IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-1, Jan 2019 E-ISSN: 2347-2693

Process Recognizing Numerical Sarcasm in Tweets

Aakarsh Mehta[1*], Sandeep Nigam[2]

[1]Department of Computer Science & Engineering, Amity University Madhya Pradesh, Gwalior, India
[2] Department of Computer Science & Engineering, International Institute of Information Technology, Bhubaneswar, India

[*]Corresponding Author: Aakarshmehta98@gmail.com, Tel.: +91-86022-90120

Available online at: www.ijcseonline.org

Accepted: 21/Jan/2019, Published: 31/Jan/2019

Abstract— In a world of social media, sentiment analysis has played a significant role in gathering useful trends and information on mass opinion towards any individual, product, organization, political group or any sports franchise. Sarcasm is one such unique sentiment, where the intended meaning is the opposite of written text (opinion). It can also be defined as concealed mockery through written or expressed remark which makes it complicated in sentiment detection systems. Numerical sarcasm is one such field that attracted researchers. Finding the sarcasm due to the presence of numerical data in the given statement can be concluded as numerical sarcasm detection. There are various computational systems in this paper that we tried to incorporate with Machine Learning and Deep learning approaches. We have used techniques such as SVM, K-NN, and LSTM for numerical sarcasm detection and incorporated sci-kit, numpy, tensor flow in our proposed work.

***Keywords*—**Sentimental Analysis, Social Media, Machine Learning, DNN, Semantic features.

I. Introduction

Sarcasm is a type of oral communication behaviour in which the speakers express their message in an implicit manner. A general form of sarcasm on Twitter contains a +ve sentiment contrasted with a -ve condition like many of sarcastic tweets contains a +ve sentiment, such as Love or Fun, followed by a feeling that gives information about an undesirable activity or situation. Identification of sarcasm can support several sentiment analysis NLP applications such as in review summarization, communication approaches, and review rating methods. Sarcasm demands some mutual understanding between presenter and target audience. It is an extraordinarily contextual method. Generally, computational approaches are methods of sarcasm detection, however, treat it as a purely linguistic problem, by using information and facts such as lexicon hints and their corresponding sentiment as predictive features. An experience of 3 milestones is achieved in the investigation of semi-supervised pattern extraction to determine implicit feelings, use of hashtag depending supervision, and use of context beyond the written text. The study to detect Sarcasm in numbers has been presented [1]. For example, "A person works for 40 hours all seven days to be this poor". The number of '40 hours' is a significant pointer of the sarcasm in this sentence. The challenges of the problem are analyzed. This paper manages to recognize sarcasm in numerical segments of content, as in this example. At that point, Rule-Based approach, Machine learning and deep learning-based approach and its variations that takes diverse features as a contribution to learning have been introduced.

II. Related Work

Computational detection of sarcasm has transformed into a famous area of NLP research nowadays. It uses simple and easy linguistic features like an interjection, changed names, etc for sarcasm detection [2,3]. Train a sarcasm classifier with syntactic and pattern-based features [4]. Suggest that sarcasm converts the polarity of an apparently +ve or -ve utterance into its opposite [5]. Advice that sarcasm is a comparison between a positive emotion expression and negative situation. Predict a pattern-based approach to recognize sarcasm on Twitter [6]. They presented four sets of features that cover different kinds of sarcasm. While deep learning strategies acquire popularity, varieties of deep learning-based architectures for sarcasm detection have also shown up in literature work, which will also provide a neural network semantic structure for sarcasm detection [7]. Their approach consists of Long Short-Term Memory (LSTM) network and Deep Neural Network (DNN). Provided a deep-learning based architecture to automatically recognize user embeddings [8].

III. APPROACHES

In this approach, to identify numerical sarcasm, rule-based methodology [1] is applied which has been explained in the upcoming subsections.

A. Noun Phrase Exact Meaning

In this concept, Firstly, two repositories are built, i.e.., Sarcastic and Non-Sarcastic utilizing a training dataset. Every entry in the repository is of the format: (Tweet Index No., Noun Phrase list, Mean of Number unit, Std Dev of Number unit, Number Unit)

- Extract noun keywords in the tweet, using a parser (nltk parser)
- Choose Number unit as the word following the word POS tagged as CD. Some of the number units are minutes, hour, days, and years and soon.
- Entry to the corresponding repository according to the label of the tweet is then added. This entry is of the structure as described above. For example, if there is a sarcastic tweet- This mobile phone has an excellent battery power back-up for 50 minutes.

