Neural Metaphor Detection with a Residual biLSTM-CRF Model

Anonymous ACL submission

Abstract

Automatic metaphor detection is a task that consists on correctly labeling metaphoric uses of words in a given context. In this paper we present a novel resource-inexpensive architecture for metaphor detection based on a residual bidirectional long short-term memory and conditional random fields. Current approaches on this task rely on deep neural networks to identify metaphorical words, using additional linguistic features or word embeddings. We evaluate our proposed approach using different model configurations that combine embeddings, part of speech tags, and semantically disambiguated synonym sets. This evaluation process was performed using the training and testing partitions of the VU Amsterdam Metaphor Corpus. We use this method of evaluation as reference to compare the results with other current neural approaches for this task that implement similar neural architectures and features, and that were evaluated using this corpus. Results show that our system achieves competitive results with a simpler architecture compared to previous approaches.

1 Introduction

This paper presents a new model for automatic metaphor detection which has participated at the FigLang 2020 metaphor detection shared task. Our approach, which is based on neural networks, has been developed in the framework of [name of project anonymized for blind review] (Authors, 2018), a project devoted to the analysis of metaphors in mental health discourses.

As it is well known in Cognitive Linguistics, a conceptual metaphor (CM) is a cognitive process which allows to understand and communicate an abstract or diffuse concept in terms of a more concrete one (cf. e.g. Lakoff and Johnson (1980)). This process is expressed linguistically by using metaphorically used words (MUW).

The study of metaphor is a prolific area of research in Cognitive Linguistics, being the Metaphor Identification Procedure (MIP) (Pragglejaz Group, 2007) and its derivative MIPVU (Steen et al., 2019) the most standard methods for manual MUW detection. MIPVU is the method that was used to annotate the VU Amsterdam Metaphor Corpus (VUA corpus), used in FigLang 2020. Moreover, in the area of Corpus Linguistics, some methods have been developed for a richer annotation of metaphor in corpora (Ogarkova and Soriano Salinas, 2014; Shutova, 2017), (Authors, 2019).

CM is pervasive in natural language text and therefore it is crucial in automatic text understanding (Shutova, 2010). For this reason automated metaphor processing has become an increasingly important concern in natural language processing, as shown by the holding of the Metaphor in NLP workshop series (at NAACL-HLT 2013, ACL 2014, NAACL-HLT 2015, NAACL-HLT 2016 and NAACL-HLT 2018) and a growing body of research — see (Shutova, 2017) and (Veale et al., 2016) for quite recent reviews.

Automatic metaphor processing involves two main tasks: identifying MUW (metaphor detection or recognition) and attempting to provide a semantic interpretation for the utterance containing them (metaphor interpretation). This work deals with metaphor detection.

This problem has been mainly approached in the last decade by supervised and semi-supervised machine learning techniques but recently this paradigm has largely shifted to the use of deep learning algorithms, such as neural networks. Leong et al. (2018) report that all but one of participating teams on the 2018 VUA Metaphor Detection Shared Task used this kind of architectures. Our system follows this trend by trying to improve on previous neural network methods.

Below we describe the main related works (section 2). Next we present our methodology and model (section 3), experiments (section 4) and results (Section 5). We finish with the discussion and our overall conclusions (sections 6 and 7).

2 Background

Current approaches regarding metaphor recognition and interpretation include the works of Rosen (2018), which focus on metaphor interpretation, and Wu et al. (2018) and Mu et al. (2019), which focus on the detection of metaphorical instances in general corpora. Rosen (2018) developed an algorithm using deep learning techniques that uses a representation of metaphorical constructions in an argument-structure level. The algorithm allows for the identification of source-level mappings of metaphors. The author concludes that the use of deep learning algorithms including construction grammatical relations in the feature set improves the accuracy of the prediction of metaphorical source domains.

Wu et al. (2018) propose to use a Convolutional Neural Network - Long-Short Term Memory (CNN-LSTM) with a Conditional Random Field (CRF) or Softmax layer for metaphor detection in texts. They combine CNN and LSTM to capture both local and long-distance contextual information to represent the input sentences. Meanwhile, Mu et al. (2019) argue that using broader discourse features can have a substantial positive impact for the task of metaphorical identification. They obtain significant results using document embeddings methods to represent an utterance and its surrounding discourse. With this material a gradient boosting classifier is trained.

We propose a model that uses residual bidirectional long short-term memory (biLSTM) with a CRF, using ELMo embeddings along with additional linguistic features, such as part of speech tags (POS) and semantically disambiguated WordNet[1] synonym sets (synsets) (Fellbaum and Miller, 1998). Our model could be grouped in the same category as the aforementioned approaches: deep neural networks models for metaphor detection.

