Using Language Learner Data for Metaphor Detection

Egon W. Stemle
Eurac Research
Bolzano-Bozen, Italy
egon.stemle@eurac.edu

Alexander Onysko
Alpen-Adria-Universität
Klagenfurt a.W., Austria
alexander.onysko@aau.at

Abstract

This article describes the system that participated in the shared task (ST) on metaphor detection (Leong et al., 2018) on the Vrije University Amsterdam Metaphor Corpus (VUA). The ST was part of the workshop on processing figurative language at the 16th annual conference of the *North American Chapter of the Association for Computational Linguistics* (NAACL2018).

The system combines a small assertion of trending techniques, which implement matured methods from NLP and ML; in particular, the system uses word embeddings from standard corpora and from corpora representing different proficiency levels of language learners in a LSTM BiRNN architecture.

The system is available under the APLv2 open-source license.

1 Introduction

Ever since conceptual metaphor theory was laid out in Lakoff and Johnson (1980), the most vexing question has remained a methodological one: how can conceptual metaphors be reliably identified in language use? Although manual identification was put on a stronger methodological footing with the Metaphor Identification Procedure (MIP) ("Pragglejaz Group", 2007) and its elaboration into MIPVU (Steen et al., 2010), fuzzy areas remain due to the fact that conceptual metaphors can vary between primary metaphors and complex metaphors (cf. Grady, 1997). Furthermore, highly conventionalized metaphorical expressions might not be processed in the same way as novel metaphors. The core process of manual metaphor identification is not completely unproblematic either since it can be difficult to establish whether the meaning of a lexical unit in its context deviates from its basic meaning or not. In the face of that slippery terrain, automatic metaphor identification emerges as an extremely challenging task. An increasing volume of research since the start of annual workshops at NAACL in 2013 has shown first promising results using different methods of automated metaphor identification (see for example Shutova et al. (2015) and Klebanov et al. (2016) for previous events). The current shared task of metaphor identification provided a further opportunity to put the computational spotting of metaphors to the test.

Our bid for this task combines (cf. Section 2) `fastText` word embeddings (WEs) with a single-layer long short-term memory bidirectional recurrent neural network (BiRNN) architecture. The input, sequences of WE representations of words, is fed into the BiRNN which predicts metaphorical usage for each word.

The WEs were trained (cf. Section 4.2) on different large corpora (BNC, Wikipedia, enTenTen13, ukWaC) and on the Vienna-Oxford International Corpus of English (VOICE) as well as on the TOEFL11 Corpus of Non-Native English. The latter corpus was used, among others, in the First Native Language Identification Shared Task (Tetreault et al., 2013) held at the *8th Workshop on Innovative Use of NLP for Building Educational Applications* as part of NAACL-HLT 2013.

We were led by the idea (cf. Section 2.3) that metaphorical language use changes while gaining proficiency in a language, and so we hoped to be able to utilise the information contained in corpora of different proficiency levels.

The paper is organised as follows: We present our system design with related work in Section 2, the implementation in Section 3, and the experimental setup with an evaluation in Section 4. Section 5 concludes with an outlook on possible next steps.

Proceedings of the Workshop on Figurative Language Processing, pages 133–138
New Orleans, Louisiana, June 6, 2018. ©2018 Association for Computational Linguistics

2 Design

Generally, our design builds upon the foundation laid out by Collobert et al. (2011) for a neural network (NN) architecture and learning algorithm that can be applied to various natural language processing tasks. The most related task specific design is given in Do Dinh and Gurevych (2016) who used a NN in combination with WEs to detect metaphors. In contrast to our study, they used a dense multi-layer NN while we adapted the design of Stemle (2016a,b), who combined WEs with a recurrent NN (RNN) to predict part-of-speech (PoS) tags of computer-mediated communication (CMC) and Web corpora for German and Italian. RNNs are usually considered to be more suitable for labelling sequential data such as text.

2.1 Word Embeddings

Recently, state-of-the-art results on various linguistic tasks were accomplished by architectures using neural-network based WEs. Baroni et al. (2014) conducted a set of experiments comparing the popular word2vec (Mikolov et al., 2013a,b) implementation for creating WEs with other well-known distributional methods across various (semantic) tasks. These results suggest that the WEs substantially outperform the other architectures on semantic similarity and analogy detection tasks. Subsequently, Levy et al. (2015) conducted a comprehensive set of experiments that suggest that much of the improved results are due to the system design and parameter optimizations, rather than the selected method. They conclude that "there does not seem to be a consistent significant advantage to one approach over the other".

WEs provide high-quality low dimensional vector representations of words from large corpora of unlabelled data. The representations, typically computed using NNs, encode many linguistic regularities and patterns (Mikolov et al., 2013b).

