Design and Evaluation of Inflectional Stemmer for Bulgarian

Preslav Nakov

University of California at Berkeley, EECS, Berkeley CA 94720, USA

nakov@eecs.berkeley.edu

Abstract: *The paper starts with an overview of some important approaches to stemming for English and other languages. Then, the design, implementation and evaluation of the BulStem inflectional stemmer for Bulgarian are presented. The problem is addressed from a machine-learning perspective using a large morphological dictionary. A detailed automatic evaluation in terms of under-stemming, over-stemming and coverage is provided. In addition, the effect of stemming and BulStem parameters setting is demonstrated on a particular task: text categorisation using kNN+LSA.*

Keywords: *Stemming, lemmatisation, text categorisation, k-nearest-neighbour, vector-space model, latent semantic analysis, information retrieval.*

1. INTRODUCTION

The contemporary statistical text data processing relies exclusively on individual words, and less often on multiword terms, as the basic language element. Thus, the definition of *wordfrom* from the point of view of the particular algorithm is of crucial importance. Some options include: the *surface form* as seen in the text (possibly converted to lowercase), the *lemma* (the canonical form) after inflexions removal, the *root* or the *stem* (a prefix shared by the different forms of the same word). In the latter case a stem can group together a set of *inflected forms* only (of the same root), but often includes *derivational* variants as well.

The basic aim of stemming is to normalise the word variants by means of convertion to corresponding stems. The stemming algorithms are usually limited to suffix stripping only and do not cope with prefixes (although this might be useful in some domains, e.g. chemistry [*Paice,1990*]) since this can change the word sense dramatically (e.g. *able* vs. *unable*). Generally speaking, the wordforms variety can be due to both inflectional and derivational morphology (also compounding, misspelling, proper names, foreign words etc. [*Nakov et al.,2003b*]), and the stemming algorithms use both.

Historically, the primary usage and most of the research on stemming was motivated by *information retrieval* (*IR*). Despite the contradictory evidence in the past [*Harman,1991*], nowadays it is largely accepted that stemming improves IR, although not necessarily significantly: while Krovetz reports 30-40% [*Krovetz, 1993*], in a later evaluation Hull announces only 1-3% [*Hull,1996*]. The stemming is often regarded as a recall enhancement tool, although it can also lead to better precision at low recall levels [*Kraaij,1996*].

Due to the low inflectional variability of English, building a stemmer is relatively simple, unless one tries to address the derivational variants. This is not the case for morphologically richer languages, where the impact of stemming is potentially bigger, but building an accurate algorithm becomes a much more challenging task. Experiments show stemming is beneficial for several European languages including French [*Savoy,1993*], Dutch [*Kraaij& Pohlmann,1994*], Latin [*Schinke et al.,1996*], as well as the highly inflectional Slavonic ones: e.g. Slovene [*Popovic&Willett,1992*], Russian and Ukrainian [*Kovalenko,2002*].

According to Porter, stemming is applicable to all Indo-European (and Uralic) languages. An evidence for this is the Snowball project whose aim is to provide both a specialised programming language and a centralised repository for description and implementations in Snowball, C and Java of algorithms following the Porter stemmer [*Snowball*]. The currently supported languages include: English, French, Spanish, Portuguese, Italian, German, Dutch, Swedish, Norwegian, Danish, Russian and Finnish.

2. RELATED WORK

Most of the research on stemming was done for English and there are several approaches of different complexity. The simplest ones are limited to "-*s*" removal only. An example stemmer, described in [*Harman,1991*], includes just the following 3 plural stripping rules:

1) "*-ies*" → *"-y"* (*not* applied, when "*-eies*" or "*-aies*")

2) "*-es*" → *"-e"* (*not* applied, when "*-aes*", "*-ees*" or "*-oes*")

3) "*-s*" → *"-"* (*not* applied, if "*-ss*" or "*-us*")

Some simple stemmers, in addition to "*-s*", remove in a similar fashion other frequent English suffixes, e.g. "*-ed*" and "*-ing*". More sophisticated algorithms often work iteratively by submitting the word to several subsequent transformations, and rely on dozens and even hundreds of rules: e.g. Dawson [*Dawson,1974*] has over 1,200 rules, Lovins [*Lovins,1968*] – 294, Paice/Husk [*Paice,1990*] – 115, and Porter [*Porter,1980*] – 60. Experiments show the Porter algorithm, despite being the most compact one, is at least as good as the others in a precision/recall evaluation for IR [*Hull,1996*].

According to the Frakes' classification [*Frakes,1982*], the algorithms mentioned above are all members of the class of the *affix removals*. Here are also the *prefix n-gram* stemmers, which use no grammatical information at all, and just strip anything but the first *n* letters.

