Правительство Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
Национальный исследовательский университет
«Высшая школа экономики»

Факультет гуманитарных наук
Образовательная программа
«Фундаментальная и компьютерная лингвистика»

Картина Элен Геннадьевич

Правиловый морфологический парсер для шугнанского языка: существительные, глаголы и прилагательные

Выпускная квалификационная работа студента 4 курса бакалавриата группы БКЛ211

Академический руководитель образовательной программы
канд. филологических наук, доц.
Ю.А. Ландер
____
«_____» ____________ 2025 г.

Научный руководитель
канд. филологических наук, доц.
Г.А. Мороз
____

Научный консультант
Стажёр-исследователь
М.Г. Мельченко
____

Москва 2025

Contents

Abstract

In this work I present a rule-based morphological analysis tool based on Helsinki Finite-State Technology (HFST) for the Shughni language (ISO: sgh; glottocode: shug1248), a language of the Iranian branch of the Indo-European family, a member of 'Pamiri' areal language group. While one existing rule-based parser exists for Shughni (Melenchenko, 2021), it does not utilize finite-state transducer technology. This work proposes the first HFST-based morphological parser implementation for Shughni, offering the advantages of this well-established framework for morphological analysis. The parser is presented in two variations: a morphological parser that breaks each word-form into stem and morphemes and assigns morphological tags to each one of them; a morphological generator that outputs word-forms taking a stem and morphological tags as an input. **TODO: prev sentence is questionable** This is a continuation my previous work, where nouns, pronouns, prepositions and numerals were implemented (Osorgin, 2024). This project covers **TODO: what**

TODO: Review abstract after finishing the work

1 Introduction

1.1 Shughni

The Shughni language (ISO: sgh; Glottolog: shug1248) is a language of the Iranian branch of the Indo-European family (Plungian, 2022, p. 12). As of June 1997, it was estimated to be spoken by approximately 100,000 people (Edelman & Yusufbekov, 1999, p. 225) in the territories of Tajikistan and Afghanistan. Both countries have a subregion where Shughni is the most widely spoken native language. The Shughni-speaking subregion of Tajikistan is called 'Shughnon' and it belongs the to 'Gorno-Badakhshan Autonomus' province. In Afghanistan, the Shughni-speaking region is called 'Shughnan' and it lies within the territory of 'Badakhshan' province (Parker, 2023, p. 2). Shughni belongs to 'Pamiri' areal language group, which is spoken along the Panj river in Pamir Mountains area.

Figure 1: Mountainous Badakhshan Autonomous Province of Tajikistan and Badakhshan Province of Afghanistan, (Parker, 2023, Fig 1.1)

There are three alphabets for Shughni that were derived from Cyrillic, Arabic and Latin scripts. Geographically the usage of said scripts correspond to the dominant script of each country where Shughni is spoken. In Tajikistan both official languages (Tajik and Russian) use Cyrillic script, so does Shughni on territory of Tajikistan. In Afghanistan Arabic script is used in Shughni, matching official languages (Pashto and Dari).

Latin script was developed and used in Tajikistan in 1930s (Edelman & Yusufbekov, 1999, p. 226) (Edelman & Dodykhudoeva, 2009, p. 788), but according to Edelman and Yusufbekov (1999) was not widely adapted. Later around 1980s a Cyrillic script gained popularity in Tajikistan, having some poetic literature and school materials based on Tajik's alphabet, which is Cyrillic (Edelman & Yusufbekov, 1999). Today, Latin script is mostly used by researchers in scientific works.

The morphological parser developed in this work is based on materials that focus on Shughni spoken in Tajikistan. All the base lexicon is Cyrillic and comes from dictionaries that cover Shughni in 'Gorno-Badakhshan Autonomus Province'. Latin script is supported with the help of transliteration.

1.2 Morphology modeling

Today there are two general approaches to the task of morphology modeling. The deep learning (DL) approach and the rule-based approach.

The DL approach typically makes use of training transformer models like `BERT` (Devlin et al., 2019) on vast amounts of marked-up data. This task becomes challenging, considering that Shughni is a low-resource language, meaning it lacks digital textual data. Although, DL approach was not utilized in this work, some existing DL approaches for low-resource languages are covered in section 2.1.

With the rule-based approach, morphological model is being built by writing grammar rules using some formalism language and listing base lexicon. In this work, rule-based approach was utilized, as it does not depend on the amount of available marked-up data as the DL approach does. It requires lexicons and morphological grammar descriptions, which exist for Shughni and which are discussed in Section 3.

