Journal of Computer Science, Information Technology and Telecommunication Engineering (JCoSITTE)
Vol. 6, No. 2, September 2025, pp. 876~890
ISSN: 2721-3838, DOI: 10.30596/jcositte.v6i2.25562

Arabic NLP: A Survey of Pre-Processing and Representation Techniques

Hussein Ala'a Alkaabi[1], Ali Kadhim Jasim[2], Ali Darroudi[3]
[1]Department Ministry of Eduaction, Najaf Ashraf, Iraq
[2]Department of Computer Engineering, Imam Ja'far al Sadiq University, Iraq
[3]Department of Electrical Engineering, Sadjad University of Technology, Mashhad, Iraq

ABSTRACT

The rapid growth of Arabic Natural Language Processing (NLP) has underscored the vital role of upstream tasks that prepare raw text for modeling. This review systematically examines the key steps in Arabic text pre-processing and representation learning, highlighting their impact on downstream NLP performance. We discuss the unique linguistic challenges posed by Arabic, such as rich morphology, orthographic ambiguity, dialectal diversity, and code-switching phenomena. The survey covers traditional rule-based and statistical methods and modern deep learning approaches, including subword tokenization and contextual embeddings. Special attention is given to how pre-trained language models like AraBERT and MARBERT interact with pre-processing pipelines, often redefining the balance between explicit text normalization and implicit representation learning. Furthermore, we analyze existing tools, benchmarks, and evaluation metrics, and identify persistent gaps such as dialect adaptation and Romanized Arabic (Arabizi) processing. By mapping current practices and open issues, this review aims to guide researchers and practitioners towards more robust, adaptive, and linguistically-aware Arabic NLP pipelines, ensuring that the data fed into models is as clean, consistent, and semantically meaningful as possible.

Keyword : Arabic NLP, Pre-processing, Morphological Analysis, Dialectal Arabic, Deep Learning.

Corresponding Author:
Hussein Ala'a Alkaabi,
Department Ministry of Eduaction, Najaf Ashraf, Iraq.
Email : hussain.njf7@gmail.com

Article history:
Received Jul 1, 2025
Revised Jul 20, 2025
Accepted Aug 26, 2025

1. INTRODUCTION

In recent years, the rapid expansion of Natural Language Processing (NLP) applications has intensified the need for effective linguistic pre-processing techniques, especially for morphologically rich and structurally complex languages such as Arabic. With over 400 million native speakers spread across more than 24 countries, Arabic is the fourth most spoken language globally and holds immense cultural, political, and computational significance. However, the Arabic language presents unique challenges for NLP practitioners due to its intricate morphology, extensive inflection and derivation systems, diverse dialectal landscape, and distinctive orthographic features. These include the presence of optional diacritics, script ambiguity, and non-concatenative root-and-pattern morphology that distinguish Arabic from many other world languages (Salloum et al., 2018). Arabic requires more linguistically informed methods than English, where pre-processing often involves relatively straightforward tasks like tokenization, lowercasing, and stopword removal. Essential pre-processing steps include diacritic removal or restoration, morphological segmentation, stemming or lemmatization, orthographic normalization, and dialect identification. These processes are vital for mitigating lexical ambiguity, standardizing text input across various dialects and orthographies, and improving the semantic coherence of the data to be processed. Pre-processing, therefore, forms the backbone of any Arabic NLP pipeline, directly impacting the effectiveness of downstream applications like machine translation (Stahlberg, F. 2018), sentiment analysis (Alnawas, A., & Arici, N. 2019), information retrieval, and text classification (Derakhshi et al., 2024). One of the primary challenges lies in diacritic ambiguity. Arabic script frequently omits short vowels (diacritics), leading to identical consonantal sequences representing multiple meanings. For example, the consonantal root كتب can correspond to كَتَبَ (kataba, "he wrote"), كُتُب (kutub, "books"), or كُتِبَ (kutiba, "was written"). Without diacritics, embeddings must capture these multiple semantic senses or rely heavily on contextual information for disambiguation, complicating the representation learning process. Another significant hurdle is clitic agglutination and sparsity. Arabic orthography allows multiple morphemes to attach as prefixes or suffixes to a single

