Metaphor Detection using Ensembles of Bidirectional Recurrent Neural Networks

Jennifer Brooks
The George Washington University
Washington, DC
jtbrooks@gwu.edu

Abdou Youssef
The George Washington University
Washington, DC
ayoussef@gwu.edu

Abstract

In this paper we present our results from the Second Shared Task on Metaphor Detection, hosted by the Second Workshop on Figurative Language Processing. We use an ensemble of RNN models with bidirectional LSTMs and bidirectional attention mechanisms. Some of the models were trained on all parts of speech. Each of the other models was trained on one of four categories for parts of speech: "nouns", "verbs", "adverbs/adjectives", or "other". The models were combined into voting pools and the voting pools were combined using the logical "OR" operator.

1 Introduction

Figurative language is common in everyday speech and generally easy for humans who are speaking the same language to interpret, yet machines have trouble with it, limiting our interaction with them. If a machine has trouble understanding our natural language, then it will have trouble interpreting our intentions and translating them correctly to another language or human. Therefore, the goal of our research is to improve metaphor detection to facilitate the interpretation and translation of natural language in discourse.

The Second Shared Task on Metaphor Detection used the Vrije University Amsterdam Metaphor Corpus (VUAMC) (Steen et al., 2010), which has been the most widely used database for training machines to detect metaphors. The metaphor labels in the VUAMC are per word and indicate whether the word is a metaphor related word (mrw) or not. An mrw may be an indirect, direct, or implicit metaphor. The VUAMC contains text from four sources: academic texts, newspapers, conversations, and fiction. Each word was labeled using the MIPVU procedure, with greater than 0.8 inter-annotator reliability (Steen et al., 2010). About 13% of the words in the VUAMC are labeled as metaphor related words.

The Second Shared Task on Metaphor Detection with the VUAMC demonstrated state of the art performance with the best performer achieving an F1 score of 0.769. On the same training and test sets, we were able to achieve an F1 score of 0.703, and when we randomly split the VUAMC data around sentences vs. fragments (which may contain more than one related sentence), we were able to achieve an F1 score of 0.730. Leong et al. (2020) provide a summary of the results from all participants in the shared task.

Listed below are our major findings and contributions:

- When forwarding information from a bidirectional LSTM to an attention layer, better performance can be obtained when each attention cell receives output from only one bidirectional LSTM cell (vs a fully connected architecture where the output of every bidirectional LSTM cell is forwarded to every attention cell). For reference, see the architectural differences between Figures 2 and 1.

- It is possible to get better performance from logically combining the outputs of an ensemble, compared with using only the usual ensemble approaches of combining models: boosting, bagging, or stacking.

- Splitting the training and data sets by randomly sampling sentences rather than fragments (or paragraphs), which contain more than one sentence, provides for better results.

- Concatenating ELMo (Peters et al., 2018) with GloVe (Pennington et al., 2014) word embeddings gives better results than using either one alone.

Proceedings of the Second Workshop on Figurative Language Processing, pages 244–249

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

Next we present related work followed by a discussion of our approach. Then we present a summary of our results followed by conclusions. We conclude with a brief discussion about our future research plan.

2 Related Work

Dinh et al. (2016) showed that using only word embeddings to train a neural network to detect metaphor related words in the VUAMC resulted in performance that was comparable to the performance of approaches that incorporated additional features, such as parts-of-speech. Using only word embeddings they achieved 52.4% recall with 58.3% precision on the shared task data, covering all parts of speech, from the NAACL 2018 Workshop on Figurative Language Processing.

Wu et al. (2018) used a layered model with lemmatized-word embeddings. They layered a bidirectional LSTM (bi-LSTM) on top of a CNN. For the output layer, they experimented with Conditional Random Fields (CRF) vs. softmax. Their best performance was with softmax. They used 300d word2vec embeddings and the RMSProp optimizer. They also input one-hot vectors for parts-of-speech and one-hot vectors for cluster ids from clustering the word embeddings with k-means. They demonstrated the best performance on the VUAMC shared task, over all parts of speech, with 70% recall and 60.8% precision.