2 Existing methods

2.1 Machine learning methods

There are a variety of LLM (Large language model) architectures that were applied to the task of language modeling. One significant example is `LSTM` (Long short-term memory) model, that was introduced by Hochreiter and Schmidhuber (1997). `LSTM` is a variation of `RNN` (Recurrent neural network), and it was widely applied to language modeling, including morphology modeling. Another more recent significant example is the transformer architecture presented by Vaswani et al. (2017), off which two years later `BERT` model was based (Devlin et al., 2019).

One of the biggest downsides of ML methods is that its quality depends on training data quantity, which makes it challenging to apply to low-resourse languages such as Shughni. However, with introduction of LLMs this problem was shown to be solvable, for example, as shown by developers of `UDify` model (Kondratyuk & Straka, 2019). In their work authors show, that a `BERT` model pretrained on a large corpus of 104 languages can be fine-tuned on very little amounts of other languages' data and still show decent results. For an example, they report that for Belarusian, `UDify` model achieved $UFeats = 89.36\%$ (accuracy of tagging Universal Features) after training on only 261 sentences from 'Belarusian HSE' Universal Dependencies treebank (Kondratyuk & Straka, 2019, Table 7).

However, working with LLM models is a highly resource-demanding task. The authors of `UDify` state, that the fine-tuning of their model for a new language would require at least 16 Gigabytes of RAM and at least 12 Gigabytes of GPU video memory, and the training process would take at least 20 days depending on the GPU model. While a deep learning approach would be interesting to explore, such computational resources are not available for this project. The neural approach is not the main target of this work and is implemented.

2.2 Rule-based methods

2.2.1 Finite-state transducers

The Rule-based approach historically is usually applied with the help of Finite-state transducers (FST), which is a variation of Finite-state machine, a mathematical abstract computational model. Following the terminology of Turing machines (Turing, 1937), a FST has two tapes: the input tape and the output tape. At any point it can read a next symbol from the input tape and then write a symbol to the output tape. Once a symbol was read from the input tape, it can not be read again, as the input tape shifts one symbol forward.

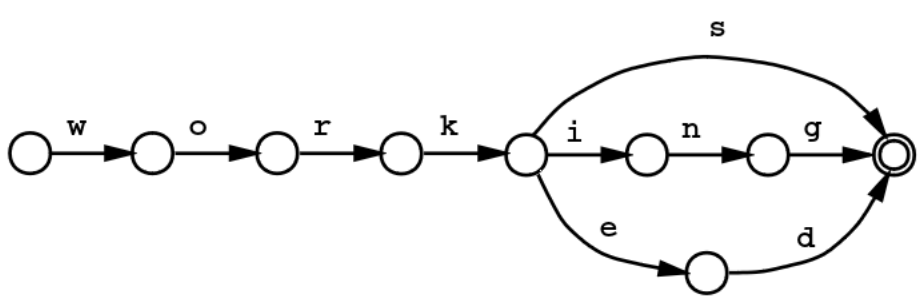

Figure 2: An example of FST with a single initial state (most left node) and a single final state (most right node) for a language where only three words exist: *works*, *working* and *worked*. The word *worker*, for example will not be recognized as a valid word by this FST, since there is no 'd' transition at state *worke*. The only way from *worke* state is via 'd' transition, which corresponds to the *worked* word. (Beesley & Karttunen, 2002)

The inner structure of FST can be illustrated as a directed graph with a set of all *states* (represented by graph's nodes), a set of *transitions* (represented by graph's edges), a set of *initial* *states* (a subset of all the states, these are states where FST can start reading from the input tape) and a set of *final states* (a subset of all the states, these are the states where FST can stop reading from the input tape). A simplified FST is shown on Figure 2. The letters above the graph's edges denote *transition* rules, for an example *transition* 'w' means 'read *w* from the input tape THEN write *w* to the output tape'.

While working, FST will only make transitions that are possible from the current state. If there are no valid transitions then FST fails to process the input, and the input is considered to be impossible in the current language model. Of course, the ideal FST model of a language has valid paths for all the grammatical wordforms and does not have any valid paths for any ungrammatical wordforms. The measure of the amount of language's grammatical wordforms that successfully pass through the FST from an *initial state* to a *final state* will be called *Coverage* from now and on. The measure of the amount of language's ungrammatical wordforms that successfully pass through the FST from an *initial state* to a *final state* will be called *Overgeneration* from now and on.

