Feature Engineering in Maximum Spanning Tree Dependency Parser

Václav Novák and Zdeněk Žabokrtský

Institute of Formal and Applied Linguistics, Charles University
Malostranské nám. 25, CZ-11800 Prague, Czech Republic
{novak,zabokrtsky}@ufal.mff.cuni.cz

Abstract. In this paper we present the results of our experiments with modifications of the feature set used in the Czech mutation of the Maximum Spanning Tree parser. First we show how new feature templates improve the parsing accuracy and second we decrease the dimensionality of the feature space to make the parsing process more effective without sacrificing accuracy.

1 Introduction

Dependency parsing has become an increasingly popular discipline in the field of Natural Language Processing. There are numerous systems engaged in competitions such as CoNLL Shared Task 2006 [1]. The most successful parsers use syntactically annotated corpora for supervised training. While most of the attention is naturally drawn to the algorithms employed by the parsers, in our paper we focus on the problem of feature extraction, which is to some extent orthogonal to the problem of finding the appropriate statistical models and algorithms.

Our exploration is limited to the *a-layer* (analytical layer, layer of surface syntax) data of the Prague Dependence Treebank (PDT 2.0),[1] a corpus of Czech texts annotated with syntactical information consisting mainly of dependency relationships and labels of these relationships [2]. At this layer, PDT 2.0 contains 1,500,000 syntactically annotated tokens (around 80,000 sentences) divided into 80% for training, 10% as the development test set and 10% as the evaluation test set. Examples of a-layer sentence representations is given Figures in Figure 1 and in Figure 2. The choice of using only one treebank and one language enables us to experiment with linguistically motivated features and features taking into account the particular annotation scheme.

Our feature functions are evaluated using Ryan McDonald's implementation of Maximum Spanning Tree (MST) parser with Margin Infused Relaxed Algorithm (MIRA) for estimating the optimal feature weights [3], a modification of which was the best performing system for PDT 2.0 at the CoNLL Shared Task 2006. Other parsers with high accuracy reported on PDT 2.0 use feature extraction as well [4,5].

Experiments described in [6] show that there is still a space for accuracy improvement of automatic parsing: the combination of several parsers can reduce errors by 10%. At least a part of this improvement may be caused by the fact that the parsers use their own specific feature functions which detect some relationships hidden to the

[1] http://ufal.mff.cuni.cz/pdt2.0/

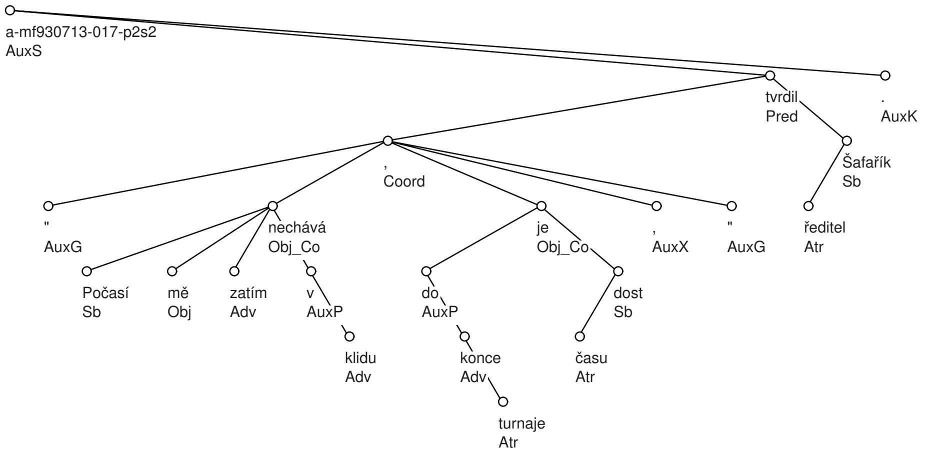


Fig. 1. Sample a-layer tree: PDT 2.0 surface-syntactic representation of the sentence *"Počasí mě zatím nechává v klidu, do konce turnaje je času dost," tvrdil ředitel Šafařík.* ("The weather leaves me calm now, there is enough time till the end of the tournament" director Šafařík said.)