  The Noun phrases from this constituency parse tree are extracted and a noun-phrase list of this sarcastic tweet has been made as given below: [‘mobile’, ‘phone’, ‘excellent’, ‘battery’, ‘backup’, ‘minutes’]

  After accumulating the noun phrase list, tweet is saved in the sarcastic repository as: [Tweet Index [‘mobile’, ‘phone’, ‘excellent’, ‘battery’, ‘backup’, ‘minutes’], mean of numbers having unit as ’hours’, Std dev of number having unit as ’minutes’, ’minutes’]. For all the other tweets, i.e., numerical sarcastic or non-sarcastic tweets, a similar approach is used to store them in their corresponding repositories. Numerical Sarcasm in a test tweet is predicted as below.

  Noun phrases, number and number unit from the test tweet are extracted. Then, the following rules are implemented.
- Initially, check with the sarcastic tweet repository is done. On matching words in the noun phrase list between the test tweet and entries in the repository, the best similar entry is elected from the sarcastic repository. Then the number unit of the entry with that in the test tweet is matched.
- When the number unit satisfies, inspection is done whether the number present in the test tweet lies inside 2.586 (2.586 indicates the 99 percent confidence interval) Std dev of the mean value for that number unit present in the matched sarcastic entry.

- When the number unit fails to match, a check with the non-sarcastic tweet repository is done and then finds the best equivalent entry based on exact matching of the noun phrase list of test tweet an entry in the repository. In case the number unit matches, then consultation is done, whether the number present in the test tweet lies within 2.58 std dev of the mean value for the number unit inside the matched non-sarcastic entry. When it exists, so, predict the tweet as non-sarcastic but if it will not, then predict sarcastically.

B. Machine Learning Based Methods

In order to identify Numerical Sarcasm, machine learning based techniques such as SVM, KNN can be used by using different categories of features as discussed below.

1) Selection of Capital Words: This can be shown as a tone which can be useful for identifying Sarcasm. When phrase holds capital words it may very well indeed be Sarcastic. Example ’Okay THAT makes sense’. The capital ’THAT’ indicates the sarcasm. In this case ’THAT’ is used as a feature.
2) Selection of punctuation marks entries: These kinds of features consist of a variety of !? , .’.....
3) The numeral in the tweet: This feature will be just the numeral contained in the tweet.
4) Sentiment-based features: All these features consist of a bunch of positive words, a bunch of negative words. Positive/Negative word is mentioned to be strongly sentimental when it is a part-of-speech tag.

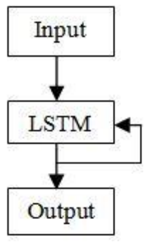

Figure 1: The basic design of LSTM

C. DNN Based Approach

The machine learning techniques give good results but not that efficient because of the constraint on the size of the dataset. So, the DNN (Deep Neural Network) technique which works efficiently with large sized data set opts. In DNN there are techniques like CNN, RNN, and LSTM etc which give effective results. In this, LSTM technique, with four layers is proposed. Comparing to CNN and RNN, the proposed technique LSTM is giving a better result because of the reason that the other two techniques don’t have the capability to learn long-term dependencies. The basic operations to be performed in LSTM are shown in figure 1.

- LSTM (Long Short Time Memory) is an amazing type of recurrent neural network which performs, for several works, more desirable because they are suitable for avoiding the issue of long-term dependency difficulty.

Recollecting the information for long periods is practically their default behaviour. All the recurrent neural networks gain the form of a sequence of repeating modules of a neural network. LSTMs also contain this chain-like structure, but the reproducing module is of different structure. Rather than obtaining one neural network layer, there are four interacting in a unique manner. LSTM holds special units known as memory blocks in the recurrent hidden layer. These types of memory blocks hold memory cells which store the temporal state of a network with self-connections, in addition to special multiplicative units known as gates which are usually used to handle the flow of information [9]. Every memory block carries an input gate, output gate. The approach of input activations into the memory cell is managed by the input gate, whereas the output gate handles the output flow of cell activations into the remaining of the network. The forget gate is added later to the memory block which is used to determine whatever detail will be thrown-away from the cell state or reset the cell's memory as shown in figure 3[10].