The model from the Figure 2 works effectively as a wordform paradigm dictionary, echoing back input wordforms that are grammatical and failing to output the whole ungrammatical wordforms. Now we can slightly adjust the transition rules in put example to make a morphological analysis tool that can bee seen on the Figure 3. The notation of the *transition* 'w:w' is an alias for the notation 'w' from the Figure 2. If the spot on the right side of the semicolon (':') sign is left empty, it means 'write nothing to the output tape'. An important note to remember is that FST can output only one symbol to the output while making a single transition. In this example *'<inf>'*, *'<pst>'* and *'<prs><2sg>'* are treated as 'multichar' symbols, meaning they are treated as three individual symbols by a FST, it will be covered in more detail in the Section 4.

2.2.2 FST formalisms

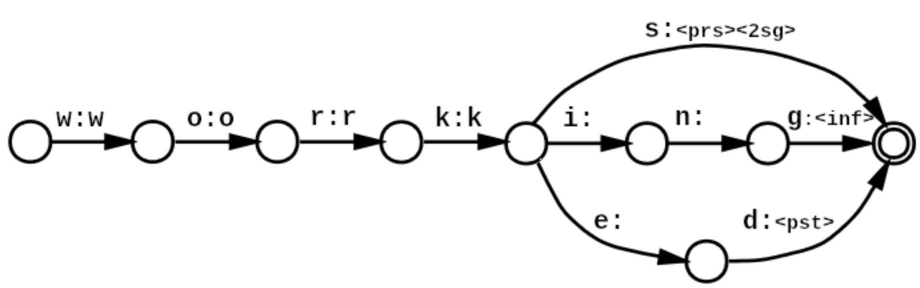

Figure 3: A modified version of Figure 2 which takes as input *works*, *working*, *worked* and outputs *work<prs><2sg>*, *work<inf>*, *work<pst>* respectively.

2.3 Existing morphology models for Shughni

At this time only one morphological parser exists for Shughni. It was developed by Melenchenko (2021) and was later included in 'Digital Resources for the Shughni Language' project (Makarov et al., 2022). It is a rule-based parser implemented in Python which shows good coverage and accuracy results

3 Data

3.1 Grammar descriptions

3.2 Dictionaries

3.3 Text corpora

4 Methods

4.1 Finite-state transducers TODO: Move to introduction?

4.2 Rule declaration TODO: rethink heading

4.2.1 Nouns

4.2.2 Verbs

4.2.3 Adjectives

4.2.4 Pronouns

4.2.5 Numerals

4.2.6 Anything else(?)

4.3 Transliteration

4.4 Russian lemmas TODO: rethink heading

4.5 Testing

4.6 Metrics

5 Results

6 Conclusion

References

Beesley, e. R., & Karttunen, L. (2002). *Finite-state morphology: Xerox tools and techniques*. CSLI, Stanford.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

Edelman, D. I., & Yusufbekov, S. (1999). Шугнанский язык [Shughni language]. In *Языки мира: Иранские языки. III: Восточноиранские языки [Languages of the world: Iranian languages. III. Eastern Iranian languages]*. https://iling-ran.ru/web/ru/publications/langworld/volumes/7

Edelman, D. I., & Dodykhudoeva, L. R. (2009). Shughni. In G. Windfuhr (Ed.), *The iranian languages* (pp. 787–824). London & New York: Routledge.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kondratyuk, D., & Straka, M. (2019). 75 languages, 1 model: Parsing universal dependencies universally.

Makarov, Y., Melenchenko, M., & Novokshanov, D. (2022). Digital Resources for the Shughni Language. *Proceedings of The Workshop on Resources and Technologies for Indigenous, Endangered and Lesser-Resourced Languages in Eurasia within the 13th Language Resources and Evaluation Conference*, 61–64. https://aclanthology.org/2022.eurali-1.9

Melenchenko, M. G. (2021). *Автоматический морфологический анализ шугнанского языка [Automatic full morphology analysis for Shughni]*, NRU HSE.

Osorgin, I. G. (2024). *Создание морфологического парсера для шугнаского языка в системе lexd и twol [Creating a morphological parser for the Shughni language using lexd and twol systems]*, NRU HSE.

Parker, C. (2023). *A grammar of the Shughni language* [Doctoral dissertation, Department of Linguistics, McGill University].

Plungian, V. (2022). The study of shughni: The past and the future. *RSUH/RGGU Bulletin: "Literary Teory. Linguistics. Cultural Studies", Series. 2022;(5):11-22*.

Turing, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. https://doi.org/10.1112/plms/s2-42.1.230

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf