Named Entity Recognition

Jakob Božič
University of Ljubljana
Faculty of Computer and Information Science
Večna pot 113, SI-1000 Ljubljana
jakob.bozic@gmail.com

Abstract

Named entity recognition (NER) is used to extract the named entities from the text into the pre-defined categories. Most commonly these categories are at least names, locations, organizations and other, however in various branches of NER different, branch-specific categories are added. State-of-the-art (SOTA) results are, similarly as in other fields of Machine Learning, given by deep (recurrent) neural networks, which have in last years completed the transition from hand-crafted to deep features. In our work we first give a brief overview of recent development of the field with more focus on Slovene language and then use of the current SOTA (RNN) approaches on ssj500k (Krek et al., 2019) and SentiCoref (Žitnik, 2019) datasets. We also investigate how well the knowledge is transferred between the both datasets

1 Related work

In our work we focus on more recent, deep learning based approaches, using Recurrent Neural Networks (RNN). A brief overview of history of NER is given in (Yadav and Bethard, 2019), which also contains results of different NER approaches for four different languages and shows that deep learning approaches outperform traditional methods across the board. Extensive study of NER for slovene language has been presented in (Štajner et al., 2013). Authors have used supervised learning and Conditional Random Fields to propose then state-of-the-art system. They evaluated how introduction of lexicons, part-of-speech tags and conjuctions of neighbouring properties effect the system's capability to correctly classify named entities. In their work they showed that using part-of-speech tags can further improve model's capabilites. When enabling all of their improvements, their model achieves 74% precision and 72% recall, however this drops to 63% precision and 59% recall, when using basic model with no part-of-speech tags and no lexicons.

Deep Recurrent neural networks have played the most significant role in recent advances in natural language processing. One of the key ingredients which has been around for quite a while is Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997). Only recently a similar, but simpler gating mechanism called Gated recurrent unit (GRU) (Cho et al., 2014) has started gaining popularity.

ELMO embeddings (Peters et al., 2018) are deep contextualized word representations which model word usage and how the usage varies in different contexts. They are easily adaptable to existing NLP solutions and usually introduce noticeable performance improvements.

Currently, SOTA results on many NLP tasks are achieved by approaches that incorporate Bidirectional encoder representations from transformers (BERT) (Devlin et al., 2018).

2 Data

We train and evaluate our approach on both ssj500k and SentiCoref datasets.

2.1 ssj500k

ssj500k (Krek et al., 2019) contains approximately 500.000 words, manually annotated on different levels. The data comes from FidaPLUS corpus, which contains very diverse set of text from various daily newspapers, books, parliament speeches, adverts, etc. Roughly half of the dataset is also annotated with named entities, on which we focus our attention. There are in total 4.398 named entities in total, separated in five classes: loc(ation), org(anization), per(son), misc(ellaneous) and der(ivative)-per(son), however we decided to merge person and derivative-

person in per, since derivative-person contained very few samples. Due to only half of the corpus being labeled for named entities we only used those sentences with at least one named entity present, since we could not otherwise tell whether the sentence was labeled for named entities or not. In order to avoid or at least minimise the problems associated with class-imbalance (Buda et al., 2017) we used over-sampling for all categories. In particular, we first create one collection of sentences for each tag, a sentence is put in a collection if it contains at least one named entity for that tag. The last 20% of each collection is put in the test set. For each tag, we then over-sample the remaining part, by repeating each sentence as many times, as the quotient of the length of the collection for that and the length of collection for most common tag. This ensures that all tags are represented approximately equally. We could presumably further improve the data by introducing sentences without any named entities, however we were not able to determine which sentences were labeled and which not.

2.2 SentiCoref

SentiCoref (Žitnik, 2019) dataset contains 31.419 named entities, separated in three categories, there are 15.285 org(naizations), 8.606 per(sons) and 7.528 loc(ations). The data consists of news articles on various Slovenian online news portals. Unlikely in the ssj500k dataset, there is no misc(ellaneous) category. Again, we used over-sampling to avoid or minimise class-imbalance related drawbacks, the process is described above. We again used 80% of data for training and 20% for testing. The dataset is split into documents and not into sentences, which meant that we had to split it ourselves. Since we could not come up with efficient way to take advantage of existing text segmentation tools, we decided for very simple and not very good split. We removed all non alphanumeric characters except for dot, question mark and exclamation mark, which we used as splitting points. We then discarded sentences which were only single word, since those were mainly there because of the wrong split. While this is definitely not a optimal sentence splitting approach, we were still able to achieve very good results on both datasets, which shows that our method is indifferent to a minor number of wrongly split sentences.

