Character-Based Neural POS Tagger

Paweł Rychlikowski*†, Michał Zapotoczny*†, Jan Chorowski*

*Institute of Computer Science
University of Wroclaw
Joliot-Curie 15, 50-383 Wrocław, Poland

†Neurosoft Sp. z o.o.
Życzliwa 8, 53-030 Wrocław, Poland

Abstract

We present our entry to the PolEval 2017 tagging task: a part of speech tagger implemented as a single deep neural network that reads orthographic representations of words and generates probability distribution for every POS category. With this kind of output we are able not only to rate a given POS tag (by e.g. morphological analyzer) but also to generate tags for out-of-vocabulary words.

1. Introduction

Part of speech tagging (POS-tagging) is a task in which we assign to every word in the text a special tag describing its syntactic category. Tags contain the information about the part of speech (noun, verb, adjective, adverb, pronoun, ...), as well as other (mostly language dependent) parameters like gender, person, case, or number.

Even if we have full information about all words used in an analyzed text, tagging is not trivial because words are ambiguous (have many syntactic interpretations). Moreover, the prevalence of proper names, jargon words, neologism and similar makes it is impossible to create a complete dictionary containing all words that belong to a language. Therefore a robust POS-tagger must be able to deduce possible interpretation of out-of-vocabulary words from their morphology.

2. Background and related work

Historically first POS-taggers were rule based (for instance (Greene and Rubin, 1971)). Since 1980's one can observe the expansion of statistical taggers (most often based on Hidden Markov Models) (Church, 1988),(Kupiec, 1992) and later using Maximum Entropy Markov Models (Ratnaparkhi, 1996) or Conditional Random Fields (Lafferty et al., 2001).

Nowadays, as in many other NLP tasks, artificial neural networks are used for tagging. One of the first successful attempts of using neural networks in this task was (Schmid, 1994) (with simple multilayer perceptron). Recently, more modern architectures were used, see for example (Wang et al., 2015), when the authors used bi-directional recurrent neural network with LSTM gates.

First taggers were designed for English. Although in principle statistical taggers are language agnostic, one can expect some improvements by exploiting the properties of the target language. This is particularly important for languages that differ from English in an important trait, such as morphology present in Slavic languages, Hungarian, or Turkish.

There were several taggers designed specifically for Polish language, for instance trigram HMM tagger (Dębowski, 2004), the adaptation of Brill tagger designed for languages with rich morphology (Pantera)(Acedanski, 2010), or tiered CRF tagger (Radziszewski, 2013). In (Kobyliński,) the voting scheme, joining the results of various taggers is presented.

The tagger presented in this paper uses a deep, recurrent neural network, similar to the one from (Zapotoczny et al., 2017). It was tested on polish language, however it could be easily adapted to other languages.

This work is our contribution to the PolEval 2017 shared task, organized by The Linguistic Engineering Group (Institute of Computer Science, Polish Academy of Sciences) and Sages. The task is divided into 3 subtasks. In the first two, the authors provided a gold segmentation of the sentences and participants were expected to perform morphosyntactic disambiguation by choosing the correct interpretation (POS tag and lemma) of each word among a few possible interpretations given by the authors (or in case of unknown word – guess it). The data for task 3 was in a raw format and systems were supposed to apply a full NLP pipeline to obtain correct POS tags and lemmas.

3. Model

The network architecture consists of three main parts: reader, tagger and POS-tag predictor (see Figure 1). The reader subnetwork is evaluated on each individual word in a sentence, and using convolutions on their orthographic representation produces words embeddings. Next, the tagger subnetwork implemented as bidirectional RNN equips each word with a context of whole sentence. Finally, the POS-tag predictor part computes part-of-speech value for each category (we use the NKJP tagset, see Table 1). In the following paragraphs we will describe all parts of our model in detail.