word. For instance, the phrase وسَيَكْتُبُونَها (wa-sa-yaktubūna-hā, "and they will write it") consists of at least five morphemes: the conjunction و (wa, "and"), the future tense marker سـ (sa-), the verb stem يكتب (yaktub, "write"), the plural suffix ون (-ūna, "they"), and the object pronoun ها (-hā, "it"). Whole-word embedding approaches face extreme sparsity due to the combinatorial explosion of such forms. At the same time, subword models must learn consistent and linguistically informed segmentation to preserve semantic integrity and avoid fragmentation (Muaad et al., 2022). Further complicating Arabic NLP is dialectal synonymy and out-of-vocabulary (OOV) effects. Arabic dialects such as Egyptian, Levantine, Gulf, and Maghrebi vary substantially from Modern Standard Arabic (MSA) in vocabulary, phonology, and syntax. For example, the word for "car" is سيارة (sayyāra) in MSA, but commonly عربية (ʿarabiyya) in Egyptian Arabic. Dialect-specific tokens often appear rarely or not in standard MSA corpora, causing OOV issues and distributional mismatches in pre-trained models trained predominantly on MSA data. Addressing this requires explicitly incorporating dialectal data and dialect-sensitive pre-processing techniques (Alotaiby et al., 2014).

Historically, Arabic NLP has lagged behind English and other major languages, primarily due to the scarcity of annotated corpora, linguistic resources, and language processing tools tailored to Arabic's complexities. Early research predominantly relied on rule-based systems crafted from linguistic expertise (Diab et al., 2007). However, recent advances in machine learning, intense learning, and the emergence of Arabic-specific pre-trained language models such as AraBERT, MARBERT, and Arabic-GPT2, have significantly propelled the field forward (Abdul-Mageed et al., 2020; Antoun et al., 2020). These models leverage subword or character-level tokenization to handle morphological complexity better and require less extensive pre-processing. Nonetheless, robust pre-processing remains critical, especially when dealing with noisy user-generated content, dialectal variation, or code-switched data, which continue to challenge current NLP pipelines. Moreover, Arabic NLP must contend with code-switching, the practice of mixing Arabic with other languages like English or French within the same utterance, which is common in informal and social media contexts. This phenomenon demands multilingual and dialect-aware pre-processing frameworks that go beyond traditional monolingual approaches (Elnagar et al., 2021).

While several studies have addressed various aspects of Arabic NLP, comprehensive surveys focusing specifically on Arabic text pre-processing remain limited. Among the few notable efforts, Guellil et al. (Guellil et al., 2021) provided a broad overview of Arabic NLP research, primarily concentrating on MSA and dialects written in Arabic script. However, such surveys often overlook less standardized but increasingly relevant forms of the language, including Classical Arabic and Arabizi (Arabic text rendered in Latin script with numerals). The present survey offers a systematic and in-depth review of Arabic text pre-processing tools and techniques in response to this gap. It encompasses traditional rule-based and statistical methods and contemporary deep learning approaches (Nafea et al., 2024). Special attention is given to the evolving role of pre-processing in pre-trained language models (PLMs), where the trade-off between extensive pre-processing and the capabilities of subword-level tokenization must be carefully considered. This work aims to inform and guide future research toward developing more robust, adaptive, and inclusive Arabic NLP systems by synthesizing recent advancements, identifying persistent challenges, and extending the scope beyond prior surveys.

