# KernelTagger – a PoS Tagger for Very Small Amount of Training Data

Pavel Rychlý

Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
pary@fi.muni.cz

**Abstract.** The paper describes a new Part of speech (PoS) tagger which can learn a PoS tagging language model from very short annotated text with the use of much bigger non-annotated text. Only several sentences could be used for training to achieve much better accuracy than a baseline. The results cannot be compared to the results of state-of-the-art taggers but it could be used during the annotation process for a pre-annotation.

**Key words:** PoS tagging, morphological tagging, language model, Czech

## 1 Introduction

Part of speech (PoS) tagging is one of the most important tasks in corpus linguistics. PoS taggers assign a PoS tag for each word from an input. They usually learn a language model (or a set of rules) using manually annotated texts. Some taggers could also exploit an optional lexicon to help annotation of words which are not found in the manually annotated text.

One of the main feature of a text corpus in the field of natural language processing is its big size. Text corpora contains from millions to billions tokens. It is not a problem to create a corpus with tens of million tokens even for small languages [1]. On the other hand, manual annotation of such corpora is a big problem, it is a time consuming and expensive process. As a result, manually annotated corpora are rather small, most of them have less than one million tokens. For example, there are only five larger corpora in the Universal Dependencies[1] – the most comprehensive database of annotated corpora. Many smaller corpora have only a few hundred sentences annotated.

It is very hard to learn anything automatically from such small corpora because they contain only a few thousands of different words and most words have only one hit in the whole corpus. The performance (accuracy) of a PoS tagger trained on such corpus is close to the baseline. A bit better performance is achieved by taggers which use a lexicon or a morphological database containing all possible PoS tags for large amount of words. That could be helpful for languages where such lexicon or database is available. Otherwise, construction of them is more expensive than PoS annotation of a small corpus.

[1] http://universaldependencies.org/

Aleš Horák, Pavel Rychlý, Adam Rambousek (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2017, pp. 107–110, 2017. 

## 2 KernelTagger

KernelTagger is a new PoS tagger. It is optimize to exploit as much knowledge as possible from a large non-annotated corpus with the help of possibly very small PoS annotated corpus.

The main idea behind KernelTagger is computation of word similarity from the non-annotated corpus and using the kernel trick to derive a PoS tag for a given word from similarity to words with tag known from the small annotated corpus. There is no learning of any features from the annotated corpus, the tagger remembers PoS tags for each word from the annotated corpus. If the given word occurred in the training corpus the tagger outputs the most probable (most frequent) PoS tag for such word. The tag for unseen words is computed using a modification of a kernel perceptron on known words. The modification consists of using only top 5 most similar known words instead of all known words.

We use this modification because similarity of most (almost all) word pairs is near zero (they are not similar) and the exact number is mostly a noise. On the other hand, the similarity of similar words is quite reliable and could be used for computation.

### 2.1 Word Similarity Computation

In the early stages of development, we have used several different settings of a word embedding system [2] but the final version use very simple distributional similarity computed from small contexts. We use only one preceding and one following word for each keyword, we compute the logDice [3] salience score and assign the similarity of two words $w_a$ and $w_b$ using the following formula:

$$sim(w_a, w_b) = \frac{\sum_c min(D(w_a, c), D(w_b, c))}{\sum_c D(w_a, c) + \sum_c D(w_b, c)}$$

where $D(w_a, c)$ is the logDice score of word $w_a$ and context $c$. We use only contexts with positive logDice. The left and right contexts are handled separately: the same word before and after a keyword are treated as two different contexts.

### 2.2 PoS of Unseen Words

The PoS tag for a word which occurs in the training annotated corpus is the most frequent tag for that word (one is chosen randomly for several tags with same frequency).

The PoS tag for an unseen word is computed from PoS tags of most similar known words. First, we set a list of up to 5 most similar words. Then the tag for a word $w$ is defined by the following formula:

$$argmax_t \sum_x sim(w, x) P(x, t)$$

where $P(x, t)$ is the probability of the tag $t$ for the word $w$.

## 3 Evaluation

We have evaluated the tagger on the DESAM corpus [4], the Czech corpus of about one million manually annotated tokens. It contains lemma and morphological tag for each word. As Czech has quite complex morphology the DESAM tag-set is huge, it consists of 13 attributes with up to 7 different values. We have used only the main part of speech, that means only 12 different values, 11 for words (including one for numbers) and one for punctuation. The most frequent PoS tag is NOUN, it represents 30% of tokens.

We have used two setups for a non-annotated corpus:

1. The whole DESAM corpus. We choose this setup to demonstrate that even small (1 mil.) corpus could be useful for computing word similarities.
2. Part of czTenTen[1] corpus. Only a small part of the whole corpus was used to simulate low resource language. The used part contains 33 million tokens. There was a limit of 10 million word pairs during word similarity computation. That means only 10 million most similar word pairs are stored and used for evaluation. This limit caused that the size of the temporary data files was less than a half of the size from the DESAM setup.

The results are listed in Table 1. There are four test cases for each setup. They differ in the number of annotated tokens used for training (the first column). We can see that even 1000 tokens (representing several dozens of sentences) provides interesting accuracy.

## 4 Future Work

We would like to make more evaluation on the Czech corpus to measure the influence of the size of the non-annotated corpus. There are also questions on influence of several KernelTagger parameters which we have set according to just a few tests:

1. N for the top N most similar words (now: 5),
2. the size of context in computing similarity of words (now: 1),
3. the threshold of minimal logDice and minimal count of a context to be included in similarity (now: 0 and 2).

**Table 1.** Evaluation results: The accuracy of KernelTagger for different number of training tokens with annotation and different corpora for computing word similarities.

| train tokens | DESAM (1 mil.) | czTenTen (33 mil.) |
|---|---|---|
| 1,000 | 70.7 | 72.9 |
| 10,000 | 78.8 | 81.7 |
| 100,000 | 87.7 | 88.5 |
| 980,000 | 92.9 | 92.8 |

We are also going to add two modules to handle the most common errors in the KernelTagger annotation. First one for handling unknown words according to sub-strings. Second one for handling ambiguous words depending on context. That could increase the usability of KernelTagger for languages with weak morphology and high ambiguity of PoS tags for individual word forms.

## 5 Conclusion

In this paper, we have presented the new PoS tagger *KernelTagger*. It trains a PoS model from (small) annotated text and (big) non-annotated text. The main advantage of the tagger is its ability to provide competitive results for very small annotated texts, as small as several sentences. The tagger could be used especially for low-resource languages and for pre-annotation during manual annotation of texts. It works well for morphologically rich languages.

**Acknowledgments.** This work has been partly supported by the Academy of Sciences of Czech Republic under the project 15-13277S and by the Ministry of Education of CR within the LINDAT-Clarin project LM2015071.

## References

1. Jakubíček, M., Kilgarriff, A., Kovář, V., Rychlý, P., Suchomel, V.: The tenten corpus family. In: 7th International Corpus Linguistics Conference CL. (2013) 125–127
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
3. Rychlý, P.: A lexicographer-friendly association score. Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN **2008** (2008) 6
4. Pala, K., Rychlý, P., Smrž, P.: Desam—annotated corpus for czech. In: SOFSEM'97: Theory and Practice of Informatics, Springer Berlin/Heidelberg (1997) 523–530