Identification of Sarcasm in Textual Data: A Comparative Study

Pulkit Mehndiratta[†], Devpriya Soni

Jaypee Institute of Information Technology, Noida, India

Citation: Pulkit Mehndiratta, Devpriya Soni (2019). Identification of sarcasm in textual data: A comparative study. *Journal of Data and Information Science*, 4(4), 56–83
DOI: 10.2478/jdis-2019-0021
Received: Sep. 6, 2019
Revised: Nov. 28, 2019
Accepted: Nov. 29, 2019

Abstract

Purpose: Ever increasing penetration of the Internet in our lives has led to an enormous amount of multimedia content generation on the internet. Textual data contributes a major share towards data generated on the world wide web. Understanding people's sentiment is an important aspect of natural language processing, but this opinion can be biased and incorrect, if people use sarcasm while commenting, posting status updates or reviewing any product or a movie. Thus, it is of utmost importance to detect sarcasm correctly and make a correct prediction about the people's intentions.

Design/methodology/approach: This study tries to evaluate various machine learning models along with standard and hybrid deep learning models across various standardized datasets. We have performed vectorization of text using word embedding techniques. This has been done to convert the textual data into vectors for analytical purposes. We have used three standardized datasets available in public domain and used three word embeddings i.e Word2Vec, GloVe and fastText to validate the hypothesis.

Findings: The results were analyzed and conclusions are drawn. The key finding is: the hybrid models that include Bidirectional LongTerm Short Memory (Bi-LSTM) and Convolutional Neural Network (CNN) outperform others conventional machine learning as well as deep learning models across all the datasets considered in this study, making our hypothesis valid.

Research limitations: Using the data from different sources and customizing the models according to each dataset, slightly decreases the usability of the technique. But, overall this methodology provides effective measures to identify the presence of sarcasm with a minimum average accuracy of 80% or above for one dataset and better than the current baseline results for the other datasets.

Practical implications: The results provide solid insights for the system developers to integrate this model into real-time analysis of any review or comment posted in the public domain. This study has various other practical implications for businesses that depend on user ratings and public opinions. This study also provides a launching platform for various researchers to work on the problem of sarcasm identification in textual data.

Originality/value: This is a first of its kind study, to provide us the difference between conventional and the hybrid methods of prediction of sarcasm in textual data. The study also provides possible indicators that hybrid models are better when applied to textual data for analysis of sarcasm.

Keywords Machine learning; Artificial neural networks; Word embedding; Text vectorization; Accuracy

1 Introduction

The advancement in information and communication technologies has a significant impact on our daily lives. The rise of social media platforms has provided rich and correct source of data for researchers across various domains like computer vision and imaging, natural language processing, speech and signal processing, and much more. When it comes to opinion and emotions of the people towards any product or service, evaluation of the correct sentiment is of the essence because biased or wrongly identified comment can lead to ambiguity for users as well as for the system itself. Current analytical systems have very less to zero capability for detection of figurative language. This makes the problem more interesting and relevant to be explored specifically for the areas like social sciences, politics and online markets because people tend to be more creative while using sarcasm and verbal irony to express their sentiment and opinions.

Sarcasm is often referred to as opinion of any individual on very complex matters and has very specific targets as mentioned in (Carvalho et al., 2009). Many people try to express opinions and use sarcasm for getting the attention of other people, as most of the time its intended meaning is literally opposite as mentioned by the user. So, one can say that people use sarcasm as a device that enables them to express the emphatic negative sentiment.

Primitive models that were developed to detect the sarcasm and verbal irony were not capable enough to detect the utterance of it, because they rely on the very low and primitive level features for the detection like number of tokens in a sentence. But, soon enough people in the research community realized that only lexical features will not suffice the purpose. Moving forward on this thought, analysis based on the contextual features of any sentence were evaluated and these techniques performed better compared to the previous models. In an article (Wallace et al., 2014), authors mentioned explicitly that context of utterance is as important as lexical features for identification of verbal irony and sarcasm. They even mentioned, it is no surprise if the same statement is sarcastic in one situation and not sarcastic in others.

At this moment, most of the researchers are focused towards either identification of sarcasm or finding out the sentiments of the user, without taking into consideration

Research Paper

that there may be a possibility that these two tasks are not mutually exclusive rather dependent on each other. Thus, it is important to find out sarcasm in text based data and its implications on the actual sentiment of the sentence. In our experiments, we try to identify the presence of the sarcasm in the labeled datasets available in the public domain. We made our machines to learn from the already existing labeled data and later tried to identify the accuracy of the machines, in terms of recall, precision, F-measure and the overall accuracy of the system. The existing work on the sarcasm mainly focused on the usage of application-specific features like Twitter emoticons, hashtags (#) or unigrams (Gonzalez-Ibanez et al., 2011) and bigram techniques (Barbieri et al., 2014). In another article by (Riloff et al., 2013) authors have proposed a semi-supervised approach to identify sarcasm in textual data. Others (Maynard & Greenwood, 2014) have also proposed many unsupervised techniques of categorizing sarcastic words and sentences from the non-sarcastic ones by the help of mining various patterns. Many others (Davidov et al., 2010; Joshi et al., 2015; Tsur et al., 2010) tried to perform the n-gram technique and evaluate patterns that helped them find out the correct sentiment from the sentence.

In our approach, we are proposing a framework that automatically learns from a corpus of the sarcastic sentences. Once done, we also tried to make alterations in the framework by changing the layout and further tried to assess the results. Our approach uses various word embedding for the purpose of vectorization of each token in the dataset. The results we evaluated, clearly show that by tweaking the hyper-parameters like a number of epochs, spatial dropouts we can improve the overall accuracy of the proposed framework also. The major outcomes from this article are as follows:

- This is first of its kind study to best of our knowledge, where we are using hybrid models of deep learning for sarcasm detection.
- Unlike only applying the model directly, we have evaluated the impact of the different hyper-parameters to tune the settings of framework.
- The models which have performed exceedingly well in the domains like Image Processing or Computer Vision has been applied to the new area of Natural Language Processing (NLP). And we have tried to make these models learn from various datasets.

