

An innovative Arabic light stemmer developed using a hybrid approach

Driss Namly¹, Karim Bouzoubaa^{2,3}

¹Department of Computer Science, Mohammed V University in Rabat, Rabat, Morocco

²Mohammadia School of Engineers, Mohammed V University in Rabat, Rabat, Morocco

³Department of Computer Science, University of Roehampton, London, United Kingdom

Article Info

Article history:

Received Jun 27, 2024

Revised Nov 25, 2024

Accepted Dec 2, 2024

Keywords:

Arabic language

Large lexicon

Natural language processing

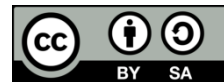
Stemming

Supervised learning

ABSTRACT

Our study introduces an innovative light stemming tool tailored for Arabic morphology challenges. In conformance with the templatic and concatenative structures, our stemmer utilizes a combination of clitic stripping, lexicon-based, and statistical disambiguation techniques to ensure accurate stemming. To accomplish this, we rely on our clitic rules lexicon to detect all potential combinations of clitics for each input entry. Subsequently, we depend on an extensive lexicon of over 7 million stems to verify the potential stems. Lastly, we employ a statistical model to ascertain the most likely stem based on the sentence's context. Experimental results demonstrate the effectiveness of the proposed stemmer in comparison with existing ones. Using different datasets, our stemmer achieves higher accuracy and F1 scores, highlighting its efficiency in Arabic stemming tasks.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Driss Namly

Department of Computer Science, Mohammed V University in Rabat,

Avenue des Nations Unies, B.P: 8007.N.U, Agdal, Rabat, Morocco

Email: d.namly@um5r.ac.ma

1. INTRODUCTION

Stemming is the process of removing prefixes, infixes, and suffixes from a word that has undergone derivation or inflection, resulting in its stem form [1]. Stemming tools can be classified as either root-based or stem-based, depending on the type of the resulting form [2]. For example, when the word “المكتبة” is stemmed using a root-based stemmer, it results in the root “كتب”, whereas a stem-based stemmer produces the stem “مكتبة”. These tools typically rely on one or more of the five main stemming approaches. Firstly, clitic stripping involves removing some clitics from words without any additional processing [3]. Pattern detection relies on linguistic rules to explain the derivation or inflection of Arabic words [4]. Lexicon-based methods use manually constructed lexicons as lookup tables to store stems or roots [2]. Statistical approaches identify word features through a training phase, using the trained model to determine the stems of new words [3]. Lastly, heavy stemming utilizes morphological analysis to extract stems or roots from input words, offering a more thorough analysis [5]–[7].

A survey of Arabic light stemming exhibits various tools, each with advantages and drawbacks. Larkey *et al.*'s Light10 stemmer [8], [9], which utilizes affix stripping, is widely used despite issues with erroneous, single, and ambiguous outcomes. Saad and Ashour [10] introduced a novel affix-removal algorithm incorporated into WEKA and RapidMiner, though it also faced problems with ambiguous single output and erroneous stemming. ARLStem [11] focuses on eliminating prefixes, suffixes, and infixes. FARASA [12] employs a support vector machine to rank potential stems but lacks diacritic marks, resulting in ambiguous output. CondLight [13] enhances Light10 with rules for definite articles and plural suffixes,

showing a 5% retrieval improvement but still struggling with erroneous stemming. Tashaphyne 0.4 [14] uses a Finite-State Automaton, producing single, ambiguous stems. Assem Arabic light stemming algorithm [15], part of the Snowball stemmer, performs clitic stripping followed by pattern matching but is prone to incorrect stemming. Morphological analyzers [5]–[7], [16]–[19] provide comprehensive diacritized forms and various features, leading to increased ambiguity by presenting all possible solutions for each word.

Despite advancements in Arabic stemming, existing stemmers encounter the challenges of erroneous stemming, ambiguous output, single outcome, and the linguistic specificities of Arabic that hinder their effectiveness. The first issue is erroneous stemming, where substrings within words are incorrectly identified as affixes or parts of stems. For instance, the word “أوساط” (OwsAT) can be inaccurately stemmed as “أوساط” (Midsts), “أوساط” (Is WASAT), or “أوساط” (Did he flagellate?). Another significant challenge is the ambiguity in output due to non-diacritized stems. This leads to multiple interpretations of a single stem. For example, the stem of the word “وجملها” (wjmlhA) is “جمل” (jml) with the prefix “و” (w) and the suffix “ها” (hA). However, the non-diacritized stem “جمل” can refer to various meanings, including “جمل” (Camel), “جمل” (Be comely), “جمل” (Make it pretty), or “جمل” (Sentences). Furthermore, many stemmers produce only a single outcome, disregarding the linguistic reality that words can possess multiple stems. For instance, the word “أولاد” (OwlAd) can be stemmed as the plural “أولاد” (Children), the noun “أولاد” (Was he born from?), or the verb “أولاد” (And did he quarrel with him?). Lastly, the unique characteristics of the Arabic language, including the lack of capitalization for proper nouns and the absence of clear rules for broken plurals, further diminish the effectiveness of current stemming algorithms.

