Dealing with Small, Noisy and Imbalanced Data: Machine Learning or Manual Grammars?

Adam Przepiórkowski[1,2], Michał Marcińczuk[3], and Łukasz Degórski[1]

[1] Institute of Computer Science, Polish Academy of Sciences, Warsaw
[2] Institute of Informatics, Warsaw University
[3] Institute of Applied Informatics, Wrocław University of Technology

Abstract. This paper deals with the task of definition extraction with the training corpus suffering from the problems of small size, high noise and heavy imbalance. A previous approach, based on manually constructed shallow grammars, turns out to be hard to better even by such robust classifiers as SVMs, AdaBoost and simple ensembles of classifiers. However, a linear combination of various such classifiers and manual grammars significantly improves the results of the latter.

1 Introduction

Machine learning (ML) methods gave a new stimulus to the field of Natural Language Processing and are largely responsible for its rapid development since the early 1990ies. Their success is undisputed in the areas where relatively large collections of manually annotated and balanced data of reasonably good quality are available; a prototypical such area is part-of-speech tagging.

Matters are less clear when only small amounts of noisy and heavily imbalanced training data are available; in such cases knowledge-intensive manual approaches may still turn out to be more effective. One such task is definition extraction, which may be approximated by the task of classifying sentences into those containing definitions of terms and those not containing such definitions. Previous approaches to this task usually rely on manually constructed shallow or deep grammars, perhaps with additional filtering by ML methods.

In this paper we deal with the task of extracting definitions from instructive texts in Slavic, as described in Przepiórkowski *et al.* 2007b. The aim of definition extraction here is to support creators and maintainers of eLearning instructive texts in the preparation of a glossary: an automatically extracted definition is presented to the maintainer who may reject it or, perhaps after some editing, accept it for the inclusion in the glossary. It follows from this intended application that recall is more important than precision here: it is easy to manually reject false positives while it is difficult to manually find false negatives not presented by the definition extraction system.

The approach described in Przepiórkowski *et al.* 2007b eschews Machine Learning and relies on manually constructed grammars of definitions. The aim of the work presented here is to examine to what extent the same task may be

carried out with the use of ML classifiers. In particular, we adopt the Polish data set of Przepiórkowski *et al.* 2007a consisting of 10830 sentences, 546 of which are definition sentences (i.e., they are or contain definitions). Obviously, this is a relatively small data set on which to train classifiers. Moreover, the classes are heavily imbalanced, with the ratio of definitions to non-definitions $\approx$ 1:19.

To complicate matters further, it is often not clear even for humans whether a given sentence contains a definition or not: whenever a sentence describes a certain characteristic of a notion, the annotator must decide whether this characteristic is definitional or just one of many traits of the notion. Correspondingly, the inter-annotator agreement reported in Przepiórkowski *et al.* 2007a is very low: when measured as Cohen's κ it is equal to 0.31 (the value of 1 would indicate perfect agreement, the value of 0 — complete randomness).

In the rest of the paper we first (§2) briefly present the manual grammar approach of Przepiórkowski *et al.* 2007a. In the following two sections (§§3–4) we report on the experiments of applying ML classifiers and homogeneous ensembles of classifiers to the same data, with results uniformly worse than those of manually constructed grammars. However, the ensuing section (§5) demonstrates that a combination of ML classifiers and linguistic grammars, while still not fully satisfactory, significantly improves on the results of either approach. Sections presenting some comparisons, suggesting future work and drawing conclusions end the paper (§§6–7).

2 Manual Grammars

As described in Przepiórkowski *et al.* 2007a, a rather simple shallow grammar of Polish definitions, containing 48 rules (some of them consisting of rather complex regular expressions) was developed on the basis of a development corpus of 5218 sentences (containing 304 definitions[4]) and fine tuned on a thematically different corpus of 2263 sentences (with 82 definitions). The whole grammar development process took less than 2 weeks of intensive work. The resulting grammar, called GR$'$, and a relatively sophisticated baseline grammar B3, looking for copula and similar clues for definitions, were then tested on an unseen corpus containing 3349 sentences (with 172 definitions). The results, in terms of precision (P), recall (R), the standard F-measure (F_1), as well as two F-measures giving twice (F_2) and five times (F_5; cf. Saggion 2004) more weight to recall, are presented in Table 1(a).[5]

Using the same grammars, we evaluated them on the whole corpus (hence, also on parts of the corpus which were seen during grammar development), obtaining the results in Table 1(b). Thus, any classifier with F_2 higher than 36,

[4] Note that these are definitions, not definition sentences: one sentence may contain a number of definitions and, although rarely, a definition may be split into a number of sentences.