The rest of the paper is divided into four major sections: Section 2 discusses the literature review of this domain. The datasets and its parameters have been discussed under section 3 along with various models and its functionalities. Section 4 is totally dedicated to the analysis of the results we obtained from various models. In the end, section 5 conclude this article discussing the various future aspects and research directions related to this domain.

2 Related work

The detection of sarcasm in textual data has gained much-needed momentum in the past decade. Its significance is not only limited to human-machine interactions but also play an important role in the analysis of the sentiment of any given piece of textual data. In a study conducted by (Jorgensen et al., 1984), it was explained that whenever figurative meaning is used as an opposite to the literal meaning of the utterance of any word or sentence it arise sarcasm. In another study by (Clark & Gerrig, 1984), authors mentioned that sarcasm is induced into any sentence or word by deliberately canceling the indirect negative message and replacing it with the one which is implicated. In the year 2003 for the first time, sarcasm was defined with the help of six tuples that consisted of *Speaker, Listener, Context, Utterance, Literal, and Intended Prepositions* (Ivanko & Pexman, 2003). Further, they studied the cognitive aspects of the processing of sarcasm.

Many other authors have already addressed the problem of identifying sarcasm and tried to solve it using various rule based and statistics based methods. Studies proposed by Joshi et al. (2015), Barbieri et al. (2014), and Gonzalez-Ibanez et al. (2011) uses features like unigrams and other pragmatics features. In few other studies by Davidov et al. (2010) and Riloff et al. (2013), they have used the patterns related to style of writing as well as the patterns related to disparity of the text.

In other studies related to the short message service named Twitter[®], Maynard and Greenwood (2014) and Liebrecht et al. (2013) used features which are specific to the application itself like hashtags (#) and emoticons.

Rajadesingan et al. (2015) tried to build a system that was capable enough to identify sarcasm in the text. The approach proposed by them was based on the behavioral and psychological sciences. The technique used many tools specifically designed for computer linguistics like Part-Of-Speech (POS) tagger, various lexicons. Also, the user profile and posts were also evaluated to model various behavioural patterns like mood, style of writing and much more. In another approach by Wallace et al. (2015), they have tried to predict sarcasm in the reddit[®] comments using the sub-reddit mentions. But the major limitation of this technique is its dependency on the structure of the sub-reddit posts. Another techniques proposed by Bamman and Smith (2015) used very rich set of features to collect extensive list of contextual information about the text. In a few recent study by Poria et al. (2016), representing authors tried to evaluate the embeddings related to the sentiment

^① <https://twitter.com>

^② <https://reddit.com>

Research Paper

expressed and also the personality traits of the speaker. Some user-specific representations of the embeddings have also been discussed in (Kolchinski & Potts, 2018).

The identification of sarcasm is not only limited to text but has also been estimated and evaluated in speech using harmonics-to-noise ratio by Cheang and Pell (2008). In another approach by Rockwell (2000), the vocals and tones of the sarcastic speech were evaluated. A few years later in a study (Tepperman et al., 2006), the spectral features of the sound were tested for both inside and outside of the context to detect sarcasm. A hadoop based framework has also been proposed by Bharti et al. (2015) to process any tweet and identify the presence of sarcasm in it.

Lately, researchers in the field of NLP have used techniques based on artificial neural networks like convolutional neural networks (CNN). Authors have tried to apply deep learning in many domains like in (Wang et al., 2018) where they try to identify alcoholism using Convolutional Neural Networks. In a study proposed (Zhang et al., 2016), authors have tried to identify sarcasm in a tweet posted on Twitter using the contextual features of the post itself. The work is not limited to this, Felbo et al. (2017) tried to come-up with a new model which is pre-trained for the extraction of the sentiment from a sentence using neural networks and later used it to detect sarcasm along with the personality traits of any person. Some work has also been performed on the conversational sentences for identification of sarcasm in them using the Long Short Term Memory (LSTM) technique by Ghosh and Muresan (2018).

Hybrid models have also been proposed by many authors to make the process of identification more accurate. In a study by Hazarika et al. (2018), a new idea has been shared with the community named CASCADE. This method is very different in its procedure to use word embeddings for encoding of the various features related to the stylometry of the user and later perform simple classification using CNN model.

This work is related to an evaluation new hybrid frameworks proposed by authors to improve the accuracy and test them under different setting across various datasets. We have tuned various hyper-parameters also to evaluate the impact they have on the overall accuracy of the system. We have used three datasets which are:

- Sarcasm Corpus V2 (Oraby et al., 2017)
- ACL Irony Dataset 2014 (Wallace et al., 2014)
- News Headlines Dataset for Sarcasm Detection (Mishra, 2018)

All the models have been applied to the above mentioned datasets so as to check the correctness of the hypothesis.

3 Preliminaries

3.1 Datasets

3.1.1 News headlines dataset for sarcasm detection

The News headline dataset for Sarcasm detection is a dataset generated by collecting news from two major websites i.e. The Onion[®] that provided the sarcastic version of various events happened around the globe and HuffPost[®] provided the non-sarcastic version of the same news so as to maintain the balance in the dataset. The major advantage of this dataset is it eliminates the noise and inaccuracies posted by them in various Twitter datasets thus making it more robust in nature. Another advantage is the way of writing, as the news blogs and posts are written by professionals and they are very accurate in terms of usage of grammar and syntax, the chances of finding pre-trained embeddings increase many folds compared to normal user-generated content. Most importantly, the news articles and posts are complete in itself unlike Tweets or short messages which requires other tweets for complete understanding of the sarcastic outlook of the text. Each row of the dataset has three attributes associated with it:

- **is_sarcastic**: if the sentence is sarcastic in nature the label of this attribute is 1.
- **headline**: this is the actual headline of the news article
- **article_link**: this provides the link to the original news article, where one can find other useful accompanying material of the analysis.

