The Significance of Word Order in Sarcasm Detection

Nicholas Benavides
Department of Computer Science
nbenav@stanford.edu

Raymond Thai
Department of Computer Science
raythai@stanford.edu

Crystal Zheng
Department of Computer Science
crystalz@cs.stanford.edu

Abstract

We explored the importance of word order on both the model and embedding level for sarcasm detection. Using the SARC dataset, we found that utilizing ELMo embeddings with a bidirectional LSTM model had the best results and the only statistically significant improvement from baseline, indicating that it is important to take into consideration word order on both model and embedding levels. Furthermore, we found that as training set size increases, there are significant benefits to using contextualized word embeddings on a contextualized model. Through error analysis, we've found evidence that indicates that comment-level word order, phrase-level word order, and word meaning is important for sarcasm detection.

1 Introduction

Sarcasm detection is a challenging problem due to the highly contextual nature of sarcastic remarks. It's also a difficult problem for humans, since it is very difficult to identify whether a statement is sarcastic without context about the author and the circumstances in which the statement was delivered. The rise of social media, forums, blogs, and similar channels have led to an increase in informal and sarcastic text across the internet. The ability to detect sarcasm more accurately is critical to understanding and enabling sentiment analysis for text from these sources. Due to the ambiguous nature of the factors involved in sarcasm, many previous studies focused on looking into leveraging background information and novel architectures to better detect the various contextual dependencies, latent meaning, and dynamic variants of sarcastic language.

We hypothesize that word order and context is an important factor in sarcasm detection. Intuitively, word order could be important for several reasons. At a longer sentence or comment level, it's possible that sarcasm appears later in the writing because sarcasm is often intended as a comedic "punchline." For example, several sarcastic Reddit comments begin with phrases that could potentially be non-sarcastic before ending with an obviously sarcastic last phrase. For instance, the comment *"well, the silver lining is that whenever there's a terrorist attack on dc next, the jihadists will have a more prominent religious target, so federal workers and metro riders can rest a little more easily*" has sarcasm that's primarily concentrated in the last clause. We also hypothesize that word order is crucial in sarcasm detection because of phrase level ordering. Small changes in the ordering of a few words can drastically impact the perceived sarcastic-ness of the sentence. For example, the sarcastic Reddit comment "*that's totally not going to look like a beard in 15 years"* sounds much less sarcastic when rephrased as "*that's not going to totally look like a beard in 15 years."*

Understanding the precise meaning of words and understanding word meanings in relation to their surroundings can also be crucial for sarcasm detection. The complicated nature of sarcasm detection requires natural language understanding models to be fully optimized in all aspects. Furthermore, certain words may be much more sarcastic in some of their usages compared to others. For example, when the word "buy" is used as a synonym for "believe", it is often used to denote sarcastic skepticism in phrases such as

"yeah, his past really makes me buy that story". Therefore, it's probable that "buy" is much less likely to be sarcastic when used as a synonym for "purchase" compared to when it used as a synonym for "believe". These intuitive hypotheses motivate our idea to explore the significance of word order and context for sarcasm detection.

We also aim to better understand the situations where word order and context are most important. We hypothesize that with increasing training set sizes, the choice of embedding may play less of a role in improving model performance. We have this initial belief because we suspect the importance of word embeddings can be partially offset when additional training data allows models to better understand word nuances.

2 Related Work

Previous sarcasm detection studies tend to trend similarly in their focus and in their tasks. Kolchinski and Potts (2018), Hazarika et al. (2018), and Ilic et al. (2018) all achieve state-of-the-art or near state-of-the-art results in classifying statements as sarcastic or nonsarcastic. However, their approaches do have some key differences that are worth noting. Kolchinski and Hazarika et al.'s papers use very different architectures and features to address the same task. First, Hazarika et al.'s CASCADE model is built on a CNN, while both of Kolchinski's models extend a bidirectional RNN with GRU units. Kolchinski's bidirectional RNN partially incorporates word order because it evaluates words as a function of previous words. Both Hazarika et al. and Kolchinski are focused on examining the role of background information in sarcasm detection. Hazarika et al. uses a wealth of contextual information, namely user embeddings, stylometric features, personality features, and forum embeddings, to learn more complicated relationships about authors' uses of sarcasm in various situations. In contrast, Kolchinski takes a much simpler approach, using either a Bayesian prior to model an author's propensity for sarcasm or an embedding vector to encode contextual information about the author. Despite not using forum embeddings or stylometric features, Kolchinski's model performs comparably but slightly worse than Hazarika's model, suggesting that Hazarika's additional embeddings yield only incremental performance gains.

