside’ cases.

Example	Target
Love when you don’t have two minutes to send me a quick text.	you
Don’t you just love it when Microsoft tells you that you’re spelling your own name wrong.	Microsoft
I love being ignored.	being ignored
He is as good at coding as Tiger Woods is at avoiding controversy.	He, Tiger Woods
Yeah, right! I hate catching the bus on time anyway!	Outside

Table 3.11: Examples of sarcasm targets

3.3.3 Architecture

Our hybrid approach for sarcasm target identification is depicted in Figure 3.1. The input is a sarcastic sentence while the output is the sarcasm target. The approach consists of two kinds of extractors: (a) a **rule-based extractor**, and (b) a **learning-based extractor** that uses statistical classification techniques. The two extractors individually generate lists of candidate sarcasm targets. The third component is the **integrator** that makes an overall prediction of the sarcasm target by choosing among the sarcasm targets returned by the individual extractors. The overall output is a subset of the words in the sentence. In case no word is found to be a sarcasm target, a fall-back label ‘Outside’ is returned. In the forthcoming subsections, we describe the three modules in detail.

Rule-based Extractor

Our rule-based extractor consists of nine rules that take as input the sarcastic sentence, and return a subset of words as the candidate sarcasm targets. The rules are summarized in Table 3.12.

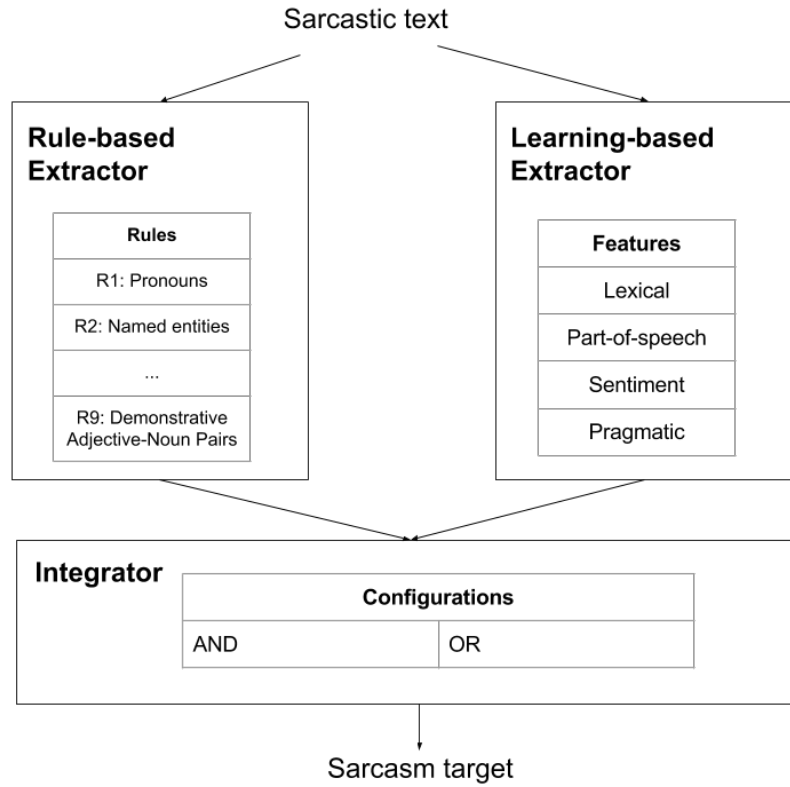


Figure 3.1: Architecture of our sarcasm target identification approach

We now describe each rule, citing past work that motivated the rule, wherever applicable:

1. **R1 (Pronouns and Pronominal Adjectives):** This rule applies to sentences that contain pronouns and pronominal adjectives. In case of pronouns, R1 returns the pronouns such as ‘*you, she, they*’. In case of pronominal adjectives, R1 returns the adjective and the phrase that follows until the next noun, as in the case of ‘*your shoes*’ or ‘*your brand new shoes*’. Thus, for the sentence ‘*I am so in love with my job*’, the phrases ‘*I*’ (pronoun) and ‘*my job*’ (based on the pronominal adjective ‘*my*’) are returned as candidate sarcasm targets. The notion of using pronouns as targets is based on observations by Shamay-Tsoory et al. [2005].
2. **R2 (Named Entities):** This rule applies to sentences containing named entities since named entities in a sentence may be sarcasm targets. This rule returns all *named entities* in the sentence. In case of the sentence ‘*Olly Riley is so original with his tweets*’, R2 predicts the phrase ‘*Olly Riley*’ as a candidate sarcasm target.
3. **R3 (Sentiment-bearing verb as the pivot):** This rule is based on the idea by Riloff et al.

[2013] that sarcasm may be expressed as a contrast between a positive sentiment verb and a negative situation. In case of the sentence '*I love being ignored*', the sentiment-bearing verb '*love*' is positive. The object of '*love*' is '*being ignored*'. Therefore, R3 returns '*being ignored*' as the candidate sarcasm target. If the sentiment-bearing verb is negative, the rule returns '*Outside*' as a candidate sarcasm target. This is applicable in case of situations like humble bragging¹³ as in '*I hate being so popular*' where the speaker is either ridiculing the listener or just bragging about themselves.

4. **R4 (Non-sentiment-bearing verb as the pivot):** This rule returns either the subject or the object of a non-sentiment-bearing verb as the pivot. The rule is triggered when a non-sentiment-bearing verb is present in the sentence. Then, the portion preceding the non-sentiment-bearing verb and that following the verb is separated. Sentiment of the two portions are computed through a simple sentiment lexicon-based word count. The portion with lower sentiment score is returned as the sarcasm target. For example, in the case of the sentence '*Excited that the teacher has decided to have a test on my birthday!*' where '*decided*' is the non-sentiment-bearing verb, rule R4 returns '*to have a test on my birthday*' as the candidate sarcasm target. This is also based on Riloff et al. [2013].
5. **R5 (Gerundial verb phrases and Infinitives):** This rule applies in the case of sentences containing gerundial verb phrases. The rule is implemented as follows. A POS-tagger marks the POS tags in the text. When a gerundial verb phrase or infinitive is encountered, the rule picks up all words until the next verb or end of the sentence. In the case of the sentence '*Being covered in rashes is fun.*', rule R5 returns the gerundial phrase '*being covered in rashes*' as the sarcasm target. In this case, the target phrase ends at the first verb '*is*'. Similarly, in case of '*Can't wait to wake up early to babysit!*', the infinitive '*to wake up early to babysit*' is returned. In this case, the target phrase extends until the end of the sentence.
6. **R6 (Noun phrases containing positive adjective):** R6 extracts noun phrases of the form '*JJ NN*' (where JJ is a positive adjective), and returns the noun indicated by NN. Specifically, 1-3 words preceding the nouns in the sentence are checked for positive sentiment. In case of the sarcastic sentence '*Look how realistic the walls in this video game are!*', the noun '*walls*' is returned as the sarcasm target.

¹³<http://www.urbandictionary.com/define.php?term=humblebrag>

7. **R7 (Interrogative sentences):** R7 returns the subject of an interrogative sentence as the sarcasm target. Thus, for the sentence ‘*A murderer is stalking me. Could life be more fun?*’, the rule returns the word ‘*life*’ as the target.
8. **R8 (Sarcasm in Similes):** This rule captures the *subjects/noun phrases* involved in similes and ‘as if’ comparisons. The rule returns the subject on both sides. The sentence is POS-tagged and contiguous sequences of nouns on either sides of the ‘as’ are returned as sarcasm targets as in ‘*He is as good as Tiger Woods is at avoiding controversy.*’. Both ‘*He*’ and ‘*Tiger Woods*’ are returned as targets. This is derived from work on sarcastic similes by Veale and Hao [2010].
9. **R9 (Demonstrative adjectives):** This rule captures nouns associated with demonstrative adjectives such as *this/that/these/those*. For example, for the sentence ‘*Oh, I love this jacket!*’, R9 returns ‘*this jacket*’ as the sarcasm target. The target begins at the demonstrative adjective and extends until the first noun.

Combining the outputs of individual rules to generate candidate sarcasm targets for the rule-based extractor: To generate the set of candidate sarcasm targets returned by the rule-based extractor, a weighted majority approach is used as follows. Every rule above is applied to the input sarcastic sentence. Then, every word is assigned a score that sums the importance weight of rules which predicted that this word was a part of the sarcasm target. The importance weight of a rule is the overall accuracy of the rule as determined by solely the learning-based extractor alone. Thus, the integrator weights each word on the basis of how reliable the rule predicting it is. The words corresponding to the maximum value of this score are returned as candidate sarcasm targets.

Learning-based Extractor

The learning-based extractor uses a classifier that takes as input a word (along with its features) and returns if the word is a sarcasm target. To do this, we decompose the task into n classification tasks, where n is the total number of words in the sentence. This means that every word in input text is considered as an instance, such that the label can be 1 or 0 depending on whether or not the given word is a part of the sarcasm target. For example, ‘*Tooth-ache is fun*’ with sarcasm target as ‘*tooth-ache*’ is broken down into three instances: ‘*tooth-ache*’ with label 1, ‘*is*’ with label 0 and ‘*fun*’ with label 0. In case the target lies outside the sentence, all words

Rule Definition	Example
R1 Return pronouns (including possessive pronouns) and pronoun based adjectives	Love when you don't have two minutes to send me a quick text .. I am so in love with my job .
R2 Return named entities as target	Don't you just love it when Microsoft tells you that you're spelling your own name wrong.
R3 Return direct object of a positive sentiment verb	I <i>love</i> being ignored .
R4 Return phrase on lower sentiment side of primary verb	So happy to just find out it has been <i>decided</i> to reschedule all my lectures and tutorials for me to night classes at the exact same times!
R5 Return Gerund and Infinitive verb phrases	Being covered in hives is so much fun! Can't wait to wake up early to babysit...
R6 Return nouns preceded by a positive sentiment adjective	Yep, this is indeed an <i>amazing</i> donut ..
R7 Return subject of interrogative sentences	A murderer is stalking me. Could life be more fun?
R8 Return subjects of comparisons (similes)	He is as good at coding as Tiger Woods is at avoiding controversy.
R9 Return demonstrative adjective-noun pairs	Oh, I love this jacket!

Table 3.12: Summary of rules in the rule-based sarcasm target extractor

Feature	Description
Lexical	
Unigram	Words
Part of Speech	
Current POS	POS tag of the current word
Previous POS	POS tag of the previous word
Next POS	POS tag of the next word
Sentimental Polarity	
Word Polarity	Polarity of the current word
Phrase Polarity	Polarity of the phrase formed by ‘previous + current + next’ words
Pragmatic	
Capitalization	Number of capital letters in the word.

Table 3.13: Features of learning-based sarcasm target extractor

have the label 0.

We then represent the instance (*i.e.*, the word) as a set of following features: (A) **Lexical**: *Unigrams*, (B) **Part of Speech (POS)**-based features: *Current POS*, *Previous POS*, *Next POS*, (C) **Polarity**-based features: *Word Polarity* : Sentiment score of the word, *Phrase Polarity* : Sentiment score of the trigram formed by considering the previous word, current word and the next word together (in that order). These polarities lie in the range [-1,+1]. These features are based on our analysis that the target phrase or word tends to be more neutral than the rest of the sentence, and (D) **Pragmatic features**: *Capitalization* : Number of capital letters in the word. Capitalization features are chosen based on features from Davidov et al. [2010a]. The features are summarized in Table 3.13.

The classifiers are trained with words as instances while the sarcasm target is to be computed at the sentence level. Hence, the candidate sarcasm target returned by the learning-based extractor consists of words for which the classifier returned 1. For example, the sentence ‘*This is fun*’ is broken up into three instances: ‘*this*’, ‘*is*’ and ‘*fun*’. If the classifier returns 1, 0, 0 for the three instances respectively, the learning-based extractor returns ‘*this*’ as the candidate sarcasm target. If the classifier returns 1, 0, 1, the learning-based extractor returns ‘*this*’ and ‘*fun*’

	Snippets	Tweets
Count	224	506
Average #words	28.47	13.06
Vocabulary	1710	1458
Total words	6377	6610
Average length of sarcasm target	1.6	2.08
Average polarity strength of sarcasm target	0.0087	0.035
Average polarity strength of portion apart from sarcasm target	0.027	0.53

Table 3.14: Statistics of sarcasm target datasets

as the candidate sarcasm target. Similarly, if the classifier returns 0, 0, 0 for the three instances, the extractor returns the fall-back label ‘*Outside*’. For each word in a particular sentence, our classifier makes the prediction, and the actual target list is compared with the list of words predicted to be target by the classifier for overlap. Correct overlaps are recorded for calculating accuracy (results are discussed in the Evaluation section).

Integrator

The integrator determines the sarcasm target based on the outputs of the two extractors. We consider two configurations of the integrator:

1. Hybrid **OR**: In this configuration, the integrator predicts the set of words that occur in the output of either of the two extractors as the sarcasm target. If the lists are empty, the output is returned as ‘*Outside*’.
2. Hybrid **AND** : In this configuration, the integrator predicts the set of words that occur in the output of both the extractors as the sarcasm target. If the intersection of the lists is empty, the output is returned as ‘*Outside*’.

The idea of using the two configurations OR and AND is based on a rule-based sarcasm detector by Khattri et al. [2015]. While AND is intuitive, the second configuration OR is necessary because our extractors individually may not capture all forms of sarcasm target.

3.3.4 Experiment Description

For the purpose of our experiments, we create a sarcasm target dataset where each textual unit is annotated with the entity being ridiculed (*i.e.*, the sarcasm target). This target is either a subset of words in the text or a fall-back label ‘*Outside*’ to indicate that the target is not present in the text. Note that all textual units (tweets as well as book snippets) in both datasets are sarcastic.

Our sarcasm target dataset consists of two text forms: book snippets and tweets. The dataset of book snippets is a sarcasm-labeled dataset by Joshi et al. [2016b]. A subset of 224 sarcastic book snippets from the dataset are used. On the other hand, for our dataset of tweets, we use the sarcasm-labeled dataset by Riloff et al. [2013]. Out of these, 506 sarcastic tweets are used. These datasets are manually annotated by the same two annotators with an experience of more than 8000 hours of text annotation each. Each annotation is either a subset of words in the text or the fall-back label ‘*Outside*’. The statistics of the sarcasm target dataset are shown in Table 3.14. The average length of a sarcasm target is 1.6 words in case of book snippets, and 2.08 words in case of tweets. The last two rows in the table point to an interesting observation about polarity strength. The polarity strength is the sum of polarities of words, based on a sentiment word-list [McAuley and Leskovec, 2013b]. The table shows that polarity strength in case of sarcasm target portion of a sentence is lower than polarity strength of the rest of the sentence. This shows that sarcasm target is more likely to be neutral than sentiment-bearing.

For our experiments, we consider two baselines:

1. **Baseline 1: All Objective Words:** As the first baseline, we design a naïve approach for our task: include all words of the sentence which are not stop words, and have neutral sentiment polarity, as the predicted sarcasm target. We reiterate that in case of no past work, simplistic baselines have been commonly reported in sentiment analysis [Pang and Lee, 2005, Tan et al., 2011].
2. **Baseline 2:** Baseline 2 is derived from past work in opinion target identification because sarcasm target identification may be considered as a form of opinion target identification. Sequence labeling has been reported for opinion target identification [Jin et al., 2009]. Therefore, we use SVM-HMM [Joachims, 2008] with default parameters as the second baseline.

For rules in the rule-based extractor, we use tools in NLTK [Bird, 2006], wherever necessary. We use SVM Perf [Joachims, 2006] for the learning-based extractor. SVM Perf optimizes

for a performance metric. In our case, we train the classifiers, optimized for F-score with epsilon $\epsilon=0.5$ and on RBF kernel. We set $C=1000$ for tweets and $C=1500$ for snippets. C is the trade-off between training error and margin, while ϵ is the tolerance for termination criterion. We report our results on four-fold cross validation for both datasets. Note that we convert individual sentences into words. Therefore, the book snippets dataset has 6377 instances, while the tweets dataset has 6610 instances. A sentence is never split into multiple folds for classification within the sarcasm target identification, and for the baseline systems. This means that for all the approaches, exactly the same sentences are present in each fold, although the representation (individual instances in classifier, and sequences in sequence labeler) may differ. This makes the comparison of the approaches valid, for all the configurations (rule-based, AND, OR, etc.). In case of Baseline 2, the task is sequence labeling. The k-folds are the same for each learning algorithm.

We report performance using two metrics: Exact Match Accuracy and Dice coefficient. These metrics have been used in information extraction [Michelson and Knoblock, 2007]. As per their conventional use, these metrics are computed at the sentence level. The metrics are described as follows:

- **Exact Match (EM) Accuracy** : An exact match occurs if the list of predicted target(s) is exactly the same as the list of actual target(s). Each target is the set of words in the sentence or the fall-back label. The accuracy is computed as the number of instances with exact match divided by total instances.
- **Dice coefficient** : Dice coefficient is used to compare similarity between two samples [Sørensen, 1948]. This is considered to be a better metric than EM accuracy because it accounts for missing words and extra words in the target. Exact match is extremely restrictive for evaluating this new concept. Dice coefficient can effectively capture the similarity between our predictions and actual target, and serves as a more reliable measure of the strength of the system under consideration. Let the two lists (predicted and actual) be X and Y . Dice coefficient is given by $(2X \cap Y)/(X + Y)$.

Note: If the actual target is the fall-back label ‘Outside’, then the expected predicted target is either ‘Outside’ or an empty prediction list. In such a case, the instance will contribute to EM accuracy.

Rule	Overall		Conditional		% Increase	
	Exact Match	Dice coefficient	Exact Match	Dice coeff.	Exact Match	Dice coeff.
R1	7.14	32.8	7.65	35.23	7.14	7.40
R2	8.48	16.7	19.19	37.81	126.29	126.40
R3	4.91	6.27	16.92	21.62	244.60	244.81
R4	2.67	11.89	4.38	19.45	64.04	63.58
R5	1.34	6.39	2.32	11.11	73.13	73.86
R6	4.01	6.77	8.91	15.02	122.19	121.86
R7	3.12	10.76	9.46	32.6	203.20	202.97
R8	4.91	6.78	35.02	45.17	613.23	566.22
R9	4.46	6.94	34.48	53.67	673.09	673.34

Table 3.15: Performance of sarcasm target identification for individual rules for **book snippets**; % Increase shows how much the score increases in case of ‘Conditional’ as compared to ‘Overall’

3.3.5 Results & Analysis

This section presents our results in two steps: performance of individual rules in the rule-based extractor, and performance of the overall approach.

Performance of Rule-based Extractor

Tables 3.15 and 3.16 present the performance of the rules in our rule-based extractor, for snippets and tweets respectively. The two metrics (EM accuracy and Dice coefficient) are reported for two cases: Overall and Conditional. ‘Overall’ spans all text units in the dataset whereas ‘Conditional’ is limited to text units which match a given rule (*i.e.*, where the given linguistic phenomenon is observed). Values in bold indicate the best performing rule for a given performance metric. The % Increase column is difference between value for ‘Conditional’ and the value for ‘Overall’ divided by that for ‘Overall’ (and then multiplied by 100).

As expected, the values for ‘conditional’ are higher than those for ‘Overall’. For example, consider rule **R7** in Table 3.15. Exact match of 3.12 (for overall accuracy) is observed as against 9.46 (for conditional accuracy). This situation is typical of rule-based systems where rules may not cover all cases but be accurate for situations that they do cover. This is likely to be because R5 returns gerundial phrases and infinitives as target. Such phrases may occur infrequently in

Rule	Overall		Conditional		% Increase	
	Exact Match	Dice coeff.	Exact Match	Dice coeff.	Exact Match	Dice coeff.
R1	6.32	19.19	8.69	26.39	37.5	37.51
R2	11.26	16.18	30.32	43.56	169.27	169.22
R3	12.45	20.28	34.24	55.77	175.02	175
R4	6.91	13.51	18.42	36	166.57	166.469
R5	9.28	23.87	15.36	39.47	65.51	65.354
R6	10.08	16.91	19.31	32.42	91.56	91.720
R7	9.88	15.21	32.25	49.65	226.41	226.42
R8	11.26	11.26	50	50	344.04	344.04
R9	11.46	13.28	43.59	50.51	280.36	280.34

Table 3.16: Performance of sarcasm target identification for individual rules for **tweets**; % Increase shows how much the score increases in case of ‘Conditional’ as compared to ‘Overall’

sarcastic text. However, when they do, they are likely to be sarcasm targets. For six out of nine rules, the increase in the metric values is high - the % increase is more than 120% for six rules in case of book snippets. The % increase is more than 166% for six out of nine rules, in case of the dataset of tweets. This shows that these rules may be specific (*i.e.*, apply to a small set of examples) but are *powerful* (*i.e.*, have a high conditional accuracy).

For tweets, R3 has a very high dice score (conditional) (55.77). This rule validates the benefit of utilizing the structure of sarcastic tweets as explored by Riloff et al. [2013] : ‘contrast of positive sentiment with negative situation’ is a strong indicator of sarcasm target. Sarcastic tweets tend to be strongly positive on the surface, but criticize or mock the target. For both the datasets, we can see that rule 8 (R8) (in conditional case) shows good results for exact match. This is likely to occur in situations like similes.

Overall Performance

We now compare the performance of the approach with the baseline (as described in Section 3.3.4). In order to understand the benefit of individual extractors, we also show their performance when they are used individually. Thus, we compare five approaches: (A) Baseline, (B) Rule-based (when only the rule-based extractor is used), (C) Learning-based (when machine learning is used), and (D) & (E) Hybrid (two configurations: OR and AND). We compare these

against two baselines, as described in a previous section.

Tables 3.17 and 3.18 compare the five approaches for snippets and tweets respectively. All our approaches outperform the baseline. In case of tweets, Table 3.18 shows that the rule-based extractor achieves a dice score of 29.13 while that for learning-based extractor is 31.8. Combining the two (as in case of Hybrid architectures) improves the dice score to 39.63. This improvement also holds for book snippets. This justifies the hybrid nature of our approach. Hybrid OR performs the best in terms of Dice coefficient. However, for exact match accuracy, Hybrid AND achieves the best performance (16.51 for snippets and 13.45 for tweets). This is likely because Hybrid AND is restrictive with respect to the predictions it makes for individual words. The learning-based extractor performs better than rule-based extractor for all three metrics. For example, in case of tweets, the dice score for learning-based extractor is 31.8 while that for rule-based extractor is 29.13.

We observe that R3 is weak in book snippets. R3 states that object of a positive sentiment verb should be returned as the sarcasm target. Therefore, a possible reason for the weak performance is that the idea of positive verb followed by a phrase towards which sarcasm is expressed is peculiar to first-person text - a characteristic of tweets more than book snippets. In addition, apart from R1, other rules perform better on tweets. The reason here again seems to be the fact that R1 is not good enough for the first-person nature of tweets. Tweets are expected to have lot more pronouns than book snippets because they are often a communicative act. Therefore, using presence of a pronoun directly as an evidence of a sarcasm target does not seem to do well.

Also, nearly all results (across approaches and metrics) are higher in case of tweets as compared to snippets. Since tweets are shorter than snippets, it is likely that they are more direct in their ridicule as compared to snippets. To validate this assumption that tweets are more direct in their ridicule than snippets, we conduct the following experiment. Book snippets and tweets containing the word ‘man’ are selected. Thirteen tweets contain the word ‘man’. Many book snippets contain the word ‘man’, of which 13 are randomly chosen among these. Thirteen tweets and 13 discussion forum posts are randomly paired up. Two human annotators (not the PhD student or his advisors) are asked to choose which of the two was more ‘direct in its ridicule’. The two human annotators are 25-30 years old, one male engineer (A1) and one female linguist (A2). The two annotators have no prior experience in sarcasm annotation but have studied English as a primary language, from school onwards. The two annotators are not

told about the claim to be verified, and also that one of them is a tweet and one of them a book snippet. For every pair, the annotator answers the question “Which of the two is more direct in its ridicule?”

A2 selects tweets to be more direct in 11 cases out of 13 while A1 does so in 10 out of 13. In other words, in 11 out of 13 cases, A2 states that for a given pair, the tweet is more direct in its ridicule than the book snippet. The cases where they thought a book snippet example was more direct in ridicule than the tweet example consisted of book snippets which were 2-3 sentences long (as opposed to longer book snippets). It must be noted that not all book snippets which were 2-3 sentences long were selected as ‘more direct than tweets’.

Approach	Exact Match	Dice coeff.
Baseline 1: All Objective Words	0.0	16.14
Baseline 2: Seq. Labeling	12.05	31.44
Only Rule-Based	9.82	26.02
Only Learning-Based	12.05	31.2
Hybrid OR	7.01	32.68
Hybrid AND	16.51	21.28

Table 3.17: Performance of sarcasm target identification for snippets

Approach	Exact Match	Dice coeff.
Baseline 1: All Objective Words	1.38	27.16
Baseline 2: Seq. Labeling	12.26	33.41
Only Rule-Based	9.48	29.13
Only Learning-Based	10.48	31.8
Hybrid OR	9.09	39.63
Hybrid AND	13.45	20.82

Table 3.18: Performance of sarcasm target identification for tweets

An analysis of errors shows that our system goes wrong in cases where the target lies outside the text.

	Book Snippets		Tweets	
	EM	DS	EM	DS
Overall	7.01	32.68	9.09	39.63
‘Outside’ cases	6.81	6.81	4.71	4.71

Table 3.19: Evaluation of sarcasm target identification for examples with target outside the text (indicated by ‘Outside’ cases)

In our dataset of book snippets, there are 11 texts (5%) which have the sarcasm target outside the text. In case of tweets, such cases are much higher: 53 tweets (10%). Table 3.19 compares the results of our hybrid (OR) approach for the specific case of target being ‘outside’ the text (indicated by ‘Outside cases’ in the table), with the results on the complete dataset (indicated by ‘Overall’ in the table). Dice coefficient (DS) for book snippets is 6.81 for ‘outside’ cases as compared to 32.68 for the complete dataset. In general, the performance for the ‘outside’ cases is significantly lower than the overall performance. This proves the difficulty that the ‘Outside’ cases present. The EM and DS values for ‘Outside’ cases are the same by definition. This is because when the target is ‘Outside’, a partial match and an exact match are the same.

3.4 Summary

This chapter described three studies that were conducted to understand the phenomenon of sarcasm. To study how negative sentiment in sarcasm may be understood, we considered the peculiar case of sarcasm-annotated datasets labeled by non-native annotators. We used two datasets annotated by American annotators: one consisting of tweets, and another consisting of discussion forum posts. We obtained an additional set of sarcasm labels from two annotators of Indian origin. We discussed our findings in three steps. The key insights from each of these steps were as follows: (1) The Indian annotators agree with each other more than they agree with their American counterparts. Also, in case of short text (tweets), the agreement is higher in sarcastic text while for long text (discussion forum posts), it is higher in non-sarcastic text. Our annotators face difficulties due to unfamiliar situations, named entities, etc. (2) Sarcasm classifiers trained on labels by Indian annotators show a statistically insignificant (as desired) degradation as compared to classifiers trained on labels by American annotators, for Tweet-A (AUC: 0.816 v/s 0.771), and for Discussion-A (AUC: 0.734 v/s 0.669). (3) Finally, using textual

features, the disagreement/difficulty in annotation can be predicted, with an AUC of 0.56.

The second component of sarcasm that we analyzed is the presence of a target. In order to do so, we presented an annotation study for sarcasm-versus-irony classification. Our annotation study validated the challenges of the sarcasm-versus-irony classification task, because sarcasm and irony are closely related to one another. We presented our findings from two perspectives: human and computational perspective. In the human perspective, we showed that our annotators agree with each other and the gold labels more in case of sarcasm and philosophy but get confused between sarcasm and irony. In the computational perspective, we trained supervised classifiers in three configurations: (a) sarcasm-versus-philosophy, (b) sarcasm-versus-philosophy (class-balanced) and (c) sarcasm versus irony. Sarcasm-versus-irony performs 12-15% lower than sarcasm-versus-philosophy. Even in case of sarcasm class, the difference is 13%.

Finally, we show if a sarcasm target can be identified using computational techniques. Towards this, we introduced a novel problem: sarcasm target identification. This problem aims to identify the target of ridicule in a sarcastic text. This target may be a subset of words in the text or a fall-back label ‘Outside’. The task poses challenges such as multiple sarcasm targets or sarcasm targets that may not even be present as words in the sentence. We present an introductory approach for sarcasm target identification that consists of two kinds of extractors: a rule-based and a learning-based extractor. Our rule-based extractor implements nine rules that capture forms of sarcasm target. The learning-based extractor splits a sentence of length n into n instances, where each instance is represented by a word, and a label that indicates if this word is a sarcasm target. Then, a statistical classifier that uses features based on POS and sentiment, predicts if a given word is likely to be a target or not. Finally, an integrator combines the outputs of the two extractors in two configurations: OR and AND. We evaluate our approach on two datasets: one consisting of book snippets, and another of tweets. In general, our hybrid OR system performs best with a Dice coefficient of 39.63. This is higher than two baselines: a naïve baseline designed for the task, and a baseline based on sentiment target identification. Our hybrid approach is also higher than the two extractors when used individually. This shows that the two extractors collectively form a good sarcasm target identification approach. Finally, we discuss the performance in case of examples where the target is outside the sentence. In such cases, our approach performs close to the overall system in terms of exact match, but there is a severe degradation in the Dice coefficient.