A snapshot of the dataset is provided in figure 1 having three columns/attributes. The datasets have total 26709 object entries in it figure 2 provides the exact number of labels having 1 and 0 for sarcastic and non-sarcastic headlines respectively. Also we have the word cloud of sarcastic and non sarcastic sentences from the dataset as mention in figure 3 & 4 respectively.

article_link	headline	is_sarcastic
https://www.huffingtonpost.com/entry/versace-black-code_us_5861fbefe4b0de3a08f600d5	former versace store clerk sues over secret 'black code' for minority shoppers	0
https://www.huffingtonpost.com/entry/roseanne-revival-review_us_5ab3a497e4b054d118e04365	the 'roseanne' revival catches up to our thorny political mood, for better and worse	0
https://local.theonion.com/mom-starting-to-fear-son-s-web-series-closest-thing-she-1819576697	mom starting to fear son's web series closest thing she will have to grandchild	1
https://politics.theonion.com/boehner-just-wants-wife-to-listen-not-come-up-with-alt-1819574302	boehner just wants wife to listen, not come up with alternative debt-reduction ideas	1
https://www.huffingtonpost.com/entry/jk-rowling-wishes-snape-happy-birthday_us_569117c4e4b0cad15e64fdcb	j.k. rowling wishes snape happy birthday in the most magical way	0

Figure 1. Snippet of the news headlines dataset for sarcasm detection.

[®] <https://www.theonion.com>

[®] <https://www.huffpost.com>

Figure 2. Graph to show number of sarcastic and non-sarcastic labels.

Figure 3. Word cloud of sarcastic words.

3.1.2 Sarcasm corpus V2

The Sarcasm version 2 dataset is a balanced subset of Internet Argument Corpus (IAC) and is a publicly available dataset. The content of the dataset is text from some quote-response pairs which have been annotated for the presence of sarcasm in them. The three major categories of the quotes present in the dataset are Generic (Gen) which are general quote-response sentences having sarcasm, Rhetorical Questions (RQ) are the questions that implies a humorous assertion and Hyperbole (Hyp) are the ones that portray exaggeration of the reality. The main reason we have (RQ) and (Hyp) category is to make the corpus more diverse in nature, letting us

Figure 4. Word cloud of non-sarcastic words.

explore more about the semantics of Sarcastic and Non-Sarcastic occurrences. The CSV of the data contains following columns:

- **Corpus:** It tells to which corpus GEN, RQ or HYP the particular quote belongs to.
- **Label:** this has information if the particular text is sarcastic or not.
- **ID:** Every quote-response pair has a unique ID associated with it thus, to mark that every pair is independent of each-other.
- **Quote Text:** It is the first part of the dialog to which we have the response post
- **Response Text:** It is the response of the quote, and is annotated for the presence of sarcasm in it.

The dataset has 4,712 instances in total, out of which 3,260 instances are of Generic type, while 582 and 850 are of Hyperbole and Rhetorical Questions type respectively. As mentioned earlier, the dataset is balanced in nature as it has equal number of sarcastic and non-sarcastic argument instances in it.

The actual dataset image is given in figure 5.

Corpus	Label	ID	Quote Text	Response Text
0	GEN	1 GEN_1_0000	First off, That's grade A USDA approved Libera...	Therefore you accept that the Republican party...
1	GEN	1 GEN_1_0001	watch it. Now you're using my lines. Poet has ...	More chattering from the peanut gallery? Haven...
2	GEN	1 GEN_1_0002	Because it will encourage teens to engage in r...	Yep, suppressing natural behavior is always th...
3	GEN	1 GEN_1_0003	Obviously you missed the point. So sorry the h...	I guess we all missed your point Justine, what...
4	GEN	1 GEN_1_0004	This is pure paranoia. What evidence do you ha...	Evidence, I dont need no sticking evidence. Th...

Figure 5. Snippet of the sarcasm corpus V2.

The ACL-2014-Irony dataset was first introduced by (Wallace et al., 2014) in year 2014. The data has been collected from a *reddit*, it is a website on which social news is posted and commented upon. This website has 1,000's of user comments who comment upon any active post and they are referred to as sub-reddits. The data has been manually annotated as sarcastic or not the word cloud of the comments and structural representation of the data is given in figure 6 & 7 respectively.

Figure 6. Word cloud of various ACL dataset comments.

	comment_text	label
	I suspect atheists are projecting their desires when they imagine Obama is one of their number. Does anyone remember the crazy preacher with whom he was associated? \nhttp://www. examiner. com/article/obama-and-wright-throw-each-other-under-the-bus/\nI can understand a career politician in the USA needing to feign belief to get elected, but for that purpose I'd imagine a more vanilla choice of church. \n\n/nHHe's not an atheist. He's not a liberal either.	
	It's funny how the arguments the shills are making here are still so close to the racist remarks the GOP has already admitted to. Always attacking "lazy minorities and young people. " \n\n/n&t;[I]f it hurts a bunch of college kids that's too lazy to get up off their bunks [sic] and get a photo ID, so be it," and "if it hurts a bunch of lazy blacks that wants the government to give them everything, so be it. " \nthe law is going to kick the Democrats in the butt. "	0
	We are truly following the patterns of how the mandarins took over empires, not because of the sword, but because control of the endless paper, and regulations, that do more to stagnate most people's lives, then to do anything productive. \n/nBut then because they don't see what else they can do, they write up even more laws and regulations, that either do nothing, or hinder more freedom and production.	0
	air pressure dropping as altitude goes higher. Less pressure around the exhaust stream (which is high pressure itself), and it expands wider as you go higher/n/nIf you look at the video now, the exhaust stream is far wider than the craft itself	0
	Absolutely. I think we'd be hard pressed to find a website that didn't experience some sort of slowdown after getting 2. 6 million or 5 million visits in 18 hours. Also there are bound to be some glitches when rolling out a program of this scope.	0
	Democrats don't know how to manage money? Shocking!	1
	It's not like automation has eliminated the need for any manual labor since the 40 hour work week was established or anything.	

Figure 7. Snippet of the ACL irony dataset.

