MorAz: an Open-source Morphological Analyzer for Azerbaijani Turkish

Berke Özenç†, Razieh Ehsani††, Ercan Solak†
† Computer Engineering ,Işık University, Istanbul, Turkey
†† Department of Modern Languages, University of Helsinki, Finland
{berke.ozenc, ercan.solak}@isikun.edu.tr
razieh.ehsani@helsinki.fi

Abstract

MorAz is an open-source morphological analyzer for Azerbaijani Turkish. The analyzer is available through both as a website for interactive exploration and as a RESTful web service for integration into a natural language processing pipeline. MorAz implements the morphology of Azerbaijani Turkish following a two-level approach using Helsinki finite-state transducer and wraps the analyzer with python scripts in a Django instance.

1 Introduction

Morphological analysis is a crucial part of processing languages with complex morphologies, such as the agglutinative Azerbaijani Turkish. The morphological analysis provides a number of "readings" or analysis for each word, as a part of the overall NLP task. Indeed, morphological analysis yields some properties of a word like "stem", "root" and morphological role of "suffixes" inside word. Naturally, when the number of suffixes and their combination increase, so does the number of possible analysis of a word.

Since its application to morphology by Koskenniemi [Koskenniemi, 1983], Finite State Transducer (FST) has become a favored computational tool for representing morphology and phonology. In the two-level approach developed in Koskenniemi [Koskenniemi, 1983], the morphotactics is represented as a separate FST in the first level. The output of the first level is then re-written by a sequence of phonological re-write rules.

The two-level approach to morphology has been successfully applied to many languages with several publicly available analyzers, [Karp et al., 1992], [Piskorski et al., 2009]. There are also a number of open-source toolkits that provide the underlying FST implementation, [Hulden, 2009], [Lindén et al., 2011], [Schmid, 2005].

In this paper, we present an open-source FST implementation of the full morphology of Azerbaijani Turkish (AT). Noun and Verb morphology were previously discussed in [Ehsani et al., 2017]. The source code is available for use as a local analyzer. It is also available as a RESTful web service.

The rest of the paper is organized as follows. In the next section, we review related work. In Section 3, we outline the structure of MorAz. Section 4 introduces the website and the web service of MorAz. In Section 5, we report some statistics on the performance of the analyzer. Finally, the paper finishes with some concluding remarks.

2 Related analyzers

MorAz is the first complete morphological analyzer for AT. There is also a partial implementation of AT morphology within the Apertium project [Forcada et al., 2011]. This analyzer is based upon the Trmorph [Çöltekin, 2010] with the assumption that the Azerbaijani Turkish and Anatolian Turkish are similar, whereas our analyzer was developed from scratch directly for the Azerbaijani Turkish. Apertiums coverage of the morphotactics and phonology and the extent of its lexicon are quite narrow compared with MorAz. So, Apertium Azerbaijani analyzer is not sufficient for testing. Moreover, the only way to use Apertium analyzer is through incorporating the code base into the NLP pipeline, with all its dependencies and libraries. The web service interface to MorAz does not require anything other than `json` constructors and parsers. The manually constructed lexicon of MorAz reduces the number of redundant analyses due to trivial derivations resulting from an automatic root lexicon such as the one used in Apertium. The coverage of morphotactics rules in MorAz is wider and thus results in correct anal-

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (System Demonstrations), pages 25–29
Brussels, Belgium, October 31–November 4, 2018. ©2018 Association for Computational Linguistics

yses where Apertium analyzer results in out-of-vocabulary analyses.

Morphological analyzers for other Turkic languages have varying levels of completeness and availability. Among these, the most widely studied language is Anatolian Turkish. [Oflazer, 1994] presented a two-level description and implementation of Turkish morphology. Their implementation uses `xfst` [Beesley and Karttunen, 2003] as the underlying FST implementation. Their analyzer is not available publicly. Following the same approach as Oflazer's, Şahin's [Şahin et al., 2013], re-implemented the analyzer on Xerox [Beesley and Karttunen, 2003]. Şahin's analyzer is available through a web interface and as a web service, though, the source is closed. TRMorph [Çöltekin, 2010] is an open-source analyzer for Anatolian Turkish, implemented over SFST. It is available as an interactive web tool but lacks a web service interface. Zemberek [Akın and Akın, 2007] is another open-source, Java-based analyzer for Anatolian Turkish.

