"When Numbers Matter!": Detecting Sarcasm in Numerical Portions of Text

Abhijeet Dubey[*]
IIT Bombay

Lakshya Kumar[*†]
AI Research Einstein
Salesforce

Arpan Somani[*†]
Big Data Labs
American Express

Aditya Joshi
CSIRO

Pushpak Bhattacharyya
IIT Bombay

Abstract

Research in sarcasm detection spans almost a decade. However a particular form of sarcasm remains unexplored: sarcasm expressed through numbers, which we estimate, forms about **11%** of the sarcastic tweets in our dataset. The sentence *'Love waking up at 3 am'* is sarcastic because of the number. In this paper, we focus on detecting sarcasm in tweets arising out of numbers. Initially, to get an insight into the problem, we implement a rule-based and a statistical machine learning-based (ML) classifier. The rule-based classifier conveys the crux of the numerical sarcasm problem, namely, incongruity arising out of numbers. The statistical ML classifier uncovers the indicators i.e., features of such sarcasm. The actual system in place, however, are two deep learning (DL) models, CNN and attention network that obtains an F-score of **0.93** and **0.91** on our dataset of tweets containing numbers. To the best of our knowledge, this is the first line of research investigating the phenomenon of sarcasm arising out of numbers, culminating in a detector thereof.

1 Introduction

Sarcasm is a challenge to sentiment analysis because it uses verbal irony to express contempt or ridicule, thereby, potentially confusing typical sentiment classifiers. Several approaches for sarcasm detection have been reported in the recent past (Hazarika et al., 2018; Joshi et al., 2017; Ghosh and Veale, 2017; Buschmeier et al., 2014; Riloff et al., 2013). In this paper, we focus on a peculiar form of sarcasm: sarcasm expressed through numbers. In other words, the goal of this paper is the classification task where a tweet containing one or more numbers is classified as sarcastic due to numbers or non-sarcastic. For example, the sentence '*Having 2 hours to write a paper is fun*' is sarcastic. The numeral 2 plays a key role in conveying sarcasm. Therefore, in this paper, we focus on different approaches for the detection of sarcasm due to numbers. Towards this, we first introduce the task, identify its challenges, introduce a labeled dataset and devise three approaches for the task. Our approaches are based on three prevalent paradigms of NLP: rule-based, statistical machine learning-based[1] and deep learning-based.

The contribution of the paper is as follows:

1. The paper details the purpose and challenges of the problem.
2. We introduce a labeled[2] dataset of 60949 tweets containing numbers.
3. Finally, we present approaches which will serve as strong baselines for future work in detecting sarcasm arising due to numbers.

The rest of the paper is organized as follows. In Section 2, we present our motivation. In Section 3, we discuss the related work in detail. Then, we present insights into the problem using rule-based and statistical machine learning-based approaches in Section 4. Then, in Section 5, we present two deep learning-based approaches. In Section 6, we outline the experimental setup and present the results of our experiments in Section 7. We present both qualitative as well as quantitative error analysis in Section 8. Finally, we conclude the paper and discuss future work in Section 9.

2 Motivation

The challenge that sarcastic text poses to sentiment analysis has led to research interest in com-

[*] Equal Contribution.

[†] The work was done when authors were doing their Masters at IIT-Bombay.

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

[2] labels: sarcastic due to number, non-sarcastic.

Proceedings of the 10th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 72–80
Minneapolis, June 6, 2019. ©2019 Association for Computational Linguistics

putational sarcasm. While several approaches to detect sarcasm have been reported (González-Ibáñez et al., 2011; Joshi et al., 2015), they may fall short in case of sarcasm expressed via numbers. Consider the following three sentences:

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

At the time of writing this paper, a battery backup of 38 hours is good for phones while a battery backup of 2 hours is bad. Therefore, sentences 1 and 2 are non-sarcastic because the sentiment of the adjectives (‘*awesome*’ and ‘*terrible*’) conforms with the sentiment associated with the corresponding numerical values. On the contrary, the sarcasm in sentence 3 above occurs because of incompatibility/incongruity[3] between the word ‘*awesome*’ (positive word) and ‘*2 hours*’ (numerical value). The above examples illustrate that the sarcasm can arise due to numbers which can mislead a normal sarcasm detection system. Therefore, in this paper, we aim to solve the problem of detecting sarcasm arising due to numbers. The utility of our work lies in the fact that our system is a crucial link in a pipeline for sarcasm detection where the input tweets first pass through a general sarcasm detector, out of which the tweets labeled as non-sarcastic are then subjected to further scrutiny of the numerical sarcasm detector. Figure 1 shows the interfacing of our module with the overall sarcasm detection system.

