Robust Ending Guessing Rules with Application to Slavonic Languages

Preslav NAKOV
EECS, CS Division,
University of California, Berkeley
Berkeley, CA, 94720
USA
nakov@cs.berkeley.edu

Elena PASKALEVA
Linguistic Modelling Department, IPP
Bulgarian Academy of Sciences
25A, Acad. G. Bontchev St
Sofia, Bulgaria, 1113
hellen@lml.bas.bg

Abstract

The paper studies the automatic extraction of diagnostic word endings for Slavonic languages aimed to determine some grammatical, morphological and semantic properties of the underlying word. In particular, ending guessing rules are being learned from a large morphological dictionary of Bulgarian in order to predict POS, gender, number, article and semantics. A simple exact high accuracy algorithm is developed and compared to an approximate one, which uses a scoring function previously proposed by Mikheev for POS guessing. It is shown how the number of rules of the latter can be reduced by a factor of up to 35, without sacrificing performance. The evaluation demonstrates coverage close to 100%, and precision of 97-99% for the approximate algorithm.

1 Introduction

An important property of the Slavonic languages is the rich morphology, which determines the specifics of their representation and processing in NLP applications. This variety is arranged not only linearly along the paradigmatic axe, i.e. abundance of wordforms for a given lemma (up to 52 forms for the Bulgarian verb), but also in the derivational tree (up to 30 members per word formation). The grammatical system of the Slavonic languages and their descriptions differentiate these two mechanisms as *word formation* and *word derivation*.

The word formation building blocks define the so called *inflectional classes*, which represent sequential letter strings associated with word classes as well as with individual words, also known as *i-suffixes* in Porter-like stemmers (Porter,1980). The derivational building blocks represent derivational suffixes listed in grammars (*d-suffixes* in Porter-like stemmers). A considerable part of the Slavonic *d*-suffixes change not only the part of speech (POS) but also the semantics of the newly formed word. When multiple *d*-suffixes are concatenated, the word formation chain yields also a semantic derivation. For example, the chain (*observe → observer → observing → observability*):

наблюд-(авам) →
наблюд-**ател** →
наблюдател-**ен** →
наблюдателн-**ост**

represents the derivation:

verb → noun → adjective → noun

but also the following semantic transformation:

action → actor → feature → abstract feature

The combination of grammatical and semantic functions of the Slavonic *d*-suffixes, together with their frequent usage (at least for some of them) and the high productivity, make very attractive the idea to study the regularities and the predictive power of ending letter combinations in a large text set. We believe the results obtained over a representative collection can be used in a variety of *robust analysis applications*. Linguistically, we interpret the last term as operations over a large text set with insufficient linguistic support, typically given by a lexical database, grammatical rules, parsing rules etc. We target applications like POS tagging, text categorisation, information extraction, word sense disambiguation, question answering etc.

Below we concentrate on the automatic extraction of a set of diagnostic word endings for Bulgarian that can determine the POS as well as some grammatical, morphological and semantic properties of the underlying word. This is a two-step process including endings identification & learning and application & evaluation.

The paper is organised as follows. Section 2 discusses the related work on POS guessing and general morphology. Section 3 introduces our basic resource: the Large Grammatical Dictionary of Bulgarian. Section 4 describes two algorithms for ending guessing rules induction (an exact and an approximate one) and how to reduce the number of rules by a factor of up to 35. Section 5 contains the experimental setup and evaluation trying to predict POS, gender, number, article and semantics. Section 6 discusses the results and Section 7 points to direction for future work.

2 Related Work

POS guessing. Kupiec (1992) uses pre-specified suffixes and performs statistical learning for POS guessing. The XEROX tagger comes with a list of built-in ending guessing rules (Cutting et al.,1992). In addition to the ending, Weischedel et al. (1993) exploit capitalisation. Thede and Harper (1997) consider contextual information, word endings, entropy and open-class smoothing. A similar approach is presented in (Schmid,1995). Ruch et al. (2000) combine POS guessing, contextual rules and Markov models to build a POS tagger for biomedical text. A very influential is the work of Brill (1997), who induces more linguistically motivated rules exploiting both a *tagged corpus* and a *lexicon*. He does not look at the affixes only, but also checks their POS class in a lexicon. Mikheev (1997) proposes a similar approach, but learns the rules from *raw* as opposed to *tagged* text. Daciuk (1999) speeds up the process by means of finite state transducers.

General morphology. Nakov et al. (2003) use ending guessing rules to predict the morphological class of unknown German nouns. Schone and Jurafsky (2000) apply latent semantic analysis for a knowledge-free morphology induction. DeJean (1998), Hafer and Weiss (1974) follow a successor variety approach: the word is cut, if the number of distinct letters after a pre-specified sequence surpasses a threshold. Goldsmith (2001) performs a minimum description length analysis of the morphology of several European languages using corpora. Gaussier (1999) induces derivational morphology from a lexicon by means of *p*-similarity based splitting. Jacquemin (1997) focuses on the morphological processes. Van den Bosch and Daelemans (1999) propose a memory-based approach, which maps directly from letters in context to categories that encode morphological boundaries, syntactic class labels and spelling changes. Yarowsky and Wicentowski (2000) present a corpus-based approach for morphological analysis of both regular and irregular forms based on four models including: relative corpus frequency, context similarity, weighted string similarity and incremental retraining of inflectional transduction probabilities. Another interesting work, exploiting capitalisation and fixed/variable suffixes, is presented in Cucerzan and Yarowsky (2000).