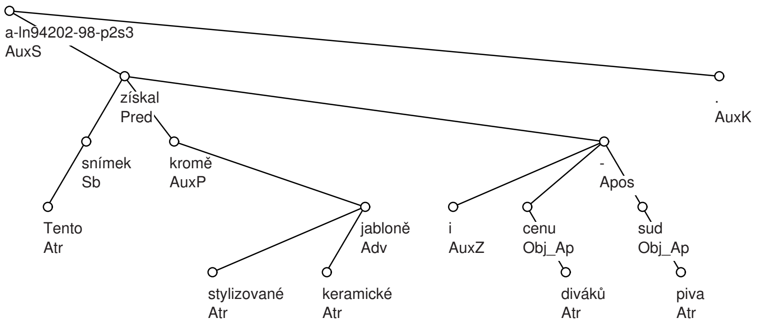


Fig. 2. Sample a-layer tree: PDT 2.0 surface-syntactic representation of the sentence *Tento snímek získal kromě stylizované keramické jabloně i cenu diváků – sud piva.* (Besides the stylized ceramic apple tree, this movie obtained also the audience prize—a barrel of beer).

others. This shows that by combining additional features we should be able, in principle, to increase the accuracy of any of these parsers. In practice this is hindered by speed and memory requirements of the algorithms. However, MIRA turns out to be able to effectively estimate a sufficiently large number of features and thus enables us to freely experiment with various feature templates.

Using millions of features requires the user to use at least a 64bit machine and the model is still being loaded for about 9 minutes. To decrease the hardware requirements we explored the possibility to exclude some of the features in the final model and show the impact on both parsing speed and accuracy.

Section 2 describes the added features and Section 2.1 presents the experimental results. Section 3 addresses the possible feature space reduction, showing the results in Section 3.2. Future work is discussed in Section 4 and we conclude by Section 5.

2 Adding New Feature Templates

Here is the list of all feature templates added to the templates used originally in the MST parser:

CAPI. Feature template indicating whether the token has the first letter in upper case. Original MST parser ignores the case of all letters.
CAPL. Indication whether the lemma given by the lemmatizer has the first letter in upper case. Original MST parser ignores the lemmatizer output.
COOR. Indication whether the two tokens are "coordinable" (based on their distance, their tags and the words in between)
DECT. Individual positions of the 15-letter tags as individual features. Original MST parser uses only 2-letter tags proposed by [5].
FULT. Full 15-letter morphological tags of candidate words as new features
LEME. Lemmata technical suffices given by the lemmatizer. As described in [7], these suffices distinguish proper names, locations, etc.
5TAG. First 5 letters of the tags as opposed to the full 15-letter tags.

2.1 Evaluation

Preliminary tests on a portion of train data showed that the **DECT** and **5TAG** feature templates decrease the accuracy. After excluding them from the set, we were able to train the modified parser on the whole training data. The modified parser achieved 84.69% accuracy, which is 2.9% relative error reduction from the baseline MST parser accuracy of 84.24%. If any of the feature templates is omitted, the parsing accuracy decreases as summarized in Table 1. The feature space dimension increased from original 15,486,593 to 21,817,386 in the best model.

Table 1. Parsing accuracy with various feature templates included. The ✓marks indicate the included templates.

CAPI	CAPL	COOR	FULT	LEME	Accuracy	Feature Space Dimension
✓	✓	✓	✓	✓	**84.69**%	21,817,386
	✓	✓	✓	✓	84.61%	21,817,362
✓		✓	✓	✓	84.66%	21,817,362
✓	✓		✓	✓	84.69%	21,816,918
✓	✓	✓		✓	84.58%	16,989,434
✓	✓	✓	✓		84.65%	19,908,745

3 Feature Space Reduction

The original feature space dimension (15,486,593) for the MST parser trained on PDT 2.0 train data causes significant difficulties for practical parsing purposes. In this section we explore the effects of reducing the dimension by removing features with the lowest weights. This approach differs from the typical frequency cutoff [8] in that it's applied after the training phase.