3.1. Reader

As stated before, the reader subnetwork is run on each word producing its embedding. This architecture is based on (Kim et al., 2015). Each word w is represented by sequence of its characters plus a special beginning-of-word <bow> and end-of-word <eow> tokens. First, we find the

category	POS values
subst	number, case, gender
num	number, case, gender, accommodability
adj	number, case, gender, degree
pact	number, case, gender, aspect, negation
praet	number, gender, aspect, *agglutination*
aglt	number, person, aspect, vocalicity

Table 1: Sample categories from NKJP tagset. A category denoted in *italics* means that it is optional

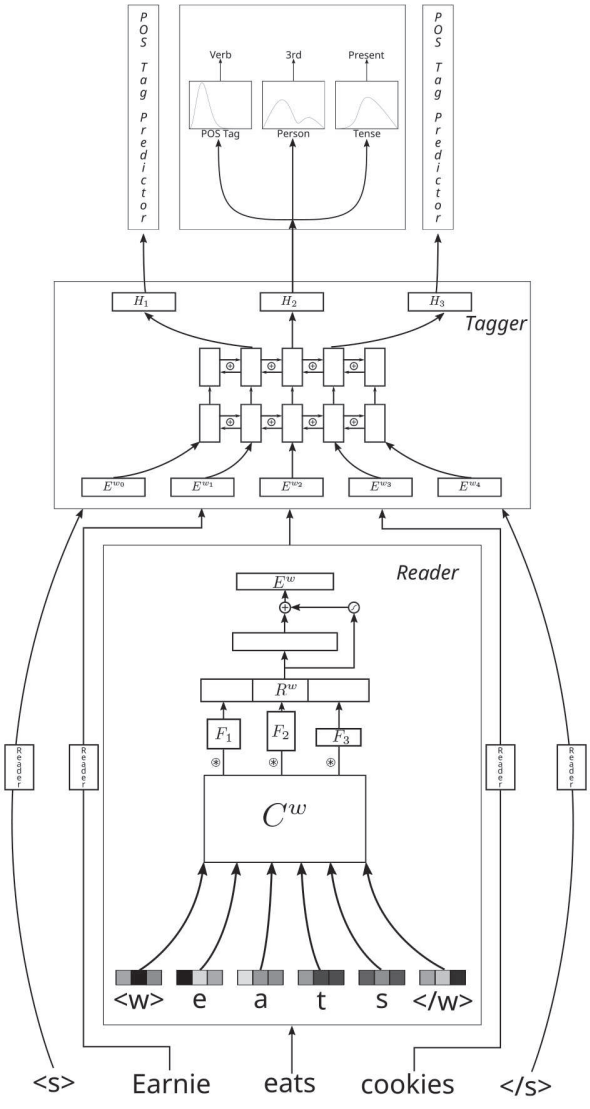

Figure 1: Schematic view of the model architecture. The POS tag predictor predicts one value for every possible category.

low-dimensional characters embeddings which are concatenated to form a matrix C^w. Then we run 1D convolutional filters on C^w which are then reduced to a vector of filter responses computed as:

$$R_i^w = \max(C^w * F^i) \quad (1)$$

Where F^i is i-th filter. The purpose of the convolutions is to react to specific parts of words, which in morphologically rich languages such as Polish can depict its grammatical role. Please note that due to the use of `bow` and `eow` tokens the model can distinguish between prefixes, suffixes and infixes. Finally we transform filter responses R^w with a simple multi-layer perceptron obtaining the final word embedding $E^w = \text{MLP}(R^w)$.

3.2. Tagger

Having obtained the word embeddings E^w we can proceed to equip each word with the context of the whole sentence. To do this we use multi-layer bidirectional GRU layer (Cho et al., 2014). We combine the backward and forward passes by adding them.

3.3. POS tag predictor

Finally we compute the probability distribution over all possible values for each POS category (part of speech, case, gender, and so on) by using single affine layer with softmax activation. Each category is computed separately.

3.4. Regularization and Training

Regularization is an essential part of neural network training. It improves generalization and prevents overfitting. One of the most popular regularization technique is called Dropout (Srivastava et al., 2014). Using it, we randomly drop part of the connections from a certain network layer during training. In our case dropout is applied to the *reader* output and between the BiRNN layers of the *tagger*.

The models are trained using Adadelta (Zeiler, 2012) learning rule, with weight decay and adaptive gradient clipping (Chorowski et al., 2014). All experiments are early stopped on validation set accuracy. The chosen parameters are identital as in our previous publication (Zapotoczny et al., 2017) and were obtained using the Spearmint system (Snoek et al., 2012). The parameters are as follows: The *reader* embeds each character into 15 dimensions, and contains 1050 filters (50·k filters of length k for k = 1, 2,. . . , 6) whose outputs are projected into 512 dimensions transformed by a 3 equally sized layers of feedforward neural network with ReLU activation. The *tagger* contains 2 BiRNN layers of GRU units with 548 hidden states for both forward and backward passes which are later aggregated using addition. Therefore the hidden states of the tagger are also 548-dimensional. The *POS tag predictor* consists of a single affine transformation followed by a softmax predictor for each POS category. We apply 20% dropout to the *reader* output and 70% between the BiRNN layers of the *tagger*. The weight decay is 0.95.