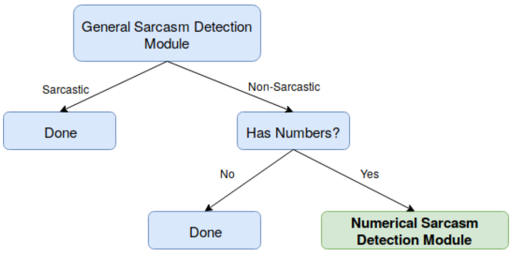
General Sarcasm Detection Module
Sarcastic
Non-Sarcastic
Done
Has Numbers?
No
Yes
Done
Numerical Sarcasm Detection Module

Figure 1: Interfacing of our module with the overall sarcasm detection system

3 Related Work

Sarcasm and irony detection has been extensively studied in linguistics, psychology, and cognitive science (Gibbs, 1986; Utsumi, 2000). Computational detection of sarcasm has become a popular area of natural language processing research in recent years (Joshi et al., 2017). Tepperman et al. (2006) present sarcasm recognition in speech using spectral (average pitch, pitch slope, etc.), prosodic and contextual cues. Carvalho et al. (2009) use simple linguistic features like an interjection, changed names, etc. for irony detection. Davidov et al. (2010) train a sarcasm classifier with syntactic and pattern-based features. González-Ibáñez et al. (2011) state that sarcasm transforms the polarity of an apparently positive or negative utterance into its opposite. Liebrecht et al. (2013) show that sarcasm is often signaled by hyperbole, using intensifiers and exclamations; in contrast, non-hyperbolic sarcastic messages often receive an explicit marker. Riloff et al. (2013) capture sarcasm as a contrast between a positive sentiment word and a negative situation. Joshi et al. (2015) show how sarcasm arises because of implicit or explicit incongruity in the sentence. Buschmeier et al. (2014) analyze the impact of different features for sarcasm/irony classification. Bouazizi and Ohtsuki (2016) propose a pattern-based approach to detect sarcasm on Twitter. As deep learning techniques gain popularity, Ghosh and Veale (2016) propose a neural network semantic model for sarcasm detection. They use Convolutional Neural Network (CNN) followed by a Long Short Term Memory (LSTM) network and finally a fully connected layer. Poria et al. (2016) propose a novel method to detect sarcasm using CNN. They use a pre-trained CNN for extracting sentiment, emotion and personality features for sarcasm detection. Amir et al. (2016) propose a deep-learning-based architecture to incorporate additional context for sarcasm detection. They propose an approach to learn user embeddings to provide contextual features, going beyond the lexical and syntactic cues. Finally, they use these user embeddings for sarcasm detection. Zhang et al. (2016) use a bi-directional Gated Recurrent Unit (GRU) followed by a pooling neural network to detect sarcasm. Ghosh and Veale (2017) propose a neural architecture that considers the speaker’s mood on the basis of most recent prior tweets for sarcasm detection. Farías et al. (2016) propose a

[3] Ivanko and Pexman (2003) describe the relationship between incongruity and sarcasm.

novel model using affective features based on a wide range of lexical resources available for English for detecting irony in tweets. Sulis et al. (2016) study the difference between sarcasm and irony in tweets. They propose a novel set of sentiment, structural and psycholinguistic features for distinguishing between irony and sarcasm. Peled and Reichart (2017) and Dubey et al. (2019) model sarcasm interpretation as a monolingual machine translation task. They use Moses[4], attention networks, and pointer generator networks for the task of sarcasm interpretation. Van Hee et al. (2018) present the first shared task in irony detection in tweets. Recently, Hazarika et al. (2018) propose a hybrid approach for sarcasm detection in online social media discussions. They extract contextual information from the discourse of a discussion thread. They also use user embeddings that encode stylometric and personality features of users and content-based feature extractors such as CNN and show a significant improvement in classification performance on a large Reddit corpus.