Bizzoni and Ghanimifard (2018) presented two alternative architectures, a bidirectional LSTM and a novel bigram model which is a sequence of fully connected neural networks (concatenation and ReLU for each network in the sequence). They experimented with different word embeddings (300d GloVe and word2vec) and the inclusion of concreteness scores. They used a maximum sentence length of 50 words. Sentences with more than 50 words were broken up into smaller chunks. Sentences with less than 50 words were padded. The best performance between the two models was with the bidirectional LSTM using the GloVe embeddings and concreteness scores, but an ensemble of the bidirectional LSTM and novel bigram model performed even better. Over all parts of speech on the VUAMC, they achieved 68% recall with 59.5% precision.

Stemle and Onysko (2018) also split sentences into segments depending on a maximum sequence length. Shorter sequences were padded. They used a layered model with input to a bidirectional LSTM which provided input to a fully connected output layer that was activated by the softmax function. The length of the output equaled the length of the input. The output predicted whether each word from the input layer was a metaphor related word or not. They used a categorical cross-entropy loss function to address the imbalance of non-metaphor related words to metaphor related words. They experimented with word embeddings from various models that were pre-trained on corpora that varied in their language proficiency levels. On the VAUMC shared task data sets for all parts of speech, they achieved 69.8% recall with 55.3% precision.

Leong et al. (2018) used logistic regression and random forest classifiers with lemmatized unigrams, generalized WordNet semantics, and difference in concreteness ratings between verbs/adjectives and nouns (Leong et al., 2018; Beigman Klebanov et al., 2016). During training, each class was weighted by the inverse of its frequency. For the optimization function, they used the f1-score. They achieved 69.6% recall with 51% precision on the VUAMC shared task data sets for all parts of speech.

Mykowiecka et al. (2018) trained an LSTM on 300d GloVe embeddings. They also experimented with using part-of-speech information and features from the General Inquirer, which worsened their results on the test data. Swarnkar and Singh (2018) presented an architecture that used a context encoder inspired by a bidirectional LSTM. The output of the encoder was fed to a feature selection module to select features for the token word. They showed that re-weighting examples and using parts of speech, WordNet, and concreteness ratings improved the performance of their model. Skurniak et al. (2018) presented a CRF sequence model that was trained using GloVe word embeddings and contextual information. Pramanick et al. (2018) used a hybrid model of bi-LSTM and CRF trained with word2vec embeddings for the token word and its lemma, 20d vectors representing the POS, and one-hot vectors for whether the lemma and the token were the same, and whether the lemma was present in the token.

Other researchers have made progress in metaphor detection at the word level, but the results were reported for data sets other than the VUAMC. Hovy et al. (2013) used SVMs with tree kernels on syntactic features and achieved an f1-score of 75%.

They built a corpus of 3872 labeled metaphors which they also released. Su et al. (2017) presented results from using the theory of meaning to identify metaphors in a subset of the BNC, which they labeled themselves. They achieved an f1-score of 87%. (The VUAMC is also a subset of the BNC.) Krishnakumaran and Zhu (2007) used WordNet bigram counts to identify metaphors in the Master Metaphor List created by (Lakoff, 1994). They reported 58% accuracy, 70% precision, and 61% recall.

3 Method

We designed and experimented with various RNN architectures using bidirectional LSTMs and attention mechanisms (Bahdanau et al., 2014; Zhou et al., 2016). The input was an 11-gram for each word in the training set (or test set during testing). Each word was represented by an 11-gram and appeared at the center of the 11-gram. Furthermore, each word in the 11-gram was represented by a 1,324 dimensional word embedding which was the result of concatenating a 1,024 dimensional ELMo (Peters et al., 2018) embedding with a 300 dimensional GloVe (Pennington et al., 2014) embedding, because preliminary testing revealed that ELMo concatenated with GloVe resulted in better performance than either one of them alone.

The 11-grams were from one sentence; i.e., they never extend across two sentences. Padding was used if the center word was not in the context of exactly 5 words to the left or right, so the first word in a sentence would always have 5 pads to its left and the last word in a sentence would always have 5 pads to its right. Five was chosen for the window size because it produced the best results in preliminary experiments.

The output was a two-dimensional vector representing the probabilities for the center word being a metaphor or not. Softmax was used to choose the highest probability.

Two architectures were used for our final results on the Shared Task. They are described below.