This dataset is unlike the above two datasets and is unbalanced in nature with a total of 1,949 reddit posts. Out of these 537 posts are labelled sarcastic, while the remaining 1,412 posts are non-sarcastic in nature. As this dataset was among the first few datasets that were available publicly for the research purposes, it has a very limited scope and training can not be improved after a certain limit.

3.2 Word embeddings

The way to represent any word in the form of a vector is done with the help of word embedding. This can be understood in a way that words which have the same meaning or synonyms are close to each other when represented in a vector space. The main goal is to apprehend some sort of relationship between words which can be in terms of sense, morphology, milieu or context. Thus, by representing words in space, the dimension of the textual data can also be reduced which can be very large otherwise as each word is considered as an independent dimension. One more reason to use word embeddings is to encode words into numerical form because by default computers do not understand words very well like they comprehend numerical data. Approaches based on deep learning architecture require data to be in numerical form hence, the importance of word embeddings cannot be ignored. Many word embeddings have been proposed till date and we have used three most common word embeddings in our study to evaluate the frameworks proposed.

3.2.1 Word2Vec

Word2Vec is a word embedding that takes text corpus as input and generates the vectors as output. This works in a manner, where vocabulary is constructed from the training data and then vector representation is learned from it. The file generated is collection of the vectors generated and can be used directly for many machine learning and natural language processing applications.

This was first introduced by Mikolov et al. (2013), and finding the closest word on the basis of the distance was the key measure in this algorithm. This technique of representing words to vector is also helpful for converting phrases and sentences together and has many real time applications related to it. One example to understand the working of this embedding is: it tries to find out if a man is complementing to woman then king will be complementing a queen as shown in figure 8.

Vector Representation of Word in Space

Figure 8. Representation of words as vectors in space.

Research Paper

The quality of the generated vectors is very important for any application. Thus, many hyper-parameters are tuned and setting which are complex are altered to get best vectors from the embedding implementation. The authors of the model have designed a simple test also that tells the quality of the vectors generated. Performance of the system can be evaluated in terms of speed and accuracy. Further, this can be improved by tweaking the hyper-parameters with various possible options. The available choices are as:

- **The architecture:** It has two major architectures to follow: **skip-gram** and **Continuous Bag-Of-Words (CBOW)**
- **The training algorithm:** Two options are available while choosing the training algorithm i.e. **Hierarchical Softmax** and **Negative Sampling**
- **Sampling of words:** Increasing the sub-Sampling of the words increases the overall accuracy and speed of the system.
- **Dimensions:** It is often considered to have more than one dimension as it improves the system efficiency
- **Context window size:** The size of the context window is around 10 for skip-gram and 5 for CBOW technique

3.2.2 GloVe

GloVe or Global Vector for Word Representation is an unsupervised learning algorithm to obtain vector representation of words. It was introduced by Pennington et al. (2014) to the word in 2014. Unlike Word2Vec, it provides good contribution to generate the global statistics of any corpus given to it. It has no context window to obtain any local context. Rather, it learns from word-context/word co-occurrence matrix which has statistics of the whole corpora.

GloVe is a log-bilinear model that has weighted least square errors. The model words on simple observations that co-occurrences of any two or more words have a potential for some form of meaning. The final resultant is a linear substructure of words in a vector space. This can be very well understood by considering an example: Say, we have two words **ice** and **steam** and we have to find out the co-occurrence probabilities of these two words along with other words. Then as ice is more frequently occurs with solid and steam is more frequent with gas. Also, they both are frequently related to water. The main training objective of this representation is to learn vectors such that the logarithm of probability of words co-occurrence is equal to the dot product of these vectors. The overall performance of word vectors generated via GloVe performs better than those generated by Word2Vec from same corpus. The authors of this technique also proposed a new method to calculate the

weighted least square regression where, the weighting function (X_{ij}) is introduced in cost function of the model as in equation 1:

$$J = \sum_{i,j=1}^V f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \quad (1)$$

3.2.3 fastText

On the path of extending the Word2Vec, Bojanowski et al. (2017) introduced a new embedding named *fastText*. This technique of generating the embedding is better as most of the existing embeddings do not consider the morphology of the words and assign a different vector to each word. The technique uses continuous skip-gram (Mikolov et al., 2013) which calculates the maximum log-likelihood as given in equation 2, but with negative sampling and bag of n-grams characters representatio of each word.

$$\sum_{i=1}^T \sum_{c \in C_t} \log p(w_c | w_t) \quad (2)$$

The modified version of finding the log-likelihood using skip gram is given in equation 3 and the scoring function to obtain score from sub-word model is given in equation 4.

$$\sum_{i=1}^T \left[\sum_{c \in C_t} l(s(w_t, w_c)) + \sum_{c \in C_t} l(-s(w_t, n)) \right] \quad (3)$$

$$s(w, c) = \sum_{g \in G_w} Z_g^T V_c \quad (4)$$

The vector of the word is generated by performing the summation of these n-grams words. This method is more efficient and fast, allowing to learn and train better from a large corpus. Also, it allows sharing of the representations across words and making more consistent representations for words which occur rarely.

3.3 Machine learning models

Machine learning is a branch of computer science where computers enhances its capabilities to learn and improve overtime without any external interventions or explicit programs. Most of the machine learning programs focus on the ways to access data and observe it for finding out patterns in them. The algorithms in machine learning are categorized as:

1. Supervised Algorithms: These set of algorithms learn from the already existing labelled data to predict future values or events. The dataset available for learning the patterns is known as training data, while the dataset on which these patterns are applied for prediction is known as testing data.

2. Unsupervised Algorithms: In these set of algorithms, no training or any labelled data is present to learn about the patterns in the dataset. These algorithms are mostly used to find out inferences from the datasets and describe the hidden structures which are present in the unlabeled data.
3. Semi-Supervised Algorithms: These algorithms fall somewhere between supervised and unsupervised algorithms. Here a very small training data is present to learn the patterns and algorithms try to improve on the accuracy gradually.
4. Reinforcement Algorithms: These are the learning methods, where system interacts to produce actions and discover rewards and errors. These methods are used to provide an ideal behaviour to achieve maximum performance for software agents and machines. In simple words, this can be considered as feedback based algorithms.

