Towards the New Czech Grammar-checker

Vojtěch Mrkývka

Faculty of Arts, Masaryk University
Arne Nováka 1, 602 00 Brno, Czech Republic
mrkyvka@phil.muni.cz

Abstract. I created a basis for the new grammar-checker of Czech. This was positively accepted by the committee and I was allowed to continue its development in my further study. In this paper, I want to describe the proximate issues of its active development.

Keywords: spell checker; grammar checker; text analysis; correction

1 Introduction

In September 2018 I've published the first version of the new grammar-checker of Czech (see Figure 1). I was motivated by the fact that the presumably best current option is part of the proprietary software. Additionally, it doesn't provide satisfactory results. Limitations of it can be discovered by comparing functionality described in the article *Kontrola české gramatiky (český grammar checker)* by Vladimír Petkevič and the status quo provided by the most recent versions of the same software, where some of the described features are no longer available. [1]

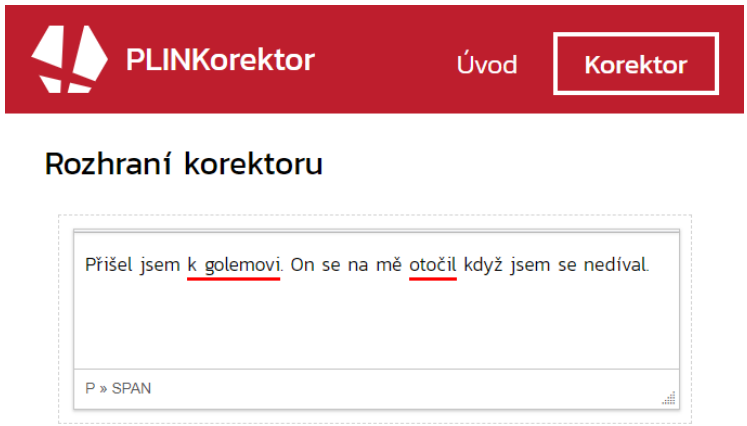

Fig. 1: Current version of the new grammar-checker. The red underlines depict errors.

Aleš Horák, Pavel Rychlý, Adam Rambousek (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2018, pp. 3–8, 2018. © Tribun EU 2018

2 The Current Version

2.1 The Corrector Interface

For the corrector to be truly open to use, I developed an online application using widely spread text processor TinyMCE v4 and provided different correction mechanisms as separate modules. [2] Due to its JavaScript nature, I decided to make it asynchronous as faster modules wouldn't need to wait for the slower ones. Obviously, in some cases, there were necessary dependencies. It was likely that tasks such as tokenisation or lemmatisation would be required by multiple correction modules, so it was more than convenient to perform these tasks separately (see Figure 2). The approach could be seen on the final render, where corrections made by faster modules are also displayed sooner in the text processor window. To push the speed even further, I separated the processing to individual paragraphs, where the process of grammar-checking is repeated only on one(s), which were modified.

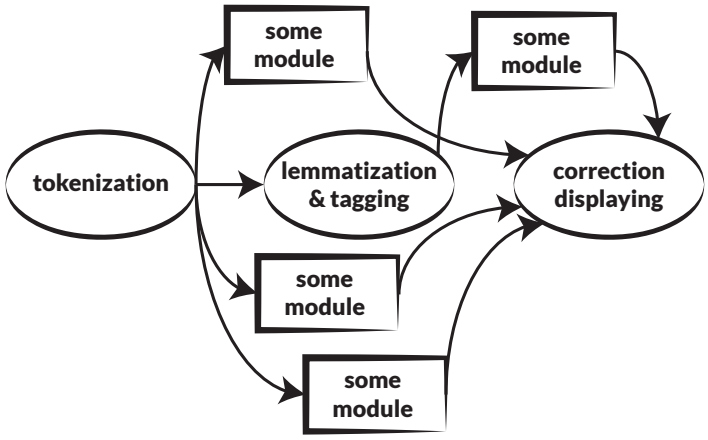

Fig. 2: Visualisation of the new corrector's asynchronous nature.