4 Getting Insight into the Problem

End-to-end deep learning (DL) architectures are very popular for solving NLP problems these days. However, DL approaches do not give insight into the problem. To better understand the "numerical sarcasm problem" (detecting sarcasm arising due to numbers in tweets), we first implement a rule-based and statistical machine learning-based approach before embarking on the deep learning-based approach. In this section, we introduce a rule-based approach that conveys the crux of the numerical sarcasm problem, namely, incongruity arising out of numbers. We also present a statistical machine learning-based approach that conveys the importance of handcrafted features for decision making.

4.1 Rule-based Approach

Figure 2 shows our rule-based system. This approach considers noun phrases in the tweet as candidate contexts and determines the optimal threshold of a numerical measure for each context.

We divide tweets into two sets, namely sarcastic and non-sarcastic repository. We represent each tweet in the form of a tuple containing tweet index number, noun phrase vector, numerical value, and unit of measurement. For example, assume that the 14th instance in the dataset is the sarcastic tweet *'This phone has an awesome battery backup of 2 hours'*. This tweet contains two noun phrases: '*phone*' and '*awesome battery backup*'. The words in these two noun phrases are *'phone', 'awesome', 'battery', 'backup'*. We first convert these words into 200-D word vectors (initialized using GloVe (Pennington et al., 2014) and fine-tuned on our dataset). Then we sum up word vectors of words in the noun phrase list and normalize them by the length of the noun phrase list. We call this the noun phrase vector. Given these entities, the tweet representation is: *(14, Noun Phrase Vector, 2, 'hours')*. Since the tweet is sarcastic, it is stored in the sarcastic repository. In addition to tweet entries, both sarcastic and non-sarcastic repositories also maintain two dictionaries: (a) Dictionary of mean values where each entry is a key-value pair where key is the unit of measurement and value is the average of all the numbers corresponding to that number unit and (b) Dictionary of standard deviation is created in a similar manner.

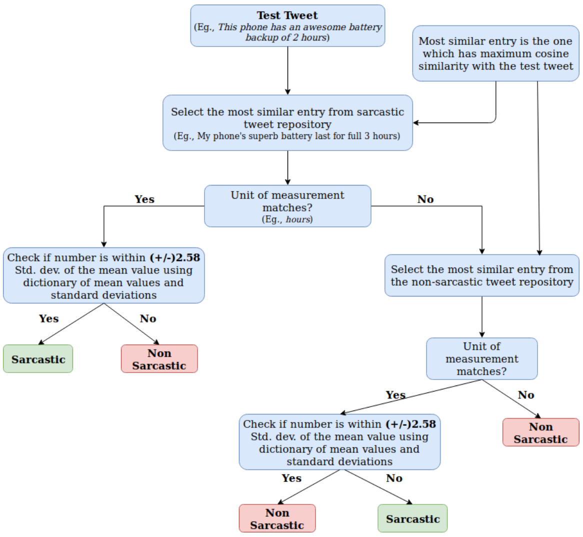
Test Tweet
(Eg., This phone has an awesome battery backup of 2 hours)
Most similar entry is the one which has maximum cosine similarity with the test tweet
Select the most similar entry from sarcastic tweet repository
(Eg., My phone's superb battery last for full 3 hours)
Yes
Unit of measurement matches?
(Eg., hours)
No
Check if number is within (+/-)2.58 Std. dev. of the mean value using dictionary of mean values and standard deviations
Select the most similar entry from the non-sarcastic tweet repository
Yes
No
Sarcastic
Non Sarcastic
Unit of measurement matches?
Yes
No
Check if number is within (+/-)2.58 Std. dev. of the mean value using dictionary of mean values and standard deviations
Non Sarcastic
Yes
No
Non Sarcastic
Sarcastic

Figure 2: Rule-Based Approach

A test tweet is classified as sarcastic or non-sarcastic according to the following steps:

1. **Computation of noun phrase vector:** We create a noun phrase vector from the words in the noun phrase list of the test tweet as described above.