Our research aims to create a novel, precise, and error-free Arabic light stemmer that addresses the limitations of existing stemming algorithms. Previous approaches often struggle with ambiguous outputs and tend to provide a single-stem outcome, which does not reflect the linguistic richness of the Arabic language. Our proposed stemming approach leverages a deep morphological understanding of Arabic words to overcome these challenges. This method generates all potential stems for a given word, allowing for a more comprehensive analysis. Following this, a comprehensive stems lexicon verifies the suggested stems, and a statistical algorithm evaluates the context to determine the most likely stem, ensuring that the output is accurate and contextually relevant.

The implementation of our proposed stemming approach has demonstrated significant improvements. Our developed stemmer effectively identifies all possible diacritized stems, with the first stem being the most probable based on the context. This advancement directly addresses the shortcomings of existing stemmers by minimizing ambiguous outputs and providing multiple stem options, thereby enhancing the overall accuracy and reliability of Arabic stemming.

2. PROPOSED APPROACH

Our Arabic light stemmer (ALStemmer) is structured into three distinct stages, as illustrated in Figure 1. The stemming process begins with a preprocessing including tokenization, normalization, and vocabulary generation. In the initial stage, potential clitics are eliminated from words based on a predefined clitic rules lexicon, giving rise to the candidate stems. The list of candidate stems is then validated in the second stage using a large stems lexicon. The final stage focuses on resolving the ambiguity of the valid stems by employing a statistical algorithm to determine the most probable one within the given context.

2.1. Phase 1: rules-based phase

In this phase, our primary goal is to leverage grammatical rules to extract all possible stems for each word. By considering multiple potential stems, we effectively address the limitation of singular outcomes observed in other stemming methods. This approach aligns with the linguistic reality that Arabic words can have multiple valid stems. Furthermore, we employ a clitic stripping technique to enhance the efficiency of the stemming process. Unlike other stemming algorithms, this approach significantly reduces processing time, making the process more effective and practical for real-world applications.

Arabic concatenative morphology is defined by the formation of words through the agglutination of a sequence that includes a proclitic, a stem, and an enclitic. In this structure, the proclitic attaches before the stem, while the enclitic is positioned after the stem. Both proclitics and enclitics can exist in atomic forms or as combinations. When two or more atomic proclitics (or enclitics) are combined, they create a single combined proclitic (or enclitic). For instance, the combined proclitic “أسن” (do - will) is formed from the atomic proclitics “أ” (do) and “سن” (will).

In this phase, the stemmer first tokenizes and normalizes the input text. For each vocabulary entry, the stemmer exploits the “clitics lexicon” to identify all potential combinations of clitics attached to the word. This lexicon provides a set of candidate stems based on the identified clitics. The lexicon of clitic rules includes 12 atomic proclitics modeled using 9 grammatical rules, along with 14 atomic enclitics defined by 6 corresponding rules. The application of these clitic rules results in a total of 94 proclitics and 73 enclitics,

encompassing both atomic and combined forms. Table 1 provides examples of the generated atomic and combined proclitics and enclitics and their usage, composition, and features.

To illustrate this process, let's consider the input word “أجله” (Ojlh). By applying the clitic identification rules, the stemmer generates four potential segmentations (هـ+أجل+هـ, أوجل+هـ, أوجل+هـ, أوجل+هـ). From these segmentations, the stemmer identifies the potential clitics “أ-” and “-هـ” and extracts the probable stem “جـل” (jl) based on the segmentation “هـ+أجل+هـ = أوجل+هـ”.

