EPiC Series in Computing

Volume 81, 2022, Pages 121–131

Proceedings of 11th International Congress on Advanced Applied Informatics

Sarcasm Detection with External Entity Information

Xu Xufei and Kazutaka Shimada

Kyushu Institute of Technology, Fukuoka, Japan
xufei.xu874@mail.kyutech.jp, shimada@ai.kyutech.ac.jp

Abstract

Sarcasm is generally characterized as ironic or satirical that is intended to blame, mock, or amuse in an implied way. Recently, pre-trained language models, such as BERT, have achieved remarkable success in sarcasm detection. However, there are many problems that cannot be solved by using such state-of-the-art models. One problem is attribute information of entities in sentences. This work investigates the potential of external knowledge about entities in knowledge bases to improve BERT for sarcasm detection. We apply embedded knowledge graph from Wikipedia to the task. We generate vector representations from entities of knowledge graph. Then we incorporate them with BERT by a mechanism based on self-attention. Experimental results indicate that our approach improves the accuracy as compared with the BERT model without external knowledge.

1 Introduction

Sarcasm is generally characterized as ironic or satirical that is intended to blame, mock, or amuse in an implied way. Sarcasm expresses a negative sentiment without negative words. In other words, the meanings of sentences/utterances with sarcasm are the opposite of what writers/speakers said. Recognizing whether a sentence/utterance contains sarcastic meanings is essential to downstream applications for correctly understanding speakers' intended sentiments and beliefs [1]. In the last decade, sarcasm detection has attracted a considerable interest from computational linguistics researchers. The task is usually defined as a binary classification task; a sentence is sarcastic or non-sarcastic. With the development of deep learning and the increasing availability of datasets, sarcasm detection has achieved remarkable advancements in the last few years. Recently, pre-trainined language models (LM) are widely used in many natural language processing (NLP) tasks including sarcasm detection. These models have achieved remarkable successes. The LMs are pre-trained on unlabeled data and then applied in other tasks with a feature-based approach [2] or a fine-tuning approach [3]. In these models, BERT [4], which is based on Transformer [5], is one of the most successful pre-trained models at present. It has also produced the state-of-the-art result in sarcasm detection. By pre-training with large amouts of data, advanced LMs with sufficiently deep architectures are able to capture complex linguistic phenomena and perform better in understanding language than previously approaches such as CNN and RNN.

However, in order to understand complex linguistic phenomena including sarcasm, additional knowledge that supports sophisticated reasoning is required. Hiai and Shimada [6] have

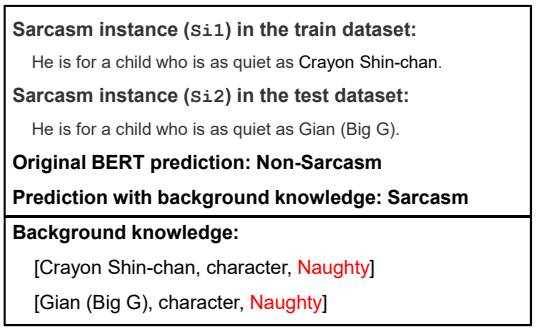

Figure 1: Two similar sarcastic sentences. For the correct prediction, models need to understand the background knowledge.

reported the effectiveness of role relations expressed by relation vectors for a sarcasm detection task. In this paper, we argue that integrating background knowledge with pre-trained LMs contributes to the improvement of the sarcasm detection accuracy, especially knowledge about entities. The entity knowledge are extracted from knowledge bases. We explain our motivation with an example in Figure 1. Assume that `Si1` appears in the train dataset and the label is "sarcasm". The word "Crayon Shin-chan" is a comic character name and he is well-known as a naughty child. Although the whole meanings of `Si1` may be "non-sarcasm", the character name boosts up the sarcasm likelihood of this sentence. In this situation, the models such as BERT probably predicts `Si2` in the test dataset to "non-sarcasm" because the quiet child often contains positive meaning. However, the keypoint of `Si2` is the word "Gian (Big G)" and he is also a naughty comic character. If we have background knowledge about `[Crayon Shin-chan is a character and naughty]` and `[Gian (Big G) is a character and naughty]`, we can estimate that the label of `Si2` is "Sarcasm" from the label of `Si1` and the relation between two characters. This example shows the importance of background knowledge in sarcasm detection. In this paper, we incorporate such extra knowledge with BERT to obtain the better performance in sarcasm detection tasks.