These studies help us understand the phenomenon of sarcasm. The observations described

in this chapter influence design decisions in our work on sarcasm detection and generation in the forthcoming chapters.

Chapter 4

Sarcasm Detection Using Incongruity Within Target Text

Prior work in sarcasm detection uses indicators such as (a) unigrams and pragmatic features (such as emoticons, etc.) [González-Ibáñez et al., 2011, Carvalho et al., 2009, Barbieri et al., 2014b], or (b) patterns extracted from techniques such as hashtag-based sentiment [Maynard and Greenwood, 2014, Liebrecht et al., 2013], a positive verb being followed by a negative situation [Riloff et al., 2013], or discriminative n-grams [Tsur et al., 2010, Davidov et al., 2010b]. In our work for sarcasm detection, we take a linguistically-grounded approach to tackle the problem of sarcasm detection. As stated in Chapter 1, our approaches are based on the linguistic notion of incongruity.

In this chapter, we present four approaches to detect sarcasm using incongruity within the text to be classified. This implies that these approaches are not able to detect sarcasm if it cannot be understood from only the text¹. We first present sarcasm detection using sentiment incongruity in Section 4.1, followed by sarcasm detection using semantic incongruity in Section 4.2. Thus, the first two sections describe approaches that present features for sarcasm detection. We show how topic models can use sentiment incongruity to detect sarcasm in Section 4.3. Finally, Section 4.4 describes a technique to use sentence completion to capture language model incongruity as an indicator for sarcasm. In Section 4.5, we compare these systems individually and in combination, to identify the best possible approach for sarcasm detection when using incongruity within the text to be classified. We summarize the chapter in Section 4.6.

¹Using information from the source of the text, and incorporating context therein is described in the next chapter

4.1 Sentiment Incongruity as Features

Our first approach uses a prominent indicator of incongruity: sentiment. Consider the sarcastic sentence ‘*I love running late*’. The sentence has a positive word ‘*love*’ followed by a negative word ‘*late*’. A positive appraisal of a negative situation of being late characterizes the sentiment incongruity in this sentence. In this section, we describe our work on sarcasm detection that captures such sentiment incongruity in the form of features of a classifier. This work was presented at the Association for Computational Linguistics (ACL) Conference, 2015².

The rest of the section is organized as follows. We describe the notion of sentiment incongruity in Section 4.1.1. We then present the architecture of our sarcasm detection system in Section 4.1.2. The experimental setup and results are in Sections 4.1.3 and 4.1.4 respectively. We analyze the errors made by our approach in Section 4.1.5.

4.1.1 Motivation

Sentiment incongruity occurs when a sentence contains interacting components with contradicting sentiment. Consider the sentence ‘*Being stranded in traffic is the best way to start my week!*’. The sentiment incongruity in this case arises from the negative sentiment in the phrase ‘*being stranded in traffic*’ and the positive word ‘*best*’. Ivanko and Pexman [2003] hypothesize that sarcasm processing time for humans depends on the degree of incongruity. Taking inspiration from this idea, we consider two cases that roughly correlate with two degrees of sentiment incongruity. We call them **explicit sentiment incongruity** and **implicit sentiment incongruity**. The two types of sentiment incongruity are described as follows.

Explicit sentiment incongruity

Explicit sentiment incongruity is sentiment incongruity expressed using the presence of sentiment words of both positive and negative polarities. Consider the example ‘*I love being ignored*’. This sentence has a positive word ‘*love*’ and a negative word ‘*ignored*’. The incongruity can be captured using these sentiment words of opposite polarity. Presence of sentiment words of such polarity being indicators of incongruity is the reason why we refer to sentiment as a ‘prominent indicator of incongruity’ in the beginning of this section.

²<https://www.aclweb.org/anthology/P/P15/P15-2124.pdf>

However, it must be noted that the converse is not true. Not all sentiment words of opposite polarity imply sarcasm. The nature of interaction also plays a key role. Consider the sentence *‘I liked the book initially but it was boring towards the end’*. This sentence contains words of opposite polarity here, namely *‘liked’* and *‘boring’*, but is not sarcastic.

Implicit sentiment incongruity

An implicit sentiment incongruity occurs when the sentiment incongruity occurs without presence of sentiment words of both polarities. We consider a simple case of implicit incongruity where a phrase implies a certain sentiment and this sentiment is incongruous with a sentiment word in the sentence. Consider the example *‘I love this paper so much that I made a doggy bag out of it’*. The polar word *‘love’* carries positive sentiment, but there is no word of contrasting polarity to make the sentiment incongruity explicit. However, the clause *‘I made a doggy bag out of it’* bears an implied sentiment that is incongruous with the polar word *‘love’*.

Estimating prevalence

To estimate prevalence of the two kinds of sentiment incongruity, we perform two evaluations on a dataset of tweets. We first conduct a naïve, automatic evaluation on a dataset of 18,141 sarcastic tweets to understand the proportion of the two kinds of sentiment incongruity. As an estimate, we consider an explicit sentiment incongruity as presence of both positive and negative words. Around 11% sarcastic tweets have at least one explicit sentiment incongruity. However, it cannot be concluded that the rest have implicit sentiment incongruity. This is because there may be other forms of context incongruity. One such example is hyperbolic sarcasm as in the case of *‘This is the best movie ever’* where the context incongruity occurs through context beyond the target text. To estimate the proportion between explicit and implicit sentiment incongruity, we manually annotate 50 sarcastic tweets that contain sentiment incongruity. We observe that 10 have explicit sentiment incongruity, while the others have implicit sentiment incongruity.

4.1.2 Sentiment Incongruity-based Features

We propose features based on the two types of sentiment incongruity, in addition to past features for sarcasm detection. The complete set of features is shown in Table 4.1. We use four kinds of

features:

1. **Lexical:** Lexical features are unigrams obtained using feature selection techniques such as χ^2 Test and Categorical Proportional Difference.
2. **Pragmatic:** Pragmatic features include emoticons, laughter expressions, punctuation marks and capital words as given by Carvalho et al. [2009].
3. **Explicit sentiment incongruity features:** Explicit sentiment incongruity features are included as real-valued features. Our explicit sentiment incongruity features are a relevant subset of features from a past system to detect thwarting by Ramteke et al. [2013]. These features are:
 - Number of sentiment flips: The number of times a positive word is followed by a negative word, and vice versa
 - Largest positive/negative sub-sequence: The length of the longest series of contiguous positive/negative words
 - Number of positive words and number of negative words
 - Lexical Polarity: The polarity based on the basis of lexical features, as determined by Lingpipe SA system [Alias-i, 2008]. Note that the ‘*native polarity*’ need not be correct. A tweet that is strongly positive on the surface is more likely to be sarcastic than a tweet that seems to be negative. This is because sarcasm, by definition, tends to be caustic/hurtful. This also guards against misclassification in the case of humble bragging, for example, in the tweet: ‘*So i have to be up at 5am to autograph 7,000 pics of myself? Sounds like just about the worst Wednesday morning I could ever imagine*’.
4. **Implicit sentiment incongruity features:** Implicit sentiment incongruity features are phrases that are of importance to sarcasm. Specifically, these features are sets of sentiment-bearing verb and noun phrases, the latter being situations with implied sentiment (e.g. ‘*running late for work*’). To extract these phrases, we adapt the algorithm given in Riloff et al. [2013]. This iterative algorithm learns a set of positive verb phrases (PVPs) and negative situations (NSs), *i.e.* noun phrases that have implicit negative sentiment. It uses a corpus of sarcastic tweets to generate implicit sentiment phrases as follows:

- (a) Start with an initial set of known PVPs, (e.g. ‘love’)
- (b) Extract noun phrases that occur with the known PVPs in sarcastic tweets, and add them to the set of NSs. For example, from the tweet ‘*I love having a broken foot on my birthday*’, the phrase ‘*having a broken foot on my birthday*’ is added to the set of NSs.
- (c) Identify PVPs that co-occur with known NSs in sarcastic tweets, and add them to the set of PVPs. For example, from the tweet ‘*Totally enjoy having a broken foot on my birthday*’ and the set of NSs from previous step, we identify a new PVP ‘*enjoy*’.
- (d) Repeat previous two steps to iteratively expand the two sets.
- (e) At the end, remove subsumed phrases from the extracted phrase list. For example, ‘*being ignored*’ subsumes ‘*being ignored by a friend*’ and the latter is removed from the pattern list.

It is possible that the set of NSs may contain some erroneous noun phrases that do not carry an implied negative sentiment. We hope that the limited size of the tweet guards against such false positives becoming large in number. The subsumption step in the above algorithm was not used by Riloff et al. [2013]. While they use rule-based algorithms that employ these extracted phrases to detect sarcasm, the PVPs and NSs discovered in the algorithm described in the previous subsection are added to the feature vector as count-based features. To extract these sets of PVPs and NSs, we run the iterative algorithm described above, on a dataset of 4,000 tweets (50% sarcastic). The algorithm results in a total of 79 PVPs and 202 NSs.

In summary, we modify the algorithm given in Riloff et al. [2013] in two ways: (a) they extract only positive verbs and negative noun situation phrases. We generalize it to both polarities, (b) they remove subsumed phrases (i.e. ‘*being ignored*’ subsumes ‘*being ignored by a friend*’) while we retain both phrases.

4.1.3 Experiment Setup

We perform our experiments on three datasets:

1. **Tweet-A (5,208 tweets, 4,170 sarcastic)**: Hashtags in the tweet text as included by the creator of the tweet are used to search and download these tweets. This dataset consists of

Lexical		
Unigrams		Unigrams in the training corpus
Pragmatic		
Capitalization		Numeric feature indicating presence of capital letters
Emoticons & laughter expressions		Numeric feature indicating presence of emoticons and ‘lol’s expressions
Punctuation marks		Numeric feature indicating presence of punctuation marks
Implicit sentiment incongruity		
Implicit Phrases	Sentiment	Boolean feature indicating phrases extracted from the implicit phrase extraction step
Explicit sentiment incongruity		
#Explicit sentiment incongruity		Number of times a word is followed by a word of opposite polarity
Largest positive /negative subsequence		Length of largest series of words with polarity unchanged
#Positive words		Number of positive words
#Negative words		Number of negative words
Lexical Polarity		Polarity of a tweet based on words present

Table 4.1: Sentiment incongruity features for sarcasm detection

5,208 tweets, out of which 4,170 are sarcastic. We create this dataset as follows. We first download tweets with hashtags *#sarcasm* and *#sarcastic* as sarcastic tweets and *#notsarcastic* and *#notsarcastic* as non-sarcastic, using the Twitter API³. A similar hashtag-based approach to create a sarcasm-annotated dataset was employed in Bamman and Smith [2015]. The hashtags used to download the tweets are removed from the text so that they act as labels but not as features. Elementary normalization is performed. This includes replacing abbreviations (such as *lol*, *brb*) with their complete forms. This dataset was first reported in Joshi et al. [2015].

2. **Tweet-B (2,278 tweets, 506 sarcastic)**: This dataset was manually labeled by Riloff et al. [2013]. Some tweets were unavailable, due to deletion or privacy settings.
3. **Discussion-A (1,502 discussion forum posts, 752 sarcastic)**: This dataset is created from the Internet Argument Corpus [Walker et al., 2012] that contains manual annotations for sarcasm. We randomly select 752 sarcastic and 752 non-sarcastic discussion forum posts.

To extract the implicit sentiment incongruity features, we run the iterative algorithm to extract patterns as described above, on a dataset of 4,000 tweets (50% sarcastic) (also created using hashtag-based supervision). The algorithm results in a total of 79 verb phrases and 202 noun phrases. We train our classifiers for different feature combinations, using LibSVM with RBF kernel [Chang and Lin, 2011a], and report average 5-fold cross-validation values.

4.1.4 Results

Table 4.2 shows the performance of our classifiers in terms of Precision (P), Recall (R) and F-score (F), for Tweet-A. The table first reports values from a re-implementation of Riloff et al. [2013]’s two rule-based algorithms: the ordered version predicts a tweet as sarcastic if it has a positive verb phrase followed by a negative situation/noun phrase, while the unordered does so if the two are present in any order. We see that all statistical classifiers surpass the rule-based algorithms. The best F-score obtained is 0.8876 when all four kinds of features are used. This is a gain of about 5% over the baseline, and 40% over the algorithm by Riloff et al. [2013]. Table 4.3 shows that even in the case of the Discussion-A dataset, our features result in an improved performance. The F-score increases from 0.568 to 0.640, a gain of about 8% in case

³<https://dev.twitter.com/>

Features	P	R	F
Original Algorithm by Riloff et al. [2013]			
Ordered	0.774	0.098	0.173
Unordered	0.799	0.337	0.474
Our system			
Lexical (Baseline)	0.820	0.867	0.842
Lexical+Implicit	0.822	0.887	0.853
Lexical+Explicit	0.807	0.985	0.8871
All features	0.814	0.976	0.8876

Table 4.2: Performance on Tweet-A using rule-based algorithm and statistical classifiers using our feature combinations

Features	P	R	F
Lexical (Baseline)	0.645	0.508	0.568
Lexical+Explicit	0.698	0.391	0.488
Lexical+Implicit	0.513	0.762	0.581
All features	0.489	0.924	0.640

Table 4.3: Performance on Discussion-A using our feature combinations

of discussion forum posts, when all features are used.

Approach	P	R	F
[Riloff et al., 2013] (best reported)	0.62	0.44	0.51
[Maynard and Greenwood, 2014]	0.46	0.38	0.41
Our system (all features)	0.77	0.51	0.61

Table 4.4: Comparison of our sarcasm detection approach using sentiment incongruity with two past works, for Tweet-B

To confirm that we indeed do better, we compare our system, with their reported values. This is necessary for several reasons. For example, we re-implement their algorithm but do not have access to their exact extracted phrases. Table 4.4 shows that we achieve a 10% higher

F-score than the best reported F-score of Riloff et al. [2013]. This value is also 20% higher than our re-implementation of Maynard and Greenwood [2014] that uses their hashtag retokenizer and rule-based algorithm.

4.1.5 Error Analysis

Finally, we analyze errors made by our classifiers that use sentiment incongruity features. Some common errors are:

1. **Sentiment incongruity within hashtags:** We do not perform hashtag tokenization (*i.e.*, splitting ‘*#letthefunbegin*’ to ‘*let the fun begin*’). We observe that in about 10% examples that we analyzed, the incongruous nature of the tweet was present in a hashtag consisting of concatenated words. Consider the example of the tweet ‘*Time for soccer practice, so much fun! #mycoachsucks #welearnthesameoldsameold*’. If the hashtags had been split, the feature vector would contain the word ‘*sucks*’, and the classifier would be able to identify the sentiment incongruity.
2. **Subjective polarity:** The tweet ‘*Yay for 3 hour Chem labs*’ is tagged by the author as sarcastic, which may not be common perception. This author’s negative sentiment towards ‘*3 hour Chem labs*’ may not agree with common sentiment. Hence, our algorithm to extract implicit sentiment incongruity phrases may not be able to capture it.
3. **No sentiment incongruity within text:** This system does not detect sarcasm where sentiment incongruity is expressed outside the text. About 10% incorrectly classified examples that we analyzed, contained such a semantic incongruity.
4. **Implicit sentiment incongruity due to numbers:** Our system could not detect sentiment incongruity arising due to numbers as in ‘*Going in to work for 2 hours was totally worth the 35 minute drive.*’. ‘*So exhausted. It was a tough day having only one class today*’. The specific numeric values in these tweets convey the sentiment incongruity.
5. **Dataset granularity:** Some discussion forum posts are marked as sarcastic, but contain non-sarcastic portions, leading to irrelevant features. For example, ‘*How special, now all you have to do is prove that a glob of cells has rights. I happen to believe that a person’s life and the right to life begins at conception*’.
6. **Politeness:** In some cases, implicit sentiment incongruity was less evident because the speaker expresses sarcasm by pretending to be polite. Such politeness may not use sen-

timent words of opposing polarity as expected by this algorithm. For example, ‘*Post all your inside jokes on facebook, I really want to hear about them*’.

4.2 Semantic Incongruity as Features

In the previous section, we described how sentiment incongruity can be exploited as features for sarcasm detection. In this section, we consider a more difficult form of sarcastic expression. Consider the sentence ‘*With a sense of humor like that, you could make a living as a garbage man anywhere in the country.*’⁴ The speaker makes a subtle, contemptuous remark about the sense of humor of the listener. However, absence of sentiment words makes the sarcasm in this sentence difficult to recognize automatically.

One way to capture incongruity in the absence of sentiment words is to consider semantic incongruity between words in the sentence. The intuition is that semantic similarity/discordance is a handle for context incongruity. In case of the ‘*sense of humor*’ example above, the words ‘*sense of humor*’ and ‘*garbage man*’ are semantically dissimilar and their presence together in the sentence provides a clue to sarcasm. Hence, our set of features based on word embeddings aim to capture such semantic similarity/discordance. We measure semantic similarity using features based on distance between word embeddings. Since such semantic similarity is but one of the components of context incongruity and since existing feature sets rely on features to capture context incongruity, it is useful that the two (word embedding-based similarity, as proposed, and other forms of incongruity) be combined for sarcasm detection. Thus, our work deals with the question:

Can word embedding-based features when supplemented with the features reported in prior work help to capture semantic incongruity in order to detect sarcasm?

To the best of our knowledge, this is the first attempt in sarcasm detection that uses word embedding-based features. We establish our hypothesis using four types of word embeddings, to show that the benefit of using word embedding-based features is independent of the kind of word embedding used. This work was presented in Conference on Empirical Methods in Natural Language Processing (EMNLP) in 2016 ⁵.

⁴All examples in this section are actual instances from our dataset. Some of them may be considered offensive.

⁵<https://aclweb.org/anthology/D/D16/D16-1104.pdf>

The rest of the section is organized as follows. Section 4.2.1 motivates the use of word embedding scores for incongruity detection. Section 4.2.2 presents our word embedding-based features. Section 4.2.3 gives the experiment setup and Section 4.2.4. An error analysis is in Section 4.2.5.

4.2.1 Motivation

Consider the sarcastic sentence ‘*A woman needs a man like a fish needs a bicycle*’⁶. The sarcasm is understood from the fact that a fish has no use for a bicycle. However, this sentence does not contain any sentiment-bearing word. Existing sarcasm detection systems relying on sentiment incongruity (as in the case of Section 4.1) may not work well (or at all) in such cases of sarcasm. Word vector similarity, however, can be helpful here. Consider similarity scores between two pairs of words in the sentence:

$$\text{similarity}(\text{man}, \text{woman}) = 0.766$$

$$\text{similarity}(\text{fish}, \text{bicycle}) = 0.131$$

Words in one part of this sentence (‘*man*’ and ‘*woman*’) are more similar than words in another part of the sentence (‘*fish*’ and ‘*bicycle*’). This difference can be a clue to presence of semantic incongruity. Hence, we propose features based on similarity scores between word embeddings of words in a sentence. In general, we wish to capture the most similar and most dissimilar word pairs in the sentence, and use their scores as features for sarcasm detection.

4.2.2 Word Embedding-based Features

We now introduce our word embedding-based features. We reiterate that these features will be augmented to other features as reported in prior work. Our word embedding-based features are based on similarity scores between word embeddings. The similarity score is the cosine similarity between vectors of two words. To illustrate our features, we use our example ‘*A woman needs a man like a fish needs a bicycle*’. The scores for all pairs of words in this sentence are given in Table 4.5.

Thus, we compute two sets of features:

⁶This quote is attributed to Irina Dunn, an Australian writer (https://en.wikipedia.org/wiki/Irina_Dunn)

	man	woman	fish	needs	bicycle
man	-	0.766	0.151	0.078	0.229
woman	0.766	-	0.084	0.060	0.229
fish	0.151	0.084	-	0.022	0.130
needs	0.078	0.060	0.022	-	0.060
bicycle	0.229	0.229	0.130	0.060	-

Table 4.5: Motivational example for sentiment incongruity in sarcastic text

1. **Unweighted similarity features (S):** We first compute similarity scores for all pairs of words (except for stop words). We then return four feature values per sentence⁷:

- Maximum score of most similar word pair
- Minimum score of most similar word pair
- Maximum score of most dissimilar word pair
- Minimum score of most dissimilar word pair

For the first feature, we consider the most similar word to every word in the sentence, and the corresponding similarity scores. These most similar word scores for each word are indicated in bold in Table 4.5. Thus, the first feature in case of our example would have the value 0.766 for the man-woman pair and the second feature would take the value 0.078 for the needs-man pair. The other features are computed in a similar manner.

2. **Distance-weighted similarity features (WS):** Like in the previous case, we first compute similarity scores for all pairs of words (excluding stop-words). For all similarity scores, we divide them by square of distance between the two words. Thus the similarity between terms that are close in the sentence is weighted higher than terms which are distant from one another. Thus, for all possible word pairs, we compute four features:

- Maximum distance-weighted score of most similar word pair
- Minimum distance-weighted score of most similar word pair
- Maximum distance-weighted score of most dissimilar word pair
- Minimum distance-weighted score of most dissimilar word pair

⁷The feature values consider all words in the sentence, *i.e.*, the maximum is computed over all words.

4.2.3 Experiment Setup

We augment our word embedding-based features to the following four feature sets:

1. [Liebrecht et al., 2013]: They consider unigrams, bigrams and trigrams as features.
2. [González-Ibáñez et al., 2011]: They propose two sets of features: unigrams and dictionary-based. The latter are words from a lexical resource called LIWC. We use words from LIWC that have been annotated as emotion and psychological process words, as described in the original work.
3. [Buschmeier et al., 2014]: In addition to unigrams, they propose features such as: (a) Hyperbole (captured by three positive or negative words in a row), (b) Quotation marks and ellipsis, (c) Positive/Negative Sentiment words followed by an exclamation mark or question mark, (d) Positive/Negative Sentiment Scores followed by ellipsis (represented by a ‘...’), (e) Punctuation, (f) Interjections, and (g) Laughter expressions.
4. [Joshi et al., 2015]: In addition to unigrams, they use features based on implicit and explicit incongruity. Implicit incongruity features are patterns with implicit sentiment as extracted in a pre-processing step. We obtained the complete set of patterns from the authors of this work for our experiments. Explicit incongruity features consist of the number of sentiment flips, length of positive and negative sub-sequences and lexical polarity.

For each of the four works, we experiment with four configurations:

1. Features given in work X
2. Features given in work X + unweighted similarity features (S)
3. Features given in work X + weighted similarity features (WS)
4. Features given in work X + S + WS (*i.e.*, weighted and unweighted similarity features)

For our experiments, we create a dataset of book snippets from GoodReads, a book recommendations website. For this dataset, user-defined tags are used to determine sarcasm labels. We download snippets with the tag ‘sarcasm’ as sarcastic snippets, and the ones with ‘philosophy’ as the non-sarcastic snippets. We ensure that there is no overlap in the classes (*i.e.*, no snippet has both these tags). This results in a dataset of 3,629 snippets out of which 759 are labeled as sarcastic. This skew is similar to sarcasm detection experiments from past work [Riloff et al.,

Features	P	R	F
Baseline			
Unigrams	67.2	78.8	72.53
S	64.6	75.2	69.49
WS	67.6	51.2	58.26
Both	67	52.8	59.05

Table 4.6: Performance of unigrams versus our semantic similarity-based features using embeddings from word2vec

2013]. This dataset was reported in Joshi et al. [2016b]. We report five-fold cross-validation results on the above dataset. We use $SV M_{perf}$ [Joachims, 2006] with the parameter c set to 20, w as 3, and loss function optimizing the F-score.

We experiment with four types of pre-trained word embeddings:

1. **LSA**: We use pre-trained word embeddings based on LSA [Landauer and Dumais, 1997]⁸. The vocabulary size is 100,000.
2. **GloVe**: We use pre-trained vectors available from the GloVe project⁹. The vocabulary size in this case is 2,195,904.
3. **Dependency Weights**: We use pre-trained vectors¹⁰ weighted using dependency distance, as given in Levy and Goldberg [2014]. The vocabulary size is 174,015.
4. **word2vec**: use pre-trained Google word vectors. These were trained using word2vec tool¹¹ on the Google News corpus. The vocabulary size for word2vec is 3,000,000.

We understand that the four types of pre-trained word embeddings are trained on different data sizes. To compare among them, we consider an intersection of the word embeddings in a future subsection. However, the fact that a certain kind of word embedding was trained on a larger dataset than another, continues to hold.

To interact with the first three pre-trained vectors, we use scikit library [Pedregosa et al., 2011]. To interact with Word2vec, as well as compute various features, we use gensim library

⁸<http://www.lingexp.uni-tuebingen.de/z2/LSAspaces/>

⁹<http://nlp.stanford.edu/projects/glove/>

¹⁰<https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/>

¹¹<https://code.google.com/archive/p/word2vec/>

	LSA			GloVe			Dep. Weights			word2vec		
	P	R	F	P	R	F	P	R	F	P	R	F
L	73	79	75.8	73	79	75.8	73	79	75.8	73	79	75.8
+S	81.8	78.2	79.95	81.8	79.2	80.47	81.8	78.8	80.27	80.4	80	80.2
+WS	76.2	79.8	77.9	76.2	79.6	77.86	81.4	80.8	81.09	80.8	78.6	79.68
+S+WS	77.6	79.8	78.68	74	79.4	76.60	82	80.4	81.19	81.6	78.2	79.86
G	84.8	73.8	78.91	84.8	73.8	78.91	84.8	73.8	78.91	84.8	73.8	78.91
+S	84.2	74.4	79	84	72.6	77.8	84.4	72	77.7	84	72.8	78
+WS	84.4	73.6	78.63	84	75.2	79.35	84.4	72.6	78.05	83.8	70.2	76.4
+S+WS	84.2	73.6	78.54	84	74	78.68	84.2	72.2	77.73	84	72.8	78
B	81.6	72.2	76.61	81.6	72.2	76.61	81.6	72.2	76.61	81.6	72.2	76.61
+S	78.2	75.6	76.87	80.4	76.2	78.24	81.2	74.6	77.76	81.4	72.6	76.74
+WS	75.8	77.2	76.49	76.6	77	76.79	76.2	76.4	76.29	81.6	73.4	77.28
+S+WS	74.8	77.4	76.07	76.2	78.2	77.18	75.6	78.8	77.16	81	75.4	78.09
J	85.2	74.4	79.43	85.2	74.4	79.43	85.2	74.4	79.43	85.2	74.4	79.43
+S	84.8	73.8	78.91	85.6	74.8	79.83	85.4	74.4	79.52	85.4	74.6	79.63
+WS	85.6	75.2	80.06	85.4	72.6	78.48	85.4	73.4	78.94	85.6	73.4	79.03
+S+WS	84.8	73.6	78.8	85.8	75.4	80.26	85.6	74.4	79.6	85.2	73.2	78.74

Table 4.7: Performance obtained on augmenting semantic incongruity features to features from four prior works, for four word embeddings; L: Liebrecht et al. (2013), G: González-Ibáñez et al. (2011a), B: Buschmeier et al. (2014) , J: Joshi et al. (2015)

[Řehůřek and Sojka, 2010].

4.2.4 Results

Comparison of only word embedding-based features

Table 4.6 shows the performance of sarcasm detection when our word embedding-based features are used on their own. The embedding in this case is word2vec. The four rows show baseline sets of features: unigrams, unweighted similarity using word embeddings (S), weighted similarity using word embeddings (WS) and both (*i.e.*, unweighted plus weighted similarities using word embeddings). Using only unigrams as features gives a F-score of 72.53%, while

only unweighted and weighted features gives F-score of 69.49% and 58.26% respectively. This validates our intuition that word embedding-based features alone are not sufficient, and should be augmented with other features.

Supplementing word embedding-based features with features from prior work

Following this, we show performance using features presented in four prior works: [Buschmeier et al., 2014], [Liebrecht et al., 2013], [Joshi et al., 2015] and [González-Ibáñez et al., 2011], and compare them with those including word embedding-based features in Table 4.7.