For Kazakh, there is an open-source analyzer in Apertium project. There is also, the analyzer described in [Kessikbayeva and Cicekli, 2016], however, currently, the implementation is not publicly available. For Turkmen and Uighur, the analyzers described in [Tantug et al., 2006] and [Orhun et al., 2009] are not publicly available.

3 Structure of MorAz

Azerbaijani Turkish (AT) is a Turkic language spoken by about 30 million people, mainly in Iran and Azerbaijan. AT is an agglutinative language with a predominant SOV word order, although scrambling is common especially in spoken form. The phonology of AT has vowel harmony, devoicing, and apocope. Written AT uses Latin alphabet in Azerbaijan and Arabic alphabet in Iran. The current implementation of MorAz works with the Latin alphabet.

The FST description of the morphology of AT as implemented in MorAz consists of 4 main parts; nominal and verb inflections, nominal predicate, and derivation. Derivation FST is the bridge that connects the other 3 FSTs. In detail, the derivation FST has 36, nominal inflection has 36, nominal predicate has 22 and verb inflection has 145 rules. Morphotactics level which is also called level 1 has 239 rules and 67 states in total. Complete FST diagram of MorAz is shown in Figure 2. Since adjectives in AT behave like nouns when their suffixation is concerned, we treat adjective and nouns as a single morphological class Nominal. At a morphosyntactic level, there will still be two distinct POS tags for adjectives and nouns. In MorAz, we used 8 morphological categories: Nominal, Verb, Predicate, Adverb, Number, Postposition and Interjection.

In MorAz we represent the abstract form of a morpheme either as a key-value pair or just as a key.

The key-value form is more suited for consistently representing the inflection paradigms where a zero surface realization of the abstract morpheme corresponds to a particular assignment of the inflection feature. For example, Number feature has zero surface form when Singular. When it is Plural, it is realized as -lar or -lər depending on vowel harmony. Since every Nominal has a Number feature, we reserve a number slot in Nominal Inflection.

We denote the key-value abstract morphemes as `<Key_Mnemonic:Value>`.

When a morphological feature is optional, we use just a mnemonic key to represent the corresponding morpheme in the form `<Key>`. For example, all derivational morphemes are optional.

The following example illustrates the use of abstract morphemes.

(1) xəstəlikdən

xəstə<NOM>
<State><NOM>
<Num:Sg><Poss:No><Case:Abl>

The documentation for all the mnemonic keys and their possible values are provided on the website of MorAz. There are a total of 38 keys in the key-value pair form and 40 optional keys, 20 of which correspond to derivational morphemes.

Figure 1 gives the FST for Nominal inflection as an illustration of the morphotactics of MorAz. The expansion of transition labels sn_1-sn_1 is given in full in the expanded diagrams on the MorAz website.

The root lexicon includes 2707 verb roots, 35547 nominal roots as well as 14937 person names and 929 adverbs. We obtained the root lexicon of MorAz, by reducing a large lexicon of roots. In the reduction, we manually eliminated the roots that can be trivially derived from other