2. **Sarcastic repository consultation:** We compute the cosine similarity of noun phrase vectors of test tweet and tweets in sarcastic repository respectively. Then, we select

[4] http://www.statmt.org/moses/

the tweet from the sarcastic repository whose noun phrase vector has the maximum cosine similarity with the noun phrase vector of the test tweet. We call this the *'most similar entry'*. If the unit of measurement in the most similar entry is same as that in the test tweet, we use the dictionary of mean values and dictionary of standard deviations to check whether the number present in the test tweet is within ± 2.58 standard deviation of the mean value for that unit of measurement. If it is, the tweet is predicted as sarcastic, otherwise, non-sarcastic.

3. **Non-Sarcastic repository consultation:** If the unit of measurement in the most similar entry from the sarcastic repository is not the same as that in the test tweet, we select the most similar entry to the test tweet from the non-sarcastic repository and proceed in a similar manner.

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

4.2 Statistical Machine Learning-based Approach

We use two statistical machine learning-based classifiers: **SVM** and **Random-forest**. We use the following features in our statistical machine learning-based approach.

- **Sentiment-based features (S)**: Number of positive words, number of negative words[5], number of highly emotional positive words, number of highly emotional negative words (Positive/Negative word is said to be highly emotional if it is an adjective, adverb or verb).

- **Emoticon-based features (E)**: Number of positive emoticons, number of negative emoticons, contrast between word and emoticon which is a boolean feature that takes the value as 1 when either positive word and negative emoticon is present or negative word and positive emoticon is present in the tweet.

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

- **Numerical value (NV)**: The actual number in the tweet.

- **Numerical unit (NU)**: One-hot representation of the unit of measurement.

5 Deep Learning-based Approach

In this section, we describe two deep learning-based models.

5.1 CNN-FF Model

Figure 3 shows the architecture of CNN-FF model. We use embedding matrix $E \in \mathbb{R}^{|V| \times d}$ with $|V|$ as the vocabulary size and d as the word vector dimension. For the input tweet, we obtain an input matrix $I \in \mathbb{R}^{|S| \times d}$ where $|S|$ is the length of the tweet. I_i is the d-dimensional vector for i-th word in the tweet in the input matrix. Let k be the length of the filter, and the vector $f \in \mathbb{R}^{|k| \times d}$ is a filter for the convolution operation. For each position p in the input matrix I, there is a window w_p of k consecutive words, denoted as:

$$w_p = [I_p, I_{p+1}, ..., I_{p+k-1}] \quad (1)$$

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

$$c_p = func(w_p \circ f + b) \quad (2)$$

where $\circ$ is the element-wise multiplication, $b \in \mathbb{R}$ is a bias term and $func$ is a nonlinear transformation function. We use multiple convolution filters of different sizes to obtain a feature map of the given tweet. We further apply max-over-time pooling over the obtained feature map. The output from each filter is concatenated to get the final feature vector. This feature vector acts as input to the fully-connected layer. We train the entire model by minimizing the binary cross-entropy loss.

$$E(y, \widehat{y}) = \sum_{i=1}^{e} y_i \log(\widehat{y_i}) \quad (3)$$

[5] Positive and negative words are selected using SentiWordNet (Baccianella et al., 2010).

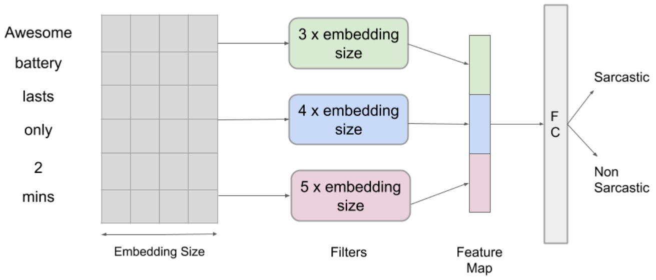

Figure 3: CNN followed by Fully Connected Layer for Numerical Sarcasm Detection

5.2 Attention Network

Figure 4 shows the architecture of our attention network. It consists of two main parts: a word encoder and a word level attention layer. We describe these two components as follows,

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

$$x_t = E^T w_t, t \in [1, T] \tag{4}$$

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(x_t), t \in [1, T] \tag{5}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(x_t), t \in [T, 1] \tag{6}$$