3 Source Data

As the related work above shows, a large lexical database is often needed for the automatic identification of good diagnostic word endings. In particular, in our experiments we used the *Large Grammatical Dictionary of Bulgarian* (Paskaleva,2003), created at the Linguistic Modelling Department of the Bulgarian Academy of Sciences (CLPP-BAS) and comprising approximately 995,000 wordforms (about 65,000 lemmas), encoded in DELAF format (Silberztein,1993). The following information is listed for each wordform: 1) *lemma*; 2) *lemma properties* (POS, additional grammatical features related to the word formation: gender, e.g. for the nouns; degree, e.g. for the adjectives; transitivity, for verbs; kind for pronouns/numerals, etc.); and 3) *properties of the wordform* as a member of the lemma paradigm. The first group of properties represent our primary learning resource, as we focus on the extraction of ending rules for whole word *classes* and not for individual words.

4 Ending Guessing Rules Extraction

Our learning algorithms produce lists of endings of various length (up to 8 letters), predicting different kinds of linguistic information (see below for details):

- *POS: adjective/adverb/noun/numeral/verb*
- *article: definite/indefinite/none*
- *gender: feminine/masculine/neutre/none*
- *number: singular/plural/none*
- *semantics*: *human/animate/none*

We use two different algorithms, inducing *exact* and *approximate* ending rules, accordingly.

4.1 Exact Rules

A study of the ending letter sequences of the dictionary entries and their properties shows the well known inverse correlation between the length of a word ending and its ambiguity: the shorter the string, the more likely to be ambiguous.

This raises the idea of a simple algorithm producing 100% correct rules[1]. Suppose we want to predict POS and let us consider all wordforms in the dictionary that end on "-a". There are 203,420 of them, distributed as follows[2]: *V=128,162(63.00%)*, A=42,262(20.78%), N=32,597(16.02%), NU=240(0.12%), ADV=99(0.05%), PRO=38(0.02%), INTJ=7(0.00%), CONJ=7(0.00%), PC=6(0.00%), PREP=2(0.00%). Let us now consider a sequence with an additional starting letter, e.g. "-та". There are 83,375 wordforms with this ending, distributed in POS as follows: *V=42,843(51.39%)*, A=22,225(26.66%), N=18,092(21.70%), NU=157(0.19%), ADV=48(0.06%), PRO=9(0.01%), CONJ=1(0.00%). When a further letter is included,

[1] As it is 100% precise it risks over fitting and thus a low coverage. We will return to this issue later.

[2] We use the following abbreviations for the ten POS: A (adjective), ADV (adverb), CONJ (conjunction), INTJ (interjunction), N (noun), NU (numeral), PC (particle), PREP (preposition), PRO (pronoun) and V (verb).

we obtain e.g. "-ата" with a total frequency of 72,235 and a POS distribution: *V=42249(58.49%)*, A=21415(29.65%), N=8399(11.63%), NU=119(0.16%), ADV=47(0.07%), PRO=6(0.01%). Next, for "-цата" we have a frequency of only 799 and an even lower ambiguity: *N=793(99.25%)*, A=6(0.75%). Finally, there is a single POS tag for "-ицата": *N=726(100.00%)*. Note how the most likely tag (shown in italic for each ending above) and the degree of certainty about it change. At the beginning, the most likely tag was V, but later it changed to N. In addition, the uncertainty does not necessarily decrease monotonically as the most likely tag changes from V(63.00%) to V(51.39%) to V(58.49%) to N(99.25%) and to N(100.00%). Generalizing this example, we obtain the following

Exact Algorithm:

1. $S = \varnothing$

   E = {all possible endings of dictionary wordforms, up to k letters long};

2. While $E \neq \varnothing$

   2.1. Take a random ending e from E of *minimum length*.

   2.2. If all wordforms in the dictionary that end on e have the same POS then $S \Leftarrow e$.

3. Output S.

Number of Different POS	Wordforms	
	count	%
1	936,409	97.37%
2	24,913	2.59%
3	356	0.03%
4	1	0.00%

Table 1: Dictionary ambiguity with respect to POS.

While it is clear that this approach produces only 100% correct rules (and also the shortest possible ones), its coverage is not guaranteed to be 100% due to homography, i.e. the same graphemic wordform can be met in the dictionary multiple times with different annotations. For example, "отбрана" is annotated as[3]:

отбрана,отбера.V+F+T:Psf
отбрана,отбран.ADJ:sf
отбрана,отбрана.N+F:s

The first one denotes the inflected wordform *selected* of the finite transitive verb *select*, the second one stands for the feminine adjective *selected*, and the last one, for the feminine noun *defence*.

In fact, the level of ambiguity is relatively low: 97.37% of the wordforms in the dictionary are unambiguous, so ignoring the ambiguity on training is not unreasonable. See Table 1 for a detailed dictionary ambiguity distribution with respect to POS.

4.2 Approximate Rules

Our approximate rules are similar to the ones proposed by Mikheev (1997), who uses a dictionary to build POS prediction rules with four parts: deletion (−), addition (+), checking against the dictionary (?) and POS assignment (→). Generally speaking, each rule operates either on the *beginning* or the *ending* of the target wordform. For example, the following rule says that if an unknown word ends on "-ied", this ending should be stripped, "-y" should be appended, a check should be performed of whether the newly created word is in the dictionary and annotated as (VB VBP) there, and if so, (JJ VBD VBN) for the original word should be predicted:

e[−ied +y ?(VP VBP) → (JJ VBD VBN)]

All rule elements are optional, except for the POS assignment. This means that a rule can just add and/or remove letters, without looking in the dictionary (although it could potentially benefit from doing so). When both removal and addition are used, one can account for mutations in the word stem. In fact, Mikheev uses the following restricted types of rules: *Prefix* (prefix deletion and dictionary lookup), *Suffix*^{0} (suffix deletion and dictionary lookup), *Suffix*^{1} (suffix deletion with mutation in the last letter and dictionary lookup), *Ending* (suffix deletion). There are separate ending guessing rules for *hyphenated*, *capitalised* and *all other* words.

