An extended spell checker for unknown words

Balázs Indig
(Supervisor: Dr. Gábor Proszéky)
indig.balazs@itk.ppke.hu

***Abstract*—Spell checking is considered a solved problem, but with the rapid development of the natural language processing the new results are slowly extending the means of spell checking towards grammar checking. In this article I review some of the spell checking error classes in a broader sense, the related problems, their state-of-the-art solutions and their different nature on different types of languages (English and Hungarian), arguing that these methods are insufficient for some language classes. Finally, I present my own method of batch spell checking in large volumes of coherent text.**

***Keywords*-spellchecking; context-sensitive; batch-correction**

I. Introduction

Tools called "spell checkers" are widely used in current word processing systems as an error correcting tool. By the rapid changing of the Internet and computers, the current spell checking is gaining an increasing importance in our lives by the growing capacity of computers, because of the increasing number of ways and volumes content created. Traditionally, spell checkers did subsequent word-by-word analysis, and then transferred to do the analysis while typing. This made it possible for spell checkers to have significance beyond word processors. Nowadays spell checkers can be found everywhere from web browsers to e-mail clients and people use them actively. As in the beginning, today as well the basic principle is the word-by-word analysis, thus the spell checking procedure is stuck at word level. Developers in the IT industry concentrate on these local tools, for example the increasingly better support of agglutinative languages and word compounding appeared approximately 5-6 years ago[1], and in the meantime dictionaries follow the changes of individual languages (by adding new words). Meanwhile, in the field of Natural Language Processing things are developing rapidly as well, but these novel approaches have rarely been applied in spell checking systems yet. A 10 million word English corpus has less than 100,000 different word forms, a corpus of the same size for Hungarian contains well over 800,000[2]. While an open class English word has about 46 different word forms, it has several hundred or thousand different productively suffixed forms in agglutinating languages[3]. The standard tools, which have been proven good in English cannot be applied without any modification. In the literature there exist a lot of separate algorithms that have proven good for partial problems in the English language. I am going to review these state-of-the-art methods and I am going to argue that they cannot be applied because of the nature of the Hungarian language. I will describe my paradigm of spell checking in detail.

All of the aforementioned methods have something in common. They are working with a larger volume of texts. I will set another constraint: I will suppose that all the texts which are examined are coherent. So I can rely on the text-level information, which lies in the text to be extracted, examined and used to improve spell checking performance.
I want to show that spelling errors can be widely different. One must classify these errors and make special sub-solutions for each class to locate and correct most of the errors found in current Hungarian texts with the lowest false positive rate as possible.

II. Types of spelling errors

The academic Hungarinan spelling rules are very complex. They involve semantic features like substance names, occupation names, etc. and the way one should imagine the word: e.g. "légikísérő" is written in one word because the word "kísérő" is in the air physically and not figuratively. The rough listing of the types of errors is as follows:

- in-word errors: One take a word, and modify it by edit distance (e.g. the so called Damerau-Levenshtein distance[4][5]), so the word does not become some other valid word. This is the oldest error observed and most of the errors in English can be corrected by searching the word no more than one distance from the erroneous form. The English language is so sparse that there are only a few candidates. In Hungarian this type of error has not been a problem for a long time. There are several models for this type of errors (e.g. the Noisy Channel Model[6]), but the rate of these errors is much lower then in English.
- real-word errors: One take a word, and modify it, so the modified word becomes a valid meaningful word that has nothing to do with its context. For example: "He had lots of *honey* (money), he wanted to buy a bigger house." These errors must be approached differently. If one knows that the writer has a specific mother tongue and English is his second language one can collect statistical information about the typical misspellings and use them to correct errors [7]. In this type one must distinguish between the words that changed their word species and those which did not. (e.g. money $\rightarrow$ honey, defuse $\rightarrow$ diffuse) In Hungarian there are more word species, so there are more errors of this type.
- word compounding errors: One take two words, and write them as one or take a compound word and write it in two words. The real problem is that the former can be detected and corrected at word level, but the latter cannot.

