Di-LSTM Contrast : A Deep Neural Network for Metaphor Detection

Krishnkant Swarnkar and **Anil Kumar Singh**
Indian Institute of Technology (BHU), Varanasi, India
{ krishnkant.swarnkar.cse15, aksingh.cse } @iitbhu.ac.in

Abstract

The contrast between the contextual and general meaning of a word serves as an important clue for detecting its metaphoricity. In this paper, we present a deep neural architecture for metaphor detection which exploits this contrast. Additionally, we also use cost-sensitive learning by re-weighting examples, and baseline features like concreteness ratings, POS and WordNet-based features. The best performing system of ours achieves an overall F1 score of 0.570 on All POS category and 0.605 on the Verbs category at the Metaphor Shared Task 2018.

1 Introduction

Lakoff (1993) defines a metaphorical expression as a linguistic expression which is the surface realization of a cross-domain mapping in a conceptual system. On one hand, metaphors play a significant role in making a language more creative. On the other, they also make language understanding difficult for artificial systems.

Metaphor Shared Task 2018 (Leong et al., 2018) aims to explore various approaches for word-level metaphor detection in sentences. The task is to predict whether the target word in the given sentence is metaphoric or not. There are two categories for this shared task. The first one, All POS, tests the models for content words from all types of POS among nouns, adjectives, adverbs and verbs, while the second category, Verbs, tests the models only for verbs.

2 Related Work

Various attempts have been made for metaphor detection in recent years, but only a few of them utilize the power of distributed representation of words (Bengio et al., 2003) combined with deep neural networks. Rei et al. (2017) proposed and evaluated the first deep neural network for metaphor identification on two datasets, Saif M. Mohammad and Turney (2016) and Tsvetkov et al. (2014). Do Dinh and Gurevych (2016) explore MLP classifier with trainable word embeddings on VUAMC corpus and achieve comparable results to other systems which use corpus-based or based on handcrafted features.

Other attempts which employ supervised learning approaches for metaphor detection on VUAMC corpus involve the use of logistic classifier (Beigman Klebanov et al., 2014) on a set of features, which include unigrams, topic models, POS, and concreteness features. Later, Beigman Klebanov et al. (2015) showed a significant improvement by re-weighting examples for cost sensitive learning and experimenting with concreteness information. Gargett and Barnden (2015) focused on utilizing the interactions between concreteness, imageability, and affective meaning for metaphor detection. Rai et al. (2016) explored Conditional Random Fields with syntactic, conceptual, affective, and contextual (word embeddings) features. Beigman Klebanov et al. (2016) experimented with unigrams, WordNet (Miller, 1995) and VerbNet (Schuler, 2006) based features for detection of verb metaphors.

3 Data

The dataset provided for this task is VU Amsterdam Metaphor Corpus (VUAMC). VUAMC is extracted from the British National Corpus (BNC Baby) and is annotated using MIPVU Procedure (Steen, 2010). It contains examples from four genres of text: Academic, News, Fiction and Conversation.

Table 1 and Table 2 summarize the statistics of the data for this shared task.

Proceedings of the Workshop on Figurative Language Processing, pages 115–120
New Orleans, Louisiana, June 6, 2018.

	Content Tokens	% Metaphors
Training Set	72611	15.2%
Test Set	22196	17.9%

Table 1: Summary of data statistics for All POS category (Content Tokens: nouns, adjectives, adverbs and verbs)

	Content Tokens	% Metaphors
Training Set	17240	27.8%
Test Set	5873	29.9%

Table 2: Summary of data statistics for Verbs category (Content Tokens: verbs)

4 System Description

This section describes our proposed system for this shared task, which we call Di-LSTM Contrast (illustrated in Figure 1[1]) and is divided into three modules trained in an end to end setting. The input to the model is given as pre-trained word embeddings. An encoder uses these word embeddings to encode the context of the sentence with respect to the target word using forward and backward LSTMs (Hochreiter and Schmidhuber, 1997). The output from the encoder is fed to the feature selection module (section 4.2) for generating contrast-based features for the token word. The classifier module (section 4.3) then predicts the probabilities for the target word being metaphoric.