In contrast, Ilic et al.'s model differs significantly from both Kolchinski and Hazarika et al.'s models. In her paper, she does not attempt to encode any information about the author or forum. Instead, she takes a character-based approach that utilizes character-level vector embeddings for words through ELMo, which differentiates it from previous work in the field. However, Ilic et al.'s architecture does have some commonalities with Hazarika et al. and Kolchinski. Like Hazarika et al., Ilic et al. relies on CNNs to generate the contextualized embeddings, although her model combines the character-level representations into a word-level representation. Similar to Kolchinski's model, Ilic et al.'s model passes the embeddings to a bidirectional RNN, although she uses LSTM units instead of GRU units. Comparing the performances of these three models on the SARC dataset, it appears inconclusive whether Ilic et al.'s character-based approach is superior to the author-focused approaches of Hazarika et al., Kolchinski et al., and Ilic et al.'s model obtained an accuracy of 0.773, compared to an accuracy of 0.77 for Hazarika's CASCADE model and an accuracy of 0.753 for Kolchinski's system. While the architecture of Hazarika et al. and Ilic et al.'s models differ greatly, a difference in accuracy of 0.003 makes it difficult to argue that one approach is strictly better than the other.

Recently, a new form of contextual word embeddings called Embeddings from Language Models, commonly known as ELMo, was developed (Peters et al., 2018). More specifically, ELMo is a deep, contextualized way to represent words that models both the complex aspects of a word's usage and how a word's uses vary in different linguistic contexts. The ELMo word vectors are learned functions derived from a deep bidirectional long short-term memory (BiLSTM) language model trained on a large corpus that combines the BiLSTM's internal states with the top layer to capture relevant contextual information. Unlike other embedding methods, each token's representation is determined by the whole input sentence, which allows ELMo embeddings to capture more information about the context the token was used in. In addition, the model makes use of sub-word information through the use of character convolutions. ELMo embeddings are not only unique, but they also have been found to improve performance on a variety of natural language understanding tasks, including question answering, textual entailment,

semantic role labeling, coreference resolution, named entity extraction, and sentiment analysis.

3 Data

The dataset we used for evaluation is the Self-Annotated Reddit Corpus (SARC), a corpus of 1.3 million sarcastic statements developed by Khodak et al. (2018). The SARC dataset is large, diverse, and has been previously used for sarcasm detection research. The dataset has been used in numerous studies on sarcasm detection, including Kolchinski and Potts (2018), Hazarika et al. (2018), and Ilic et al. (2018) studies. They constructed the corpus from Reddit comments with author-labeled sarcasm tags as well as user, topic, and contextual information. However, since the dataset only considers comments with author-labeled sarcasm tags to be sarcastic, it contains numerous sarcastic comments that are labeled as non-sarcastic. Examining the dataset, it appears that these comments tend to be comments that are so obviously sarcastic that the author did not think a sarcasm tag was necessary. These incorrectly labeled comments could negatively impact our training and testing accuracy, especially when examining small subsets that could contain many falsely labeled examples.

In this study, we used the r/main balanced dataset containing 257,082 comments from a variety of subreddits (forums), 50% of which are sarcastic. Each example in the balanced dataset contains two comments, one that is sarcastic and one that is non-sarcastic. Thus, the goal of the model is to identify which of the two comments is the sarcastic one. We chose to focus on the balanced datasets due to the time constraints of the project, the reduced bias due to evaluating models on a balanced dataset, and to challenge the models to learn distinctions between sarcasm and non-sarcasm that they may not learn on an unbalanced dataset heavily biased towards non-sarcastic comments.