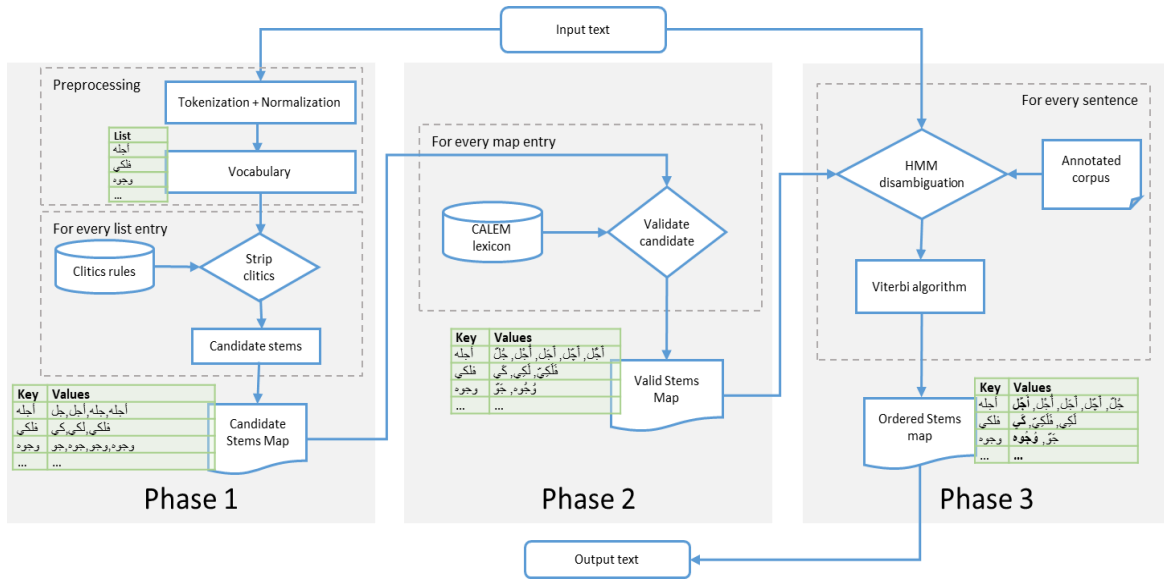


Figure 1. Proposed architecture for ALStemmer

Table 1. Samples of atomic and combined clitics

Clitic	Type	Example	Composition	Features
ب	Proclitic	قَطَعْتُ بالسَّكِّينِ	ب حرف الجر	يجر الكلمة
With		I cut with the knife	With Preposition	Puts the word in the genitive case
كُنَّا	Enclitic	إِخْمَلَا حَقِيْبَتَيْكُمَا	كُنَّا ضمير المخاطبين	للمذكر والمؤنث، المثني، المخاطب
Your		Carry your two bags	Your Addressees pronoun	feminine, masculine, dual, 2nd person
نِيْنَهَا	Enclitic	الْكُوْسُ الَّتِي أُسْقِيْتُمْوْنِيْنَهَا	نون الوقاية+ياء المتكلم	المذكر والمؤنث، المفرد، المتكلم
it to me		The cups you gave it to me	Prevention nwn and speaker yA'	feminine, masculine, singular, 1st person
			ها ضمير الغائبة	المؤنث، المفرد، الغائب
			it Absent pronoun	feminine, singular, 3rd person

2.2. Phase 2: Lexicon-based phase

The second phase of our stemmer is crucial in ensuring the validity of the candidate stems identified in the previous phase. We effectively address erroneous and ambiguous stemming concerns using a comprehensive lexicon of valid and diacritized stems. To achieve this, we rely on comprehensive Arabic LEMmas (CALEM) [20], our large lexicon of Arabic stems and their corresponding lemmas. The initial set of candidate stems produced in the preceding phase is used to authenticate the stems by checking their presence in the “CALEM lexicon”. If a candidate stem is found in CALEM, it is considered valid. Conversely, if a candidate stem is absent in CALEM, it indicates that the segmentation resulting in this stem is invalid. For instance, by checking the probable stem “جـل” (jl) in CALEM, we obtain two valid and diacritized stems: “جَلْ، جُلْ” (Be majestic, most of).

CALEM was constructed using a database comprising the most commonly used Arabic verbs, consisting of 24,171 verbs generated from Arabic roots. After conjugating these verbs, derived nouns were obtained by applying the derivation process to all verbal categories. The lexicon was further enriched with Arabic particles and non-derived nouns, such as proper nouns and broken plurals, to encompass all Arabic language specificities. As a result, CALEM incorporates 166,963 lemmas depicted by 7,133,106 stems in their diacritized form. This comprehensive approach helps prevent language specificity and ambiguous output shortcomings during the stemming process.

