CASCADE + BERT: Using Context Embeddings and Transformers to Predict Sarcasm

Stanford CS224N Custom Project

Langston Nashold
Department of Computer Science
Stanford University
lnashold@stanford.edu

Abstract

Sarcasm is a form of verbal irony, in which a writer or speaker states the opposite of their intended message in order to mock or show contempt. Sarcasm is commonly used online, and being able to detect sarcasm is crucial to understand and classify online pieces of text. In this work, we attempt to classify Reddit comments (from the SARC corpus) as either sarcastic or sincere. Sarcasm is heavily reliant on context – information about the world or a text author. Therefore, we propose a BERT model, augmented with three different context types: discourse context, user context, and community context. After testing this model, we found that the addition of user context shows increased performance compared to training without user context. However, incorporating community context did not improve performance. We were able to achieve a maximum accuracy of 75.8 percent.

1 Key Information to include

- Mentor: Rachel Gardner

2 Introduction

Sarcasm is a form of verbal irony, in which a writer or speaker states the opposite of their intended message to mock or show contempt. For example, a speaker might say "I *totally* agree with you," to convey that they disagree with someone. Sarcasm can be present in both spoken and written language.

Detecting sarcasm is key to understanding a passage of text – if one interprets all statements at face value, they might receive the exact opposite message that a sarcastic speaker intended. Furthermore, sarcasm detection is an important subset of sentiment classification, or determining the tone or mood of a piece of writing. The online communities like Facebook or Twitter on which sentiment classifiers make predictions commonly employ sarcasm. Because sarcasm can flip the polarity of text sentiment relative to its literal meaning, sentiment classification algorithms to recognize sarcasm in order to perform optimally. Typically, the task is set up as a binary classification problem, in which a model receives a written piece of text (sometimes including metadata) as input, and produces a binary label as output (sarcastic or nonsarcastic).

Sarcasm detection is also an especially difficult task, for several reasons. First, it is commonly performed using written, not oral, datasets because of relatively difficulty collecting spoken data. Consequently, any model or human attempting to make predictions cannot rely on the tonal or facial expressions that indicate a speaker is being sarcastic. Furthermore, sarcasm often requires a great deal of topic-specific or speaker-specific context to understand correctly. For example, to classify the sentence "I think Reagan was one of the best Presidents of our lifetime" correctly, a model or human needs to understand (a) who Reagan was and (b) what the writer's personal views on Reagan are. This can be an especially difficult when the topic at hand is niche. As a consequence of these factors, sarcasm detection is challenging even for humans. On one dataset, humans were only able to label sarcastic sentences with 82% accuracy.

Because sarcasm is so heavily dependent on context, many prior works attempt to leverage context to make prediction. For example, the CASCADE model created embeddings for every author and every online community in their dataset before training. When the text is in context of an ongoing discussion (e.g. a Twitter thread), other works attempt to train on both the text and surrounding discussion [1]. Indeed, Joshi et al. found that incorporation of context is one of three major milestones in sarcasm was incorporating context beyond the target text into decision making [2].

In addition, Vaswani et al. recently proposed the transformer architecture, which relies on attention units to establish long-range dependencies within a text [3]. Transformer-based architectures have been shown to achieve remarkable performance on many NLP tasks, including question answering, translation, and text classification [3]. BERT is one variety of transformer that was pre-trained on a very large text-corpus, and has has been shown to achieve very high accuracy NLP benchmarks, while simultaneously requiring minimal fine-tuning for specific tasks [4]. BERT has also been previously shown to perform well on sarcasm detection tasks [1, 5].

In this work, we attempt to determine if augmenting BERT with additional context beyond the target text improves performance. We use a dataset taken from Reddit, an online forum that users use to discuss a wide variety of topics. We augment BERT with three additional types of context: the discussion context (other comments the user is in conversation with), the user context (other comments a particularly user has made), and the community context (or how other people in the same community talk). The user and community context are based on embedding vectors created by the CASCADE model.

We found that the augmenting the model with user embeddings caused a one percentage-point improvement over training the model without context; however, the subreddit embeddings were not shown to have any significant performance impact.

