Metaphor Detection Using Contextual Word Embeddings From Transformers

Jerry Liu, Nathan O'Hara, Alexander Rubin, Rachel Draelos, Cynthia Rudin
Department of Computer Science, Duke University
{jwl50, nmo4, acr43, rlb61}@duke.edu, cynthia@cs.duke.edu

Abstract

The detection of metaphors can provide valuable information about a given text and is crucial to sentiment analysis and machine translation. In this paper, we outline the techniques for word-level metaphor detection used in our submission to the Second Shared Task on Metaphor Detection. We propose using both BERT and XLNet language models to create contextualized embeddings and a bi-directional LSTM to identify whether a given word is a metaphor. Our best model achieved F1-scores of 68.0% on VUA AllPOS, 73.0% on VUA Verbs, 66.9% on TOEFL AllPOS, and 69.7% on TOEFL Verbs, placing 7th, 6th, 5th, and 5th respectively. In addition, we outline another potential approach with a KNN-LSTM ensemble model that we did not have enough time to implement given the deadline for the competition. We show that a KNN classifier provides a similar F1-score on a validation set as the LSTM and yields different information on metaphors.

1 Introduction

A metaphor is a form of figurative language that creates a link between two different concepts and conveys rich linguistic information (Lakofi and Johnson, 1980). The complex information that accompanies a metaphorical text is often overlooked in sentiment analysis, machine translation, and information extraction. Therefore, the detection of metaphors is an important task in order to achieve the full potential of many applications in natural language processing (Tsvetkov et al., 2014).

The differences between a metaphorical text and a non-metaphorical text can be subtle and require specific domain information. For instance, in the phrase

the trajectory of your legal career

the word *trajectory* is used metaphorically. To identify this metaphor, both the meaning of the word in the context of the sentence and its literal definition must be recognized and compared. In this case, the word *trajectory* is used to describe the path of a legal career in the sentence, whereas its basic definition involves the path of a projectile. As a result of the ambiguity present in determining the basic meaning of a word, as well as whether it deviates significantly from a contextual use, detecting metaphors at a word-level can be challenging even for humans. Additionally, the Metaphor Identification Procedure used to label the datasets (MIPVU) accounts for multiple kinds of metaphors (Steen et al., 2010). Capturing implicit, complex metaphors may require different information than capturing direct, simple metaphors.

This paper describes the techniques that we utilized in the Second Shared Task on Metaphor Detection. The competition provided two datasets: a subset of ETS Corpus of Non-Native Written English, which contains essays written by test-takers for the TOEFL test and was annotated for argumentation relevant metaphors, and the VU Amsterdam Metaphor Corpus (VUA) dataset, which consists of text fragments sampled across four genres from the British National Corpus (BNC) – Academic, News, Conversation, and Fiction. For each dataset, participants could compete in two tracks: identifying metaphors of all parts of speech (AllPOS) or verbs only (Verbs).

Our final submission uses pretrained BERT (Devlin et al., 2018) and XLNet (Yang et al., 2019) transformer models, part-of-speech (POS) labels, and a two-layer bi-directional long short-term memory (Bi-LSTM) neural network architecture. BERT and XLNet are used to generate contextualized word embeddings, which are then combined with POS tags and fed through the Bi-LSTM to predict metaphoricity for each word. By creating contex-

Proceedings of the Second Workshop on Figurative Language Processing, pages 250–255

https://doi.org/10.18653/v1/P17

tualized word embeddings using transformers, we hoped to capture more long-range interdependencies between words than would be possible using methods such as word2vec, GloVe, or fastText. Indeed, our model achieved F1-scores of 68.0% on VUA AllPOS and 73.0% on VUA Verbs, improving upon results from the First Shared Task (Leong et al., 2018). On the TOEFL task, we achieved F1-scores of 66.9% on AllPOS, and 69.7% on Verbs. Our scores placed 7th, 6th, 5th, and 5th respectively in the Second Shared Task on Metaphor Detection (Leong et al., 2020).

2 Related Works

Historically, approaches to automatic metaphor detection have focused on hand-crafting a set of informative features for every word and applying a supervised machine learning algorithm to classify words as metaphorical or non-metaphorical. Previous works have explored features including POS tags, concreteness, imageability, semantic distributions, and semantic classes as characterized through SUMO ontology, WordNet, and VerbNet (Beigman Klebanov et al., 2014; Tsvetkov et al., 2014; Dunn, 2013; Mohler et al., 2013).