[5] We follow Przepiórkowski *et al.* 2007a,b in using F_2 as the main measure summarising the quality of the approach, but with an eye on F_5. Also, we adopt their formula for F_α as equal to $\frac{(1+\alpha)\cdot P\cdot R}{\alpha\cdot P+R}$.

	(a) testing corpus					(b) whole corpus				
	P	R	F_1	F_2	F_5	P	R	F_1	F_2	F_5
B3	10.54	88.46	18.84	25.54	39.64	9.12	89.56	16.55	22.73	36.26
GR′	18.69	59.34	28.42	34.39	43.55	18.08	67.77	28.54	35.37	46.48

Table 1. Evaluation of B3 and GR′ on (a) the testing corpus and on (b) the whole corpus

when measured with the standard 10-fold cross-validation (10CV) procedure on the whole corpus, would clearly improve on these results.

3 Single Classifiers

In the experiments reported here we assumed a relatively simple feature space: a sentence is represented by a vector of binary features, where each feature represents an n-gram, present or not in a given sentence. More specifically, after some experiments we adopted as features unigrams, bigrams and trigrams of base forms, parts of speech and grammatical cases. We chose those n-grams which were most frequent in definitions or in non-definitions. Given the 9 n-gram types (e.g., single base forms, bigrams of base forms, trigrams of cases, etc.), for each type we selected 100 most frequent n-grams of this type. As a result, each sentence is represented by a binary vector of length 781.[6]

For the experiments we used the WEKA tool (Witten and Frank, 2005) and its implementation of simple decision trees (ID3 and C4.5), Naïve Bayes (NB) classifiers, a simple lazy learning classifier IB1, as well as the currently more popular classifiers AdaBoost (AdaBoostM1 with Decision Stumps; AB+DS) and Support Vector Machines (nu-SVC; cf. `http://www.cs.iastate.edu/~yasser/wlsvm/`). Because of the very high prevalence of one class, we experimented with different ratios of subsampling, in each case using all definitions: 1:1 (equal number of definitions and non-definitions), 1:5 (5 non-definitions for each definition), 1:10 and 1:all ($\approx$ 1:19, i.e., no subsampling). All experiments followed the general 10-fold cross-validation (10CV) methodology, with the corpus split randomly into 10 buckets of roughly the same size in such a way that each bucket contains roughly the same number of definitions (a balanced random split). The results are presented in Table 2.

As was expected, SVM and AdaBoost turned out to be the best classifiers for the task at hand, as measured by F_2. However, even the best classifier, based on Support Vector Machines with the 1:5 ratio of subsampling, turned out to give results significantly worse than the manual grammar GR′. Moreover, somewhat surprisingly, different ratios of subsampling turned out to be optimal for different types of classifiers: for AdaBoost the best ratio was 1:1, for C4.5, ID3, IB1 and SVM it was 1:5, while for Naïve Bayes it turned out to be 1:all (no subsampling).

[6] The length is shorter than 900 because the numbers of grammatical classes, cases and bigrams of cases are smaller than 100 each.