2.3. Phase 3: statistical phase

In the statistical stemming phase, we resolve ambiguities by selecting the most appropriate stem from a list of valid stems based on the sentence's context. This process employs a supervised learning method to identify the best stem for each input word, considering its surrounding words. To accomplish this task, we can use a generative model, such as the “hidden Markov model (HMM)” or “long short-term memory (LSTM)” networks. Recent research [21] has indicated that HMMs are simpler and more transparent compared to LSTMs, making them effective for approximating the performance of LSTMs. This simplicity allows for more efficient training and can improve overall performance. Therefore, we implement an HMM for our statistical stemming process.

In our HMM model, “observed states” correspond to the words in the input sentence, while “hidden states” represent the potential stems identified during the second phase of the stemming process. For example, as shown in Figure 2, if the observed state is “سر” (sr), the hidden states could include “سير” (walk), “سُر” (umbilical cord), “سُرّ” (be happy), “سَرّ” (delight), and “سِرّ” (secret). In a more formal way, to find for the sentence $Ph = (w_1, w_2, \dots, w_n)$ the most probable sequence of stems $(s_1^*, s_2^*, \dots, s_n^*)$, the HMM model $\lambda = (S, A, B, \pi)$ admits the following parameters: $S = \{s_1, s_2, \dots, s_m\}$ the set of stems in the Arabic language, $a(i, j)$ the probability for a stem s_i to be followed by the stem s_j , $b_i(t)$ the probability for the word w_t to give the stem s_i , and π_i the probability for Ph to start with the stem s_i . The elements of matrices A, B, and π are defined by equations (1), (2), and (3) as follows:

$$a(i, j) = \frac{n_{ij}}{n_i} \text{ with } 1 \leq i \leq N, 1 \leq j \leq N \tag{1}$$

$$b_i(t) = \frac{m_{it}}{n_i} \text{ with } 1 \leq i \leq N, 1 \leq t \leq n \tag{2}$$

$$\pi_i = \frac{n_{i0}}{M} \tag{3}$$

where the model parameters are estimated using a training corpus C composed of N words and M sentences, n_{ij} is the occurrence number in C of the stem s_i followed by the stem s_j , n_i is the occurrence number in C of the stem s_i , m_{it} is the occurrence number in C of the word w_t associated with the stem s_i , and n_{i0} is the occurrence number in C of sentences starting with the stem s_i .

To refine our model, we apply the “absolute discounting smoothing technique”, which helps adjust the elements of the matrices that may have been estimated as zero. Finally, we utilize the “Viterbi algorithm” to find the best sequence of hidden states (stems) that correspond to the observed states (words) in the input sentence. This algorithm efficiently determines the most likely sequence of stems, ensuring a contextually appropriate and accurate output.

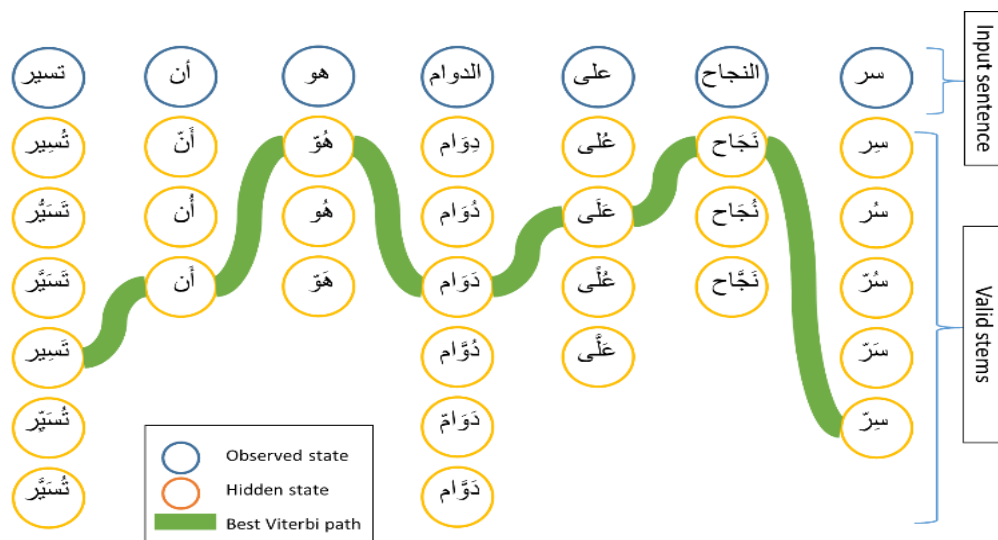


Figure 2. The disambiguation phase of the stemmer

3. EXPERIMENTS AND RESULTS

This section presents two experiments. The first experiment compares ALStemmer and other state-of-the-art stemmers to evaluate their efficiency. The second experiment showcases that utilizing ALStemmer is the optimal option for retrieval tasks.

