Concept Tagging for the Movie Domain by using Tranfer Learning and Named Entity Recognition.

Giovanni De Toni (197814)
University of Trento
Via Sommarive, 9, 38123 Povo,Trento TN
giovanni.detoni@studenti.unitn.it

Abstract

Neural Networks have produced remarkable results in many Natural Language Processing tasks, for example, when tasked to assigning concepts to words of a sentence. Their successes are made possible by employing good word representations (embeddings) which a Neural Network can understand. This work evaluates several newly developed pre-trained embeddings (ELMo, BERT and ConceptNet) on the task of tagging sequences from the movie domain. We then compare the measurements with previous results of the literature.

1 Introduction

Concept Tagging sequence is a classical problem in the NLP field. It consists of assigning to each word of a sentence a given concept which represents the meaning of that word (e.g., "*star wars*" is a movie title, hence the concept " *movie.title*"). Over the past years, it was extensively studied and several techniques were developed such to efficiently tag words given a pool of concepts. The most basic methods are statistical language models (e.g., Weight Finite State Automa) which are easy to train and yield quite impressive results with a little tuning. However, with the rise of recurrent neural networks, we can now train models which have better performances than their statistic counterparts. Moreover, thanks to the use of specific word embeddings (e.g., word2vec, GloVe, ELMo etc.), we can boost the performances even more. This work takes as a starting point a previous work on the topic (Gobbi et al., 2018) and it tries to improve the results of some of the neural models presented there, by adding new features (NER and POS tags) and by trying to employ more intelligent embeddings (ELMo, BERT and ConceptNet). The report is structured as follow: the second section describe the models used for the experiments, the third section describes the choosen embeddings. The forth and fitfth sections describe the analysis of the corpora and the embeddings. Ultimately, the sixth and seventh section reports the experiments performed and the conclusions.

2 Models

To evaluate how the performance changes with respect to the previous results we selected two models from the original work. We concentrated on the neural networks approaches by selecting the ones which held the best results over the test set. We chose the following architectures: LSTM and LSTM-CRF. Moreover, we extended the features used by the various models by providing also the Part-of-Speech tags and the Named Entities Recognition (NER) entities as one-hot-encoded vectors. Long-Short Term Memor (LSTM) (Hochreiter and Schmidhuber, 1997) is a model which extends and improve the previous RNN by solving the vanishing/exploding gradients problem. We tested a simple LSTM which receives in input the embeddings of the words concatenated with the new features. Moreover, we tested another LSTM model which uses also the characters embeddings obtained using a convolutional layer (LSTM-CHAR-REP). We experimented also with the LSTM-CRF model (Yao et al., 2014; Huang et al., 2015) in which the LSTM provides the class scores for each token and then the Viterbi algorithm is used to determine the labels at a global level. Again, this model was also tested by adding the new features concatenated to the word embeddings. Moreover, an LSTM-CRF version extended with the characters convolution was also evaluated (LSTM-CRF-CHAR-REP).

3 Embeddings

As a first step to improve the performances of the previous work we concentrated on evaluating several recently proposed pre-trained language models to produce words representations. Generally speaking, language representations can be either *context-free* or *contextual*. Context-free means that the word representation is generated without looking at the context in which the word is used. For instance, the word "*season*" may have different meanings with respect to the context in which it is placed (e.g., "*Summer is my favourite season*" and "*Add some season to the chicken and then serve*"), but in a context-free model it will get the same representation (embedding). A contextual representation takes care also of the context in which a word is used. Moreover, the contextual representations can be further divided into two other categories: *unidirectional* or *bidirectional*. A unidirectional model contextualizes a given term by just looking at the words on the right (or on the left) of it. A bidirectional model instead looks both on the left and on the right of the target word before producing a representation. There are also *swallow bidirectional models* which combine two unidirectional models (one for the left side and one for the right side) to give a more complete representation. In our work, we tested three different embeddings, one for each category (context-free, contextual swallow bidirectional and contextual bidirectional): *ConceptNet*, *ELMo* and *BERT*.