3 Methodology

Our model consists of three main parts, (i) an embedder, (ii) an encoder, and (iii) a projector. Embedder transforms characters and words to embeddings and then encoder transforms them in representation which projector uses to make final predictions. Different embedders and different encoders are used and compared against each other. We use either pretrained ELMO embeddings (Ulčar and Robnik-Šikonja, 2019) or we train an embedder with embedding dimension of 64. For encoder we use either two layer LSTM (Hochreiter and Schmidhuber, 1997) or two layer GRU (Cho et al., 2014) network, with hidden dimension of 64, both in standard or bi-directional configuration. We tried using greater dimensions, however we observed either overfitting or the training was simply too long to perform all the experiments we wanted to. Projector was the only constant component, a simple fully-connected layer which maps outputs from the encoder in final tags.

Together we have three variable parts, for eight experiments on each dataset. In order to quantify, how well the knowledge is transferred from one dataset to the other, we evaluated the best performing model from each dataset on the other. Some minor adaptations were necessary however, since the datasets do not have the same labels.

4 Results

We extensively evaluated previously described variations of main components of our model. Results for both datasets are given in Table 1. The most noticeable conclusion we can draw is that quality of word representations plays a key role in the model's performance, since the performance boosts introduced by ELMO are overwhelming. Besides for the overall better word representations, one additional explanation why ELMO brings such significant performance boost is, that it effectively introduces much more training data, since it was pretrained on a very large corpus. We did not observe significant effects on performance when using either LSTM or GRUs, regardless whether we use them in bi-directional configuration or not. We report the average F-measure, which is calculated as simple mean of per-category F-measures, not the weighted average. We decided for that, since we believe that normal average better represents the model's ca-

Dataset	Embedder	Encoder	Bi-directional	Avg. F-measure	Per F-measure	Loc F-measure	Org- F-measure	Misc F-measure
ssj500k	Classical	LSTM	✓	48.30	58.02	60.14	52.57	22.48
	Classical	LSTM		51.26	60.67	60.91	54.38	29.07
	Classical	GRU	✓	49.37	55.40	63.49	56.70	21.90
	Classical	GRU		49.22	61.74	58.84	49.15	27.16
	ELMO	LSTM	✓	86.43	95.63	93.09	83.40	73.62
	ELMO	LSTM		85.74	96.02	92.14	82.66	72.15
	ELMO	GRU	✓	**87.74**	96.02	93.14	85.65	76.14
	ELMO	GRU		86.37	95.78	91.60	84.59	73.52
SentiCoref	Classical	LSTM	✓	75.11	81.15	66.16	78.91	/
	Classical	LSTM		76.07	83.72	66.59	77.89	/
	Classical	GRU	✓	75.56	82.26	66.20	78.22	/
	Classical	GRU		75.30	83.24	65.00	77.46	/
	ELMO	LSTM	✓	90.35	97.41	84.10	89.53	/
	ELMO	LSTM		90.53	97.21	84.28	90.11	/
	ELMO	GRU	✓	90.20	97.06	84.04	89.48	/
	ELMO	GRU		**90.59**	97.30	84.76	89.70	/

Table 1: Evaluation of different combinations of embedders and encoders on ssj500k and SentiCoref datasets. Highlighted in bold are the best average results on both datasets.

pabilities, instead of the properties of the data on which it was evaluated.

Hyperparameters In all of the experiments we used identical hyperparameters, we trained the model for 50 epochs, with early stopping if there were no improvements for 15 epochs, using Adam (Kingma and Ba, 2015) optimizer, with learning rate of 0.001, and weight decay 10^{-4}. Batch size was 4, since we were limited by the memory of our GPU. Both LSTM and GRUs had 2 layers, with hidden dimension of 64. When we used the classical embedder, the embedding dimension was also 64.

ssj500k comparison On the ssj500k dataset, our model performs significantly better as the one from (Štajner et al., 2013), however since the authors did not report how the train and test data was formed and also how they calculated the final average F1-measure, comparison may not be completely accurate. Their best reported model achieved 72% average F1 measure, whereas ours achieves 88% average F1 measure, which represents more than two-fold reduction of error. It is also worth noting, that our approach did not use any additional information, such as part-of-speech tags or dictionaries. In the same configuration, their average F1 measure was 61%.