4. Decoding and Lemmatization

In the previous section we have described how to obtain a probability distribution over POS tags based only on text characters. Now we will outline how to convert these data to get the final POS tags and lemmatization. We can distinguish two situations:

- the word is known to some dictionary (If the authors gave some interpretations - we use only them, otherwise we use output of Morfeusz(Woliński, 2006) morphological analyzer)
- the word is not known.

Subtask	Split	Known words		Unknown words		Overall	
		POS	lemma	POS	lemma	POS	lemma
subtask a,b	dev	95.4	98.9	68.2	78.3	94.3	98.1
	test	94.2	97.4	64.9	84.6	93.6	97.1
subtask c	dev	94.9	99.1	78.8	86.5	94.1	98.4
	test	-	-	-	-	91.6	97.0

Table 2: POS tagger accuracy for dev and test data. For subtask c the word is considered unknown if there are no candidates returned by Morfeusz dictionary.

System	Known words		Unknown words		Overall	
	POS	lemma	POS	lemma	POS	lemma
OpenNLP	88.1	-	63.6	-	87.0	-
WCRFT	93.5	97.8	59.7	73.9	92.1	96.8

Table 3: Baseline systems trained on the same dev split as in Table 2

System	Subtask a,b Known words		Subtask a,b Unknown words		Subtask c Overall	
	POS	lemma	POS	lemma	POS	lemma
AvgPer_Forced	91.4104	-	**67.0841**	-	-	-
AvgPer_RAW	89.4776	-	62.4329	-	-	-
KRNNT_AB	94.4888	**98.194**	61.1807	80.8587	-	-
KRNNT_ABv	94.3022	97.959	62.0751	79.7853	-	-
KRNNT_AB_morf1	93.6716	97.7724	59.7496	80.1431	-	-
KRNNT_raw	-	-	-	-	92.6242	96.8581
KRNNT_voted	-	-	-	-	**92.9833**	96.9089
MorphoDiTaPL	-	-	-	-	89.6746	95.7769
NeuroParser	94.2090	97.3731	64.9374	**84.6154**	91.5865	**97.0032**
Toygger	**95.2425**	-	65.4741	-	-	-

Table 4: Comparison of systems submitted to Poleval

In the first case we are given a list of available POS tags and corresponding lemmas. We evaluate all POS tag interpretations and choose the most probable one. To score each lemma we sum all POS tag probabilities corresponding to given base word and choose the most probable.

In the case of the word being unknown to the dictionary, we return the most probable POS tag (among all possible combinations) and set the lemma to be the same as the input word.

5. Evaluation

For evaluation we have randomly split the PolEval data into training and dev subset (90% and 10%, respectively). For subtasks a and b we trained the network on single sentences, whereas for subtask c our network was trained on whole paragraphs (without splitting into individual sentences). With this trick we could ignore problems with sentence boundaries prediction. The results are shown in Table 2. During development phase we have compared our system to OpenNLP and WCRFT (Radziszewski, 2013) taggers, both trained on the same data as our solution. This baseline solution is shown in Table 3. The speed of our implementation is about 1600 words/s on Nvidia TITAN X GPU.

In Table 4 we show the final results of PolEval task. The best results are shown in **bold**.