Given a dictionary, a scan through the wordforms is performed, during which all possible rules are collected and scored, and those above some threshold are selected. Finally, rule merging is applied to rules with identical preconditions but different predictions: the new rule predicts the union of the predictions of the original rules, which results in higher ambiguity but possibly allows the new rule to pass above the threshold after being rescored.

We do not use the full power of the Mikheev-like rules and we limit ourselves to ending rules without dictionary lookup and single class predictions. Further, at present we do not treat the hyphenated or capitalised wordforms in any special way.

The intuition behind the Mikheev's rule score is that a good guessing rule should be *unambiguous* (predicts a particular class without or with only

[3] The format used is as follows "*inflected_form*, *lemma* . *lemma_properties* : *wordform_properties*"

very few exceptions), *frequent* (must be based on a large number of occurrences) and *long* (the longer the rule the lower the probability that it will happen by chance and thus the better its prediction). These criteria are combined in the following formula:

$$score = p - \frac{t_{(1-\alpha)/2}^{(n-1)} \sqrt{\frac{p(1-p)}{n}}}{1+\log(l)}$$

where:

- l is the rule length;
- x is the number of successful rule guesses;
- n is the total number of training instances compatible with the rule;
- p is a smoothed version of the maximum likelihood estimation $\hat{p}$, which ensures that neither p nor $(1-p)$ could be zero: $p = (x+0.5)/(n+1)$;
- $\sqrt{\frac{p(1-p)}{n}}$ is an estimation of the dispersion;
- $t_{(1-\alpha)/2}^{(n-1)}$ is a coefficient of the t-distribution with $n-1$ degrees of freedom and confidence level α.

It is important to note that Mikheev weights the rule frequencies with the frequencies of the wordforms they match as estimated from raw text. We performed experiments both with and without such weighting.

Threshold	Original	Cleaned	
		100% only	everything
0.00	738,446	115,474	20,846
0.50	597,238	89,324	18,663
0.80	122,439	27,477	4,881
0.90	55,144	15,071	2,459
0.95	22,015	7,673	1,402

Table 2: Mikheev-like rules for POS guessing count: original and cleaned (100% correct and all).

Column 2 of Table 2 gives an idea about the number of selected ending guessing rules for POS prediction when different thresholds are used (and when the training was performed on a subset of the dictionary, containing 894,915 wordforms, as described below). We were unhappy with such a large number of rules, especially after we observed that they were highly redundant. For example, if the threshold is set to 0.95, all the rules listed in Table 3 (and many more) are selected. In fact, all these are covered by the ending "-хте", which is 100% correct, and they all predict that the POS should be *verb*. So, all we need is to keep "-хте", while dropping all other longer endings that have additional starting letters[4]. This reduces the number of rules by a factor of 3 to 7 (see column 3 of Table 2).

Thinking again, we can see that we can reduce the number of rules even further. For example, there is a rule "-ваха", which is scored 0.99967073 and was met 6,593 times as a verb and only once as a noun (i.e. it is 99.98% correct). There is another one "-яваха", which is scored 0.99943267, and was met 1,498 times, always as a *verb*. There are also rules like "-аваха", "-кваха", etc. Obviously, all they, and any other ending on "-ваха", will make the same prediction, so we do not need to keep them. Removing the redundancies of this kind leads to another dramatic drop in the number of rules by a similar factor (see column 4 of Table 2). In the experiments below we always applied this kind of cleaning.[5]

Ending	Score	Frequency
-енявахте	0.98336703	47
-нявахте	0.99666399	241
-явахте	0.99944650	1,489
-вахте	0.99987014	6,546
-ахте	0.99992176	11,346
-хте	0.99995697	22,074

Table 3: Some redundant selection for "-хте".

5 Evaluation

We ran two different general types of experiments: using the dictionary only and using additional raw text to estimate the frequencies of the dictionary words. We split the dictionary into two parts at random: 894,915 wordforms for training (about 90%) and the remaining 99,624 wordforms for testing. In the *dictionary-only* experiments we extracted the ending guessing rules by observing the endings of all wordforms from the training part of the dictionary[6]. We then applied the rules thus learned (each time preferring the longest one that is compatible with the target word) to the testing part of the dictionary and we measured the *precision P* (what % of the cases the predicted class matched the hypothesised one) and the *coverage C* (what % of the cases there was at least one rule that was compatible with the target wordform). We also calculated a kind of *F*-measure, which is normally

[4] Table 3 does not list all of them and there are several dozens additional highly scored ones, e.g. "-ехте".

[5] It looks like Mikheev (1997) did not observe that kind of redundancy.

[6] For a given word, we extracted all the corresponding endings up to 8 letters long. This can possibly be the whole word.

defined as $2PR/(P+R)$, where R is the *recall* (proportion of proposed instances out of all that have to be found). Precision, recall and F-measure are defined in the information retrieval community in terms of positive and negative documents for a given query, i.e. with respect to a *single class*, but here we have multiple of them. While we can define both an *overall* and a *class-specific* precision, it makes sense to talk about *recall* with respect to a particular class, but about *coverage*, when this is a measure for *all* classes. So, we redefined the F-measure as $2PC/(P+C)$.

In the *dictionary+text* experiments, we use the same training and testing parts of the dictionary, and in addition, the frequencies for the corresponding words in the training and testing text sets, accordingly. I.e. a wordform in the text that is not in the dictionary is ignored and the rest are treated as if they have been repeated in its training/testing part the same number of times as they were met in the training/testing raw text.