2.2 Bidirectional Recurrent Neural Network

NNs consist of a large number of simple, highly interconnected processing nodes in an architecture loosely inspired by the structure of the cerebral cortex of the brain (O'Reilly and Munakata, 2000). The nodes receive weighted inputs through their connections on one side and *fire* according to their individual thresholds of their shared activation function. A firing node passes on an activation to all connected nodes on the other side. During learning the input is propagated through the network and the actual output is compared to the desired output. Then, the weights of the connections (and the thresholds) are adjusted step-wise so as to more closely resemble a configuration that would produce the desired output. After all training data have been presented, the process typically starts over, and the learned output values will usually be closer to the desired values.

Recurrent NNs (RNNs), introduced by Elman (1990), are NNs where the connections between the elements are directed cycles, i.e. the networks have loops, and this enables the NN to model sequential dependencies of the input. However, regular RNNs have fundamental difficulties learning long-term dependencies, and special kinds of RNNs need to be used (Hochreiter, 1991); a very popular one is the so called long short-term memory (LSTM) network proposed by Hochreiter and Schmidhuber (1997).

Bidirectional RNNs (BiRNN), introduced by Schuster and Paliwal (1997), extend unidirectional RNNs by introducing a layer, where the directed cycles enable the input to flow in opposite sequential order. While processing text, this means that for any given word the network not only considers the text leading up to the word but also the text thereafter.

Overall, we benefit from available labelled data with this design but also from large amounts of available unlabelled data.

2.3 Language Learner Data

Our experimental design also utilizes data from language learner corpora. This is based on the intuition that metaphor use might vary depending on learner proficiency. Beigman Klebanov and Flor (2013) indeed found a correlation between higher proficiency ratings of learner texts and a higher density of metaphors in these texts. Their study is also one of the few in the field of automated metaphor detection that are concerned with learner language. Their aim, however, is quite different to the current study as they try to establish annotations for metaphoric language use that can help to train an automated classifier of metaphors in test-taker essays. The current study, by contrast, utilizes learner corpus data to build WEs among other corpora representing written standard language. Learner language could be a particularly helpful source of information for automated metaphor de-

tection via WEs as learner language provides different usage patterns compared to WEs derived from standard language corpora.

3 Implementation

We maintain the implementation in a source code repository[1]. Our system uses sequences of word features as input to a BiRNN with a LSTM architecture.

3.1 Word Embeddings

We use gensim[2], a Python tool for unsupervised semantic modelling from plain text, to load precomputed WE models and to compute embedding-vector representations of words. Words missing in a WE model, i.e. out-of-vocabulary words (OOV), are first estimated by looking at a fixed context of their non-OOV words. If this fails, OOVs are mapped to their individual, randomly generated, vector representations.

3.2 Neural Network

Our implementation uses Keras (Chollet, 2015), a high-level NNs' library written in Python, on top of TensorFlow (Abadi et al., 2016), an open source software library for numerical computation.

The number of input layers corresponds to the number of employed feature sets. For multiple feature sets, e.g. multiple WE models or additional PoS tags, sequences are concatenated on the word level such that the number of features for an individual word grows.

Input sequences have a pre-defined length and represent original textual sentence segments. In case a sentence is longer than the sequence length, the input is split into multiple segments. And if a segment is shorter than the sequence length, the remaining slots are padded, i.e. they are filled with identical dummy information.

Each input layer feeds into a masking layer such that the padded values from the input sequence will be skipped in all downstream layers.[3] The masked input is fed into a bidirectional LSTM layer that, in turn, projects to a fully connected output layer that is activated by a softmax function.

The output is a single sequence of matching length with labels indicating whether the corresponding word is used metaphorically or not.

During training, we use dropout for the linear transformation of the recurrent state, i.e. the network drops a fraction of recurrent connections, which helps prevent overfitting (Srivastava et al., 2014); and we use a weighted categorical cross-entropy loss function to counteract the fact that far fewer words in our sequences are labelled as metaphorical than non-metaphorical, which usually hampers classification performance (cf. Kotsiantis et al., 2006).

4 Experiments and Results

Participants of the ST could either participate in the metaphor prediction tracks for verbs only, all content part-of-speech only, or both. For a given text in VUA, and for each sentence, the task was to predict metaphoricity for each verb or content word respectively, and submit the result to CodaLab[4] for evaluation. Results were calculated as the harmonic average of the precision and recall (F1-score) of the metaphoricity label. We participated with our system in both tasks.

The remainder of this section introduces the official data set, our WE models and describes our fixed hyper-parameters. The results of different combinations of WE models are shown in Table 1. Also note that *all results* in this paper refer only to the all content part-of-speech task.