Classifier	Ratio	P	R	F_1	F_2	F_5	Comments
NB	1:1	9.50	60.07	16.41	21.66	31.84	
	1:5	10.53	54.58	17.65	22.79	32.16	
	1:10	10.75	51.83	17.80	22.79	31.66	
	1:all	10.94	49.82	17.94	**22.80**	31.28	
C4.5	1:1	8.25	59.89	14.50	19.41	29.31	
	1:5	14.81	30.04	19.84	**22.37**	25.65	
	1:10	19.48	16.48	17.86	17.37	16.92	
	1:all	32.35	10.07	15.36	13.07	11.38	
ID3	1:1	8.66	66.85	15.33	20.63	31.53	
	1:5	12.79	37.91	19.12	**22.91**	28.56	
	1:10	14.78	26.00	18.85	20.75	23.08	
	1:all	15.65	17.77	16.64	17.00	17.37	
IB1	1:1	9.68	50.73	16.26	21.02	29.73	
	1:5	15.94	26.19	19.82	**21.57**	23.66	
	1:10	20.00	18.86	**19.42**	19.23	19.04	
	1:all	21.85	14.28	17.28	16.15	15.16	
nu-SVC	1:1	11.79	69.05	20.14	26.37	38.16	nu=0.5
	1:5	20.75	37.55	26.73	**29.57**	33.08	nu=0.2
	1:10	27.11,	27.66	**27.38**	27.47	27.56	nu=0.1
	1:all	33.33	16.67	22.22	20.00	18.18	nu=0.05
AB+DS	1:1	11.59	68.32	19.82	**25.97**	37.63	1000 iterations
	1:5	28.13	23.44	**25.57**	24.82	24.11	1000 iterations

Table 2. Performance of the classifiers for different ratio of positive to negative examples evaluated on the whole corpus with balanced random split

4 Homogeneous Ensembles

In the next stage of experiments, homogeneous ensembles of classifiers were constructed. Experiments were conducted with the 6 types of classifiers with the best subsampling (cf. the numbers in bold in Table 2), plus additional subsampling ratios of IB1, SVM and AdaBoost which gave promising results in other experiments, not reported here for lack of space. In case of Naïve Bayes, the best performance was obtained without subsampling, although the results were only insignificantly better than 1:5 and 1:10 subsampling, with the subsampling configurations performing better in ensembles. For this reason NB without subsampling was not considered further. The summary of the best remaining ensembles, in comparison with the two grammars, is presented in Table 3. For each of these 9 classifiers, homogeneous ensembles (i.e., collections of classifiers of the same type) were constructed consisting of 1, 3, 5, 9 and 15 classifiers, with the final decision reached via simple voting. In most cases, with the exception of one type of IB1 and one type of AdaBoost, ensembles of 15 or 9 classifiers gave best results. Note that, again, SVM and AdaBoost gave best results and, again, while the ensemble of 9 SVMs (with 1:5 subsampling) reached F_2 close to that of GR′ (31.49 vs. 34.39/35.37), no classifier surpassed the manual approach in terms of the two F-measures favouring recall.

Classifier	P	R	F_1	F_2	F_5
9 × nu-SVC (1:5)	24.11	37.18	29.25	31.49	34.10
9 × AdaBoost 1000 it. (1:1)	13.24	72.34	22.39	29.08	41.48
15 × ID3 (1:5)	24.73	29.30	26.82	27.60	28.43
15 × NB (1:10)	10.75	52.01	17.81	22.81	31.71
9 × C4.5 (1:5)	24.07	24.91	24.48	24.62	24.76
15 × IB1 (1:5)	17.80	24.54	20.63	21.79	23.08
9 × nu-SVC (1:10)	30.79	26.56	28.52	27.83	27.18
1 × AdaBoost 1000 it. (1:5)	28.13	23.44	25.57	24.82	24.11
1 × IB1 (1:10)	20.00	18.86	19.42	19.23	19.04
Grammar B3	9.12	89.56	16.55	22.73	36.26
Grammar GR′	18.08	67.77	28.54	35.37	46.48

Table 3. Performance of the selected classifiers and the grammars evaluated on the whole corpus (10CV)

At this point, much more time had been spent on ML experiments than the "less than two weeks" spent by Przepiórkowski *et al.* 2007a on the development of manual grammars for the same task. Of course, this does not warrant the conclusion that definition extraction should be approached linguistically rather than statistically, as many factors play a role here, including the level of expertise in grammar writing, the experience in constructing classifiers, the assumed feature space, the exact character of the data, etc. Nevertheless, it seems that in case of small, noisy, imbalanced data, a manual "linguistic" approach may be a viable alternative to the dominant statistical machine learning paradigm.

5 Linear Combination of Grammars and Ensembles

If simple homogeneous ensembles of common classifiers do not give better results than manual grammar, perhaps they can be combined with the grammars to improve their results? Various such modes of combination are possible and, in a different paper, we describe some promising results of a sequential combination of the baseline grammar B3 and ML classifiers (Degórski *et al.*, 2008).