There are three other classes: *table lookup*, *successor variety* and *n-gram*. *Table lookup* refers to storing the desired stem for each particular surface form. It is fast (no word transformation is required), but also huge, domain-dependent, and necessarily of limited coverage: no table can list all possible words of a natural language since new ones are constantly added. In addition, the Slavonic languages are rich in homographs and best disambiguated by means of *part-of-speech* (*POS*) information.

The *n-gram* approach has been introduced by Adamson and Boreham, who split a word into digrams (e.g. *orders* => {*or,rd,de,er,rs*}, *ordering* => {*or,rd,de,er,ri,in,ng*}), calculate the *Dice coefficient* and apply *single-link clustering*. The algorithm is costly and does not create real stems: just a set of equivalence classes (see [*Adamson&Boreham,1974*]).

The *successor variety* approach has been introduced by Hafer and Weiss. It is corpus-based and observes the number of distinct letters following a particular prefix: the successor variety. It scans the word to be stemmed and finds the cut point where the successor variety increases sharply. Several variations are possible, including: *cut-off*, *peak and plateau*, *complete word* and *entropy* [*Hafer&Weiss,1974*].

More recent stemming algorithms combine several different sources including: dictionary lookup, statistical information based on observations in real texts, POS tagging and morphological analysis. For example the *KSTEM* algorithm proposed by Krovetz is based on a combination of machine-readable dictionary and a set of inflectional and derivational morphological rules [*Krovetz,1993*]: the word passes through several transformations, but the process is interrupted immediately, if the current form is found in the dictionary. This prevents "news" from being transformed into "new", but also never conflates "stocks" and "stock", which causes problems in some domains. As a result *KSTEM* does not outperform consistently the Porter stemmer [*Krovetz,1993*]. Xu and Croft [*Xu& Croft,1998*] propose a corpus-based way around. They start with an aggressive stemmer (e.g. Porter or *prefix n*-gram) and then break the equivalence classes (defined as the set of words sharing the same stem) using graph theoretic algorithms.

Stemming for the highly inflectional Slavonic languages (Bulgarian has up to: 7 wordforms for nouns, 9 wordforms for adjectives and 52 wordforms for verbs) is often regarded as not easier than full morphological analysis. The latter usually requires a large morphological dictionary of wordforms and a POS tagger to disambiguate the homographs in context. At the same time, despite the difficulties, it is possible to perform stemming for Slavonic languages without the need of neither POS tagging nor morphological analysis. The Porter algorithm for example, has been successfully adapted to Russian under the *Snowball* project [*Snowball*]. Another system, named *Stemka*, has been built for Russian and later adapted to Ukrainian [*Kovalenko,2002*]. It is claimed to be more accurate than *Snowball* and *SegmMorph* (morphological lemmatiser, [*Kukushkina&Polikarpov, 1996*]).

We will describe *Stemka* in more detail since our approach is somewhat related. *Stemka* deals with inflectional morphology only and addresses the problem as a machine-learning task. The stemmer is trained on a large text corpus pre-processed by a morphological analyser, which finds the exact stem (defined as the prefix shared by all inflected forms) for each known word token (the unknown words are simply ignored during training). For each word a rule consisting of suffix to remove (as determined by the morphological analyser) within a two-letter left context is defined, e.g. for the word *морями* this will be (*–ор–,–ями*). The algorithm scans through the words in the text and calculates the frequency of each rule. The least frequent ones are discarded and the rest are used for stemming. An interesting property of *Stemka* is that it does not try to disambiguate in case more than one rule can be applied to a particular word but returns all possible stems instead (with the additional limitation that the stem should contain at least one vowel). E.g. for *начинающийся* it would return *начина|ющ|ий|ся*, leaving to the calling program the decision where to cut.

3. BulStem: A Bulgarian Stemmer

We address the problem as a machine-learning task from a large morphological dictionary of Bulgarian, created at the Linguistic Modeling Laboratory, CLPOI-BAS (for contacts: Elena Paskaleva). The dictionary is rich in morphological information and the version we used contained 889,665 wordforms (59,670 lemmas), encoded in DELAF format [*Silberztein,1993*]. Each line contains a wordform, corresponding lemma and morphological information, e.g.:

```
отбран,отбера.Г+С+Т:Ps
отбран,отбран.ПРИ:s
отбрана,отбера.Г+С+Т:Psf
отбрана,отбран.ПРИ:sf
отбрана,отбрана.С+Ж:s
отбраната,отбера.Г+С+Т:Psfd
отбраната,отбран.ПРИ:sfd
отбраната,отбрана.С+Ж:sd
```

Our purpose is to assign the same stem to all inflected forms for a given combination of lemma and its POS. The examples above are included in the following groups (*Г*: verb *select*, *ПРИ*: adj. *selected*, *С*: noun *defense*):

отбера.Г+С+Т: отбера отберат отбере отберели отберем отберете отбереш отбереше отбери отберял отберяла отберяло отберях отберяха отберяхме отберяхте отбра отбрал отбрала отбралата отбрали отбралите отбралия отбралият отбрало отбралото отбран отбрана отбраната отбрани отбраните отбрания отбраният отбрано отбраното отбрах отбраха отбрахме отбрахте