3.1 ConceptNet

ConceptNet Numberbatch (Speer et al., 2017) is a context-free language model. It is a set of word embeddings and it was made by combining previously existing models, namely, GloVe (Pennington et al., 2014), word2vec (Mikolov et al., 2013) and OpenSubtitles 2016 (Lison and Tiedemann, 2016). Moreover, it leverages the data of the ConceptNet knowledge graph to enrich its representation. The authors reached this goal by using a retrofitting procedure to adjust the word embedding matrix by using their knowledge graph. Thanks to this procedure, ConceptNet is also intrinsically multilingual since this method pushes the model to use the same embedding space both for English words and words in different languages (since they share the same meaning). This work uses the Numberbatch 19.08 English-only model [1] which contains 516782 embeddings with a dimension of 300.

3.2 ELMo

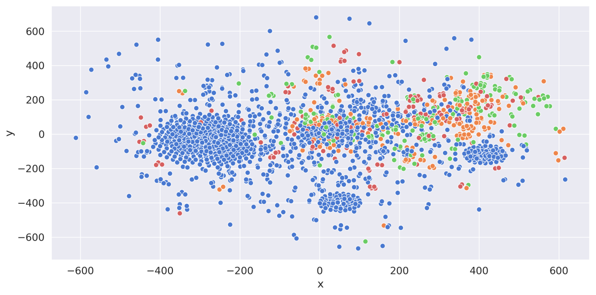
ELMo (Peters et al., 2018) is a deep contextualized word representation which captures both syntactic, semantic and polysemy characteristics of a given word. It uses a deep shallow bidirectional language model (biLM) trained on a large corpus. The ELMo method differs from the classical context-free approaches because each word embedding is obtained by looking at the entire input sentence. The generated representation are deep because they are a function of all the layers of the biLM. Moreover, each layer models certain features of the given word. The original work states that the higher-level LSTM states capture context-dependent aspects of the word meaning, while lower-level states capture aspects of the syntax. We used the pre-trained small ELMo model [2] with the embedding dimension of 1024. In this work, we collected all the ELMo layer representation (3) of each word of each sentence in the training dataset. We then devised two strategies to combine them to obtain the final embeddings: the first strategy produces a linear combination given an equal weighting over the layers and the second strategy learns directly the weights of the linear combination during the training of the LSTM concept tagger such to fine-tune the embeddings to our problem.

3.3 BERT

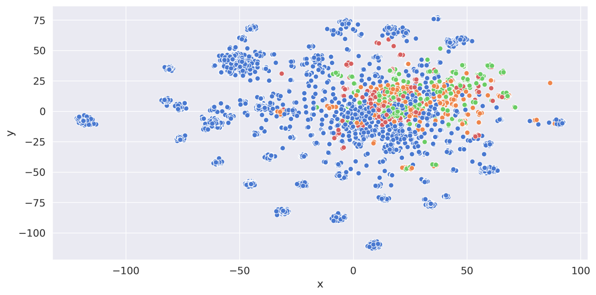
BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) is a relatively new language representation model released in 2019. It pre-trains deep bidirectional representations from unlabeled texts by exploiting both the left and right context. The model is a multi-layer bidirectional Transformer encoder. The authors state that this new representation solves the previous limitations of pre-trained models (e.g., ELMo). They argue that the standard LMs are unidirectional, thus limiting the kind of architectures which can be used during training. They also state that other pre-trained embeddings like ELMo are just shallow bidirectional since they use several separate models to capture the left and right context. There are two steps in the BERT framework: pre-training and fine-tuning. During the first phase, the model is trained on unla-