The first architecture is a many-to-one bidirectional LSTM with bidirectional attention (see Figure 1). In this architecture, the outputs of the forward and backward LSTM cells in the attention layer are concatenated only at the output for the center, target word, of the 11-gram. The expected output is a 1 or 0 for the center word in the 11-gram, depending on whether it is a metaphor related word

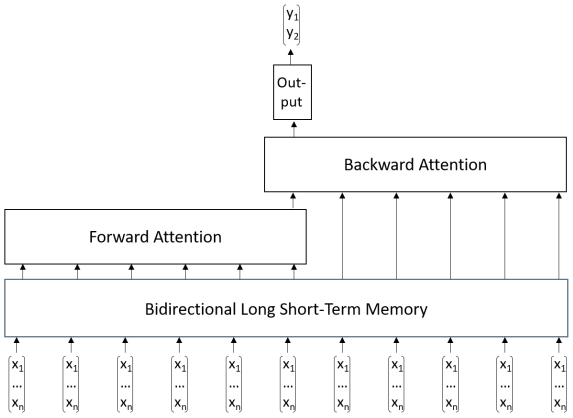

Figure 1: Many-to-One Bidirectional LSTM with Bidirectional Attention

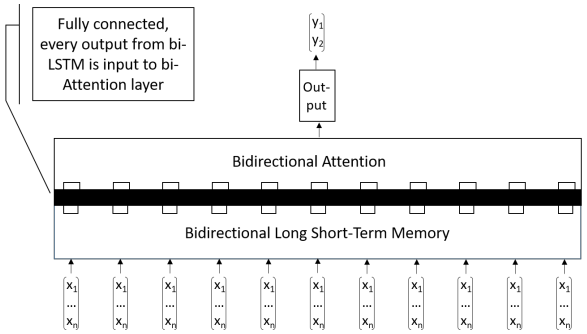

Figure 2: Many-to-One Fully-Connected Bidirectional LSTM with Bidirectional Attention

or not. Each attention cell receives output from only one bidirectional LSTM cell (vs. a fully connected architecture where the output of every bidirectional LSTM cell is forwarded to every attention cell). See the difference between Figures 2 and 1 for reference. The intuition behind the choice to forward only one cell's output per attention cell is that the attention cells are intended to process input in a sequential order (i.e., one word at a time). Providing each step with the entire matrix of weights for all words from the bidirectional LSTM seems to violate the design of the attention mechanisms. We tried both architectures and got better results with the architecture in which each attention cell receives output from only one bidirectional LSTM cell.

The second model is a many-to-many bidirectional LSTM with bidirectional attention (see Figure 3). The expected output is a 1 or 0 for each word in the 11-gram, depending on whether the word is a metaphor related word or not. However, in the trained model, only the output for the center word, w, is used to assign a prediction to w.

	Precision	Recall	F1
Many-to-Many	**0.683**	0.678	0.681
Many-to-One	0.655	**0.715**	0.684

Table 1: Performance from voting with many-to-many vs. many-to-one models

The key difference between this model and the first model is that the many-to-many model updates its weights based on the performance of the model on the target word's context words, in addition to the performance of the model on target word. The performance feedback includes whether or not the target word is in the context (within a window of 5 words on either side) of another metaphor related word. This is also why we must split across sentences and not allow a sentence from the training set to also appear in the test set. The many-to-many model was chosen because its performance complements the first model (i.e., the many-to-one bidirectional LSTM with bidirectional attention). Voting among trained instances of the many-to-many model gives better precision, while voting among trained instances of the many-to-one model gives better recall. Both have comparable F1 scores. Table 1 shows results from voting with five models of each architecture type. If at least two of the five models labeled the word as a metaphor related word, then it was scored as a metaphor related word.

Another important note about the many-to-many model is that the output of the backward attention layer starting at the last word in the 11-gram, w_{10}, is concatenated with the output of the forward attention layer for w_10; the output of the backward attention layer for w_9 is concatenated with the output of the forward attention layer for w_9; and so on until the output of the backward attention layer for w_0 is concatenated with the output of the forward attention layer for w_0.

We used a training batch size of 200. An Adam optimizer was used with a learning rate 0.006 and a decay rate of 0.001. The loss function was categorical cross entropy. A dropout rate of 0.2 was used for the bi-LSTM layer and a dropout of 0.1 was used for both the forward and backward attention. The hidden states for both the bi-LSTM and attention layers were vectors of length 128. The output layer of each model used the softmax activation function. Keras with a Tensorflow backend was used for the implementation.