Table 4.7 shows results for four kinds of word embeddings. All entries in the tables are higher than the simple unigrams baseline, *i.e.*, the F-score for each of the four is higher than for unigrams. This indicates that these are better features for sarcasm detection than simple unigrams. Values in bold indicate the best F-score for a given prior work-embedding type combination. In the case of Liebrecht et al. [2013] for word2vec, the overall improvement in F-score is 4%. Precision increases by 8% while recall remains nearly unchanged. The overall F-score increases by 5%. For Buschmeier et al. [2014] for word2vec, we observe an improvement in F-score from 76.61% to 78.09%. Precision remains nearly unchanged while recall increases. In case of Joshi et al. [2015] and word2vec, we observe a slight improvement of 0.20% when unweighted (S) features are used. Finally, for features given in González-Ibáñez et al. [2011], there is a negligible degradation of 0.91% when word embedding-based features based on word2vec are used. This shows that word embedding-based features are useful.

The maximum improvement is observed in case of Liebrecht et al. [2013]. It is around 4% in case of LSA, 5% in case of GloVe, 6% in case of Dependency weight-based and 4% in case of word2vec.

Comparison between the embeddings

We now wish to see which of the four embeddings benefits sarcasm detection the most. A challenge to do so is that because the four embeddings have different vocabularies (since they are trained on different datasets) and vocabulary sizes, their results cannot be directly compared. Therefore, we take an intersection of the vocabulary (*i.e.*, the subset of words present in all four embeddings) and repeat all our experiments using these intersection files. The vocabulary size of these intersection files (for all four embeddings) is 60,252.

Table 4.8 shows the average increase in F-score when a given word embedding and a

	word2vec	LSA	GloVe	Dependency Weights
+S	0.835	0.86	0.918	0.978
+WS	1.411	0.255	0.192	1.372
+S+WS	1.182	0.24	0.845	0.795

Table 4.8: Average gain in F-Scores obtained by using an intersection of the four word embeddings

Word Embedding	Average F-score Gain
LSA	0.453
Glove	0.651
Dependency	1.048
word2vec	1.143

Table 4.9: Average gain in F-scores for the four types of word embeddings

word embedding-based feature is used. These gain values are lower than in the previous case. This is because these are the values in case of the intersection versions - which are subsets of the complete embeddings. Each gain value is averaged over the four prior works. Thus, when unweighted similarity (+S) based features computed using LSA are augmented to features from prior work, an average increment of 0.835% is obtained over the four prior works. The values allow us to compare the benefit of using these four kinds of embeddings. In case of unweighted similarity-based features, dependency-based weights give the maximum gain (0.978%). In case of weighted similarity-based features, word2vec gives the maximum gain (1.411%). The same holds for the +S+WS configuration. Table 4.9 averages these values over the three types of word embedding-based features. Using Dependency-based and word2vec embeddings results in a higher improvement in F-score (1.048% and 1.143% respectively) as compared to the other embeddings.

4.2.5 Error Analysis

Some categories of errors made by our system are:

1. **Embedding issues due to incorrect senses:** Because words may have different senses, some embeddings lead to error, as in case of ‘*Great. Relationship advice from one of*

America's most wanted.'. This sentence uses a negative sense of the word 'wanted'.

2. **Lack of understanding of conversational context:** Consider the sarcastic quote '*Oh, and I suppose the apple ate the cheese*'. The similarity score between 'apple' and 'cheese' is 0.4119. This comes up as the maximum similar pair. The most dissimilar pair is 'suppose' and 'apple' with similarity score of 0.1414. The sarcasm in this sentence can be understood only in context of the complete conversation of the target text. The semantic knowledge that an apple cannot be a subject of the verb 'ate' can also be an indicator for semantic incongruity in this sentence.
3. **Metaphors in non-sarcastic text:** Figurative language may compare concepts that are not directly related but still have low similarity. Consider the non-sarcastic quote '*Oh my love, I like to vanish in you like a ripple vanishes in an ocean - slowly, silently and endlessly*'. Our system incorrectly predicts this as sarcastic.
4. **Lack of incongruity within text:** Rhetorical questions use only positive words to make a sarcastic remark. An example of such an error is '*You really are a ray of sunshine, aren't you?*'.

4.3 Sentiment Incongruity Using Topic Model

Our third approach for sarcasm detection employs a novel machine learning technique to capture sentiment incongruity. Specifically, we introduce a topic model for extraction of sarcasm-prevalent topics and sarcasm detection. Our model based on a supervised version of the Latent Dirichlet Allocation (LDA) model [Blei et al., 2003] is able to discover clusters of words that correspond to sarcastic and non-sarcastic topics.

A key idea of this model is that sarcastic tweets are likely to have a different distribution of positive and negative words as compared to literal positive or negative tweets. While in a previous section, we showed how sentiment incongruity can be captured as features, this topic model presents an alternate mechanism to capture sentiment incongruity. The work described in this section was presented at 3rd Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM), at COLING 2016¹².

Our sarcasm topic model is learned on tweets that are labeled with three sentiment labels:

¹²<https://arxiv.org/pdf/1722.113.pdf>

literal positive, literal negative and sarcastic. In order to extract sarcasm-prevalent topics, the model uses three latent word-level variables: a topic variable to indicate words that are prevalent in sarcastic discussions, a sentiment variable for sentiment-bearing words related to a topic, and a switch variable that switches between the two kinds of words (topic and sentiment-bearing words). It is important to note that the sarcasm topic model does not enforce the intuition that sarcastic tweets are likely to have a mixture of sentiment. It endeavors to discover the same from the data.

The rest of the section is organized as follows. Section 4.3.1 presents our motivation. Section 4.3.2 describes the design rationale and structure of our model. Section 4.3.3 contains the experiment setup. Section 4.3.4 discusses the results while Section 4.3.5 shows how our topic model can be applied to sarcasm detection.

4.3.1 Motivation

Topic models are popular for sentiment aspect extraction. Jo and Oh [2011] present an aspect-sentiment unification model that learns different aspects of a product, and the words that are used to express sentiment towards the aspects. In terms of the fact that they use two latent variables: one for aspect and one for sentiment, their model is similar to ours. Mukherjee and Liu [2012] use a semi-supervised model in order to extract aspect-level sentiment. The role of the supervised sentiment label in our model is similar to their work. Finally, McAuley and Leskovec [2013a] generate rating dimensions of products using topic models. However, the topic models that have been reported in past work have been for sentiment analysis in general. They do not have any special consideration to either sarcasm as a label or sarcastic tweets as a special case of tweets.

Our motivation for using topic models is two-fold. The first reason is the presence of sarcasm-prevalent topics. Consider the analogy with sentiment aspect extraction. For a restaurant review, the word ‘*spicy*’ is more likely to describe food as opposed to ambiance. Similarly, the discovery that a set of words belong to a sarcasm-prevalent topic - a topic regarding which sarcastic remarks are common - can be useful as additional information for sarcasm detection. The key idea is that some topics are more likely to evoke sarcasm than some others. For example, a tweet about working until late in the office or on homework is more probable to be sarcastic than a tweet about Mother’s Day. The second reason is the difference in sentiment distributions. A positive tweet is likely to contain only positive words, while a negative tweet is

likely to contain only negative words. On the other hand, a sarcastic tweet may contain a mix of the two kind of words (for example, ‘*I love being ignored*’ where ‘*love*’ is a positive word and ‘*ignored*’ is a negative word), except in the case of hyperbolic sarcasm (for example ‘*This is the best movie ever!*’ where ‘*best*’ is a positive word and there is no negative word).

4.3.2 Model

Design Rationale

Our topic model used sentiment labels of tweets, as in Supervised LDA [Ramage et al., 2009]. This sentiment can be positive or negative. However, in order to incorporate sarcasm, we re-organize the two sentiment values into three: literal positive, literal negative and sarcastic. The observed variable l in our model indicates this sentiment label. For sake of simplicity, we refer to the three values of l as positive, negative and sarcastic, in rest of the section.

Every word w in a tweet is either a topic word or a sentiment word. A topic word is generated due to a topic, whereas a sentiment word is generated due to combination of both topic and sentiment. This notion is common to sentiment-based topic models [Jo and Oh, 2011]. To determine whether a given word is a topic word or a sentiment word, our model uses three latent variables: a tweet-level topic label z , a word-level sentiment label s , and a switch variable is . Each tweet is assumed to have a single topic indicated by z . The single-topic assumption is reasonable considering the length of a tweet. The single-topic assumption is also similar to probabilistic Latent Semantic Analysis (pLSA) [Hofmann, 1999]. At the word level, we introduce two variables is and s . For each word, is denotes the probability of the word being a topic word or a sentiment word. Thus, the model estimates three sets of distributions: (A) probability of a word belonging to topic (ϕ_z) or sentiment-topic combination (χ_{sz}), (B) sentiment distributions over label and topic (ψ_{zl}), and (C) topic distributions over label (θ_l). The switch variable is is sampled from η_w which indicates the probability of the word being a topic word or a sentiment word ¹³.

Observed Variables and Distributions	
w	Word in a tweet
l	Label of a tweet; takes values: positive, negative, sarcastic)
Distributions	
η_w	Distribution over switch values given a word w
Latent Variables and Distributions	
z	Topic of a tweet
s	Sentiment of a word in a tweet; takes values: positive, negative
i_s	Switch variable indicating whether a word is a topic word or a sentiment word; takes values: 0, 1
Distributions	
θ_l	Distribution over topics given a label l , with prior α
ϕ_z	Distribution over words given a topic z and switch =0 (topic word), with prior γ
χ_s	Distribution over words given sentiment s and switch=1 (sentiment word), with prior δ_1
χ_{sz}	Distribution over words given a sentiment s and topic z and switch=1 (sentiment word), with prior δ_2
ψ_l	Distribution over sentiment given a label l and switch =1 (sentiment word), with prior β_1
ψ_{zl}	Distribution over sentiment given a label l and topic z and switch =1 (sentiment word), with prior β_2

Table 4.10: List of variables/distributions in sarcasm topic model

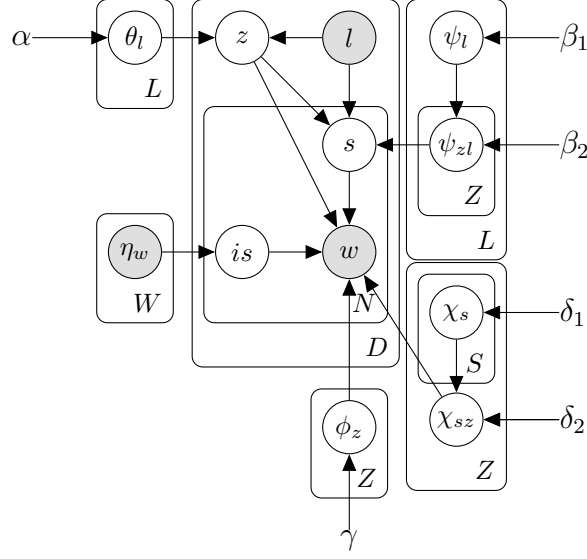


Figure 4.1: Plate diagram of sarcasm topic model

Plate Diagram

Figure 4.1 shows the plate diagram while Table 4.10 details the variables and distributions in the model. Every tweet consists of a set of observed words w and one tweet-level, observed sentiment label l . The label takes three values: positive, negative or sarcastic. The third label ‘sarcastic’ indicates a scenario where a tweet appears positive on the surface but is implicitly negative (hence, sarcastic). The tweet-level latent variable z denotes the topic of the tweet. The number of topics, Z is experimentally determined. The word-level latent variable is represents if a word is a topic word or a sentiment word, similar to Mukherjee and Liu [2012]. If the word is a sentiment word, the variable s represents the sentiment of that word. It can take S values. Intuitively, S is set as 2 (indicating positive/negative).

Among the distributions, η_w is estimated in advance. It denotes the probability of the word w being a topic word or a sentiment word. Distribution θ_l represents the distribution over z given the label of the tweet as l . ψ_l and ψ_{zl} are a hierarchical pair of distributions. The variable ψ_{zl} represents the distribution over sentiment of the word given the topic and label of the tweet and that the word is a sentiment word. The distributions χ_s and χ_{sz} denote an hierarchical pair of distributions, where χ_{sz} represents distribution over words, given the word is a sentiment word with sentiment s and topic z . The distribution ϕ_z is a distribution over words given the word is an topic word with topic z . The generative story of our model is:

¹³Note that η_w is not estimated during the sampling but learned from a large-scale corpus, as will be described later.

1. For each label l , select $\vec{\theta}_l \sim \text{Dir}(\alpha)$
2. For each label l , select $\vec{\psi}_l \sim \text{Dir}(\beta_1)$
For each topic z , select
 $\vec{\psi}_{z,l} \sim \text{Dir}(\beta_2 \vec{\psi}_l)$
3. For each topic z and sentiment s , select $\vec{\chi}_s \sim \text{Dir}(\delta_1)$, and $\vec{\chi}_{s,z} \sim \text{Dir}(\delta_2 \vec{\chi}_s)$
4. For each topic z select $\vec{\phi}_z \sim \text{Dir}(\gamma)$
5. For each tweet k ,
 - (a) l_k is the label as assigned on the basis of hashtag (l_k takes one out of three values, as described above)
 - (b) Sample topic $z_k \sim \vec{\theta}_{l_k}$
 - (c) For each word position j
 - i. Sample switch value, $is_{kj} \sim \vec{\eta}_{w_{kj}}$
 - ii. If $is_{kj} = 1$, sample $s_{kj} \sim \vec{\psi}_{z_k, l_k}$
 - iii. If $is_{kj} = 0$, generate w_{kj} from distribution of topic words $\sim \vec{\phi}_{z_k}$
 - iv. If $is_{kj} = 1$, generate w_{kj} from distribution of sentiment words $\sim \vec{\chi}_{s_{kj}, z_k}$

We estimate these distribution using Gibbs sampling. The joint probability over all variables is decomposed into these distributions, based on dependencies in the model.

4.3.3 Experiment Setup

We create a dataset of English tweets using the Twitter API¹⁴ based on hashtag-based supervision. Tweets containing hashtags #happy, #excited are labeled as positive tweets. Tweets with #sad, #angry are labeled as negative tweets. Tweets with #sarcasm and #sarcastic are labeled as sarcastic tweets. The tweets are converted to lowercase, and the hashtags used for supervision are removed. Function words¹⁵, punctuation, hashtags, author names and hyperlinks are removed from the tweets. Duplicate tweets (same tweet text repeated for multiple tweets) and re-tweets (tweet text beginning with RT) are discarded. The words which occur less than three times in the vocabulary are also removed. As a result, the tweets that have less than 3 words remaining are removed. This results in a dataset of 1,66,955 tweets. Out of these, 70,934 are

¹⁴<https://dev.twitter.com/rest/public>

¹⁵www.sequencepublishing.com

positive, 20,253 are negative and the remaining 75,769 are sarcastic. A total of 35,398 tweets are used for testing, out of which 26,210 are of positive sentiment, 5535 are of negative sentiment and 3653 are sarcastic. The total number of distinct labels (L) is 3, and the total number of distinct sentiment (S) is 2. The total number of distinct topics (Z) is experimentally determined as 50. We use block-based Gibbs sampling to estimate the distributions. The block-based sampler samples all latent variables (for each tweet) together based on their joint distributions. We set asymmetric priors based on a sentiment word-list [McAuley and Leskovec, 2013b]. A key parameter of the model is η_w since it drives the split of a word as a topic or a sentiment word. SentiWordNet [Esuli and Sebastiani, 2006] is used to learn the distribution η_w . We average across multiple senses of a word. Based on the SentiWordNet scores to all senses of a word, we determine this probability.

The scoring is done as follows. SentiWordNet provides three scores: positive, negative and objective. These scores add up to 1. Given a word w , we look it up in SentiWordNet and select the subjective-objective scores of all synsets that the word belongs to. The subjective score is the sum of positive and negative scores as given in SentiWordNet while the objective score is directly from SentiWordNet. The two scores (namely, the subjective and objective scores) are averaged over all synsets to produce the η_w values for the given word w .

In order to establish the efficacy of our sarcasm topic model, we compare against two simpler LDA-based models:

1. LDA Model provided by Mallet¹⁶ [McCallum, 2002]
2. Supervised LDA Model, an extension of LDA which estimates distributions using a labeled corpus [McAuliffe and Blei, 2008].

Both of the above models are run for 100 topics, as is the total number of topics in the sarcasm topic model.

4.3.4 Results

Qualitative Evaluation

We now present the topics discovered by our sarcasm topic model in two steps. We first describe the topics generated when only sarcastic tweets are used to estimate the distributions. In this

¹⁶<http://mallet.cs.umass.edu/>

case, the topics generated indicate words corresponding to sarcasm-prevalent topics.

The top five words (as estimated by ϕ) for a subset of topics are shown in Table 4.15. The headings in boldface are manually assigned. Sarcasm-prevalent topics, as discovered by our topic model, are work, party, weather, etc. The corresponding sentiment topics for each of these sarcasm-prevalent topics (as estimated by χ) are given in Table 4.11. The headings in boldface are manually assigned. For topics corresponding to ‘party’ and ‘women’, we observe that the two columns contain words from opposing sentiment polarities. An example sarcastic tweet about work is ‘Yaay! Another night spent at office! I love working late nights’.

Work		Party		Jokes		Weather	
0	1	0	1	0	1	0	1
Love	Great	Lol	Hate	Funny	Lol	Love	nice
Good	Sick	Attractive	Allergic	Liar	Fucks	Glad	wow
Awesome	Seriously	Love	Insulting	Hilarious	Like	Fun	really
Women		School		Love		Politics	
0	1	0	1	0	1	0	1
Compliment(s)	Talents	excited	fun	best	love	Losing	issues
Thrilled	Sorry	love	omg	awesome	ignored	lies	weep
Recognized	Bad	really	awesome	greatest	sick	like	really

Table 4.11: Sentiment-related topics estimated when the sarcasm topic model is learned on only sarcastic tweets

The previous set of topics are from sarcastic text. We now show the topics extracted by our model from the full corpus. These topics will indicate peculiarity of topics for each of the three labels, allowing us to infer what topics are sarcasm-prevalent. Table 4.12 shows top 5 topic words for the topics discovered (as estimated in ϕ) from the full corpus (*i.e.*, containing tweets of all three tweet-level sentiment labels: positive, negative and sarcastic). Table 4.13 shows the top 3 sentiment words for each sentiment (as estimated by χ) of each of the topics discovered. Like in the previous case, the heading in boldface is manually assigned. One of the topic discovered was ‘Music’. The top 5 topic words for the topic ‘Music’ are Pop, Country, Rock, Bluegrass and Beatles. The corresponding sentiment words for Music are ‘love’, ‘happy’, ‘good’ on the positive side and ‘sad’, ‘passion’ and ‘pain’ on the negative side. The remaining sections present results when the model is learned on the full corpus.

Music	work/school	Orlando Incident	Holiday	Quotes	Food
pop	work	orlando	summer	quote(s)	food
country	sleep	shooting	weekend	morning	lunch
rock	night	prayers	holiday	inspiration	vegan
bluegrass	morning	families	friends	motivation	breakfast
beatles	school	victims	sun,beach	mind	cake

Stock(s)/ Commodities	Father	Gun	Pets	Health
silver	father(s)	gun(s)	dog	fitness
gold	dad	orlando	cat	gym
index	daddy	trump	baby	run
price	family	shooting	puppy	morning
consumer	work	muslim	pets	health

Table 4.12: Topics estimated when the sarcasm topic model is learned on full corpus

Quantitative Evaluation

In this section, we answer three questions: (A) What is the likely sentiment label, if a user is talking about a particular topic?, (B) We hypothesize that sarcastic text tends to have mixed-polarity words. Does it hold in the case of our model?, (C) How can the sarcasm topic model be used for sarcasm detection?

Probability of sentiment label, given topic We compute the probability $P(l|z)$ based on the model. Table 4.14 shows these values for a subset of topics. Topics with a majority positive sentiment are Father’s Day (0.9224), holidays (0.9538), etc. The topic with the highest probability of a negative sentiment is the Orlando shooting incident (0.95). Topics regarding gun laws (0.5230), work and humor are discovered as sarcasm-prevalent.

Distribution of sentiment words for tweet-level sentiment labels Figure 4.2 shows the proportion of word-level sentiment labels, for the three tweet-level sentiment labels, as estimated by our model. The X-axis indicates the percentage of positive sentiment words in a tweet, while Y-axis indicates the percentage of tweets which indicate a specific value of sentiment proportion. More than 60% negative tweets (bar in red) have 0% positive content words. The ‘positive’

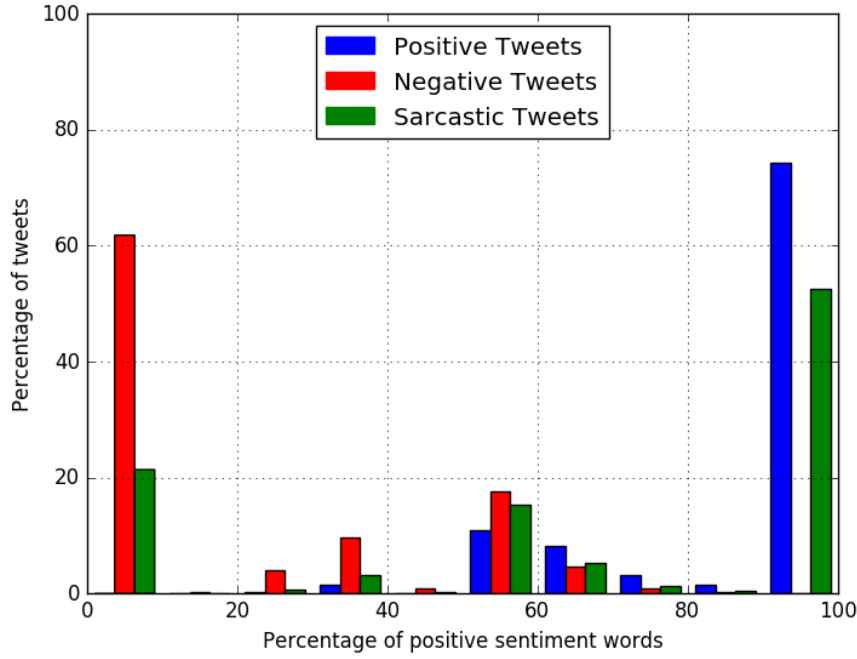


Figure 4.2: Distribution of word-level sentiment labels for tweet-level labels as extracted by sarcasm topic model

here indicates the value of s for a word in a tweet. In other words, the said red bar indicates that 60% tweets have 0% words sampled with s as positive.

It follows intuition that negative tweets have a low percentage of positive words (red bar on the left part of the graph) while positive tweets have high percentage of positive words (blue bar on the right part of the graph). Interesting variations are observed in case of sarcastic tweets. It must be highlighted that the sentiment labels considered for these proportions are as estimated by our topic model. Many sarcastic tweets contain very high percentage of positive sentiment words. Similarly, the proportion of tweets with around 50% positive sentiment words is around 20%, as expected. Thus, the model is able to capture the sentiment mixture as expected in the three tweet-level sentiment labels.

Comparison with simpler LDA-based models

Table 4.16 shows the topics learned from Supervised LDA. In general, the topics are descriptive words with few sentiment words. Therefore, we look at topics as per their sentiment prevalence. Because supervised LDA estimates sentiment-topic distributions, we list top 5 topics per label (positive, negative and sarcastic) in Table 4.17. The negative topics contain negative words as expected (*‘terrorism’*, *‘dumb’*, *‘poorly’*, etc.). However, the sarcastic topics do not seem to

Stock(s)/ Commodities		Father		Gun		Pets		Health		Music	
0	1	0	1	0	1	0	1	0	1	0	1
gains	risks	happy	lol	like	sad	happy	small	love	tired	love	sad
happiness	dipped	love	little	good	hate	love	sad	fun	sick	happy	passion
unchanged	down	best	bless	wow	angry	cute	miss	laugh	unfit	good	pain

Work/School		Orlando Incident		Holiday		Quotes		Food	
0	1	0	1	0	1	0	1	0	1
great	sick	love	tragedy	love	beauty	positive	simple	happy	foodie
fun	hate	like	hate	smile	hot	happy	kind	yummy	seriously
yay	ugh	want	heartbroken	fun	sexy	happiness	sad	healthy	perfect

Table 4.13: Sentiment-related topics estimated when the sarcasm topic model is learned on full corpus

point to sarcastic topics as such.

The LDA model, on the other hand, seems to do better. Table 4.18 shows topics extracted by LDA on full corpus. We observe that the topics correspond to: father’s day, birthdays, Orlando incident, weekends, sarcasm, food, fitness, etc. These are also the topics that are discovered as top topics in the sarcasm topic model. However, the LDA model does not associate sentiment labels with topics, since it is unsupervised. We demonstrate two ways to find sarcasm-prevalent topics using LDA:

1. To identify sarcastic topics, we select topics that contain the word ‘sarcasm’ or its forms (‘sarcastic’, ‘sorrynotsorry’) within top 10 words of the topic. ‘sorrynotsorry’ is a popular hashtag that is used to express sarcasm, and was observed in a topic. Table 4.19 shows topics that contain these words.
2. An alternative is to learn LDA on only the sarcastic tweets. Table 4.20 shows topics when LDA model is trained only on sarcastic tweets. The topics discovered are: weather, same-sex marriage, gun laws, racism, etc.

In summary, the comparison with simpler topic models shows that:

Topics $P(l z)$	Positive	Negative	Sarcastic
Holiday	0.9538	0.0140	0.0317
Father	0.9224	0.0188	0.0584
Quote	0.8782	0.0363	0.0852
Food	0.8100	0.0331	0.1566
Music	0.7895	0.0743	0.1363
Fitness	0.7622	0.0431	0.1948
Orlando Incident	0.0130	0.9500	0.0379
Gun	0.1688	0.3074	0.5230
Work	0.1089	0.0354	0.8554
Humor	0.0753	0.1397	0.7841

Table 4.14: Probability of sentiment label for topics discovered by sarcasm topic model

1. When learned only on sarcastic tweets, the LDA model discovers a set of sarcasm-prevalent topics but the supervised LDA model does not.
2. Since the LDA model is unsupervised, it does not provide sarcasm prevalence information when trained on the complete corpus.
3. The supervised LDA model does not satisfactorily discover the sarcastic topics. Our sarcasm topic model uses the notion of sentiment distribution of words, that is absent in the supervised LDA model. Thus, through its use of sentiment mixture over words, the sarcasm topic model provides an improvement over the supervised LDA model.

4.3.5 Application to Sarcasm Detection

We now use our sarcasm topic model to detect sarcasm, and compare it with two prior work.

We use the topic model for sarcasm detection using two methods:

1. **Log-likelihood based:** The topic model is first learned using the training corpus where the distributions in the model are estimated. Then, the topic model performs sampling for a pre-determined number of samples, in three runs - once for each label. For each run, the log-likelihood of the tweet, given the estimated distributions (in the training phase) and the sampled values of the latent variables (for this tweet), is computed. The label of the tweet is returned as the one with the highest log-likelihood.

Work	Party	Jokes	Weather
day	life	Quote	Snow
morning	friends	Jokes	Today
night	night	Humor	Rain
today	drunk	Comedy	Weather
work	parties	Satire	Day
Women	School	Love	Politics
Women	tomorrow	love	Ukraine
Wife	school	feeling	Russia
Compliment(s)	work	break-up	again
Fashion	morning	day/night	deeply
Love	night	sleep	raiders

Table 4.15: Topics estimated when the sarcasm topic model is learned on only sarcastic tweets

2. **Sampling-based:** Like in the previous case, the topic model first estimates distributions using the training corpus. Then, the topic model is learned again where the label l is assumed to be latent, in addition to the tweet-level latent variable z , and word-level latent variables s , and is . The value of l as estimated by the sampler is returned as the predicted label.

We compare our results with four baselines: (a) two previously existing techniques [Buschmeier et al., 2014, Liebrecht et al., 2013], and (b) using top 5 words of all topics as returned by supervised LDA¹⁷ and LDA models. We use SVM perf Joachims [2006] to train the classifiers.

Table 4.21 shows the results for sarcasm detection. Both prior work show poor F-score (around 18-19%) while our sampling based approach achieves the best F-score, for the positive sarcastic class. Similarly, LDA and supervised LDA also do not perform as well as the sarcasm topic model. The low values, in general, may be because our corpus is large in size, and is diverse in terms of the topics. Also, features in Liebrecht et al. [2013] are unigrams, bigrams and trigrams which may result in sparse features.