To transform the raw text comments into a format that was understandable for our models, we encoded each comment using two different word embeddings, GloVe and ELMo. GloVe word embeddings are constructed from a co-occurrence matrix. This means a word gets modeled identically every time it's used regardless of context. For example, GloVe uses the same value for the word "buy" every time it appears. Thus, GloVe usually works well when used on individual words or phrases where context in a sentence isn't easily available. As discussed in "Related Work", ELMo embeddings are generated from the hidden layers of an LSTM and each word has a unique embedding for every usage. In doing so, ELMo is able to use whole sentences as context, easily handle unseen words, and handle polysemy and nuance well, which is good for downstream tasks.

4 Models

For this project and task, we evaluated three different models - Logistic Regression, a shallow neural network called TorchShallowNeuralClassifier (TSNC), and a BiLSTM. The group of models vary in their complexity as well as how they treat word order, which helped us to identify where having contextual information, either at the embedding level or model level, was most important.

We chose logistic regression as our baseline because it is a simple model that does not take word order into account; each element of the embedding is treated as a different feature, and the order of the features is irrelevant to the model. The TSNC, like the logistic regression model, does not take the order of words into account, although it is a more complex model. This model is a shallow neural network, which gives it the ability to learn more complicated relationships between features than logistic regression can. Like logistic regression, each element of the word embedding is treated as a different feature, and the order of the features is not taken into consideration. The figure below shows an example of a shallow neural network with one hidden layer and 4 units in the hidden layer. The TSNC resembles this architecture but contains more hidden layers and more units, or nodes, per hidden layer.

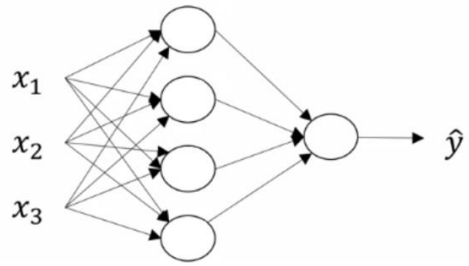

Figure 1. Shallow Neural Network Architecture

The BiLSTM model is the most complex and does take word order into account in its architecture. In a traditional RNN, inputs are passed into the model sequentially, giving it a sense of word order. Making the RNN bidirectional also gives each node access to the

previous node and vice versa, so that both preceding and successive words can be taken into account as the model is trained and makes predictions. Adding the LSTM units enables the model to store values from previous states that are further away from the current state, giving the model more flexibility in learning and store important information to apply deeper in the model. The figure below illustrates the general architecture of a BiLSTM where the word embedding has dimension 4. For reference, the BiLSTM we trained with ELMo embeddings had an input size of 2048, and the BiLSTM we trained with GloVe embeddings had an input size of 600.

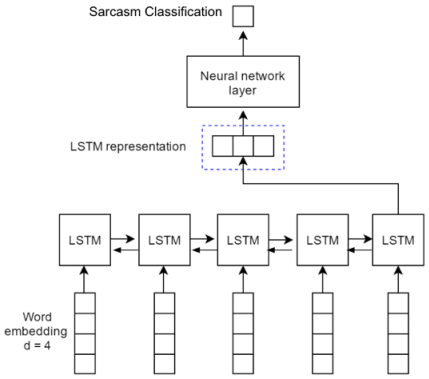

Figure 2. Bidirectional LSTM Architecture

For each model, a single training example consisted of the concatenation of embeddings for a sarcastic comment and a non-sarcastic comment. Our GloVe embeddings are 300-dimensional representations, so a model that uses GloVe embeddings would take a 600-dimensional vector as input. Our ELMo embeddings are 1024-dimensional vectors, so a model that uses ELMo embeddings would take a 2048-dimensional vector as input. All of the models produce a binary output, where 0 indicates that the first comment in the pair is not sarcastic (and thus the second comment is sarcastic since the dataset is balanced) and 1 indicates that the first comment in the pair is sarcastic (and thus the second comment is not sarcastic since the dataset is balanced).