отбран.ПРИ: отбран отбрана отбраната отбрани отбраните отбрания отбраният отбрано отбраното

отбрана.С+Ж: отбрана отбраната отбрани отбраните отбрано

We find the corresponding stem for each group (*отб*, *отбран* and *отбран*) and generate a contextual removal rule for each inflected form: e.g. for *отбраният* and *отбран* we obtain a rule saying that the suffix *-ият* is removed, when its left context is *-ран-*. The length of the left context is a fixed program parameter (here 3). We go through the words in the dictionary and collect the removal rules together with their frequency. We then drop the most infrequent ones and build a substitution list (e.g. *–раният* changes to *-ран*). The top rules look like this:

```
вания ==> ван 2587
ване ==> ван 2548
ванията ==> ван 2524
ването ==> ван 2524
остите ==> ост 2259
ости ==> ост 2259
ост ==> ост 2247
остта ==> ост 2238
ява ==> ява 1632
яваше ==> ява 1631
явало ==> ява 1631
яваха ==> ява 1616
```

The stemming is performed by applying the longest possible rule (if any), provided that the stem produced contains at least one vowel. An example follows:

Example text (from www.mediapool.bg):
Има първи вероятен случай на атипична пневмония в България, съобщи министърът на здравеопазването Божидар Финков. Става дума за 33 годишен пациент, който на 16 април е пристигнал в България след продължителен престой в Торонто, Канада, където вече са регистрирани 19 смъртни случая вследствие на тежкия остър респираторен синдром (ТОРС). Точната диагнозата обаче не може да бъде установена в България и пробите ще бъдат изпратени за изследване в Световната здравна организация (СЗО).

After stemming (left context 3, min rule frequency 2):
има първ вероят случа на атипич пневмони в ***българи****, съобщ минист на здравеопазван божидар финков. став дум за 33 годиш пациент, който на 16 април е пристигн в българи след продължител престо в торонт, канад, където вече са регистрира 19 смърт случа вследстви на теж ост респиратор синдром (торс). точ диагноз обаче не може да бъде установ в българи и проб ще бъдат изпрат за изследван в светов* ***здрав*** *организаци (сзо).*

Although we wanted to cope with the *inflectional* morphology only, the resulting stems sometimes conflate different *derivational* variants as well. E.g. *здрав* covers not only *здравна* (healthy), but also *здравен* (health). This side effect is a direct consequence of the stemming approach adopted: affix removal. This should not be regarded as a bug but rather as an advantage: a IR engine user would be happy to be presented documents containing *healthy*, in response to a query containing *health*. On the other hand, it is just a side effect: e.g. *българи* covers *България* (*Bulgaria*), but not *български* (*Bulgarian*). In fact, we cannot learn directly the morphological variants due to the structure of the dictionary used (although it may still be possible to do so *indirectly*).

context size	min frequency	rules count	COVERAGE		ERROR		
			dictionary	raw text	UNDER stemming	OVER stemming	**"Overall"**
1	1	6693	98.13%	72.18%	11.95%	27.86%	39.81%
1	2	5033	98.13%	72.16%	16.37%	24.11%	40.48%
1	5	3966	98.13%	72.16%	16.17%	23.47%	39.64%
1	10	3095	98.13%	72.16%	15.28%	20.74%	36.02%
1	**20**	**2238**	**98.11%**	**70.86%**	**13.41%**	**20.13%**	**33.54%**
2	1	30755	97.62%	62.14%	9.09%	18.57%	27.66%
2	2	22199	97.58%	61.89%	9.00%	17.93%	26.93%
2	5	14455	97.27%	60.70%	9.27%	16.71%	25.98%
2	**10**	**9528**	**96.48%**	**57.93%**	**10.40%**	**15.36%**	**25.76%**
3	1	93066	94.65%	43.76%	9.66%	12.92%	22.58%
3	**2**	**56797**	**93.25%**	**40.85%**	**10.89%**	**10.28%**	**21.17%**
3	5	26890	88.82%	35.58%	15.31%	8.15%	23.46%

Table 1. Dictionary accuracy assessment for different context size and minimum rule frequency.

