Proceedings of the 11th International Conference on Applied Informatics
Eger, Hungary, January 29–31, 2020, published at http://ceur-ws.org

Feature Reduction for Dependency Graph Construction in Computational Linguistics

László Kovács, László Csépányi-Fürjes

University of Miskolc, Institute of Information Science, Hungary
kovacs@iit.uni-miskolc.hu
csepanyifl@iit.uni-miskolc.hu

Abstract

Dependency grammar is an important tool in semantic analysis of text sources. In the transition-based approach of dependency graph construction, the engine detects the special features of the source text and determines the next construction steps using a machine learning method. Traditionally, the feature set is constructed manually, and some of the features may be irrelevant or redundant. In this paper, we investigate the efficiency of two known feature reduction methods in a grammar induction problem. The first method uses variance minimization algorithm and the other works with an approach based on mutual information content. We propose also a normalized feature similarity for alternative cluster-based feature reduction approach. For the test evaluation, we use the sentence bank of UD (Universal Dependency) homepage taking examples from two different languages: English and Hungarian.

Keywords: computational linguistics, dependency graph, feature reduction

MSC: 68R15

1. Introduction

The analysis of sentence structures is an important field of computational linguistics. In the literature, we can find different approaches to describe the grammar of human languages as different languages may require different structure models to represent the very rich language specialties. The most widely used grammar model is the phrase-structure grammar or constituency grammar [1], where the sentence structure is based on the structural axis between the Noun-phrase and the Verb-phrase units. The phrase-structure grammar provides very good results especially

for languages with fixed word order. The other main approach of sentence level grammar is the family of dependency grammars. The idea of dependency structures dates back to the work of Frege on algebraic logic [2]. According to Frege, the semantic of a sentence is based on the predicate (verb) phrase, where the exact meaning of the sentence is given by the arguments of the predicate. In this sense, the words related to the arguments depend on the words of the predicate. The first explicit application of this semantic dependency model on sentence structuring relates to the works of Tesniere [3].

In dependency grammars, the semantic structure of a sentence is represented with the dependency graph connecting the words of the sentence. In the graph, the edges may be assigned to different semantic labels [4]. In a dependency relationship, one of the words is called head word, the other is the dependent word. This dependency corresponds to a parent - child relationship, every word as dependent unit can be bound only to one head word. The maximal number of related dependent words is called the valency of the head. In the sentence parsing module, first POS (part of speech) and other main morphological attributes are determined for the words, then the full dependency hierarchy is constructed. The grammar is called projective if the order of the words in the sentence is identical to the order of the corresponding leaf nodes in the dependency graph. The languages with flexible word order like Hungarian, theses sequences may be different, the language contains non-projective grammar structures, too. According to the analysis of Sartorio [5], the ratio of sentences with some non-projective phenomena, can achieve a frequency ratio of 25%.

In the literature, there are two main approaches to construct the dependency graph. One solution is the transition-based method where the words in the sentence are processed sequentially and the next elementary graph construction step is determined from the current processing context [6]. The method uses a predefined set of elementary transformation rules and the appropriate rule for execution is predicted using some machine learning methods. This construction module uses the following data elements: state descriptor variables, initial and finals state descriptors. Considering the practical implementations, the most widely used variant is the arc-eager method [7], where the state description is given by three base lists: the buffer of words not tested yet, stack of words under investigation and the list of words already processed. The model contains the following transformation operators: Local-left, Local-right, Reduce and Shift. The Shift operator moves one word from the buffer of words not tested yet into the stack. The Reduce operator removes the word from the head of this stack.

2. Features vectors in graph construction

The winner elementary graph construction step is generally predicted from the current context parameters. These context parameters are called context features and they relate to some grammatical parameters of the words like the POS tag or the position. The applied classifier engine uses these feature vectors as input to

determine the winner operation category. It can be shown that the efficiency of the classification process depends significantly from the feature set used to describe the context status. According to the experiences presented in [8], the application of large, extended feature set yields in an improved classification accuracy. To provide good prediction result, in the classical approach, the parser engines use a large set of features in order to access rich information about current context. In the practice, there exists a default feature set which is dominantly applied in the different application systems. Only few articles focus on the problem of optimality of the selected feature set. Among the related approaches, the dominant solution use a greedy expansion algorithm. Starting from a minimal feature set as a subset of the global feature pool, the set is extended iteratively with new features providing the largest quality increase. The main quality factor is the classification accuracy of the prediction engine.