¹⁷The estimator in Supervised LDA model estimator returned all instances as non-sarcastic. Therefore, we use them as features, like in case of the LDA

elbow	scrolling	entire
pokemon	scream	fella
rico	instalove	esp
ears	pies	head
tragedy	original	takeover
leading	put	kids
roosevelt	loyal	movies
dropped	terrorism	allowed
accident	dumb	fierce
owning	weapon	adds
nature	sunflowers	cash
ashamed	munich	glass
chairman	explain	warrior
extinct	violence	grin
spray	arctic	homo

Table 4.16: Topics estimated when Supervised LDA is learned on full corpus

4.4 Language Model Incongruity Using Sentence Completion

In the previous sections, we describe approaches that use words, and interactions between words for sarcasm detection. This subsection presents another simple yet systematic rule-based technique for sarcasm detection. In this case, we move beyond the word level and use incongruity in language model to detect sarcasm. To gain an intuition for the technique, consider the first three words of the sarcastic sentence: *‘I love being...’*. The word likely to appear next in the sequence would be a positive sentiment word such as *‘praised’* or *‘happy’*. However, in the sarcastic sentence above, the word *‘ignored’* occurs. The word *‘ignored’* in the sarcastic sentence is semantically distant from the expected words, *‘praised’*, *‘happy’*, etc. In our technique, we use this (dis)similarity between expected word at a given word position, and the observed word at that position. In this case, this incongruity is with respect to the expected language model, since the expected word and observed word are likely to be distant in a sarcastic text. The work described in this subsection was selected for publication at PACLING 2017 and for select publication in post-proceedings of the conference.

In order to obtain the expected word at a given position, we use automatic sentence

positive	negative	sarcastic
scrolling	kids	uninformed
scream	movies	resistance
instalove	allowed	thevoice
pies	fierce	noon
original	adds	grey
entire	mvp	sarca
fella	poorly	sugar
head	trouble	automatic
takeover	domestic	nomnom
grace	ima	worldwide
leading	sorted	video
roosevelt	blackout	excite
dropped	garden	kitty
alligator	eid	head
accident	mankind	layout
doesn	put	humanity
identify	loyal	alberta
summers	terrorism	brands
spongebob	dumb	nomnom
cops	weapon	brandon
newly	company	league
dustin	tanks	entity
sword	term	part
regulations	muddy	yaay
youve	temps	memo

Table 4.17: Top 5 topics per label when Supervised LDA is learned on full corpus

father	sad	weekend
fathersday	music	friday
dad	happy	happy
love	depression	saturday
happy	cry	tgif
birthday	orlando	sarcasm
happy	shooting	laughing
cake	gay	loud
love	people	lol
bday	lgbt	laughs
rip	great	food
death	service	eat
shot	customer	healthy
family	internet	yummy
rest	happy	dinner
year	positive	healthy
great	thankful	fitness
today	blessed	love
fall	affirmation	life
asleep	grateful	gym

Table 4.18: Topics estimated when LDA model is learned on full corpus

sarcasm	tweet	year
laughing	end	great
loud	word	today
lol	words	fall
laughs	awesome*	asleep*
funny	people	room
lol	talk	house
fun	guys	web
humor	care	open
comedy*	telling*	door#

Table 4.19: Topics containing ‘sarcasm’ or related indicators, when LDA is learned on full corpus; * : ‘sarcasm’ is present in top 10 words, # : ‘sorrynotsorry’ is present in top 10 words

hour	gun	car	school	gay
half	guns	driving	high	word
home	people	road	tomorrow	south
late	control	drive	college	marriage
waiting	laws	parking	love	support
work	oscar	dad	snow	white
tomorrow	won	mom	weather	women
today	win	family	rain	black
excited	shocked	made	today	men
week	surprised	parents	cold	people
			storm	

Table 4.20: Topics estimated by LDA model on sarcastic corpus

Approach	P (%)	R (%)	F (%)
Buschmeier et al. [2014]	10.41	100.00	18.85
Liebrecht et al. [2013]	11.03	99.88	19.86
LDA	100.0	23.55	38.12
Supervised LDA as features	100.0	10.26	18.61
Sarcasm Topic Model: Log Likelihood	55.70	32.87	41.34
Sarcasm Topic Model: Sampling	58.58	13.11	21.42

Table 4.21: Performance of topic model-based approach for sarcasm detection

completion. Sentence completion predicts the most likely word at a given position in a sentence [Zweig and Burges, 2011]. For our experiments, we use context2vec, a sentence completion toolkit [Melamud et al., 2016]. Because sarcasm is an infrequent phenomenon, a sentence completion toolkit trained on a large, general-purpose corpus is likely to have learned the language model of non-sarcastic text. Thus, we explore the question:

How can sentence completion be used to detect that for a sentence, an observed word is semantically distant from the expected word, and hence, the sentence is sarcastic?

However, it must be noted that the word where the incongruity occurs (we refer to this word as the ‘*incongruous word*’) is not known in advance. We compare our performance with values reported in the past, for two datasets: the first consisting of short text (tweets), and the second consisting of long text (discussion forum posts). To the best of our knowledge, this is the first work in sarcasm detection, that uses sentence completion in order to capture incongruity as per the language model.

4.4.1 Motivation

In the sarcastic example ‘*I love being ignored*’, the word ‘*ignored*’ is observed instead of positive sentiment words. Specifically, if context2Vec is consulted to complete the sentence ‘*I love being []*’ where [] indicates the position for which the most likely word is to be computed, the word ‘*happy*’ is returned. The word2vec similarity between ‘*happy*’ and ‘*ignored*’ is 0.0204, on pre-trained word2vec embeddings. This low value of similarity between the expected and observed words can be used as an indicator for sarcasm.

However, a caveat lies in the candidate positions for which sentence completion will be consulted. This is because the incongruous word may not necessarily be the one in the last position. For example, the sentence ‘*I could not make it big in Hollywood because my writing was not bad enough*’ is sarcastic because of the incongruous word ‘*bad*’ which is at the penultimate position in the sentence.

4.4.2 Approach

In absence of the knowledge about the exact position of incongruity, our technique must iterate over multiple candidate positions. Therefore, we consider two approaches: (a) All words approach, and (b) Incongruous words approach. For both the approaches, the following holds:

Input: A text of length n

Output: Sarcastic/non-sarcastic

Parameters:

- Similarity measure $\text{sim}(w_i, w_k)$ which returns the similarity between words w_i and w_k
- Threshold T (a real value between minimum and maximum value of $\text{sim}(w_i, w_k)$)

(A) All words approach

As the name suggests, this approach considers all content words as candidate positions. This approach is as follows:

1. Set min as maximum value that sim can return
2. For position $p = 0$ to n^{18} :
 - (a) Word at position p is w_p
 - (b) Generate input string by replacing w_p with the placeholder symbol ‘[]’. Thus, for the position p of sentence “ $w_0, w_1 \dots w_n$ ”, the input string is “ $w_0, w_1, \dots w_{(p-1)} [] w_{(p+1)} \dots w_n$ ”
 - (c) Get the expected word e_p for the input string above, using context2vec
 - (d) Compute the similarity between e_p and w_p , $\text{sim}(e_p, w_p)$.
 - (e) If $\text{sim}(e_p, w_p)$ is less than min , set min as $\text{sim}(e_p, w_p)$.

¹⁸We ignore the function words, the reason for which is discussed in the next subsection.

3. If min is less than the threshold T : Predict the text as sarcastic. Else, predict the text as non-sarcastic

Thus, for the sentence ‘*A woman needs a man like a fish needs a bicycle*’ containing five content words, sentence completion will be consulted in following ways:

1. A [] needs a man like a fish needs a bicycle.
2. A woman [] a man like a fish needs a bicycle.
3. A woman needs a [] like a fish needs a bicycle.
4. A woman needs a man like a [] needs a bicycle.
5. A woman needs a man like a fish [] a bicycle.
6. A woman needs a man like a fish needs a [].

The incongruity in the sentence above occurs because the word ‘*bicycle*’ follows the preceding words. However, the sentence completion look-up will be performed for all content words, since the position of incongruity is not known to the algorithm. Therefore, looking up sentence completion for input strings 1, 2 and 4 may result in extraneous values that may affect the overall prediction. Therefore, the second approach called the Incongruous words approach works towards finding a subset of content words where the incongruity is likely to occur.

(B) Incongruous words Approach

The Incongruous words approach uses incongruity-based estimates using word2vec (not context2vec) to determine a subset of words in the sentence. The approach reduces the set of words to be checked by sentence completion to half, thereby eliminating likely redundant comparisons as shown in the previous approach. The notion of ‘most incongruous’ refers to discovering words which have least average pair-wise similarity with other content words in the text. The Incongruous words approach is as follows:

1. For position $p = 0$ to n : (As in the case of the previous approach, we ignore the function words, the reason for which is discussed in the next subsection.)
 - (a) Compute the average word2Vec similarity between word at p and words at positions other than p

2. Sort the words based on average word2vec similarity
3. Select W_I as the set of 50% most ‘*incongruous*’ words as the ones with minimum average similarity. The average is computed as a given word paired with other content words in the sentence.
4. Set min as maximum value that sim can return
5. For position $p = 0$ to n :
 - (a) Word at position p is w_p
 - (b) If w_p is not in W_I : *continue*
 - (c) As in the case of the all words approach, generate input string, and get the expected word e_p , using context2vec. Following this, compute the similarity between the expected word e_p and the observed word w_p , $sim(e_p, w_p)$.
 - (d) If $sim(e_p, w_p)$ is less than min , set min as $sim(e_p, w_p)$.
6. If min is less than the threshold T : Predict the text as sarcastic. Else, predict the text as non-sarcastic

Steps 1-3 select the required subset of words in the sentence, while Steps 4-6 are the same as in the all words approach. As a result, for the sentence ‘*A woman needs a man like a fish needs a bicycle*’, ‘*fish*’, ‘*needs*’ and ‘*bicycle*’ are returned as most incongruous Incongruous words. Hence, the sentence completion is now consulted for the following input strings:

1. A woman [] a man like a fish needs a bicycle.
2. A woman needs a man like a [] needs a bicycle.
3. A woman needs a man like a fish [] a bicycle.
4. A woman needs a man like a fish needs a [].

4.4.3 Experiment Setup

We evaluate our approach on two datasets: (a) Tweets by Riloff et al. [2013] (2278 total, 506 sarcastic, manually annotated), and (b) Discussion forum posts by Walker et al. [2012] (752 sarcastic, 752 non-sarcastic, manually annotated). We ignore function words when we iterate over word positions. We use a list of function words available online¹⁹.

¹⁹<http://www.ranks.nl/stopwords>

		P	R	F
	Riloff et al. (2013)	62	44	51
	Joshi et al. (2015)	77	51	61
Similarity	T	P	R	F
All Words Approach				
Word2Vec	0.11	67.85	45.51	54.48
WordNet	0.11	67.68	74.84	71.08
50% Approach				
Word2Vec	0.42	68.00	77.64	72.50
WordNet	0.12	82.77	77.87	80.24

Table 4.22: Performance of language model incongruity-based approaches on dataset of tweets; T: Optimal threshold, P: Precision, R: Recall, F: F-score

For both approaches, we experimentally determine the value of threshold as per the best F-score obtained. As similarity measures, we use (a) **word2vec similarity** computed using pre-trained embeddings given by the Word2Vec tool. These embeddings were learned on the Google News corpus²⁰, (b) **WordNet similarity** from WordNet::similarity by [Pedersen et al., 2004] (specifically, Wu-Palmer Similarity). The word2vec similarity in Incongruous words approach is computed in the same manner as word2vec similarity above. Since word2vec similarity may not be low for antonyms, we set the similarity measure for antonyms (as given by WordNet) as 0.

For sentence completion, we use context2vec by Melamud et al. [2016]. It is a sentence completion toolkit that uses Bidirectional LSTM to predict a missing word, given a sentence. We use the top word returned by context2vec, as per the model trained on Ukwac corpus²¹.

4.4.4 Results

In this subsection, we evaluate our approaches on the two datasets: the first consisting of short text, and the second consisting of long text.

²⁰<https://code.google.com/archive/p/Word2Vec/>

²¹<http://u.cs.biu.ac.il/~nlp/resources/downloads/context2vec/>

		P	R	F
	Joshi et al. (2015)	48.9	92.4	64
Similarity	T	P	R	F
All Words Approach				
Word2Vec	0.48	56.14	52.17	54.08
WordNet	0.27	45.12	47.68	46.37
50% Approach				
Word2Vec	0.36	37.04	47.48	41.61
WordNet	0.15	42.69	48.18	45.27

Table 4.23: Performance of language model incongruity-based approaches on dataset of discussion forum posts; T: Optimal threshold, P: Precision, R: Recall, F: F-score

Performance on short text (tweets)

Table 4.22 shows the performance for tweets. When word2vec similarity is used for the all words approach, an F-score of 54.48% is obtained. We outperform two past works ([Riloff et al., 2013] and [Joshi et al., 2015]) which have reported their values on the same dataset. The best F-score of 80.24% is obtained when word2vec is used as the similarity measure in our Incongruous words approach. We observe that the Incongruous words approach performs significantly better than the all words approach. In the case of word2vec similarity, the F-score increases by around 18%, and by around 9% in the case of WordNet similarity. Also, the optimal threshold values are lower for the all words approach as compared to the Incongruous words approach.

Performance on long text (discussion forum posts)

Table 4.23 shows the performance of our approaches for discussion forum posts. In this case, our approaches do not perform as well as the past reported value in Joshi et al. [2015]. Also, unlike the tweets, the Incongruous words approach results in a degradation as compared to all words approach, for discussion forum posts. This shows that while our approach works for short text (tweets), it does not work for long text (discussion forum posts). This is because the average length of discussion forum posts is higher than that of tweets. As a result, iterating

over all words or even Incongruous words may introduce similarity comparison with irrelevant words (*‘man’* and *‘woman’* in the example in subsection 3).

4.4.5 Discussion

Since our approaches perform well for short text like tweets but not for long text such as discussion forum posts, choosing the right set of candidate positions appears to be crucial for the success of the proposed technique. The Incongruous words is a step in that direction, but we observe that it is not sufficient in the case of discussion forum posts. Therefore, we now look at the ideal case for our technique: the situation in which the exact incongruous word is known so that the similarity comparison is performed only for the incongruous word. Hence, we now compare our all words approach with an *‘only incongruous word’* (oracle) approach, when the exact incongruous word is known. In this case, we do not iterate over all word positions but consult sentence completion only for the position of the incongruous word. For the purpose of these experiments, we need a dataset where the exact incongruous word is marked. Hence, we use the dataset of tweets reported by Ghosh et al. [2015b]. Each instance in their dataset consists of three components: (a) a word, (b) a tweet containing the word and (c) the sarcastic/non-sarcastic label. In the case of sarcastic tweets, the word indicates the specific incongruous word. This dataset helps us simulate the oracle case as described above. Table 4.24 compares the all words approach with the only incongruous word approach. We observe that the F-score increases from 55.43% to 63.42% when the incongruous word is known apriori.

4.4.6 Error Analysis

Some errors made by our techniques are:

1. **Absence of WordNet senses:** For an input, the word *‘cottoned’* is returned as the most likely word. However, no sense corresponding to the word exists in WordNet. In such a case, the word is ignored.
2. **Errors in sentence completion:** The sarcastic sentence *‘Thank you for the input, I’ll take it to heart’*. For the position where the word *‘input’* is present, the expected word as returned by context2vec is *‘message’*.

Approach	T	P	R	F
All words	0.29	55.07	55.78	55.43
Only incongruous	0.014	59.13	68.37	63.42

Table 4.24: Performance of language model incongruity-based technique when the incongruous word is known; T: Optimal threshold, P: Precision, R: Recall, F: F-score

4.5 Comparative Analysis of Approaches

In this chapter, we presented approaches for sarcasm detection using incongruity within target text. In this section, we wish to compare these approaches with each other. Therefore, we evaluate these approaches individually against each other and in combination.

4.5.1 Individual Modules

For the purpose of this experimentation, we use two baselines: Liebrecht et al. [2013] and Buschmeier et al. [2014]. We consider two datasets: book snippets and tweets Riloff et al. [2013], as introduced in Section 4.2 and Section 4.1 respectively. We perform two-fold cross-validation.

The approaches that we compare are:

1. **S1:** This is the set of sentiment incongruity-based features described in Section 4.1. We use the same set of implicit incongruity patterns as used in the experiments of that section. SVM-perf [Joachims, 2006] is used to train the classifier.
2. **S2:** This is the set of semantic incongruity-based features described in Section 4.2. For this experimentation, S2 uses the best configuration that was observed in the results in Section 4.2. To compute the semantic incongruity-based features, we use word embeddings that return the best performance [Pennington et al., 2014]. SVM-perf [Joachims, 2006] is used to train the classifier.
3. **S3:** This is the approach for language model incongruity as reported in Section 4.4. For the two-fold cross-validation setup, we learn the optimal threshold in the training fold and use it for the test fold.

The folds are exactly the same across the three configurations (and the two baselines) above.

	Book Snippets			Tweets		
	P	R	F	P	R	F
Prior Work						
Buschmeier et al. [2014]	77.48	49.22	60.04	54.89	62.76	58.55
Liebrecht et al. [2013]	75.99	44.58	56.13	50.43	79.01	61.51
Our Approaches						
S1: Sentiment Incongruity	64.37	71.91	67.89	58.82	96.43	73.07
S2: Semantic Incongruity	57.245	72.345	63.77	40.97	88.52	55.7
S3: Language Model Incongruity	65	77.5	70.5	69	75.5	72

Table 4.25: Comparison of individual approaches

Note: Section 4.3 describes sentiment incongruity using a sarcasm topic model. However, for it to be able to estimate the distributions, the model requires a dataset with text marked as literal positive, literal negative or sarcastic. Since literal positive/negative labels are not available for the book snippets and the tweet dataset, we are unable to include the approach described in Section 4.3 in the comparison here.

Table 4.25 reports two-fold cross-validation values of classifiers. In case of book snippets, the best baseline is by Buschmeier et al. [2014] which achieves a % F-score of 60.04. On the other hand, all our approaches: S1, S2, S3 perform better than the baseline. The best performance is obtained in case of S3 (with an F-score of 70.5%). For the dataset of tweets, the best baseline is by Liebrecht et al. [2013] which achieves a F-score of 61.51%. S2 performs lower than the baseline (55.7%), while S1 and S2 perform better. The best performance is by S1 (73.07%).

In case of both the datasets, we observe that our systems perform with an improved recall as compared with the baselines. In case of book snippets, the highest improvement in recall is from 44.58% for Liebrecht et al. [2013] to 77.5% for S3. In case of tweets, the corresponding improvement is from 62.76% (for Buschmeier et al. [2014]) to 96.43% in case of S1.

In terms of statistical significance,

1. **Book Snippets:** The result for S1 as compared to Liebrecht et al. [2013] is significant at $p < 0.01$, with Chi values of 746.19 and 661.85 for the two folds. The result for S2 as

	Book Snippets			Tweets		
	P	R	F	P	R	F
Ensemble						
ALL	83.76	63.81	66.91	84.45	88.73	86.20
Leave one-out systems						
ALL - S1	74.62	79.07	76.26	67.94	75.79	65.73
ALL - S2	81.46	63.56	66.45	83.10	90.91	85.72
ALL - S3	74.44	79.33	76.19	69.46	77.97	65.57

Table 4.26: Comparison of ensemble-based approach for sarcasm detection; ALL indicates S1 + S2 + S3

compared to Liebrecht et al. [2013] is significant at $p < 0.05$, with Chi values of 37.2708 and 36.6128 for the two folds. The result for S4 as compared to Liebrecht et al. [2013] is significant at $p < 0.01$, with Chi values of 1434 and 1550 for the two folds.

2. **Tweets:** The result for S1 as compared to Liebrecht et al. [2013] is significant at $p < 0.01$, with Chi values of 9.99 and 31.28 for the two folds. The result for S2 as compared to Liebrecht et al. [2013] is significant at $p < 0.01$, with Chi values 86.2 and 132.2 for the two folds. The result for S4 as compared to Liebrecht et al. [2013] is significant at $p < 0.01$, with Chi values of 663.65 and 794 for the two folds.

4.5.2 Majority-based prediction

We now evaluate how these approaches would perform in combination with each other. Towards this, we use a majority-based predictor. The predictor combines output from the three systems as follows: This predictor predicts a text as sarcastic if majority of the systems predict it as sarcastic. These experiments are performed on the two datasets, book snippets and tweets, as in the previous section. In this case as well, the values are for two-fold cross-validation.

Table 4.26 shows the results of the ensemble-based approach. System *ALL* stands for the system that combines all the three predictions, while the leave one-out systems (*ALL-S1/2/3*) remove one of the three systems at a time. It must be noted that the leave one-out systems then require that both the systems need to predict a text as sarcastic for the *majority* predictions to

be sarcastic.

In case of book snippets, the best performance is observed in case of ‘ALL - S1’ where the F-score is 76.26%. This is more than 6% of the best accuracy obtained when individual systems are tested. As argued in Section 4.2, the book snippets dataset was introduced to cover forms of sarcasm where semantic incongruity (and not sentiment incongruity alone) is used for prediction of sarcasm.

In case of tweets, system *ALL* obtains the best F-score of 86.20%. This is substantially higher than what individual systems achieve, in Table 4.25, for the tweet dataset.

Thus, based on this comparative analysis, a good baseline for sarcasm detection depends on the form of dataset. It can be prescribed as follows:

1. In case of informal, short text such as tweets, our ensemble of sentiment, semantic and language model incongruity is a good baseline for future work.
2. In case of formal text such as book snippets, semantic and language model incongruity is a good baseline for future work.

4.6 Summary

This chapter presents four approaches to sarcasm detection that capture incongruity present within target text.

As the first approach, we harness the relationship between sentiment incongruity and sarcasm as a basis for sarcasm detection. Our sarcasm classifier uses four kinds of features: lexical, pragmatic, explicit sentiment incongruity, and implicit sentiment incongruity features. We evaluate our system on two text forms: tweets and discussion forum posts. We observe an improvement of 40% over a reported rule-based algorithm, and 5% over the statistical classifier baseline that uses unigrams, in the case of tweets. The corresponding improvement in the case of discussion forum posts is 8%. Our system also outperforms two past works [Riloff et al., 2013, Joshi et al., 2015] with 10-20% improvement in F-score.

The second approach shows the benefit of features based on word embeddings for sarcasm detection. We experiment with four past works in sarcasm detection, where we augment our word embedding-based features to their sets of features. Our features use the similarity score values returned by word embeddings, and are of two categories: similarity-based (where

we consider maximum/minimum similarity score of most similar/dissimilar word pair respectively), and weighted similarity-based (where we weight the similarity scores as in the previous case). We experiment with four kinds of word embeddings: LSA, GloVe, Dependency-based and Word2Vec. In the case of word2vec, for three of these past feature sets to which our features were augmented, we observe an improvement in F-score of at most 5%. Similar improvements are observed in the case of other word embeddings. A comparison of the four embeddings shows that word2vec and dependency weight-based features outperform LSA and GloVe.

As a third approach to capture incongruity within text, we presented a topic model that discovers sarcasm-prevalent topics consisting of topic words based on sentiment incongruity between sentiment words. Our topic model uses a dataset of tweets (labeled as positive, negative and sarcastic), and estimates distributions corresponding to the prevalence of a topic, and the prevalence of a sentiment-bearing words. We observed that topics such as work, weather, politics, etc. were discovered as sarcasm-prevalent topics. We evaluated the model in three steps: (a) Based on the distributions learned by our model, we show the most likely label, for each of the topics. This is to understand sarcasm-prevalence of topics when the model is learned on the full corpus. (b) We then show distribution of word-level sentiment for each tweet-level sentiment label as estimated by our model. Indeed, our intuition that sentiment distribution in a tweet is different for the three labels: positive, negative and sarcastic, holds true. (c) Finally, we show how topics from this topic model can be harnessed for sarcasm detection. We implement two approaches: one based on most likely label as per log likelihood, and another based on last sampled value during iteration. In both the cases, we are able to significantly outperform two prior work based on feature design by F-Score of around 25%.

The fourth approach shows how sentence completion can be used for sarcasm detection. Using context2vec, a sentence completion toolkit, we obtain the expected word at a given position in a sentence, and compute the similarity between the observed word at that position and the expected word. This similarity is used to predict sarcasm. We present two approaches: (a) all words approach in which context2vec is consulted for all content words in the sentence, (b) Incongruous words approach where 50% most incongruous words in the sentence are first computed, and context2vec is invoked only for them. We present our experiments on two datasets: tweets and book snippets, and for two similarity measures: word2vec similarity, and WordNet similarity. in the case of short text (tweets), our approach (80.24%) outperforms reported work (61%). Also, as expected, our Incongruous words approach performs better than all words ap-

proach. However, the improvement does not hold true for long text (discussion forum posts). This is likely due to the length of these posts and the higher number of content words they contain. Finally, we investigate an upper bound of the performance for the approach using an oracle case where the exact incongruous word is known apriori. The approach results in a 8% higher F-score as compared to the all words approach.

Finally, we also present a comparative analysis of our approaches for two datasets: book snippets and tweets. When used individually, each of our approaches (except sentiment incongruity as a topic model which could not be compared due to data limitations) outperform two baselines. In case of book snippets, language model incongruity-based approach obtains the best performance while, in case of tweets, sentiment incongruity-based approach obtains the best performance. In terms of ensembles, we observe that a combination of semantic and language model incongruity does the best for book snippets while a combination of the three (sentiment, semantic and language model incongruity) gives the best performance for tweets.

Chapter 5

Sarcasm Detection Using Contextual Incongruity

In the previous chapter, we presented approaches that capture incongruity within target text. However, as observed in errors reported by these approaches, some sarcastic text may require additional contextual information so that the sarcasm to be understood. This is true in case of sentences like ‘*Nicki Minaj, don’t I hate her!*’ or ‘*Your parents must be really proud of you!*’. These forms of sarcasm can be detected using contextual incongruity. Here, ‘*contextual*’ refers to information beyond the target text. In this chapter, we present approaches that capture contextual incongruity in order to detect sarcasm. We consider two settings. The first setting is a monologue (in Section 5.1) where a single author is being analyzed. In this case, we consider the historical context of the author *i.e.*, the text created by the author of the target text and create a sentiment map of entities. The second setting is a dialogue (in Section 5.2) where multiple participants take part in a conversation. In this case, we use sequence labeling as a novel formulation of sarcasm detection to capture contextual incongruity in the dialogue.

5.1 Contextual Incongruity in a Monologue

In this section, we present an approach that considers historical context of the author of a text to detect sarcasm in the text. Specifically, we aim to answer the questions:

What sentiment did the author express in the past about the entities in the tweet that is to be classified? Can this information help us understand if the author is being sarcastic?

To the best of our knowledge, this work is the first quantitative evidence to show that

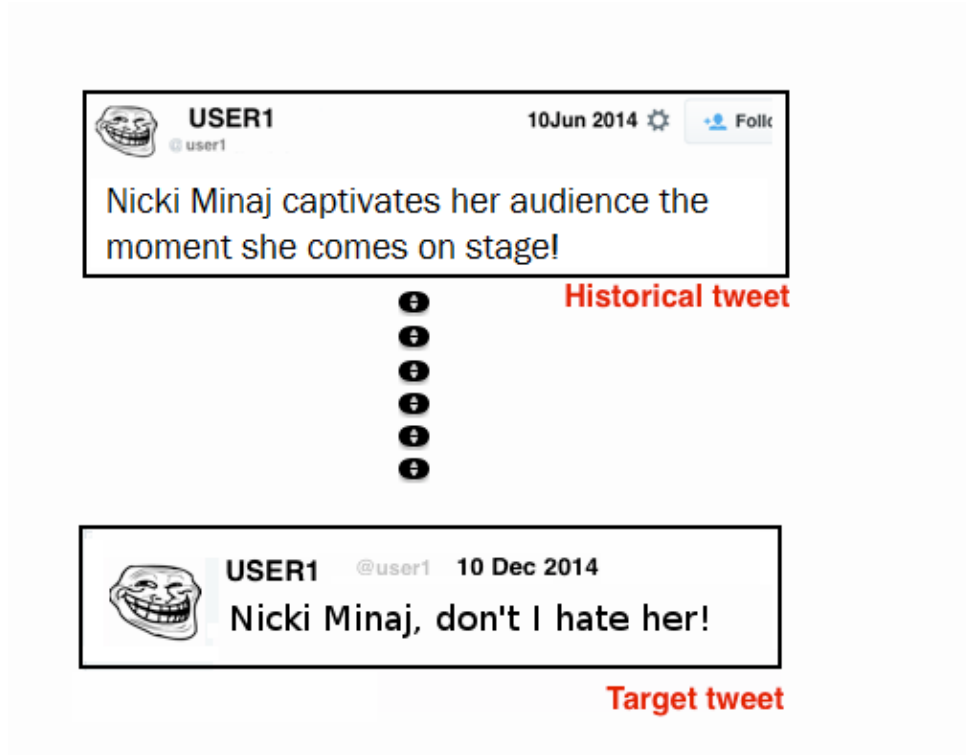


Figure 5.1: Motivating example for use of author’s historical context

historical text generated by an author may be useful to detect sarcasm in text written by the same author. In this work, we exploit the timeline structure of twitter for sarcasm detection of tweets. To gain additional context, we look up the twitter timeline of the author of the target tweet. We refer to the tweet to be classified (as sarcastic or not) as the ‘*target tweet*’ and the past tweets by the author of the target tweet as ‘*historical tweets*’. This work was presented at the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA 2015) held at EMNLP 2015¹.