4. EVALUATION

4.1. Dictionary accuracy assessment

A natural way to evaluate a stemmer is in terms of observed performance with respect to a particular task, e.g. precision/recall in the case of IR [*Hull,1996*]. Unfortunately, this kind of black box assessment is not only task-dependent, but also provides no information for the kind of errors committed (although we do it below). There are 3 different ways a stemmer can be wrong: *under-stemming*, *over-stemming* and *mis-stemming*. While the latter case is somewhat subjective, the first two can be assessed automatically. Given a fixed lemma (e.g. *отбран.ПРИ*) we measure the *local* under-stemming error as the proportion of wordforms whose stem is different from the majority stem for that lemma. The *global* under-stemming error is obtained as the average over all lemmas in the dictionary. To find the over-stemming error we calculate the average over all stems of the following quantity: for a fixed stem we count the number of distinct lemmas with at least one inflection, which is conflated to that stem. This number is then decremented by one. Another important characteristic of the stemmer performance is the rules *coverage*. We calculated this on a *per token* basis, both over the dictionary (training coverage) as well as over 12.2 MB of raw Bulgarian literature text obtained from *Slovoto* [*Slovoto*]. The results for different context sizes and minimum rule frequencies are listed in Table 1.

Category	Size	%
Agriculture&Foresty	*12*	9.45%
Culture	33	25.98%
Defence	*15*	11.81%
Sport	67	52.76%
TOTAL	***127***	***100.00%***

Table 2: Collection categories and their sizes.

We can see that using a longer left context produces more accurate results but leads also to three essential problems: 1) over-fit (may not generalise well); 2) smaller coverage (the longer the rules, the higher the probability to have no rule applicable to a particular word); and 3) no predictions are presented for words whose length is less than the context size. Note however, that the low coverage on the test set is not necessarily a problem: the coverage is calculated on *per token*, as opposed to *per type*, basis and, unlike the dictionary, the real text contains a lot of frequent proposition, conjunctions etc., and other short words that do not need to be stemmed. More detailed analysis is needed in order to determine what the real coverage should be. It would be interesting though to try a *hybrid model* that backs-up from context of size 3 to 2, and then to 1, when needed.

4.2. Text classification accuracy assessment

We performed an additional evaluation with respect to the task of text categorisation using the *k*-nearest-neighbour classifier (*kNN*) and both vector-space model and latent semantic analysis (*LSA*). For the purpose we used a collection of news articles from Bulgarian online sources containing 127 documents distributed into 4 disjoint categories (see Table 1). The collection was originally built for experiments focusing on deeper understanding of the factors influencing the LSA performance (see [*Nakov et al.,2003a*] for details), but the results gave us a good idea of the performance of *BulStem* also.

We present experiments when using raw words (as in the text), stemming and lemmatisation (manually checked) for different values of the classifier's parameters. The collection contains (types/tokens): 19,429/406,783 words, 4,487/71,879 three-context stems and 4,558/73,018 lemmas. The stop-words (from a list), numbers, non-Cyrillic symbols, words met in a single document and single-letter ones were excluded. It was also interesting to see how the stemmer performs with stop-words kept, so we run such experiments as well. The results are presented in Table 3.