The models we have used in our study are as follows:

3.3.1 Naive Bayes technique

Naive Bayes algorithms are set of supervised learning algorithms which are derived from the Bayes theorem. The name Naive is given to this class of algorithm because they execute with an assumption that all features are independent of each other, but in real-world scenarios this is not the case. Bayes theorem states the following relationship as given a class variable y and various features say x_1 till x_n dependent on it:

$$P(y | x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n | y)}{P(x_1, \dots, x_n)} \quad (5)$$

As we consider $P(x_1, \dots, x_n)$ are considered as constant, the classification rule can be given as:

$$P(y | x_1, \dots, x_n) \propto P(y) \prod_{i=1}^n P(x_i | y) \Rightarrow \hat{y} = \underset{y}{\operatorname{argmax}} \propto P(y) \prod_{i=1}^n P(x_i | y) \quad (6)$$

The major advantage of using Naive Bayes algorithms is that they are extremely fast compared to other more complex methods. But on one end where it is considered to be a above average classifier, it is not a very good estimator making it not very suitable for predicting values.

We have used the Gaussian Naive Bayes algorithm where the maximum likelihood of the features is calculated as:

$$P(x_1, \dots, x_n) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \quad (7)$$

3.3.2 Decision tree

Decision trees is a classification or regression technique which comes under the category of supervised learning. It works on a simple protocol to learn the rules which have been inferred from the various features of data and apply them on predicting the target value. They have many advantages compared to other classification algorithms such as:

- They are simple to understand and interpret.
- Very less data preparation is required. Other techniques need large amount of data pre-processing before any inferences can be made.
- The height of the tree has logarithmic complexity, the cost of accessing any tree is very less compared to other models.
- Tree can be applied easily to numerical as well as categorical data.
- Tree uses white box modeling, thus any situation can easily be understood or explained by the boolean logic. Unlike black box modeling where interpreting any thing is not easy.
- Its accuracy does not vary much even if the parameters or data somewhat violates the actual model inferences.

3.3.3 Random forest

Random forest is an ensemble technique which means it is based on divide-and-conquer approach and used for both classification and regression purposes. Like decision tree algorithm, this technique has many trees, each working on a random sample of data to generate rules and inferences. For the purpose of classification each tree contributes towards the voting process and the class which is most popular among all is selected.

3.4 Deep learning techniques

As we are moving in to the era of Artificial Intelligence (AI), more complex and intricate problems are being faced by researchers. Deep learning is a powerful tool for bridging the gap between human thinking and computer logic. Many applications surrounding us like Amazon's Echo dots, self-driving cars, image and object detection, real-time translation are evident examples of its usability and applicability in our daily lives.

In a nutshell, deep learning is an extension to already existing machine learning algorithms used for problems where we have very large volume of data and a lot of computation is required to solve the problem. The word deep is associated to it because there are large number of hidden layers of computation and interconnections of these layers creates a network known as deep learning network.

Research Paper

The major difference between conventional machine learning approaches and deep learning is that, the process of extraction of features and classification of the data is knotted together with each other and algorithm is capable enough to learn from the features itself. Whereas, in traditional machine learning algorithm we need to explicitly mention about the various features that will enable the system to classify the object or data among various labels.

Few techniques we have used in our study are as follows:

Figure 9. Typical CNN applied on textual data.

3.4.1 Convolutional neural network

Convolutional neural networks (CNN) was originally developed by LeCun et al. (1998) for classification of hand written numbers and is now considered as to go to system for any type of work related to classification or feature extraction of image. It belongs to the class of feed forward artificial neural network (ANN). The underlying mechanism uses the multi-layer perceptron model for no to least possible pre-processing. When this model first applied to the field of natural language generation and processing the results looked very promising.

The working of typical CNN consists of four major layers:

1. The Convolutional Layer
2. Pooling or Down-sampling Layer
3. Flattening Layer
4. Fully Connected Layer

To understand how these layers work in case of textual data can be explained in figure 9. Here the results from each convolution will produce a different pattern and by varying the size of the filters and merging it with outputs, we can have multiple size patterns to work with. The patterns can be expressions thus, the major advantage of using CNN is that it can identify these patterns in the sentences irrespective of their positions.

3.4.2 Long Term Short Memory

Humans need not learn everything from the start whenever they want to do some work or to read or write. The human memory is persistent along with the learning learned by them. The conventional neural networks are not capable to perform and learn in the same fashion. Thus, recurrent neural network addressed this issue and solved the problem by allowing information to persist in the network. These networks have loops which help to pass the information from one step of the network to the next step. But, these networks suffer from a problem of long-term dependencies because as the network grows more context and knowledge is required to answer any question and as the gap between relevant and not useful information increases. This leads to wrong contexts and incorrect results. To solve this problem of long-term dependency we use Long Term Short Memory networks or just (LSTMs). They were first introduced by Hochreiter and Schmidhuber (1997), later it was improved and applied to a large variety of problems. They solve the problem of long-term dependencies as remembering information over a long period of time is part of their default behavior.

The architecture of a LSTM module is given in figure 10. The forget, input and the output state can be calculated as shown in equations:

$$i_t = \sigma(w_{i_x} x_t + w_{i_h} h_{t-1} + b_i) \quad (8)$$

$$f_t = \sigma(w_{f_x} x_t + w_{f_h} h_{t-1} + b_f) \quad (9)$$

$$o_t = \sigma(w_{o_x} x_t + w_{o_h} h_{t-1} + b_o) \quad (10)$$

$$\hat{C} = \tanh(w_{c_x} x_t + w_{c_h} h_{t-1} + b_c) \quad (11)$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C} \quad (12)$$

$$h_t = \tanh(C_t) * o_t \quad (13)$$

Figure 10. Structure of LSTM module.