Although large feature sets can increase the classification accuracy, the large data set decreases the time cost efficiency of the system. From this point of view, it seems reasonable to reduce the applied feature set. At the same time we can also see that the usual feature sets (like unigrams, bigrams and trigrams) are sporadically overlapping each others which suggests that some of the features are either irrelevant or even misleading (see [8]). The reduction mechanism [6] for feature set optimization eliminates the redundant features while it tries to maximize the information content of the set. In the literature, there are only few research works on this optimization problem.

In our investigation, we focus on the reduction approach to provide an optimal reduced feature set for the investigated source language. In the evaluation tests we use Hungarian and English text sources, these two languages belong to different language categories regarding the projective status of the grammar. Our hypothesis says that the default feature sets can be reduced providing a more efficient feature set selection. The main goal in our investigation is to construct a method to discover the features with low relevance.

3. Related work

The construction of a language dependency treebank is labor-intensive process. Researchers at the University of Szeged developed the first Hungarian dependency corpus by transforming the existing expression-based Szeged Treebank. The converted dependency trees were manually checked and repaired (see [9]). The creation of the dependency Treebank was motivated by the so-called UD CoNLL 2007 Shared Task, which required participants to train and test their own dependency parser system using the same data set (see [10]). The development of the Hungarian Treebank enabled the Hungarian researchers to appear in the shared task not only as the developer of a dependency parser, but also as the owner of the Hungarian data set.

The current website of the Universal Dependency (UD) project contains the so-called Hungarian Szeged Universal Treebank, which is an extract from the Hun-

garian Dependency Treebank. This publicly available excerpt contains 910 train, 441 dev, and 449 test sentences and largely follows the UD 2.0 annotation principles (see [11]).

The magyarlanc library is a tightly coupled tool chain that performs NLP tasks and allows you to perform basic operations such as: text and sentence segmentation and tokenization, lemmatization, POS tagging and dependency tree parsing of a Hungarian text. It was developed in accordance with the already mentioned Szeged Universal Treebank. For doing the dependency parsing the graph-based MATE parser was integrated into the magyarlanc system (see [12]). The reason why they selected a graph-based parser was an experiment in which they compared the popular ArcEager transition-based algorithm with the graph-based MATE parser. The result showed that the graph-based parser over performs its competitor.

You may wonder why we still focus on the transition-based algorithms. We believe that it is worth considering the use of these algorithms for the Hungarian language for two reasons. Firstly the ArcEager version of the algorithm is not capable of exploring the dependency tree of non-projective sentence structures, and is therefore unable to compete with a graph-based parser. Since Hungarian language is rich in non-projective structures, this characteristic is an important factor when we are selecting the algorithm variant. For instance there is the non-projective list-based variant, which is specifically designed to analyze non-projective sentence structures. Secondly, the text processing from left to right is similar to the way the human brain parses a written text. We think that experimenting with a transition-based algorithm - which is a classic left-to-right processor - can provide some interesting insights about the human parsing-learning process as well.

4. Feature reduction using maximum entropy

We can observe that a certain feature may be more valuable than the others from the viewpoint of the classification accuracy if its information value regarding the category label is higher than the average. If a feature shows very similar or even the same values for all the different transitions then that feature is weak in terms of classification. The above mentioned conditions are showing similarities with the relevance of information in a probabilistic system. The transitions can be viewed as possible signals in a communication channel:

$$\Omega = \{T_1, T_2, T_3, \dots\}.$$

The probabilities ($p(T_i)$) of the features determined by the frequency values in the training set can be viewed as a probability space:

$$P_\Omega = \{p(T_1), p(T_2), p(T_3), \dots\}.$$

The information content of P_Ω can be expressed with the entropy value:

$$H(P_\Omega) = -\sum_i p(T_i) log(p(T_i)).$$

The entropy gets the greatest value when the probabilities calculated for all transitions are the same and the smallest if the probability calculated for a certain transition is 1. It means that the lower the entropy the more valuable the feature template is.

To evaluate the information content of a feature f, we group the test cases by the feature value into disjoint subsets. The weighted entropy of f is calculated with

$$H_f = \sum_v |\Omega_v| \cdot H(P_{\Omega_v}),$$

where the summation runs over the possible feature values (v) and Ω_v denotes the set of test cases where the f value is equal to v.