3.1. Efficiency experiment

To validate the effectiveness of ALStemmer, we conducted a series of experiments in which we meticulously compared its performance. We aim to demonstrate its efficiency and superiority over existing stemmers. To ensure a comprehensive analysis, we utilize the most reliable data sets among the available ones.

- The normalized Arabic fragments for inestimable stemming (NAFIS) [22] is a corpus used to evaluate the effectiveness of Arabic stemmers. It encompasses a comprehensive collection of Arabic clitics covering all possible combinations. Each word in the corpus is manually annotated with multiple potential stems and roots, with the initial annotation indicating the correct solution within the sentence's context.
- The Al-Mushaf-corpus (AMC) [23] is a compilation of the Quranic text enriched with morphological tags. It contains 77,883 words manually annotated with the stem tag.

We note the availability of another corpus called “The golden Arabic corpus” [24] for assessing Arabic stemmers. However, its manual verification demonstrates its limitations such as “استفتي، بإخراج، ببوابته، “فدرس، ببيتهم، وأطفالهم (He was asked for a fatwa, he brought out, at his gate, so he studied, with their house, and their children) were stemmed as “استف، خراج، ببواب، فدرس، ببيت، طفل” (Astf, xrAj, bbwAb, then he studied, in the house, TfAl) which is inaccurate.

Therefore, our stemmer's performance is evaluated by comparing it to the most available Arabic light stemmers like ARLStem, Assem, CondLight, FARASA, Light10, Saad, and Tashaphyne. Camelira analyzer [7] and ChatGPT are added to the evaluation. Camelira analyzer is incorporated into the evaluation because it provides multiple diacritized solutions, with the most probable one determined by the sentence context, unlike light stemmers that offer a single solution without diacritics. Furthermore, in alignment with the growing trend of utilizing LLMs, a preliminary assessment was conducted to evaluate various LLMs (such as LLaMA 3 and Mixtral 8x7B) for stemming purposes. The findings revealed that ChatGPT yielded the most favorable outcomes.

To conduct the evaluation, each word in the two datasets underwent stemming using all of these stemmers and is then classified as true positive, false positive, true negative, or false negative. The evaluation metrics employed in the experiments are Accuracy and F1 score. The evaluation results using the two corpora are illustrated in Figures 3 and 4. The ALStemmer demonstrates superior performance, achieving F1 scores of 0.8660, and 0.9282, along with Accuracy values of 0.8256, and 0.8659 when utilizing NAFIS, and AMC, respectively.

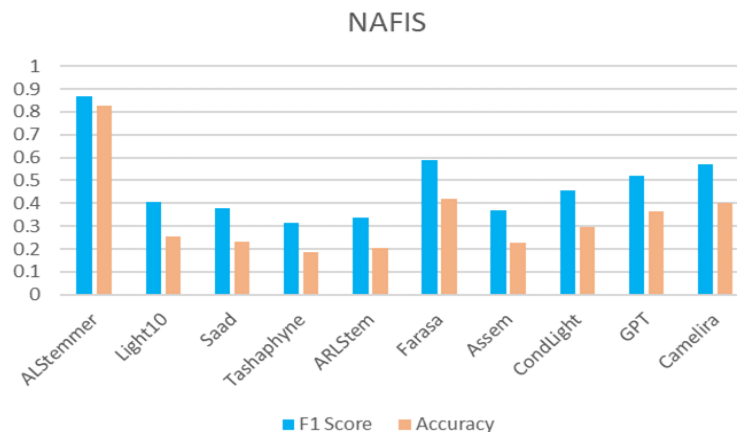


Figure 3. The F1 score and accuracy of the stemmers evaluated using the NAFIS corpus

3.2. Information retrieval experiment

Previous studies have suggested that lemmatization [25], [26] and root-based stemming [27]–[29] are better suited for retrieval tasks due to their ability to significantly reduce vocabulary size in comparison to light

stemming. Nevertheless, these methods may group words with distinct semantic meanings together, resulting in decreased precision. Consequently, our experiment aims to demonstrate that stemming, especially our light stemmer, is more effective for an information retrieval task. To achieve this, we compiled and retrieved five variations of a set of Arabic documents to assess their impact on precision and recall measures. Each variation employs different indexing terms. The initial variation of the corpus uses surface forms (words), followed by two stem variations using ALStemmer and FARASA, then lemmas, and roots variations.