3.5 Hyper-parameters

Hyper-parameters are the properties of any model that actually administrates the entire functionality and training process. Various hyper-parameters are present in a single model that can be tuned and results can be improved. Selection of good hyper-parameters plays an important role in the success of any neural network architecture, as it can improve the overall accuracy of the model. The exhaustive list of hyper-parameters any neural model has are:

- Learning Rate
- Number of Epochs
- Hidden Layers
- Hidden Units
- Rate of Dropout
- The size of the training data
- Optimizer
- Activation Function

Out of all these above mentioned hyper-parameters we have considered two parameter which we have tuned for our proposed framework. They are as:

3.5.1 Spacial dropout

The meaning of term dropout in neural network means dropping out units (i.e. neurons) during the training process at random and they further not passed in either forward or backward pass of the neural network. The main reason to perform this operation to prevent over-fitting of the model.

In a fully connected layer most of the parameter get occupied due to which neurons develop a sense of dependency among other neurons i.e. if one go then others will follow. This nature of neurons do not allow them to neurons to learn own their own and over-fitting of the training data starts. Figure 11, tells how dropout works in a neural network.

Dropout forces neural network to learn more robust and unique features from the data during training. It also doubles the number of iterations required to converge in a dataset.

Figure 11. Network before and after applying the dropout.

3.5.2 Epochs

Epoch is one entire cycle of forward as well as backward movement of data from the neural network. The importance of passing the entire dataset from the network make the learning more optimized and accurate. The number of epochs to be used for the learning should not be tuned to the perfection as this leads to over-fitting and it affects the accuracy of the system. Also, there is no formal technique or method to find the appropriate number of epochs to train a model thus, hit and try technique has to be adopted to find correct number of epochs. In case of text, the system accuracy can show correct results for epochs as low as 2 or 4. We have evaluated our model for 2, 4, 8, and 16 epochs.

Other parameter like optimizer and activation function can also be altered to see the change in the results. For our study we have used ADAM optimizer (Kingma & Ba, 2014) and Sigmoid activation function. ADAM is an adaptive learning rate or stochastic gradient descent based optimizer which try to optimize every parameter at individual level with different rates.

4 Methodology and result

This section discusses the various techniques applied to the datasets under consideration and results we obtained to prove our hypothesis. We have performed our experimentation with both machine learning models that are not neural network-based and also with the models which use the neural networks and concepts of deep learning for the purpose of identification of sarcasm in textual data.

The ensemble models which include Naïve Bayes, Decision Tree, and Random Forest have been applied and compared for the overall accuracy of the techniques on three data sets discussed in section 3.1. Before starting with the actual classification of textual data, a series of steps are involved in pre-processing and training of the data. We have used the standard 80:20 split for training and testing purposes respectively. A flowchart of all the steps involved in the classification is given in figure 12. The results obtained for these three algorithms for the datasets we have used for this study are given in figure 13.

Figure 12. The system architecture for shallow machine learning algorithms.

These models are very basic and are not capable to capture the features like the semantics, word co-occurrences. These models can only be used to find out the decisions on simple tokens generated after pre-processing and feature extraction. From the results, we can easily infer that random forest classifier gives the highest accuracy because it uses multiple decision trees (i.e. 1000 in our case) to take decisions and predicts the label for being sarcastic or non-sarcastic in nature. It can

Figure 13. Results from shallow machine learning models.

also be observed from the bar-plot that Naïve Bayes classifier has given the least system accuracy because it considers every word as an independent feature and does not include the probability of co-occurrence and dependency of words on other words.

Moving on to the process and modeling techniques used in deep learning models we have observed that deep learning has outperformed the shallow machine learning techniques. Also the tuning of the parameters has provides us great in-sites about the datasets and overall accuracy measures of the system. As discussed in section 3.4 about the techniques and strengths to which they work, we observed that these techniques try to find out more complex and novel features from the data of limited training set, and everything is done without any sort of human interventions. These models have tried to learn from the previously extracted features set and to further improve the overall accuracy of the system.

Stirred by the intelligence and ability of deep learning algorithms to learn and explore features at this rate, we have used these techniques in our model. We have not used the only the conventional deep learning algorithms like Convolutional Neural Networks (CNNs) and Long-Term Short Memory (LSTM) a variant of Recurrent Neural Network (RNN), but we also introduced new hybrid model that takes the combination of these two techniques to include the features of both algorithms.

The hybrid model tries to bridge the gap between learning of special and temporal features of the textual data. Our hypothesis states that the information generated in the LSTM module will help CNN module to classify the data more accurately. The

Research Paper

LSTM network can retain knowledge from the network for a long time due to its resolution of long-term dependencies thus, adding this to the classification process by CNN will add to the features and entropy related to them. More words which are uncommon but can play a critical role during classification can be used as feature for classification. Also, patterns related to words that have contrasting sentiment related to them can benefit the task at hand.

Word embeddings are play the most important role when we have to perform any task related to natural language processing using deep learning. The word embeddings we have used in this study are: Word2Vec, GloVe, fastText. The system architecture of this model setting is given in figure 14. Also, all the parameter settings have been listed in table 1. The plated diagram of two hybrid frameworks are given in figure 15 and 16.

Figure 14. The system architecture with Deep Learning Models.

Figure 15. Plated framework for CNN-LSTM architecture.

Figure 16. Plated framework for LSTM-CNN architecture.

Table 1. Parameter list for our models under training and testing.

Parameter	Set-Value
Filters	64
Kernel	3
Embedding Dimension	300
Epochs	2, 4, 8, 16
Activation Function	Sigmoid
Batch Size	128
Word Embedding	Word2Vec, GloVe and fastText
Pool Size	2
Dropouts	0.15, 0.25, 0.35:ConvNet, 0.25:Bi-LSTM
Optimizer	Adam

We have compared the results of our proposed hybrid frameworks with conventional frameworks and found that these models perform well compared to conventional machine learning model. Also, the accuracy of these system varies with hyper-parameter tuning across all three datasets. Some observations that we inferred from the table 2, and 4 are important for us to understand.