Having an input training set, we calculate the frequency values and using these values as approximation of the probabilities, we get also the entropy for every features. Based on the entropy values, we can eliminate the features with high entropy.

5. Feature reduction using mutual information content

In the field of data analysis, the attribute reduction is a widely used preprocessing step. A key factor in attribute reduction is the dependency measure among the different features. One of the most widely used base measures is the correlation coefficient, where the correlation for two variables with continuous values (X, Y) can be given [13] with

$$r = \frac{\sum_i (X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_j (X_j - \overline{X})^2} \sqrt{\sum_j (Y_j - \overline{Y})^2}}.$$

In the literature, we can find many extensions of the base correlation measure, like the Fast Correlation-based Filter method [14] using symmetric uncertainty measure. As the features in the transition-based graph construction model are mainly categorical variables, the dependency between two variables is usually given with a contingency table which displays the multivariate frequency distribution of the variables. To measure the strength of association between the two variables, we can use measures like contingency coefficient or phi co-efficient.

In our investigation we selected an information theory oriented approach using mutual information content to measure the mutual dependence between the features. Mutual information can be calculated with

$$I(X,Y) = \sum_{x \in X, y \in Y} p_{X,Y}(x,y) \log \frac{p_{X,Y}(x,y)}{p_X(x)} - \sum_{x \in X, y \in Y} p_{X,Y}(x,y) \log p_Y(y),$$

where joint distribution is $P_{(X,Y)}$ and the marginal distributions are P_X and P_Y. The summation runs over the possible value pairs in the contingency table. This

formula can be expressed also with

$$I(X,Y) = H(X) + H(Y) - H(X,Y),$$

where $H(X)$ is the entropy for the variable X and $H(X,Y)$ denotes the joint entropy of X and Y.

5.1. Similarity measure using normalized mutual information content

The mutual information measure can take any value from the set of non-negative real numbers. In order to use a normalized similarity value, we propose the $Esim(X,Y)$ measure given with

$$Esim(X,Y) = \frac{I(X,Y)}{\min(H(X),H(Y)}.$$

It can be shown that

$$0 \leq Esim(X,Y) \leq 1$$

on the following way. As $H(X), H(Y) \geq 0$ and $I(I,Y) \geq 0$, thus $Esim(X,Y) \geq 0$. On the other hand, we know that

$$H(X,Y) \geq \max(H(X),H(Y)).$$

This means that

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \leq H(X) + H(Y) - \max(H(X),H(Y)),$$

thus

$$Esim(X,Y) = \frac{I(I,Y)}{\min(H(X),H(Y))} = \frac{H(X) + H(Y) - H(X,Y)}{H(X) + H(Y) - max(H(X),H(Y)} \leq 1.$$

Based on the $Esim(X,Y)$ similarity measure, we can perform a clustering of the features to determine the group of similar feature elements. The discovered clusters can be used to eliminate features very close to other features.

6. Experiments

6.1. Evaluating the features with the Hungarian data set

In our experiment we implemented four variants of the transition-based algorithm, namely ArcEager-Stack, ArcStandard-Stack, Projective-List and NonProjective-List (see [6]). We incorporated the feature list from the literature. Then we applied the following iterative experiment: train ⇒ evaluate ⇒ entropy calculation

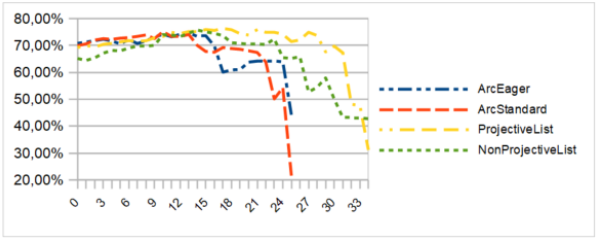

Figure 1: Feature reduction based on maximum entropy

described in Section 4 ⇒ feature reduction. In the feature reduction phase we omitted the feature that has the maximum entropy.

The numbers on the horizontal axis of Figure 1 graph show how many features have been omitted. The vertical axis shows the mean UAS (unlabeled attachment score) of the test, validation, and train data sets. It is interesting to see that omitting the high entropy features does increase the accuracy until we reach a point where even the highest entropy features are too valuable to leave out from the calculation. These results suggest that the maximum entropy calculation can be used to evaluate the features and to find ones that can be eliminated from the training. This elimination helps reducing the calculation cost of the algorithms and even increases the accuracy.