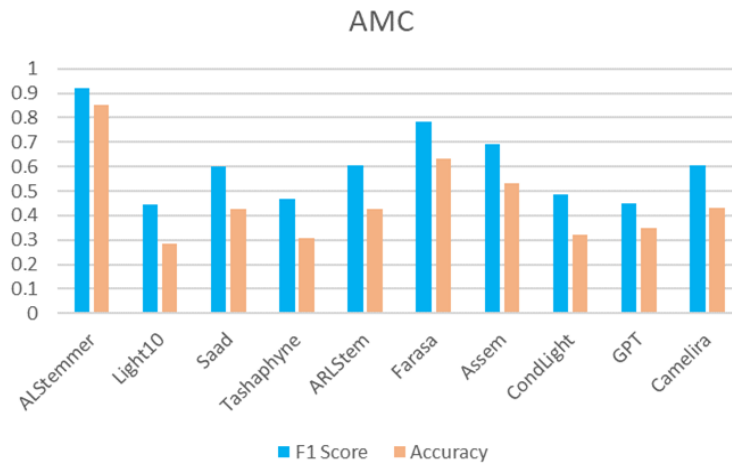


Figure 4. The F1 score and accuracy of stemmers evaluated using the Quranic Corpus

Thus, the “Arabic news articles from Aljazeera.net” dataset obtained from Kaggle is utilized, consisting of 5,870 news articles written in the Arabic language sourced from Aljazeera.net website. To generate five distinct variations of the dataset, in addition to the initial corpus composed of words, the documents are processed using ALStemmer, FARASA stemmer, Safar lemmatizer [20], and Khoja stemmer [30] to obtain variations with stems, lemmas, and roots respectively. The five corpus variations are indexed in the Elasticsearch engine, utilizing the inverted indexing method. This approach associates each token in the corpus (a word, stem, lemma, or root) with the relevant documents containing it. Subsequently, the effectiveness of the five retrieval systems is measured by analyzing their precision and recall metrics. Figure 5 demonstrates a notable enhancement in root-based retrieval compared to the word's surface form. Conversely, stem-based retrieval using ALStemmer surpasses lemma-based retrieval and exhibits a significant advantage over root-based retrieval.

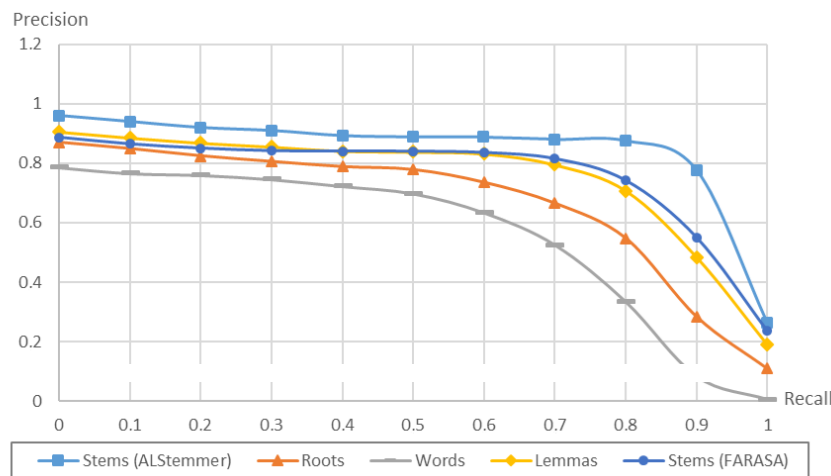


Figure 5. Precision and recall fluctuation

Additionally, it is worth noting that in the region with higher precision, the lemma-based retrieval slightly outperforms the stem-based one with FARASA. This inconsistency can be attributed to the diacritics' lack in the output produced by the FARASA stemmer. This region with higher precision and lower recall in our figure holds greater significance, as users in a Web-like medium are unlikely to read numerous retrieved documents thoroughly. Also, the performance degradation observed when using the Arabic surface form can be attributed to the numerous inflected variants of a word in the Arabic language. This abundance of variants reduces the likelihood of finding a match between the query and the documents. For example, the terms “وكتب” (And he writes), “وكتبت” (And she writes), “فكتب” (And he writes), “وكتبه” (And he writes it), “وكتبها” (And he writes it) represent variations of the word “كتب” (To write), yet classified as distinct words.

4. CONCLUSION

The discussion on Arabic stemming encompasses an exploration of Arabic morphology, various stemming approaches, and the introduction of a novel light stemming algorithm. The basis for our stemming technique is defined by the templatic and concatenative features that characterize the structured nature of Arabic morphology. The proposed light stemming algorithm presents a three-stage process: clitic removal, stem validation, and statistical disambiguation. Experiments conducted to evaluate Arabic stemmers demonstrate that ALStemmer effectively identifies stems based on context, addressing limitations observed in existing stemmers. The stemmer consistently achieves higher accuracy and F1 scores through rigorous analysis across different datasets, affirming its efficiency and effectiveness in Arabic stemming tasks.

