The Journal of Open Source Software

Shekar: A Python Toolkit for Persian Natural Language Processing

Ahmad Amirivojdan [1¶]

1 University of Tennessee, Knoxville, United States ¶ Corresponding author

DOI: 10.21105/joss.09128

Software

- Review
- Repository
- Archive

Editor: Chris Vernon

Reviewers:

- @linuxscout
- @evamaxfield

Submitted: 24 August 2025
Published: 21 October 2025

Summary

Shekar is an open-source Python toolkit for Persian natural language processing (NLP). It provides a modular, efficient, and easy-to-use framework for tasks such as text preprocessing, tokenization, stemming, lemmatization, part-of-speech (POS) tagging, named entity recognition (NER), keyword extraction, embeddings, spell checking, and visualization. Its composable pipeline design supports reproducible workflows that scale effectively from basic text processing to large-scale corpus analysis.

Statement of need

Persian natural language processing has expanded rapidly over the past decade, supporting applications in digital humanities, social media analysis, conversational systems, and language modeling. Accordingly, there has been substantial progress in developing tools for Persian language processing. Libraries such as Hazm, Parsivar (Mohtaj et al., 2018), and DadmaTools (Jafari et al., 2025) provide essential functionalities including normalization, tokenization, and part-of-speech tagging. While these tools are valuable, they often come with limitations that restrict flexibility, extensibility, and efficient deployment in diverse settings.

Common challenges include:

- Limited modularity, making it difficult to adapt components or extend functionality.
- Tight coupling between processing stages, which hinders the creation of custom workflows.
- Heavy dependencies and large transformer models that require GPUs or specialized hardware for reasonable performance.
- Irregular updates and delayed issue resolution, which affect long-term maintainability.

Shekar was developed to address these challenges by offering a lightweight, modular, and extensible toolkit for Persian natural language processing. It adopts a clean and composable pipeline architecture, allowing users to build workflows from independently configurable components. This modular approach is especially effective for handling the complexities of Persian script, including inconsistent diacritics, spacing conventions, and the blend of formal and colloquial writing styles. It also supports practical applications such as OCR post-processing, social media text normalization, and data preparation for training language models.

A key distinction of Shekar is its focus on algorithmic efficiency and accessibility. The toolkit combines optimized preprocessing algorithms with lightweight, quantized transformer models, making it practical for low-resource devices as well as diverse environments ranging from research laboratories to production systems with limited computational capacity.

By emphasizing simplicity, performance, and modular design, Shekar fills a critical gap in the Persian NLP ecosystem. It offers a practical and user-friendly toolkit that supports both rapid experimentation and robust deployment across a wide range of use cases.

Main Components

Shekar provides a set of key functionalities covering essential tasks in Persian natural language processing. Each functionality is implemented as an independent component that can be combined into customizable pipelines.

- **Preprocessing** Provides text normalization tools to handle Persian-specific challenges such as inconsistent diacritics and spacing rules. The normalization steps follow the orthographic and typographic guidelines defined by The Academy of Persian Language and Literature and include filters for punctuation, digits, emojis, non-Persian characters, and corrections for spacing and script variants.
- **Tokenization** Offers word-level, sentence-level, and SentencePiece (Kudo & Richardson, 2018) tokenizers built specifically for Persian text. These tokenizers use Unicode-aware rules to handle zero-width non-joiners, punctuation boundaries, and other language-specific edge cases.
- **Morphological Analysis** Includes stemming, lemmatization, inflection, and verb conjugation tools. The stemmer applies rule-based suffix removal, while the lemmatizer combines vocabulary lookups with morphological rules to generate accurate base forms. The inflector and verb conjugator support the generation of correct word forms and verb tenses for diverse linguistic applications.
- **Part-of-Speech (POS) Tagging** Provides lightweight transformer-based models for assigning POS tags to tokens using the Universal Dependencies tagging scheme. The models are based on a pretrained ALBERT model (Lan et al., 2019) trained on the large-scale Naab corpus (Sabouri et al., 2022) and fine-tuned on The Persian Universal Dependency Treebank (PerUDT) (Rasooli et al., 2020) for accurate and efficient POS tagging.
- **Named Entity Recognition (NER)** Offers models trained on the publicly available Persian NER dataset and fine-tuned from the same pretrained ALBERT model used for POS tagging. These models identify entities such as persons, locations, organizations, and dates in Persian text with high accuracy.
- **Keyword Extraction** Implements algorithms for identifying the most informative terms and phrases within a text. The default implementation uses the RAKE (Rapid Automatic Keyword Extraction) algorithm (Rose et al., 2010) for efficient and unsupervised keyword extraction.
- **Embeddings** Supports both static embeddings (FastText) and contextual embeddings (ALBERT-based) for representing words and sentences as numerical vectors.
- **Sentiment and Toxicity Detection** The Sentiment Analysis module features an ALBERT-based transformer model trained on the Snapfood dataset (Farahani et al., 2021) to classify Persian text as positive or negative. The Toxicity Detection module includes an offensive language classifier based on Logistic Regression trained on character-level TF-IDF features using the Naseza dataset (Amirivojdan, 2025). Together, they provide efficient and accurate tools for sentiment understanding and content moderation in Persian text.
- **Spell Checking** Identifies and corrects common spelling errors and spacing mistakes in Persian text. The spell checker uses a frequency-based approach built on a combined dictionary constructed from Persian generative lexicon dataset (Eslami et al., 2004) and unique words extracted from the Naab dataset (Sabouri et al., 2022), cross-checked against the Moein and Dehkhoda dictionaries for improved accuracy and coverage.
- **Visualization**