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

2. **Word Level Attention:** We claim that numbers play a crucial role while predicting sarcasm in tweets containing numbers. Hence, we introduce the attention network to extract information which is important to the overall meaning of the tweet. Our attention architecture is similar to the attention model introduced in Yang et al. (2016), except that we do not use hierarchical attention since tweets are short sentences and do not have a hierarchical structure.

$$u_t = tanh(W_w^T h_t + b_w) \tag{7}$$

$$\alpha_t = \frac{exp(u_t^T u_w)}{\sum_t exp(u_t^T u_w)} \tag{8}$$

$$s_i = \sum_t \alpha_t h_t \tag{9}$$

$$p = softmax(W_c^T s_i + b_c) \tag{10}$$

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

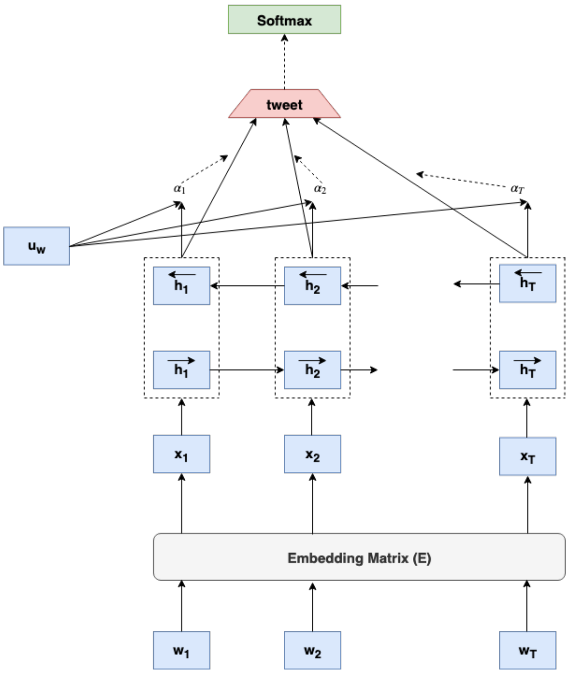

Figure 4: Attention Network for Numerical Sarcasm Detection

6 Experimental Setup

We create two datasets containing tweets as follows. We download tweets containing hashtags

Dataset	Sarcastic	Non Sarcastic
Dataset 1	100000 (28.57%)	250000 (71.43%)
Dataset 2	11024 (18.1%)	49925 (81.9%)

Table 1: Statistics of Datasets. From Dataset 1, we extract sarcastic and non-sarcastic tweets containing numbers and then manually annotate them to obtain a high quality labeled dataset of tweets containing numbers.

#sarcasm, #sarcastic, #BeingSarcastic as sarcastic, and those with #nonsarcasm, #notsarcastic as the non-sarcastic, using the Twitter-API. We eliminate duplicate tweets, retweets, remove URLs, usernames, hashtags and other Non-ASCII characters from the tweets. We call this Dataset 1 which contains a total of 350000 tweets. From Dataset 1, we select a subset of tweets which contain numerical values. Then, we remove irrelevant tweets from this subset, like the ones which contain alphabet or special character adjacent to a number like *Model34d, 4s,* < 3 (heart smiley), etc. As a final step, to improve the quality of our dataset, we give the following instructions to two annotators who independently annotate tweets to evaluate if the tweets containing numbers are really sarcastic due to the number or not. We call this Dataset 2 (Dataset of tweets containing numbers) which is a subset of Dataset 1 and contains a total of 60949 tweets.

1. Mark the tweet with *label = 1*, if it is sarcastic and the sarcasm is arising due to numbers.
2. Mark the tweet with *label = 0*, otherwise.

The value of Cohen's Kappa which measures inter-annotator agreement is **0.81**. Table 1 shows the percentage of sarcastic and non-sarcastic tweets in Dataset 1 and Dataset 2 respectively.

As baselines, we re-implement the work (by adapting features wherever necessary) reported by González-Ibáñez et al. (2011), Liebrecht et al. (2013) and Joshi et al. (2015). The choice of our baselines is based on approaches that use only the text to be classified. For statistical machine learning-based approaches, we use **SVM** with RBF kernel and $c = 1.0$ using grid-search and **Random-forest** with $number\ of\ estimators$ = 10. For deep learning-based approaches, we use $200D$ tweet word embeddings, initialized using GloVe and fine tuned on our data. For CNN-FF Model, we use 128 filters each of size 3, 4 and 5, i.e., a total 128×3 filters. We use a dropout of 0.5. We train the network using mini-batch gradient descent. Finally, we report the average 5-fold cross-validation values in Table 4.