[1] https://github.com/commonsense/conceptnet-numberbatch

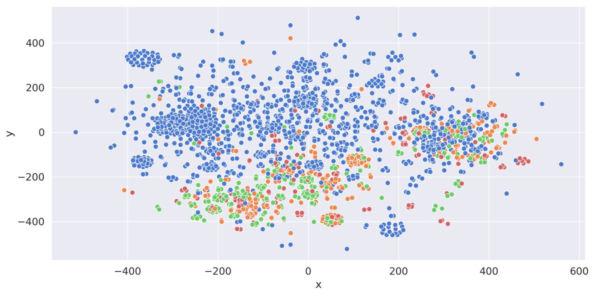
[2] https://allennlp.org/elmo



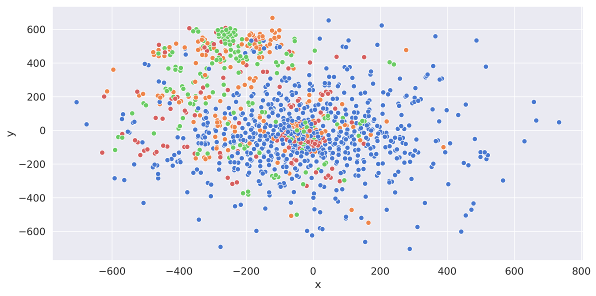
(a) Default Embeddings



(b) Elmo Embeddings



(c) Bert Embeddings



(d) ConceptNet Embeddings

Figure 1: These plots shows the embeddings representation of the four major concepts of the train dataset. Namely, movie.name (blue), actor.name (orange), producer.name (red) and director.name (green). We can clearly see how some embeddings enable much nicer clusters for some concepts, while others seems not to be able to do it.

beled data over two separate unsupervised training tasks: Masked LM and Next Sentence Prediction (NSP). The Masked LM task consists of predicting some words of the given input phrase which are masked beforehand. This would enable to train a real deep bidirectional representation since bidirectional conditioning on standard LM is not possible. The Next Sentence Prediction (NSP) consists of prediction relationships between sentences which are usually not captured by LM. During the fine-tuning phase, the model is initialized with the pre-trained parameters obtained in the previous stage and it is fine-tuned on the desired task. In our work, we used a pre-trained BERT model[3] to get the word embeddings directly without doing the fine-tuning phase. More specifically, the BERT model used was pre-trained on the BookCorpus (Zhu et al., 2015) and English Wikipedia. The embedding dimension was 768.

4 Data Analysis

We evaluated our solutions by using the NL2SparQL4NLU dataset (Chen et al., 2014; Gobbi et al., 2018). It corresponds to the MOVIE dataset of the original paper. It includes sentences taken from the movie domain (e.g., "*trailer for star wars a new hope*"). The dataset contains 3338 and 1084 sentences for the train and test set respectively. The dictionary size (# of unique tokens) is of 1728 (train) and 1039 (test). The average sentence length is 6.42 (train) and 6.52 (test) with an OOV rate between the datasets of 24%. We have a total of 23 concepts (with a total of 43 concepts in IOB format). Each word was also POS tagged and there are 50 different tags between the training and test set (taken from the Tree Tagger POS list[4]). We generated a new dataset by analyzing each sentence using a Named Entity Recognition (NER) tool, spaCy (**?**). It found 398 (1,86%) and 169 (2,37%) entities inside the training and test set respectively. These entities were of 9 different types[5].

5 Embedding Analysis

We evaluated also the various embeddings using the embedding coverage (namely, how many words in the dataset dictionary can be found in the embedding matrix) and we compared it with the embedding used in the original work. With the default embeddings (w2v_trimmed), the coverage is 96.18% (66 words not recognized) while with