2.2 Implemented modules

Although the current version does not achieve qualities of its competitor, as a relatively open software it does have a potential to do so by including a multitude of correctly working submodules. At the time of writing this article, there were six focused on four different types of mistakes – spelling, syntactic, morphological and typographical errors. These modules stood fairly well in testing, where I've provided text with artificial mistakes (see table 1; for further information see the thesis [3]). The modules vary in success, but that is mainly because the less successful modules deal with more complex issues.

Table 1: Number of true and false positive/negative values (TP/FP/TN/FN) for the individual modules and corresponding precision (pre) and recall (rec) rates.

Correction	TP	FP	TN	FN	pre	rec
Misspellings (excl. proper nouns)	24	0	487	16	1,000	0,600
Misspellings (incl. proper nouns)	7	17	497	6	0,292	0,538
Vocalisation of prepositions	4	0	8	0	1,000	1,000
Multiple whitespaces	4	0	515	0	1,000	1,000
Whitespace in the interpunction proximity	7	0	119	0	1,000	1,000
Conditionals	2	0	1	0	1,000	1,000
Commas in a sentence	3	0	0	4	1,000	0,429

3 The proximate issues

As I suggested before, the corrector is far from being perfect. There are many issues, which have different difficulty as well as different priority. In the long run, the development will consist of adding new modules, which will improve corrector as a whole. In the short run, there are tasks, which should improve further development and user acceptance alike.

3.1 Genuine testing

The quality evaluation of modules' success is crucial part of the development. The approach I used, however, is far from ideal. The problem is twofold. The text I used was artificial, eg. it didn't reflect real distribution of users' mistakes, and its length was insufficient. Because of this, the results could be hardly accepted as generally valid. The solution lies in the existence of a collection of genuine texts with correctly labelled errors. On Masaryk University corpus *Chyby* was created, containing various annotated types of errors in texts made by students. [4] Unfortunately, access to this corpus is limited, so I cannot assess its suitability and I did not find any other evidence of the Czech corpus of such quality.

Additionally, the current version of the corrector lacks any kind of interface, which would allow evaluating results automatically. This is necessary as the manual inspection would cost excessive amount of time, which could be used for the further development.

3.2 Error reporting

Next issue covers incorrectly found (non-)errors and users' feedback. Though it isn't implemented in the current version, the reporting itself should be relatively easy. The problem unfolds with keeping these cases hidden even after the minor change of text. Although the current approach is based on tokens, it is not bound to token's content, as there could be a mismatch, but to its position (token

number, see Fig. 3). One option is to extend the binding to different criteria, but due to the various nature of different modules, there would be problem to find any general approach and this issue will have to be addressed by the different correction modules separately. In some cases, I believe the solution will be fairly easy, such as building ignore list for the spell-checker, but for other, it could provide a significant challenge.

Fig. 3: Example of correction, word *runing* would be highlighted and user would be suggested with the correct form.

3.3 Spell-checking

Spell-checking is a distinct chapter of the corrector and such as it has specific issues. They are often described as *non-word error detection*, *isolated-word error correction*, and *context-dependent error correction*. [5] The context-dependent error correction is an advanced task, so in this phase of development, I do not plan to focus on it. The other two, however, are very important as they would be among the first things required by the end users. The non-word error detection aims solely on distinguishing words from non-words. Its precision is closely linked to the quality and size of the lexicon used. Although morphological analyser Majka, whose lexicon is used in the current version of the grammar-checker, provides a fairly big number of word forms, there are contained substandard word forms beside the standard ones. If it would be used in the future, there is the necessity of filtering these forms. Additionally, even though Majka as an analyser, unlike its predecessor, is not dependent on its lexicon, the standard lexicon is not often updated. [6] This raises the question of whether to use it in the future or if would be better to switch to different, expandable resource such as *hunspell*. [7] Either way, there should be created service (if it doesn't already exist with given system), such as a programme or internet application, which would allow the moderator to easily add new words, as they can come in large numbers, which should be then automatically included in the right format.