4.1 Context Encoder

The context encoder is inspired by Bidirectional LSTM (BLSTM, Graves and Schmidhuber (2005)). Given an input sentence $S = \{w_1, w_2, ...w_n\}$, with n as the number of tokens in a sentence and i as the index of target token, we make two sets $A = \{w_1, w_2, ...w_i\}$ and $B = \{w_n, w_{n-1}, ...w_i\}$ and feed them into forward and backward LSTMs respectively. The motivation for this split is to produce the context with respect to the target word (w_i).

$$h_f = LSTM_f(A)$$

$$h_b = LSTM_b(B)$$

The hidden states $h_f \in \mathbb{R}^d$ and $h_b \in \mathbb{R}^d$, so obtained from forward and backward LSTMs are

[1]Figure generated using https://www.draw.io/

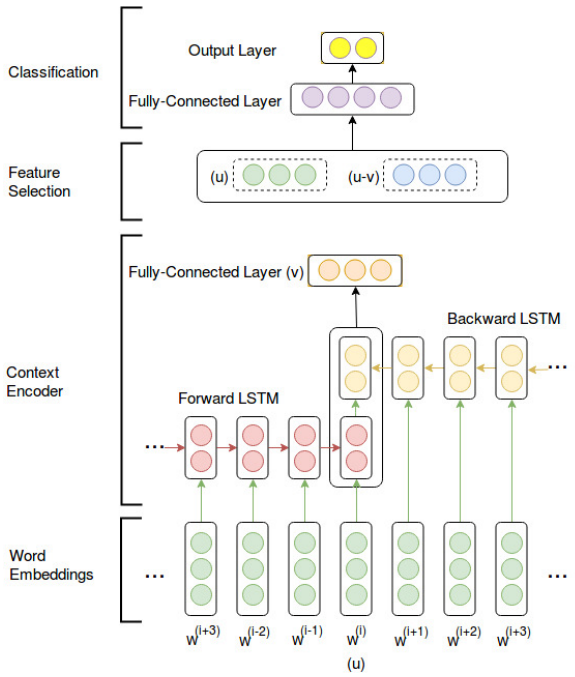

Figure 1: The Architecture of DiLSTM Contrast Model

combined by concatenation or averaging, followed by a fully connected layer to produce $v \in \mathbb{R}^d$, the context encoding.

$$h = [h_f; h_b]$$

$$v = sigmoid(W_{(1)}h + b_{(1)})$$

$W_{(1)} \in \mathbb{R}^{(d \times 2d)}$ is the transformation weight matrix, and $b_{(1)} \in \mathbb{R}^d$ is bias.

4.2 Feature Selection

A combination of the context encoding (v) and the word vector of the target word $u = w_i$ is then fed to the classification module as

$$g = [u; (u - v)]$$

The intuition behind this feature set $g \in \mathbb{R}^{2d}$ is that the properties of the word and the difference between the general and contextual meanings play a major role in determining the metaphoricity of a word (Steen, 2010).

4.3 Classification

The vector g from the previous module is transformed to a hidden layer and then to the output layer to obtain the softmax probabilities ($p \in \mathbb{R}^2$) for metaphoricity.

$$h_1 = sigmoid(W_{(2)}g + b_{(2)})$$

Model Variants	Val.	Test All POS	Test Verbs
DC (avg)	0.541	0.538	0.572
DC	0.554	0.542	0.584
DC +R	0.570	0.562	0.590
DC +RL	**0.575**	0.570	**0.605**
Task Baseline	-	**0.589**	0.600

Table 3: Comparision of F1 scores on Validation, All POS (Test) and Verbs (Test) scores between the various approaches. DC = DiLSTM Contrast with concatenation, DC (avg) = DiLSTM Contrast with averaging, R = Reweighting of Examples, L = Additional Linguistic Features (Baseline), Task Baseline = The baseline system used by the task organizers

$$p = softmax(W_{(4)}h_1 + b_{(4)})$$

$W_{(2)} \in \mathbb{R}^{(m \times 2d)}, W_{(4)} \in \mathbb{R}^{(2 \times m)}$ are the weight matrices and $b_{(2)} \in \mathbb{R}^m, b_{(4)} \in \mathbb{R}^2$ are the biases.