2. Characteristics of the Arabic Language Relevant to Pre-Processing

Arabic presents a distinctive set of linguistic phenomena that directly impact the strategies employed in preprocessing for NLP pipelines. A thorough understanding of these intrinsic properties is critical to designing, implementing, and evaluating preprocessing tools that are effectively tailored to the complexities of Arabic text

2.1 Morphological Complexity

Arabic is renowned for its rich morphological system, characterized by a non-concatenative root-and-pattern structure. Words are typically formed by embedding roots, usually sequences of three consonants, into morphological patterns that convey grammatical and semantic nuances (Issa, 2023). For instance, the root ك-ت-ب (k-t-b) can generate words such as كَتَبَ (kataba, "he wrote"), مَكْتَب (maktab, "office"), and كِتَاب (kitāb, "book"). This system results in extensive word variability, which poses significant challenges to basic NLP tasks such as tokenization, stemming, and lemmatization. Furthermore, Arabic frequently employs clitic attachment, where conjunctions, prepositions, and pronouns are appended as prefixes or suffixes to base words, complicating segmentation and part-of-speech tagging.

2.2 Dialectal Variations

The Arabic language is not monolithic but a continuum of regional dialects, including Egyptian, Iraqi, Gulf, and Maghrebi Arabic. These dialects differ substantially from MSA and one another across lexical, phonological, and syntactic dimensions (Elnagar et al., 2021). For example, the word for "car" in MSA is سيارة (sayyāra), whereas in Egyptian Arabic it is commonly عربية (ʿarabiyya). Dialectal varieties often lack a standardized orthography and heavily utilize colloquial expressions and informal spellings. As a result, pre-processing tools that are solely trained on MSA corpora tend to perform poorly on dialectal or code-switched texts, necessitating adaptive and dialect-aware methods for accurate linguistic analysis.

2.3 Orthographic Ambiguities

Arabic script introduces unique orthographic challenges. Diacritics, which represent short vowels and other phonetic elements, are generally omitted in most contemporary writing, leading to high lexical ambiguity (Boumaraf et al., 2022). For example, the consonantal skeleton علمmay correspond to عَلِمَ (ʿalima, "he knew"), عِلْم (ʿilm, "knowledge"), or عَلَّمَ (ʿallama, "he taught"), depending on diacritic placement and lengthening of consonants. Additionally, orthographic inconsistencies—such as variations in the representation of the Hamza (ء) or the Alif (ا/أ/إ/آ)—require normalization processes during pre-processing to ensure uniformity and reduce sparsity. For instance, the name أحمد may appear as احمد without the Hamza, necessitating normalization for consistent processing.

2.4 Romanization and Arabizi

In informal digital communication, particularly on social media platforms, Arabic speakers often use Romanized Arabic or "Arabizi," transcribing Arabic words using Latin characters and numerals (Allehaiby, 2013). For example, the phrase حبيبي ("my dear") is commonly written as 7abibi in Arabizi. This phenomenon introduces substantial complexity for pre-processing pipelines since traditional Arabic NLP systems are typically designed for Arabic script and lack built-in capabilities for processing Romanized text without specialized transliteration and normalization modules.

2.5 Code-Switching

Code-switching, the alternation between Arabic and other languages such as English or French within a single utterance or sentence, is widespread in Arabic-speaking online communities (Bentahila & Davies, 1995). For instance, a user might write: أنا رايح to the market ("I am going to the market"), blending Arabic and English within one sentence. This linguistic behavior presents formidable challenges for tokenization, language identification, and syntactic parsing, as NLP systems must dynamically adapt to language shifts and mixed-language input within the same context.

2.6 Rich Syntax and Free Word Order

Arabic syntax is marked by relatively flexible word order compared to many other languages. Both Subject-Verb-Object (SVO) and Verb-Subject-Object (VSO) orders are standard (Ghomri & Souadkia, 2020). For example, both الولد قرأ الكتاب (al-walad qara'a al-kitāb, "the boy read the book") and قرأ الولد الكتاب (qara'a al-walad al-kitāb) are grammatically correct and semantically equivalent. Sentences often feature dropped subjects, implicit pronouns, and nominal sentences (sentences lacking an explicit verb). This syntactic richness adds complexity to sentence segmentation, syntactic parsing, and named entity recognition, complicating the design of robust pre-processing systems.