5 Methods

For the baseline model, we fit a logistic regression, which did not require any hyperparameter tuning. On the other hand, both the TSNC and the BiLSTM required tuning hyperparameters to optimize their performance. Given our time and resource constraints for the project, we conducted a randomized hyperparameter search for each model over spaces of 72 possible combinations for the TSNC and 108 possible combinations for the BiLSTM. We trained 20 models for the TSNC and 25 models for the BiLSTM, representing 28% and 23% of the sample spaces, respectively. We decided to fix the batch size to 32 for the bidirectional LSTM to reduce the size of our hyperparameter space.

For our evaluation metric, we chose to use the macro-averaged F1 score, which combines precision and recall into a single metric and gives each output class equal weight, regardless of their relative sizes. This metric is widely used in the sarcasm detection literature, which allows to make comparisons between our findings and related work.

In order to determine whether the differences in performance between models was significant, we utilized several approaches. First, we ran the best model for the TSNC and the BiLSTM for each embedding type 10 times on random train-test splits, which we used to generate 95% confidence intervals of the macro-averaged F1 score. We also ran McNemar's test comparing the two embedding types for the TSNC and the BiLSTM models, which used the predictions of two models to generate a table similar to a confusion matrix and produces a p-value to assess statistical significance.

We also experimented with various amounts of training data for the TSNC and BiLSTM models to understand how the relative performance between embeddings changed. We chose four different training set sizes – 10,000 examples, 25,000 examples, 50,000 examples, and the full dataset (122,114 examples). For each training set size and word embedding, we trained a TSNC and a BiLSTM using the hyperparameter search procedure described above as well as computing 95% confidence intervals.

6 Results and Discussion

In Table 1 below, you can see the results of the experiments we ran. For each model type, we trained two models on the full training set – one with ELMo embeddings and one with GloVe embeddings.

	ELMo	GloVe
Logistic Regression	66.4%	67.3%

TSNC	66.6% 95% confidence interval: (65.9%, 67.3%)	66.4% 95% confidence interval: (66.0%, 66.9%)
BiLSTM	68.9% 95% confidence interval: (68.3%, 69.4%)	66.7% 95% confidence interval: (66.2%, 67.2%)

Table 1: Macro-Averaged F1 Scores.

The BiLSTM model with ELMo embeddings performed the best out of all the models with a macro-average F1 score of 68.9% (Table 1). Examining the confidence intervals, we can see that the intervals for the TSNCs overlap, indicating that the difference in F1 scores is not statistically significant. Similarly, we see that the intervals between both models that use GloVe embeddings also overlap, also indicating that the difference is not statistically significant. However, the confidence interval for the BiLSTM model with ELMo embeddings does not overlap with the confidence intervals of any of the other models, suggesting that the result was a statistically significant improvement over the other models. The results of McNemar's test further support these results. Comparing the two TSNC models, McNemar's test produced a p-value of 0.18, suggesting that the difference was not statistically significant. However, when comparing the two BiLSTM models, McNemar's test produced a p-value of 0.029, indicating that the ELMo model was significantly better than the GloVe model. Thus, it appears that contextual information is needed at both the embedding and model level to produce significantly better results. Only having contextual information at one level, corresponding to the TSNC model with ElMo embeddings and the BiLSTM model with GloVe embeddings, did not yield a meaningful improvement.

We also examined how the size of the training set impacted the relative performance of models using the different embeddings. For each of our neural network model types, embeddings, and training sizes, we trained a sarcasm detection classifier. The results of these models are shown below in Table 2.

Model	**Embed -dings**	**Train Set Sizes**	**F1 Score (%)**	**95% Confidence Intervals (%,%)**
TSNC	ELMo	10k	63.6	(63.1, 64.2)
		25k	66.6	(65.8, 67.4)
		50k	66.7	(66.3, 67.1)
		Full Data	**66.6**	(65.9, 67.3)
TSNC	GloVe	10k	63.6	(62.9, 64.4)
		25k	63.8	(63.3, 64.3)
		50k	63.9	(63.1, 64.8)
		Full Data	66.4	(66.0, 66.9)

Bi-LSTM	ELMo	10k	64.3	(63.9, 64.7)
		25k	65.0	(64.3, 65.9)
		50k	67.1	(66.6, 67.6)
		Full Data	**68.9**	(68.3, 69.4)
Bi-LSTM	GloVe	10k	63.3	(62.6, 64.0)
		25k	63.5	(62.7, 64.2)
		50k	65.0	(64.4, 65.6)
		Full Data	66.7	(66.2, 67.2)