In Figure2 one can see the result of our tagger when ap-

Brzdęśniało **praet:sg:n**:perf już **qub** *ślimonne* **adj:sg:nom:n:pos** *prztowie* **subst:sg:nom:n**
Wyrło **praet:sg:n:perf** i **conj** *warło* **praet:sg:n**:imperf się **qub** w **prep**:acc:**nwok**
gulbieży **subst**:pl:acc:**m3**
Zmimszałe **adj:pl**:acc:m3:**pos** *ćwiły* **praet:pl:f:imperf** *borogowie* **subst:pl:nom**:m1
I **conj** *rcie* **subst:pl:nom**:n *grdypały* **praet:pl:f:imperf** z **prep:gen:nwok** *mrzerzy* **subst:sg:gen:f**

Figure 2: Famous Jabberwocky poem written by Lewis Carroll, translated by Stanisław Barańczak. The words in italic are neologisms, parts of the tags in bold are correct.

plied to the first stanza of Lewis Carrol's nonsense poem Jabberwocky. Tags are generally chosen correctly, with some minor mistakes. For instance tagger had a problem with deciding which aspect (perfect or imperfect) should be assigned to verb, but in this text it is a very hard task, even for human (at least for the authors of this paper). Moreover there are two words (*zmimszałe* and *borogowie*) whith tags which are acceptable, but not in this context.

6. Conclusions and Future Work

Although the results of our tagger are quite promising, there are many directions of possible improvements. First, we can combine character level word embeddings of out

tagger (where words are sequences of letters) with standard word embeddings (where the word is atomic). This standard embeddings can be computed by the tagger, or precomputed using word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014) on a separate corpus. Second, the network can use the information from dictionary (or the information of the possible tag combinations) in earlier stages. Moreover, the trivial lemmatization procedure for OOV words can be replaced by a more sophisticated one, which could involve using the deduced tag and possible language-dependent word transformations. We also plan to test this architecture on other languages.

7. Acknowledgments

The experiments used Theano (Bergstra et al., 2010), Blocks and Fuel (van Merriënboer et al., 2015) libraries. The authors would like to acknowledge the support of the Sonata grant (National Science Center (Poland) 8 2014/15/D/ST6/04402) for computing support.

8. References

Acedanski, Szymon, 2010. A morphosyntactic brill tagger for inflectional languages. In *Advances in Natural Language Processing, 7th International Conference on NLP, IceTAL 2010, Reykjavik, Iceland, August 16-18, 2010*.

Bergstra, James et al., 2010. Theano: a CPU and GPU math expression compiler. In *Proc. SciPy*.

Cho, Kyunghyun, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Chorowski, Jan, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 2014. End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results. *arXiv:1412.1602 [cs, stat]*. 00000 arXiv: 1412.1602.

Church, Kenneth Ward, 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, ANLC '88. Stroudsburg, PA, USA: Association for Computational Linguistics.

Debowski, Lukasz, 2004. Trigram morphosyntactic tagger for polish. In *Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'04 Conference held in Zakopane, Poland, May 17-20, 2004*.

Greene, Barbara B. and Gerald M. Rubin, 1971. Automatic grammatical tagging of English. Department of Linguistics, Brown University, Providence, Rhode Island.

Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M. Rush, 2015. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*. 00011 bibtex: kim_character_2015a.

Kobyliński, Łukasz. Improving the accuracy of Polish POS tagging by using voting ensembles.

Kupiec, Julian, 1992. Robust part-of-speech tagging using a hidden markov model. 6:225–242.

Lafferty, John D., 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. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*.

Pennington, Jeffrey, Richard Socher, and Christopher Manning, 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

Radziszewski, Adam, 2013. A tiered CRF tagger for polish. In *Intelligent Tools for Building a Scientific Information Platform - Advanced Architectures and Solutions*. pages 215–230.

Ratnaparkhi, Adwait, 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*.

Schmid, Helmut, 1994. Part-of-speech tagging with neural networks. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics.

Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams, 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*. 00414.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958. 00282.

van Merriënboer, Bart et al., 2015. Blocks and fuel: Frameworks for deep learning. *arXiv:1506.00619 [cs, stat]*.

Wang, Peilu, Yao Qian, Frank K. Soong, Lei He, and Hai Zhao, 2015. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *CoRR*, abs/1510.06168.

Woliński, Marcin, 2006. *Morfeusz — a Practical Tool for the Morphological Analysis of Polish*. Berlin, Heidelberg: Springer Berlin Heidelberg, pages 511–520.

Zapotoczny, Michal, Pawel Rychlikowski, and Jan Chorowski, 2017. On multilingual training of neural dependency parsers. In *Proceedings of the 20th International Conference Text, Speech and Dialogue*, TSD2017.

Zeiler, Matthew D., 2012. ADADELTA: An Adaptive Learning Rate Method. *arXiv:1212.5701 [cs]*. 00017 arXiv: 1212.5701.