Few inferences that we inferred from the table 2, 3, and 4 are important for us, to understand the behavior of our proposed frameworks and its functionality also. We have assessed the results on the basis of accuracy of the system, we also collected the data for other key performance indicators like F-Score, Precision and Recall of the system. It can be observed that the LSTM-CNN model outperforms other framework for all the three datasets. Also, we can infer that, due to scarcity of the data in case ACL irony 2014 dataset, the overall accuracy of the system tends to decrease after running the machine for 16 epochs. This is due to the over-fitting of the graph and system is not able to learn any further. This same thing happened with Sarcasm Corpus V2 also.

Research Paper

Table 2. Results obtained from ACL 2014 Irony Dataset.

Word2Vec					
Dropout	Epochs	Accuracy (%)			
		CNN	LSTM	CNN-LSTM	LSTM-CNN
0.15	2	54.34	55.28	56.98	58.07
	4	58.31	58.38	59.93	59.62
	8	59.62	59.32	60.87	60.33
	16	60.12	60.08	61.23	62.23
Avg (0.15)		58.097	58.265	59.752	60.062
Avg (0.25)		58.575	58.927	60.747	60.897
Avg (0.35)		59.292	59.327	60.545	61.132
GloVe					
Avg (0.15)		58.73	58.83	58.15	60.53
Avg (0.25)		59.78	59.76	58.202	59.817
Avg (0.35)		59.29	59.43	57.882	59.922
fastText					
Avg (0.15)		59.69	60.25	58.647	60.831
Avg (0.25)		59.26	58.87	59.99	60.32
Avg (0.35)		59.66	59.01	59.74	59.87

Table 3. Results obtained from News Headlines Dataset.

Word2Vec					
Dropout	Epochs	Accuracy (%)			
		CNN	LSTM	CNN-LSTM	LSTM-CNN
0.15	2	80.8	80.6	80.7	80.8
	4	80.5	81	81.1	80.4
	8	79.6	80.6	80.3	80.7
	16	78.1	80.3	78.3	81.23
Avg (0.15)		79.75	80.63	80.1	80.7825
Avg (0.25)		79.9	80.85	80.075	80.865
Avg (0.35)		80.1	80.88	80.125	80.8825
GloVe					
Avg (0.15)		81	81.18	81.025	81.275
Avg (0.25)		81.1	81.21	81.2	81.25
Avg (0.35)		81	81.56	81.175	81.6
fastText					
Avg (0.15)		80.96	81.38	80.6125	80.65
Avg (0.25)		81.23	81.26	80.975	81.45
Avg (0.35)		81	81.06	81	81.075

Table 4. Results obtained from Sarcasm Corpus V2 Dataset.

Word2Vec					
Dropout	Epochs	Accuracy (%)			
		CNN	LSTM	CNN-LSTM	LSTM-CNN
0.15	2	56.55	58.67	58.57	58.99
	4	56.76	56.55	56.55	57.61
	8	56.93	57.08	56.07	57.87
	16	57.25	57.08	55.69	55.69
	Avg (0.15)	56.873	57.345	56.72	57.54
Avg (0.25)		57.185	57.44	56.757	57.565
Avg (0.35)		57.033	57.238	56.17	57.267
GloVe					
Avg (0.15)		58.637	59.167	58.74	58.94
Avg (0.25)		59.075	59.082	58.775	59.277
Avg (0.35)		59.127	58.952	58.91	59.225
fastText					
Avg (0.15)		58.655	58.86	58.28	59.155
Avg (0.25)		59.205	59.085	58.285	59.497
Avg (0.35)		59.212	59.13	58.7	59.277

The impact of the dropout can be seen clearly in the result as by increasing it to 0.35 from 0.15, has resulted in more accuracy of the system. This is due to the reason that, the system tries to learn from each nodes of the network rather finding same nodes in the network again. From the results it can also be observed that models involving LSTM, tries to learn context and utilize this information to a great extent, thus improving the performance of the model. This brings us to the conclusion of our study.

Comparison with the other studies is given in the table 5, 6, and 7 that proves our hypothesis that deep learning methods are more accurate and efficient when it comes to textual data analytics.

Table 5. Comparison table for News Headline Sarcasm Dataset.

Technique	Accuracy (%)
NBOW (Logistic Regression with neural Words)	0.724
NLSE (Non-Linear Subspace Embedding)	0.72
CNN (Convolutional Neural Network)Kim (2014)	0.742
Shallow CUE CNN ((Context and User Embedding Convolutional Neural Network)Amir et al. (2016)	0.793
<i>Our Proposed technique</i>	<i>0.816</i>

Table 6 has the comparison of ACL Irony Dataset 2014 with the other baseline techniques. The comparison is done on the basis of precision and recall parameters.

Research Paper

Table 6. Comparison of ACL irony Dataset 2014.

Features	Recall (%)	Precision (%)
Baseline(BoW)Wallace et al. (2015)	0.288	0.129
NNP (Noun Phrase)	0.324	0.129
NNP + Subreddit	0.337	0.131
NNP + subreddit + sentiment	0.373	0.132
<i>Our Proposed technique</i>	0.489	0.472

Table 7. Comparison of Sarcasm Corpus Version 2.

Technique	Recall (%)	Precision (%)
Baseline (SVM) Oraby et al. (2017)	GEN	0.75
	RQ	0.73
	HYP	0.63
Our proposed technique	GEN	0.72
	RQ	0.71
	HYP	0.68

Also, how any feature adds more weight to the overall performance of the system has been evaluated. The calculation of the recall and precision of any system is done as given in equation 14 and 15 respectively.

$$recall = \frac{true_positives}{true_positives + true_negatives} \quad (14)$$

$$precision = \frac{true_positives}{true_positives + false_negatives} \quad (15)$$

Table 7 provides the comparison of the results with paper by Oraby et al. (2017). The authors have used SVM and we have compared the results of our proposed technique against the same dataset. The results are fractionally improved in terms of precision and recall as compared to the baseline provided.

5 Conclusion

The rise of social media has provided means to the masses to talk and communicate with each other. But the language people tend to use on these platforms is sometimes consist of some abusive words, slang, sarcastic words and sentences. These things not only make the classification process difficult but also increases the chances of ambiguity and confusion. We have presented some hybrid techniques that can be applied to identify the presence of sarcasm in the text. Also, our proposed model performs well across all the datasets thus, validating our hypothesis since the beginning of this study.