3 Model Description

The approaches mentioned in section 2 used the VUA corpus (Steen et al., 2010) in order to carry out model training and testing. They divided the training and test sets according to the VUA Metaphor Detection Shared Task specifications. To train and test our model we used the VUA corpus partitions, using ELMo embeddings to represent words and lemmas, and POS and synsets as additional linguistic features. ELMo (*Embeddings from Language Models*) embeddings (Peters et al., 2018) are derived from a bidirectional language model (biLM) and they are contextualized, deep and character based. ELMo embeddings have been successfully used in several NLP tasks.

To process the VUA corpus we used the Natural Language Toolkit (NLTK) (Loper and Bird, 2002) for Python, with this tool we performed tokenization, lemmatization, and POS tagging. Then we used Freeling (Padró and Stanilovsky, 2012) to obtain the respective synset of each token. Although NLTK provides a method for obtaining synsets – using POS tags or Lesk's Algorithm–, Freeling implements UKB (Agirre et al., 2014), a graph-based word sense disambiguation (WSD) algorithm that is used to obtain semantically disambiguated synsets. These features along the ELMo embeddings were used –in different configurations– as input for our model. We set a sequence padding value equal to 116, which is the maximum sentence length observed in the corpus. This process normalizes the input in order to train in batches, but might contribute to sparsity on training data.

We used one-hot encoded representation for POS, and computed local 100-dimension embeddings for synsets. In the case of POS, we have a small set of tags (43), and therefore resulting in a low dimensionality of the one-hot embeddings. For synsets, the computation of local embeddings provides the semantically disambiguated relations that exist between the units that compose the training data. These embeddings, in addition with their EMLo counterparts, shall provide enough contextual and semantic data to understand metaphorical instances of words.

The main architecture of our model (shown in Figure 1) is composed by a residual biLSTM (Kim et al., 2017; Tran et al., 2017) for sequence labeling. One of the particularities of this architecture lies in the implementation of an additive operation that takes the outputs from each biLSTM layer and combines them to calculate the residual connection between them, in order to obtain previously seen information from both instances.

After computing the residual connection from both biLSTM layers, our model includes a dropout

[1] Freeling implements WordNet version 3.0.

layer, followed by a time distributed layer in which a dense activation with 2 hidden units to each timestep is applied. We used ReLU (Nair and Hinton, 2010) as activation function in combination with a He-normal (He et al., 2015) kernel initialization function for the time distributed layer, which results in a zero-mean Gaussian distribution with a standard deviation equal to $\sqrt{\frac{2}{\hat{n}_l}}$. Finally, after the time distributed layer we used a conditional random field (CRF) implemented for sequence labeling (Lafferty et al., 2001).

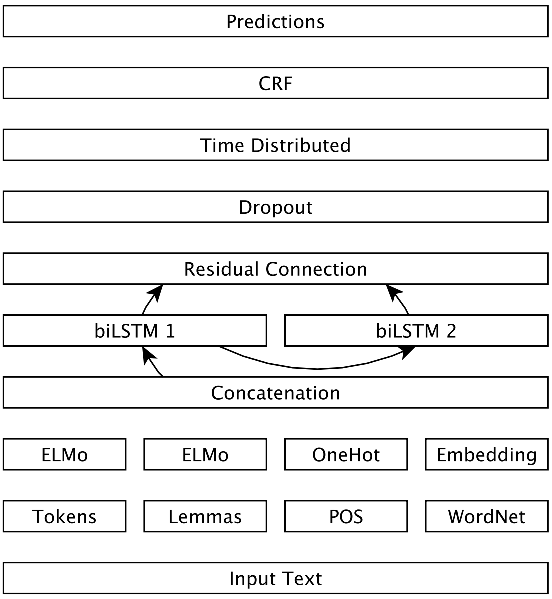

Figure 1: Summarized model diagram.

Given that the VUA corpus is composed by more negative –or literal– labels than positive –or metaphoric– labels, and that the sequence padding process added non-informative features to the input array, we opted to treat the training partition as an imbalanced dataset. We selected the Nadam optimizer (Dozat, 2016), which is based on Adam (Kingma and Ba, 2014) and tends to perform better with sparse data. This last optimization algorithm has two main components: a momentum and an adaptive learning rate component. Nadam modifies the momentum component of Adam using Nesterov's accelerated gradient (NAG). The Nadam update rule can be written as follows:

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{w}_t + \epsilon}} \left(\beta_1 \hat{m}_t + \frac{1-\beta_1}{1-\beta_1^t} \cdot \frac{\partial L}{\partial w_t} \right) \quad (1)$$

4 Experiments

To carry out the evaluation of our model we used the train and test splits provided in VUA shared task partitions (Shutova, 2017). In order to obtain a validation split we divided the training partition using the following percentages: 80% for training 20% for validation. With these partitions, we trained a total of 6 different model configurations: words and POS (W+POS); lemmas and POS (L+POS); words, POS and synsets (W+POS+SS); lemmas, POS and synsets (L+POS+SS); words, lemmas and POS (WL+POS); and words, lemmas, POS and synsets (WL+POS+SS).