To enable the use of some additional binary baseline features (section 6.3), we modify the equations as

$$h_1 = sigmoid(W_{(2)}g + b_{(2)})$$

$$l_2 = W_{(3)}g_{baseline} + b_{(3)}$$

$$l_1 = W_{(4)}h_1 + b_{(4)}$$

$$p = softmax(\alpha\, l_1 + (1 - \alpha)\, l_2)$$

$W_{(2)} \in \mathbb{R}^{(m \times 2d)}, W_{(3)} \in \mathbb{R}^{(2 \times k)}, W_{(4)} \in \mathbb{R}^{(2 \times m)}$ are the corresponding weight matrices, $b_{(2)} \in \mathbb{R}^m, b_{(3)} \in \mathbb{R}^2, b_{(4)} \in \mathbb{R}^2$ are the corresponding biases, $g_{baseline} \in \mathbb{R}^k$ is the baseline feature vector and α is a trainable variable which determines the weights to be given to the baseline features and the contrast features.

5 Implementation Details

We split the provided training data in 90:10 ratio as training set and development set. We use this development set to tune our hyperparameters for the different variations of our model. We use 300-dimensional GloVe vectors (Pennington et al., 2014) trained on 6B Common Crawl corpus as word embeddings, setting the embeddings of out-of-vocabulary words to zero. To prevent overfitting on the training set, we use dropout regularization (Srivastava et al., 2014) and early stopping (Yao et al., 2007). We set the minibatch size to 50 examples and we zero pad the A and B split sets (as defined in section 4.1). More details on the hyperparameter settings can be found in the table 4.

Hyperparameter	Value
GloVe dimension (d^+)	300
Hidden dimension (m^+)	200
Dropout	0.15
Initial learning rate	0.3
# epochs	30
Early stopping*	2

Table 4: Hyperparameter settings for out best performing model; +: d, m as indicated in section 4; *: stop training after loss divergence for 2 consecutive iterations .

We use TensorFlow (Abadi et al., 2015) library in Python[2] to implement our model. AdaGrad (Duchi et al., 2011) optimizer is used for optimization of the model.

We train our models only on the All POS category training set, and evaluate it on the test sets of both All POS and Verb categories, since the training set for all the verbs is a subset of the ALL POS category .

6 Experiments and Evaluation

In this section, we present evaluation results for our model. Table 3 shows their comparison on the test set using F1 score as the metric for evaluation. Experimental results indicate that our model generalizes well on the tests for both the task categories and the performance trends on tests are consistent with those on validation. Table 3 also shows the performance comparison of the variants of our model with the baseline results for the shares task provided by the organizers. Our best performing model surpasses the baseline results on the Verbs category, while it achieves a lesser but comparable performance with the baseline on

[2] https://www.python.org/

Text Genre	All POS			Verbs		
	P	R	F	P	R	F
Academic	0.641	0.683	0.661	0.736	0.753	0.744
Conversation	0.346	0.724	0.469	0.308	0.729	0.433
Fiction	0.413	0.596	0.488	0.416	0.665	0.512
News	0.566	0.591	0.578	0.643	0.665	0.654
Average	0.491	0.648	0.549	0.525	0.703	0.585
Overall	0.511	0.644	0.570	0.529	0.708	0.605

Table 5: Analysis of our best performing system on the Test Sets (both categories). P = Precision. R = Recall, F = F1 Score

All POS category.