Includes a WordCloud class offering an easy way to create visually rich Persian word clouds. It supports reshaping and right-to-left rendering, Persian fonts, color maps, and custom shape masks for accurate and elegant visualization of word frequencies.

Figure 1: Word cloud visualization of selected words from Ferdowsi's Persian epic, the *Shahnameh*, arranged within the outline of Iran.

Efficiency and Reliability

Shekar is designed with both performance and robustness in mind. All models are 8-bit integer quantized, exported to ONNX format, and executed using onnxruntime to minimize dependencies and enable efficient CPU inference on a wide range of hardware, including low-resource systems.

The codebase is extensively tested across macOS, Linux, and Windows environments, with continuous integration workflows ensuring consistent behavior on all platforms. Unit tests cover more than 95 percent of the codebase, helping maintain reliability and stability as the toolkit evolves.

Availability

Shekar is distributed as a Python package on PyPI under the MIT License, with full source code, test suite, documentations, pre-trained models, example scripts, and Jupyter notebooks available on https://github.com/amirivojdan/shekar.

Acknowledgement

The author sincerely thanks Dr. Shaghayegh Yaraghi and Siavash Alizadeh for their continued support, which provided encouragement throughout the development of this work.

References

Amirivojdan, A. (2025). *Naseza: A large-scale dataset for Persian hate speech and offensive language detection* (Version v1.0.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.17355123

Eslami, M., Atashgah, M. S., Alizadeh, L., & Zandi, T. (2004). Persian generative lexicon. *The First Workshop on Persian Language and Computer. Tehran, Iran*.

Farahani, M., Gharachorloo, M., Farahani, M., & Manthouri, M. (2021). Parsbert: Transformer-based model for Persian language understanding. *Neural Processing Letters*, *53*(6), 3831–3847. https://doi.org/10.1007/s11063-021-10528-4

Jafari, S., Farsi, F., Ebrahimi, N., Sajadi, M. B., & Eetemadi, S. (2025). DadmaTools V2: An adapter-based natural language processing toolkit for the Persian language. *Proceedings of the 1st Workshop on NLP for Languages Using Arabic Script*, 37–43.

Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv Preprint arXiv:1808.06226*. https://doi.org/10.48550/arXiv.1808.06226

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv Preprint arXiv:1909.11942*. https://doi.org/10.48550/arXiv.1909.11942

Mohtaj, S., Roshanfekr, B., Zafarian, A., & Asghari, H. (2018). Parsivar: A language processing toolkit for Persian. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (Lrec 2018)*.

Rasooli, M. S., Safari, P., Moloodi, A., & Nourian, A. (2020). The Persian dependency treebank made universal. *arXiv Preprint arXiv:2009.10205*. https://doi.org/10.48550/arXiv.2009.10205

Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, 1–20. https://doi.org/10.1002/9780470689646.ch1

Sabouri, S., Rahmati, E., Gooran, S., & Sameti, H. (2022). Naab: A ready-to-use plug-and-play corpus for Farsi. *arXiv Preprint arXiv:2208.13486*. https://doi.org/10.22034/jaiai.2024.480062.1016