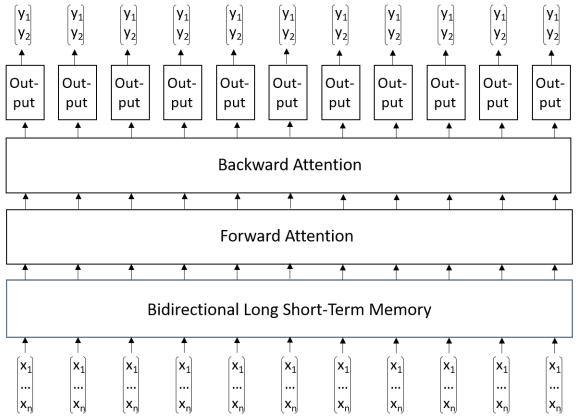

Figure 3: Many-to-Many Bidirectional LSTM with Bidirectional Attention

We trained and tested our models independently on two data splits. For the first split, 25% of the data samples (11-grams from the VUAMC) were randomly selected and held out for testing. Among the remaining 75%, one-third of the samples were randomly selected and preserved, along with all of the positive samples (labeled 1) in the remaining two-third. The rest of the training samples were discarded to achieve a more balanced training set. (However, testing was always performed on the entire test set.)

For the second split, we used the training and test sets from the Second Shared Task on Metaphor Detection. For the Shared Task, the training and test sets were sampled by fragments, in which a fragment (e.g., a paragraph) may contain more than one sentence. We initially used one-third of the training samples, the same way we did with the first split, but then we tried using the entire training set and passed class weights to the Adam optimizer to mitigate the imbalanced number of samples per class. The class weights were proportional to the percentage of samples in each class. We got better results using all of the samples in the training set. (We did not go back to the first split to train with all of the training samples and class weights, but we hypothesize that we'd get better results if we did.)

First, we trained and tested our model using 300 dimensional GloVe vectors (Pennington et al., 2014). Next, we tried 1024 dimensional ELMo vectors (Peters et al., 2018). Finally, we used 1324 dimensional vectors from combining GloVe with ELMo. In the last case, for each word, we simply concatenated the ELMo vector representation for that word with the GloVe vector representation for

	All POS	Verbs	Adv/Adj	Nouns	Other
Many-to-Many	0.681	0.726	0.643	0.665	0.647
Many-to-One	0.684	0.721	0.627	0.672	0.702
Ensemble	0.689	0.737	0.641	0.677	0.672

Table 2: F1 score per models and ensembles per part-of-speech category

that word, resulting in a vector of length 1324.

We trained each architecture independently multiple times on all parts of speech and then on each of four categories for parts of speech: "nouns", "verbs", "adverbs/adjectives", or "other". We used the NLTK toolkit (Bird et al., 2009) to derive the parts of speech.

4 Results

We achieved the best results with an ensemble of trained models. The ensemble consisted of five models per architecture trained on all parts of speech, and five models per architecture trained independently on each of four parts-of-speech categories: "nouns", "verbs", "adverbs/adjectives", or "other". The five models per architecture per part-of-speech category (including "all parts of speech") were assembled into a voting pool, so there were ten models total per category. Each set of ten models were combined into a voting pool and the voting pools were combined using the logical "OR" operator.

Table 2 shows the F1 score per part-of-speech category that resulted from voting on whether or not the target word was a metaphor related word (mrw). For the many-to-many and many-to-one models, if at least two of the five models per category labeled a word as an mrw, then it was scored as an mrw. The row for "Ensemble" shows the results from voting among the many-to-many and many-to-one models per category. The Ensemble row is meant to show the level of improvement that can be obtained by combining all ten models per category. An overall F1 score of 0.703, with 0.702 precision and 0.704 recall, was obtained by combining the "All POS" label with the appropriate part-of-speech category using the "OR" operator. For example, if the target word is a verb, then the verb was labeled as an mrw if the Ensemble for "All POS" labeled it as an mrw OR the Ensemble for "Verbs" labeled it as an mrw.