In the future, we aim to enhance our stemmer in two primary ways: expanding the stem/lemma lexicon to include missing lemmas like named entities, and improving context detection to reduce deficiencies in the stemmer. These improvements will not only enhance the accuracy of our system but also contribute to a deeper understanding of language nuances, ultimately leading to better outcomes in various natural language processing (NLP) applications.




REFERENCES

- [1] M. Y. Dahab, A. I. Al Ibrahim, and R. Al-Mutawa, “A comparative study on Arabic stemmers,” *International Journal of Computer Applications*, vol. 125, no. 8, pp. 38–47, 2015, doi: 10.5120/ijca2015906129.
- [2] M. Mustafa, A. S. Eldeen, S. Bani-Ahmad, and A. O. Elfaki, “A comparative survey on Arabic stemming: approaches and challenges,” *Intelligent Information Management*, vol. 09, no. 02, pp. 39–67, 2017, doi: 10.4236/iim.2017.92003.
- [3] S. Memon, G. A. Mallah, K. N. Memon, A. Shaikh, S. K. Aasoori, and F. U. H. Dehraj, “Comparative study of truncating and statistical stemming algorithms,” *International Journal of Advanced Computer Science and Applications*, no. 2, pp. 563–568, 2020, doi: 10.14569/ijacsa.2020.0110272.
- [4] A. Wulandari, K. Rahmat SW, and A. Romadhony, “Pattern-based stemmer analysis and implementation on Arabic Text,” *Seminar Nasional Teknologi Informasi Komunikasi dan Industri*, pp. 31–41, 2011.
- [5] M. Boudchiche, A. Mazroui, M. O. A. O. Bebah, A. Lakhouaja, and A. Boudlal, “AlKhalil Morpho Sys 2: A robust Arabic morpho-syntactic analyzer,” *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 2, pp. 141–146, 2017, doi: 10.1016/j.jksuci.2016.05.002.
- [6] D. Taji, S. Khalifa, O. Obeid, F. Eryani, and N. Habash, “An Arabic morphological analyzer and generator with copious features,” in *Proceedings of the fifteenth workshop on computational research in phonetics, phonology, and morphology*, 2019, pp. 140–150, doi: 10.18653/v1/w18-5816.
- [7] O. Obeid, G. Inoue, and N. Habash, “Camelira: an Arabic multi-dialect morphological disambiguator,” in *EMNLP 2022 - 2022 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Demonstrations Session*, 2022, pp. 319–326, doi: 10.18653/v1/2022.emnlp-demos.32.
- [8] L. S. Larkey, L. Ballesteros, and M. E. Connell, “Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis,” *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, pp. 275–282, 2002.
- [9] L. S. Larkey, L. Ballesteros, and M. E. Connell, “Light stemming for Arabic information retrieval,” *Arabic Computational Morphology*, pp. 221–243, 2007, doi: 10.1007/978-1-4020-6046-5_12.
- [10] M. Saad and W. Ashour, “Arabic morphological tools for text mining,” in *6th International Conference on Electrical and Computer Systems (EECS'10), Nov 25-26, 2010, Lefke, Cyprus.*, 2010, vol. 18, p. 19.
- [11] K. Abainia, S. Ouamour, and H. Sayoud, “A novel robust Arabic light stemmer,” *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 29, no. 3, pp. 557–573, 2017, doi: 10.1080/0952813X.2016.1212100.
- [12] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, “Farasa: A fast and furious segmenter for arabic,” in *NAACL-HLT 2016 - 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Demonstrations Session*, 2016, pp. 11–16, doi: 10.18653/v1/n16-3003.
- [13] Y. Al-Lahham, K. Al Matarneh, and M. Hassan, “Conditional Arabic light stemmer: CondLight,” *International Arab Journal of Information Technology*, vol. 15, no. 3A Special Issue, pp. 559–564, 2018.
- [14] R. M. Al-Khatib, T. Zerrouki, M. M. Abu Shquier, and A. Balla, “Tashaphyne0.4: a new arabic light stemmer based on rhizome modeling approach,” *Information Retrieval Journal*, vol. 26, no. 1–2, p. 14, Dec. 2023, doi: 10.1007/s10791-023-09429-y.
- [15] A. Chelli, “Assem’s Arabic light stemmer (BETA),” *arabicstemmer.com*, <https://arabicstemmer.com> (accessed May 23, 2024).
- [16] K. Darwish, “Building a shallow Arabic morphological analyzer in one day,” *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2002, doi: 10.3115/1118637.1118643.
- [17] T. Buckwalter, “Buckwalter Arabic morphological analyzer version 1.0,” *Linguistic Data Consortium, University of Pennsylvania, LDC Catalog No.: LDC2002L49*. 2002.
- [18] D. Graff, M. Maamouri, B. Bouziri, S. Krouna, S. Kulick, and T. Buckwalter, “Standard Arabic morphological analyzer (SAMA)