3.1 Selection of the Least Important Features

The absolute weights ogive in Figure 3 shows that about one third of the features has exactly zero weight after the MIRA training. Removing these features from the feature space doesn't change the parsing outcome at all.

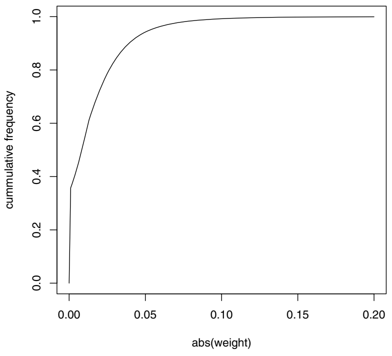


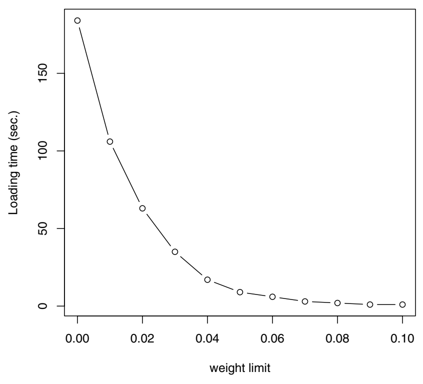
Fig. 3. Ogive of absolute values of feature weights

The next step is to remove features with the lowest absolute values of weights. After removing these features, the model is retrained to ensure that all the remaining features receive the correct weights.

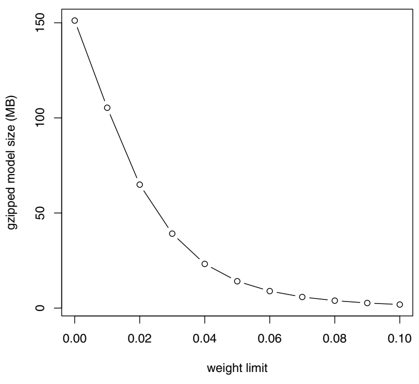
3.2 Evaluation

The experimental results are presented in Figure 4. The x-axis of all the graphs is the threshold for weight absolute value. The experiments show that introducing the threshold significantly reduces the size of the model file as well as the initial loading time. It has also a positive impact on the parsing speed. Most importantly, the parsing accuracy deterioration is much slower than the decrease in the number of features. Setting the threshold to less than 0.02 even doesn't affect the accuracy at all.

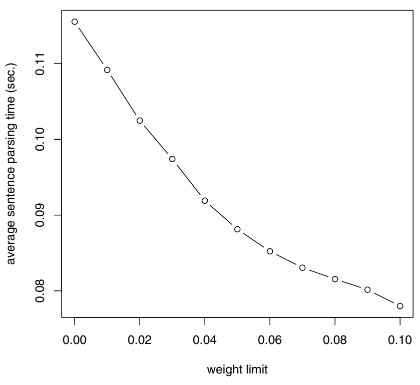
– Figure 4(a) shows the loading time of the model in seconds. The loading time is the initialization time of the parser.



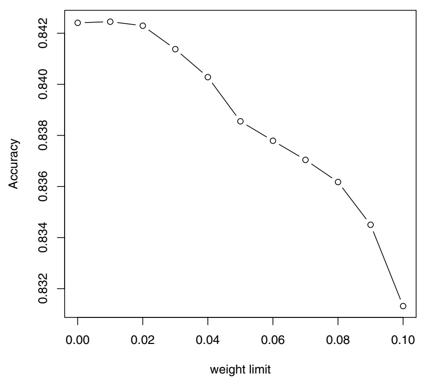
(a) Loading time of the model



(b) Size of the model file



(c) Average sentence parsing time



(d) Parsing accuracy

Fig. 4. Experimental results of feature space reduction

- Figure 4(b) shows the size of the model file loaded by the parser. The loading time is linear with the model size.
- Figure 4(c) shows the average sentence parsing time with the initialization time ignored.
- Figure 4(d) shows the unlabeled parsing accuracy, the number of correctly attached nodes.

