Multilingual spell checker based on morphological analyzer for Turkic languages

N.A. Prokopyev

TurkLang 2021
1. Introduction

- Automatic spell checking task is one of the actual problems in Natural Language Processing

- Especially: development of unified spell checker for all Turkic languages covered on Turkic Morpheme portal
2. Turkic Morpheme portal as a resource for spell checker

- There is an ongoing research and development of Turkic Morpheme portal database in form of a unified linguistic resource for Turkic languages

- Software tools based on this database inherit unification quality as they work on the same basis, algorithms and data structures
3. Spell checker development methodology

- Morphological analyzer is developed based on the portal database which potentially supports all Turkic languages in database provided they have linguistic data – it is the base NLP preprocessor for spell checker

- Morphological analyzer functions as programming library in NLP Pipeline format and as a web-service – spell checker should have the same structure

- Morphological analyzer and portal database allows to implement partially correct words analysis and error correction variants generation – this should be implemented in spell checker
4. NLP Pipeline structure

Input text

Language model

Config. json

Proces sor 1

Proces sor 2

Proces sor 3

Processing result

Common intermediate data format
5. The Turkic morphoanalysis library

- **importer.py** – module providing data export from portal database to language model in .sqlite format
- **analyzer.py** – main module of morphoanalyzer with analysis methods
- **generator.py** – module of generator for making all possible word variants from morphoanalysis scheme
- **translator.py** – module that uses analyzer and generator for making all possible word translations of input text
6. Morphotactic rules model from Turkic Morpheme portal
7. Morphoanalysis algorithm

- LL-analysis of root morpheme taking into account the morph. type root strip at the end
- Aff. allomorph search taking into account the linking chars from morphotactic rules

-Recursive search of aff. allomorphs using morphotactic rules

- Checking if the next word is an analytical allomorph corresponding to morphotactics

- Morphoanalysis scheme output
8. Morphoanalysis interface
9. Analysis scheme examples
10. Spell checker algorithm

Getting morphoanalysis scheme from NLP Pipeline

Scheme ends with error

Transition to last nodes of analysis scheme, finding the remaining word part

Search for next possible allomorph chains according to morphotactics up to the remaining prat length

Sort the list of found allomorph chains based on Levenshtein distance to the remaining word part

Output of correct word part + first N of found allomorph chains

Output None
11. Error correction example

Correct: законнары

Error: законлары

According to morphotactics after закон root can be the next allomorphs:
-сыз, -нан, -ны, -га, -чыг, -чык, -ча, -ны, -ның, -ка, -кай, etc. (38 variants)

Taking into account the remaining word part -лары:
1. Longer length allomorphs are discarded;
2. Other allomorphs are connected with next morphotactics allomorphs until remaining word length reached;
3. Obtained allomorph chains are compared with the remaining word part by the Levenshtein distance;
Output: Correct word part + up to N closest allomorph chains.
11. Conclusion

- A task is given to develop the programming library and web-service tools for automatic spell checking of Turkic languages in a unified architecture

- It can be solved using the already created databases and morphoanalysis software based on Turkic Morpheme portal

- Quality of spell checker functioning for some specific language is correlated to completeness of database for this language
Thank you for attention!

Игътибарыгъыз өчен рәхмәт!

Спасибо за внимание!