LWF	GWF	LSA dim.	STOP-WORDS KEPT				STOP-WORDS REMOVED			
			raw	stem 2:1	stem 3:1	lemma	raw	stem 2:1	stem 3:1	lemma
0	**0**	**10**	78.74%	88.98%	85.04%	84.25%	92.13%	95.28%	92.13%	96.85%
0	**0**	**30**	83.46%	86.61%	88.98%	84.25%	96.85%	99.21%	100.00%	99.21%
0	**0**	**orig.**	74.80%	89.76%	91.34%	85.83%	96.06%	96.06%	96.06%	98.43%
0	**1**	**10**	76.38%	89.76%	89.76%	81.10%	96.85%	97.64%	98.43%	96.85%
0	**1**	**30**	83.46%	89.76%	88.19%	85.83%	95.28%	97.64%	98.43%	98.43%
0	**1**	**orig.**	**61.42%**	**87.40%**	**87.40%**	**85.04%**	96.06%	95.28%	96.06%	98.43%
0	**2**	**10**	55.91%	61.42%	54.33%	65.35%	92.13%	91.34%	94.49%	93.70%
0	**2**	**30**	55.91%	69.29%	64.57%	71.65%	90.55%	95.28%	97.64%	96.85%
0	**2**	**orig.**	57.48%	68.50%	68.50%	72.44%	93.70%	93.70%	98.43%	98.43%
0	**3**	**10**	95.28%	98.43%	97.64%	99.21%	97.64%	98.43%	98.43%	99.21%
0	**3**	**30**	94.49%	100.00%	100.00%	99.21%	99.21%	100.00%	100.00%	100.00%
0	**3**	**orig.**	92.13%	98.43%	98.43%	96.85%	99.21%	100.00%	100.00%	100.00%
0	**4**	**10**	89.76%	83.46%	85.83%	80.31%	92.13%	96.85%	93.70%	93.70%
0	**4**	**30**	89.76%	96.06%	91.34%	95.28%	96.85%	98.43%	97.64%	100.00%
0	**4**	**orig.**	73.23%	89.76%	91.34%	83.46%	99.21%	97.64%	96.85%	97.64%
0	**5**	**10**	97.64%	98.43%	98.43%	99.21%	96.06%	98.43%	99.21%	99.21%
0	**5**	**30**	99.21%	100.00%	100.00%	100.00%	98.43%	100.00%	100.00%	100.00%
0	**5**	**orig.**	96.85%	100.00%	100.00%	99.21%	99.21%	100.00%	100.00%	100.00%
1	**0**	**10**	96.85%	95.28%	96.06%	96.85%	94.49%	98.43%	96.85%	97.64%
1	**0**	**30**	90.55%	97.64%	98.43%	96.85%	99.21%	100.00%	99.21%	99.21%
1	**0**	**orig.**	90.55%	94.49%	96.06%	95.28%	96.06%	96.85%	98.43%	99.21%
1	**1**	**10**	92.91%	96.85%	96.85%	96.85%	96.85%	98.43%	98.43%	97.64%
1	**1**	**30**	85.83%	91.34%	92.13%	92.13%	96.06%	96.06%	96.06%	98.43%
1	**1**	**orig.**	62.99%	85.04%	81.10%	90.55%	95.28%	91.34%	92.91%	96.85%
1	**2**	**10**	84.25%	89.76%	89.76%	88.19%	93.70%	95.28%	96.06%	96.06%
1	**2**	**30**	84.25%	91.34%	89.76%	88.98%	92.13%	99.21%	99.21%	97.64%
1	**2**	**orig.**	82.68%	93.70%	95.28%	92.13%	96.85%	99.21%	98.43%	98.43%
1	**3**	**10**	97.64%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%
1	**3**	**30**	99.21%	99.21%	100.00%	100.00%	99.21%	99.21%	100.00%	100.00%
1	**3**	**orig.**	98.43%	100.00%	99.21%	99.21%	99.21%	100.00%	100.00%	100.00%
1	**4**	**10**	97.64%	96.85%	96.85%	96.85%	96.06%	97.64%	96.85%	96.85%
1	**4**	**30**	95.28%	96.85%	96.85%	96.85%	95.28%	98.43%	98.43%	97.64%
1	**4**	**orig.**	96.85%	96.85%	95.28%	96.85%	97.64%	97.64%	96.06%	97.64%
1	**5**	**10**	98.43%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%
1	**5**	**30**	99.21%	99.21%	99.21%	100.00%	99.21%	99.21%	100.00%	99.21%
1	**5**	**orig.**	98.43%	100.00%	100.00%	100.00%	99.21%	100.00%	100.00%	100.00%
AVERAGE			**86.33%**	**92.19%**	**91.73%**	**91.51%**	**96.46%**	**97.68%**	**97.86%**	**98.27%**

Table 3. Text classification accuracy: raw, stemming and lemmatisation.

We performed a stratified 20-fold cross-validation. We chose 20 since we did not want to lose too much data for training. An alternative would be to follow a leave-one-out strategy: train on 126 documents and test on the remaining one. But, since we remove a document from one class only, this class would suffer, unlike the rest, and the results will not be a good approximation of the real performance. We decided that 20 is a good compromise between the need to