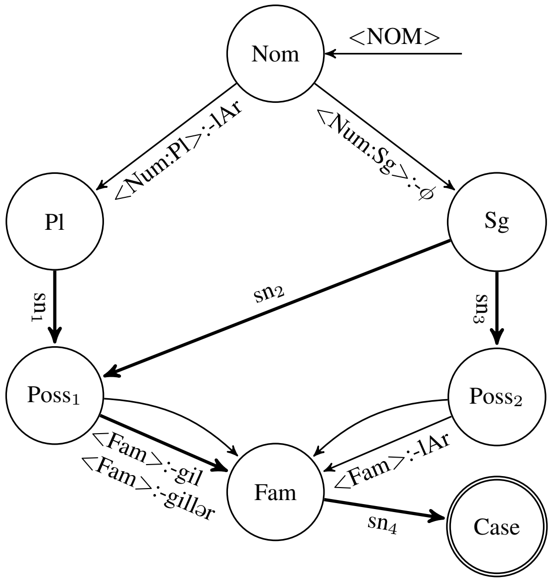

Figure 1: Nominal inflection

roots that are not eliminated. The cases where the derived form undergoes a meaning drift away from the one that the derivational morpheme nominally entails are distinguished. If the drift is so large that the meaning of the derivation cannot be inferred from those of the root and the suffixes, then a new word needs to be added to the dictionary [Ehsani et al., 2018]. For example, the large lexicon contains both

(2) xəstə
xəstə<NOM>
sick

and

(3) xəstəlik
xəstə<NOM><State>
sickness

where 3 is trivially derived from 2.

In AT, there are 4 distinct morphemes for Causative and 2 morphemes for Passive.

In order to handle the selection of Causative and Passive morphemes, we manually marked our verb lexicon of about 2700 verb roots with 15 verb classes. These include the classes representing the cases where a verb root cannot be suffixed with Causative for some intransitive verbs and the cases where a Passive is semantically impossible. For example, “öyren” (learn) has no Causative and “dol” (be filled) has no Passive form.

The second level of MorAz deals with the phonology. The first level output consists of base morphemes and archmorphemes. Archmorphemes use 5 archiphonemes which are given in Table 1.

Archiphoneme	Surface forms
A	ə, a
I	ı, i, u, ü
K	k, y
Q	q, ğ
D	d, t
N	d, n

Table 1: Archiphonemes used at the first level output of MorAz

The archiphoneme A maps to its surface form to satisfy back-front harmony. Similarly, I maps to its surface forms under back-front and roundedness harmony. K and Q choose their surface forms through palatalization and velarization, respectively. D chooses its surface form to adapt to the voicing feature of its context. Finally, N is a convenience archiphoneme that we use to unify two surface forms of the Ablative morpheme.

A common phonological phenomenon in AT is the insertion of epenthetic letters y, n, ş, and s. The choice of the epenthesis phoneme depends on the phonological and morphological context. In MorAz implementation, we consider epenthetic as optional phonemes attached to morphemes. So, the phonological rules in the second level drop the epenthetic depending only on the phonological context.

4 Website and API

MorAz uses Helsinki finite-state transducer (HSFT) for the implementation of the two-level morphology. We wrapped the compiled analyzer with python scripts in a Django web server. The source code for the analyzer is available in GitHub [1].

MorAz website includes an interactive query screen shown in Figure 3. It allows querying multiple tokens separated by line breaks.

The web service API [2] uses the `json` format for posting the list of tokens to be analyzed. The output is also in `json` format as an array of arrays where the innermost array contains the list of analyses for a single token.