We focus on knowledge about entities used in sarcastic sentences. We introduce an entity embedding as additional knowledge into our sarcasm detection model based on BERT. Our aim is to utilize both linguistic features obtained by deep language models and knowledge extracted from knowledge base to find a better approach for sarcasm detection. We use the knowledge graph of Wikipedia as knowledge base for capturing famous names or events appearing in sarcasm. Instead of directly using symbolic facts, we generate the embedding of the knowledge graph into distributed representations. Then, we integrate the embedding as knowledge that is relevant not only locally to the reading text but also globally about the whole knowledge graph.

2 Related Work

Sarcasm detection is usually considered as a classification task which is to classify texts to sarcasm or non-sarcasm.

2.1 Feature-based approaches

Early researches have focused on finding effective features for sarcasm detection. Lexical features, such as interjections and punctuation, were effective for the task [7]. Carvalho et al. [8] have reported that oral/gestural expressions represented by emoticons and special keyboard characters are useful in sarcasm detection. Liebrecht et al. [9] have discussed the role of hyperbole in sarcasm detection. Bharti et al. [10] have proposed a rule-based approach based on patterns and they also highlighted the importance of hyperbole in sarcastic texts. Riloff et al. [11] have found that a common form of sarcasm on Twitter consists of a positive sentiment contrasted with a negative situation. They developed a sarcasm recognizer based on contrast patterns to identify sarcasm in tweets. Joshi et al. [12] have discussed the role of incongruity for sarcasm detection. The negative situation or incongruity is sometimes hidden behind a reference such as famous names: e.g., "Crayon Shin-chan" in Figure 1. We utilize the latent information in texts for sarcasm detection tasks.

Machine learning is also applied in sarcasm detection. Rajadesingan et al. [13] have proposed a behavioral modeling-based approach. They employed SVM (Support Vector Machine), logistic regression, and decision tree as machine learning models in sarcasm detection. Yang et al. [14] trained a SVM-based model for detecting sarcastic expressions from general tweets. Ptacek et al. [15] have created datasets from Czech and English tweets and then proposed a sarcasm detection model with several features.

2.2 Deep learning based approaches

With the rise of deep learning models, neural network based models have been applied to sarcasm detection tasks: e.g., LSTM by Ghosh and Veale [16], a bi-directional gated recurrent neural network (Bi-LSTM) by Zhang et al. [17], and a convolutional neural network (CNN) model on embedding of content and user by Amir et al. [18]. Tay et al. [19] have employed an attention-based neural model for sarcasm detection. Hazarika et al. [20] have proposed a novel hybrid sarcasm detector, CASCADE, that models both content and contextual information. Dubey et al. [21] have proposed a model that converted sarcastic texts into non-sarcastic interpretation by using encoder-decoder, attention, and pointer generator architectures.

Dubey et al. [22] have found that unusual numerical portion, such as 'use 3 minutes to decide the outcome of a football match' is a clue for sarcasm and applied numerical portion as a feature in sarcasm detection. The research shows that we need some knowledge or common sense which does not appear in texts to understand sarcasm.

Recently, pre-trained language models such as BERT, have achieved a remarkable success in various tasks. The models have also been applied to sarcasm detection tasks. Kalaivani and Thenmozhi [23] have used BERT for sarcasm detection. The BERT-based model obtained a better result as compared with LSTM and other traditional machine learning approaches such as SVM and Navie Bayes. Dong et al. [24] have compared models in the BERT family: BERT, RoBERTa, and ALBERT. The RoBERTa-Large model gives the highest F1-scores for both the target-oriented and context-aware models. We also use BERT as the basic model for our sarcasm detection method with external knowledge.

2.3 Approaches with external knowledge

External knowledge have been recently utilized for obtaining a better performance in various tasks. Bin et al. [25] have integrated sub-graphs in a medical knowledge graph with a pre-trained language model for knowledge generalization. Robert et al. [26] used knowledge graphs