[3]https://pypi.org/project/bert-embedding/

[4]https://courses.washington.edu/hypertxt/csar-v02/penntable.html

[5]https://spacy.io/api/annotation#named-entities

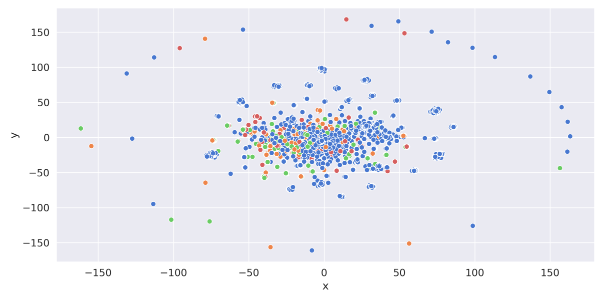
Model	Hidden	Epochs	Batch	Lr	Drop rate	Emb	Min F_1 / Mean F_1 / Best F_1			
lstm	200	15	20	0.001	0.7	none	81.88	83.17	84.35	
							82.85	**83.50**	84.37	(NER+POS+CHAR)
lstm	200	15	20	0.001	0.7	conceptnet	81.58	82.49	83.75	
							82.44	**83.03**	83.63	(NER+POS+CHAR)
lstm	200	30	20	0.001	0.7	bert	77.06	79.10	81.39	
							80.80	**82.29**	83.30	(NER+POS+CHAR)
lstm	200	30	20	0.001	0.7	elmo	81.77	82.65	83.48	
							83.14	**84.29**	85.56	(NER+POS+CHAR)
lstm	200	30	20	0.001	0.7	elmo (comb)	75.19	82.29	85.14	
							83.17	**84.71**	85.78	(NER+POS+CHAR)
lstmcrf	200	10	1	0.001	0.7	none	84.85	**85.70**	86.83	
							84.53	85.53	86.49	(NER+POS+CHAR)
lstmcrf	200	10	1	0.001	0.7	conceptnet	85.46	**85.99**	86.52	
							85.37	85.37	85.37	(NER+POS+CHAR)
lstmcrf	200	20	1	0.001	0.7	bert	82.33	**83.73**	84.95	
							82.01	83.61	85.67	(NER+POS+CHAR)
lstmcrf	200	20	1	0.001	0.7	elmo	84.50	**85.53**	86.39	
							84.37	85.40	86.88	(NER+POS+CHAR)
lstmcrf	200	10	1	0.001	0.7	elmo (comb)	84.34	85.72	86.56	
							84.80	**85.75**	86.84	(NER+POS+CHAR)

Table 1: Evaluation results of the various models. Each model is bidirectional. The performance were obtained by recording the results for 25 runs by varying the inizialization of the models and by keeping their hyperparameters fixed. The first line show the results without any improvements. The second line shows the results by adding the NER and POS tagging features. The third line shows the results by adding the character convolution and the NER and POS tagging features. The **bold** values indicates the best result for that particular embedding.

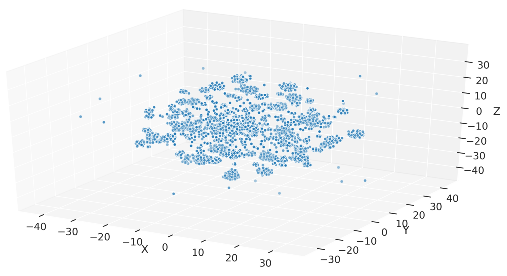
ConceptNet the coverage is 91.90% (140 words not recognized). For BERT and ELMo models we did a visual analysis instead. More specifically, we used T-sne (van der Maaten and Hinton, 2008) to reduce the dimensionality of the BER/ELMo embeddings down to 2-dimensional data points and then we plotted the results for the four most represented concepts in the dataset. See Figure X for the results. It is possible to notice how the default embeddings, ELMo and BERT produce nice clusters for the "movie.name" category. The other categories are slightly mixed together which is understandable since person names may end to have the same embedding. ConceptNet does not produce significant clusters (apart from a small one composed by "*director.name*" elements.