[1] https://github.com/berkeozenc/MorAz
[2] http://ddil.isikun.edu.tr/morazws/

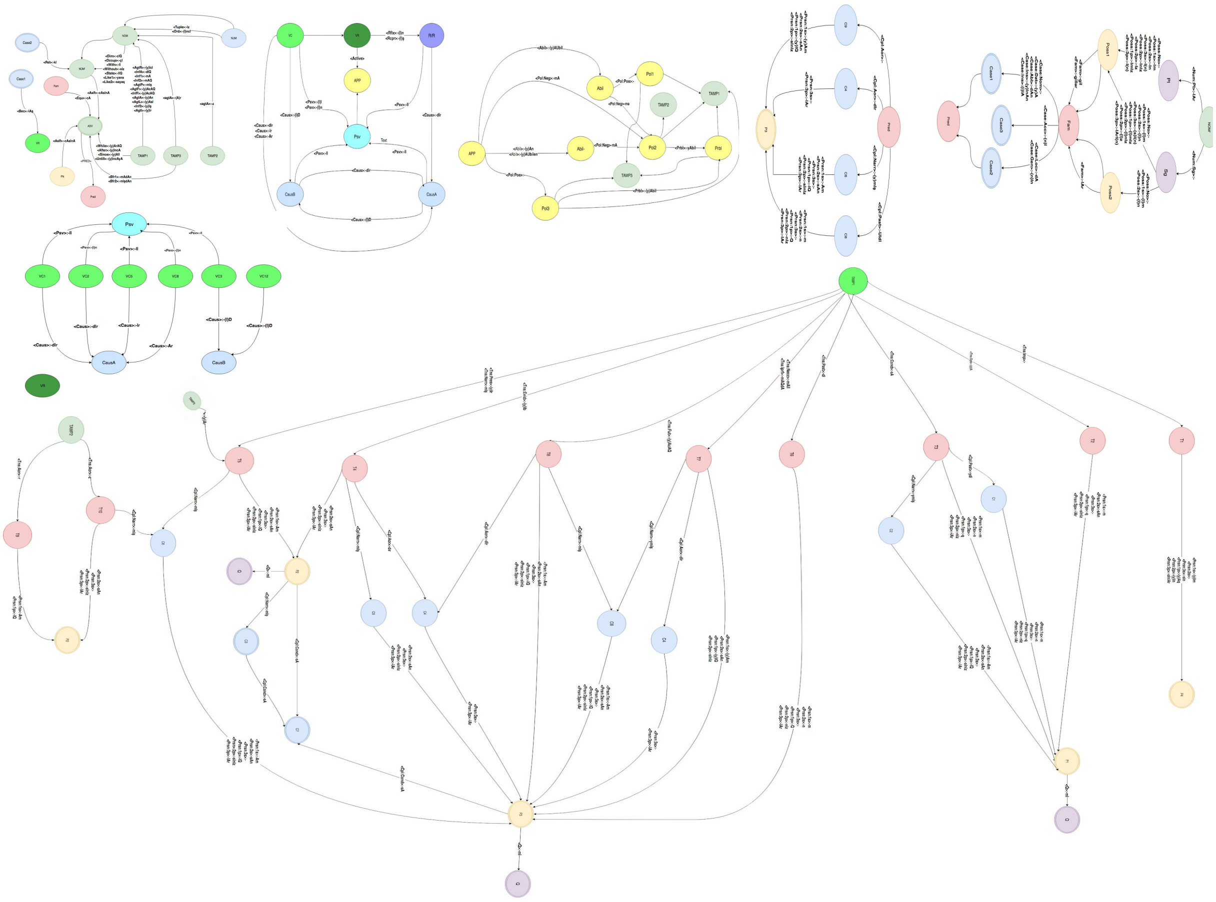

Figure 2: Complete FST Diagram of MorAz

5 Statistics

In order to measure the performance of MorAz, we ran it over an input text collected from BBC Azərbaycanca. Since MorAz lexicon is not complete in terms of named entities, we eliminated from the input all the tokens that start with capital letters. What remained was a test input is a list of 10890 distinct Azerbaijani words. We also eliminated punctuation marks.

Of all the tokens fed into the analyzer, MorAz did not return an analysis for %23.92 of total words. For the ones that it provided an analysis, on average there were 1.96 analyses per word. Since the token is Azerbaijani word, it is possible to use them to test other Azerbaijani morphological analyzers.

6 Conclusion

In this paper, we presented MorAz, an open-source morphological analyzer for Azerbaijani Turkish. MorAz provides an interactive query interface for short pieces of tokenized text through its website. For larger inputs, it exposes a simple RESTful web interface.

MorAz has a manually crafted minimal lexicon, with an aim to reduce the number of redundant analyses. Manual configuration is an ongoing process and we modify the lexicon by inspecting the results of analyses.

As a further development, we are planning to provide an interactive tool to generate surface forms out of abstract morphemes which will be useful for exploring the language.

References

Ahmet Afsin Akın and Mehmet Dündar Akın. Zemberek, an open source nlp framework for turkic languages. *Structure*, 10:1–5, 2007.

Kenneth R. Beesley and Lauri Karttunen. *Finite State Morphology*. CSLI Publications,, Stanford, CA, 2003.