3. Fundamental Pre-Processing Techniques in Arabic NLP

Pre-processing constitutes the essential first stage in any Arabic NLP system. The intricacies of the Arabic language necessitate specialized techniques at each step to ensure accurate linguistic representation in downstream analysis. This section reviews the core pre-processing operations applied to Arabic texts, their methodologies, and associated challenges.

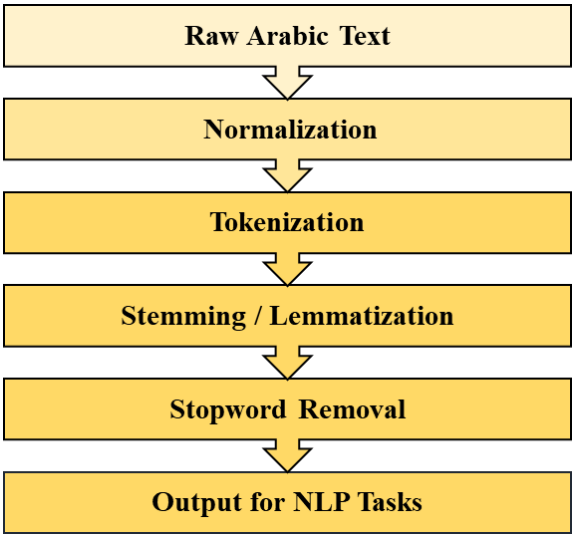

Figure 1. Arabic Pre-Processing Steps for NLP Tasks

3.1 Tokenization
Tokenization is segmenting text into meaningful units such as words, subwords, or sentences (Al-Kabbi et al., 2024). In Arabic, tokenization is particularly challenging due to clitic attachment, agglutination, and the absence of clear word boundaries in some cases (Attia, 2007). Traditional approaches to Arabic tokenization have utilized rule-based systems that manually segment conjunctions, prepositions, articles, and pronouns affixed to base words. For example, the phrase وكتبته (wa-katabtuhu, "and I wrote it") includes the conjunction و ("and") and the pronoun suffix ـه("him/it"), which need to be separated correctly into و+ كتبت+ ـه. However, these approaches often suffer from over-segmentation or incorrect disambiguation, especially in dialectal or noisy texts. Recent advances employ statistical methods, Conditional Random Fields (CRFs), and deep learning models such as BiLSTMs and Transformers. Pre-trained language models like AraBERT incorporate tokenization strategies based on subword units that partially mitigate the need for extensive manual segmentation. Nevertheless, dialectal variation and code-switching remain substantial obstacles, often requiring adaptive or multi-lingual tokenization strategies.

3.2 Normalization
Normalization involves the transformation of text to reduce orthographic variance while preserving semantic content (Chennafi et al., 2022). In Arabic, normalization typically includes:
- Unification of Alif variants: Converting آ, أ, إ to ا.
- Unification of Ya variants: Mapping ى and ئ to ي.
- Removal of diacritics: Eliminating short vowels and case endings to simplify text.
- Removal or standardization of Tatweel (-): Decorative character often used for elongation (e.g., تـطـويل) converted to تطويل.

Normalization is critical for minimizing ambiguity in lexical matching and retrieval tasks and improving model generalization. For instance, the word الإنسان may be normalized to الانسان by removing the Hamza and diacritics, ensuring consistent matching across documents. Some advanced normalization pipelines also address common typographical errors and Romanized Arabic transformations into standard Arabic script. Despite these benefits, normalization can lead to a loss of valuable morphological or phonological information, which must be considered depending on the target NLP task.

3.3 Stemming and Lemmatization
Stemming and lemmatization are critical morphological pre-processing techniques in Arabic NLP, aiming to reduce lexical variation and improve linguistic generalization. Stemming involves stripping affixes to obtain a base form, whereas lemmatization maps a word to its canonical dictionary form, or lemma. Applying these techniques in Arabic is particularly complex due to the language's non-concatenative morphology and root-pattern system (Zeroual & Lakhouaja, 2017). For example, the

words كاتب (kātib, “writer”), كتبت (katabt, “I wrote”), and مكتوب (maktūb, “written”) share the root كتب but have different morphological forms. Stemming approaches are typically classified into light stemming, which removes common prefixes and suffixes without retrieving the root (as exemplified by the Farasa Stemmer), and heavy or root-based stemming, which seeks to extract the triliteral root but often risks over-stemming and semantic distortion, as seen in tools like the Khoja Stemmer. Lemmatization, while more computationally intensive, tends to preserve semantic fidelity and is thus more suitable for tasks requiring nuanced linguistic understanding (Namly et al., 2020).