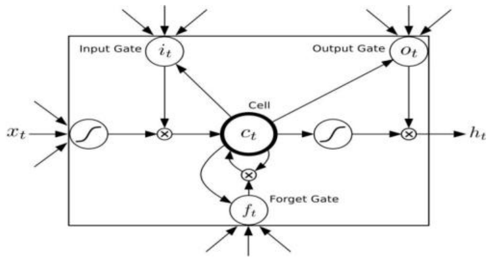

Figure 2: Gates in LSTM

The modern LSTM framework contains a peephole connection from its internal cells to the gates in the same cell to learn the precise timing of outputs [11]. The fundamental equations in LSTM frame-work are listed below which include the validation function, loss function and the functioning of the gates.

$$f_t = \sigma(W_f .[h_t-1, x_t] +b_f) \quad (1)$$

$$i_t=\sigma(W_i.[h_{t-1},x_t]+b_i) \quad (2)$$

$$C_t =tanh(W_C.[h_t-1, x_t]+b_C) \quad (3)$$

$$C_t =f_t * C_t-1 + i_t*C_t \quad (4)$$

$$o_t=\sigma(W0[h_{t-1},x_t]+b0) \quad (5)$$

$$ht = o_t*tanh(C_t) \quad (6)$$

$$f_t = \sigma(W_f .[C_t-1, h_t-1, x_t] +b_f) \quad (7)$$

$$i_t=\sigma(W_i.[C_{t-1},h_{t-1},x_t]+b_i) \quad (8)$$

$$o_t=\sigma(W0.[C_t, h_t-1, x_t] +b0) \quad (9)$$

Where W is the Weight matrix, b denotes the bias vector, σ is the logistic sigmoid function, C denotes the cell state. X is the set of all inputs,i_t is the input,o_t is the output.

The proposed LSTM is an extension to the existing recurrent neural networks for training that also avoids long short-term memory problem. In the proposed model, LSTM with four layers is used. The four layers are:

1. Input a layer.
2. Embedding layer.
3. Feature extraction layer.
4. Output sigmoid layer.

In these four layers, embedding and feature extraction layers are hidden layers. Input for the LSTM is provided to the input layer. In the input layer, the process of cleaning the data like Tokenization, stop words, etc. is performed. In Embedding layer, the data from the input layer is received and the words are embedded. An embedded layer is required because the dataset of large size is used; therefore, there is a need for more efficient representation for the input data than one-hot encoded vectors. The output from the embedded layer is passed to the feature extraction layer. In this layer, for every word, features will be generated. And at the end, the output from feature extraction layer is passed on to output sigmoid layer where it determines whether the given sentence is sarcastic or not.

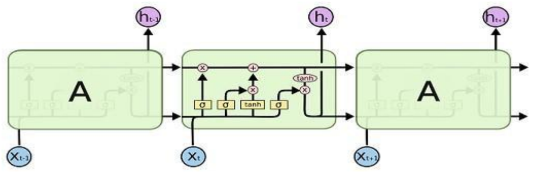

Figure 3: Basic Operations in LSTM

IV. DATASET

Work with 3 datasets for performing experiments that are talked about is done and creation details are explained below:

A. *DATASET-1*

To build this corpus, tweets are extracted from Twitter-API (https://dev.twitter.com). The tweet containing hashtags such as #sarcasm and #nonsarcasm. #sarcasm labelled with 1 and #nonsarcasm labelled with 0. For Deep learning analysis, a larger dataset is required. So, for this, one lakh sarcastic tweet, two lakhs twenty thousand Non-sarcastic tweets and Eleven Thousand Numerical Sarcastic tweet are created.

B. *DATASET-2*

This dataset is divided into a training set containing ten thousand Non-Sarcastic tweets and ten thousand Numerical Sarcastic tweets.

C. *TEST DATASET*

The dataset contains three thousand Numeric Sarcastic and ten thousand Non-Sarcastic data.

V. DATA PRE-PROCESSING

Tweets in the training and test sets are initially pre-processed before passing to the classifier. This is done to the following pre-processing procedures to pattern the tweets in the right way for classifying it.

- Modifies patterns by modifying words like won't to will not, can't to cannot, we'd to we would etc.
- Changes tweets into the smaller case by converting uppercase letters to a smaller case.
- Modifies by changing the words extra white spaces in order to clear out any noise that can have an impact on the classifier's performance.
- Removes stop words like a, an, the, etc.
- Modifies internet website links with URL so links to websites that begin with www.* or http? : //* were changed with URL symbol.
- Stemming and Lemmatization were applied. By stemming on all the words in a text to reduce into its root as an example - 'loving' converted into a root word 'love' with the same meanings, such as government, governance, and governing. Porter stemmer for stemming and word net lemmatize class in the NLTK library is implemented.