B. Indig, "An extended spell checker for unknown words,"
in *Proceedings of the Interdisciplinary Doctoral School in the 2012-2013 Academic Year*, T. Roska, G. Prószéky, P. Szolgay, Eds.
Faculty of Information Technology, Pázmány Péter Catholic University.
Budapest, Hungary: Pázmány University ePress, 2013, vol. 8, pp. 29-32.

The Hungarian Academy rules are so complex in this case that in Hungarian a lot of errors fall into this class.
- Out of Vocabulary (OOV) errors: The traditional spell checkers work with a list of words or the list of stems and the production rules (these two are together called lexicon), but there are open word-classes and the spell checker must distinguish between the unknown or OOV words and the misspelled ones. Not to mention the right and consistent use of these words. This can only be detected in a larger volume of coherent text.
- punctuation errors: The right punctuation in the text is not closely related to spell checking, but helps people and the programs to interpret the written text. And can be checked and corrected with the same tool-set as the aforementioned error classes.
- grammar errors: These kind of errors cannot be clearly separated from the cases mentioned above, so I list this class here.

A. *How Hungarian and English differ*

There are several tools that work language independently, but the most important resources are language dependent. With the help of the self-developed tools in the MTA-PPKE-NLPG research group I can split any raw text to sentences and tokenize it[8]. I can recognize named-entities for future use[9]. Then with the POS-tagger I can couple every word with a tag that reflects its distributional preferences and therefore can classify them into groups[10]. The number of the groups vary from language to language. For example, in English there are only 36 and in Hungarian there are more than 1000 word class tags[11][12]. This makes the task much harder for Hungarian, and the problem becomes even worse when one restricts the domain to clinical texts[13]. As Hungarian is a highly inflected language there are many word forms that belong to the same stem. And there are many homonyms as well, so all in all it is far less sparse than English. Therefore the error types mentioned above cannot be corrected by word-level easily. One can apply Machine Learning methods for extracting features from the context and make decisions, but the liberal word ordering of the Hungarian language makes this task ineffective.

III. Methods in the literature

The current state-of-the-art methods approaching different parts of the whole spell checking. I will list some techniques and argue that they cannot work in Hungarian.

- Take the function words and record their contextual features, because subsequent function words can identify what should come after them and that can be checked for validity[14]. This technique has been successfully applied for the German language on compound words and punctuations. In Hungarian the function words can be omitted and therefore this method cannot achieve much success.
- Make a confusion set of the common misspellings and their right forms[15]. This method can be successfully applied for accenting and word-sense disambiguation. But only on languages that are not inflected and have few word forms. In Hungarian the morphological production rules can be theoretically infinite, and the resources are not available. If the right resource existed, then still one would face the sparse data problem. This highlights other problems: for example, to use stop words or not, and when to use the real word form over the distributional tag. It is desired to automatically choose the right candidate suggestion, but the sufficient features cannot be retrieved from the text because of data sparsity. One way to help this is to rank the suggestions by weighing the edit distance[16].
- One can approach by defining a hash function that collide only on the misspelled and right spelled words and therefore one gets automatically the correct word form for the misspelled word[17][18]. This method can only work if one has a list of misspelled words and the correct forms to train the hash function to work as expected.

IV. My own method

Text corpora forms a consistent closely related text in one topic. That information can be used. I am trying to reduce the number of false positive results of traditional spell checking algorithms. At the same time I want to collect information of the new words and make their usage more consistent[1] by the interaction of the user. I also want to reduce the time consumed by the proofreading of the text by classifying the spelling errors by the stems and guessed production paradigms, so the user does not have to correct every occurrence of the same misspelling (or those which belong to the same stem) one-by-one[19]. This method would stay at word level, but will not be restricted to a fixed lexicon that is integrated into the spell checking programs. I use all of our tools in pipeline and make statistical inferences from the decorated text.[20]