Many features have been learnt by the system and framework proposed, deep learning methods clearly outcast the shallow modeling techniques that identify sarcasm in the textual data. Vectorization of the textual data is one of the most important step to be followed carefully, as small variation can lead to large change in the accuracy of the system. The tuning of the hyper-parameters has really turned the tables around the accuracy of our system. All the experiments done on the datasets are very intensive and complete in itself to give best possible results. This study can further be extended to work on the creation of more advanced and easier to deploy algorithms, embeddings and correct predictions. The process of usage of emojis, typo-graphical images had led to more problems than solutions. New challenges have posted to the existing system thus, we need to work upon and come up with an idea of making the systems robust and dynamic enough that they can adapt to the situation in real time and predict the presence of sarcasm in textual data.

References

- Amir, S., Wallace, B.C., Lyu, H., & Silva, P.C.M.J. (2016). Modelling context with user embeddings for sarcasm detection in social media. arXiv preprint arXiv:1607.00976.
- Bamman, D., & Smith, N.A. (2015). Contextualized sarcasm detection on twitter. In Proceedings of the Ninth International AAAI Conference on Web and Social Media.
- Barbieri, F., Saggion, H., & Ronzano, F. (2014). Modelling sarcasm in twitter, a novel approach. In Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, 50–58.
- Bharti, S.K., Babu, K.S., & Jena, S.K. (2015). 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, 1373–1380. ACM.
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135–146.
- Carvalho, P., Sarmiento, L., Silva, M.J., & De Oliveira, E. (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, 53–56. ACM.
- Cheang, H.S., & Pell, M.D. (2008). The sound of sarcasm. Speech Communication, 50(5), 366–381.
- Clark, H.H., & Gerrig, R.J. (1984). On the pretense theory of irony. American Psychological Association, 113(1), 121–126.
- Davidov, D., Tsur, O., & Rappoport, A. (2010). Semi-supervised recognition of sarcastic sentences in twitter and amazon. In Proceedings of the Fourteenth Conference on Computational Natural Language Learning, 107–116. Association for Computational Linguistics.
- Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. arXiv preprint arXiv:1708.00524.

Research Paper

- Ghosh, D., & Muresan, S. (2018). "with 1 follower I must be AWESOME : P.". Exploring the role of irony markers in irony recognition. In Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, Stanford, California, USA, June 25–28, 2018.
- Gonzalez-Ibanez, R., Muresan, S., & Wacholder, N. (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, 2, 581–586. Association for Computational Linguistics.
- Hazarika, D., Poria, S., Gorantla, S., Cambria, E., Zimmermann, R., & Mihalcea, R. (2018). CASCADE: Contextual sarcasm detection in online discussion forums. CoRR, abs/1805.06413.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
- Ivanko, S.L., & Pexman, P.M. (2003). Context incongruity and irony processing. *Discourse Processes*, 35(3), 241–279.
- Jorgensen, J., Miller, G.A., & Sperber, D. (1984). Test of the mention theory of irony. *Journal of Experimental Psychology: General*, 113(1), 112.
- Joshi, A., Sharma, V., & Bhattacharyya, P. (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, 2, 757–762.
- Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
- Kingma, D.P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
- Kolchinski, Y.A., & Potts, C. (2018). Representing social media users for sarcasm detection. arXiv preprint arXiv:1808.08470.
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P., et al. (1998). Gradient-based learning applied to document recognition. In Proceedings of the IEEE, 86(11), 2278–2324.
- Liebrecht, C., Kunneman, F., & van Den Bosch, A. (2013). The perfect solution for detecting sarcasm in tweets #not. Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA).
- Maynard, D., & Greenwood, M.A. (2014). Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis. In Proceedings of LREC 2014. ELRA.
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. Workshop on ICLR.
- Mishra, R. (2018). Github news headlines dataset for sarcasm detection: High quality dataset for the task of sarcasm detection. Retrieved from <https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection>
- Oraby, S., Harrison, V., Reed, L., Hernandez, E., Riloff, E., & Walker, M. (2017). Creating and characterizing a diverse corpus of sarcasm in dialogue. arXiv preprint arXiv:1709.05404.
- Pennington, J., Socher, R., & Manning, C.D. (2014). Glove: Global vectors for word representation. In Proceedings of EMNLP, 1532–1543. ACL.
- Poria, S., Cambria, E., Hazarika, D., & Vij, P. (2016). A deeper look into sarcastic tweets using deep convolutional neural networks. arXiv preprint arXiv:1610.08815.

- Rajadesingan, A., Zafarani, R., & Liu, H. (2015). Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 97–106. ACM.
- Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., & Huang, R. (2013). Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 704–714.
- Rockwell, P. (2000). Lower, slower, louder: Vocal cues of sarcasm. *Journal of Psycholinguistic Research*, 29(5), 483–495.
- Tepperman, J., Traum, D., & Narayanan, S. (2006). “yeah right”: Sarcasm recognition for spoken dialogue systems. In *Proceedings of the Ninth International Conference on Spoken Language*.
- Tsur, O., Davidov, D., & Rappoport, A. (2010). A great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*.
- Wallace, B.C., Choe, D.K., & Charniak, E. (2015). Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference* (pp. 1035–1044). (ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference; Vol. 1). Association for Computational Linguistics (ACL).
- Wallace, B.C., Choe, D.K., Kertz, L., & Charniak, E. (2014). Humans require context to infer ironic intent (so computers probably do, too). In *Long Papers* (pp. 512–516). (52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference; Vol. 2). Association for Computational Linguistics (ACL).
- Wang, S.-H., Muhammad, K., Hong, J., Sangaiah, A.K., & Zhang, Y.-D. (2018). Alcoholism identification via convolutional neural network based on parametric relu, dropout, and batch normalization. *Neural Computing and Applications*, 1–16.
- Zhang, M., Zhang, Y., & Fu, G. (2016). Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical Papers*, 2449–2460.

This is an open access article licensed under the Creative Commons Attribution-NonCommercial-NoDerivs License (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).