4.1 Shared Task Data

The VUA, the corpus that was used in the shared task, originates from the British National Corpus (BNC). Altogether, it is comprised of 117 texts covering four genres (academic, conversation, fiction, news). For the ST, VUA was pre-divided by the organisers into a training and a test set. The training set was labelled and could be used to train classifiers, while the participants were supposed to label the test set and submit it. The distribution of metaphorical vs. non-metaphorical labels was imbalanced with a ratio of roughly 1:6 (11044 : 61567).

4.2 Word Embedding Models

We use pre-built WE models of the following corpora: *BNC* and *enTenTen13* web cor-

[1] https://github.com/bot-zen/

[2] https://radimrehurek.com/gensim/

[3] This is considered good practice and speeds up processing with long sequences and many padded values – with our rather short sequences it did not help much.

[4] http://codalab.org

	Tokens (Mio)	min Cnt	dim	T11 (low)	T11 (med)	T11 (high)	T11 (l+m+h)	VOICE	BNC	enTenTen13	ukWaC	ukWaC T11-size	Wikipedia17	F1-score on Test Set	10-fold CV Accuracy on Training Set μ — σ	
T11 (low)	0.3	1	50	X										0.207	0.917	0.016
T11 (med)	1.8	1	50		X									0.526	0.924	0.011
T11 (high)	1.4	1	50			X								0.514	0.930	0.007
T11 (l+m+h)	3.5	1	50				X							0.541	0.928	0.008
VOICE	1	1	50					X						0.495	0.923	0.010
BNC	100	5	100						X					0.597	0.942	0.005
enTenTen13	19,000	5	100							X				0.594	0.947	0.004
ukWaC	2100	5	100								X			**0.598**	0.945	0.004
ukWaC T11-size	3.5	1	50									X		0.564	0.933	0.009
Wikipedia17	ca 2300	5	300										X	0.586	**0.947**	0.003
	7			X	X	X						X		0.576	0.941	0.003
	7						X					X		0.567	0.936	0.008
	103.5			X	X	X			X					0.596	0.944	0.008
	103.5						X		X					**0.613**	0.945	0.005
	103.5								X			X		0.597	0.948	0.003
	104.5			X	X	X		X	X					0.601	0.950	0.004
	107						X		X			X		0.586	0.951	0.002
	108						X	X	X			X		0.550	0.948	0.003
	19,004.5			X	X	X		X		X				0.603	0.947	0.006
	21,400								X	X			X	0.605	0.951	0.003
	21,401							X	X	X			X	0.594	**0.953**	0.003
	21,404.5			X	X	X		X	X	X			X	0.597	0.952	0.003

Table 1: Overview of the word embedding models we used, and evaluation results for individual models and some combinations on the metaphor prediction track for *all content part-of-speech*.
Number of tokens in the original corpus, parameters `minCount` and `dim` for `fastText` during training of the models. Our calculated F1-scores on the official labelled test set (they should coincide with the organisers' results). The mean accuracy as well as the standard deviation in the accuracy for 10-fold cross validation runs on the training set.

pus (Jakubíček et al., 2013) from SketchEngine[5], as well as *Wikipedia17*[6] from `fastText` (Bojanowski et al., 2016).

We trained WE models using `fastText`'s SkipGram model with the default parameters[7] except for the two parameters `-minCount` (the minimal number of word occurrences) and `-dim` (size of word vectors). The two parameters were altered to take the smaller sizes of our corpora into account. See Table 1 for details.

Three individual models were trained for the different proficiency levels low, medium and high of the training subset of the *TOEFL11* (Blanchard et al., 2013); another model was trained for the full training set comprising all three proficiency levels. One model was trained for the *VOICE* (Seidlhofer et al., 2013), a corpus of English as it is spoken by a non-native speaking majority of users in different contexts.

Two models were trained for ukWaC (Baroni et al., 2009), a corpus constructed from the Web using medium-frequency words from the BNC as seeds. The first model for the full corpus and

[5] https://embeddings.sketchengine.co.uk/static/index.html

[6] https://fasttext.cc/docs/en/pretrained-vectors.html

[7] https://github.com/facebookresearch/fastText/archive/v0.1.0.zip

the second model for a random sample of documents approximating the token count of the full TOEFL11 training set.

4.3 Hyper-Parameter Tuning

Hyper-parameter tuning is important for good performance. The parameters of our system were optimised via an ad-hoc grid search in 3-fold cross validation (CV) runs.

Parameters were: NN optimizer (*rmsprop*, adadelta, adam), recurrent dropout rate for the LSTM layer (0.1, *0.25*, 0.5), dropout for the input layer (*0*, 0.1, 0.2), sequence length (5, 10, *15*, 50), learning epochs (3, 5, *20*, 32) and batch size (16, *32*, 64), and the network architecture, e.g. introducing a second LSTM abstraction layer or using a Gated Recurrent (GRU) layer instead of the LSTM layer. Furthermore, we trained WE models with different values for the `dim` (25, *50*, 100, 150, 200, 250) and `minCount` (*1*, 2, 5, 10) parameters.