A. *Statistical methods on the decorated text*

The text was split into sentences and tokens, then I added the POS-tag and lemma for every token with the information of the candidate lemma-tag couples. I also added the information, whether a token is recognised as a correct word form or not. Then I examined the following features of the tokens:

- the frequency of each word form
- the frequency of lemmas of the incorrect words
- the combination of the above

While examining word forms classified by their lemmas, one can find features that characterize the Hungarian morphological production system, which is hard-coded in the morphological analyser[21] for the fixed list of words. If one can find a sufficient number and quality of word forms one can construct an inflectional paradigm that makes a good point to examine the less frequent words against. If these words meet the expectations of their lemma's paradigm, then they

[1] as the program has no information about how the different forms of these words should be spelled

are considered good, otherwise they are considered misspelled and the user is asked to decide. The paradigm also helps to generate suggestions of the misspelled word. They come from the paradigm and it is not necessary for them to appear in the text. The possibility of automatically correcting these words becomes available. There is a threshold that must be set in order to distinguish between low frequency misspelled words and the ones that are too frequent to be misspelled. This threshold can be set safely between 3-5. As the non-systematic misspellings are so diverse that there cannot be such coincidence. The systematic misspellings are considered to be right as the program does not have any external information of the text. Just helps to increase the consistence of the text. The words that are above the threshold are considered "certainly good", the others need to be checked with the extended spell checker. From "certainly good", frequent word forms and their lemmas, the program generates the paradigms. With that, the program checks the other "possibly misspelled" words. The traditional spell checkers' engines can be extended to accept the new words and generate an inflectional paradigm to work with. This can save a lot of time and effort as generating the suggestions is not a trivial task. The classified word forms with their accompanying suggestions can be displayed to the user at once and he can accept or decline the suggestions for each occurrence by examining the context of the word without even proofreading the whole document, just looking at the critical parts of the text if it is necessary. To apply the changes at once the program must map the corrected text to the original one. This could be done for example by Dynamic Time Warping (DTW)[22]. By finding anchors in the text and make the two versions parallel. This could be very useful on environments with special formatted texts, where the formatting is destroyed during the preprocessing steps.

B. Adapting POS-tagger to the text with a posteriori information

The tokenized text is passed to the POS-tagger, to couple each word with its stem, tag and the possible other candidates. For the known words this task is easy. The morphology module can help the tagger, but when it comes to the new words, that are not known either by the morphology module or by the POS-tagger the number of candidates can grow from one up to ten. These candidates mostly differ in the lemmas of the words. The statistical module tries to guess the appropriate lemmas. But this module does not care for the words seen previously. Guessing is totally local to the word in the text. No context is taken into account, but the information is lying in the text. Therefore, after the preprocessing task my program selects the lemmas of the unknown words (choosing also from the candidates) in the text which are frequent enough to not being noise (see table II). I feed these selected lemmas to the POS-tagger. In another pass the POS-tagger selects the fed lemma from the candidates unconditionally if he can. This method can be repeated and all the repetitions improve the performance of the guesser for the current text to a level and decrease the number of the candidates which the POS-tagger chooses from. (There can still be more candidate tags for the same lemma.)

V. Results

The efficiency of the method was tested on two corpora (table I). One is a book (Orwell: 1984) full of theoretically good, but self-invented words. Some of these words are not known by the spell checker but those words are in control. The other is taken from the Internet, contains newspaper articles from a specific site. The size of the two corpora is almost identical. The language model is taken from Szeged corpus 2.0 [12]. The table shows two stages before and after the following heuristic filtering: I filtered out the tokens that were definitely some affix or were not containing four alphabetic letters beside each other (table I). With this filtering, I hope that the real words come into view. Later, I worked with these set of tokens.