7 Results

Table 3 shows the evidence of degradation in the performance of three previous approaches on Dataset 2 (dataset of tweets containing numbers). We observe that among the three previous approaches, features from Joshi et al. (2015) perform the best and obtain an F-score of **0.72** and **0.27** on Dataset 1 and Dataset 2 respectively. There is a **degradation of 45% points** in F-score from Dataset 1 to Dataset 2 which clearly shows that the past approaches are not able to capture the sarcasm arising due to numbers because their features are not designed to capture the incongruity arising due to numbers. This strengthens our claim. To further strengthen the importance of our approaches, we evaluate them on Dataset 1 and Dataset 2 respectively using the strategy illustrated in Figure 1. On Dataset 1, we apply our approaches on tweets that are misclassified by the best performing past approach of Joshi et al. (2015). We also evaluate our approaches on Dataset 2 and show the evidence of overall improvement in F-score in Table 4. Our CNN-FF model obtains the best F-score of **0.88** and **0.93** which is a **significant improvement of 16%** and **66%** points in F-score over the best performing past approach of Joshi et al. (2015) on Dataset 1 and Dataset 2 respectively.

To check if our results are statistically significant, we perform **Kolmogorov-Smirnov test** (Karson, 1968) and find that our results are statistically significant.

8 Error Analysis & Visualization

Table 2 shows the distribution of attention weights over input tweets and illustrates the importance of numbers while making the sarcastic/non-sarcastic decision. We also perform a qualitative analysis of errors which results in six categories:

1. **Sarcasm not due to numbers**: Sarcastic sentences which contain a number but the sarcasm is not due to the number. For example, '*phelps will be the mvp for 2014 lmao phelpshaterhere*'

Sarcastic due to numbers	Non-Sarcastic
i 'm so excited to work 14 hours during a blizzard tomorrow & get to drive home in the worst	just got done with a nice 25 min run clear air feels good
i love being wide awake at 345 in the morning	it 's 12.00 pm and im still in my pj 's god i love lazy non working days
5 hours left and am down to 50 % battery battery yea like that 's going to last x	2 finals today wish me luck : x i still have a lot to study after my first final

Table 2: Distribution of attention weights over some input tweets while making the numerical sarcastic/non-sarcastic decision. The darker the color and larger the font, the higher is the weight

Approach	Dataset 1	Dataset 2
González-Ibáñez et al. (2011)	0.68	0.17
Liebrecht et al. (2013)	0.67	0.21
Joshi et al. (2015)	**0.72**	**0.27**

Table 3: Evidence of F-score degradation of previous approaches on Dataset 2 (numerical sarcasm dataset)

Approaches	Dataset 1	Dataset 2
Rule-Based Approach	0.83	0.78
SVM	0.86	0.82
Random Forest	0.86	0.84
CNN-FF	**0.88**	**0.93**
Attention Network	0.87	0.91

Table 4: Evidence of overall improvement in F-score using our approaches

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

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

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

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

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

To clearly understand the proportion of errors made by each of our approaches, we also perform quantitative analysis of errors which results in three categories: (A) Examples where the rule-based approach fails to detect sarcasm but machine learning-based approach detects it, (B) Examples where the machine learning-based approach fails to detect sarcasm but deep learning-based approach detects it, and (C) Examples where none of the approaches detect the sarcasm.

Error Category	(A)	(B)	(C)
Sarcasm not due to numbers	34	32	10
Numbers enhancing sarcasm	12	22	20
Comparison of numbers	4	12	12
Unseen situations	32	14	18
'Special' numbers	12	12	30
Additional Context Required	6	8	10

Table 5: Percentage of errors for the three configurations; (A): Rule-based approach goes wrong but statistical machine learning-based approach is correct, (B): Statistical machine learning-based approach goes wrong but the deep learning-based approach is correct, (C): All three approaches go wrong

Table 5 shows the proportion of errors in the three configurations. The ad-hoc nature of the rule-based approach reflects in percentage values. Similarly, analyzing tweets in which sarcasm is enhanced due to numbers and sarcasm arising due to a comparison between numbers appear as useful pointers for future work.