We used a collection of 23.5 MB of various genres of Bulgarian texts as follows:

- *legal*: 742 KB
- *poetry*: 236 KB
- *prose*: 1,032 KB
- *religion*: 393 KB
- *newspapers*: 21,118 KB

We used 2,211 KB of the newspaper texts for testing (about 10% of the collection) and the rest for training. As we already mentioned above, the same graphemic wordform can be met in the dictionary multiple times with different annotations. In such cases, we treated them as equally likely both on training and testing. This resulted in 1,751,963 wordforms tokens on training and 18,832 on testing. The huge difference is due to the fact that on testing we have both 10 times smaller dictionary and 10 times smaller text set to estimate the wordform frequencies from, which multiply and result in 100 fold drop.

For all experiments, we excluded the wordforms from a stoplist composed of the closed class words, i.e. the ones with the following POS (counts in parentheses): auxiliary verbs (91), conjunctions (31), interjections (28), particles (41), prepositions (69) and pronouns (286). We have been hesitating also about the numerals but there were 582 of them in the dictionary and one can produce more, so they do not represent a closed class and we did not include them. A potential problem with the stopwords removal is that many of them can also be non-stop ones depending on their POS, e.g.: while под/preposition (*under*), тези/pronoun (*these*) and бил/auxiliary (*has been*) are stop-words, под/noun (*floor*), тези/noun (*theses*) and Бил/person (*Bill*) are not. We did not try to address this problem (which would have required POS tagging and possibly morphological analysis, which is unacceptable, given our task) and we simply removed all homographs that matched a stoplist wordform.

We performed several experiments trying to assess the performance of the ending guessing rules as predictors for POS, article, gender, number and semantics. The details follow.

5.1 POS

We do a major distinction, between the following five open POS classes: *A* (adjective), *ADV* (adverb), *N* (noun), *NU* (numeral) and *V* (verb). Remember that we already excluded the auxiliary verbs, conjunctions, interjections, particles, prepositions and pronouns (and all their homographs). Some statistics are shown in Table 4 and the results of the evaluation are presented on Figure 1. Note the differences in the distribution of the dictionary vs. text ending frequency estimations. Note also how the results for training and testing using raw text lead to consistently lower performance. The same observation can be made for the other kinds of predictions, see Figures 2-7.

Class	Dictionary		Text	
A	129,828	17.34%	217,035	17.33%
ADV	661	0.09%	62,996	5.03%
N	84,303	11.26%	646,890	51.65%
NU	408	0.05%	12,112	0.97%
V	533,453	71.26%	313,327	25.02%

Table 4: Prior (training) distribution of *POS*.

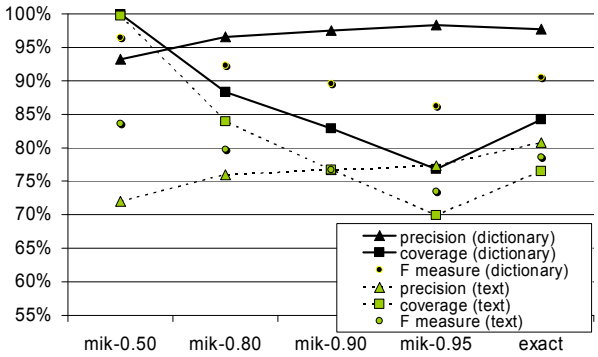

Figure 1: Results for *POS*.

5.2 Article

We learn rules to distinguish between three classes of articles: *definite*, *indefinite* and *none*. Unlike English, the articles in Bulgarian[7] appear augmented at the end of one of the words in a noun phrase, typically the first one. The *feminine* and *neutre* no-

[7] Bulgarian and Macedonian are the only Slavonic languages with definite articles of this kind.

uns, adjectives, numerals and some verb forms (e.g. participles) have the same form for both definite and indefinite articles (e.g. *defence*: отбрана → отбраната/*(in)def*), while for masculine these are distinct (e.g. *man*: човек → човека/*indef*, човекът/*def*). We ran two experiments: *with* (see Table 5 and Figure 2) and *without* POS (see Table 6 and Figure 3). Note that we certainly need the *none* class in a real system so we had to include it.

Class	Dictionary		Text	
def	324,253	39.31%	393,658	28.01%
indef	250,345	30.35%	703,116	50.02%
none	250,345	30.35%	308,802	21.97%

Table 5: Prior (training) distribution for *article* (no POS).

Class	Dictionary		Text	
A+def	73,866	10.00%	96,909	7.89%
A+indef	48,327	6.54%	116,282	9.47%
N+def	43,426	5.88%	230,506	18.78%
N+indef	40,343	5.46%	390,609	31.82%
NU+def	229	0.03%	4,927	0.40%
NU+indef	179	0.02%	7,185	0.59%
V+def	142,500	19.28%	5,635	0.46%
V+indef	137,692	18.63%	66,798	5.44%
none	252,407	34.16%	308,802	25.15%

Table 6: Prior (training) distribution of *article* (with POS).

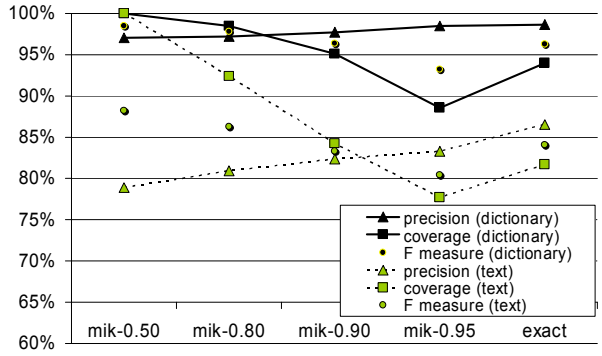

Figure 2: Results for *article* (*no POS*).