We also evaluated our many-to-one model with respect to the novelty scores provided by Dinh (Do Dinh et al., 2018) for the VUAMC. On novel metaphors (i.e. metaphors with a novelty score of at least 0.5 from (Do Dinh et al., 2018)) the many-to-one architecture found 52/77, or 67.5% in the shared task test set.

5 Conclusions and Next Steps

We have described two model architectures and an ensemble approach for metaphor detection. We have shown that splitting the training and data sets by randomly sampling sentences rather than fragments (or paragraphs), which may contain more than one sentence, may lead to better results. We believe this may be because there are patterns of language in the test fragments that were not seen in the training fragments. Allowing a model to train on some sentences from a fragment and then test on the other sentences in the same fragment may produce better results overall.

We shared that when forwarding information from a bidirectional LSTM to an attention layer, better performance can be obtained when each attention cell receives output from only one bidirectional LSTM cell.

Finally, we have revealed that it is possible to get better performance from logically combining the outputs of an ensemble.

In future work, we will continue improving metaphor detection with a focus on novel metaphors. According to Shutova et al. (2013), "Cameron (2003) conducted a corpus study of the use of metaphor in educational discourse for all parts of speech. She found that verbs account for around 50% of the data, the rest shared by nouns, adjectives, adverbs, copula constructions and multiword metaphors." About 43% of metaphors in the VUAMC are verbs, while verbs are only 23% of all tokens in the VUAMC database. However, Do Dinh et al. (2018) found that only about 24% of the novel metaphors are verbs and 41% are nouns. The rest are adjectives and adverbs. (Other POS were not included.) Therefore, future work in detecting novel metaphors may place a heavier weight on nouns (vs. verbs as has been the case with conventional metaphors).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Beata Beigman Klebanov, Chee Wee Leong, E Dario Gutierrez, Ekaterina Shutova, and Michael Flor. 2016. Semantic classifications for detection of verb metaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 101–106.

Steven Bird, Edward Loper, and Ewan Klein. 2009), publisher = O'Reilly Media Inc. *Natural Language Processing with Python*.

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.

Lynne Cameron. 2003. Metaphor in educational discourse.

do Dinh, Erik-Lân, and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 28–33.

Erik-Lân Do Dinh, Hannah Wieland, and Iryna Gurevych. 2018. Weeding out conventionalized metaphors: A corpus of novel metaphor annotations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1424, Brussels, Belgium. Association for Computational Linguistics.

Dirk Hovy, Shashank Shrivastava, Sujay Kumar Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. 2013. Identifying metaphorical word use with tree kernels. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 52–57.

Saisuresh Krishnakumaran and Xiaojin Zhu. 2007. Hunting elusive metaphors using lexical resources. In *Proceedings of the Workshop on Computational approaches to Figurative Language*, pages 13–20. Association for Computational Linguistics.

George Lakoff. 1994. *Master metaphor list*. Berkeley, CA: University of California.

Chee Wee 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.

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.

Agnieszka Mykowiecka, Aleksander Wawer, and Malgorzata Marciniak. 2018. Detecting figurative word occurrences using word embeddings. In *Proceedings of the Workshop on Figurative Language Processing*, pages 124–127.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Malay Pramanick, Ashim Gupta, and Pabitra Mitra. 2018. An lstm-crf based approach to token-level metaphor detection. In *Proceedings of the Workshop on Figurative Language Processing*, pages 67–75.

Ekaterina Shutova, Barry Devereux, and Anna Korhonen. 2013. Conceptual metaphor theory meets the data: A corpus-based human annotation study. *Language Resources and Evaluation*, 47.

Filip Skurniak, Maria Janicka, and Aleksander Wawer. 2018. Multi-module recurrent neural networks with transfer learning. a submission for the metaphor detection task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 128–132.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. 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.

Chang Su, Shuman Huang, and Yijiang Chen. 2017. Automatic detection and interpretation of nominal metaphor based on the theory of meaning. *Neurocomputing*, 219:300–311.

Krishnkant Swarnkar and Anil Kumar Singh. 2018. Dilstm contrast: a deep neural network for metaphor detection. In *Proceedings of the Workshop on Figurative Language Processing*, pages 115–120.

Chuhan Wu, Fangzhou Wu, Yubo Cen, 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 110–115.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.