VI. RESULTS

In this paper, we have calculated the accuracy for SVM, K-NN, and LSTM Model. For machine learning (SVM and K-NN) and Neural Network (LSTM) are used. Python modules such as Sk-learn and Tensor flow is used for the above models. RBF Kernel for calculating accuracy with the optimization parameters such as C=1(C is a regularization parameter) and grid- search in SVM is used. For K-NN, the neighbour value K=4 for the three datasets is taken. In the LSTM model, the batch-gradient descent technique is applied for training taking optimizer Adadelta with a learning rate of 0.5. After experimenting for SVM, the accuracy has improved to 83.10 percent. Since the dataset is very large for K-NN, accuracy is 72.20 percent and by using LSTM we have got an accuracy of 89.23 percent. By comparing, it can be analyzed that neural network technique LSTM is providing more accurate result than SVM, K-NN.

VII. CONCLUSION AND FUTURE WORK

In this work, an attempt to develop a process that recognizes numerical sarcasm in the tweets is made. The paper introduces the unique concept for analyzing sarcasm that occurs due to the occurrence of numerical portions of the tweets. Numerical sarcasm is a unique case of sarcasm where incongruity occurs between textual and numerical content. Furthermore, Rule-based, Machine learning and Deep learning techniques for numerical sarcasm detection is presented and receives the best overall accuracy of 89.23 percent from LSTM model. Some of the tweets have number present in them, but it is not obligatory that it is sarcastic because of the existence of number. For example- First heart attack in 6 years. Forgot how much fun they are, in this tweet the sarcasm is occurring by reason of the word fun present by the end of the tweet. A large-scale sentiment and Sarcasm detection on social media, tweets, blogs etc. can be performed which contains numbers.

REFERENCES

[1] Lakshya Kumar, Arpan Somani, and Pushpak Bhattacharyya. "Having 2 hours to write a paper is fun!: Detecting Sarcasm in Numerical Portions of Text", arXiv:**1709.01950v1, 2017**.

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

[3] Paula Carvalho, Lus Sarmento, Mario J Silva, and Eugenio De Oliveira, "Clues for detecting irony in user-generated contents: oh...!!its so easy", In Proceedings of the 1st international CIKM workshop on Topic- sentiment analysis for amass opinion, ACM page **5356.**

[4] Dmitry Davidov, Oren Tsur, and Ari Rappoport, "Semi-supervised recognition of sarcastic sentences on Twitter and Amazon", In Proceedings of the fourteenth conference on computational natural language learning. Association for Computational Linguistics pages**107116, 2016**.

[5] Roberto Gonzalez-Ibanez, Smaranda Muresan, and Nina 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, Association for Computational Linguistics pages **581586, 2016**.

[6] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang, "Sarcasm as the contrast between a positive sentiment and negative situation", In EMNLP.Volume **13**, pages **704714**, **2013**.

[7] Mondher Bouazizi and Tomoaki Otsuki Ohtsuki, "A pattern-based approach for sarcasm detection on twitter", IEEE Access4:**54775488, 2016.**

[8] Aniruddha Ghosh and Tony Veale, Fracking "sarcasm using the aneural network", In Proceedings of NAACL-HT. Pages **161169,2016.**

[9] Haim Sak, Andrew Senior, Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large-ScaleAcoustic Modeling", arXiv.org.cs.arXiv:**14021128**.

[10] F.A. Gers, J. Schmidhuber, and F.Cummins "Learning to forget: Continual prediction with LSTM, Neural Computation", vol. **12**, no. **10**, pp. **24512471,2000**.

[11] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber "Learning precise timing with LSTM recurrent networks", Journal of Machine Learning Research, vol. **3**, pp. **115143**, Mar **2003**.

Authors Profile

Aakarsh Mehta is pursuing his Bachelor of Technology degree majorly in Computer Science and Engineering from Amity University Madhya Pradesh, Gwalior. He is currently in his pre-final year.

Sandeep Nigam received the B.E. and M.Tech degrees in Computer Science and Engineering from ITM Universe, Gwalior in 2015 and International Institute of Information Technology, Bhubaneswar in 2018, respectively. Currently, he is working as a Software Engineer at Srikari Impetus Solutions Pvt. Ltd. in Hyderabad.

.