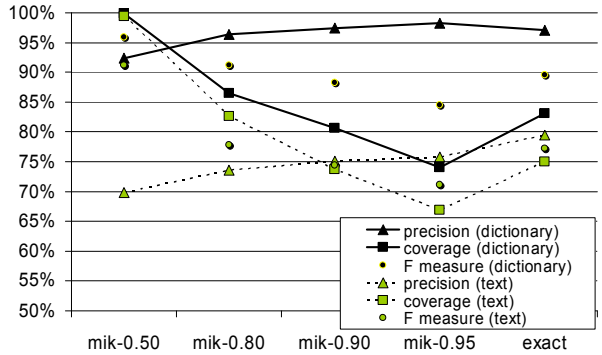

Figure 3: Results for *article* (*with POS*).

5.3 Gender

There are three genders in Bulgarian: *masculine*, *feminine* and *neuter*. Only some of the word classes can have gender, namely: adjectives, nouns, numerals and some verb forms (e.g. participles). The results of the gender guessing experiments are shown in Tables 7, 8 and Figures 4, 5.

Class	Dictionary		Text	
Fem	112,201	13.65%	87,856	5.76%
Mas	150,426	18.30%	110,361	7.24%
Neu	121,134	14.73%	87,608	5.74%
none	438,386	53.32%	1,239,183	81.26%

Table 7: Prior (training) distribution of *gender* (no POS).

Class	Dictionary		Text	
A+fem	30,465	3.96%	56,414	3.89%
A+mas	38,490	5.00%	59,306	4.09%
A+neu	25,258	3.28%	30,556	2.11%
N+neu	276	0.04%	4,592	0.32%
NU+fem	68	0.01%	2,862	0.20%
NU+mas	161	0.02%	6,582	0.45%
NU+neu	65	0.01%	1,139	0.08%
V+fem	67,385	8.76%	12,326	0.85%
V+mas	96,529	12.55%	28,660	1.97%
V+neu	72,040	9.37%	9,671	0.67%
none	438,386	57.00%	1,239,183	85.38%

Table 8: Prior (training) distribution of *gender* (with POS).

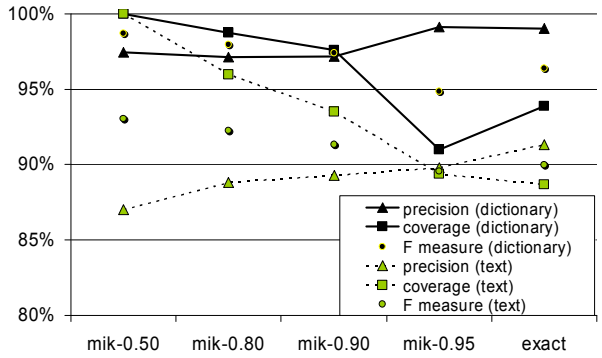

Figure 4: Results for *gender* (*no POS*).

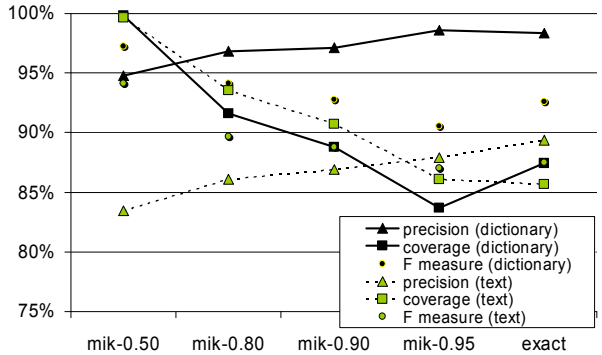

Figure 5: Results for *gender* (*with POS*).

5.4 Number

There are two grammatical numbers in today's Bulgarian: *singular* and *plural*[8]. Again, only some of the word classes can have number, namely: adjectives, nouns, numerals and some verb forms (e.g. participles). Tables 9, 10 and Figures 6, 7 for the results of the number guessing experiments.

Class	Dictionary		Text	
Plural	146,592	17.49%	299,134	20.84%
Singular	455,186	54.32%	827,131	57.62%
none	236,260	28.19%	309,276	21.54%

Table 9: Prior (training) distribution of *number* (no POS).

Class	Dictionary		Text	
A+pl	29,281	3.92%	68,144	5.48%
A+sg	100,535	13.44%	148,845	11.97%
N+pl	31,766	4.25%	163,403	13.14%
N+sg	46,317	6.19%	468,577	37.69%
NU+pl	57	0.01%	69	0.01%
NU+sg	197	0.03%	8,218	0.66%
V+pl	67,557	9.03%	25,939	2.09%
V+sg	235,896	31.54%	50,657	4.07%
none	236,260	31.59%	309,276	24.88%

Table 10: Prior (training) distribution of *number* (with POS).

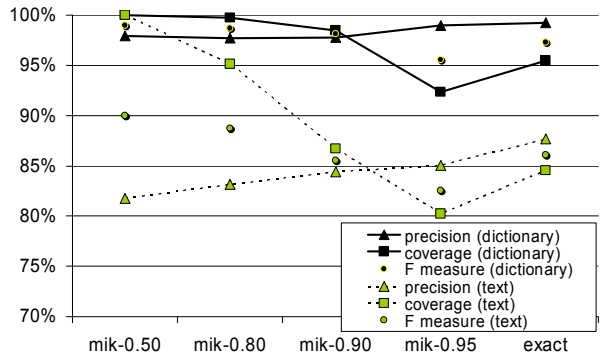

Figure 6: Results for *number* (*no POS*).

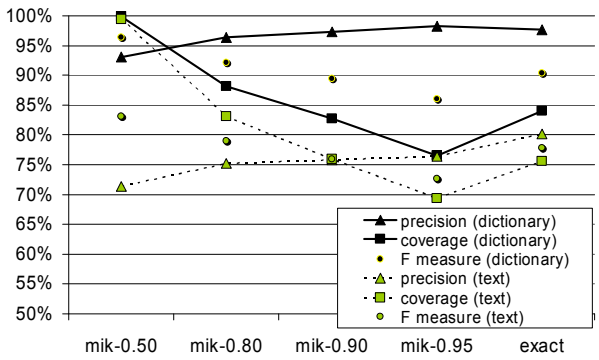

Figure 7: Results for *number* (*with POS*).