6.1 Experiment with the Encoder

We experiment with the combining function of the hidden states of forward and backward LSTM (in section 4.1) using both averaging and concatenation. The validation results on both the categories show that concatenation performs much better than averaging. This observation is supported by the fact that concatenation followed by a fully connected layer allows more parameterized interactions between the two states than averaging.

6.2 Re-weighting of Training Examples

We employ cost-sensitive learning (Yang et al., 2014) by re-weighting examples for our model. This brings an appreciable improvement in the performance of our model, 1.6% F1 gain on Validation, 2.0% on All POS category (Test) and 0.6% on verb category (Test). This increment in the performance agrees with the previous works on metaphor detection (Beigman Klebanov et al., 2015, 2016) which show the effectiveness of re-weighting training examples on VUAMC corpus.

6.3 Additional Baseline Features

The use of baseline features like WordNet (Miller, 1995) features, part-of-speech tags and Concreteness features (Brysbaert et al., 2014) in our model additionally improves the F1 score by 0.8% on the All POS category (Test) and 1.5% on verb category (Test), though it shows a relatively lesser improvement on the Validation set.

To obtain the POS-tag-based features, we encode the POS tag of the target tokens into a one-hot vector. By Wordnet features, we refer to one-hot encoding of the 26 class classification of the words based on their general meaning. The concreteness features represent the concatenation of the one hot representation of concreteness-mean-binning-BiasDown, and concreteness-mean-binning-BiasUp features (as indicated in Beigman Klebanov et al. (2015, 2016)).

7 Analysis

After the completion of the shared task, we downloaded the publicly available labels of the test data to analyze the results of our best performing model across all the four genres of text (section 3) on both the categories (as shown in the Table 5). Our system performs comparatively better on academic and news texts than on conversation and fiction texts.

8 Conclusion and Future Work

We described a deep neural architecture Di-LSTM Contrast Network for metaphor detection, which we submitted for Metaphor Shared Task 2018 (Leong et al., 2018). We showed that our system achieves appreciable performance solely by using the contrast features, generated by our model using pre-trained word embeddings. Additionally, our model gets a significant performance boost from the use of extra baseline features, and re-weighting of examples.

For our future work, we plan to experiment with CNNs along with LSTM for capturing the context representation of the sentence in light of the target word. Another interesting idea is the use of attention mechanism (Mnih et al., 2014), which has proven to be effective in many NLP tasks.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful reviews and suggestions.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

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. Association for Computational Linguistics.

Beata Beigman Klebanov, Chee Wee Leong, and Michael Flor. 2015. Supervised word-level metaphor detection: Experiments with concreteness and reweighting of examples. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 11–20. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Marc Brysbaert, AB Warriner, and V Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *BEHAVIOR RESEARCH METHODS*, 46(3):904–911.

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

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

Andrew Gargett and John Barnden. 2015. Modeling the interaction between sensory and affective meanings for detecting metaphor. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 21–30. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. 2005 special issue: Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Netw.*, 18(5-6):602–610.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

George Lakoff. 1993. *The contemporary theory of metaphor*, 2 edition. Cambridge University Press.

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*, New Orleans, LA.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. 2014. Recurrent models of visual attention. *CoRR*, abs/1406.6247.

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.

Sunny Rai, Shampa Chakraverty, and Devendra K. Tayal. 2016. Supervised metaphor detection using conditional random fields. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 18–27. Association for Computational Linguistics.

Marek Rei, Luana Bulat, Douwe Kiela, and Ekaterina Shutova. 2017. Grasping the finer point: A supervised similarity network for metaphor detection. *CoRR*, abs/1709.00575.

Ekaterina Shutova Saif M. Mohammad and Peter D. Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics (*Sem)*, Berlin, Germany.

Karin Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

G. Steen. 2010. *A Method for Linguistic Metaphor Identification: From MIP to MIPVU*. Converging evidence in language and communication research. John Benjamins Publishing Company.

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), volume 1, pages 248–258.

X. Yang, A. Loukina, and K. Evanini. 2014. Machine learning approaches to improving pronunciation error detection on an imbalanced corpus. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 300–305.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315.