LWF	GWF	SVD	1:1	1:2	1:5	1:10	1:20	2:1	2:2	2:5	2:10	3:1	3:2	3:3
0	**0**	**10**	96.85%	92.91%	92.91%	95.28%	93.70%	88.98%	93.70%	92.91%	92.91%	85.04%	92.91%	92.91%
0	**0**	**30**	99.21%	99.21%	98.43%	98.43%	99.21%	86.61%	93.70%	99.21%	99.21%	88.98%	100.00%	99.21%
0	**0**	**orig.**	96.06%	94.49%	95.28%	96.06%	96.06%	89.76%	85.83%	97.64%	98.43%	91.34%	95.28%	95.28%
0	**1**	**10**	98.43%	98.43%	98.43%	98.43%	98.43%	89.76%	94.49%	99.21%	98.43%	89.76%	97.64%	97.64%
0	**1**	**30**	96.85%	95.28%	96.06%	97.64%	97.64%	89.76%	94.49%	97.64%	97.64%	88.19%	96.85%	97.64%
0	**1**	**orig.**	93.70%	92.91%	94.49%	96.06%	96.06%	87.40%	81.10%	97.64%	98.43%	87.40%	96.06%	95.28%
0	**2**	**10**	92.13%	92.91%	95.28%	93.70%	94.49%	61.42%	71.65%	94.49%	92.13%	54.33%	93.70%	93.70%
0	**2**	**30**	88.98%	97.64%	96.85%	96.85%	97.64%	69.29%	81.89%	98.43%	97.64%	64.57%	98.43%	98.43%
0	**2**	**orig.**	90.55%	91.34%	94.49%	96.06%	98.43%	68.50%	74.80%	99.21%	96.85%	68.50%	98.43%	96.85%
0	**3**	**10**	98.43%	98.43%	98.43%	98.43%	98.43%	98.43%	98.43%	98.43%	98.43%	97.64%	98.43%	98.43%
0	**3**	**30**	100.00%	100.00%	99.21%	100.00%	100.00%	100.00%	100.00%	99.21%	100.00%	100.00%	100.00%	100.00%
0	**3**	**orig.**	100.00%	100.00%	100.00%	100.00%	100.00%	98.43%	99.21%	100.00%	100.00%	98.43%	100.00%	100.00%
0	**4**	**10**	92.91%	96.06%	96.06%	96.85%	92.91%	83.46%	94.49%	95.28%	94.49%	85.83%	92.13%	92.91%
0	**4**	**30**	98.43%	98.43%	98.43%	98.43%	98.43%	96.06%	97.64%	97.64%	96.85%	91.34%	99.21%	98.43%
0	**4**	**orig.**	96.85%	96.85%	96.85%	98.43%	98.43%	89.76%	91.34%	97.64%	96.85%	91.34%	97.64%	97.64%
0	**5**	**10**	99.21%	98.43%	99.21%	99.21%	99.21%	98.43%	98.43%	97.64%	98.43%	98.43%	98.43%	98.43%
0	**5**	**30**	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.21%	100.00%	100.00%	100.00%	100.00%
0	**5**	**orig.**	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
1	**0**	**10**	97.64%	97.64%	97.64%	97.64%	97.64%	95.28%	96.06%	97.64%	98.43%	96.06%	97.64%	96.06%
1	**0**	**30**	98.43%	99.21%	99.21%	100.00%	100.00%	97.64%	97.64%	98.43%	98.43%	98.43%	98.43%	98.43%
1	**0**	**orig.**	97.64%	96.85%	97.64%	96.85%	97.64%	94.49%	98.43%	96.85%	97.64%	96.06%	96.85%	96.06%
1	**1**	**10**	97.64%	98.43%	98.43%	98.43%	98.43%	96.85%	97.64%	97.64%	97.64%	96.85%	98.43%	98.43%
1	**1**	**30**	96.06%	95.28%	96.06%	96.06%	96.06%	91.34%	94.49%	96.06%	94.49%	92.13%	96.06%	96.85%
1	**1**	**orig.**	92.91%	90.55%	92.13%	91.34%	91.34%	85.04%	92.91%	90.55%	90.55%	81.10%	89.76%	89.76%
1	**2**	**10**	96.06%	95.28%	95.28%	96.06%	95.28%	89.76%	93.70%	96.06%	95.28%	89.76%	96.06%	94.49%
1	**2**	**30**	97.64%	99.21%	100.00%	99.21%	100.00%	91.34%	95.28%	100.00%	100.00%	89.76%	100.00%	100.00%
1	**2**	**orig.**	99.21%	97.64%	99.21%	97.64%	98.43%	93.70%	89.76%	99.21%	99.21%	95.28%	99.21%	99.21%
1	**3**	**10**	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	98.43%
1	**3**	**30**	99.21%	99.21%	99.21%	99.21%	100.00%	99.21%	100.00%	99.21%	98.43%	100.00%	100.00%	99.21%
1	**3**	**orig.**	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.21%	100.00%	100.00%
1	**4**	**10**	96.85%	98.43%	98.43%	98.43%	97.64%	96.85%	96.85%	96.85%	96.85%	96.85%	96.85%	96.85%
1	**4**	**30**	97.64%	97.64%	99.21%	98.43%	98.43%	96.85%	96.85%	98.43%	97.64%	96.85%	98.43%	97.64%
1	**4**	**orig.**	99.21%	97.64%	97.64%	97.64%	97.64%	96.85%	96.85%	96.85%	96.85%	95.28%	96.06%	96.85%
1	**5**	**10**	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%
1	**5**	**30**	99.21%	99.21%	99.21%	99.21%	99.21%	99.21%	100.00%	99.21%	99.21%	99.21%	99.21%	99.21%
1	**5**	**orig.**	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
AVERAGE (above)			**97.29%**	**97.33%**	**97.73%**	**97.90%**	**97.92%**	**92.19%**	**94.34%**	**97.86%**	**97.66%**	**91.73%**	**97.68%**	**97.49%**
ERROR (Table 1)			***39.81%***	***40.48%***	***39.64%***	***36.02%***	***33.54%***	***27.66%***	***26.93%***	***25.98%***	***25.76%***	***22.58%***	***21.17%***	***23.46%***
UNDER (Table 1)			***11.95%***	***16.37%***	***16.17%***	***15.28%***	***13.41%***	***9.09%***	***9.00%***	***9.27%***	***10.40%***	***9.66%***	***10.89%***	***15.31%***
OVER (Table 1)			***27.86%***	***24.11%***	***23.47%***	***20.74%***	***20.13%***	***18.57%***	***17.93%***	***16.71%***	***15.36%***	***12.92%***	***10.28%***	***8.15%***

Table 4. Text classification: stemming parameters evaluation (no stop-words, Table 1 references listed).

model to some extent the original distribution and to waste for testing as less documents as possible.