3.4 Stopword Removal

Stopword removal, a common pre-processing step in information retrieval, involves eliminating frequently occurring functional words that typically carry limited semantic weight. However, this process demands particular caution in Arabic NLP due to several linguistic complexities (Kaur & Buttar, 2018). Many Arabic stopwords, such as the conjunction و (“and”) or the preposition في ("in"), play critical syntactic roles, and indiscriminate removal can disrupt sentence structure or omit contextually relevant cues—an issue especially essential in deep learning applications where such signals may influence model performance. For example, removing وfrom the phrase وذهب إلى السوق ("and he went to the market") could affect the semantic flow and the relationship between clauses. Furthermore, diverse dialects introduce region-specific stopwords, necessitating the development of customized or domain-adapted lists. While standardized stopword sets for MSA exist, task-specific and corpus-tuned lists often yield superior outcomes in applications like sentiment analysis and social media content processing.

3.5 Part-of-Speech (POS) Tagging

Part-of-Speech (POS) tagging, the process of assigning grammatical categories such as noun, verb, or adjective to individual tokens, is a foundational component in Arabic NLP. However, POS tagging in Arabic presents unique challenges from the language’s morphological complexity. The presence of word-internal clitics, the frequent omission of diacritics, and the resulting syntactic ambiguity contribute to a highly variable tagging space, often yielding multiple plausible tag assignments for a single word (Zeroual et al., 2017). For example, the word كتب without diacritics could be tagged as a verb (كَتَبَ, “he wrote”) or a noun (كُتُب, “books”), depending on context. Traditional approaches have included rule-based systems and statistical models like Hidden Markov Models. Yet, these methods have been largely supplanted by neural sequence labeling architectures, particularly BiLSTM-CRF models trained on distributed representations from Arabic-specific embeddings such as AraVec and contextualized models like AraBERT. Achieving high POS tagging accuracy is essential for the success of downstream applications, including syntactic parsing, machine translation, and information extraction, as errors at this stage can propagate and degrade overall system performance.

4. Survey of Arabic Pre-Processing Tools

Over the past two decades, several Arabic pre-processing toolkits have been developed, reflecting the evolving methodologies in the field, from rule-based systems to data-driven and deep learning approaches. This section critically surveys the major Arabic pre-processing tools, their design philosophies, capabilities, and limitations.

4.1 Traditional Toolkits

Traditional toolkits use rule-based and statistical methods to address core Arabic NLP tasks. They offer high linguistic precision but are less adaptable to informal or dialectal text.

4.1.1 MADAMIRA

MADAMIRA, developed by (Pasha et al., 2014) in 2014, is one of the most comprehensive and widely adopted pre-processing frameworks for Arabic. Built upon statistical models trained on annotated corpora, MADAMIRA offers an integrated suite of functionalities, including tokenization, stemming, lemmatization, POS tagging, and diacritization. Its support extends to MSA and Egyptian Arabic, providing users with flexible configuration options tailored to various linguistic scenarios. MADAMIRA delivers high morphological analysis and disambiguation accuracy, positioning it as a valuable resource

for linguistically informed NLP applications. However, the toolkit's reliance on computationally intensive processes and its limited adaptability to informal or non-Egyptian dialectal varieties constrain its applicability in large-scale or heterogeneous text settings, such as those encountered in user-generated content on social media platforms.