6.2. Comparing the Hungarian result to English

We compared the Hungarian result with an English training set downloaded from the UD website[1]. To eliminate the difference that comes from the different training set sizes we shortened the English set to the same size as the Hungarian one (1800 sentences). Figure 2 concludes that our feature reduction procedure appears to be independent of the examined language. The other three algorithm variants produced similar results.

6.3. Comparing the maximum entropy based reduction to randomized reduction

Finally, we wanted to investigate whether maximal-entropy based elimination is actually more beneficial than just leaving out a randomly selected feature. In this final test, the features to be discarded were randomly selected and the UAS values were compared with the aforementioned results. It can be seen that the accuracy achieved by the randomly dropped features is constantly decreasing and falling under the entropy-selected features (see Figure 2).

[1] `https://github.com/UniversalDependencies/UD_English-EWT`

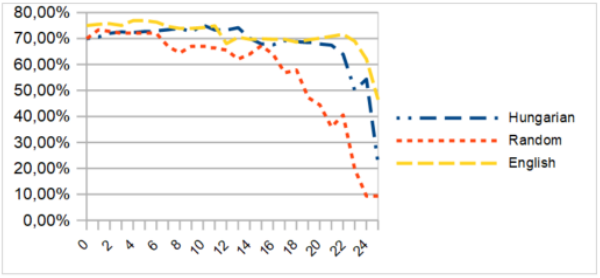

Figure 2: Feature reduction using ArcStandard-Stack algorithm, based on maximum entropy in English and Hungarian samples, and random elimination

Stack	List
$\beta[2]pos;$	$\beta[2]pos;$
$\beta[1]pos;$	$\lambda_2[0]pos;$
$\beta[0]pos;$	$\lambda_2[n]pos;$
$\sigma[0]pos;$	$\lambda_1[1]pos;$
$\beta[0]pos\beta[1]pos$	$\beta[1]pos;$

Table 1: The 5 worst performing features

6.4. Feature Clustering with Esim measure

In the mutual information content similarity approach, we used the $Esim(C, T_i)$ value to measure the importance of the feature T_i. This means, that the relevance of a feature is given with the mutual information content related to the category label (transition code). As Figure 3 shows, both measures provided the same importance order of the features.

Thus both methods can be used for feature reduction, they provide the same or similar priority orders. On the other hand, we can mention an additional benefit of the Esim method, namely, it can be used also to measure the general similarity among the different features, independently from the category label. Based on the generated distance matrix, also a clustering of the features can be constructed. We have selected the MDS (Multidimensional Scaling) method to map the elements of the feature set into the points of an Euclidean space. In our test results (see Figure 4), the single outlier point with thick border denotes the category variable. We can use clustering techniques like k-means algorithm, to determine the group of similar features. The MDS result in Figure 4 shows, for example, that S0_FORM_B0_FORM_POS and S0_FORM_POS_B0_FORM are very similar to each others. The related cluster-based feature reduction and the entropy-based reduction provide consistent results.

Esim	entropy
B2_POS	B2_POS
B1_POS	B1_POS
S0_POS	S0_POS
B0_POS	B0_POS
B2_FORM	B2_FORM
B2_FORM_POS	B2_FORM_POS
B0_POS_B1_POS	B0_POS_B1_POS
B1_FORM	B1_FORM
B1_FORM_POS	B1_FORM_POS
B0_POS_B1_POS_B2_POS	B0_POS_B1_POS_B2_POS
S0_FORM	S0_FORM
B0_FORM	B0_FORM
S0_FORM_POS	S0_FORM_POS
B0_FORM_POS	B0_FORM_POS
S0_POS_B0_POS	S0_POS_B0_POS
S0_POS_S0R_POS_B0_POS	S0_POS_S0R_POS_B0_POS
S0_POS_S0L_POS_B0_POS	S0_POS_S0L_POS_B0_POS
S0_POS_B0_POS_B0L_POS	S0_POS_B0_POS_B0L_POS
S0_POS_B0_POS_B1_POS	S0_POS_B0_POS_B1_POS
S0H_POS_S0_POS_B0_POS	S0H_POS_S0_POS_B0_POS
S0_POS_B0_FORM_POS	S0_POS_B0_FORM_POS
S0_FORM_POS_B0_POS	S0_FORM_POS_B0_POS
S0_FORM_POS_B0_FORM_POS	S0_FORM_POS_B0_FORM_POS
S0_FORM_POS_B0_FORM	S0_FORM_POS_B0_FORM
S0_FORM_B0_FORM_POS	S0_FORM_B0_FORM_POS
S0_FORM_B0_FORM	S0_FORM_B0_FORM