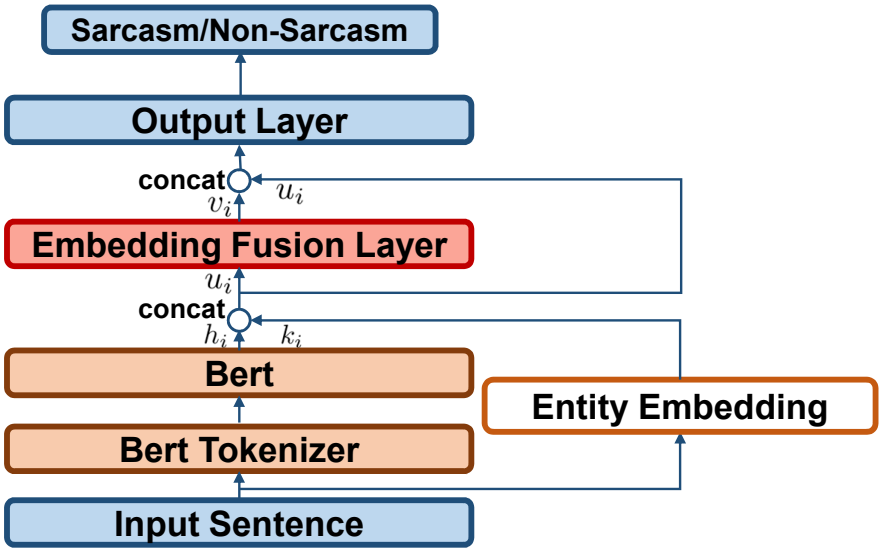

Figure 2: Overall architecture of our approach.

for fact-aware language modeling. Deepanway et al. [27] have used commonsense knowledge for utterance-level emotion recognition in conversations . In sarcasm detection, external knowledge was also effective. Hiai and himada. [6] have proposed a RNN model with a relation vector about role information in texts: 'boss and subordinates'. The model outperformed the models without the relation vector. Inspired by these successful related studies, we incorporate external entity information with our sarcasm detection method.

2.4 Corpus for sarcasm

Early work on building sarcasm corpora relied on small-scale manual annotation. Filatova [28] have constructed a sarcasm corpus from Amazon product reviews by using crowdsourcing. Davidov et al. [29] have reported the strong influence of hashtags in sarcasm detection. There are several corpora about sarcasm: tweet corpora by Gonzalez-Ibanez et al. [30], Rajadesingan et al. [13], and Bamman and Smith [31]. More recently, Khodak et al. [32] constructed a large-scale sarcasm corpus. We use the corpus by Khodak for our experiment.

3 Our Approach

Our task is defined as that given parent comments and a response, the system predicts whether the response is sarcastic or not. As showed in Figure 2, our approaches employs

(1) a BERT encoding layer to compute context-aware representations for the parent comments and the response,

(2) an entity embedding layer to obtain knowledge embeddings of each entity in the input sentence,

(3) an embedding fusion layer to fuse knowledge representations into the BERT output , and

(4) an output layer to predict whether the response is sarcastic or not.

3.1 BERT encoding layer

BERT is a transformer-based deep learning model for natural language processing (NLP). It is pre-trained on two tasks from BooksCorpus (800M words) and English Wikipedia (2,500M words): language modeling and next sentence prediction. As a result of the training process, it learns contextual embeddings for words. After pre-training, we can finetune the model with smaller datasets to optimize its performance on specific tasks.

We use BERT for modeling parent comments and the response in this layer. Given parent comments $C_i = \{c_{ij}\}_{j=1}^{m_i}$ and a response $R = \{r_j\}_{j=1}^{n}$, we first pack them into a single sequence:

$$S = [< CLS >, C_1, C_2, ..., C_i, < SEP >, R, < SEP >]$$

where $\{c_{ij}\}$ means the j-th token of the i-th comment and $\{r_j\}$ means the j-th token of the response. $< CLS >$ is a special token which is always added at the beginning of the text. It can be used as the representation of the whole sentence for classification tasks[1].

The input representation of each token s_i in S, namely all of C and R, is constructed as:

$$h_i^0 = s_i^{tok} + s_i^{pos} + s_i^{seg}$$

where s_i^{tok} is the token embeddings for s_i, s_i^{pos} is the position embeddings for s_i, and s_i^{seg} is the segment embeddings for s_i, respectively.

h_i^0 is fed into Transformer blocks in BERT, i.e.,

$$h_i^l = Transformer(h_i^{l-1})$$

In BERT_{base}, the number of layers of Transformer is 12.

3.2 Entity Embedding

We use entity knowledge as the additional information for sarcasm detection. Therefore, we need to detect entities in parent comments and response. For the purpose, we use the Natural Language Toolkit (NLTK)[2]. We also use knowledge graph of Wikidata[3]. Each datum consists of two entities and its edge: e.g., $(entity1, relation, entity2)$.