In all cases we used the same training parameters, all model configurations were trained in batches for 5 epochs, using a learning rate = 0.0025. Then, the resulting models were evaluated –using the precision, recall and F$_1$ score metrics– on both the all POS metaphor detection task and the metaphoric verbs detection task.

5 Results

Regarding the all POS prediction task, the L+POS+SS model had the best performance with a 0.5729 in precision, 0.6027 in recall and an F$_1$ score equal to 0.5874. Overall, all configuration obtained a mean F$_1$ score of 0.58 being the WL+POS model the one with the lowest score (0.5615). Regarding the recall score, the highest observed value was obtained by the W+POS+SS model, with a recall equal to 0.6438.

Model	Precision	Recall	F$_1$
W+POS	0.5635	0.6098	0.5857
W+POS+SS	0.5313	**0.6438**	0.5822
L+POS	0.5685	0.5956	0.5817
L+POS+SS	**0.5729**	0.6027	**0.5874**
WL+POS	0.5064	0.6302	0.5615
WL+POS+SS	0.5601	0.6174	0.5873

Table 1: All POS task model comparison.

It could be said that a less diverse lexicon obtained by using lemmas instead of words to obtain embeddings, helped to improve the performance of the L+POS+SS model. Nevertheless, when comparing the W+POS and L+POS configuration, both obtained similar results, with less than 1% difference in performance between them. Meanwhile, when comparing the W+POS+SS and L+POS+SS models, it can be observed that both models obtained similar F_1 scores, but a variation of 4% between the precision and recall that favours precision in the L+POS+SS model, and recall in the W+POS+SS model.

In the case of the metaphoric verb labeling task, the W+POS model obtained the best scores in precision and F_1 score (0.6695 and 0.6543 accordingly), while the W+POS+SS model obtained the highest recall value (0.7032). Overall, the mean F_1 score of all configurations was equal to 0.6411, being the WL+POS the poorest performing configuration with a F_1 score of 0.6101. Similarly to the all POS task, the W+POS+SS and L+POS+SS configurations obtained precision and recall scores with a difference of 6% in both metrics.

Model	**Precision**	**Recall**	**F_1**
W+POS	**0.6695**	0.6397	**0.6543**
W+POS+SS	0.5933	**0.7032**	0.6436
L+POS	0.6398	0.6413	0.6405
L+POS+SS	0.6544	0.6474	0.6509
WL+POS	0.5576	0.6735	0.6101
WL+POS+SS	0.6201	0.6779	0.6477

Table 2: Verbs task model comparison.

Unlike in the all POS task, combining features did not improve the performance of the models for verbs labeling. While using synsets to disambiguate the meaning of the different words or lemmas that were fed to the model, using ELMo embeddings and POS tags yielded better results in this task. One of the possible explanations for this behavior could be that verbs tend to be more polysemous than nouns and, therefore, obtain greater benefit from this feature. According to WordNet statistics[2], verbs have an average polisemy index of 2.17, while nouns have an average of 1.24.

6 Discussion

Our proposed architecture has similarities to other current approaches such as Wu et al. (2018) who propose a LSTM with Softmax model, and Mu et al. (2019) who implement an XGBoost classifier using ELMo embeddings. In comparison to these approaches, our model shows an improvement in precision on the verb labeling task with a value equal to 0.6695, while Mu et al. (2019) reported a precision score of 0.600[3], and Mu et al. (2019) a precision equal to 0.589. Nevertheless Wu et al. (2018) reported the highest F_1 score (0.671), and Mu et al. (2019) the highest recall (0.771).

All POS task			
Model	**Precision**	**Recall**	**F_1**
Wu et al. (2018)	0.608	0.700	0.651
L+POS+SS	0.5729	0.6027	0.5874
Verbs task			
Wu et al. (2018)	0.600	0.763	**0.671**
Mu et al. (2019)	0.589	**0.771**	0.668
W+POS	**0.6695**	0.6397	0.6543

Table 3: Comparison with other current approaches.

Regarding the all POS labeling task, the model presented by (Wu et al., 2018) performs better in all metrics, with a difference of 3% in precision, 10% in recall and 8% in F_1 score. It has to be noted that our model presents a simpler architecture (as shown in section 3). Wu et al. (2018) trained their model using 200 biLSTM hidden states and 100 CNN units for 15 epochs, and trained it 20 times using an ensemble method. On the other hand, the most simple W+POS architecture that we presented takes an average time of 5 minutes by epoch[4] to train and validate, thus producing a less complex model that is faster and less expensive to train.