3 Related Work

The first attempts at sarcasm detection involved hand-engineered sets of rules to determine if a piece of text is sarcastic. For example, Riloff et al. noticed that sarcasm was often the result of a negative situation combined with a positive sentiment, and trained a classifier based on this pattern [6]. Another rule-based approach attempted to to capture instances of hyperbolic expressions (i.e. "Oh wow!") that occur together [7]. Other early strategies have involved hand-engineered feature sets based on word shape, the presence of synonyms, punctuation, frequency of rare words, etc. [2]. These rule-based or feature-based approaches were often combined with classical machine learning methods, such as logistic regression on support vector machines [2**?**].

However, recently the trend in natural language processing is to use less feature engineering, and let deep-learning models learn more appropriate representations in a "hands-off" approach. For example, the architecture proposed by Khotijah et al. uses a Long Short Term Memory model to make predictions on a set of input embeddings [8]. Another approach, proposed by Potomias et al., fed the hidden outputs of the roBERTa transformer model through a bidirectional LSTM to achieve strong performance on the Reddit dataset [5]. These "hands-off" models have been shown to make high-accuracy predictions.

In the last four years, a specific class of deep learning model, transformers, have become especially prominent both in sarcasm detection and natural language processing as a whole [3]. In the 2020 Sarcasm Detection shared task at the Figurative Language Processing workshop, the top performing models all used some combination of transformers, generally a variation of a transformer model [9]. For example, Lee et al., the top performing team, implemented a classifier composed of BERT, followed by a BiLISTM and NeXtVLad, (an alternative to mean or max pooling) [10]. Other approaches included using roBERTa large, and off-the-shelf BERT models.

Another common pattern in top-performing prior works is various attempts to leverage context. Indeed Joshi et al. claims that leveraging context has been one of the major milestones in sarcasm detection [2]. For example, Lee et al. proposed a novel mechanism to incorporate the discussion context of a piece of text (or the chain of comments a response is responding to) [10]. They use an ensemble model to determine what the optimal length of context to include in the model is. With this model, they were able to easily outperform other models that did not use context.

The model CASCADE, proposed by Hazarika et al., also leveraged multiple types of context to achieve state-of-the-art performance (at the time), without the use of transformers [11]. CASCADE operates on two types of context: user context (other comments a user has posted) and community context (other comments posted in the same community). They concatenate all comments a user has posted, and feed them into the ParagraphVector algorithm to create a user embedding vector. They do a similar thing for each community. They then train a CNN on these two embedding vectors along with the actual target text.

4 Approach

Given the success of both transformer-based methods and context-based methods, we sought to understand if existing transformer models could be improved by concatenating the transformer output with an embedding vector representing a target's context.

The the model used was a pre-trained BERT model [4]. BERT is a transformer with a pre-trained encoder trained on the Wikipedia corpus. Relying on bi-directional context, it trained using masked language models to predict random tokens on an unlabeled input corpus. At the time of its publication, BERT achieved state-of-the-art performance on several NLP benchmarks, including raising the GLUE score by 7.7 percentage points from the previous state of the art.

The BERT model was augmented with three different types of context: discussion context, user context, and community context. The dataset used was based on Reddit, meaning every labeled target text was a response to a different ancestor comment (see Section 5.1). We name this the "discussion context." This discussion context and the target text are used as the input to the BERT classifier, according following formula, where the `CLS` token denotes a classification task and the `SEP` token denotes boundaries between sentences:

`<CLS>` ANCESTOR COMMENT `<SEP>` TARGET COMMENT `<SEP>`

The user context are the comments by the same author of a text, and the community context is based on all comments posted in the same community. These were generated using the same method proposed by Hazarika et al. in their CASCADE model [11]. Unlike Hazarika et al., however, the proposed architecture uses the embeddings as part of a transformer instead of a convolutional network.

To create a user embedding, every comment by a particular user was concatenated into one large document. This document was fed into the ParagraphVector algorithm, an unsupervised algorithm that creates a fixed length embedding based on a variable-length input [12]. Two separate versions of this algorithm were created, one to represent the user's writing style and another to represent the user's personality. The personality embedding also relied on a pre-trained CNN designed to recognize Big-5 personality traits. These two representations were combined using canonical correlation analysis, or CCA, which infers correlations between the two vectors to produce the final user context embedding. A similar process was applied to every community (or subreddit) in the datase to create the community context embeddings.