The experiments show that after reducing the feature space dimension to 0.07% of the original size, loading time as well as size of the model decreases drastically – from 7 minutes to 1 sec., resp. from 231 MB to 1.9 MB – while the accuracy drops only by 1.11 percent points (from 84.24% to 83.13%). Values for other thresholds can be derived by combining Table 2 with the graphs in Figure 4.

Table 2. Feature space dimension with various weight absolute value thresholds

Weight Threshold	Feature Space Dimension
0.00	9,856,377
0.01	6,833,618
0.02	4,121,610
0.03	2,438,019
0.04	1,429,533
0.05	858,809
0.06	540,734
0.07	353,412
0.08	237,704
0.09	163,321
0.10	114,621

4 Future Work

The improvement caused by introducing the coordination feature template should be increased by introducing a more precise rules for coordination ability of toke pairs. Most of the useful added feature templates are not PDT 2.0 dependent and we should test them on other parsers and treebanks as well.

5 Conclusion

We have shown that by adding more feature templates we can improve the state of the art dependency parser accuracy for PDT 2.0 Czech texts. We have described the feature templates and we have shown that when we are able to train over a large feature space, the addition of full 15-letters morphological tags for Czech outperforms the 2-letters tags commonly used since [5].

We have also shown that, for practical purposes, the MST parser model size can be decreased to a fraction of the original size without a large loss of the accuracy. Moderately pruned models with the threshold less than 0.02 can even maintain the same accuracy while saving over 70% of resources.

Acknowledgment

This work was supported by Czech Academy of Science grants 1ET201120505 and 1ET101120503, and by Czech Ministry of Education, Youth and Sports projects LC536 and MSM0021620838. The views expressed are not necessarily endorsed by the sponsors.

We also thank Ryan McDonald for providing us with his parser implementation.

References

1. Buchholz, S., Marsi, E.: CoNLL-X Shared Task on Multilingual Dependency Parsing. In: Màrquez, L., Klein, D. (eds.) Proceedings of CoNLL-X, New York City, USA (2006)
2. Hajič, J. et al.: Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia (2006)

3. McDonald, R., Pereira, F., Ribarov, K., Hajič, J.: Non-projective dependency parsing using spanning tree algorithms. In: HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Vancouver, British Columbia, Canada, Association for Computational Linguistics, pp. 523–530 (2005)
4. Nivre, J., Hall, J., Nilsson, J.: Memory-Based Dependency Parsing. In: Ng, H.T., Riloff, E. (eds.) Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL), Boston, Massachusetts, USA, pp. 49–56 (2004)
5. Collins, M., Hajič, J., Ramshaw, L., Tillmann, C.: A statistical parser for Czech. In: Proceedings of the 37th Annual Meeting of the ACL, College Park, MD, USA, Association for Computational Linguistics (1999)
6. Holan, T., Žabokrtský, Z.: Combining Czech Dependency Parsers. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2006. LNCS (LNAI), vol. 4188, pp. 95–102. Springer, Heidelberg (2006)
7. Hana, J., Zeman, D., Hajič, J., Hanová, H., Hladká, B., Jeřá, E.: Manual for Morphological Annotation, Revision for the Prague Dependency Treebank 2.0. Technical Report TR-2005-27, ÚFAL MFF UK, Prague, Czech Rep. (2005)
8. Malouf, R., van Noord, G.: Wide coverage parsing with stochastic attribute value grammars. In: Su, K.-Y., Tsujii, J., Lee, J.-H., Kwong, O.Y. (eds.) IJCNLP 2004. LNCS (LNAI), vol. 3248, Springer, Heidelberg (2005)