On both tasks the poorest performing configuration was WL+POS, combining these features improved recall but lowered both precision and F_1. Combining words and lemmas might create redundancy in certain features that is not possible to leverage using POS. On the other hand, while the dimensionality becomes higher than the previous configuration (1024 + 1024 + 43), once synsets are added in the WL+POS+SS architecture (and increasing the feature dimensionality by 100) the performance of the model improves on both precision and recall on the all POS task, and in all metrics on the verbs task.

One of the strategies that we implemented to leverage the imbalance of the training data was using a kernel initialization function. The He-normal function uses the size of the last layer in order to generate weights that have different ranges. In this case, the time distributed layer is activated using RELu, and takes the size of the dropout layer and then initializes it with a He-normal distribution.

[2] https://wordnet.princeton.edu/documentation/wnstats7wn

[3] Both authors reported metric results using three digits.

[4] The model was trained using a shared NVIDIA Tesla P100 GPU.

7 Conclusions and further work

In this paper we have described the system we have presented at the FigLang 2020 metaphor detection shared task. Our approach is based on neural networks using a residual biLSTM with a CRF and using ELMo embeddings along with the inclusion of linguistic features (several combinations of words, lemmas, POS and WordNet synsets). The system achieves competitive results with a simpler architecture compared to systems found in the literature. Such systems implement similar elements such as the use of bidirectional LSTM, CRF and ELMo embeddings in different configurations, and with different combination of linguistic features.

As future work, we plan to further analyse which POS benefits most from the inclusion of synset information. Other aspect we want to explore is how to deal with imbalanced data, i.e. how we can leverage a dataset with only two classes (metaphoric/literal) where most of the samples are literal. Other interesting questions that deserve more research is the effects on optimal dimensionality of the addition of linguistic information. Other features that could be implemented are concreteness value of certain words, or as an strategy to balance classes according to the influence that this feature has on literal and metaphoric classes.

Other future lines of work might include the implementation of this type of model for the detection of metaphors and source domain identification in Spanish. Current developments on metaphor detection are being carried out mainly in English, while this is a great resource it could be interesting to create resources in other languages to broaden the scope of metaphor detection and interpretation. A possible pipeline could be configured with two separated model, one that performs the detection of metaphorical words, followed by another classifier that predicts the domain of those metaphors.

References

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

Timothy Dozat. 2016. Incorporating nesterov momentum into adam. In *Proceedings of the International Conference on Learning Representations (ICLR-2016) - Workshop Track*, San Juan (Puerto Rico).

C. Fellbaum and G.A. Miller. 1998. *WordNet: An Electronic Lexical Database*. Language, speech, and communication. MIT Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852.

Jaeyoung Kim, Mostafa El-Khamy, and Jungwon Lee. 2017. Residual LSTM: design of a deep recurrent architecture for distant speech recognition. *CoRR*, abs/1701.03360.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. ArXiv: 1412.6980.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

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

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.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70.

Jesse Mu, Helen Yannakoudakis, and Ekaterina Shutova. 2019. Learning Outside the Box: Discourse-level Features Improve Metaphor Identification. *arXiv:1904.02246 [cs]*. ArXiv: 1904.02246.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

Anna Ogarkova and Cristina Soriano Salinas. 2014. *Variation within universals: The 'metaphorical profile' approach to the study of ANGER concepts in English, Russian and Spanish*, Metaphor and Intercultural Communication. Bloomsbury, London. ID: unige:98101.

Lluís Padró and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey. ELRA.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Pragglejaz Group. 2007. MIP: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol*, 22(1):1–39.

Zachary Rosen. 2018. Computationally Constructed Concepts: A Machine Learning Approach to Metaphor Interpretation Using Usage-Based Construction Grammatical Cues. In *Proceedings of the Workshop on Figurative Language Processing*, pages 102–109, New Orleans, Louisiana. Association for Computational Linguistics.

Ekaterina Shutova. 2010. Models of Metaphor in NLP. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 688–697.

Ekaterina Shutova. 2017. *Annotation of Linguistic and Conceptual Metaphor*, pages 1073–1100. Springer Netherlands, Dordrecht.

Gerard Steen, Lettie Dorst, J Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Tryntje Pasma. 2019. *MIPVU: A manual for identifying metaphor-related words*, pages 24–40.

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*. John Benjamins.

Quan Tran, Andrew MacKinlay, and Antonio Jimeno-Yepes. 2017. Named entity recognition with stack residual LSTM and trainable bias decoding. *CoRR*, abs/1706.07598.

Tony Veale, Ekaterina Shutova, and Beata Beigman Klebanov. 2016. Metaphor: A Computational Perspective. *Synthesis Lectures on Human Language Technologies*, 9(1):1–160.

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.