TABLE I
THE STATISTICS OF THE USED CORPORA

	1984		*Articles*	
Filtering:	*before*	*after*	*before*	*after*
Tokens:	99913	50586	74053	40716
Tokens (unique):	20393	18211	20916	18465
Not known by Humor:	301	283	1431	1224
Not known by Humor (unique):	181	168	1029	886

TABLE II
EXAMPLE OF WORD FORM FREQUENCIES

word form	**frequency**	**stem**
Obama	40	Obama
Obamaáról	1	Obamaá
Obamák	1	Obamá
Obama-kormány	1	Obama-kormány
Obamának	3	Obam
Obamának	3	Obamá
Obamára	1	Obamá
Obamáról	3	Obam
Obamáról	3	Obamá
Obamát	5	Obam
Obamát	5	Obamát
Obamával	1	Obamával

As seen in table III, there were many words that were found to be good and with the traditional spell checking methods would become false positives. There were word forms above the threshold and these were selected to be the base of the inflection paradigm for other flexed form of the same stem (see table IV). Finally, the remaining words were considered to be misspellings and suggestions were generated (see table V). In table V one can see the faults of the trivial suggestion generation algorithm. This can be vastly improved by using the engine of some traditional spell checker program.

VI. Conclusion

The described method can correct a wider class of the aforementioned misspellings than the traditional spell checkers. This initial phase of the research shows that with my new method the entire proofreading process becomes simpler and faster as the size of the text grows. The amount of text processed per unit of time clearly increases.

TABLE III
RESULTS

	1984	*Articles*
Stems altered:	34	65
Stems altered (unique):	19	48
Frequent stems:	14	55
Frequent word forms:	40	51
Inflection paradigms:	17	58
Suggestions (for new words):	3	8

TABLE IV
GOOD INFLECTION PARADIGMS

1984	
Stem	
beszélír	
Good form	**Rare form**
beszélírba	beszélírja
beszélírral	beszélírtól
beszélír	
beszélírt	

Articles	
Stem	
Obama	
Good form	**Rare form**
Obamának	Obamáék
Obamáról	Obamára
Obamát	Obamával
Obama	

TABLE V
SUGGESTIONS

Articles	
Misspelled word	**Suggestion**
BruxInfo	Bruxinfo
Gingrics	Gingrich
Mtelekom	MTelekom
Obamaáról	Obamáról
Osama	Obama
Sandber	Sandberg
stent	sztent
Unicredit	UniCredit

1984	
Misspelled word	**Suggestion**
aszondom	Aszondom
beszélírja	beszélírba
jógondoló	jógondol

VII. FUTURE WORK

The method is currently not able to make corrections automatically, but beside this the other paths of future research are:

- extending the spell checker program's lexicon efficiently
- building a misspelling dictionary
- making collaborated spell checking and correction easier with shared lexica
- rapid domain adaptation

These workflows are quite demanding today, with my proposed method it becomes much easier.

ACKNOWLEDGMENT

I would like to thank my Professor and Colleagues for their help.

REFERENCES

[1] N. László. (2005, Jul.) Hunspell, hungarian spell checker. [Online]. Available: http://sourceforge.net/projects/hunspell/

[2] C. Oravecz and P. Dienes, "Efficient stochastic part-of-speech tagging for hungarian," in *In Proc. of the Third LREC, Las Palmas, Espanha*, 2002, p. 710717.

[3] O. György and N. Attila, "Purepos – an open source morphological disambiguator," in *Proceedings of the 9th International Workshop on Natural Language Processing and Cognitive Science*, 2012.

[4] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, no. 3, pp. 171–176, Mar. 1964. [Online]. Available: http://doi.acm.org/10.1145/363958.363994

[5] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals." *Soviet Physics Doklady.*, vol. 10, no. 8, pp. 707–710, Feb. 1966.

[6] M. D. Kernighan, K. W. Church, and W. A. Gale, "A spelling correction program based on a noisy channel model," in *Proceedings of the 13th conference on Computational linguistics - Volume 2*, ser. COLING '90. Stroudsburg, PA, USA: Association for Computational Linguistics, 1990, pp. 205–210. [Online]. Available: http://dx.doi.org/10.3115/997939.997975

[7] A. Rozovskaya and D. Roth, "Generating confusion sets for context-sensitive error correction," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 961–970. [Online]. Available: http://dl.acm.org/citation.cfm?id=1870658.1870752

[8] B. Indig, "Puretoken: egy új tokenizáló eszköz." Szeged: Szegedi Egyetem, 01/2013 2013.