In this section we present the results of a linear combination of the 9 ensembles of classifiers introduced in the previous section, each treated as a single classifier here, with the two grammars: B3 and GR′.

In order to assign weights to particular classifiers, let us first introduce some notation. Let $D^+(x)$ mean that x is a definition, $D^-(x)$ — that x is not a definition, $D_i^+(x)$ — that x is classified as a definition by the classifier i, $D_i^-(x)$ — that x is classified as a non-definition by the classifier i, and finally, TP_i, etc. are the numbers of true positives, etc., according to the classifier i.

We can estimate the probability $p_i^+(x)$ that a given sentence x is really a definition, if the classifier i says that it is a definition, in the following way:

$$p_i^+(x) = p(D^+(x)|D_i^+(x)) = \frac{p(D^+(x) \wedge (D_i^+(x)))}{p(D_i^+(x))} \approx \frac{TP_i}{TP_i + FP_i}$$

Similarly, given that the classifier says that x is not a definition, the probability of x actually being a definition is:

$$p_i^-(x) = p(D^+(x)|D_i^-(x)) = \frac{p(D^+(x) \wedge (D_i^-(x)))}{p(D_i^-(x))} \approx \frac{FN_i}{FN_i + TN_i}$$

Let us then define $d_i(x)$ as follows:

$$d_i(x) = \begin{cases} \frac{TP_i}{TP_i+FP_i}, & \text{if } x \text{ is classified as definition} \\ \frac{FN_i}{FN_i+TN_i}, & \text{if } x \text{ is classified as non-definition} \end{cases}$$

Assuming that each of the N classifiers votes for the definitory status of x with the strength proportional to the estimated probability given above, the decision of the whole ensemble of N classifiers may be calculated as:

$$d(x) = \frac{\sum_{i=1}^{N} d_i(x)}{N}$$

If $d(x) > \delta$, the linear combination classifies x as a definition, otherwise — as a non-definition.

What is the best value of the cut-off point δ? The examination of different values close to the estimated probability that a sentence is a definition (cf. Table 4) shows that for $\delta = 0.08$, F_2 reaches almost the value of 39, significantly higher than either the F_2 for the grammar GR′ alone or the best F_2 for pure ML classifiers.[7]

It is interesting to what extent the improvement is the effect of combining various types of ML classifiers, and to what extent the presence of grammars B3 and GR′ affects the results. To this end, final experiments were performed, where three linear combinations of classifiers were trained on the part of the corpus seen when developing the grammars (cf. §2) and tested on the remaining unseen portion of the data.[8] These combinations are: the 9 ML classifiers (9ML), 9 ML classifiers and B3 (9ML+B3), and finally all 11 classifiers (9ML+B3+GR′). The best results of these combinations (i.e., for the best cut-off points) are presented in Table 5 and they clearly indicate the crucial role played by the full grammar GR′ in such heterogeneous ensembles.[9]

[7] In fact, in some of the other experiments, with weights assigned in less principled ways, F_2 exceeded 39. Moreover, this value is also higher than F_2 for the unanimous voting combination of B3 and GR′, where $F_2 = 37.28$, as measured on the whole corpus.

[8] This way of evaluation is unfavourable both to the grammars (they are tested on data unseen during their development) and to ML classifiers (they are trained on a smaller part of the corpus than in case of 10CV). When tested on the whole corpus, with 10CV, the best F_2 results for 9ML, 9ML+B3 and 9ML+B3+GR′ were, respectively, 35.80, 35.75 and, as already reported, 38.90. Note that the first two results, for combinations without GR′, are still lower than the results for the combination of B3 and GR′ mentioned in the previous footnote.

[9] But note that here the result of 9ML+B3+GR′ is only slightly better than that of GR′ alone as tested on the same data; cf. Table 1(a).