Deep learning methods have also been employed for automatic metaphor detection. In the First Shared Task on Metaphor Detection, the top three highest scoring teams all employed an LSTM model with word embeddings and additional features (Leong et al., 2018). Stemle and Onysko (2018) trained fastText word embeddings on various native and non-native English corpora, and passed the sequences of embeddings to an Bi-LSTM. The highest-performing model from Bizzoni and Ghanimifard (2018) employed a Bi-LSTM on GloVe embeddings and concreteness ratings for each word. Wu et al. (2018) appended POS and semantic class information to pretrained word2vec word embeddings, and utilized a CNN in addition to a Bi-LSTM in order to better capture local and global contextual information. In all these cases, the word embeddings used are context-independent: the same word appearing in two different sentences will nonetheless have the same embedding. Thus, these embeddings may not be able to fully capture information about multi-sense words (for example, the word *bank* in *river bank* and *bank robber*), which is crucial for properly identifying metaphors.

More recently, Mao et al. (2019) proposed two RNN models for word-level metaphor detection based on linguistic theories of metaphor identification. GloVe and ELMo embeddings are used as input features that capture literal meanings of words, which are compared with the hidden states of Bi-LSTMs that capture contextual meaning. We chose to explore transformer-based embeddings as an alternative way to capture contextual information.

Transformer-based models have shown state-of-the-art results on a wide variety of language tasks, including sentence classification, question answering, and named entity recognition. These models rely on self-attention mechanisms to capture global dependencies, and can be used to generate contextualized word embeddings. We chose to examine the models BERT, GPT2, and XLNet. These three models all achieve remarkable performances on various NLP tasks, but they capture long-distance relationships within the text in different ways. BERT is an autoencoder model, consisting of a stack of encoder layers, and is able to capture bi-directional context using masking during training (Devlin et al., 2018). GPT2 is an autoregressive model, consisting of a stack of decoder layers, and thus is only able to capture unidirectional context (Radford et al., 2018). XLNet is also autoregressive, but it captures bi-directional context by considering all permutations of the given words (Yang et al., 2019). Each of these models has its advantages and disadvantages that are worth exploring in the context of metaphor detection.

3 Methodology

Our method for metaphor detection begins with generating contextualized word embeddings for each word in a sentence using the hidden states of pretrained BERT and XLNet language models. Next, those embeddings are concatenated together, POS tags for each word are appended to the embeddings, and a Bi-LSTM reads the features as input and classifies each word in the sentence.

Word Embeddings Due to limited metaphor-annotated data, rather than training a transformer model on our downstream task, we instead opted to take a feature-based approach to generating contextualized word embeddings from pretrained transformer models. This idea was inspired by the approach to the token-level named entity recognition task described in Devlin et al. (2018), which used a number of strategies for combining hidden