6 Experiments

We evaluated the models by doing a hyperparameter search by starting from the values used in the original work. We tried to find the hyperparameter combination which maximizes the F1 score over the test dataset. The final results can be seen in Table X. When using a classical LSTM model, it is possible to see how combining the character representation and the POS/NER additional features lead to results which are better than the default baseline. This happens for all the embeddings considered. Moreover, it seems that the ELMo embeddings provide better performances than the other competitors. The BERT embeddings give instead poor results (more than 3% difference between the F1 mean baseline in some cases). When using the LSTM-CRF we obtained slightly different results. The LSTM-CRF generally outperforms the simple LSTM model, but here the combination CHAR+NER+POS gives good results only when using the ELMo embedders. By employing BERT or ConceptNet we obtain a better F1 score without using those features (the difference is minimal). Generally, we also noticed that by making the model learn how to combine the ELMo weightings (elmo-comb) to generate the final embeddings is beneficial for the final performances. Apparently, by fine-tuning the ELMo weightings, it is possible to generate better "concept clusters" in the embedding space. Then, the network can effectively learn the hyperplanes which separate each clusters thus leading to less misstagging. This can be seen from Figure X in which the we show the new 3-dimensional cluster of the "*movie.name*" concept. By using LSTM-CRF and ELMo (learned) we were able to get the best max F1 score of 87.18. Ultimately, the best



(a) Learned ELMo Embeddings 2D



(b) Learned Elmo Embeddings 3D (*movie.title* only).

Figure 2: These plots shows the embeddings representation obtained by learning the ELMo linear combination. Namely, movie.name (blue), actor.name (orange), producer.name (red) and director.name (green). The new ELMo embeddings seems to not have improved the previous situation. However, in higher dimension we notice how the enabled better clusterizations in a 3-dimensional plane.

combination overall with a mean F1 value of 85.99 is obtained by using the ConceptNet embeddings without additional features and the LSTM-CRF model. All these results also surpass the previous baselines obtained with the WFST (83.73).

7 Conclusion

The experiments showed us how the usage of the correct embeddings is of utmost importance for obtaining state-of-the-art performances on concept tagging tasks. BERT and ELMo pre-trained models provide new embeddings obtained by looking at the context of the phrase. However, they cannot be used directly out-of-the-box but they need to be fine-tuned on the task at hand to be effective. This can be clearly seen by looking at the difference between the ELMo embeddings created by mere averaging of the layers or by learning the weights of the linear combination. This is also the reason why BERT underperformed in some cases. We used an implementation which extracts directly the embeddings without going through the fine-tuning phase described in the original paper. Therefore, the BERT embeddings are not “trained” to represent the current concept tagging task. However, the fine-tuning must be done properly to be useful since otherwise we just increase the variance of the results (e.g., the ELMo model showed a higher standard deviation with respect to the other competitors when learning the weights for combining the layers). We showed also the impact of adding more features to the original model, more specifically the POS tag and the NER entities. However, we did not see any important improvement. We thought that having extra information like the NER tags could have improved the performances of the models but it was not the case. This process had several issues. For instance, one major problem we found with this method is related to the sparsity of the NER tagging. Over 21000 tokens, only 400 were tagged with an entity definition (just the 1.9%). This adds just too little representational power to the original model and therefore it does not produce any huge gain in performances.

References

Yun-Nung Chen, Dilek Z. Hakkani-Tür, and Gökhan Tür. 2014. Deriving local relational surface forms from dependency-based entity embeddings for unsupervised spoken language understanding. *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 242–247.

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

Jacopo Gobbi, Evgeny Stepanov, and Giuseppe Riccardi. 2018. Concept tagging for natural language understanding: Two decadelong algorithm development.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):17351780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *LREC*.

Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing data using t-sne.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

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. In *Proc. of NAACL*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. pages 4444–4451.

Kaisheng Yao, Baolin Peng, Geoffrey Zweig, Dong Yu, Xiaolong(Shiao-Long) Li, and Feng Gao. 2014. Recurrent conditional random field for language understanding. In *ICASSP 2014*. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP).

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*.