It is hard to incorporate the symbolic facts, namely the triple, directly to our model with BERT. Therefore, we generate the embedding of the knowledge graph into distributed representations. We apply TransE [33] to the generation. TransE is an energy-based model and aims to learn low-dimensional embeddings of entities. Given knowledge triple $(e1, rel, e2)$, which is a basic component of a knowledge graph K, TransE aims to let the embedding of the tail entity $e2$ be close to the embedding of the head entity $e1$ plus some vector which depends on the relationship rel.

The basic idea of TransE is that the functional relation induced by the rel-labeled edges corresponds to a translation of the embeddings. For example, when $(e1, rel, e2)$ is in a knowledge graph K, there should be $e1 + rel \approx e2$ and $e2$ should be a nearest neighbor of $e1 + rel$. Otherwise, $e1 + rel$ should be far away from $e2$. As an energy-based framework, the energy of a triplet is equal to $d(e1 + rel, e2)$ for a dissimilarity measure d. In this work, we use $L2$-norm

[1] We don't use it this paper.
[2] https://www.nltk.org/
[3] https://www.wikidata.org/wiki/Wikidata:Main_Page

as the dissimilarity measure d. To learn such embeddings, we minimize a margin-based ranking criterion over the knowledge graph:

$$\sum_{(e1,rel,e2)\in K} \sum_{(e1',rel,e2')\in K'_{(e1,rel,e2)}} [\gamma + d(e1 + rel, e2) - d(e1' + rel, e2')]_+$$

where $[\ ]_+$ denotes the positive part, $\gamma > 0$ is a margin hyperparameter, and $K'_{(e1,rel,e2)}$ is made by negative sampling that $e1$ or $e2$ (but not both) is replaced by a random entity:

$$K'_{(e1,rel,e2)} = \{(e1', rel, e2) \mid e1' \in E\} \cup \{(e1, rel, e2') \mid e2' \in E\}$$

For each token in output of the BERT tokenizer, if it belongs to a entity in knowledge, its knowledge embedding k_i will be the embedding of the entity. Otherwise, its knowledge embedding k_i will be a zero vector.

3.3 Embedding Fusion Layer

By concatenating $\{h_i^l\}$ and$\{k_i\}$, we get both context-aware and knowledge-aware representation u_i:

$$u_i = [h_i^l, k_i]$$

To further enable interactions among context embeddings $\{h_i^l\}$ and knowledge embeddings $\{k_i\}$, we employ a self-attention mechanism on token's knowledge-enriched representations $\{u_i\}$. In order to measure a similarity of any two tokens s_i and s_j, trilinear function [34] is employed.

$$r_{ij} = W^T[u_i, u_j, u_i \odot u_j]$$

Here $\odot$ denotes element-wise multiplication and W is a trainable weight parameter. r_{ij} reflects a similarity of s_j and s_i so that we obtain a similarity matrix R constituted by r_{ij}. Then by a row-wise softmax operation on R, we get the self-attention weight matrix A constituted by a_{ij}:

$$a_{ij} = \frac{exp(r_{ij})}{\sum_j exp(r_{ij})}$$

a_{ij} is an attention score reflecting the proportion of influence of s_j on s_i among the influences of all tokens on s_i. Therefore, a_{ij} can also be used as a weight. Then we use a_{ij} to compute how each token s_j interacts with s_i.

$$v_i = \sum_j a_{ij} u_j$$

Finally, the output of this fusion layer becomes

$$o_i = [u_i, v_i, u_i - v_i, u_i \odot v_i]$$

3.4 Output Layer

In output layer, we simply use an average pooling layer, a linear layer, and a softmax function which are usually used in the output layer of BERT-based classification models. Through an average pooling layer, the representation of comments and a response is constructed as :

$$p = \frac{\sum_i o_i}{len(S)}$$

Hyper-parameters	Value
Bert Embedding dimension	768
Bert Padding sequences	200
Knowledge Embedding dimension	50
Batch Size	16
Train Epoch	{3,4,5}
Learning Rate	2e-5
Weight Decay	1e-4

Table 1: Hyper-parameters values.

where $len(S)$ refers to the length of S, which is packed input of model and discussed in Section 3.1. Then a linear layer and a softmax function are applied:

$$q = softmax(W_L^t p)$$

Here q is a two-dimensional vector that one shows the possibility of sarcasm and the other one reflects the possibility of non-sarcasm.