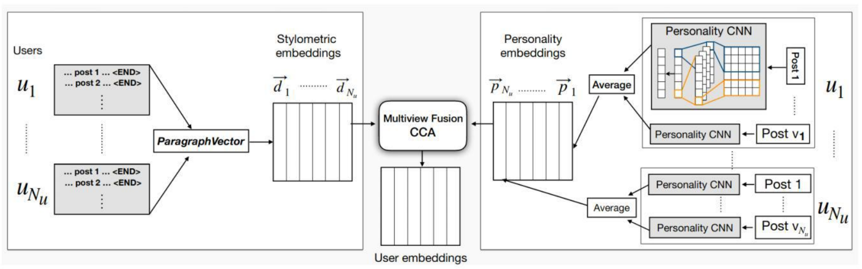

Figure 1: Embedding Generation

The final model tokenized the input sentence and its discussion context, which was passed through BERT. BERT's output was concatenated with the two user and community context embeddings, and finally fed through through two linear layers and a softmax layer to create the final prediction (see Figure 3). Since the subreddit embeddings were a very high magnitude, they were passed through a BatchNorm layer to normalize them. Although the embeddings were generated using pre-existing work, concatenating with the transformer output is a novel approach.

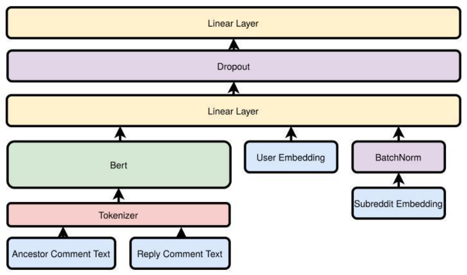

Figure 2: Proposed Model Architecture

Several sarcasm detection models have been able to achieve better performance by leveraging an LSTM layer on top of a transformer [1, 5]. Potomias et al. used an "RCNNRoberta model", which fed the hidden output of a RoBerta transformer into a Bi-LSTM before the max-pooling layer. This architecture was shown to achieve better performance than an out-of-the-box BERT classifier [5]. Therefore, this architecture was also implemented in order to determine if it would have higher performance than a BERT + context embeddings approach.

We used several baselines for comparison. The first is random chance: since the dataset was balanced, it's expected that a model making random predictions would have an accuracy of 75%. The second baseline was a bag-of-words model proposed and implemented by Khodak et al., in which a logistic regression classifier was trained on a vector composed on the word counts of the input [13]. The third baseline was human performance, also measured by Khodak et al. Finally, a BERT model without any additional context embeddings served as the final baseline.

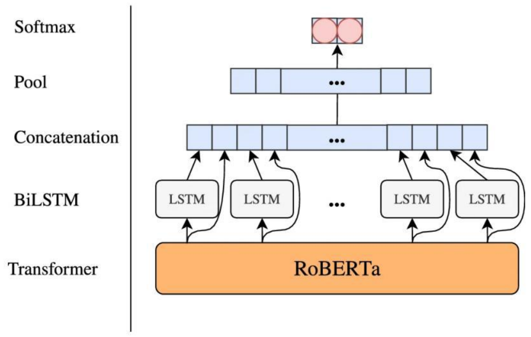

Figure 3: Potomias et al. Architecture

5 Experiments

5.1 Data

We used the Self-Annotated Reddit Corpus, which is one of the largest datasets for sarcasm detection [13]. SARC was scraped from Reddit, an internet discussion board with over 30 million active users [14]. Reddit is organized around a series of subreddits, each of which is dedicated to a different theme or community. For example, /r/politics contains discussion about politics. On each subreddit, users post comments, which other users can reply to and discuss.

SARC is structured as a list of comment threads. Each comment thread has a list of ancestor comments, a list of replies, and binary labels for each reply (sarcastic or not sarcastic). One datapoint might look something like the following:

ANCESTOR COMMENT 1: "I wonder how this announcement will impact Rockstar Gaming's bottom line"
ANCESTOR COMMENT 2: "I think we will have to wait until the game is released to see major changes."
REPLY COMMENT 1: "They seem to be taking their time to iron out all the bugs" (**Label:** non-sarcastic).
REPLY 2: "Yeah, a major gaming company would obviously never promise more than they can deliver" (**Label:** Sarcastic).