Our previous research show the choice of weighting applied prior to SVD can have a dramatic impact on the further performance (see [*Nakov et al.,2001*]). The weighting scheme can be expressed as a product of two numbers *local* and *global weight functions* (LWF and GWF). LWF $L(i,j)$ represents the weight of term i in document j, while GWF $G(i)$ expresses the weight of term i across the entire document set. Some of the most popular weighting schemes for LSA follow:

LWF = 0: $L(i,j) = X_{ij}$ — *term frequency*
LWF = 1: $L(i,j) = \log(1+X_{ij})$ — *logarithm*
GWF = 0: $G(i)=1$ — *trivial*
GWF = 1: $G(i) = 1/\sqrt{\sum_j L(i,j)^2}$ - *normalised*
GWF = 2: $G(i) = g(i) / d(i)$ — *GfIdf*
GWF = 3: $G(i) = 1+ \log(N / d(i))$ — *Idf*
GWF = 4: $G(i) = -1 / (\Sigma_j\ p(i,j) \log p(i,j))$
GWF=5: $G(i)=1/(1+\{\Sigma_j p(i,j)\log p(i,j)\}/\log N)$
where

N is the training documents count;

$g(i)$ is term frequency of i across all documents;

$d(i)$ is the number of documents containing i.

Both GWF=4 and GWF=5 represent some kind of entropy. The results of the evaluation for *k*NN with *k*=1 (i.e. the class is determined from the top candidate only) are summarised in Table 3, which shows the micro-average accuracy over the 20 cross-validation runs. The first two columns contain the LWF and GWF, as described above. Column 3 shows the LSA space dimensionality (*orig.* means: no dimensionality reduction, i.e. the *vector-space model*). We tried different space dimensionalities: 10, 20, 30, 40 and 50, but here present the results for 10 and 30 only in order to save space. The following columns represent raw text, stemming with context size 2, stemming with context size 3 (both with min rule frequency of 1) and finally – lemmatisation. All these are shown for the case with and without stop-words. In addition, we added an extra row with the average accuracy for each column.

Table 4 focuses on the stemmer performance for the different values for the context size and for the minimum rule frequency (without stop-words). Again, we calculated the average accuracy for each column to make the comparison easier. In addition, we list the under-stemming and over-stemming results as well as the overall error from Table 1.

6. DISCUSSION

As we can see from Table 3, stemming and lemmatisation are almost equally good for the highly inflectional Bulgarian and generally (but not systematically) outperform the raw text (up to 26% difference). The average from the last row suggests stemming might be slightly better than lemmatisation, if we keep the stop-words, and slightly worse, if we remove them. This is an interesting result given the fact that *BulStem* is fully automatic, while the lemmatisation has been verified by hand (we wanted to make sure it was really done correctly). On the other hand, Table 3 shows the choice of weighting scheme looks more important compared to stemming/-lemmatisation (also to stop-words removal and LSA dimensionality reduction).

In addition, Table 4 shows, the stemming parameters used in Table 3 (2:1 and 3:1) were in fact among the worse ones. But even if we had chosen the best ones, the lemmatisation would still be better (although by about .37% only).

We see in Table 4 that the measures of under-stemming, over-stemming and their sum are unreliable predictors of the actual system accuracy for text classification. This is due mostly to the very different evaluation objective, but also in part to the source used: the wordforms distribution in the dictionary is different from the one in a real text. Another explanation involves the important problem of morphological root transformations during inflections production in Bulgarian (and in many other Slavonic languages). In the examples above some forms of the verb *отбера* lose the internal "*е*": e.g. *отбра*, *отбрал*, *отбрала* etc. This forces us to produce *отб*, which is obviously an over-stemming and can produce potential problems since there are other words sharing that stem. It would be interesting to try to adapt the stemmer to learn such regularities and make use of them when needed.

Note however that sometimes the conflation of unrelated wordforms cannot be escaped. Going back to *отбран* (adjective) and *отбрана* (noun), we can see that probably the best stem for *both* is *отбран*. Unfortunately, the surface wordform *отбрана* represents two homographs (*defense*/noun; *selected*/adjective inflected from *отбран*). Anyway, there is no way around: no stemmer with the same definition of *stem* can hope to do any better unless the task is redefined as *lemmatisation*. But then we face another problem: the form *отбраната* is shared by all the three lemmas. The best way to disambiguate it is by means of POS tagger but then a full morphological analysis would also be applicable, and it would not be stemming any more.

BulStem is available on the Web at: http://www.cs.berkeley.edu/~nakov/bulstem/

7. FUTURE WORK

We would like to study in more detail mis-stemming. It would be interesting to implement an algorithm following Porter that relies on a limited number of hand-crafted rules [*Porter, 1980*], as well as a dictionary-based one following Krovetz [*Krovetz,1993*], and compare them to *BulStem*. We are also curious to experiment with the Xu & Croft [*Xu&Croft,1998*] stemmer and see how it performs when used on the output of different aggressive stemmers e.g. Porter, successor variety, *n*-gram etc. Of course the comparison would be harder because of the various definition of stem used by the different algorithms but could still be performed, by applying them to some particular task, e.g. IR, document classification etc., and measuring the precision/recall.