Table 2. Model Performance on Varying Training Set Sizes

From Table 2, we see that for the TSNC models using ELMo embeddings, the model trained on 10,000 examples is significantly worse than the models trained on the larger training sizes, but the other models are not statistically different from each other. For the TSNC model using GLoVe embeddings, the model trained on the full dataset is significantly better than the models with smaller training sizes, but the differences between the three other models is not statistically significant. For the BiLSTM with ElMo embeddings, the differences between the models trained on 10,000 and 25,000 examples are not statistically significant, but the differences between the models trained on 50,000 examples and the full dataset with any of the BiLSTM ELMo models is significant. We observe the same trend with the GloVe LSTM models.

Comparing the BiLSTM models with different embeddings, we see that ELMo outperforms GloVe by a statistically-significant amount at the 25k, 50k, and full dataset levels. Thus, for larger training sets, there are significant benefits to using contextualized word embeddings on a contextualized model. For models that do not incorporate contextual elements, like the TSNC, it seems that this pattern does not hold, as the models trained on the full dataset do not differ by a significant amount between word embedding methods.

6.1 Error Analysis

From generating the confusion matrices for each model, we gained a better understanding of how the models were classifying the data compared to the ground truth. We found that BiLSTM ELMo and shallow GloVe showed the best results, with the highest true positives and true negatives, and lowest False Positives and False Negatives (Fig. 3). The BiLSTM model with GloVe model skews more towards false positives, while the TSNC model with ELMo skews more towards false negatives.

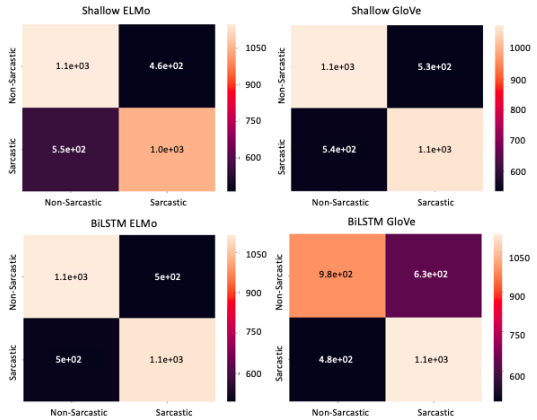

Figure 3. Confusion Matrices for Neural Network Models (x-axis: predicted label, y-axis: true label)

To assess our hypothesis that order is important for sarcasm detection in longer comments, we analyzed the word count of 105 randomly sarcastic/non-sarcastic comment pairings correctly classified by the TSNC with one embedding method but not the other. We chose the TSNC because the TSNC paired with GloVe is the only model-embedding combination with absolutely no knowledge of word order. Table 3 shows the average word count of the 210 comments. We observe that the average length of sarcastic comments that the ELMo embeddings correctly classified, 9.66, was noticeably higher than the average length of comments that GloVe correctly classified, 9.33. The small sample size of our analysis makes these results inconclusive. However, the longer word count for sarcastic comments correctly classified by ELMo aligns with our initial hypothesis.

Embedding Correctly Classified by	Method	Average Word Count - Sarcastic Comments
ELMo		9.66

GloVe	9.33

Table 3. Average word count of 210 randomly selected comments correctly classified by one of the embedding methods and misclassified by the other

As we initially suspected, the last phrases of many longer comments had the most sarcastic language and phrasing. For example, one sarcastic comment that ELMo correctly classified was "*now if only we could get some kind of live video streaming platform where they could broadcast this kind of tournament in order to grow the community and their profits at barely any cost to them... oh, that's right, there's youtube live!"* In this comment, the placement of the phrase "*oh that's right, there's youtube live!*" at the very end of this 42-word comment dramatically increases the sarcastic tone of the comment.