[8] The Old Bulgarian language used to have also a *dual number*. The only Slavonic language this grammatical number has been preserved in is Slovenian.

5.5 Semantics

The last kind of experiments we performed was recognising some kind of semantics. We tried to guess whether a wordform is a *human*, *animate* or *neither*, as we had such information in our dictionary. These are always limited to nouns (at least in our dictionary annotations), so we did not have separate experiments with and without POS (they would have produced almost the same result, except for some potential problems caused by homographs with a non-noun POS). The results are shown in Table 11 and Figure 8.

Class	Dictionary		Text	
Animate	1,765	0.21%	4,536	0.28%
Human	26,918	3.14%	121,299	7.39%
none	828,887	96.66%	1,516,053	92.34%

Table 11: Training (prior) distribution of *semantics*.

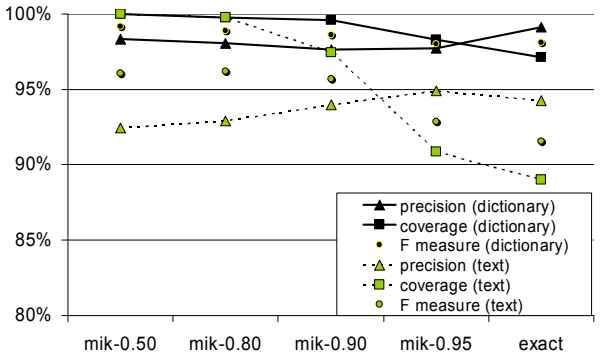

Figure 8: Results for *semantics*: *human*/*animate*.

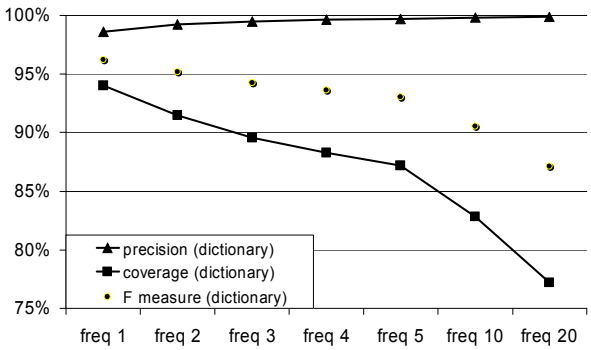

Figure 9: Results for *article*: the exact algorithm for different thresholds.

Figures 1-8 show that the approximate rules with confidence score of 0.50 perform consistently better than the exact ones, where we keep every single rule, even the ones met only once. So, we are very likely to over fit. One way to prevent this is to ignore some of the least reliable rules. The simplest criterion for this is the minimum frequency. We performed some experiments, setting it to 1, 2, 3, 4, 5, 10 and 20. The results are shown on Figure 9, where we can see that while gaining a little bit

on recall, we lose a lot on precision. Thus, if we stick to the exact rules, we apparently cannot gain by removing some of the rules based on frequency. In fact, this is not necessarily true, as it could be possible when using more complex criterion that takes into account more than just frequency, e.g. rule length.

6 Discussion

Table 12 contains summary results for the experiments with the exact and the approximate rules (with a threshold of 0.50, since, as Figures 1-8 show, it had the highest *F*-measure). The first two columns describe the kind of experiment and the method, followed by the precision, coverage and *F*-measure. Finally, the last two columns show the corresponding number of rules used and the number of target classes.

Experiment	Method	P	C	F	# rules	# class
article	exact	***98.61%***	94.00%	96.25%	53,216	3
article	mik-.50	97.02%	***99.97%***	**98.47%**	10,745	3
article+POS	exact	***97.01%***	83.09%	89.51%	85,061	9
article+POS	mik-.50	92.33%	***99.84%***	**95.94%**	27,263	9
gender	exact	***99.04%***	93.88%	96.39%	39,309	4
gender	mik-.50	97.43%	***100.00%***	**98.70%**	7,263	4
gender+POS	exact	***98.35%***	87.45%	92.58%	53,385	11
gender+POS	mik-.50	94.79%	***99.81%***	**97.24%**	12,473	11
number	exact	***99.21%***	95.49%	97.31%	40,856	3
number	mik-.50	97.90%	***100.00%***	**98.94%**	7,493	3
number+POS	exact	***97.60%***	84.07%	90.33%	81,154	9
number+POS	mik-.50	93.08%	***99.92%***	**96.38%**	20,144	9
POS	exact	***97.70%***	84.23%	90.47%	79,609	5
POS	mik-.50	93.23%	***100.00%***	**96.50%**	18,663	5
semantics	exact	***99.10%***	97.13%	98.11%	43,902	3
semantics	mik-.50	98.33%	***99.99%***	**99.15%**	9,971	3

Table 12: Experiments summary (dictionary).

There are several interesting observations about Table 12 (and Figures 1-8). First, the precision of the exact rules is consistently higher than that of the approximate ones with a threshold of 0.50. This is not surprising as the ending guessing rules produced by the exact method are guaranteed to be 100% correct on the training set (but not necessarily on the testing one, as we explained above). Figures 1-8 show that this observation holds for all other score thresholds considered, even for 0.95 (remember that the score reflects not only the rule accuracy but also its length and smoothed frequency). The situation is reversed with respect to the coverage: the exact rules have a lower coverage, which more than compensates for their higher precision. As a result, the *F*-measure is consistently lower for the exact algorithm as compared to the approximate one with a threshold of 0.50. Figures 1-8 show this is not the case for higher thresholds (especially 0.95) when the coverage becomes lower and the *F*-measure gets worse as compared to that of the exact method.