4.1.2 Farasa

Farasa, introduced by (Abdelali et al., 2016) in 2016, represents a lightweight, high-efficiency alternative for pre-processing Arabic text. Designed with practical applications in mind, Farasa encompasses modules for tokenization, segmentation, POS tagging, and named entity recognition. Its primary advantage lies in its computational efficiency and robust performance on MSA corpora, making it particularly suitable for real-time or resource-constrained environments. Moreover, its open-source availability has contributed to widespread adoption in academic and industrial contexts. Nonetheless, Farasa's performance tends to degrade in the presence of dialectal or code-switched text, reflecting its optimization for formal Arabic and highlighting the need for further enhancements to support the full spectrum of Arabic language variation.

4.1.3 Stanford Arabic Segmenter

The Stanford Arabic Segmenter, developed as part of the broader Stanford NLP toolkit, applies a Conditional Random Field (CRF)-based sequence labeling approach to segment Arabic words into constituent base forms and attached clitics (Green & Manning, 2010). Its design enables seamless integration with other components of the Stanford NLP pipeline, making it a convenient choice for researchers employing a modular processing framework. The segmenter has demonstrated strong performance on formal texts, particularly in newswire domains where linguistic regularity is higher and annotated training data is abundant. However, its reliance on substantial annotated corpora for practical model training presents challenges in low-resource settings. Moreover, its segmentation accuracy diminishes considerably when applied to non-standard or dialectal Arabic, thereby limiting its effectiveness in handling the linguistic variability characteristic of informal or user-generated content.

4.2 Deep Learning-Based Toolkits

Deep learning-based toolkits leverage neural architectures to improve accuracy and adaptability in Arabic NLP tasks. They handle dialectal variation and complex patterns more effectively than traditional methods.

4.2.1 CAMeL Tools

CAMeL Tools, introduced by (Obeid et al., 2020) in 2020, is a comprehensive suite of deep learning-based modules designed to support a range of Arabic pre-processing tasks, including dialect identification, tokenization, POS tagging, morphological disambiguation, and named entity recognition. Unlike traditional toolkits focused primarily on MSA, CAMeL Tools strongly emphasizes dialectal variation, offering broader linguistic coverage through models trained on diverse Arabic datasets. Its modular architecture allows flexible integration into NLP pipelines, making it suitable for academic experimentation and real-world applications. Despite these advantages, the toolkit's performance remains uneven across dialects and domains, reflecting the persistent challenges of modeling under-resourced and structurally divergent varieties of Arabic. Additionally, its capabilities in handling Arabizi and code-switched texts are limited, underscoring the need for further adaptation to contemporary digital communication trends.

4.2.2 AraBERT Tokenizer

AraBERT, introduced by (Antoun et al., 2020) In 2020, a family of pre-trained transformer-based language models specifically developed for Arabic was represented. As with other BERT-style architectures, AraBERT relies on subword tokenization methods, utilizing algorithms such as SentencePiece and WordPiece to effectively manage the rich morphology and lexical sparsity inherent to Arabic. The tokenizer is a foundational pre-processing step for downstream tasks involving AraBERT-

based models, enabling robust handling of OOV and morphologically complex words through subword segmentation. This approach significantly reduces the need for traditional pre-processing techniques such as stemming or lemmatization, particularly in deep learning pipelines. However, the tokenizer is closely coupled with the pre-training objectives and tokenization scheme of AraBERT, rendering it less adaptable for tasks outside the transformer ecosystem or those requiring fine-grained morphological analysis. As such, while AraBERT has achieved state-of-the-art performance across various Arabic NLP benchmarks, its utility as a standalone pre-processing tool remains task-dependent and may require augmentation for linguistically rich or rule-based applications.