Çağrı Çöltekin. A freely available morphological analyzer for Turkish. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC*

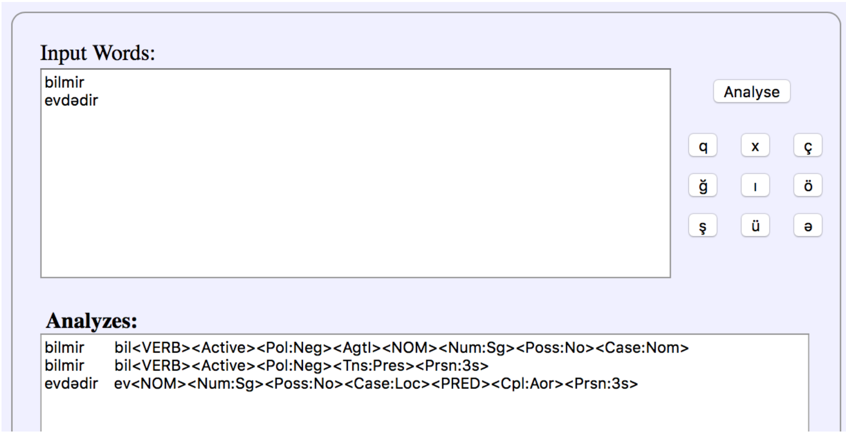

Figure 3: Interactive query in MorAz.

2010), pages 820–827, 2010. URL http://www.lrec-conf.org/proceedings/lrec2010/summaries/109.html.

Razieh Ehsani, Berke Özenç, and Ercan Solak. A fst description of noun and verb morphology of azarbaijani turkish. In *Proceedings of the 13th International Conference on Finite State Methods and Natural Language Processing (FSMNLP 2017)*, pages 62–68, 2017.

Razieh Ehsani, Ercan Solak, and Olcay Taner Yıldız. Constructing a wordnet for turkish using manual and automatic annotation. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 17(3), 2018.

Mikel L Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine translation*, 25(2):127–144, 2011.

Mans Hulden. Foma: a finite-state compiler and library. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 29–32. Association for Computational Linguistics, 2009.

Daniel Karp, Yves Schabes, Martin Zaidel, and Dania Egedi. A freely available wide coverage morphological analyzer for english. In *Proceedings of the 14th conference on Computational linguistics-Volume 3*, pages 950–955. Association for Computational Linguistics, 1992.

Gulshat Kessikbayeva and Ilyas Cicekli. A Rule Based Morphological Analyzer and a Morphological Disambiguator for Kazakh Language. *Linguistics and Literature Studies*, 4(1):96–104, 2016.

Kimmo Koskenniemi. Two-level model for morphological analysis. In *IJCAI*, volume 83, pages 683–685, 1983.

Krister Lindén, Erik Axelson, Sam Hardwick, Miikka Silfverberg, and Tommi Pirinen. HFST–framework for compiling and applying morphologies. pages 67–85, 2011.

Kemal Oflazer. Two-level description of turkish morphology. *Literary and Linguistic Computing,*, 9(2):137–148, 1994.

Murat Orhun, A. Cüneyd Tantug, and Esref Adali. Rule based analysis of the uyghur nouns. *Int. J. of Asian Lang. Proc.*, 19(1):33–44, 2009. URL http://dblp.uni-trier.de/db/journals/jclc/jclc19.html#OrhunTA09.

J Piskorski et al. Morphisto-an open source morphological analyzer for german. In *Finite-state Methods and Natural Language Processing: Postproceedings of the 7th International Workshop FSMNLP; Edited by Jakub Piskorski, Bruce Watson and Anssi Yli-Jyrä*, volume 7, page 224. IOS Press, 2009.

Muhammet Şahin, Umut Sulubacak, and Gülşen Eryiğit. Redefinition of turkish morphology using flag diacritics. In *Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013)*, Phuket, Thailand, October 2013.

Helmut Schmid. A programming language for finite state transducers. In *FSMNLP*, volume 4002, pages 308–309, 2005.

A Cüneyd Tantug, Esref Adali, and Kemal Oflazer. Computer Analysis of the Turkmen Language Morphology. *FinTAL*, 4139:186–193, 2006.