Each comment, both ancestor and replies, also has associated metadata in addition to the raw text. This includes the date is was posted, the amount of likes it received, and the board it was posted in. All text was lower case, and had no formatting (e.g. italics). The dataset is self-annotated. Reddit users have a convention of appending "/s" to a comment to indicate sarcasm, so comments with "/s" were labeled as sarcastic.

The balanced version of this dataset was used, which contained equal number of sarcastic and non-sarcastic responses. It contains 2.6 million datapoints. We subdivided the given test set into a validiation and a hold-out set.

Some preprocessing steps were also necessary before the data could be used by our model. Each reply was tokenized along with its most direct ancestor, as described in Section 4. The label for the reply was converted into a one-hot vector. Finally, every user and community context embeddings in the dataset was pre-computed, and at training time, they were queried using the response metadata. Therefore, each datapoint fed into the model was a tuple of (tokenized text, user embeddings, community embeddings). Since not every response was associated with a valid user, the data had to be filtered and subsequently re-balanced.

5.2 Evaluation Metrics

The evaluation method used was accuracy. This method was chosen because the data set had an even amount of datapoints in each class, so there was no class imbalance problem. This was also the primary metric used by other models training on this dataset, which allows for easy comparison to prior works.

5.3 Experimental details

Several model configurations were tested in order to determine if additional user or discourse embeddings would improve performance. We tested BERT without using any embeddings, BERT with just user embeddings, BERT with just community embeddings, and BERT with both context embedding types. All models tested used discourse context. We also testedand implemented the model proposed by Potomias et al., to see if it had the potential to furhter improve performance.

Hyperparameters were tuned using random search. Ultimately, it was found that the following combinations of hyperparameters produced the best results.

Hyperparameter	**Value**
Bert Size	Bert Base
Number of Bert Layers	12
Learning Rate	0.001
Decay Rate	None
Feed Forward Layer Size	128
Weight Decay	1e-5
Number of Epochs	3

Figure 4: Selected Hyperparameters

Initially, the model was overfitting to the training data. To solve this, several strategies were employed. First, a weight decay term was added to introduce L2 regularization into the model. Furthermore, dropout was employed in the linear layers in order to reduce overfitting. Finally, filtering for user embeddings before balancing the data let us use a much larger portion of the available data.

Empirically, it was found that the performance stopped increasing after three epochs, so the model was only trained for three epochs. It took roughly two hours to train one model.

5.4 Results

Model	Accuracy
Random Baseline	50
BERT Baseline	74.80
Bag of Words Baseline	73.2
Average Human Performance	81.6
With User Embeddings	75.78
With Community Embeddings	74.85
With Both Embeddings	75.10
Potomias et al.	72.80
State-of-the-Art	79.0

Figure 5: Accuracy of various models on SARC dataset

As we can see, using user embeddings improves the performance by around one percentage point over a vanilla BERT classifier. This makes sense, as having more context about a usersf personality and writing style seems like it would help determine sarcasm. However, this was not a huge difference, implying that the pre-trained BERT model is still doing the brunt of the work.

Surprisingly, however, using subreddit embeddings does not improve the model's performance. This is despite Hazarika et al. finding that using subreddit embeddings actually significantly helped improve performance of their model [11]. This could be because the transformer based architecture proposed is much less receptive to the additions of these embeddings than the CNN architecture proposed by Hazarika et al. Also, since the subreddit embeddings were added after the BERT model, the remaining two linear layers might not have had the complexity necessary to succesfully learn anything based on the subreddit embeddings. Finally, the BatchNorm layer added could have also hindered the performance - perhaps it would have been better to normalize the data as a part of pre-training, rather than within the model. These are all avenues for future experiments.

Furthermore, replicating the architecture specified in Potomias et al. did not have a significant improvement over the BERT baseline [5]. This was also unexpected, as Potomias et al. reported very high accuracies for their model. In addition, Potomias et al. reported an accuracy of 77.00 for an out-of-the-box BERT classification model, which was also higher than our experimental results. The reason for this discrepancy is unclear; however, they used a much lower learning rate (1e-5) than us (1e-3), although our model performed even worse when replicating their hyperparameters.