ACKNOWLEDGEMENTS

Special thanks are due to Elena Paskaleva for providing the Bulgarian dictionary and to Elena Valchanova, for the corpus annotation. Also to Nianjun Liu for pointing some typos.

REFERENCES

[*Adamson&Borehan,1974*] Adamson G., J. Boreham. The use of an association measure based on character structure to identify semantically related pairs of words and document titles. In Information Processing & Management. vol. 10(7/8), pp. 253-260, 1974.

[*Dawson,1974*] Dawson J. Suffix removal for word conflation. In Bulletin of the Association for Literary & Linguistic Computing. vol. 2(3), pp. 33-46, 1974.

[*Frakes,1982*] Frakes W. Term Conflation for Information Retrieval. PhD. dissertation. Syracuse University, 1982.

[*Hafer&Weiss,1974*] Hafer M., S. Weiss. Word segmentation by letter successor varieties. In Information Processing & Management. vol. 10(11/12), pp. 371-386, 1974.

[*Harman,1991*] Harman D. How effective is suffixing? In Journal of The American Society of Information Science. Vol. 42, No 1. pp. 7-15. 1991.

[*Hull,1996*] Hull D. Stemming Algorithms: A Case study for detailed evaluation. In Journal of The American Society of Information Science. Vol. 47, No 1. pp. 70-84. 1996.

[*Kovalenko,2002*] Kovalenko A. Stemka: Morphological analyser for small search systems. In System Administrator Magazine. Moscow, October 2002.

[*Kraaij&Pohlmann,1994*] Kraaij W., R. Pohlmann. Porter's stemming algorithm for Dutch. In Noordman LGM and de Vroomen WAM, eds. Informatiewetenschap 1994: Wetenschappelijke bijdragen aan de derde STINFON Conferentie. Tilburg, pp. 167-180, 1994.

[*Kraaij,1996*] Kraaij W. Viewing stemming as recall enhancement. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 40-48. New York, 1996.

[*Krovetz,1993*] Krovetz R. Viewing Morphology as an Inference Process. Proc. 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 191-202. New York, 1993.

[*Kukushkina&Polikarpov,1996*] Kukushkina O., A. Polikarpov. DicTUM-1: A system for dictionary-text universal manipulations and analysis. In Proc. of XI International Conference "History and Computing". Moscow, 1996.

[*Lovins,1968*] Lovins J. Development of a stemming algorithm. Mech. Trans. And Comp. Ling. 11., pp. 22-31. 1968.

[*Nakov et al.,2003a*] Nakov P., E. Valchanova, G. Angelova. Towards Deeper Understanding of LSA Performance. In Proc. Recent Advances in Natural Language Processing. pp. 311-318, Borovetz, Bulgaria, 2003.

[*Nakov et al.,2003b*] Nakov P., Y. Bonev, G. Angelova, E. Cius, W. von Hahn. Guessing Morphological Classes of Unknown German Nouns. In Proc. Recent Advances in Natural Language Processing. pp. 319-326, Borovetz, Bulgaria, 2003.

[*Nakov et al.,2001*] Nakov P., Popova A., Mateev P. Weight functions impact on LSA performance. EuroConference RANLP'2001 (Recent Advances in NLP). pp. 187-193. Tzigov Chark, Bulgaria, 2001.

[*Paice,1990*] Paice C. Another stemmer. In Proc. of SIGIR Forum, vol. 24(3), pp. 56-61, 1990.

[*Popovic&Willett,1992*] Popovic M., P. Willett. The Effectiveness of Stemming for Natural Language access to Slovene Textual Data. In Journal of The American Society of Information Science. Vol. 43, No 5. pp. 384-390, 1992.

[*Porter,1980*] Porter M. An algorithm for suffix stripping. Program 14, 3. pp. 130–137, 1980.

[*Savoy,1993*] Savoy J. Stemming of French words based on grammatical categories. In Journal of the American Society for Information Science. vol. 44(1), pp. 1-9, 1993.

[*Schinke et al.,1996*] Schinke R., M. Greengrass, A. Robertson, P. Willett. A stemming algorithm for Latin text databases. In Journal of Documentation. vol. 52, pp.172-187, 1996.

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

[*Slovoto*] Slovoto, http://slovoto.orbitel.bg

[*Snowball*] Snowball: http://snowball.tartarus.org

[*Xu&Croft,1998*] Xu J., B. Croft. Corpus Based Stemming Using Coocurrence of Word Variants. In ACM Transactions on Information Systems. vol. 16, No 1. pp. 61-81. 1998.