Figure 3: Comparison of feature priority lists (bottom items are the best) using the Esim and maximum entropy methods

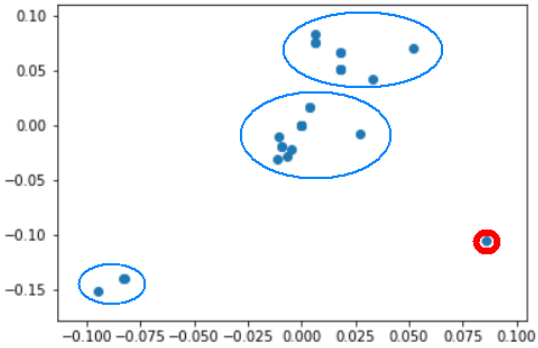

Figure 4: MDS mapping and clustering of the features based on Esim similarity

7. Conclusion

In our study, we analyze a maximum entropy based feature reduction mechanism of dependency parser algorithms. After evaluating our results, we can say that the maximum entropy-based evaluation is well suited for investigating the relevance of features. It can be seen that the accuracy achieved by the randomly dropped features is constantly decreasing and falling under the entropy-selected features. It also appears that this mechanism is language independent. By reducing the number of features in the transition-based algorithms, we can achieve cost savings

without reducing the accuracy of the parser.

Acknowledgements. The described study was carried out as part of the EFOP-3.6.1-16-00011 "Younger and Renewing University – Innovative Knowledge City – institutional development of the University of Miskolc aiming at intelligent specialization" project implemented in the framework of the Szechenyi 2020 program. The realization of this project is supported by the European Union, co-financed by the European Social Fund.

References

[1] Prószéky, G., Koutny, I., Wacha, B. (1989). A dependency syntax of Hungarian, *Metataxis in Practice* , (1989), 151–181

[2] Klement K. C., Frege and the Logic of Sense and Reference. Routledge, 2017.

[3] Tesnière, L. , Eléments de Syntaxe Structurale, Klincksiek, Paris (1959)

[4] Ágel, V. , Fischer, K. Dependency grammar and valency theory. In: *The Oxford Handbook of Linguistic Analysis*, (2009)

[5] Sartorio, F., Improvements in Transition Based Systems for Dependency Parsing, PhD Thesis, Università degli Studi di Padova (2015)

[6] Nivre, J., Algorithms for deterministic incremental dependency parsing, *Comput. Linguist.*, 34(4), (2008), 513–553.

[7] Nivre, J., Hall, J., Nilson, J. Memory-based dependency parsing, *Proceedings of CoNLL*, (2004), 49–56.

[8] Zhang, Y., Nivre, J. . Transition-based dependency parsing with rich non-local features, *Proceedings of ACL-HLT* , (2011), 188–193.

[9] Vincze, V. , Szauter, D., Almási, A. , Móra, Gy. , Alexin, Z. and Csirik, J., Hungarian Dependency Treebank, (2010)

[10] Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D., The CoNLL 2007 shared task on dependency parsing. *EMNLP-CoNLL*, (2007), 915–932.

[11] Vincze, V., Farkas, R., Simkó, K., Szántó, Z., Varga, V. . Univerzális dependencia és morfológia magyar nyelvre. *XII. Magyar Számítógépes Nyelvészeti Konferencia*,(2016)

[12] Farkas, R., Szántó, Z., Vincze, V., Zsibrita, J. , Módosított morfológiai egyértelműsítés és integrált konstituenselemzés a magyarlanc 3.0-ban, *XII. Magyar Számítógépes Nyelvészeti Konferencia.*(2016).

[13] Blessie, E. Chandra, E. Karthikeyan, Sigmis: A feature selection algorithm using correlation based method, *Journal of Algorithms and Computational Technology*, Vol 6.3, (2012), 385-394.

[14] Yu, L., Huan L., Feature selection for high-dimensional data: A fast correlation-based filter solution, *Proceedings of the 20th international conference on machine learning (ICML-03)*, (2003)