9 Conclusions & Future Work

In this paper, we present approaches to handle a special case of sarcasm: sarcasm expressed through numbers. We show that past works in sarcasm detection do not perform well for text containing numbers. We then compare our approaches with three previous works and show the significant improvements in F-score when our approaches are used on top of other approaches. To the best of our knowledge, this is the first line of research investigating the phenomenon of sarcasm arising out of numbers, culminating in a detector thereof. Our error analysis points out to specific numerical sarcasm challenges, thus creating immediate future tasks. The utility of our work lies in the fact that our system is a crucial link in a pipeline for sarcasm detection, where a tweet labeled as non-sarcastic and containing a number gets a final check of being sarcastic. Future work consists of incorporating a language model for numbers to handle unseen situations. Long term future work consists in tackling irony in general, humor and humble bragging (*'Oh my life is miserable: I have to sign 500 autographs a day'*) all of which have their genesis in incongruity.

Acknowledgement

The authors would like to thank Diptesh Kanojia, Urmi Saha, Kevin Patel, Rudra Murthy, Sandeep Mathias, Jaya Saraswati and members of the Center For Indian Language Technology, IIT Bombay. The authors would also like to thank Minali Upreti for helping with the diagrams and the anonymous reviewers for their valuable comments and feedback.

References

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mario J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 167–177. Association for Computational Linguistics.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Languages Resources Association (ELRA).

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

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

Paula Carvalho, Luís Sarmento, Mário J. Silva, and Eugénio de Oliveira. 2009. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy" ;-). In *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion*, TSA '09, pages 53–56, New York, NY, USA. ACM.

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

Abhijeet Dubey, Aditya Joshi, and Pushpak Bhattacharyya. 2019. Deep models for converting sarcastic utterances into their non sarcastic interpretation. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, CoDS-COMAD '19, pages 289–292, New York, NY, USA. ACM.

Delia Irazú Hernańdez Farías, Viviana Patti, and Paolo Rosso. 2016. Irony detection in twitter: The role of affective content. *ACM Trans. Internet Technol.*, 16(3):19:1–19:24.

Aniruddha Ghosh and Dr. Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 161–169. Association for Computational Linguistics.

Aniruddha Ghosh and Tony Veale. 2017. Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 482–491. Association for Computational Linguistics.

Raymond W Gibbs. 1986. On the psycholinguistics of sarcasm. *Journal of Experimental Psychology: General*, 115(1):3–15.

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

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848. Association for Computational Linguistics.

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

Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2017. Automatic sarcasm detection: A survey. *ACM Comput. Surv.*, 50:73:1–73:22.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. 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: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Marvin Karson. 1968. Handbook of methods of applied statistics. volume i: Techniques of computation descriptive methods, and statistical inference. volume ii: Planning of surveys and experiments. i. m. chakravarti, r. g. laha, and j. roy, new york, john wiley; 1967, $9.00. *Journal of the American Statistical Association*, 63(323):1047–1049.

Christine Liebrecht, Florian Kunneman, and Antal Van den Bosch. 2013. The perfect solution for detecting sarcasm in tweets #not. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 29–37. Association for Computational Linguistics.

Lotem Peled and Roi Reichart. 2017. Sarcasm sign: Interpreting sarcasm with sentiment based monolingual machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1690–1700. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1601–1612. The COLING 2016 Organizing Committee.

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

Emilio Sulis, Delia Iraz Hernndez Faras, Paolo Rosso, Viviana Patti, and Giancarlo Ruffo. 2016. Figurative messages and affect in twitter: Differences between irony, sarcasm and not. *Knowledge-Based Systems*, 108:132 – 143. New Avenues in Knowledge Bases for Natural Language Processing.

Joseph Tepperman, David R. Traum, and Shrikanth Narayanan. 2006. ”yeah right”: sarcasm recognition for spoken dialogue systems. In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing, Pittsburgh, PA, USA, September 17-21, 2006*.

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

Cynthia Van Hee, Els Lefever, and Veronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2449–2460. The COLING 2016 Organizing Committee.