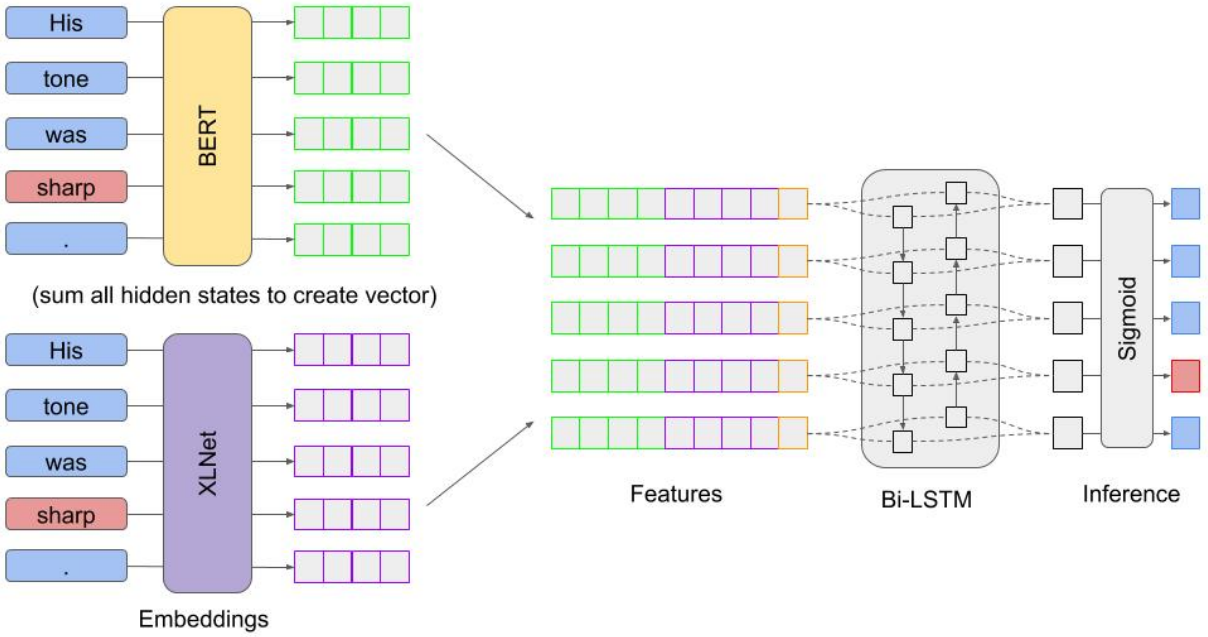

Figure 1: Our model architecture. Sentences are fed through pretrained BERT and XLNet models, concatenated along with POS tags, passed to a Bi-LSTM, and a sigmoid layer outputs probabilities.

state representations of words from a pretrained BERT model to generate contextualized word embeddings.

We installed the Python transformers library developed by huggingface (Wolf et al., 2019), which includes a PyTorch (Paszke et al., 2019) implementation of BERT and several pretrained BERT models. We opted to use the BERT base uncased model, which consists of 12-layers, 768-hidden, 12-heads, and 110M parameters. For each line in the VUA and TOEFL datasets, we use the BERT tokenizer included in the transformers package to pre-process the text, then generate hidden-state representations for each word by inputting each line into the pretrained BERT model. Each token is given a 12x768 hidden-state representation from BERT. We generate 768-dimension word embeddings by summing the values from each of the 12 hidden layers for each token. Words out-of-vocab for BERT are split into multiple tokens representing subwords. To generate embeddings for these words, embeddings are generated for each subword token, then averaged together.

Similarly, we installed the huggingface implementation of XLNet and used its pretrained XLNet base uncased model to generate embeddings for each word in the dataset using the same method as with BERT.

Once both embeddings are generated, we concatenate the BERT and XLNet embeddings for each word to generate 1536-dimensional word embeddings. By combining word embeddings from multiple high-performing pretrained transformers, we are able to capture more contextual information for each word. Additionally, we supplement these word embeddings with the POS tag for each word as generated by the Stanford parser (Toutanova et al., 2003). POS tags were shown to improve metaphor detection in the 2018 Metaphor Detection Shared Task (Leong et al., 2018), and we find a small improvement by including them here.

Neural Network We pass the features from each sentence into a Bi-LSTM. The purpose of this network is to capture long-range relationships between words in the same sentence which may reveal the presence of metaphors. We use a dense layer with a sigmoid activation function to obtain the predicted probability of being a metaphor for each word in the sentence. During training, we employ a weighted binary cross entropy loss function to address the extreme class imbalance, since non-metaphors occur significantly more frequently than metaphors. Hyperparameters were tuned via cross-validation. For the testing phase, we use an ensemble strategy which was effective for Wu et al. (2018): we trained four copies of this Bi-LSTM with different initializations and averaged the pre-

dictions from each model.

Additionally, we noted that our model tended to assign similar probabilities to different instances of the same word in different contexts, and that a prediction significantly higher than the average prediction for that word was a good indicator of the presence of metaphor, even if the prediction fell lower than the ideal threshold. Thus, we used the following procedure for the testing phase: label the word as a metaphor if its predicted probability is higher than the threshold, or if its probability is three orders of magnitude higher than the median predicted probability for that word in the evaluation set. We found this to be a useful way of addressing the domain shift between the training and the test data. This concept is further explored in Section 4.1.

4 Experiments

Word Embeddings Devlin et al. (2018) suggest that for different token-level classification tasks, different methods for combining hidden states from BERT may prove effective in generating contextualized word embeddings. For our task, to determine the optimal embedding strategy, we evaluated four different methods of combining information from hidden states of the transformer models. To determine which performed best prior to training LSTM models, we tested each strategy using logistic regression on the word embeddings with an 80/20 training-test split. Results from logistic regression on BERT embeddings from the VUA AllPOS data are in Table 1. We note that the F1 scores using different methods of generating contextualized word embeddings differ substantially. We use the "sum-all-layers" method of generating word embeddings for our further experiments.