- version 3.1,” *Linguistic Data Consortium LDC2009E73*, pp. 53–56, 2009.
- [19] A. Boudlal, A. Lakhouaja, A. Mazroui, A. Meziane, M. Bebah, and M. Shoul, “Alkhalil morpho sys1 : A morphosyntactic analysis system for Arabic texts,” *International Arab conference on information technology*, no. January 2010, 2017.
- [20] D. Namly, K. Bouzoubaa, A. El Jihad, and S. L. Aouragh, “Improving Arabic lemmatization through a lemmas database and a machine-learning technique,” *Studies in Computational Intelligence*, vol. 874, pp. 81–100, 2020, doi: 10.1007/978-3-030-34614-0_5.
- [21] L. Liu, Y.-C. Lin, and J. Reid, “Improving the performance of the LSTM and HMM model via hybridization,” *arXiv preprint arXiv:1907.04670*, 2019.
- [22] D. Namly, R. Tajmout, K. Bouzoubaa, and L. Abouenour, “NAFIS: A gold standard corpus for Arabic stemmers evaluation,” in *Proceedings of the 28th International Business Information Management Association Conference - Vision 2020: Innovation Management, Development Sustainability, and Competitive Economic Growth*, 2016, pp. 1868–1877.
- [23] I. Zeroual and A. Lakhouaja, “A new Quranic Corpus rich in morphosyntactical information,” *International Journal of Speech Technology*, vol. 19, no. 2, pp. 339–346, 2016, doi: 10.1007/s10772-016-9335-7.
- [24] linuxscout, “The golden Arabic corpus,” GitHub, Accessed: May 23, 2024. [Online]. Available: <https://github.com/ibnmalik/golden-corpus-arabic>
- [25] H. Mubarak, “Build fast and accurate lemmatization for Arabic,” in *LREC 2018 - 11th International Conference on Language Resources and Evaluation*, 2019, pp. 1128–1132.
- [26] M. Boudchiche and A. Mazroui, “A hybrid approach for Arabic lemmatization,” *International Journal of Speech Technology*, vol. 22, no. 3, pp. 563–573, 2019, doi: 10.1007/s10772-018-9528-3.
- [27] M. N. Al-Kabi, S. A. Kazakzeh, B. M. Abu Ata, S. A. Al-Rababah, and I. M. Alsmadi, “A novel root based Arabic stemmer,” *Journal of King Saud University - Computer and Information Sciences*, vol. 27, no. 2, pp. 94–103, 2015, doi: 10.1016/j.jksuci.2014.04.001.
- [28] N. Thalji, N. A. Hanin, S. Al-Hakeem, W. B. Hani, and Z. Thalji, “A novel rule-based root extraction algorithm for Arabic language,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 10, pp. 120–128, 2018, doi: 10.14569/IJACSA.2018.091015.
- [29] B. Azman, “Root identification tool for Arabic verbs,” *IEEE Access*, vol. 7, pp. 45866–45871, 2019, doi: 10.1109/ACCESS.2019.2908177.
- [30] K. Shereen, “Khoja stemmer,” *Pacific University Oregon*. <http://zeus.cs.pacificu.edu/shereen/research.htm#stemming> (accessed May 23, 2024).

BIOGRAPHIES OF AUTHORS



Driss Namly    is an assistant professor of Computer Science at Mohammed V University in Rabat, Morocco. Prof Namly received his Ph.D. degree in Computer Science from Mohammed V University in 2020. His research interests include artificial intelligence, especially natural language processing. He can be contacted at email: d.namly@um5r.ac.ma.



Karim Bouzoubaa    is a full professor of computer science in the Department of Computer Science, Mohammadia School of Engineers, Mohammed V University in Rabat. Prof/Dr. Karim Bouzoubaa received his Ph.D. degree in 1998 from Laval University. His research interests include artificial intelligence, data science, natural language processing, and computational linguistics. He can be contacted at email: karim.bouzoubaa@emi.ac.ma.