The remaining section is organized as follows. We present a motivating example in Section 5.1.1, and describe the architecture of our approach in Section 5.1.2. The experiment setup and results are in Sections 5.1.3 and 5.1.4 respectively. We discuss challenges of this approach in Section 5.1.5.

5.1.1 Motivation

To demonstrate how historical context of an author may be an indicator of sarcasm, we consider the hypothetical example in Figure 5.1. The author USER1 wrote the tweet ‘*Nicki Minaj, don’t*

¹<https://aclweb.org/anthology/W/W15/W15-2905.pdf>

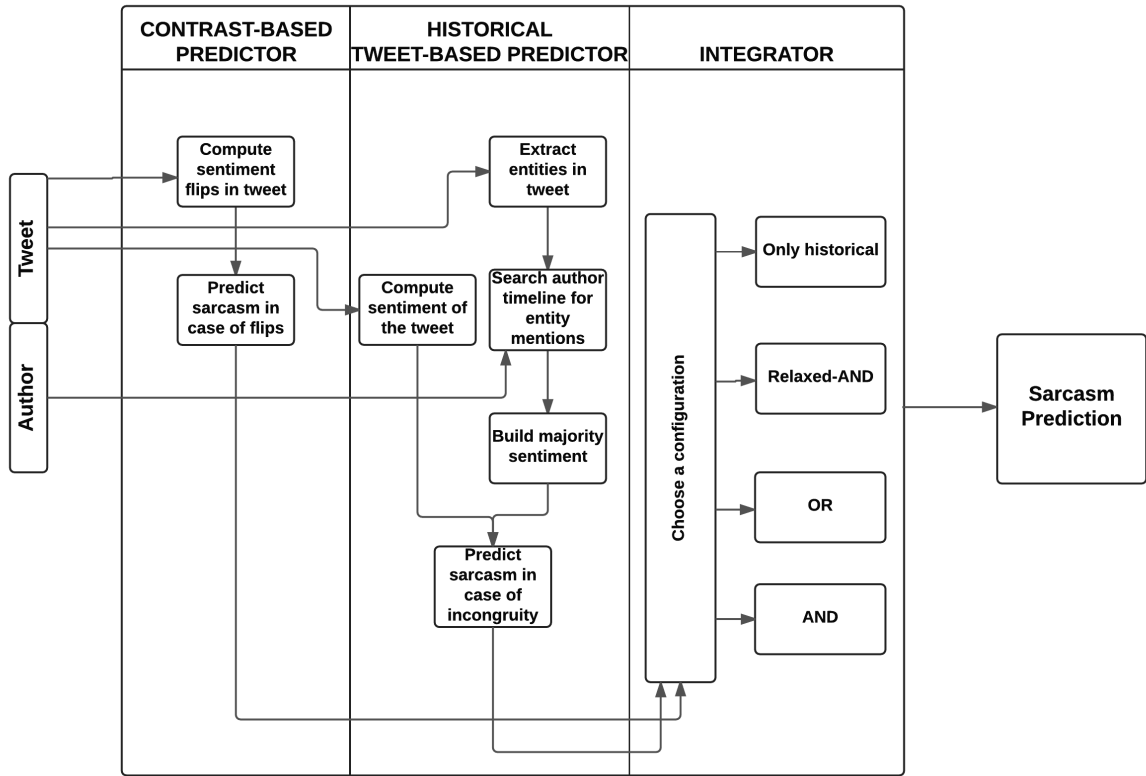


Figure 5.2: Architecture of sarcasm detection that incorporates historical incongruity in monologue

I hate her?!'. The author's historical tweets may tell us that they have spoken positively about Nicki Minaj in the past. In this case, we observe an additional tweet where the author describes having a good time at a Nicki Minaj concert. This additional knowledge helps to identify that although the target tweet contains the word '*hate*', it is intended in a sarcastic manner.

5.1.2 Architecture

Figure 5.2 shows the architecture of this system. The system takes as input the text of a tweet and the tweet id of the author, and predicts the output as either sarcastic or non-sarcastic. This is a rule-based sarcasm detection approach that consists of three modules: (a) Contrast-based Predictor, (b) Historical Tweet-based Predictor, and (c) Integrator. The working of the three modules is described hereafter.

Contrast-based Predictor

This module uses only the content of the target tweet. The contrast-based predictor identifies a sarcastic tweet using a sentiment contrast as given in Riloff et al. [2013]. A contrast is said to occur if one of the two contrasts are observed:

- **Explicit contrast:** The tweet contains one word of a polarity, and another word of another polarity.
- **Implicit Contrast:** The tweet contains one word of a polarity, and a phrase of the other polarity. The implicit sentiment phrases are extracted from a set of sarcastic tweets as described in Davidov et al. [2010b].

For example, the sentence ‘*I love being ignored.*’ is predicted as sarcastic since it has a positive word ‘*love*’ and a negative word ‘*ignored*’. We include rules to discount contrast across conjunctions like ‘*but*’. For example, ‘*I like the movie but I dislike the cinema hall*’ does not count as a contrast in terms of sarcastic expression.

Historical Tweet-based Predictor

This module uses the target tweet and the name of the author. The goal of the historical tweet-based predictor is to identify if the sentiment expressed in the tweet does not match with the historical context of tweets posted by the same author. This module makes its predictions as follows:

1. The sentiment of the target tweet is computed using a rule-based sentiment analysis system. The system takes as input a sentence, and predicts whether it is positive or negative using simple rules based on a look-up in a sentiment word list, and rules based on negation, conjunctions (such as ‘*but*’), etc. On Sentiment140² corpus, our sentiment analysis system performs with an accuracy of 58.49%. It must be noted that Sentiment140 is hashtag-annotated and hence, the accuracy is reasonable.
2. The target tweet is POS-tagged, and all NNP sequences are extracted as ‘*target phrases*’.
3. Target phrases are likely to be the targets of the sentiment expressed in the tweet. So, we download only the historical tweets which contain the target phrases³.

²<http://help.sentiment140.com/for-students>

³Twitter API allows access to the most recent 3500 tweets on a timeline. This is an additional limitation.

4. The sentiment analysis system also gives the sentiment of the downloaded historical tweets. The sentiment towards the target phrases for majority of an author's tweets is considered to be the author's historical sentiment towards the target phrase.
5. This module predicts a tweet as sarcastic if the historical sentiment is different from the sentiment of the target tweet.

A target tweet may contain more than one target phrase. In this case, the predictor considers all target phrases, and predicts the tweet as sarcastic if the above steps hold true for any of the phrases. Possible shortcomings of this approach are:

1. If the historical tweets contained sarcasm towards the target phrase, while the target tweet did not, the predictor will incorrectly mark the tweet as sarcastic.
2. If the historical tweets contained sarcasm towards the target phrase, and so did the target tweet, the predictor will incorrectly mark the tweet as non-sarcastic
3. If an entity mentioned in the target tweet never appeared in the author's historical tweets, then no input from the historical tweet is considered.

Integrator

This module combines the predictions from the historical tweet-based predictor and the contrast-based predictor. There are four versions of the module:

1. **Only historical tweet-based:** This prediction uses only the output of the historical tweet-based predictor. This also means that if this author had not mentioned the target phrase in any of his/her tweets in the past, the tweet is predicted as non-sarcastic.
2. **OR:** If either of the two predictors predict a tweet as sarcastic, then the tweet is predicted as sarcastic. If not, then it is predicted to be non-sarcastic.
3. **AND:** If both the predictors predict a tweet as sarcastic, then the tweet is predicted as sarcastic. If not, then it is predicted to be non-sarcastic.
4. **Relaxed-AND:** If both the predictors predict a tweet as sarcastic, then predict the tweet as sarcastic. If the historical tweet-based predictor did not have any tweets to look up (*i.e.*, the author had not expressed any sentiment towards the target in the past), then consider only the output of the contrast-based predictor.

Tweets with timeline restrictions are discarded.

5.1.3 Experiment Setup

For the contrast-based predictor, we obtain the implicit sentiment phrases as follows: (1) We download a set of 8000 tweets marked with #sarcasm, and assume that they are sarcastic tweets to train the system. Note that these training tweets are not the same as those used for testing. (2) We extract 3-grams to 10-grams (1-gram represents a word) in these tweets. (3) We select phrases that occur at least thrice. This results in a set of 445 phrases. These phrases are used as implicit sentiment phrases for the contrast-based predictor.

For the historical tweet-based predictor, we first POS tag the sentence using Toutanova et al. [2003]. We then select NNP sequences⁴ in the target tweet as the target phrase. Then, we download the complete timeline of the author using Twitter API⁵, and select tweets containing the target phrase. The historical tweet-based predictor then gives its prediction as described in the previous section.

Both the predictors rely on sentiment lexicons: The contrast-based predictor needs sentiment-bearing words and phrases to detect contrast, while the historical tweet-based predictor needs sentiment-bearing words to identify sentiment of a tweet. We experiment with two lexicons:

1. **Lexicon 1 (L1)**: In this case, we use the list of positive and negative words from Pang and Lee [2004].
2. **Lexicon 2 (L2)**: In this case, we use the list of positive and negative words from Mohammad and Turney [2013].

Based on the two lexicons, we run two sets of experiments:

1. **Sarcasm detection with L1 (SD1)**: In this set, we use L1 as the lexicon for the two predictors. We show results for all four integrator versions (*Only historical tweet-based*, *AND*, *OR*, *Relaxed-AND*).
2. **Sarcasm detection with L2 (SD2)**: In this set, we use L2 as the lexicon for the two predictors. We show results for all four integrator versions (*Only historical tweet-based*, *AND*, *OR*, *Relaxed-AND*).

For all experiments, we use the test corpus given by Riloff et al. [2013]. This is a manually annotated corpus consisting of 2,278 tweets⁶, out of which 506 are sarcastic.

⁴We also experimented with NN and JJ_NN sequences, but the output was unsatisfactory.

⁵<https://dev.twitter.com/overview/api>

⁶Some tweets in their original corpus could not be downloaded due to privacy settings or deletion.

	P	R	F
Best reported value by Riloff et al. [2013]	0.62	0.44	0.51
Only Historical tweet-based	0.498	0.499	0.498
OR	0.791	0.8315	0.811
AND	0.756	0.521	0.617
Relaxed-AND	0.8435	0.81	0.826

Table 5.1: Averaged Precision, Recall and F-score of the historical incongruity-based approach using SD1, for four configurations of the integrator

	P	R	F
Best reported value by Riloff et al. [2013]	0.62	0.44	0.51
Only Historical tweet-based	0.496	0.499	0.497
OR	0.842	0.927	0.882
AND	0.779	0.524	0.627
Relaxed-AND	0.880	0.884	0.882

Table 5.2: Averaged Precision, Recall and F-score of the historical incongruity-based approach using SD2, for four configurations of the integrator

5.1.4 Results

Tables 5.1 and 5.2 show Precision (P), Recall (R) and F-score (F) for SD1 and SD2 respectively. We compare our values with the best reported values in Riloff et al. [2013]. Table 5.1 shows that using only the historical tweet-based predictor, we are able to achieve a comparable performance (F-score of approximately 0.49 in case of SD1 and SD2 both) with the benchmark values (F-score of 0.51 in case of Riloff et al. [2013]). The performance values for ‘Only historical tweet-based’ are not the same in SD1 and SD2 because the lexicon used in predictors of the two approaches are different. This is obviously low because only using historical contrast is not sufficient. The AND integrator is restrictive because it requires both the predictors to predict a tweet as sarcastic. In that case as well, we obtain F-scores of 0.617 and 0.627 for SD1 and SD2 respectively. Relaxed-AND performs the best in both the cases with F-scores of 0.826 and 0.882 for SD1 and SD2 respectively.

To understand how well the two lexicons captured the positive (*i.e.*, sarcastic tweets) class, we compare their precision and recall values in Table 5.3. We observe that the positive precision

is high in case of OR, AND, Relaxed-AND. The low precision-recall values in case of ‘Only historical tweet-based’ indicates that relying purely on historical tweets may not be a good idea. The positive precision in case of Relaxed-And is 0.777 for SD1 and 0.811 for SD2. The contrast within a tweet (captured by our contrast-based predictor) and the contrast with the history (captured by our historical tweet-based predictor) both need to be applied together.

5.1.5 Error Analysis

Our target phrases are only NNP sequences. However, by the virtue of the POS tagger⁷ used, our approach predicts sarcasm correctly in following situations:

1. **Proper Nouns:** The tweet ‘*because Fox is well-balanced and objective?*’ was correctly predicted as sarcastic because our predictor located a past tweet ‘*Fox’s World Cup streaming options are terrible*’.
2. **User Mentions:** User mentions in a tweet were POS-tagged as NNPs, and hence, became target phrases. For example, a target tweet was ‘*@USERNAME ooooh that helped alot*’, where the target phrase was extracted as @USERNAME. Our approach looked at historical tweets by the author containing ‘@USERNAME’. Thus, the predictor took into consideration how cordial the two users are, based on the sentiment in historical tweets between them.
3. **Informal Expressions:** Informal expressions like ‘*Yuss*’ were tagged as NNPs. Hence, we were able to discover the common sentiment that were used in, by the author. The target tweet containing ‘*Yuss*’ was correctly marked as sarcastic.

However, some limitations of our approach are:

1. **Non-sarcastic assumption:** We assume that the author has not been sarcastic about a target phrase in the past (because we assume that the historical tweets contain an author’s true sentiment towards the target phrase).
2. **Timeline-related challenges:** Obtaining the Twitter timeline of an author may not be straightforward. A twitter timeline may be private where the user adds his/her followers, and only these followers have access to the user’s tweets. Twitter also allows change of

⁷For example, some POS taggers have a separate tag for user mentions.

twitter handle name because of which the timeline cannot be searched. In some cases, the twitter account was deactivated. Hence, we could not download the twitter timeline for 248 out of 2,273 unique authors in our dataset.

	SD1 PP	SD1 PR	SD2 PP	SD2 PR
OHTB	0.218	0.073	0.215	0.063
OR	0.647	0.785	0.691	0.978
AND	0.727	0.047	0.771	0.053
Relaxed-AND	0.777	0.675	0.811	0.822

Table 5.3: Positive Precision (PP) and Recall (PR) for historical incongruity-based sarcasm detection using SD1 and SD2; OHTB: Only Historical tweet-based

5.2 Contextual Incongruity in Dialogue

In the previous section, we described an approach for sarcasm detection in monologue. In this section, we delve into conversational context in a dialogue. Past work in sarcasm detection specifically designed for conversational text has been reported. It captures context in the form of classifier features [Rajadesingan et al., 2015, Wallace et al., 2015]. However, we propose an alternate hypothesis: in order to capture contextual incongruity, sarcasm detection of dialogue is better formulated as a sequence labeling task, instead of classification task. We aim to show that to capture contextual incongruity in dialogue, a novel formulation helps over and above features that capture such a context. In other words, the goal is to validate that the hypothesis holds irrespective of the features used, wherever applicable.

A part of the work described in this section was presented at the SIGNLL Conference on Computational Natural Language Learning (CONLL) 2016 ⁸. The deep learning experiments described in this section may be submitted to a conference in the future.

The rest of the section is organized as follows. Section 5.2.1 motivates the approach while Section 5.2.2 describes the related work. Section 5.2.3 presents our hypothesis, wherein Section 5.2.4 introduces the conversational sarcasm dataset created for the purpose of these experiments. Section 5.2.5 describes our observations in the case of traditional models. Section 5.2.6

⁸<http://aclweb.org/anthology/K/K16/K16-1015.pdf>

describes our observations in the case of deep learning models. We present the results of our experiments in Section 5.2.7. Finally, we present a discussion in Section 5.2.8.

5.2.1 Motivation

Several sentiment detection approaches for conversational data use sequence labeling [Zhao et al., 2008, Zhang et al., 2014, Mao and Lebanon, 2006, Yessenalina et al., 2010, Choi and Cardie, 2010]. However, such sequence-based formulations can be beneficial for sarcasm detection as well. Consider the possibly sarcastic statement ‘*I absolutely love this restaurant*’. Sarcasm in this sentence, if any, can be understood using context about the conversation. The conversational context may be situational: the speaker discovers a fly in her soup, then looks at her date and says, “*I absolutely love this restaurant*”. The conversational context may also be verbal: her date says, “*They’ve taken 40 minutes to bring our appetizers*” to which the speaker responds “*I absolutely love this restaurant*”. Both these examples point to the intuition that for dialogue (*i.e.*, data where more than one speaker participates in a discourse), conversational context is often a clue for sarcasm.

5.2.2 Prior Work

A past work that uses sequence labeling as a formulation is by Wang et al. [2015]. They use sequence labeling algorithm for sarcasm detection of sequences, not necessarily dialogue. They consider three kinds of sequences: (a) Sequences of text from the same user over time, (b) Sequences of text in a conversation, and (c) Sequences of text around the same topic over time. They use a labeled dataset of 1500 tweets, the labels for which are obtained automatically. Due to their automatically labeled gold dataset and their lack of focus on labeling utterances in a sequence, our analysis is more rigorous. In addition, our work substantially differs from theirs: (a) they do not deal with dialogue, (b) Their goal is to predict sarcasm of a tweet, using series of past tweets as the context, *i.e.*, only the last tweet in the sequence. Our goal is to predict sarcasm in every element of the sequence: a lot more rigorous task. The two differ in the way precision/recall values will be computed, (c) Their gold standard dataset is annotated by an automatic classifier. On the other hand, every textual unit (utterance) in our gold standard dataset is manually labeled. This makes our dataset findings more reliable.

Our dataset for these experiments consists of transcripts from a TV show. It may be ar-

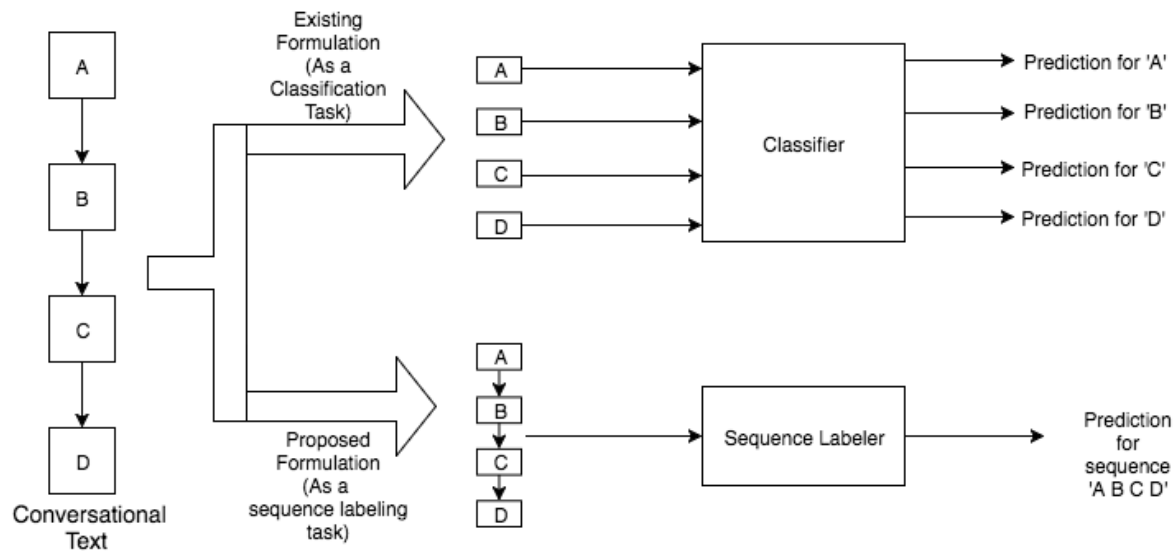


Figure 5.3: Our hypothesis for sarcasm detection of conversational text

gued that a TV series episode is dramatized and hence does not reflect real-world conversations. However, although the script is dramatized to suit the situational comedy genre, the only obvious difference from real-world conversations that we noticed was the higher volume of sarcastic sentences. Therefore, we believe our findings should generalize to real-world conversations. Moreover, datasets that are not based on fictional conversations have been used in prior work: emotion detection of children stories in Zhang et al. [2014] and speech transcripts of a MTV show in Rakov and Rosenberg [2013].

5.2.3 Architecture

In conversational text/dialogue, multiple participants take turns to speak. Consider the following snippet from TV series ‘Friends’ involving two of the lead characters, Ross and Chandler.

[Chandler is at the table. Ross walks in, looking very tanned.]

Chandler: *Hold on! There is something different.*

Ross: *I went to that tanning place your wife suggested.*

Chandler: *Was that place... The Sun?*

Chandler’s statement ‘Was that place... The Sun?’ is sarcastic. The sarcasm can be understood based on two kinds of contextual information: (a) general knowledge (that sun is both

hot and not a place one can visit) (b) Conversational context (In the previous utterance, Ross states that he went to a tanning place). Without information (b), the sarcasm cannot be understood. Thus, dialogue presents a peculiar opportunity: using sequential nature of text to capture contextual incongruity. Therefore, our hypothesis is that for sarcasm detection of dialogue, sequence labeling performs better than classification.

We consider a dialogue to be made up of a series of utterances, where an utterance is a set of one or more sentence(s) which a character speaks at one go. The goal for sarcasm detection using context incongruity in dialogue is to take as input a sequence of utterances (each utterance may be multiple sentences long), and predict every utterance as either sarcastic or non-sarcastic. This involves utterance-level prediction and does not deal with word/phrase-wise prediction.

Figure 5.3 summarizes the scope of this work. We consider two formulations for sarcasm detection of conversational text. In the first formulation (i.e. classification), a sequence is broken down into individual utterances. One utterance as an input to a classification algorithm (based on features, some of which include information from the previous utterance in the sequence) returns an output for that instance. In the second formulation (i.e. sequence labeling), a sequence is input to a sequence labeling algorithm (with same features as in the case of the classification algorithm) which returns a sequence of labels as output. Our hypothesis is that sequence labeling algorithms perform better than classification algorithms for sarcasm detection of dialogue.

To validate our hypothesis, we make use of two paradigms:

1. **Paradigm 1: Traditional Models:** In this set of experiments, we compare basic classifiers like Support Vector Machines (SVM) with sequence labeling techniques like Support Vector Machines-Hidden Markov Models (SVM-HMM). Since these models rely on manually engineered features, we experiment with two configurations: (a) features derived from our dataset, and (b) features presented in two prior works.
2. **Paradigm 2: Deep learning models:** In this set of experiments, we compare deep learning models (like Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN)) for classification with those for sequence labeling. A strength of these models is that they do not require us to provide manually engineered features.

We conduct our experiments on a conversational sarcasm dataset consisting of transcripts of dialogues from the TV show ‘*Friends*’.

	Label
[Scene: Dummy Location. Characters present, etc.]	No label
Joey: Joey's first utterance, sentence 1. Joey's first utterance, sentence 2.	Non-sarcastic
Ross: Ross' utterance, sentence 1. Ross' utterance, sentence 2.	Sarcastic
Chandler: (action Chandler does while speaking this) Chandler's utterance, sentence 1. Chandler's utterance, sentence 2. Chandler's utterance, sentence 3.	Sarcastic

Action words: (action Chandler does while speaking this)
 Spoken words: Chandler's utterance, sentence 1....
 Speaker: Chandler
 Speaker-Listener: Chandler-Ross

Figure 5.4: Example from Friends dataset: Part of a scene

5.2.4 Conversational Sarcasm Dataset

Datasets based on creative works have been explored in the past [Zhang et al., 2014]. To create our conversational sarcasm dataset, we download transcripts of a comedy TV show, ‘Friends’ (by Bright/Kauffman/Crane Productions, and Warner Bros. Entertainment Inc.)⁹. The transcripts consist of scenes which in turn consist of a series of utterances. An utterance is a set of sentences that a character says uninterrupted. Each utterance is annotated with a label while description of a scene is not annotated. This dataset was reported in Joshi et al. [2016a].

The transcripts of the show were obtained from OpenSubtitles¹⁰ [Lison and Tiedemann, 2016], with additional preprocessing from a fan-contributed website¹¹. Each scene begins with a description of the location/situation followed by a series of utterances spoken by characters. Figure 5.4 shows an illustration of our dataset.

The reason for choosing a TV show transcript as our dataset was to restrict to a small set of characters (so as to leverage speaker-specific features) that use a lot of humor. These characters are often sarcastic towards each other because of their inter-personal relationships. This is in concurrence with past linguistic studies that show how sarcasm is more common between familiar speakers, and often among friends [Gibbs, 2000]. A typical snippet from the

⁹<http://www.imdb.com/title/tt0108778/>

¹⁰<http://www.opensubtitles.org>

¹¹<http://www.friendstranscripts.tk>

Character	% sarcastic
Phoebe	9.70
Joey	11.05
Rachel	9.74
Monica	8.87
Chandler	22.24
Ross	8.42

Table 5.4: Percentage of sarcastic utterances for six lead characters in the Friends dataset

	Surface Positive Sentiment Score	Surface Negative Sentiment Score
Sarcastic	1.55	1.20
Non-sarcastic	0.97	0.75
All	1.03	0.79

Table 5.5: Average surface positive and negative scores for the two classes in the Friends dataset

	Actions (%)
Sarcastic	28.23
Non-sarcastic	23.95
All	24.43

Table 5.6: Percentage of sarcastic and non-sarcastic utterances with actions in the Friends dataset

dataset is:

[Scene: Chandler and Monica’s room. Chandler is packing when Ross knocks on the door and enters...]

Ross: Hey!

Chandler: Hey!

Ross: You guys ready to go?

Chandler: Not quite. Monica’s still at the salon, and I’m just finishing packing.

These transcripts are then annotated by two annotators. Our annotators are linguists with an experience of more than 8,000 hours of annotation. A complete scene is visible to the annotators at a time, so that they understand the entire context of the scene. They perform the

# Utterances	17338
# Scenes	913
Vocabulary	9345 unigrams
Average Length of Utterance	15.08 words
Average Length of Scene	18.6 utterances

Table 5.7: Dataset statistics for the Friends dataset

task of annotating every utterance in the scene with one out of two labels: sarcastic and non-sarcastic. The two annotators separately perform this annotation over multiple sessions. In an attempt to minimize bias beyond the scope of this annotation, we select annotators who had never watched the TV series before this annotation task.

The kappa agreement for a subset of 105 scenes¹² (around 1,600 utterances) is 0.44. This is comparable with other manual sarcasm-annotated datasets. For example, Tsur et al. [2010] report a kappa value of 0.30. Table 5.7 shows statistics of the complete dataset (in addition to 105 scenes as mentioned above). There are 17,338 utterances in 913 scenes. Out of these, 1,888 utterances are labeled as sarcastic. Average length of a scene is 18.6 utterances.

Tables 5.4, 5.5 and 5.6 show additional statistics about the dataset. Table 5.4 shows percentage of sarcastic utterances for the six lead characters in the Friends dataset. Chandler is the character with highest proportion of sarcastic utterances (22.24%). Table 5.5 shows the surface positive and negative scores for the two classes. These ‘surface’ scores are computed using a simple lexicon lookup, by counting the number of positive and negative words. We observe that sarcastic utterances have higher surface positive word score (1.55) than non-sarcastic utterances (0.97) or all utterances (1.03). Finally, Table 5.6 shows the percentage of sarcastic and non-sarcastic utterances that contain actions (indicated in brackets). The proportion of action words shows that sarcastic utterances also have higher proportion of non-verbal indicators (28.23%) than non-sarcastic or complete set of utterances. In rest of the section, we use the following terms:

1. **Utterance:** An utterance is a contiguous set of sentences spoken by an individual without interruption (from another individual). Every utterance has a speaker, and may be characterized by additional information (such as the speaker’s expressions and intonation) in

¹²For these scenes, the annotators later discussed and arrived at a consensus- they were then added to the dataset. The remaining scenes are done by either of the two annotators.

the transcript.

2. **Scene/Sequence:** A scene is a sequence of utterances, in which different speakers take turns to speak. We use the terms ‘*scene*’ and ‘*sequence*’ interchangeably.

Feature	Description
Lexical Features	
Spoken words	Unigrams of spoken words
Conversational Context Features	
Actions	Unigrams of action words
Sentiment Score	Positive & Negative score of utterance
Previous Sentiment Score	Positive & Negative score of previous utterance in the sequence
Speaker Context Features	
Speaker	Speaker of this utterance
Speaker-Listener	Pair of speaker of this utterance and speaker of the previous utterance

Table 5.8: Dataset-Derived features for sarcasm detection of dialogue

5.2.5 Paradigm 1: Traditional Models

We now describe experiments performed in the first paradigm.