If the comment was analyzed as an unordered bag-of-words, it could potentially be interpreted in a less sarcastic tone. A less sarcastic rephrasing of this comment switches the ordering of the clauses to form the comment "*Oh, that's right, there's youtube live! Now if only we could get some kind of live video streaming platform where they could broadcast this kind of tournament in order to grow the community and their profits at barely any cost to them...*" This rephrasing still sounds somewhat sarcastic. However, it could potentially be a non-sarcastic response where the author is responding to another commenter that pointed out the utility of youtube live and where the author is lamenting the lack of a different live video streaming platform. On the other hand, the original comment is almost undeniably sarcastic regardless of context.

Similarly, we attempted to analyze our hypothesis that phrase-level ordering also played a large role in sarcasm detection and achieved inconclusive results. First, we examined comments that contained several words we suspected would be very indicative of sarcasm such as "totally", "definitely", and "obviously". Analyzing trends for these words was difficult because the words appeared in very few comments. Within those few comments, we did not see clear patterns of misclassification by either embedding method with either model.

However, we did notice several comments that GloVe misclassified but ELMo did not that may have been due to the positioning of these very sarcastic words. For example, the comment discussed in our introduction, "*that's totally not going to look like a beard in 15 years"* was misclassified by GloVe on the LSTM but not by ELMo on the LSTM. The ordering for this comment could have been crucial so that it was not interpreted as a non-sarcastic comment like "*that's not going to totally look like a beard in 15 years."* Similarly, ELMo with the TSNC correctly classified the sarcastic comment "clearly, chavez is also racist" while GloVe with the TSNC failed to classify this comment. The TSNC with GloVe could have interpreted this comment as the less sarcastic phrase "chavez is also clearly racist."

We tested the impact of both phrase-level ordering and word meaning on a larger sample set size by examining word frequencies of all sarcastic/non-sarcastic comments pairings that were correctly classified with ELMo embeddings but not with GloVe embeddings. The word frequency counts of correct predicted ELMo/incorrectly predicted GloVe comments were very similar using both the TSNC and LSTM. Similarly, the word frequency counts in sarcastic comments closely resembled word frequency counts in nonsarcastic comments, suggesting that GloVe struggled with these words in either context. For reference, the words that appeared most frequently in the sarcastic comments correctly classified by ELMo but not GloVe with the BiLSTM are shown below. Common words like "*the*" and "*or"* were excluded from the frequency table.

Top 10 Most Frequent Words- Misclassified by GloVe			
1.	Good	6.	Time
2.	Well	7.	More
3.	One	8.	Out
4.	Know	9.	Want
5.	People	10.	Sure

Table 4: Top 10 most frequent words in sarcastic comments misclassified by the BiLSTM with GloVe embeddings but not by the BiLSTM with ELMo embeddings

The words that appeared most frequently in that list support our hypothesis that word order and meaning is important in sarcasm detection. "Good" appeared was the most common word in sarcastic comments that GloVe could not correctly classify as sarcastic. Looking at individual sarcastic comments demonstrates why word order is so crucial for the word "good".

The misclassified sarcastic comments that contain "good" often begin with "good". When good began comments, it typically sarcastically reflected the user's views. This meaning of good usually denoted that something was appropriate for a particular purpose. For example, two sarcastic comments that GloVe failed to classify were "good thing he's a scientist" and "good to see the money is being used to help the poor, hungry and sick."

Conversely, when good was used in the middle of sentences, it was often used as an adjective to denote something that has desirable qualities. For example, two nonsarcastic comments that GloVe failed to classify were "that, my good friend, is classified information" and "bench him, he's not good enough." In the comments where "good" appeared in misclassified sarcastic comments, it was the first word in the comments 11 out of 20 occurrences. In the 21 comments where "good" appeared in misclassified non-sarcastic comments, it was the first word only 5 out of 21 occurrences.

A similar phenomenon occurred with other words that GloVe frequently missed. The second most missed word by the GloVe embeddings was "well." When used at the beginning of sentences, the word "well" is typically used as an exclamation to convey surprise or anger. This meaning and location of well often conveys a sarcastic tone. Two sarcastic comments that GloVe failed to classify were "well, they've *promised* now so it's all good" and "well, I for one am shocked."