It's also interesting that the bag-of-words baseline is remarkably high, compared to both much more sophisticated technques and human performance. This could indicate that the presence of certain words is a relatively strong indicator of sarcasm, and could be what the BERT-based model is using to make predictions.

6 Analysis

Looking at the confusion matrix, we can see that the errors are evenly distributed across the two classes. This means it's not over-predicting sarcastic or non-sarcastic datapoints. However, it is slightly more accurate predicting sincere datapoints.

The attention weights for two example sentences were also visualized (Figure 7). We can see some attention heads pay attention to all of the words. However, a few key words are weighted more heavily, such as patriot, american, and mosque. It might be inferred that these words are crucial to understanding the sentence, and thus, making a prediction. In the right example, we can also see that smart and reasonable are weighted highly. This makes sense, as sarcasm is often the presence of an over-the-top positive phrase in the presence of a situation or context that's actually negative. This would also explain why the bag-of-words baseline performed so well.

We can also do further analysis by looking at which examples were predicted incorrectly or correctly. Take the following example, which the model predicted incorrectly:

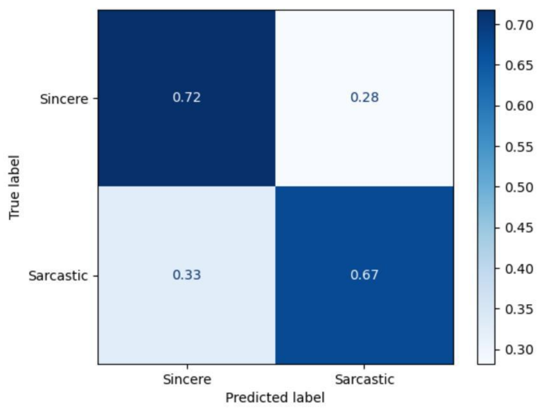

Figure 6: Confusion Matrix

Ancestor 1 Which song would make a good plot for a movie?
Response "christmas shoes" (**Label:** Sarcastic)

To correctly predict this, the model requires the world context that most humans might not know – that the song "christmas shoes" is widely considered cheesy. Furthermore, the user did not give us any information beyond the two word phrase, meaning the model would have to make a prediction based solely on world context it did not have. In general, lack of world context was one of the largest sources of errors.

Furthermore, the dataset is self-labeled. It's taken from Reddit users who annotate their comments with "/s" to indicate sarcasm. However, each user might have a different definition of sarcasm, as compared to data consistently annotated by a consistent group of people. For example, consider the following comment:

Ancestor: What will we call Bill Clinton if Hillary is elected president?
Response: I can think of a few names (**Label:** Sarcastic)

This comment was labeled as sarcastic, but it's really using an idiom, not verbal irony, to convey insulting sentiment towards former President Bill Clinton. Therefore, many people wouldn't consider it sarcasm. In essence, this is a major problem with self-labelling – sarcasm does not necessarily have a strict definition and different users might have different personal definitions.

Another problem with self-labelling is users might not feel the need to label a comment they feel is clearly sarcastic, although the authors of the dataset took some measures to prevent this [13]. Nevertheless, there are some obviously sarcastic examples that weren't labeled, such as the following:

Ancestor: God i hate reddit because of s** like this... bring the downvotes all you libt***s - i just don't give a shit anymore... grow the f*** up.
Response: Eloquent (**Label:** Non-Sarcastic.)

Although these mislabeled data points are in the minority of the errors, they are still present and are also inevitable given the dataset collection method.

There are a wide variety of issues a sarcasm detection model must overcome to produce good predictions. Even given unlimited information, some datapoints are still impossible to predict because they depend solely on a users personal opinions. Therefore, we can see sarcasm detection is a very difficult task for a human or model to perform well on; in other words, the Bayes error is very high. Future work might focus on eliminating some of these sources of error, either through leveraging more context or spending more time cleaning the dataset.