4 Experiments

4.1 Settings

We use BERT$_{Base}$ [4] in our experiments. We implemented the model by Python 3.5 and Pytorch Neural Network API. We use AdamW as the optimizer. We compared several batch sizes, {4, 8, 16, 32} and several learning rates from $1e-6$ to $1e-2$. As a result, we use 16 for the batch size and $2e-5$ for the learning rate in our experiment. Table 1 shows the hyper-parameters' values of our model.

We conducted comparative experiments between a baseline and two our methods. The baseline is a BERT classification model used in related work [24]. By comparing the baseline, we investigate the effectiveness of the external entity information for the task. For our approach, we compared two methods: Ours1 and Ours2. Ours1 is a method without embedding fusion layer. In other words, the method used the concatenated u_i, namely $[h_i^l, k_i]$, in Section 3.3 for the input of the output layer. Ours2 is our complete method that is explained in Section 3. By comparing two our approaches, we investigate the effectiveness of the embedding fusion layer.

We use a subset of sarcasm corpus by [32] to evaluate our approach. The corpus is constructed from reddit. Reddit users have adopted a common method for sarcasm annotation consisting of adding the marker "/s" to the end of sarcastic statements. The label of each comment is determined by this marker. The marker "/s" in the corpus was deleted for the experiment in advance. The corpus is a balanced dataset which is consist of 278,933 instances: the number of sarcasm instances is 139,299 and the number of non-sarcasm instances is 139,634. We divided the dataset to 8:1:1 as training, development, and test data.

The criteria in the experiment are precision, recall, and F1-score as follows :

$$Precision = \frac{TP}{FP + TP}$$

$$Recall = \frac{TP}{FN + TP}$$

	Sarcasm			Non-Sarcasm		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Baseline	0.757	0.725	0.741	0.723	0.761	0.742
Ours1	0.761†	0.724	0.742†	0.730†	**0.767**†	0.748†
Ours2	**0.763**†	**0.736**†	**0.749**†	**0.738**†	0.764†	**0.751**†

Table 2: Experimental result.

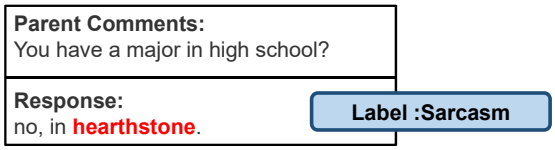

Figure 3: Sarcastic sentences. The word "hearthstone" denotes the name of a famous video game.

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where TP is defined as the number of correctly classified instances as sarcasm, FP is defined as the number of non-sarcasm instances which are misclassified as sarcasm, and FN is defined as the number of sarcasm instances which are misclassified as non-sarcasm, respectively.

4.2 Result and Discussion

Table 2 shows the experimental result. The bold face denotes the best value in each criterion. The dagger (†) denotes that the values are better than the baseline. On F1-score, Ours2, namely our complete method, obtained the best performance. In addition, all values including the precision and recall rates were better than the baseline, namely the method without external knowledge. Ours1 also obtained the better performance although the recall rate of Ours1 was slightly low as compared with that of the baseline. Our2 outperformed Our1 except the recall rate of Non-Sarcasm. This result shows the effectiveness of (1) use of external knowledge for this sarcasm detection task and (2) the embedding fusion layer for the integration of context and knowledge embeddings.

We explain an instances from the experiment to prove the effectiveness of our approach. Figure 3 shows an instance. Here, our method obtained that "hearthstone is a famous video game", from external knowledge. This knowledge boosted up the sarcasm likelihood of this instance. As a result, our method predicted this instance correctly. On the other hand, BERT probably cannot capture this information on the pre-training. In fact, many entities, not only this word, do not exits in the word list of BERT. Therefore, BERT cannot predict this instance correctly. This result also shows the effectiveness of use of external knowledge.

Finally, we discuss the computational cost of our method. Table 3 shows the learning execution time of the baseline and Ours1 in Table 2. We used NVIDIA Quadro RTX 8000 GPU

	Baseline	Ours1
Learning time	approx. 2,500s	approx. 12,100s

Table 3: Comparison of computational costs in training.

with 48GB. These values are the processing time for 223,145 instances in the training data set. By using external knowledge, our method was approximately five times slower learning speed than the baseline based on BERT only. This is a significant problem for our method. Reducing this learning cost is one of the most important future work for our method.