When used in the middle of comments, the word "well" is usually used to denote praise or a fortunate outcome. For example, a non-sarcastic comment that GloVe failed to classify were "what a well-trained dog". In the comments where "well" appeared in misclassified sarcastic comments, it was the first word in the comment 15 out of 18 occurrences, while it was the first word only 12 out of 19 occurrences in misclassified non-sarcastic comments.

7 Conclusion

Through our investigation, we determined that incorporating contextual information at the embedding and author levels is critical to strong performance for sarcasm detection, evidenced by the BiLSTM model with ELMo embeddings significantly outperforming all other models. Our initial hypothesis that the significance of embedding choice decreased as training set size increased was disproven. After the training set grew larger than 10k examples, ELMo outperformed GloVe in the BiLSTM model on the remaining training set sizes, suggesting that the contextual word embeddings make more of a difference to model performance as the size of the training dataset increases. For the TSNC models, this relationship did not hold, indicating that using a contextualized word embedding is far less important to model performance when the model itself does not take word order into account.

Through error analysis, we've found evidence that supports our all of our initial hypotheses that comment-level word order, phrase-level word order, and word meaning is important for sarcasm detection. The importance of word comment-level order is supported by our observation that sarcastic comments correctly classified by the TSNC with ELMo tend to be slightly longer than comments correctly classified by the TSNC with GloVe. The importance of phrase-level order and word meaning was illustrated by examining the words that most frequently occurred in sarcastic comments correctly classified by ELMo embeddings but not GloVe embeddings. Investigating these word frequencies revealed how frequently misclassified words like "good" and "well" have particular usages and orderings that are very likely to be sarcastic. Our error analysis on heavily sarcastic words like "totally" and "definitely" was inconclusive. In the small sample size of comments with these words, we found no overarching trends. However, we did observe several cases where GloVe potentially misclassified comments because the comment had a non-sarcastic reordering.

To further determine the significance of word order, additional research with larger sample sizes regarding the significance of heavily sarcastic words, frequently misclassified words, and comment length should be conducted. In the broader context of sarcasm detection, additional research at the intersection of background information and word order should be conducted. Current state-of-the-art sarcasm detection models heavily utilize contextual information like author information but often neglect word order. Our findings on word order and context suggest that combining word ordering methods with background information could further improve these models.

8 Acknowledgements

We would like to thank our advisor, Moritz Sudhof, and Professors Chris Potts for their advice and guidance throughout this process. We would also like to thank Lucy Li and Atticus Geiger for their assistance in bringing this paper together. We referenced and adapted code from Kolchinski's Github repository for obtaining and processing the dataset, Chris Potts' CS224U Github repository for the TSNC code and McNemar's test, and Yunjey Choi's Github repository for the BiLSTM models.

9 Contributions

All coauthors contributed to the design and implementation of this study, analysis of the results, and writing of this paper. Nicholas Benavides was primarily responsible for coding the models and running experiments. Ray Thai was primarily responsible for error analysis, interpretation of results, and writing the script for the video presentation. Crystal Zheng was primarily responsible for researching the differences between embedding methods, error analysis, and literature review. All parties were equally responsible for design of the study, filming the video presentation, and writing the final report.

10 References

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. *CASCADE: Contextual sarcasm detection in online discussion forums. Proceedings of the 27th International Conference on Computational Linguistics.* ArXiv:1805.06413. https://arxiv.org/pdf/1805.06413.pdf

Suzana Ilic, Edison Marrese-Taylor, Jorge A. Balazs, and Yutaka Matsuo. 2018. *Deep contextualized word representations for detecting sarcasm and irony.* Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. ArXiv:1809.09795. https://arxiv.org/pdf/1809.09795.pdf

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. *A Large Self-Annotated Corpus for Sarcasm.* Proceedings of the Eleventh International Conference on Language Resources and Evaluation. ArXiv:1704.05579. https://arxiv.org/pdf/1704.05579.pdf

Y. Alex Kolchinski and Christopher Potts. 2018. *Representing Social Media Users for Sarcasm Detection*. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. ArXiv:1808.08470. https://arxiv.org/pdf/1808.08470.pdf

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. *Deep contextualized word representations.* Proceedings of NAACL-HLT 2018. ArXiv:1802.05365. https://arxiv.org/pdf/1802.05365.pdf