Features

The first paradigm uses traditional statistical models that are dependent on proper feature engineering. To ensure that our hypothesis is not dependent on choice of features, we use multiple feature configurations:

1. **Dataset-derived Features:** We design our dataset-derived features based on information available in our dataset. An utterance consists of three parts:
 - (a) **Speaker:** The name of the speaker is the first word of an utterance.

- (b) **Spoken words:** This is the textual portion of what the speaker says.
- (c) **Action words:** Actions that a speaker performs while speaking the utterance are indicated in parentheses. These are useful clues that form additional context. Unlike speaker and spoken words, action words may or may not be present.

Based on this information, we design three sets of features (listed in Table 5.8):

- (a) **Lexical Features:** These are unigrams in the spoken words. We experimented with both count and boolean representations, and the results are comparable. We report values for boolean representation.
- (b) **Conversational Context Features:** In order to capture conversational context, we use three kinds of features. *Action words* are unigrams indicated within parentheses. The intuition is that a character ‘raising eyebrows’ (action) is different from saying “raising eyebrows”. As the next feature, we use *sentiment score* of this utterance. These are two values: positive and negative scores. These scores are the positive and negative words present in an utterance. The third kind of conversational context features is the *sentiment score of the previous utterance*. This captures phenomena such as a negative remark from one character eliciting sarcasm from another.
- (c) **Speaker Context Features:** We use the speaker and the speaker-listener pair as features. The listener is assumed to be the speaker of the previous utterance in the sequence¹³. The speaker feature aims to capture the sarcastic nature of each of these characters, while the speaker-listener feature aims to capture inter-personal interactions between different characters.

2. **Features from Prior Work:** We also compare our results with features presented in two prior works¹⁴:

- (a) **Features given in González-Ibáñez et al. [2011]:** These features are: (a) Interjections, (b) Punctuations, (c) Pragmatic features (where we include action words as well), (d) Sentiment lexicon-based features from LIWC Pennebaker et al. [2001]

¹³The first utterance in a sequence has a null value for previous speaker.

¹⁴The two prior works are chosen based on what information was available in our dataset for the purpose of re-implementation. For example, approaches that use the Twitter profile information or the follower/friends structure in the Twitter, cannot be computed for our dataset.

(where they include counts of linguistic process words, positive/negative emotion words, etc.).

- (b) **Features given in Buschmeier et al. [2014]:** In addition to unigrams, the features are: (a) Hyperbole (captured by three positive or negative words in a row), (b) Quotation marks and ellipsis, (c) Positive/Negative Sentiment Scores followed by punctuation (this includes more than one positive or negative words with an exclamation mark or question mark at the end), (d) Positive/Negative Sentiment Scores followed by ellipsis (this includes more than one positive or negative words with a ‘...’ at the end), (e) Punctuation, (f) Interjections, and (g) Laughter expressions (such as ‘haha’).

Experiment Setup

in the case of traditional models, we experiment with three classification techniques and two sequence labeling techniques:

1. **Classification Techniques:** We use Naïve Bayes and SVM as classification techniques. Naïve Bayes implementation provided in Scikit [Pedregosa et al., 2011], and SVM-Light [Joachims, 1999] are used. Since SVM does not do well for datasets with a large class imbalance Akbani et al. [2004]¹⁵, we use sampling to deal with this skew as done in Kotsiantis et al. [2006]. We experiment with two configurations:
 - SVM (Oversampled) *i.e.*, SVM (O): Sarcastic utterances are duplicated to match the count of non-sarcastic utterances.
 - SVM (Undersampled) *i.e.*, SVM (U): Random non-sarcastic utterances are dropped to match the count of sarcastic utterances.
2. **Sequence Labeling Techniques:** We use SVM-HMM [Joachims, 2008] and SEARN [Daumé III et al., 2009]. SVM-HMM is a sequence labeling algorithm that combines Support Vector Machines and Hidden Markov Models. SEARN is a sequence labeling algorithm that integrates search and learning to solve prediction problems. The implementation of SEARN that we use relies on a perceptron as the base classifier. Daumé III et al. [2009] show that SEARN outperforms other sequence labeling techniques (such as CRF) for tasks like character recognition and named entity class identification.

¹⁵We also observe the same.

We report weighted average values for precision, recall and F-score computed using five-fold cross-validation for all experiments. The folds are created on the basis of sequences and not utterances. This means that a sequence does not get split across different folds.

5.2.6 Paradigm 2: Deep Learning-based Models

We now describe experiments performed in the second paradigm.

Architecture

in the case of the deep learning paradigm, we design deep learning models using three kinds of neural network layers: Convolutional Neural Networks (CNN) [Collobert et al., 2011], Recurrent Neural Networks (RNN) [Graves et al., 2012] and Long Short Term Memory networks (LSTM) [Hochreiter and Schmidhuber, 1997]. Word embeddings are fed into the network as input. We train the model using stochastic gradient descent with the Adadelta update rule [Zeiler, 2012]. When training the model, a portion of the training data is used as a validation set, and the model with the best validation performance is used for testing.

Classification Architecture: Figure 5.5 shows the classification architecture. This architecture is designed to predict sarcasm in an utterance on its own, without using conversational information. Each utterance s_i is represented by concatenated word vectors (w_1 to w_p). This sequence of word vectors is fed to a sentence-level neural network (CNN or LSTM) independently (with no sequence connectivity). Since the data is skewed, we perform *random under-sampling* to bring the sarcastic:non-sarcastic skew ratio to 1:2. However, the under-sampled data (specifically the non-sarcastic utterances) is randomly chosen in each epoch of the training phase so as to not lose any information from the training data. The output of the CNN/LSTM layer is fed to a max-pooling layer, followed by a hidden layer. The output of the architecture is a single prediction for label l_i .

Sequence Labeling Architecture: Figure 5.6 shows the sequence labeling architecture. In this case, the input is a scene. A scene is a set of utterances, s_1 to s_m . This scene is fed to the network to predict a *sequence* of the corresponding sarcastic or non-sarcastic labels, l_1 to l_m .

For individual utterances, CNN and LSTM perform classification by treating the utterance as a sequence of words, where each word is represented by its word embedding. This is the same

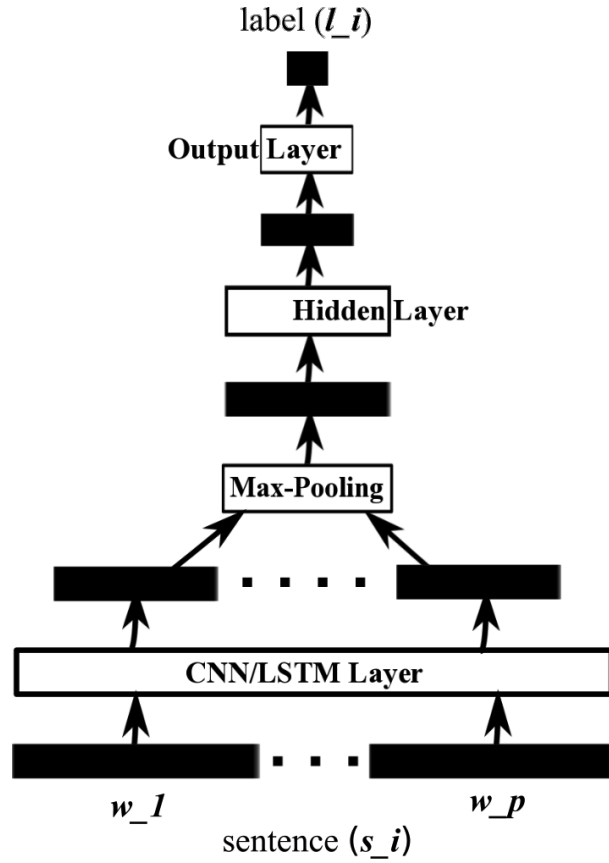


Figure 5.5: Neural network-based architecture for sarcasm detection as a classification task

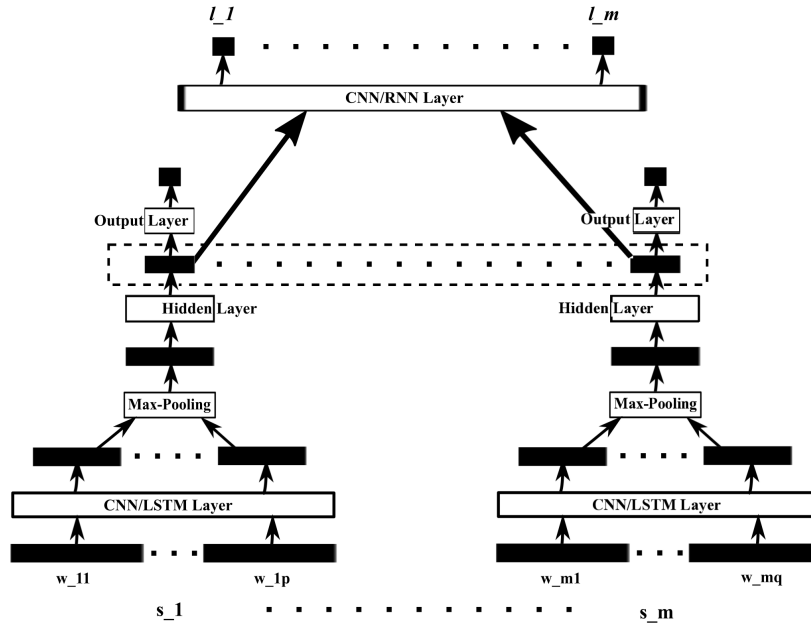


Figure 5.6: Neural network-based architecture for sarcasm detection as a sequence labeling task

as in the case of classification architecture. They differ in the way they incorporate sequences. The prediction for a sequence is made by treating the output of individual classifications as a sequence of vectors. Each unit in this case is represented by a vector obtained from the utterance-level model. Thus, our deep learning models for sequence labeling use two neural network sub-structures:

1. **Utterance-level Network:** As stated above, this network is trained on individual utterances (independent of any sequence). The figure shows that outputs of these classification models is discarded.
2. **Sequence-level Network:** This network utilizes sequence information in the conversation to label utterances. Each utterance (s_i) in the sequence is fed into the utterance-level network (CNN or LSTM) as a concatenation of the corresponding word vectors (w_{i1} to w_{ip}). The outputs of neurons from the *penultimate layer* of this utterance-level network are used as inputs to the sequence-level network (CNN or RNN) to predict the sequence of labels.

In the case of this sequence-based architecture, the training data is used with no under-sampling, because in this case, the utterances are not independent data points, but are a part of sequence which cannot be manipulated any further.

Experiment Setup

We now give details of our experiments for the deep learning paradigm.

Configuration of Classification Architecture: We use the following models for classification:

1. **Long Short Term Memory Models (LSTM):** Word vectors are fed into an LSTM layer with a feature map of size 100, followed by max-pooling layer, whose output is fed into a fully connected neural network with a hidden layer of 50 neurons.
2. **Convolutional Neural Networks (CNN):** Word vectors are fed into a convolutional layer with filter windows of sizes 3, 4, 5 and a feature map of size 100 for each of the filters. The convolutional layer is followed by max-pooling layer, whose output is fed into a fully connected neural network with a hidden layer of 50 neurons.

Configuration of Sequence Labeling Architecture: For the purpose of our experiments, we use CNN and LSTM as alternatives for the utterance-level network, with the configurations same as the classification models. The number of neurons in the penultimate layer of the utterance-level network is 50 which works as a vector representation of the utterance for the sequence-level network. For the sequence-level network, we use CNN (single convolutional layer with filter window of size 3) and RNN (single recurrent layer). This results in four deep learning configurations for sequence labeling. The models are named to identify the two sub-structures as: the first term indicates the type of utterance-level network used while the second term indicates the sequence-level network used. The four configurations are:

1. **LSTM-RNN:** The utterance-level network is a LSTM, while sequence-level network is an RNN.
2. **LSTM-CNN:** The utterance-level network is a LSTM, while sequence-level network is a CNN.
3. **CNN-RNN:** The utterance-level network is a CNN, while sequence-level network is an RNN.
4. **CNN-CNN:** The utterance-level network is a CNN, while sequence-level network is a CNN.

General Configuration Parameters We perform five-fold cross-validation, where the folds are created on the basis of sequences and not utterances. This is to ensure that a sequence never gets split across different folds. The data is not subjected to tuning prior to training/testing, apart from changing all words to lowercase, and inserting a space between letters and punctuation marks.

The word embeddings we use as input to the network are pre-trained vectors trained on a subset of Google News dataset (about 100 billion words)¹⁶. These are 300-dimension vectors for approximately 3 million words and phrases. We choose the activation function of neurons as the sigmoid, the error function as the negative log likelihood, the learning rate as 0.95 and the number of training epochs as 50. The hyperparameters of the neural networks are chosen via a grid-search on the dataset. For our experiments, we randomly choose 10% of the training data as a validation set in each epoch.

¹⁶<https://code.google.com/archive/p/word2vec/>

Algorithm	Precision (%)	Recall (%)	F-Score (%)
Formulation as Classification			
SVM (U)	83.6	48.6	57.2
SVM (O)	84.4	76.8	79.8
Naïve Bayes	77.2	33.8	42
Formulation as Sequence Labeling			
SVM-HMM	83.8	88.2	84.2
SEARN	82.6	83.4	82.8

Table 5.9: Comparison of sequence labeling techniques with classification techniques, for features reported in dataset-derived features

	Sarcastic			Non-Sarcastic		
	Precision	Recall	F-Score	Precision	Recall	F-Score
SVM (U)	14	68.8	23	92.2	46.2	61.6
SVM (O)	22.4	44	29	91.8	81	86
Naive Bayes	9.8	59.8	16.8	85.8	30.6	45
SVM-HMM	35.8	7.6	12.6	89.4	98.2	93.6
SEARN	22.2	19.4	20	90	91.6	90.4

Table 5.10: Class-wise precision/recall values for all techniques using our dataset-derived features

5.2.7 Results

In this section, we evaluate the performance of our models for the two paradigms.

Performance of the Traditional Model Paradigm for Dataset-derived Features

Table 5.9 compares the performance of the two formulations: classification and sequence labeling, for our dataset-derived features. When classification techniques are used, we obtain the best F-score of 79.8% with SVM (O). However, when sequence labeling techniques are used, the best F-score is 84.2%. In terms of F-score, our two sequence labeling techniques perform better than all three classification techniques. The increase in recall is high - the best value

for classification techniques is (SVM (O)) 76.8%, while that for sequence labeling techniques (SVM-HMM) is 88.2%. It must be noted that sentiment of previous utterance is one of the features of both the classification and sequence labeling techniques. Despite that, sequence labeling techniques perform better.

Alg.	Feature Combination (Best)	P (%)	R (%)	F (%)
Formulation as Classification				
SVM (U)	Unigram+ Spkr-Listnr+ Action+ Senti. Score	84.8	49.4	57.4
SVM (O)	Unigram+ Speaker+ Spkr-Listnr+Senti. Score	84	79	81.2
Naïve Bayes	All features	77.2	33.8	42
Formulation as Sequence Labeling				
SVM-HMM	Unigram+ Speaker+ Spkr-Listnr+ Prev. Senti. Score + Action	83.2	87.8	84.4
SEARN	All features	82.6	83.4	82.8

Table 5.11: Feature combinations for which different techniques exhibit their best performance for dataset-derived features

Table 5.10 shows class-wise precision/recall values for these techniques. The best value of precision for sarcastic class is obtained in the case of SVM-HMM, *i.e.*, 35.8%. The best F-score for the sarcastic class is in the case of SVM (O) (29%) whereas that for the non-sarcastic class is in the case of SVM-HMM (93.6%). Tables 5.9 and 5.10 show that it is due to a high recall, sequence labeling techniques perform better than classification techniques.

To validate that the performance gain is because of the formulation and not the feature set, we repeat our experiments with the five algorithms with all possible combinations of features. Table 5.11 shows the best performance obtained by each of the classifiers, and the corresponding (best) feature combinations. The table can be read as: SVM (O) obtains a F-score of 81.2% when spoken words, speaker, speaker-listener and sentiment score are used as features. The table shows that even if we consider the best performance of each of the techniques (with different feature sets), classifiers are not able to perform as well as sequence labeling. The best sequence labeling algorithm (SVM-HMM) gives a F-score of 84.4% while the best classifier (SVM(O)) has an F-score of 81.2%. We emphasize that both SVM-HMM and SEARN have higher recall values than the three classification techniques. These findings show that for our

novel set of dataset-derived features, sequence labeling works better than classification.

Performance of the Traditional Model Paradigm for Features Reported in Prior Work

We now show our evaluation on two sets of features reported in prior work. These sets of features as given in two prior works by Buschmeier et al. [2014] and González-Ibáñez et al. [2011]. Table 5.12 compares classification techniques with sequence labeling techniques for features given in González-Ibáñez et al. [2011]¹⁷. Table 5.13 shows corresponding values for features given in Buschmeier et al. [2014]¹⁸. For features by González-Ibáñez et al. [2011], SVM (O) gives the best F-score for classification techniques (79%), whereas SVM-HMM shows an improvement of 4% over that value. Recall increases by 11.8% when sequence labeling techniques are used instead of classification. In the case of features by Buschmeier et al. [2014], the improvement in performance achieved by using sequence labeling as against classification is 2.8%. The best recall for classification techniques is 77.8% (for SVM (O)). In this case as well, the recall increases by 10% for sequence labeling. These findings show that for two feature sets reported in prior work, sequence labeling works better than classification.

Algorithm	P (%)	R (%)	F (%)
Features from Gonzalez-Ibanez et al. (2011)			
Formulation as Classification			
SVM (U)	86.4	26	27
SVM (O)	84.6	75.6	79
Naive Bayes	77.2	43.8	48.4
Formulation as Sequence Labeling			
SVM-HMM	83.4	87.4	83
SEARN	82	82.4	81.8

Table 5.12: Comparison of sequence labeling techniques with classification techniques, for features reported in Gonzalez-Ibanez et al. (2011)

¹⁷The work reports best accuracy of 65.44% for their dataset. This shows that our implementation is competent.

¹⁸The work reports best F-score of 67.8% for their dataset. This shows that our implementation is competent.

Algorithm	P (%)	R (%)	F (%)
Features from Buschmeier et al. (2014)			
Formulation as Classification			
SVM (U)	85.4	40.6	46.8
SVM (O)	84.6	77.8	80.4
Naïve Bayes	76.6	27.2	32.8
Formulation as Sequence Labeling			
SVM-HMM	84.2	87.8	83.2
SEARN	82.4	83.8	82.4

Table 5.13: Comparison of sequence labeling techniques with classification techniques, for features reported in Buschmeier et al. (2014)

Performance of the Deep Learning Paradigm

Table 5.14 shows the performance of the deep learning-based architectures. The best reported values from the traditional model paradigm are included for comparison. The table shows that all deep learning-based architectures perform better than the approaches based on non-deep learning-based systems, on the same dataset. We observe that the four sequence labeling-based configurations outperform the two classification approaches (LSTM and CNN). LSTM results in a F-score of 84.12%, while CNN-CNN results in a F-score of 86.83%. Table 5.15 shows the class-wise performance of our deep learning models for the two formulations. The highest F-score (28.34%) for sarcastic class is obtained in the case of CNN, while that for non-sarcastic class is in the case of CNN-CNN (93.94%). The precision for sarcastic class is 48.19% for CNN-CNN. These findings show that sequence labeling proves to be a better formulation than classification for sarcasm detection using contextual incongruity in conversational text.

5.2.8 Error Analysis

In previous sections, we show that quantitatively, sequence labeling techniques perform better than classification techniques. In this section, we delve into the question: ‘*What does this improved performance mean, in terms of forms of sarcasm that sequence labeling techniques are able to handle better than classification?*’ To understand the implication of using sequence la-

Models	P (%)	R (%)	F (%)
Our Deep Learning-Based Models			
Classification			
LSTM	83.07	85.2	84.12
CNN	84.21	83.35	83.76
Sequence Labeling			
LSTM-RNN	83.43	86.65	85.01
LSTM-CNN	83.13	86.57	84.82
CNN-RNN	84.88	88.44	86.62
CNN-CNN	85.06	88.68	86.83

Table 5.14: Comparison of classification versus sequence labeling architecture; Best reported values from the traditional model paradigm have been mentioned for comparison

Networks	Sarcastic			Non-Sarcastic			Overall		
	P	R	F	P	R	F	P	R	F
Formulation as Classification									
LSTM	25.68	17.86	20.99	90.18	93.54	91.83	83.07	85.2	84.12
CNN	27.8	30.17	28.34	91.22	89.96	90.55	84.21	83.35	83.76
Formulation as Sequence Labeling									
LSTM-RNN	29.9	15.36	20.23	90.08	95.51	92.71	83.43	86.65	85.01
LSTM-CNN	28.15	13.85	18.52	89.94	95.6	92.68	83.13	86.57	84.82
CNN-RNN	43.21	12.61	18.89	90.03	97.85	93.77	84.88	88.44	86.62
CNN-CNN	48.19	8.32	13.54	89.66	98.66	93.94	85.06	88.68	86.83

Table 5.15: Class-wise performance of classification and sequence labeling as formulations for the deep learning paradigm

beling, we randomly select 100 examples that were correctly labeled by sequence labeling techniques but incorrectly labeled by classification techniques. Our annotators manually annotated them into one among four categories of sarcasm as given in Camp [2012]. Not all sentences could be classified as one of out of the categories. Table 5.16 shows the proportion of these

utterances. Like-prefixed and illocutionary sarcasm types are the ones that require context for understanding sarcasm. We observe that around 71% of our examples belong to these two types of sarcasm. This validates our intuition that sequence labeling will better capture conversational context, in cases of sarcasm for which sequence labeling improves over classification.

On the other hand, examples where our system makes errors can be grouped as:

- **Topic Drift:** Eisterhold et al. [2006] state that topic change/drift is a peculiarity of sarcasm. For example, when Phoebe gets irritated with another character talking for a long time, she says, “*See? Vegetarianism benefits everyone*”.
- **Short expressions:** Short expressions occurring in the context of a conversation may express sarcasm. Expressions such as “*Oh God, is it*” and “*Me too*” were incorrectly classified as non-sarcastic. However, in the context of the scene, these were sarcastic utterances.
- **Dry humor:** In the context of a conversation, sarcasm may be expressed in response to a long serious description. Our system was unable to capture such sarcasm in some cases. When a character gives long description of advantages of a particular piece of clothing, Chandler asks sarcastically, “*Are you aware that you’re still talking?*”.
- **Implications in popular culture:** The utterance “*Ok, I smell smoke. Maybe that’s cause someone’s pants are on fire*” was incorrectly classified by our system. The popular saying ‘Liar, liar, pants on fire’¹⁹ was the context that was missing in our case.
- **Background knowledge:** When a petite girl walks in, Rachel says “*She is so cute! You could fit her right in your little pocket*”.
- **Long-range connection:** In comedy shows like Friends, humor is often created by introducing a concept in the initial part and then repeating it as an impactful, sarcastic remark. For example, in beginning of an episode, Ross says that he has never grabbed a spoon before - and at the end of the episode, he says with a sarcastic tone “*I grabbed a spoon*”.
- **Incongruity with situation in the scenes:** Utterances that were incongruous with non-verbal situations could not be adequately identified. For example, Ross enters an office wearing a piece of steel bandaged to his nose. In response, the receptionist says, “*Oh, that’s attractive*”.

¹⁹<http://www.urbandictionary.com/define.php?term=Liar%20Liar%20Pants%20On%20Fire>

Type	Examples (%)
Propositional	14.28
Embedded	4.08
Illocutionary	40.81
Like-prefixed	31.63
Other	9.18

Table 5.16: Proportion of utterances of different types of sarcasm that were correctly labeled by sequence labeling but incorrectly labeled by classification techniques

- **Sarcasm as a part of a longer sentence:** In several utterances, sarcasm is a subset of a longer sentence, and hence, the non-sarcastic portion may dominate the rest of the sentence.

These errors point to many future directions in which sequence labeling algorithms could be optimized to improve their impact on sarcasm detection.

5.3 Comparative Analysis of Approaches

Section 4.5 compares the approaches described for sarcasm detection using incongruity within the target text. The role of this section is analogous.

In this section, we compare approaches for sarcasm detection described in Chapter 4 with the approach for sarcasm detection in dialogue described in this chapter. Since a dataset that would suit the monologue was unavailable, we restrict ourselves to dialogue.

We report two-fold cross-validation values for different configurations, for the dataset of transcripts from ‘Friends’ as described in Section 5.3. Table 5.3 compares four sets of approaches: (A) Two past baselines, by Liebrecht et al. [2013] and Buschmeier et al. [2014], (B) Approaches that do not use contextual incongruity (described in Chapter 4), where S1 indicates sentiment incongruity as features, S2 indicates semantic incongruity as features, and S3 indicates language model incongruity, (C) Combination of approaches in (B), where ALL indicates S1 + S2 + S3, and (D) Conversational context incongruity where sequence labeling algorithms are used to detect sarcasm in sequences.

Note: In case of S5, the dataset is split in two folds while retaining the sequences as it is.

	P	R	F
(A) Baseline			
Buschmeier et al. [2014]	57.75	30.37	37.55
Liebrecht et al. [2013]	58.73	49.66	53.59
(B) Approaches that use only target text			
S1: Sentiment Incongruity	56.81	25.91	33.14
S2: Semantic Incongruity	57.84	60.47	55.75
S3: Language Model Incongruity	80.5	86.5	83
(C) Combinations of approaches in (B)			
ALL	72.57	52.34	52.29
ALL - S1	74.41	59.87	63.21
ALL - S2	71.32	55.39	50.87
ALL - S3	74.42	63.04	61.74
(D) Conversational contextual incongruity			
S5	63.75	50.25	56.19

Table 5.17: Comparison of individual approaches, ensembles and sequence labeling-based approach for a dataset of conversational transcripts

In other cases, the two folds divide the dataset into two parts without the sequence consideration. Therefore, it is not possible to compare ensemble-based approach with S5 combined.

We observe that the best performance for this dataset is observed in case of S3 where language model incongruity-based features are used for sarcasm detection. The best performance of (B) is obtained for ‘*ALL-SI*’ where the F-score is 63.21%. However, contextual incongruity in dialogue results in F-score of 56.19%. This is above both the baselines but not above some approaches in (B) and (C).

Therefore, for a conversational dataset, either language model incongruity or a combination of semantic and language model incongruity seem to be the best combination.

5.4 Summary

This chapter described two approaches that use contextual incongruity as an indicator for sarcasm detection. We show how contextual incongruity can be captured in two situations: a monologue (where context from an author is used to determine sarcasm in the text generated by the author), and a dialogue (where context from a conversation is used to determine sarcasm in a text that is a part of the conversation).

For sarcasm detection in monologue, we presented an approach that predicts sarcasm in a target tweet, using information from the author’s timeline. Using the tweet author’s historical tweets, our system detected context incongruity with the historical sentiment by checking if the sentiment towards a given target phrase in the target tweet agrees to the sentiment expressed in the historical tweets by the same author. We implemented four kinds of integrators to combine the contrast-based predictor (which works on the content of the target tweet alone) and the historical tweet-based predictor (which uses the target tweet as well as the historical tweets). We obtained the best F-score of 0.882, in the case of the Relaxed-AND configuration of the integrator.

For sarcasm detection in dialogue, we described how contextual incongruity can be captured using sequence labeling-based approaches. We formulated sarcasm detection of dialogue as a task of labeling each utterance in a sequence, with one among two labels: sarcastic and non-sarcastic. We performed our evaluation for two paradigms: the traditional (statistical) models, and deep learning-based models. In terms of the traditional models, we experiment with: (a) a

novel set of features derived from our dataset, (b) sets of features from two prior works. Using these features, we compared two classes of learning techniques: classifiers (SVM Undersampled, SVM Oversampled, and Naïve Bayes) and sequence labeling techniques (SVM-HMM and SEARN). We observed an improvement of 2.8% for features in Buschmeier et al. [2014] and 4% for features in González-Ibáñez et al. [2011] when sequence labeling techniques were used as against classifiers. in the case of the deep learning paradigm, we observed that all four sequence labeling models (LSTM-RNN, LSTM-CNN, CNN-RNN and CNN-CNN) outperformed both the classification models (LSTM and CNN). The improvement in F-score was 3.7% when sequence labeling models are used instead of classification models. The best F-score in the case of sequence labeling techniques (CNN-CNN) is 86.83%. We observed an improvement of 3.4% in the recall. To understand which forms of sarcasm get correctly labeled by sequence labeling (and not by classification), we manually evaluated 100 examples. 71% consisted of sarcasm that could be understood only with conversational context. These observations establish the efficacy of sequence labeling techniques to capture context incongruity, for sarcasm detection of dialogue.