Comparing article, gender and number to article+POS, gender+POS and number+POS, accordingly, where the number of classes is increased by a factor of 3, we can see that the exact algorithm remains *robust* with respect to precision: there is a decrease of about 1-1.5% only. The precision of the approximate rules is decreased by 3-4%. On the other hand, the coverage of the approximate rules is virtually unaffected and stays very close to 100% (decreased by less than 0.2%), while for the exact rules it drops significantly: by 6-9%. As a result, the approximate algorithm has a more robust *F*-measure, which drops by 1-2.5% only, while for the exact algorithm this is 4-7%.

The approximate method is also more robust with respect to the number of rules, as it produces about five times less of them as compared to the exact one. When article, gender and number are combined with POS, the number of rules is increased by a factor of 2 to 3.

Overall, the approximate rules with a threshold of 0.50 exhibit a very high coverage (100% or very close) and precision/*F*-measure of about 97-99%.

Finally, the tasks are not equally hard. The easiest one is semantics, and the hardest one is POS.

Class	P	R	F
A+fem	91.67%	87.58%	89.58%
A+mas	92.21%	85.83%	88.91%
A+neu	83.78%	85.44%	84.60%
N+neu	16.24%	3.82%	6.18%
NU+fem	44.44%	80.00%	57.14%
NU+mas	85.71%	81.82%	83.72%
NU+neu	85.71%	60.00%	70.59%
V+fem	93.88%	97.73%	95.77%
V+mas	93.10%	96.94%	94.98%
V+neu	88.10%	96.79%	92.24%
None	98.66%	96.49%	97.56%

Table 13: **Testing** performance per class for *gender+POS approximate rules 0.50* (dictionary).

Class	P	R	F
A+fem	96.79%	96.96%	96.88%
A+mas	97.04%	95.74%	96.39%
A+neu	94.83%	95.03%	94.93%
N+neu	21.74%	18.29%	19.87%
NU+fem	80.00%	100.00%	88.89%
NU+mas	94.74%	81.82%	87.80%
NU+neu	85.71%	66.67%	75.00%
V+fem	98.36%	99.12%	98.74%
V+mas	98.41%	98.49%	98.45%
V+neu	95.92%	97.42%	96.67%
None	99.27%	99.01%	99.14%

Table 14: Testing performance per class for *gender+POS exact rules* (dictionary).

It is interesting to observe the performance of the different classes in a particular experiment, e.g. gender+POS. Note that now we can calculate a *true recall* as opposed to coverage, as we can work with a particular class. The results for the gender+POS, dictionary trained, experiments are shown in Tables 13 and 14. We can see that the precision, the recall and the *F*-measure of the exact rules are consistently better for each class as compared to the ones obtained using approximate rules (with threshold of 0.50). Note however that the recall here is calculated only for the part for which there was a prediction. The exact rules covered 84,512 out of all 96,643 wordforms (coverage: 87.45%) and 83,120 of them were correct (precision: 98.35%). The per-class *P*, *R* and *F* are calculated only for those 84,512 wordforms for which a prediction has been made. I.e. we did not assign the non-covered wordforms the class *none* by default, although probably we should, as it is the most frequent one. The approximate rules made predictions for 96,458 wordforms (coverage: 99.81%) 91,430 of which were correct (precision: 94.79%).

Table 15 shows the performance for the approximate rules as evaluated on the training set[9]. Out of the 867,567 wordforms, 866,786 have been covered (coverage 99.91%), 835,330 of which correctly (precision 96.37%). We see that the class N+neu was hard to predict not only on testing but also on training.

Class	P	R	F
A+fem	95.60%	91.35%	93.43%
A+mas	96.36%	90.64%	93.41%
A+neu	87.26%	90.41%	88.81%
N+neu	83.67%	8.50%	15.44%
NU+fem	98.53%	83.75%	90.54%
NU+mas	90.96%	89.44%	90.20%
NU+neu	98.48%	89.04%	93.53%
V+fem	95.91%	98.59%	97.23%
V+mas	94.58%	98.49%	96.50%
V+neu	89.21%	99.10%	93.90%
none	99.53%	97.29%	98.40%

Table 15: **Training** performance per class for *gender+POS approximate rules 0.50* (dictionary).

Something that Table 12 does not show, but one can see on Figures 1-8, is the consistently worse performance of training & testing on the dictionary vs. training & testing only on these dictionary words that are met in the raw text, using the corresponding frequencies. The major reason for this is the insufficient amount of training text. While the number of word tokens is high, the number of word types is much less than that of the dictionary. So, the significantly lower variability of wordforms more than compensates any gains of having real word frequencies. We believe weighting through real text is important and we plan to re-run these experiments with word frequencies estimated from orders of magnitude more textual data (it is cheap and freely available on the Web). Another, less attractive alternative could be to add the dictionary as a text. That way we would have incorrect frequency estimations for some of the words, but also the learning algorithm would have access to the rich word variability of the dictionary words.

[9] We do not show a corresponding training accuracy table for the exact rules as every cell there is replaced with 100%, i.e. there is a perfect fit.