4.1 Knowledge transfer

In order to evaluate, how general and data independent our approach is, we cross-evaluated the best models from each dataset on the other dataset. In particular, we evaluated the model which used ELMO as embedder with bi-directional GRUs and was trained on ssj500k dataset (7th row in Table 1), on the SentiCoref dataset, and the model which used ELMO as embedder with classical GRUs and was trained on ssj500k dataset (last row in Table 1) on the SentiCoref dataset. Since SentiCoref does not have misc(ellaneous) label, we marked all the words which were labeled as misc(ellaneous) in ssj500k dataset as other (not a named entity). For both datasets, the evaluation was performed on full dataset, not on the subset that was used to evaluate models when training on data from the same dataset.

Dataset	Avg. F-measure	Per	Org	Loc
ssj500k ->SentiCoref	86.10 (64.57)	96.68	84.52	77.09
SentiCoref ->ssj500k	82.78	93.51	73.10	81.74

Table 2: Results of cross-evaluation of best-performing models. When evaluating model that was trained on ssj500k on SentiCoref (first row), the model predicts some words as misc(ellaneous), however those are (at least considered) all wrong, since there is not misc(ellaneous) category in SentiCoref dataset. This means that the model gets 0.00 F-score for that label, and the value in parenthesis in the first row represents the average F-measure we get if we also consider the absent category when calculating average.

The results of cross-dataset evaluation are given in Table 4.1. We can see that the performance of both models drops, which was expected, since the datasets contain data from different kinds of text. Rather unexpectedly, the model trained on ssj500k dataset works better on SentiCoref, than the model trained on SentiCoref works on ssj500k.

We would expect that to not be the case, since the SentiCoref datasets is bigger. One possible explanation is, that the more diversity in texts in ssj500k translates in better overall performance and better knowledge transfer. SentiCoref contains only data from online news portals, which may not translate very well to e.g. literary works contained in ssj500k.

Example In Table 4.1 we can see how the best models from both datasets perform on a piece of news article from RTVSlo. While the example is not very hard, we can see that both models do not make a single mistake.

5 Discussion

We evaluated various combinations of embedders and encoders for a NER model and demonstrated that using an embedder pretrained on larger corpus brings significant performance improvements. Our best model trained on ssj500k dataset achieves more than two-fold reduction in error compared to the one from (Štajner et al., 2013). We also extensively evaluated our models on SentiCoref dataset, and demonstrated that findings from ssj500k dataset are not dataset-specific. We cross-evaluated some of our models, and shown that they are capable of performing well on the kind of data they did not necessarily see during the training. In future, we plan to incorporate BERT in our model and search for potentially better hyperparameter values.

References

Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. 2017. A systematic study of the class imbalance problem in convolutional neural networks. *CoRR*, abs/1710.05381.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Simon Krek, Kaja Dobrovoljc, Tomaž Erjavec, Sara Može, Nina Ledinek, Nanika Holz, Katja Zupan, Polona Gantar, Taja Kuzman, Jaka Čibej, Špela Arhar Holdt, Teja Kavčič, Iza Škrjanec, Dafne Marko, Lucija Jezeršek, and Anja Zajc. 2019. Training corpus ssj500k 2.2. Slovenian language resource repository CLARIN.SI.

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.

Matej Ulčar and Marko Robnik-Šikonja. 2019. High quality elmo embeddings for seven less-resourced languages.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models.

Slavko Žitnik. 2019. Slovene corpus for aspect-based sentiment analysis - SentiCoref 1.0. Slovenian language resource repository CLARIN.SI.

Tadej Štajner, Tomaž Erjavec, and Simon Krek. 2013. Named entity recognition in slovene text. *Slovenščina 2.0: empirical, applied and interdisciplinary research*, 1(2):58–81.

Model	Po	različnih	delih	Slovenije	zaznavamo	od	10	do	30	odstotkov
ssj500k	-	-	-	loc	-	-	-	-	-	-
SentiCoref	-	-	-	loc	-	-	-	-	-	-
Model	manj	rakavih	diagnoz	Kako	bo	to	vplivalo	na	dolgi	rok
ssj500k	-	-	-	-	-	-	-	-	-	-
SentiCoref	-	-	-	-	-	-	-	-	-	-
Model	ta	trenutek	še	ne	vemo	je	za	Televizijo	Slovenija	povedala
ssj500k	-	-	-	-	-	-	-	org	org	-
SentiCoref	-	-	-	-	-	-	-	org	org	-
Model	Vesna	Zadnjik	vodja	epidemiologije	in	registra	raka	na	Onkološkem	inštitutu
ssj500k	per	per	-	-	-	-	-	-	org	org
SentiCoref	per	per	-	-	-	-	-	-	org	org

Table 3: Example of evaluation of best models from both datasets on a piece of news article which was not included in either training set.