Because of unavailability of a sarcasm-labeled dataset that would suit both the situations: monologue and dialogue, the two are not comparable in the current situation. However, we compare our approach for context incongruity in dialogue with two works. We obtain an improvement as compared to the two past works. However, language model incongruity and a combination of language model and semantic incongruity performs better than sequence labeling-based approach to capture contextual incongruity in dialogue.

Chapter 6

Sarcasm Generation

In the previous chapters, we described our studies to understand the phenomenon of sarcasm, followed by a set of approaches to detect sarcasm. In this chapter, we describe our efforts in a new problem in computational sarcasm: **sarcasm generation**. We define sarcasm generation as the computational task of generating sarcastic sentences, in response to an input sentence. In this chapter, we present a natural language generation system that synthesizes incongruity as seen in sarcastic statements. The work described in this chapter was presented at WISDOM (Workshop on Issues of Sentiment Discovery and Opinion Mining) 2015 workshop held at SIGKDD 2015 ¹².

Our sarcasm generation system is called ‘**SarcasmBot**’. The input to SarcasmBot is a well-formed English sentence while the output is an English sentence which is a sarcastic response to the input. In the rest of this chapter, we refer to the input as ‘*user input*’ and the output as ‘*sarcastic response*’. We employ a pattern-based approach used in past chatbots [Pilato et al., 2008]. To the best of our knowledge, ours is the first work in sarcasm generation.

The rest of the chapter is organized as follows. We present the motivation for SarcasmBot in Section 6.1. The architecture of SarcasmBot is described in detail in Section 6.2. Section 6.3 discusses the evaluation, while Section 6.4 summarizes the chapter.

¹<http://sentiment.net/wisdom2015joshi.pdf>

²The source code is at: <https://github.com/adityajo/sarcasmbot>

6.1 Motivation

Research in natural language generation dates back to several decades [Wallace, 2009, Bobrow et al., 1977, Litman and Pan, 2002, Stent et al., 1999]. Chatbots have been used in a variety of roles such as information agents [Litman and Pan, 2002], instructing agents for autonomous vehicles [Stent et al., 1999], and automatic tutoring agents [Litman and Silliman, 2004].

Humor generating chatbots have also been developed [Pilato et al., 2008]. This sarcasm generation system is of a similar nature. However, in addition to its entertainment value, the motivation behind SarcasmBot lies in the idea of ‘*imitating humans*’ as given in the Turing Test [Turing, 1956]. The Turing test states that an artificially intelligent agent must be able to imitate humans. In such a case, in order to imitate humans successfully, a chatbot must be able to express sarcasm as effectively as humans do. Therefore, for sarcasm generation, we wish to address the question:

Can incongruity, as expressed in the case of sarcasm, be incorporated in a natural language generation system?

A chatbot called EHeBBy generates humor based on a set of known humorous patterns [Pilato et al., 2008]. In case of SarcasmBot, we use a similar template-based approach to generate sarcasm. The novelty of SarcasmBot is that it systematically ‘*algorithmizes*’ different forms of sarcastic incongruities, as described in Campbell and Katz [2012].

6.2 Architecture

Figure 6.1 shows the architecture of SarcasmBot. The architecture consists of three modules: Input Analyzer, Generator Selector, and Sarcasm Generator. The user input is first analyzed using the input analyzer. The input analyzer extracts properties of the input such as entities, tense, number, type of question, etc. These properties are then given to the generator selector. The generator selector chooses among eight modules of sarcasm generators, depending on the characteristics of the user input, such as the type of question (in case of a question), the sentiment, the presence of named entities, etc. Each module in the sarcasm generator then generates incongruity, in order to produce a sarcastic response. All the three modules are rule-based, and rely on lexicons and templates. In the forthcoming subsections, we describe these modules in detail. The input analyzer is described in Section 6.2.1, the generator selector in Section 6.2.2, and the generators in Section 6.2.3.

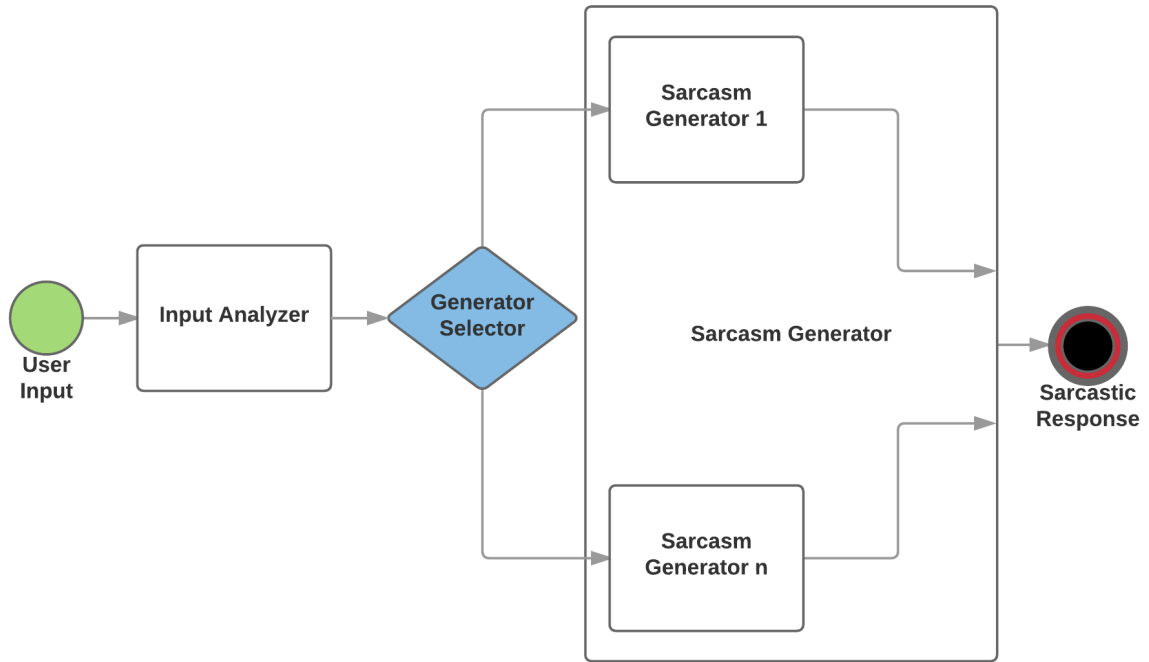


Figure 6.1: Architecture of SarcasmBot

6.2.1 Input Analyzer

The input analyzer extracts the following information from the user input:

1. **POS sequence:** We tag the user input with part-of-speech (POS) tags using a POS tagger.
2. **Question Type Determiner:** If the user input is a question (indicated by a question mark at the end of the sentence), we determine the type of question among: *Why, what, where, how, who, when and choice*.
3. **Tense:** We determine the tense of the input by looking for verb-related POS tags. The candidate tenses are: *Third person present, Non-third person present, Past and Modal* [Marcus et al., 1993].
4. **Entities:** Using the POS sequence, we create two lists of entities: noun entities (sequence of NNs) and named entities (sequence of NNPs), along with the number (singular/plural) of each entity (indicated by separate tags, NNS and NNPS).
5. **Pronouns:** Using the POS sequence, we identify pronouns in the user input. Additionally, we use a list of pronoun-pronoun pairs that correspond to question and response (example:

‘You’ must be responded with ‘I’).

6. **Sentiment:** Since sarcasm is associated with sentiment, we implement a rule-based sentiment determiner that uses the sentiment lexicon by McAuley and Leskovec [2013b], along with rules that process negations and conjunctions. For example, we reverse the polarity of adjectives that follow a negation word upto a certain window size. The sentiment determiner counts the number of positive and negative words, after application of these rules based on negations and conjunctions.
7. **Offensive language:** Sarcasm is often used as a retort for offensive language. Hence, we download a list of offensive words from LIWC³.

6.2.2 Generator Selector

Based on the properties of the user input, the Generator Selector decides which of the eight sarcasm generators is to be used to generate the sarcastic response. The algorithm to select a sarcasm generator is as follows:

1. If the input contains **an offensive word**, the *Offensive word response generator* is invoked.
2. If the input is **not a question but carries sentiment**, the *Sentiment-based sarcasm generator* is invoked.
3. If the input is **a yes/no question**, we count the number of entities in the input. We then choose among the following:
 - (a) If there is **more than one entity**, the *Opposite Polarity Verb-Situation Generator* is invoked.
 - (b) If the sentence contains **modal verbs**, the *Irrealis Sarcasm Generator* is called.
 - (c) Else, the *Hyperbole Generator* is invoked.
4. If the input is **an opinion question**, one out of these sarcasm generators is randomly invoked: *Opposite Polarity Verb-Situation generator*, *Opposite polarity person-attribute generator* or *Hyperbole generator*.

³Linguistic Inquiry and Word Count: <http://liwc.wpengine.com/>

5. If the input is a **‘why’ question and has non-zero entities**, the *Incongruous reason generator* is invoked.
6. Finally, the *random response generator* is invoked if **none of the above** applies.

Each of the above generators handles a peculiar form of sarcastic expression. The individual sarcasm generators are described in the next subsection.

Sarcasm Generator	Description
(a) Offensive Word Response Generator	In case an offensive word is used in user input, select a placeholder from a set of responses.
(b) Opposite Polarity Verb-Situation Generator	Randomly select a verb. Compute its sentiment. Discover a situation which is opposite in sentiment.
(c) Opposite Polarity Person-Attribute Generator	Randomly select a named entity. Select incongruent pairs of famous people.
(d) Irrealis Sarcasm Generator	Create a hypothetical situation that is impossible by selecting from a set of undesirable situations.
(e) Hyperbole Generator	Select a noun phrase in the user input. Generate a hyperbole with a ‘best ever’ style regular expression.
(f) Incongruent Reason Generator	Select an unrelated reason as a response for a user input.
(g) Sentiment-based Sarcasm Generator	Compute sentiment of user input. Generate a response opposite in sentiment.
(h) Random Response Generator	Select one positive exclamation and one negative exclamation randomly from a set of exclamations. Place them together.

Table 6.1: Sarcasm generator modules in SarcasmBot

6.2.3 Sarcasm Generator

Table 6.1 lists our set of sarcasm generators. The basis of all sarcasm generators is as follows: (a) select a sarcastic template, and (b) insert appropriate words from the user input and a knowledge base, into the placeholders. Both (a) and (b) are randomly selected. This random se-

lection imparts dynamic nature to the sarcastic response. Our sarcasm generators are described as follows:

1. **Offensive Word Response Generator:** This generator picks randomly from a list of responses that praise the use of the offensive word, and inserts the offensive word at the appropriate place. An example of a response is ‘*Wow! I can’t believe you just said ‘b****’. You really are very classy!’*’.
2. **Opposite Polarity Verb-Situation Generator:** This generator creates a response as per the following pattern:

(Optional Filler) (Pronoun-Response) (used to/will) (Verb of Polarity X) (Optional intensifying adverb) (Situation of Polarity Y).

The *optional filler* is picked randomly from a set of filler words like ‘*Hmm..*’. The *verb of polarity X* is selected from the sentence. Finally, a random *situation of polarity Y* (*Y is opposite of X*) is selected from a set of situations. This set of situations was extracted as follows: We first downloaded tweets marked with hashtag #sarcasm, using Twitter API⁴. Then, we manually selected a set of noun phrases that correspond to situations. An example of a response generated by Opposite polarity verb-situation generator is ‘*Hmm, well.. I used to like Tim, just the way I like being stuck in the elevator all night*’ to the question ‘*Do you like Tim?*’. The (negative) situation in this case is ‘*being stuck in the elevator all night*’.

3. **Opposite Polarity Person-Attribute Generator:** This generator expresses sarcasm towards a named entity in the input. It uses the sarcastic pattern:

I think that (Named entity) is (Popular Person) (plus/minus) (Positive/Negative Attribute).

The generator first picks a *named entity* from the input. Then, it randomly selects a *popular person* and an *attribute*. If it is a desirable attribute, the generator selects a *plus* else, a *minus*. We have compiled a list of popular actors. For the positive attributes, we use a list of positive traits for an actor, and negative traits for negative attributes.

⁴<http://dev.twitter.com/overview/api>

An example of a response generated by Opposite polarity person-attribute generator is ‘*I think that Jim is Tom Cruise minus the talent*’.

4. **Irrealis Sarcasm Generator:** Irrealis mood⁵ corresponds to sentences like ‘*I would have loved to watch this movie if I did not have absolutely anything better to do*’. The Irrealis sarcasm generator selects a verb from the input and selects from a set of hypothetical negative situations. The pattern then used for the response is:

(Pronoun-response) would (verb) (Hypothetical bad situation)

The verb and pronoun-response are selected based on the input. An example response generated by Irrealis sarcasm generator is ‘*He will marry only if badly drunk*’ to the question ‘*Will he marry me?*’.

5. **Hyperbole Generator** To create hyperbole-based sarcasm, we select a noun entity from the user input and use sarcastic patterns of the following type:

‘(Those were/That is) the best (Noun Entity) in the history of humankind!’

The choice between ‘*those were*’ and ‘*that is*’ is made using the cardinality of the noun entity. Note that the above pattern is one among many such hyperbolic patterns. An example of response generated by Hyperbole generator is ‘*Those were the best immigration reforms in the history of humankind*’ to the input ‘*What do you think of the immigration reforms announced by the Parliament this Saturday?*’.

6. **Incongruous Reason Generator:** The Incongruous reason generator is selected in case of ‘*why*’ questions. A named entity is randomly selected from the input and an incongruous reason is picked out of random unrelated reasons. One sarcastic pattern in this case is:

Because (Named entity/Pronoun-Response) (Incongruous reason)

An example response to the question ‘*Why did Jim miss his date tonight?*’ is ‘*Because Jim was in the mood for adventure.*’

7. **Sentiment-based Sarcasm Generator:** The Sentiment-based sarcasm generator aims to counter the sentiment expressed in the user input. We use a set of positive and negative

⁵http://en.wikipedia.org/wiki/Irrealis_mood

expressions. In case the user input expresses sentiment and is not asking a question, a random response from the opposite polarity is selected. An example of response generated by the random sarcasm generator is ‘*Poor you!*’ in response to a positive input such as ‘*I am excited about my new job!*’.

8. **Random Sarcasm Generator:** In case the user input does not express sentiment and is not asking a question, the Random sarcasm generator is employed. It randomly picks a phrase each from a set of positive and negative reactions, and concatenates them together. An example of response generated by the random sarcasm generator is ‘*Wow! (positive reaction) *rolls eyes* (negative reaction)*’.

(a) **User input:** Why did Greg go home?

Sarcastic response: Because Greg was in a mood for adventure.

(b) **User input:** What do you think of Greg?

Sarcastic response: Well..I like Greg. The way I absolutely appreciate Zero accountability people.

(c) **User input:** Stop being such a b*****!

Sarcastic response: Unbelievable that you just said ‘b*****!’! You are really very classy!

(d) **User input:** What do you think of the new video game that James purchased?

Sarcastic response: That was the best video game in the history of humankind!

(e) **User input:** Did James meet you?

Sarcastic response: Hmm..I used to meet James if I felt stupid.

(f) **User input:** I won a lottery!

Sarcastic response: I am so sorry for you!

Table 6.2: Example outputs of SarcasmBot

6.3 Evaluation

Table 6.2 shows some example output from SarcasmBot. The examples have been selected such that they demonstrate the ability of different sarcasm generators as described in the previous section. Automatic evaluation of SarcasmBot was difficult for a variety of reasons, one of them

being that there are multiple possible responses. Therefore, we evaluate SarcasmBot through two experiments.

6.3.1 Experiment Details

The two experiments used for evaluation of SarcasmBot are as follows:

1. **Experiment 1: Stand-alone evaluation:** This evaluation is stand-alone because the output of SarcasmBot is evaluated on its own. For a set of 31 user inputs, we obtain the sarcastic response from SarcasmBot. Three evaluators answer the following question for each output:

- (a) Coherence: Is the output a suitable response to the user input?
- (b) Grammatical correctness: Is the output grammatically correct?
- (c) Sarcastic nature: Is the output sarcastic?

All questions are answered with yes/no values. These values numerically correspond to 1/0 respectively.

2. **Experiment 2: Comparative evaluation:** For the same set of 31 user inputs, we obtain the output from SarcasmBot, as well as ALICE [Wallace, 2009]. Four evaluators participate in this experiment. An evaluator is shown the input statement, and two outputs. Their task is to identify which of the two outputs is from SarcasmBot. Specifically, they answer two questions:

- (a) Which of the two outputs is from SarcasmBot?
- (b) Was it difficult to make this judgment?

These questions are answered with yes/no values as well. As in the previous case, these values numerically correspond to 1/0 respectively.

All seven evaluators are engineering graduates and have studied for a minimum of 10 years with English as the primary language of academic instruction. No evaluator has any information about the internals of SarcasmBot. The evaluators for Experiments 1 and 2 are different.

Evaluation Parameter	Fleiss' Kappa
Coherence	0.164
Grammatical correctness	0.015
Sarcasm	0.335

Table 6.3: Multi-annotator agreement statistics for Experiment 1

6.3.2 Results

Fleiss' Kappa measures multi-annotator agreement [Fleiss, 1971]. Its values for Experiment 1 is shown in Table 6.3. The value for sarcasm is 0.335 indicating moderate correlation. As stated above, all evaluators assigned 0 (No) or 1 (Yes) to each input-output pair. The average values for the three evaluation parameters are shown in Table 6.4. The average value for grammatical correctness is high but the corresponding Fleiss Kappa in Table 6.3 is low (*i.e.*, 0.015). This may be due to low variance because of which the probability of chance agreement increases, relative to actual agreement. As seen in Table 6.4, the evaluators assign an average value of 0.806 for sarcasm. All values are greater than 0.69, indicating that the system performs well in terms of these three parameters.

Evaluation Parameter	Average
Coherence	0.698
Grammatical correctness	0.903
Sarcastic nature	0.806

Table 6.4: Evaluation of SarcasmBot: Results of Experiment 1

Evaluation Parameter	Fleiss' Kappa
Identification of SarcasmBot output	0.476
Difficulty	0.164

Table 6.5: Evaluation of SarcasmBot: Results of Experiment 2

The Fleiss' Kappa values for Experiment 2 are shown in Table 6.5. The value for identification of SarcasmBot output is 0.476, indicating a moderate agreement. The corresponding value for difficulty is 0.164 because each evaluator may have their own perception of difficulty.

Strategy	Accuracy (%)
At least one evaluator is correct	87.09
Majority evaluators are correct	70.97
All evaluators are correct	61.29

Table 6.6: Percentage of statements in which the evaluators are able to correctly identify SarcasmBot output, in case of Experiment 2

Using the annotations obtained in Experiment 2, we now check how many evaluators correctly identify the outputs generated by SarcasmBot. These results are shown in Table 6.6. We consider three evaluation strategies: (a) at least one evaluator correctly identifies the output, (b) majority of the evaluators correctly identify the output, and (c) all evaluators correctly identify the output. We observe that all evaluators correctly identify the output of SarcasmBot in 61.3% sentences, indicating that SarcasmBot generates a reasonably sarcastic response. In case of 71% sentences, the majority of evaluators are correct.

6.4 Summary

In this chapter, we presented a sarcasm generation system called SarcasmBot. SarcasmBot has three modules: Input Analyzer, Generator Selector and Sarcasm Generators. SarcasmBot synthesizes peculiar forms of sarcastic expressions such as hyperbole, incongruity and irrealis, using eight rule-based sarcasm generators. We evaluated SarcasmBot through two manual experiments. Our evaluators assigned average scores greater than 0.69 (out of 1) for the three quality parameters: coherence, grammatical correctness and sarcastic nature. In addition, we conducted a comparative evaluation where, in case of 71% of test cases, the majority of our evaluators were able to correctly identify the output of SarcasmBot out of the two candidate outputs: one from SarcasmBot, and the other from ALICE.

Chapter 7

Conclusion & Future Work

This chapter concludes the thesis. We first summarize previous chapters, draw conclusions and describe possible future directions.

7.1 Summary

Sarcasm is a form of verbal irony that is intended to express contempt or ridicule. This thesis described our work in computational sarcasm. We first presented a literature survey that captures trends in datasets, approaches and issues related to sarcasm detection. We observed that feature engineering-based approaches are more common than rule-based and deep learning-based approaches for sarcasm detection. In addition, we described three trends in past research in sarcasm detection: (a) the discovery of semi-supervised patterns, (b) the creation of large-scale datasets using distant supervision, and (c) the use of contextual information to detect sarcasm. The literature survey led us to the notion of incongruity. Incongruity is a situation where components of a text are incompatible either with each other or with some background knowledge. Based on this notion of incongruity, our investigations in computational sarcasm were divided into four parts: understanding the phenomenon of sarcasm, detection of sarcasm using incongruity within the text to be classified, detection of sarcasm using incongruity outside the text to be classified, and generation of sarcasm by synthesizing incongruity in response to a user input. We summarize these parts as follows:

1. **Understanding the phenomenon of sarcasm:** To understand components of sarcasm, we conducted the following studies:

- (a) In the first study, we investigated the role of implied negative sentiment in sarcasm. We considered a case where cultural differences make it difficult for annotators (performing the task of sarcasm annotation) to understand sarcasm. Here, we used datasets annotated by American annotators, and obtained new annotations for these datasets from non-native (specifically, Indian) annotators. Our observations had that Indian annotators showed higher agreement with each other than with their American counterparts. However, the use of annotations by Indian annotators did not result in a significant degradation in the performance of sarcasm classification.
- (b) In the second study, we analyze the role of the target of ridicule in sarcasm. Therefore, we studied challenges in sarcasm-versus-irony classification (because sarcasm requires a target of ridicule while irony does not). We showed that human annotators find it difficult to distinguish between sarcasm and irony (which are closely related labels) as compared to sarcasm and philosophy (which are not as closely associated as sarcasm and irony). This was also true in case of a statistical classifier.
- (c) In the third study, we explore if the target of ridicule (which is a crucial component as observed in the second study) may be extracted from a sarcastic text. Towards this, we presented a hybrid approach to detect targets of sarcasm. A sarcasm target is defined to be either a subset of words in the text or the fall-back label ‘Outside’. Our approach uses a set of rules, and a classifier based on features in order to detect these sarcasm targets. We evaluated our approach on two manually annotated datasets: book snippets and tweets. Our approach outperformed two baselines, one of which was based on past work in sentiment target identification, a task closely related to sarcasm target identification.

2. **Sarcasm detection using incongruity within the text to be classified:** We presented four approaches that detect sarcasm using information from within the text to be classified:

- (a) As the first approach, we proposed a set of **features based on sentiment incongruity**. Our set of sentiment incongruity-based features captured two forms of sentiment incongruity: explicit and implicit. In case of explicit incongruity, we used features from thwarting detection, while in case of implicit incongruity, we used features based on phrases bearing implicit sentiment. We tested our classifiers on

datasets of tweets and discussion forum posts. We observed an improvement on the two datasets over (a) a rule-based baseline, and (b) two past works.

- (b) As the second approach to capture incongruity, we presented **features based on semantic incongruity**. Our features were scores based on the distance between embeddings of words in a sentence as an indicator of the incongruity therein. We showed that these features when appended to four prior works resulted in an improvement in sarcasm detection. This improvement was observed for four kinds of word embeddings. Specifically, we showed that Word2Vec and dependency weight-based embeddings proved to be more beneficial as compared to other representations.
- (c) As the third approach, we presented a **topic model to detect sentiment incongruity** for sarcasm detection. Based on sentiment mixtures in sarcastic text, the model discovered sarcasm-prevalent topics from a dataset of tweets. The model was then used for sarcasm detection. We observed an improvement in positive F-score as compared with two past works.
- (d) The fourth approach used **language model incongruity**. Using context2vec, a sentence completion toolkit, we obtained the expected word at a given position in a sentence, and computed the similarity between the observed word at that position and the expected word. The hypothesis was that in a sarcastic sentence, a word present at a position is dissimilar to the word that was expected by the language model (trained on non-sarcastic text). We presented two approaches: an all-words approach, and an incongruous words-only approach, which differed in terms of the positions that they consider to invoke the sentence completion toolkit. In case of short text (tweets), our approach (80.24%) outperformed reported work (61%). We also observed that an oracle approach (when the exact incongruous word is known) outperforms the all-words approach.

3. **Sarcasm detection using context incongruity:** In many cases, sarcasm may not be understood on the basis of the sentence alone, but may require additional context. Hence, we described two approaches that require contextual incongruity:

- (a) We presented a rule-based system that uses a creator's **historical context** for monologue text. This means that to classify a tweet as sarcastic or not, we use past tweets

by the same author to provide the required context.

- (b) We then presented a sarcasm detection system that uses **conversational context for dialogue-based** text. We create a manually labeled dataset of transcripts from the TV show ‘Friends’. We showed that sequence labeling algorithms with appropriate features outperform classification algorithms, for sarcasm detection of dialogue. We validated the benefit of sequence labeling for two paradigms: traditional models (such as SVM-HMM, SVM, etc.) and deep learning models (based on LSTM and CNN).

- 4. **Sarcasm generation:** We implemented a sarcasm generation system that responds sarcastically to the user input. The system is based on eight sarcasm generators, each of which simulates a particular form of incongruity in sarcastic expression. Our evaluation showed that human annotators agreed that the chatbot was indeed producing sarcastic output. When presented with two outputs, one from our sarcasm generation system and another from a general purpose chatbot, annotators were able to identify the output from our chatbot in majority of our test cases.

7.2 Conclusion

Sarcasm is a distinctive form of sentiment expression that is likely to mislead sentiment analysis, as shown in Chapter 1. In this work, we focus on the notion of incongruity which is central to sarcasm. Hence, the recurring goal of the work in this thesis is to model different forms of incongruity for sarcasm detection and generation.

Our studies related to sarcasm-annotated datasets present novel insights. We validate past work in linguistics when we observe that sarcasm annotation from non-native annotators does not match that of native annotators. However, we extend this a step further and look at the impact of training sarcasm detection systems using non-native annotations. The degradation in sarcasm classification is not statistically significant implying that such datasets may be useful for sarcasm classification. This observation is particularly useful since when new datasets are introduced, the quality may be questioned based on non-native cultural background of annotators. In addition, our insights on difficulties faced by non-native annotators, is useful for careful selection of what sentences can be annotated by non-native annotators. Similarly, our study on sarcasm-versus-irony classification is novel in the context that it looks at both human and

computational perspectives. The study shows that sarcasm-versus-irony classification would be more challenging as compared to sarcasm classification (where the labels are sarcastic and non-sarcastic). Our observations validate that features designed for sarcasm versus non-sarcasm may not be adequate for sarcasm versus irony because the labels are closely related. Finally, to identify the target of sarcasm, we presented a hybrid approach that combines a rule-based system and a statistical system. We see that the hybrid approach allows us to hand-code known patterns in the rule-based module and learn additional patterns in the statistical module. We identify that a peculiar case of sarcasm target where the target is not present as a phrase in the sentence results in low performance.

In case of our approaches on sarcasm detection, we proceed in a bottom-up manner. We begin with sentiment incongruity which is the property of a word, and move right up to contextual incongruity derived from the conversational context in a dialogue. Across this spectrum, we explore multiple approaches to detect sarcasm. The first approach modeled two degrees of sentiment incongruity using features. Obtaining implied sentiment was a challenge faced by this approach. As a result, the second approach attempted to tackle situations where sentiment words may not be present but incongruity may occur because of semantically dissimilar words. We quantified this semantic incongruity using word embedding-based features. As an alternative mechanism to detect sarcasm using sentiment and topic mixtures, the third approach presented a topic model for sarcasm detection. The model used sentiment and topic mixture in tweets in order to discover sarcasm-prevalent topics. Topics corresponding to alternate sentiment, and prevalence of a topic to belong to the sarcastic/non-sarcastic class collectively helped to understand issues that evoke sarcasm in our dataset. This work, being an initial work in topic models for sarcasm, sets up the promise of topic models for sarcasm detection, as also demonstrated in corresponding work in aspect-based sentiment analysis. The fourth approach moved to the language model level, while considering interaction between words. Using a sentence completion library, we obtained the most likely word at a particular position, and compare it with the word that actually occurs at the position. A caveat of this approach lied in identifying the accurate position where the incongruity occurs. These four approaches attempted to capture sarcasm using different techniques (classification/topic models, etc.) and indicators (sentiment incongruity/sentence completion, etc.). However, they are limited in their ability to capture more difficult forms of sarcasm.