The weight for the categorical cross-entropy loss function is calculated as the logarithm of the ratio of number of words vs. metaphorical labels. The context for estimating OOV words was set to 10.

Once set, we used the same configuration for all experiments.

5 Conclusion & Outlook

The combination of WEs with a BiRNN is capable of recognizing metaphorical usage of words better than many other already tested approaches. More importantly, our design does not rely on WordNet or VerbNet information, and does not need concreteness or abstractness information like many successful architectures from previous annual workshops at NAACL. Besides VUA, our system only needs running text.

The best result on the test set was achieved with a combination of TOEFL11 learner data and data from the BNC. So far, the results are encouraging—but also mixed—regarding our initial idea that metaphorical language use at different proficiency levels could be utilised to recognizing metaphorical usage of words. To this end, we are looking forward to output from the *European Network for Combining Language Learning with Crowdsourcing Techniques*[8], where potentially more and more fine-grained language learner data will be collected and made available.

[8] http://www.cost.eu/COST_Actions/ca/CA16105

Acknowledgements

The computational results presented have been achieved in part using the Vienna Scientific Cluster (VSC).

References

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, Savannah, GA. USENIX Association.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Marco Baroni, Georgiana Dinu, and German Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247. Association for Computational Linguistics. http://www.aclweb.org/anthology/P14-1023.

Beata Beigman Klebanov and Michael Flor. 2013. Argumentation-Relevant Metaphors in Test-Taker Essays. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 11–20. Association for Computational Linguistics.

Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. Toefl11: A corpus of non-native english. *ETS Research Report Series*, 2013(2):i–15.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

François Chollet. 2015. Keras: Deep Learning library for Theano and TensorFlow. https://github.com/fchollet/keras.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537. https://arxiv.org/abs/1103.0398.

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, San Diego, California. Association for Computational Linguistics.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Joseph Grady. 1997. *Foundations of Meaning: Primary Metaphors and Primary Scenes*. Ph.D. thesis, University of California, Berkeley.

Sepp Hochreiter. 1991. *Untersuchungen zu dynamischen neuronalen Netzen*. diploma thesis, TU München.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlỳ, and Vít Suchomel. 2013. The TenTen corpus family. In *7th International Corpus Linguistics Conference (CL 2013)*, pages 125–127, Lancaster. http://ucrel.lancs.ac.uk/cl2013/.

Beata Beigman Klebanov, Ekaterina Shutova, and Patricia Lichtenstein. 2016. Proceedings of the Fourth Workshop on Metaphor in NLP. In *Proceedings of the Fourth Workshop on Metaphor in NLP*. Association for Computational Linguistics.

Sotiris Kotsiantis, Dimitris Kanellopoulos, and Panayiotis Pintelas. 2006. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30.

George Lakoff and Mark Johnson. 1980. *Metaphors we Live by*. University of Chicago 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.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225. https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/570.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781. http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. *CoRR*, abs/1310.4546. http://arxiv.org/abs/1310.4546.

Randall C. O'Reilly and Yuko Munakata. 2000. *Computational Explorations in Cognitive Neuroscience Understanding the Mind by Simulating the Brain*. MIT Press. http://books.google.com/books?id=BLf34BFTaIUC{&}pgis=1.

"Pragglejaz Group". 2007. MIP: A Method for Identifying Metaphorically Used Words in Discourse. *Metaphor and Symbol*, 22(1):1–39.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Barbara Seidlhofer, Angelika Breiteneder, Theresa Klimpfinger, Stefan Majewski, Ruth Osimk-Teasdale, Marie-Luise Pitzl, and Michael Radeka. 2013. The Vienna-Oxford International Corpus of English (VOICE).

Ekaterina Shutova, Beata Beigman Klebanov, and Patricia Lichtenstein. 2015. Proceedings of the Third Workshop on Metaphor in NLP. In *Proceedings of the Third Workshop on Metaphor in NLP*. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958.

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. 00:238.

Egon W. Stemle. 2016a. bot.zen @ EmpiriST 2015 - A minimally-deep learning PoS-tagger (trained for German CMC and Web data). In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 115–119. Association for Computational Linguistics.

Egon W. Stemle. 2016b. bot.zen @ EVALITA 2016 - A minimally-deep learning PoS-tagger (trained for Italian Tweets). In *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016)*, Napoli, Italy.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Building Educational Applications Using NLP*, Atlanta, GA, USA. Association for Computational Linguistics.