	VUA AllPOS			TOEFL AllPOS		
Method	P	R	F1	P	R	F1
Sum all layers	**0.672**	0.531	**0.593**	0.569	**0.596**	**0.582**
Concat. last 4 layers	0.614	**0.552**	0.581	**0.644**	0.473	0.546
Sum last 4 layers	0.623	0.534	0.575	0.594	0.550	0.571
Second to last layer	0.580	0.547	0.563	0.633	0.482	0.547
Last layer	0.628	0.493	0.553	0.542	0.551	0.546

Table 1: Logistic regression on various BERT word embeddings, VUA and TOEFL AllPOS.

Transformers Table 2 compares the performance of the Bi-LSTM using the embeddings from BERT, GPT2, and XLNet. Because the true test labels were not made available to us, here we report results on an 80/20 training-test split of the given training data. We make the following observations.

- The LSTM models perform far better than their logistic regression counterparts. Of the single embedding LSTM models, the BERT and XLNet embeddings have the best performances. Combining BERT and XLNet embeddings and using an ensemble strategy further improved our performance.
- In general, the AllPOS task is more challenging than the Verbs task. Different parts of speech are used metaphorically in different ways, and these multiple varieties of metaphor must all be captured by a single model in the AllPOS task. Correspondingly, all models perform worse on AllPOS than Verbs in both VUA and TOEFL datasets.
- Additionally, the models achieve a lower F1 score on the TOEFL dataset than the VUA in both AllPOS and Verbs track. We believe this is in part due to the smaller size of the TOEFL dataset, and in part because linguistic characteristics can differ substantially between native and non-native text. Since we used transformer models pretrained on a native corpus, the word embeddings were likely less informative for the TOEFL track.
- GPT2 and XLNet are both autoregressive language models, but GPT2+LSTM performs significantly worse than the other LSTM models. This result suggests that bi-directional relationships between words play a crucial role in metaphor detection. Because XLNet considers every possible permutation of the given words during training, the XLNet embeddings likely contain more bi-directional context than the GPT2 embeddings.

4.1 A Promising Future Approach: K-Nearest Neighbors

In our experiments, we noted that our LSTM models tended to output similar probabilities for different instances of the same word independent of context. For example, although 4 out of 14 of the occurrences of the word *capacity* in the validation set were metaphor-related, all of the LSTM predictions were less than 10^{-5}. This suggested that although word embeddings from transformer models

Model	VUA AllPOS			VUA Verbs			TOEFL AllPOS			TOEFL Verbs		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Baseline*	0.608	0.700	0.651	0.600	0.763	0.672	N/A	N/A	N/A	N/A	N/A	N/A
BERT+LSTM	0.644	0.689	0.666	**0.662**	0.730	0.694	0.618	0.648	0.633	0.611	0.670	0.639
GPT2+LSTM	0.592	0.573	0.582	0.618	0.648	0.633	0.579	0.589	0.584	0.555	0.681	0.612
XLNet+LSTM	0.622	0.622	0.622	0.650	0.684	0.667	0.644	0.646	0.645	0.633	0.681	0.656
BERT+XLNet +LSTM	0.665	0.688	0.676	0.655	0.736	0.693	0.649	**0.664**	0.656	0.618	**0.724**	0.667
BERT+XLNet +LSTM (ensemble)	**0.675**	**0.710**	**0.692**	0.656	**0.768**	**0.708**	**0.686**	0.654	**0.669**	**0.722**	0.659	**0.689**

Table 2: Performance of LSTM models. The baseline is the highest achieved score from the First Shared Task on Metaphor Detection.

contain more contextual information than embeddings from word2vec or GloVe, the model could be improved by including even more contextual information. We explored the idea of ensembling an LSTM with a K-Nearest Neighbors (KNN) classification approach. We believe that the LSTM approach would give information as to which types of words tend to be metaphors in context, whereas the KNN approach would clue into whether a specific use of a specific word is more likely to be metaphorical. We were unable to fully implement such an ensemble model for the competition, but we detail some promising results below.