δ	P	R	F_1	F_2	F_5
0.05	12.53	84.25	21.81	28.97	43.11
0.06	17.63	71.79	28.31	35.48	47.49
0.07	21.32	61.72	31.69	37.82	46.90
0.08	25.17	53.48	34.23	38.90	45.04
0.09	27.08	46.52	34.23	37.54	41.55
0.10	30.61	39.74	34.58	36.15	37.86
0.11	32.66	32.60	32.63	32.62	32.61
0.12	36.97	28.57	32.23	30.91	29.70
0.13	40.64	25.46	31.31	29.08	27.15

Table 4. Performance of the linear combination of classifiers for various values of δ as evaluated on the whole corpus (10CV)

classifier	δ	P	R	F_1	F_2	F_5
9ML	0.07	18.16	40.11	25.00	28.59	33.38
9ML+B3	0.07	18.56	41.21	25.60	29.30	34.25
9ML+B3+GR′	0.06	17.19	57.14	26.43	32.20	41.19

Table 5. The effect of grammars on the performance of the linear combination of classifiers, evaluated on the testing corpus

6 Comparisons and Future Work

We are not aware of other work of similar scope comparing and combining machine learning and linguistic approaches to definition extraction, or to other NLP tasks based on small, noisy and heavily imbalanced data, although there is a rich literature on combining inductive and manual approaches to tagging, where a similar synergy effect is usually observed. Previous work on definition extraction, mainly for English and other Germanic languages, usually consists of a simple sequential combination of grammatical parsing and ML filtering. Often only precision or only recall is cited, so it is difficult to directly compare our approach to these other approaches.

There are very many possible improvements to the work reported here, starting from the selection of features, through the selection of classifiers for the linear combination, to the better assignment of weights to particular (homogeneous ensembles of) classifiers. Moreover, in other work (Degórski *et al.*, 2008) we describe a sequential combination of the baseline grammar B3 and ML classifiers which achieves results comparable to GR′, but without the need for the development of a grammar more sophisticated than B3. A rather different approach worth pursing seems to be the employment of random forests (Breiman, 2001). Although basic random forests have already been applied in NLP with satisfactory results (Xu and Jelinek, 2004), *balanced* random forests (Chen *et al.*, 2004), particularly well suited in heavily imbalanced classification tasks, still remain to be explored (see Kobyliński and Przepiórkowski 2008).

7 Conclusion

In the days of the — fully deserved — dominance of inductive methods, any solutions involving the manual coding of linguistic knowledge must be explicitly justified. We have shown that, in case of a task relying on very low-quality (small, noisy, imbalanced) training data, manual methods still rival statistical approaches. On the other hand, even in such difficult tasks, ML may be very useful, not as a replacement of hand-coded grammars, but as a support for them: our combination of linguistic grammars and homogeneous ensembles of various classifiers achieves results significantly higher than either of the two pure approaches. We conclude that, while firing linguists may initially increase the performance of a system, perhaps a few of them should be retained in a well-balanced heterogeneous NLP team.

References

Breiman, L. (2001). Random forests. *Machine Learning*, **45**, 5–32.

Chen, C., Liaw, A., and Breiman, L. (2004). Using random forest to learn imbalanced data. Technical Report 666, University of California, Berkeley. `http://www.stat.berkeley.edu/tech-reports/666.pdf`.

Degórski, Ł., Marcińczuk, M., and Przepiórkowski, A. (2008). Definition extraction using a sequential combination of baseline grammars and machine learning classifiers. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC 2008*, Marrakech. ELRA. Forthcoming.

Kobyliński, Ł. and Przepiórkowski, A. (2008). Definition extraction with balanced random forests. In *6th International Conference on Natural Language Processing, GoTAL 2008*, Gothenburg. Forthcoming.

Piskorski, J., Pouliquen, B., Steinberger, R., and Tanev, H., editors (2007). *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing at ACL 2007*, Prague.

Przepiórkowski, A., Degórski, Ł., and Wójtowicz, B. (2007a). On the evaluation of Polish definition extraction grammars. In Z. Vetulani, editor, *Proceedings of the 3rd Language & Technology Conference*, pages 473–477, Poznań, Poland.

Przepiórkowski, A., Degórski, Ł., Spousta, M., Simov, K., Osenova, P., Lemnitzer, L., Kuboň, V., and Wójtowicz, B. (2007b). Towards the automatic extraction of definitions in Slavic. In Piskorski *et al.* (2007), pages 43–50.

Saggion, H. (2004). Identifying definitions in text collections for question answering. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004*, Lisbon. ELRA.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition. `http://www.cs.waikato.ac.nz/ml/weka/`.

Xu, P. and Jelinek, F. (2004). Random forests in language modeling. In D. Lin and D. Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 325–332, Barcelona. ACL.