7 Future Work

There are several possible extensions to the work presented above. First, the exact algorithm can be extended with non-exact rules. Second, the Mikheev-like ending guessing rules construction could be augmented with a *merging phase* as originally proposed. It would be interesting to consider using the dictionary not only during rules generation but also during their application: e.g. try to add/remove suffixes/prefixes and check whether the newly obtained word is listed in the dictionary (e.g. we might have the word наблюдател/noun (*observer*) but not наблюдател**ен**/adj (*observing*), generated following a standard derivational rule). There are prefixes, mostly foreign, that can attach to any open class word, but the resulting words are unlikely to be in our dictionary: “анти-” (anti-), “ултра-” (ultra-), “супер-” (super-), “контра-” (contra-). Furthermore, there are some important prefixes, specific to Bulgarian, that can limit the possible POS: e.g. “по-” and “най-” (“-” is part of the prefix) are used to construct a comparative and a superlative form, accordingly, and can be used only with adjectives, adverbs and some verb forms (e.g. participle). We believe in the potential of the combined evidence from both prefixes and suffixes. In addition, it seems important to allow for mutations in the word stem as these are common in Slavonic languages. Finally, it might be beneficial to learn separate rules for capitalised and dashed words (but maybe it is not that important as their usage is less frequent, especially the capitalisation).

We would like to try other scoring and smoothing approaches. We did not address the problem of selecting the best threshold (although it is clear that it should be low, maybe around 0.50). One way to do this is to split the training set into rules-training and threshold-training sets. Next, it looks promising to try to estimate the dictionary word frequencies using a search engine instead of text corpus, as proposed by Lapata and Keller (2004).

While the exact algorithm performed worse due to insufficient coverage[10], we believe it has a potential, e.g. if extended with some approximate rules. Note that the way the exact rules were built is very similar to the standard algorithm for decision tree construction. Thus the corresponding tree cutting criteria used to prevent over fitting can help decide when to go further and look at longer endings and when to stop.

It is interesting to see how the proposed rules perform for other Slavonic languages. In particular, we plan similar experiments for Russian as a comparable morphological dictionary with the same kind of linguistic annotations is already available.

Finally, we would like to explore the machine learning potential offered by morphological dictionaries with application to other related tasks such as stemming (Nakov, 2003), lemmatisation and POS tagging.

References

E. Brill. 1997. Unsupervised Learning of Disambiguation Rules for Part of Speech Tagging. *Natural Language Processing Using Very Large Corpora*. Kluwer Academic Press. 1997.

S. Cucerzan, D. Yarowsky. 2000. Language independent minimally supervised induction of lexical probabilities. *ACL*, 270-277.

D. Cutting, J. Kupiec, J. Pedersen, P. Sibun. 1992. A practical part-of-speech tagger. *ANLP*, 133-140.

J. Daciuk. 1999. Treatment of Unknown Words. *Workshop on Implementing Automata*. IX-1–IX-9.

H. DeJean. 1998. Morphemes as necessary concepts for structures: Discovery from untagged corpora. *Workshop on Paradigms and Grounding in Natural Language Learning*, 295-299.

E. Gaussier. 1999. Unsuppervised learning of derivational morphology from inflectional lexicons. *Workshop on Unsupervised Learning in Natural Language Processing (ACL)*.

R. Goldsmith. 1998. Automatic collection and analysis of German compounds. *Workshop on Computational Treatment of Nominals (COLING-ACL)*, 61-69.

M. Hafer, S. Weiss. 1974. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10:371-385.

C. Jacquemin. 1997. Guessing morphology from terms and corpora. *ACM SIGIR*, 156–167.

J. Kupiec. 1992. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6(3):225-242.

M. Lapata, F. Keller. 2004. The Web as a Baseline: Evaluating the Performance of Unsupervised Web-based Models for a Range of NLP Tasks. *ACL*, 121-128.

A. Mikheev. 1997. Automatic Rule Induction for Unknown Word Guessing. *Computational Linguistics*, 23(3):405-423.

Nakov P. 2003. BulStem: Design and Evaluation of Inflectional Stemmer for Bulgarian. *Workshop on Balkan Language Resources and Tools (Balkan Conference in Informatics)*.
http://iit.demokritos.gr/skel/bci03_workshop/papers/

P. Nakov, Y. Bonev, G. Angelova, E. Gius, W. von Hahn. 2003. Guessing Morphological Classes of Unknown German Nouns. *Recent Advances in Natural Language Processing,* 319-326.

E. Paskaleva. 2003. Compilation and validation of morphological resources. *Workshop on Balkan Language Resources and Tools (Balkan Conference on Informatics)*.
http://iit.demokritos.gr/skel/bci03_workshop/papers/

M. Porter. 1980. An algorithm for suffix stripping. *Program* 14(3):130-137.

P. Ruch, R. Baud, P. Bouillon, G. Robert. 2000. Minimal Commitment and Full Lexical Disambiguation: Balancing Rules and Hidden Markov Models. *CoNLL (ACL-SIGNLL)*, 111-115.

H. Schmid. 1995. Improvements in part-of-speech tagging with an application to German. Feldweg and Hinrichs, eds., *Lexikon und Text*, 47-50.

P. Schone, D. Jurafsky. 2000. Knowledge-Free Induction of Morphology Using Latent Semantic Analysis. *CoNLL (ACL-SIGNLL)*, 67-72.

Silberztein M. 1993. Dictionnaires electroniques et analyse automatique de textes: le systeme INTEX. Masson, Paris.

S. Thede S., M. Harper. 1997. Analysis of Unknown Lexical Items using Morphological and Syntactic Information with the TIMIT Corpus. *Workshop on Very Large Corpora,* W97-0124.

A. Van den Bosch, W. Daelemans. 1999. Memory-based morphological analysis. *ACL*, 285-292.

R. Weischedel, R. Schwartz, J. Palmucci, M. Meteer, L. Ramshaw. 1993. Coping with Ambiguity and Unknown Words through Probabilistic Models. *Computational Linguistics*, 19(2):359-382.

D. Yarowsky, R. Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. *ACL*, 207-216.

[10] Note that we can achieve 100% coverage by assigning the examples that are not covered to a default class (e.g. *none* or the most frequent one). It would be interesting to compare the precision of the exact and the approximate rules under these conditions.