5 Conclusion

In this paper, we proposed an approach using additional knowledge from knowledge bases for a sarcasm detection task. We generated knowledge representations by using TransE and then incorporated them into our method with BERT. As a result, our method can handle context-aware and knowledge-aware embeddings for the task. In addition, we introduced an embedding fusion layer into the process.

In the experiment, our method outperformed a baseline method based on BERT, namely a method without additional knowledge. We confirmed the effectiveness of the embedding fusion layer. From case analysis, we showed the effectiveness of our method.

Future work includes (1) utilization of other knowledge base and (2) evaluation with other datasets.

References

[1] Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. A report on the 2020 sarcasm detection shared task. *arXiv preprint arXiv:2005.05814*, 2020.

[2] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[3] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018.

[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*, pages 4171–4186, 2019.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998—6008. 2017.

[6] Satoshi Hiai and Kazutaka Shimada. Sarcasm detection using rnn with relation vector. *IJDWM*, 15:66–78, 2019.

[7] Roger Kreuz and Gina Caucci. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 1–4, 2007.

[8] Paula Carvalho, Luís Sarmento, Mário J Silva, and Eugénio De Oliveira. Clues for detecting irony in user-generated contents: oh...!! it's" so easy";-. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 53–56, 2009.

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

[10] Santosh Kumar Bharti, Korra Sathya Babu, and Sanjay Kumar Jena. Parsing-based sarcasm sentiment recognition in twitter data. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1373–1380, 2015.

[11] 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, 2013.

[12] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 757–762, 2015.

[13] Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 97–106, 2015.

[14] Raj Kumar Gupta and Yinping Yang. CrystalNest at SemEval-2017 task 4: Using sarcasm detection for enhancing sentiment classification and quantification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 626–633, 2017.

[15] Tomás Ptácek, Ivan Habernal, and Jun Hong. Sarcasm detection on czech and english twitter. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 213–223, 2014.

[16] Aniruddha Ghosh and Tony Veale. Fracking sarcasm using neural network. In *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 161–169, 2016.

[17] Meishan Zhang, Yue Zhang, and Guohong Fu. Tweet sarcasm detection using deep neural network. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 2449–2460, 2016.

[18] Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. Modelling context with user embeddings for sarcasm detection in social media. *CoRR*, abs/1607.00976, 2016.

[19] Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. Reasoning with sarcasm by reading in-between. *CoRR*, abs/1805.02856, 2018.

[20] Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. Cascade: Contextual sarcasm detection in online discussion forums. *CoRR*, abs/1805.06413, 2018.

[21] Abhijeet Dubey, Aditya Joshi, and Pushpak Bhattacharyya. Deep models for converting sarcastic utterances into their non sarcastic interpretation. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 289–292, 2019.

[22] Abhijeet Dubey, Lakshya Kumar, Arpan Somani, Aditya Joshi, and Pushpak Bhattacharyya. "when numbers matter!": Detecting sarcasm in numerical portions of text. In *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 72–80, 2019.

[23] Kalaivani A. and Thenmozhi D. Sarcasm identification and detection in conversion context using BERT. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 72–76, 2020.

[24] Xiangjue Dong, Changmao Li, and Jinho D. Choi. Transformer-based context-aware sarcasm detection in conversation threads from social media. *CoRR*, abs/2005.11424, 2020.

[25] Bin He, Di Zhou, Jinghui Xiao, Qun Liu, Nicholas Jing Yuan, Tong Xu, et al. Integrating graph contextualized knowledge into pre-trained language models. *arXiv preprint arXiv:1912.00147*, 2019.

[26] Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[27] Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. Cosmic: Commonsense knowledge for emotion identification in conversations. *arXiv preprint*

arXiv:2010.02795, 2020.

[28] Elena Filatova. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 392–398, 2012.

[29] D Davidov, O Tsur, and A Rappoport. Semi-supervised recognition of sarcastic sentences in twitter and amazon (pp. 15–16). *Retrieved from Association for Computational Linguistics*, 2010.

[30] Roberto I. González-Ibáñez, 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*, pages 581–586, 2011.

[31] David Bamman and Noah A. Smith. Contextualized sarcasm detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, pages 574–577, 2015.

[32] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A large self-annotated corpus for sarcasm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, 2018.

[33] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[34] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.