For example, context becomes necessary for several forms of sarcasm. This may include

hyperbolic sarcasm or sarcasm as a part of a conversation. This is where our approaches using contextual incongruity play an important role. Our first approach used an author’s historical tweets to set up a sentiment map, and then determine sarcasm in a tweet based on the entities mentioned in the tweet. However, such a sentiment map is limited by the author’s presence on the social media. Therefore, for a new author or an author who has not mentioned an entity in the past, our approach falls short. Our last approach to detect sarcasm incorporated conversational context to detect sarcasm in dialogue using sequence labeling-based algorithms. In this case, because dialogue may contain short text, it was useful to include information in the form of sentiment values, speakers, etc. to detect sarcasm. In the case of dialogue as well, knowing the identity of speaker and listener was beneficial because it captures a speaker’s tendency to be sarcastic and the listener’s propensity to be subjected to sarcasm by the speaker. We showed that this benefit of sequence labeling models holds in case of both traditional learning techniques (SVM, etc.) and deep learning-based techniques.

A comparative analysis of these individual approaches on three datasets: book snippets, tweets and conversation transcripts results in the following:

1. For informal text, sentiment-based incongruity is a useful technique for sarcasm detection.
2. For formal text, a combination of semantic-based incongruity and language model-based incongruity results in the best performance.
3. For short conversational text (with dependencies between utterances of the conversation), a combination of language model incongruity and semantic incongruity result in the best performance - although the use of sequence labeling algorithms outperforms two past works.

To generate sarcasm, our chatbot module used eight templates of sarcastic expression. The focus was on: (a) being coherent to the input (where we use elements from the input), and (b) being sarcastic (where we use elements from a knowledge base along with known patterns). While our chatbot does not use conversational context beyond using entities in the user input, the ability of the template-based approach was validated by our three human annotators.

In conclusion, our work addresses multiple problems of computational sarcasm, where each approach is motivated by the notion of incongruity. Incongruity for sarcasm detection is captured using intra-textual information such as sentiment and word embeddings, and using

contextual information in the form of a tweet creator’s historical context in a monologue and the conversational context in a dialogue. Similarly, in case of sarcasm generation, templates are used to simulate incongruity. Through each of our approaches for different sarcasm-related problems, this thesis highlights the difficulty of setting up computational approaches for sarcasm.

7.3 Future Work

Our investigations in computational sarcasm point to several directions for future work:

Short-term Future Work: These work items could make incremental additions to the findings of this thesis:

- Our word embedding-based features may work better if the similarity scores are computed for a subset of words in the sentence, or using weighting based on syntactic distance instead of linear distance.
- Integration of historical text-based features into a supervised sarcasm detection framework is a promising future work.
- Based on our experiments in conversational context incongruity, additional work on repeating these experiments for other forms of dialogue (such as twitter conversations, chat transcripts, etc.) is imperative.
- Our findings also show the viability of using sentence completion to detect sarcasm. Our work can be extended to a set of features for a statistical classifier.

Mid-term Future Work:

- A future direction is to use discourse connectors for computational sarcasm. This will help sarcasm target identification, so that, instead of pronominal references, the noun phrases that they point to are retrieved.
- A combination of unified sarcasm and emotion detection using sequence labeling is another promising line of work.
- We show how cultural differences impact sarcasm annotation. A more comprehensive study on larger sets of annotators and a larger spectrum of nationalities would strengthen the observations.

- Our sarcasm topic model is currently limited to sentiment and topic mixtures in sarcastic text. Instances such as hyperbolic sarcasm do not fit this intuition. The current approach relies only on bag-of-words which may be extended to n-grams since a lot of sarcasm is expressed through phrases with implied sentiment. These shortcomings in the topic model need to be addressed.

Long-term Future Work: This work spawns the following long-term directions:

- Our work opens several new problems in computational sarcasm such as sarcasm target identification, sarcasm generation, and irony versus sarcasm classification. Each of these problems has unique challenges and potential applications, as presented in this thesis.
- Our error analysis points to interesting directions in sarcasm detection. These include: (a) determining the role of numbers for signaling sarcasm, and (b) handling situations with subjective sentiment, (c) detecting metaphorical language for sarcasm, (d) identifying incongruity in case of multiple senses of words, and (e) detecting long distance incongruity in text. These are challenging tasks because they require a far deeper understanding of context.
- Our work on sarcasm target identification forms a foundation for future approaches to identify sarcasm targets. A special focus on the ‘Outside’ cases (*i.e.*, cases where the target of ridicule in a sarcastic text is not contained in the words present in the sentence) is likely to be helpful for sarcasm target identification.

Finally, our philosophy of a linguistically grounded approach (the notion of incongruity, in our case) will be useful for future work in computational sarcasm.

Bibliography

- G. Abercrombie and D. Hovy. Putting sarcasm detection into context: The effects of class imbalance and manual labelling on supervised machine classification of twitter conversations. *ACL 2016*, page 107, 2016.
- R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Machine learning: ECML 2004*, pages 39–50. Springer, 2004.
- Alias-i. Lingpipe natural language toolkit, 2008.
- A. Balamurali, A. Joshi, and P. Bhattacharyya. Harnessing wordnet senses for supervised sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1081–1091. Association for Computational Linguistics, 2011.
- D. Bamman and N. A. Smith. Contextualized sarcasm detection on twitter. In *Ninth International AAAI Conference on Web and Social Media*, 2015.
- F. Barbieri, F. Ronzano, and H. Saggion. Italian irony detection in twitter: a first approach. In *The First Italian Conference on Computational Linguistics CLiC-it 2014 & the Fourth International Workshop EVALITA*, pages 28–32, 2014a.
- F. Barbieri, H. Saggion, and F. Ronzano. Modelling sarcasm in twitter: a novel approach. *ACL 2014*, page 50, 2014b.
- S. K. Bharti, K. S. Babu, and S. K. Jena. Parsing-based sarcasm sentiment recognition in twitter data. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1373–1380. ACM, 2015.
- S. Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

- D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.
- D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson, and T. Winograd. Gus, a frame-driven dialog system. *Artificial intelligence*, 8(2):155–173, 1977.
- F. Boers. Applied linguistics perspectives on cross-cultural variation in conceptual metaphor. *Metaphor and Symbol*, 18(4):231–238, 2003.
- M. Bouazizi and T. Ohtsuki. Opinion mining in twitter how to make use of sarcasm to enhance sentiment analysis. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1594–1597. ACM, 2015a.
- M. Bouazizi and T. Ohtsuki. Sarcasm detection in twitter:” all your products are incredibly amazing!!!”-are they really? In *2015 IEEE Global Communications Conference (GLOBE-COM)*, pages 1–6. IEEE, 2015b.
- L. F. Bouton. A cross-cultural study of ability to interpret implicatures in english. *World Englishes*, 7(2):183–196, 1988.
- K. Buschmeier, P. Cimiano, and R. Klinger. An impact analysis of features in a classification approach to irony detection in product reviews. *ACL 2014*, page 42, 2014.
- E. Camp. Sarcasm, pretense, and the semantics/pragmatics distinction*. *Noûs*, 46(4):587–634, 2012.
- J. D. Campbell and A. N. Katz. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes*, 49(6):459–480, 2012.
- P. Carvalho, L. Sarmiento, M. J. Silva, and E. de Oliveira. 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*, pages 53–56. ACM, 2009.
- C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011a.
- C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011b.

- B. Charalampakis, D. Spathis, E. Kouslis, and K. Kermanidis. A comparison between semi-supervised and supervised text mining techniques on detecting irony in greek political tweets. *Engineering Applications of Artificial Intelligence*, pages –, 2016.
- Y. Choi and C. Cardie. Hierarchical sequential learning for extracting opinions and their attributes. In *Proceedings of the ACL 2010 conference short papers*, pages 269–274. Association for Computational Linguistics, 2010.
- T. Cohn and L. Specia. Modelling annotator bias with multi-task gaussian processes: An application to machine translation quality estimation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics on Computational Linguistics 2013*, pages 32–42, 2013.
- R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.
- A. Das and S. Bandyopadhyay. Dr sentiment knows everything! In *Proceedings of the Annual Meeting of the Association for Computational Linguistics/Human Language Technologies 2011: Systems Demonstrations*, pages 50–55, 2011.
- H. Daumé III, J. Langford, and D. Marcu. Search-based structured prediction. 2009.
- D. Davidov, O. Tsur, and A. Rappoport. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116. Association for Computational Linguistics, 2010a.
- D. Davidov, O. Tsur, and A. Rappoport. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116. Association for Computational Linguistics, 2010b.
- N. Desai and A. D. Dave. Sarcasm detection in hindi sentences using support vector machine. *International Journal*, 4(7), 2016.
- M. Deshpande. *Indian linguistic studies: festschrift in honor of George Cardona*. Motilal Banarsidass Publ., 2002.

- M. L. Dress, R. J. Kreuz, K. E. Link, and G. M. Caucci. Regional variation in the use of sarcasm. *Journal of Language and Social Psychology*, 27(1):71–85, 2008.
- J. Eisterhold, S. Attardo, and D. Boxer. Reactions to irony in discourse: Evidence for the least disruption principle. *Journal of Pragmatics*, 38(8):1239–1256, 2006.
- A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422, 2006.
- A. Fahrni and M. Klenner. Old wine or warm beer: Target-specific sentiment analysis of adjectives. In *Proc. of the Symposium on Affective Language in Human and Machine, AISB*, pages 60–63, 2008.
- D. I. H. Farías, V. Patti, and P. Rosso. Irony detection in twitter: The role of affective content. *ACM Trans. Internet Technol.*, 16(3):19:1–19:24, July 2016.
- E. Fersini, F. A. Pozzi, and E. Messina. Detecting irony and sarcasm in microblogs: The role of expressive signals and ensemble classifiers. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pages 1–8. IEEE, 2015.
- E. Filatova. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *LREC*, pages 392–398, 2012.
- J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.
- A. Ghosh and T. Veale. Fracking sarcasm using neural network. *WASSA NAACL 2016*, 2016.
- A. Ghosh, G. Li, T. Veale, P. Rosso, E. Shutova, A. Reyes, and J. Barnden. Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. In *Int. Workshop on Semantic Evaluation (SemEval-2015)*, 2015a.
- D. Ghosh, W. Guo, and S. Muresan. Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words. In *EMNLP*, 2015b.
- R. W. Gibbs. *The poetics of mind: Figurative thought, language, and understanding*. Cambridge University Press, 1994.
- R. W. Gibbs. Irony in talk among friends. *Metaphor and symbol*, 15(1-2):5–27, 2000.

- R. W. Gibbs and J. O'Brien. Psychological aspects of irony understanding. *Journal of Pragmatics*, 16(6):523 – 530, 1991.
- R. Giora. On irony and negation. *Discourse processes*, 19(2):239–264, 1995.
- R. González-Ibáñez, S. Muresan, and N. Wacholder. 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*, pages 581–586. Association for Computational Linguistics, 2011.
- A. Graves et al. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer, 2012.
- H. P. Grice, P. Cole, and J. L. Morgan. Syntax and semantics. *Logic and conversation*, 3:41–58, 1975.
- J. Haiman. Talk is cheap: Sarcasm, alienation, and the evolution of language. *Oxford University Press*, 1998.
- S. Harabagiu, A. Hickl, and F. Lacatusu. Negation, contrast and contradiction in text processing. In *AAAI*, volume 6, pages 755–762, 2006.
- I. Hernández-Farías, J.-M. Benedí, and P. Rosso. Applying basic features from sentiment analysis for automatic irony detection. In *Pattern Recognition and Image Analysis*, pages 337–344. Springer, 2015.
- S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.
- I. Hupont, P. Lebreton, T. Maki, E. Skodras, and M. Hirth. Is affective crowdsourcing reliable? In *IEEE Fifth International Conference on Communications and Electronics (ICCE) 2014*, pages 516–521, 2014.
- S. L. Ivanko and P. M. Pexman. Context incongruity and irony processing. *Discourse Processes*, 35(3):241–279, 2003.

- W. Jin, H. H. Ho, and R. K. Srihari. Opinionminer: a novel machine learning system for web opinion mining and extraction. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1195–1204. ACM, 2009.
- Y. Jo and A. H. Oh. Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 815–824. ACM, 2011.
- T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1999.
- T. Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM, 2006.
- T. Joachims. Svmhmm: Sequence tagging with structural support vector machines, 2008.
- A. Joshi, V. Sharma, and P. Bhattacharyya. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, volume 2, pages 757–762, 2015.
- A. Joshi, V. Tripathi, P. Bhattacharyya, and M. Carman. Harnessing sequence labeling for sarcasm detection in dialogue from tv series friends. *CoNLL 2016*, page 146, 2016a.
- A. Joshi, V. Tripathi, K. Patel, P. Bhattacharyya, and M. Carman. Are word embedding-based features for sarcasm detection? *EMNLP 2016*, 2016b.
- A. Khattri, A. Joshi, P. Bhattacharyya, and M. J. Carman. Your sentiment precedes you: Using an authors historical tweets to predict sarcasm. In *WASSA 2015*, page 25, 2015.
- S. Kotsiantis, D. Kanellopoulos, P. Pintelas, et al. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36, 2006.

- R. J. Kreuz and G. M. Caucci. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on computational approaches to Figurative Language*, pages 1–4. Association for Computational Linguistics, 2007.
- R. J. Kreuz and S. Glucksberg. How to be sarcastic: The echoic reminder theory of verbal irony. *Journal of Experimental Psychology: General*, 118(4):374, 1989.
- T. K. Landauer and S. T. Dumais. A solution to platos problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.
- C. J. Lee and A. N. Katz. The differential role of ridicule in sarcasm and irony. *Metaphor and Symbol*, 13(1):1–15, 1998.
- O. Levy and Y. Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 302–308, 2014.
- C. Liebrecht, F. Kunneman, and A. van den Bosch. The perfect solution for detecting sarcasm in tweets# not. 2013.
- P. Lison and J. Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of LREC 2016*, 2016.
- D. J. Litman and S. Pan. Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction*, 12(2-3):111–137, 2002.
- D. J. Litman and S. Silliman. Itspoke: An intelligent tutoring spoken dialogue system. In *Demonstration papers at HLT-NAACL 2004*, pages 5–8. Association for Computational Linguistics, 2004.
- P. Liu, W. Chen, G. Ou, T. Wang, D. Yang, and K. Lei. Sarcasm detection in social media based on imbalanced classification. In *Web-Age Information Management*, pages 459–471. Springer, 2014.
- D. L. Long and A. C. Graesser. Wit and humor in discourse processing. *Discourse processes*, 11(1):35–60, 1988.

- B. Lu, M. Ott, C. Cardie, and B. K. Tsou. Multi-aspect sentiment analysis with topic models. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 81–88. IEEE, 2011.
- S. Lukin and M. Walker. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. In *Proceedings of the Workshop on Language Analysis in Social Media*, pages 30–40, 2013.
- E. Lunando and A. Purwarianti. Indonesian social media sentiment analysis with sarcasm detection. In *Advanced Computer Science and Information Systems (ICACSIS), 2013 International Conference on*, pages 195–198. IEEE, 2013.
- Y. Mao and G. Lebanon. Isotonic conditional random fields and local sentiment flow. In *Advances in neural information processing systems*, pages 961–968, 2006.
- M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.
- D. Maynard and M. A. Greenwood. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Proceedings of LREC*, 2014.
- J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013a.
- J. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pages 897–908. ACM, 2013b.
- J. D. McAuliffe and D. M. Blei. Supervised topic models. In *Advances in neural information processing systems*, pages 121–128, 2008.
- A. K. McCallum. Mallet: A machine learning for language toolkit. 2002.
- O. Melamud, J. Goldberger, and I. Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *CONLL*, pages 51–61, 2016.

- M. Michelson and C. A. Knoblock. Unsupervised information extraction from unstructured, ungrammatical data sources on the world wide web. *International Journal of Document Analysis and Recognition (IJ DAR)*, 10(3-4):211–226, 2007.
- A. Mishra, D. Kanojia, S. Nagar, K. Dey, , and P. Bhattacharyya. Harnessing cognitive features for sarcasm detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.
- S. M. Mohammad and P. D. Turney. Crowdsourcing a word-emotion association lexicon. 29 (3):436–465, 2013.
- A. Mukherjee and B. Liu. Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 339–348. Association for Computational Linguistics, 2012.
- S. Muresan, R. Gonzalez-Ibanez, D. Ghosh, and N. Wacholder. Identification of nonliteral language in social media: A case study on sarcasm. *Journal of the Association for Information Science and Technology*, 2016.
- B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.
- B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.
- B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.
- T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics, 2004.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

- J. W. Pennebaker, M. E. Francis, and R. J. Booth. Linguistic inquiry and word count: Liwc 2001. volume 71, page 2001, 2001.
- J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
- G. Pilato, A. Augello, G. Vassallo, and S. Gaglio. Ehebby: An evocative humorist chat-bot. *Mobile Information Systems*, 4(3):165–181, 2008.
- T. Ptáček, I. Habernal, and J. Hong. Sarcasm detection on czech and english twitter. In *Proceedings COLING 2014*. COLING, 2014.
- G. Qiu, B. Liu, J. Bu, and C. Chen. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27, 2011.
- A. Rajadesingan, R. Zafarani, and H. Liu. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 97–106. ACM, 2015.
- R. Rakov and A. Rosenberg. ” sure, i did the right thing”: a system for sarcasm detection in speech. In *INTERSPEECH*, pages 842–846, 2013.
- D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP ’09, pages 248–256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
- A. Ramteke, A. Malu, P. Bhattacharyya, and J. S. Nath. Detecting turnarounds in sentiment analysis: Thwarting. In *ACL (2)*, pages 860–865, 2013.
- R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.
- A. Reyes and P. Rosso. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems*, 53(4):754–760, 2012.
- A. Reyes and P. Rosso. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, 40(3):595–614, 2014.

- A. Reyes, P. Rosso, and D. Buscaldi. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74:1–12, 2012.
- A. Reyes, P. Rosso, and T. Veale. A multidimensional approach for detecting irony in twitter. *Language Resources and Evaluation*, 47(1):239–268, 2013.
- E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing 2013*, pages 704–714, 2013.
- P. Rockwell and E. M. Theriot. Culture, gender, and gender mix in encoders of sarcasm: A self-assessment analysis. *Communication Research Reports*, 18(1):44–52, 2001.
- S. Shamay-Tsoory, R. Tomer, and J. Aharon-Peretz. The neuroanatomical basis of understanding sarcasm and its relationship to social cognition. *Neuropsychology*, 19(3):288, 2005.
- A. Silvio, B. C. Wallace, H. Lyu, and P. C. M. J. Silva. Modelling context with user embeddings for sarcasm detection in social media. *CoNLL 2016*, page 167, 2016.
- T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biologiske Skrifter*, 5:1–34, 1948.
- D. Sperber. Verbal irony: Pretense or echoic mention? *Journal of Experimental Psychology: General*, 113(1):130–136, 1984.
- A. Stent, J. Dowding, J. M. Gawron, E. O. Bratt, and R. Moore. The commandtalk spoken dialogue system. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 183–190. Association for Computational Linguistics, 1999.
- S. Stieger, A. K. Formann, and C. Burger. Humor styles and their relationship to explicit and implicit self-esteem. *Personality and Individual Differences*, 50(5):747–750, 2011.
- E. Sulis, D. Farías, Delia Irazú Hernández, P. Rosso, V. Patti, and G. Ruffo. Figurative messages and affect in twitter: Differences between #irony, #sarcasm and #not. *Knowledge-Based Systems*, 108:132 – 143, 2016. New Avenues in Knowledge Bases for Natural Language Processing.

- C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li. User-level sentiment analysis incorporating social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1397–1405. ACM, 2011.
- Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser. Svms modeling for highly imbalanced classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):281–288, 2009.
- D. Tannen. The pragmatics of cross-cultural communication. *Applied Linguistics*, 5(3):189–195, 1984.
- J. Tepperman, D. R. Traum, and S. Narayanan. ” yeah right”: sarcasm recognition for spoken dialogue systems. In *INTERSPEECH*, 2006.
- J. Thomas. Cross-cultural pragmatic failure. *Applied Linguistics*, 4(2).
- I. Titov and R. T. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *ACL*, volume 8, pages 308–316, 2008.
- K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 NAACL-HLT-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.
- O. Tsur, D. Davidov, and A. Rappoport. Icwsn-a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *ICWSM*, 2010.
- A. M. Turing. Can a machine think. *The world of mathematics*, 4:2099–2123, 1956.
- T. Veale and Y. Hao. Detecting ironic intent in creative comparisons. In *ECAI*, volume 215, pages 765–770, 2010.
- V. Verma. The mahabharata:(the great epic of ancient india)., 1986.
- M. A. Walker, J. E. F. Tree, P. Anand, R. Abbott, and J. King. A corpus for research on deliberation and debate. In *LREC*, pages 812–817, 2012.
- B. C. Wallace. Computational irony: A survey and new perspectives. *Artificial Intelligence Review*, 43(4):467–483, 2013.

- B. C. Wallace, L. K. Do Kook Choe, and E. Charniak. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 512–516, 2014.
- B. C. Wallace, D. K. Choe, and E. Charniak. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *ACL (1)*, pages 1035–1044, 2015.
- R. S. Wallace. *The anatomy of ALICE*. Springer, 2009.
- P.-Y. A. Wang. #irony or #sarcasm a quantitative and qualitative study based on twitter. In *27th Pacific Asia Conference on Language, Information, and Computation*, pages 349–256, 2013.
- Z. Wang, Z. Wu, R. Wang, and Y. Ren. Twitter sarcasm detection exploiting a context-based model. In *Web Information Systems Engineering–WISE 2015*, pages 77–91. Springer, 2015.
- J. M. Wiebe, R. F. Bruce, and T. P. O’Hara. Development and use of a gold-standard data set for subjectivity classifications. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics on Computational Linguistics 1999*, pages 246–253, 1999.
- D. Wilson. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743, 2006.
- A. Yessenalina, Y. Yue, and C. Cardie. Multi-level structured models for document-level sentiment classification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1046–1056. Association for Computational Linguistics, 2010.
- M. D. Zeiler. Adadelta: an adaptive learning rate method. *CoRR*, 2012.
- Z. Zhang, M. Dong, and S. S. Ge. Emotion analysis of children’s stories with context information. In *Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)*, pages 1–7. IEEE, 2014.
- J. Zhao, K. Liu, and G. Wang. Adding redundant features for crfs-based sentence sentiment classification. In *Proceedings of the conference on empirical methods in natural language processing*, pages 117–126. Association for Computational Linguistics, 2008.
- G. Zweig and C. J. Burges. The microsoft research sentence completion challenge. Technical report, Technical Report MSR-TR-2011-129, Microsoft, 2011.

Acknowledgments

In January 2013, I decided to leave a comfortable job in Pune to pursue a Ph.D. at the IITB-Monash Research Academy. Now that I look back, it has been a difficult journey. I pushed myself to limits, spent sleepless nights at work, and did some brave things along the way. I made (many) mistakes too, but have tried my best to learn from them. The Ph.D. journey changed my life, and the credit for that goes to the people around me. Through this acknowledgment, I wish to express my gratitude towards them.

My IITB supervisor, Pushpak Bhattacharyya, is a parent figure to me. He knows my strengths and weaknesses very well. His insistence on strong theoretical foundations and clarity in writing are two crucial lessons that a student gets from him. I have been his student since my Masters project in 2009 and will continue being that and learning from him, for the rest of my life.

My Monash supervisor, Mark Carman, is a super-cool advisor, who, in my first year, went over fundamentals of machine learning with me without laughing at my silly questions. Even after I returned to India, he ensured that our skype meetings were regular. I learned techniques of rigorous experiment design, and topic models, among other key skills from him. Mark's enthusiasm in guiding me, and his honest evaluation of the ongoing work kept me going!

In addition to my supervisors, I collaborated with many amazing people on several projects. They were very patient with my mood swings before conference deadlines. I would like to thank all my collaborators - Vaibhav Tripathi, Vinita Sharma, Pranav Goel, Prayas Jain, Kevin Patel, Anoop Kunchukuttan, Abhijit Mishra, Joe Cheri Ross, Meghna Singh, Jaya Saraswati, Rajita Shukla, Naman Gupta, Ravindra Soni, Prerana Singhal, Kashyap Popat, Anupam Khattri, Shubham Gautam, Sagar Ahire, Nivvedan S, Balamurali AR, and Diptesh Kanojia. Vaibhav and I formed a great team together and managed to get a lot of work done. I thank all annotators and evaluators, especially, Jaya Saraswati, Rajita Shukla and Meghna Singh. A research such as this depends on data, and without the efforts of these annotators, no work can even begin.

My Review Progress Committee consisted of my supervisors and examiners, Prof. Saketha Nath J, Prof. Preethi Jyothi from IIT Bombay, and Prof. Gholamreza Haffari from Monash University. Their suggestions, questions and feedback helped me immensely to identify shortcomings of my work and to improve it. Similarly, I thank all anonymous reviewers of all the conferences and journals where I submitted my papers. Their comments provided future directions for my work. I also thank Prof. Partha Pratim Talukdar, IISc Bangalore, and Prof. Mausam, IIT Delhi who were the external examiners for this thesis. Their comments have helped this thesis improve significantly.

The IITB-Monash Research Academy is a key reason why my Ph.D. journey became memorable. The staff at the Academy consisting of Laya Vijayan, Jayasree T, Kuheli Banerjee, Beena Pillai, Kiran More, Nancy Sowho, Rahul Ingle, Bharat Ingle, and others made processes at the Academy extremely seamless for the students. The communication skills sessions by Sheba Sanjay and Krishna Warriar were very beneficial. I found a good listener and a friend in Krishna Warriar. My interactions with the CEO, Dr. Murali Sastry, the COO, Dr. Kyatanahalli Nagabhushana, and the founding CEO, Dr. Mohan Krishnamoorthy were enriching. Overall, I wholeheartedly thank everyone in the Academy. I also thank the TCS Research Scholar Program Committee for supporting this Ph.D. and offering generous grants.

Center for Indian Language Technology (CFILT) lab at IIT Bombay is a place full of people passionate about NLP. Deepak Jagtap, Gajanan Rane and others kept me comfortable with their deft handling of the administrative processes in the lab. I thank every faculty, student and staff member of the lab for their support and guidance. Irawati Kulkarni shared with me valuable linguistic insights into sarcasm. I spent a lot of time discussing computational linguistics with Girish Ponkiya, Anoop Kunchukuttan, Joe Cheri Ross, Kevin Patel, Pujari Rajkumar, Ritesh Panjwani, and Raksha Sharma. The Sentiment Analysis (SA) group served as a forum to discuss new research ideas. I thank all members of the SA group over the last four years for their helpful feedback on my work. Meghna Singh and Madhav Joshi have meticulously proof-edited a near-final version of this thesis.

I express my gratitude to the Head of Department of CSE, IIT Bombay and Head of Faculty of IT, Monash University. I am also thankful to all office staff at Dept. of Computer Science and Engineering, IIT Bombay and Faculty of Information Technology, Monash University for their support.

The time at Monash was memorable due to the Academy students like Subhadeep, Sherly,

etc., and the Monash University students like Ishita, Divya, Susmit, Nader, etc. Back at IIT, Shehzaad, Pratik, Meghna, Diptesh, Samarth and I were quite the gang. I will remember the brilliant time I spent with them. The chai times with Anoop, Joe et al. saw debates on topics ranging from deep learning to demonetization. Priya Patil was the eternal sounding board and the lunch companion who always listened to me patiently. I often wonder why she has been so nice to me! Diptesh and Meghna are the favorite couple, who went out of their way on several occasions to make me feel better. Balamurali AR was my helpline - more the distress line in the last four years.

Abhinav Gupta is my closest friend for more than a decade now enough said! Mihir Bhosale helped me through some tough personal situations. Friends from Saathi (Anisha, Mihir and others) and Pravritti (Sushil, Avinash, Mahesh, Vaibhav, Sweekar, Gautam, Abhijit and others) contributed to the much-needed fun times. I also thank Conred Rodrigues and Vishal Jha for all the warmth. In addition, there were other interesting people: the ever-helpful neighbour Saurabh Suradhaniwar, the confidant Vikul Tomar, the favourite senior Sushrut Bhanushali, the first-day-of-Ph.D.-friend Vamshi Krishna, and the favourite junior Apurva Joshi.

I am also grateful to IIT Bombay for the amazing atmosphere it provides to a Ph.D. student, to the two cities of Mumbai and Melbourne for being such awesome places to live in, to Aahar café for the wholesome lunches, and to restaurants Sunny and Laxmi for helping me unwind after and between stressful conference deadlines. Swati Mali-Deshpande, Pratap Paralkar, and Uday Joshi offered valuable feedback on my Ph.D. from time to time.

I cannot thank my parents enough for the unconditional support they have shown towards me. Their support and their patience with me extends beyond this Ph.D. and nothing I do or say for them, is enough! I also thank my family (Udaykaka, Anupama, Veenakaku, Sudhirkaka, Ashutosh, Neha, and everyone else) for being a strength. I thank all my friends, and family members for their support. I am also grateful to all my teachers for enabling me to find my calling in life: teaching.

It is likely that I have missed important names here. The excitement and disbelief of completing this Ph.D. is to be blamed. I apologize.