The related issue is an isolated-word correction. This topic covers automatic corrections as well as suggestions. Unlike the competition, this is not yet covered by the corrector in any way. Some tools, such as previously mentioned hunspell, have their own methods implemented. Possible custom implementation could use algorithms based on *Damerau-Levenshtein distance* measures or other noisy channel approaches such as *Brill and Moore's model*. [8]

4 Conclusion

This paper summarises the development history of the new Czech grammar-checker and uncovers the proximate issues in future development. These issues aren't the only ones to be solved before the corrector could be considered as satisfactory. The success of the final product rests on the success rate of the added modules. There are multiple works on partial topics, which are in some stage of development or relatively freshly finished. This can provide me with quality resources for future module development. Apart from correction itself, the interface has to provide a sufficient range of metatext options, such as headers, bold and italic text, cut and others for users to start using it on daily basis. This is already implemented in the TinyMCE editor, but momentarily disabled.

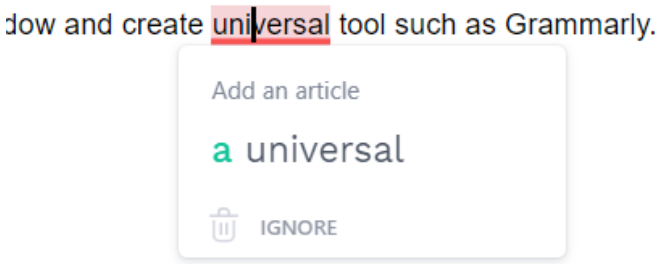

Fig. 4: Correction information in Grammarly.

Although one of the steps is to enable users these options in the editor's window, the ultimate goal is to make the corrector independent. The model for this can be seen in the American *Grammarly* [9], which provides grammar checking of English text on the internet (see Figure 4). As it provides quality and understandable results for its language domain I hope I will be able to get at least close to its successes from the Czech language point of view.

Acknowledgements. This work was supported by the project of specific research *Čeština v jednotě synchronie a diachronie* (Czech language in unity of synchrony and diachrony; project no. MUNI/A/0862/2017).

References

1. Petkevič, V.: Kontrola české gramatiky (český grammar checker). Studie z aplikované lingvistiky-Studies in Applied Linguistics [online] **5**(2) (2014 [2018-10-31]) 48–66
2. TinyMCE: Create a plugin for tinymce [online] (2018 [2018-10-31])
3. Mrkývka, V.: Webové rozhraní pro automatický jazykový korektor češtiny [online]. Diplomová práce, Masarykova univerzita, Filozofická fakulta, Brno (2018 [2018-10-31])

4. Pala, K., Rychlý, P., Smrž, P.: Text corpus with errors. In Matoušek, V., Mautner, P., eds.: Text, Speech and Dialogue [online], Berlin, Heidelberg, Springer Berlin Heidelberg (2003 [2018-10-30]) 90–97
5. Kukich, K.: Techniques for automatically correcting words in text. ACM Comput. Surv. [online] **24**(4) (December 1992 [2018-10-31]) 377–439
6. Šmerk, P., Rychlý, P.: Majka – rychlý morfologický analyzátor [online]. Technical report, Masarykova univerzita (2009 [2018-10-31])
7. Németh, L.: Hunspell: About [online] (2003 [2018-10-31])
8. Brill, E., Moore, R.C.: An improved error model for noisy channel spelling correction. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics [online]. ACL '00, Stroudsburg, PA, USA, Association for Computational Linguistics (2000 [2018-10-31]) 286–293
9. Grammarly: About | grammarly [online] (2018 [2018-10-31])