We trained a KNN-only model using our contextualized word embeddings. First, we lemmatized each word in the VUA and TOEFL datasets. For VUA, we classified each word based on a KNN classifier trained on all instances of the same lemmatized word in the training data. If no such lemmatized word existed in the training data, we classified that word using a prediction from an LSTM model, though that occurred in only 2% of cases. For TOEFL, we compared using training data from TOEFL combined with VUA due to the limited dataset. We achieved F1 scores of 0.642 and 0.608 on 80/20 training-test splits of VUA and TOEFL respectively, not much worse than our LSTM models.

There is reason to believe the LSTM and KNN approaches capture significantly different information on metaphors. On the VUA validation data, the LSTM method predicted 3751 metaphors and the KNN predicted 3190. However, only 2372 words were predicted as metaphors by the two models together. Since both models have similar F1 scores, this implies that a superior classifier can be constructed using information from both classifiers.

For our final submissions, we were able to adopt a simplified implementation of this approach, labeling an instance of a word as metaphorical if its LSTM prediction either was higher than a certain threshold, or higher by a significant amount than the median LSTM prediction of all instances of that word. This procedure improved our F1 scores by about 1% during the testing phase.

k	Precision	Recall	F1
1	0.665	0.599	0.630
2	**0.722**	0.514	0.600
3	0.676	**0.611**	**0.642**
4	0.703	0.538	0.610
5	0.679	0.604	0.639

Table 3: KNN using sum-all BERT word embeddings, VUA AllPOS

5 Conclusion

In this paper, we describe the best performing model that we submitted for the Second Shared Task on Metaphor Detection. We used BERT and XLNet language models to create contextualized embeddings, and fed these embeddings into a bi-directional LSTM with a sigmoid layer that used both local and global contextual information to output a probability. Our experimental results verify that contextualized embeddings outperform previous state-of-the-art word embeddings for metaphor detection. We also propose an ensemble model combining a bi-directional LSTM and a KNN, and show promising results that suggest the two models encode complementary information on metaphors.

Acknowledgments

The authors thank the organizers of the Second Shared Task on Metaphor Detection and the rest of the Duke Data Science Team. We also thank the anonymous reviewers for their insightful comments.

References

Beata Beigman Klebanov, Ben Leong, Michael Heilman, and Michael Flor. 2014. Different texts, same metaphors: Unigrams and beyond. In *Proceedings of the Second Workshop on Metaphor in NLP*, pages 11–17, Baltimore, MD. Association for Computational Linguistics.

Yuri Bizzoni and Mehdi Ghanimifard. 2018. Bigrams and BiLSTMs two neural networks for sequential metaphor detection. In *Proceedings of the Workshop on Figurative Language Processing*, pages 91–101, New Orleans, Louisiana. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, arXiv:1810.04805.

Jonathan Dunn. 2013. What metaphor identification systems can tell us about metaphor-in-language. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 1–10, Atlanta, Georgia. Association for Computational Linguistics.

George Lakofi and Mark Johnson. 1980. *Metaphors we live by*. University of Chicago Press, Chicago, IL.

Chee Wee Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 vua and toefl metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, Seattle, WA.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, and Ekaterina Shutova. 2018. A report on the 2018 VUA metaphor detection shared task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 56–66, New Orleans, Louisiana. Association for Computational Linguistics.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898, Florence, Italy. Association for Computational Linguistics.

Michael Mohler, David Bracewell, Marc Tomlinson, and David Hinote. 2013. Semantic signatures for example-based linguistic metaphor detection. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 27–35, Atlanta, Georgia. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Gerard J. Steen, Aletta G. Dorst, J. Berenike Herrmann, Anna A. Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A Method for Linguistic Metaphor Identification: From MIP to MIPVU*. John Benjamins Publishing.

Egon Stemle and Alexander Onysko. 2018. Using language learner data for metaphor detection. In *Proceedings of the Workshop on Figurative Language Processing*, pages 133–138, New Orleans, Louisiana. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 248–258, Baltimore, Maryland. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv*, arXiv:1910.03771.

Chuhan Wu, Fangzhao Wu, Yubo Chen, Sixing Wu, Zhigang Yuan, and Yongfeng Huang. 2018. Neural metaphor detecting with CNN-LSTM model. In *Proceedings of the Workshop on Figurative Language Processing*, pages 110–114, New Orleans, Louisiana. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv*, arXiv:1906.08237.