[9] R. Farkas, G. Szarvas, and R. Ormándi, "Improving a state-of-the-art named entity recognition system using the world wide web," in *Proceedings of the 7th industrial conference on Advances in data mining: theoretical aspects and applications*, ser. ICDM'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 163–172. [Online]. Available: http://dl.acm.org/citation.cfm?id=1770770.1770787

[10] A. Novák, G. Orosz, and B. Indig, "Javában taggelünk," Szegedi Egyetem. Szeged: Szegedi Egyetem, 12/2011 2011.

[11] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *COMPUTATIONAL LINGUISTICS*, vol. 19, no. 2, pp. 313–330, 1993.

[12] D. Csendes, J. Csirik, and T. Gyimthy, "The szeged corpus: A pos tagged and syntactically annotated hungarian natural language corpus." in *TSD*, ser. Lecture Notes in Computer Science, P. Sojka, I. Kopecek, and K. Pala, Eds., vol. 3206. Springer, 2004, pp. 41–48. [Online]. Available: http://dblp.uni-trier.de/db/conf/tsd/tsd2004.html#CsendesCG04

[13] B. Stomach and V. Hit, "Novel applications of the stomach-hit algorithm," *Commun. ACM*, vol. 8, no. 13, pp. 1687–1693, Apr. 1987. [Online]. Available: http://doi.acm.org/14.1343/345538.356446

[14] R. Kese, F. Dudda, G. Heyer, and M. Kugler, "Extended spelling correction for german," in *Proceedings of the Third Conference on Applied Natural Language Processing*. Trento, Italy: Association for Computational Linguistics, March 1992, pp. 126–132. [Online]. Available: http://www.aclweb.org/anthology/A92-1017

[15] M. P. Jones and J. H. Martin, "Contextual spelling correction using latent semantic analysis," in *Proceedings of the fifth conference on Applied natural language processing*, ser. ANLC '97. Stroudsburg, PA, USA: Association for Computational Linguistics, 1997, pp. 166–173. [Online]. Available: http://dx.doi.org/10.3115/974557.974582

[16] M. A. Elmi and M. Evens, "Spelling correction using context," in *In Proceedings of COLING/ACL 98*. Morgan Kaufmann Publishers, 1998, pp. 360–364.

[17] M. Reynaert, "Text-Induced Spelling Correction," Ph.D. dissertation, Tilburg University, Tilburg, The Netherlands, 2005. [Online]. Available: http://ilk.uvt.nl/~mre/TISC.PhD.MartinReynaert.pdf.gz

[18] ——, "Text induced spelling correction," in *Proceedings of the 20th international conference on Computational Linguistics*, ser. COLING '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004. [Online]. Available: http://dx.doi.org/10.3115/1220355.1220475

[19] B. Indig and G. Prószéky, "Ismeretlen szavak helyes kezelése kötegelt helyesírás-ellenőrző programmal." Szeged: Szegedi Egyetem, 01/2013 2013.

[20] G. Prószéky and B. Kis, "A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ser. ACL '99. Stroudsburg, PA, USA: Association for Computational Linguistics, 1999, pp. 261–268. [Online]. Available: http://dx.doi.org/10.3115/1034678.1034723

[21] A. Novák and T. M. Pintér, "Milyen a még jobb humor?" Szegedi Egyetem. Szeged: Szegedi Egyetem, 12/2006 2006.

[22] R. Bellman and R. Kalaba, "On adaptive control processes," *Automatic Control, IRE Transactions on*, vol. 4, no. 2, pp. 1–9, Nov. 1959. [Online]. Available: http://dx.doi.org/10.1109/tac.1959.1104847