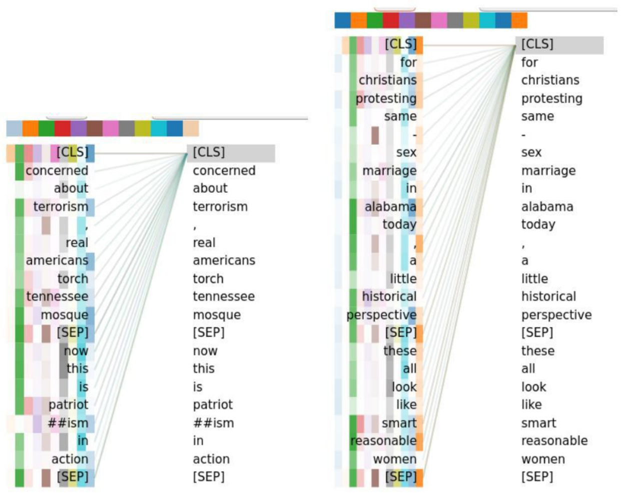

Figure 7: Attention Head visualization

7 Conclusion

Sarcasm detection is an important task both on it's own and as a part of semantic classification. However, it is also an especially difficult task. To successfully detect sarcasm in a piece of text, a model needs a wide range of context about the discussion the text was produced in, the author of the text, the platform the text is posted on, and even about the world at large. Even humans can have a very difficult time determining if a piece of text is sarcastic or not.

In this paper, we sought to improve the amount of context a model could leverage by augmenting an existing BERT model with embeddings representing the author of the post and the community they were posting to to an existing BERT model. Although the community embeddings did not improve performance, we show that adding user embeddings could create a small performance boost in the resulting predictions.

In the course of the project, I learned about how to build an end-to-end NLP model, putting techniques like early stopping, dropout, L2 regularization, and hyperparameter tuning into practice. I learned about some of the development techniques that help create a successful model. Finally, I got experience doing background research of existing papers, which helped me successfully come up with new ideas to achieve better results. I look forward to doing more NLP research in the future.

The embeddings created have the potential to be applied to future work as well. The model used was relatively simple, but in many competitions like SemEval, large transformer models are combined with LSTMs and other architectures. Perhaps similar embeddings could be used along with these more complex models, to achieve even greater performance this was shown in this paper. Alternatively, it was demonstrated that many failures were due to not having enough context about the real world. For example, in the "christmas shoes" what the model really needs to determine is if the user thinks "christmas shoes" is a good song. This problem could be ameloriated by leveraging even more world context, i.e. from Wikipedia or social media, to create a representation of whether the topics mentioned in a text generally have positive or necessary associations open-domain sentiment classification model. Finding ways to leverage even more world context is another promising avenue for future research.

References

[1] Cynthia Van Hee, Els Lefever, and Véronique Hoste. 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, June 2018. Association for Computational Linguistics.

[2] Aditya Joshi, Pushpak Bhattacharyya, and Mark J. Carman. Automatic sarcasm detection: A survey. *ACM Comput. Surv.*, 50(5), September 2017.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] Rolandos Alexandros Potamias, Georgios Siolas, and Andreas . Georgios Stafylopatis. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320, Dec 2020.

[6] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 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, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[7] S. K. Bharti, K. S. Babu, and S. K. Jena. Parsing-based sarcasm sentiment recognition in twitter data. In *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1373–1380, 2015.

[8] Siti Khotijah, Jimmy Tirtawangsa, and Arie A. Suryani. Using lstm for context based approach of sarcasm detection in twitter. In *Proceedings of the 11th International Conference on Advances in Information Technology*, IAIT2020, New York, NY, USA, 2020. Association for Computing Machinery.

[9] Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. A report on the 2020 sarcasm detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 1–11, Online, July 2020. Association for Computational Linguistics.

[10] Hankyol Lee, Youngjae Yu, and Gunhee Kim. Augmenting data for sarcasm detection with unlabeled conversation context. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 12–17, Online, July 2020. Association for Computational Linguistics.

[11] Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. CASCADE: Contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[12] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China, 22–24 Jun 2014. PMLR.

[13] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A large self-annotated corpus for sarcasm. In *Proceedings of the Linguistic Resource and Evaluation Conference (LREC)*, 2018.

[14] Combined desktop and mobile visits to reddit.com from july to december 2020. https://www.statista.com/statistics/443332/reddit-monthly-visitors/.