Tabel 1. Comparative overview of Arabic pre-processing tools, highlighting supported tasks, dialect coverage, modeling approach, and key strengths

Tool	Tasks Supported	MSA Support	Dialect Support	Deep Learning -Based	Open-Source	Key Strengths
MADAMIRA	Tokenization, POS, Lemmatization, Diacritization	✓	Partial (Egyptian)	✗	✗	Accurate morphological analysis
Farasa	Tokenization, POS, Segmentation	✓	✗	✗	✓	High-speed processing
Stanford Segmenter	Tokenization, Segmentation	✓	✗	✗	✓	Good integration into NLP pipelines
CAMeL Tools	Tokenization, POS, NER, Dialect ID	✓	✓	✓	✓	Modular, dialect-aware
AraBERT Tokenizer	Subword Tokenization	✓	Partial	✓	✓	Transformer model compatibility

5. Evaluation of Pre-Processing Tools

Rigorous evaluation of Arabic pre-processing tools is essential to assess their effectiveness and ensure their compatibility with diverse downstream NLP applications. Given the complexity of the Arabic language and the variety of tasks involved, evaluations must be multifaceted, encompassing intrinsic tool performance, impact on subsequent NLP tasks, and robustness across different domains and dialects.

5.1 Common Evaluation Metrics

The performance of Arabic pre-processing tools is typically assessed using standard evaluation metrics, adapted according to the specific pre-processing task:

- **Accuracy:** The proportion of correctly processed units (e.g., correctly tokenized or tagged words or the text classified as spam) relative to the total number of units (Al-Kaabi et al., 2025).
- **Precision, Recall, and F1-Score:** These are commonly used for sequence labeling tasks such as POS tagging, tokenization, and named entity recognition, balancing completeness and correctness (Al-Kaabi et al., 2025).
- **Word Error Rate (WER):** Employed especially in diacritization and orthographic normalization tasks (Guo et al., 2020).
- **Root or Lemma Accuracy:** This is specific to stemming and lemmatization evaluation, measuring the percentage of correctly identified roots or lemmas (El-Shishtawy & El-Ghannam, 2012).

Beyond these quantitative measures, qualitative evaluations, such as error analysis, are often critical to understanding the systematic weaknesses of pre-processing tools.

5.2 Evaluation Datasets and Benchmarks

A major challenge in evaluating Arabic pre-processing tools has been the lack of large, high-quality annotated datasets. However, several valuable resources are now commonly used for benchmarking. These include the Arabic Treebanks (ATB) for tokenization, POS tagging, and parsing (Maamouri et al., 2004); the Universal Dependencies (UD) Arabic corpora for standardized morphological and syntactic annotations (Nivre et al., 2016); and datasets from the OSACT shared tasks, which focus on dialectal and code-switched Arabic. Other useful resources are the Arap-Tweet corpus for social media analysis (Mubarak et al., 2014) and the QALB corpus for text normalization and error correction. Still, there is a need for more comprehensive benchmarks that address Romanized Arabic, code-switching, and noisy user-generated content.

5.3 Impact of Pre-Processing on Downstream Tasks

The choice and quality of pre-processing steps significantly impact the performance of downstream Arabic NLP tasks. Proper tokenization and normalization for sentiment analysis significantly improve feature extraction and classification accuracy, especially in informal texts (Darwish, 2013). In machine translation, morphological segmentation and lemmatization help reduce data sparsity and enhance alignment models in phrase-based statistical and neural systems (El-Khair, 2016). Named Entity Recognition (NER) benefits from correct clitic segmentation and diacritization, which improve entity boundary detection and classification (Habash & Rambow, 2005). While recent deep learning approaches often rely on minimal pre-processing and learn from subword units directly, normalization and careful tokenization still provide valuable gains, particularly when training data is limited (Darwish, 2013).

5.4 Comparative Performance Studies

To evaluate the cross-model effectiveness of Arabic pre-processing tools, we conducted a comparative study using two distinct learning paradigms: a classical Support Vector Machine (SVM) classifier and the deep learning-based AraBERT-base model. Both models were applied to the Arabic Sentiment Tweets Dataset (ASTD) (Nabil et al., 2015), a benchmark corpus for sentiment analysis in Arabic that includes a mix of MSA and dialectal content. The goal was to assess how different pre-processing tools influence performance in feature- and representation-based architectures. TF-IDF vectors were computed after applying each pre-processing tool for the SVM experiments. For AraBERT, the same pre-processing output was fed into the tokenizer of each respective tool, replacing the default subword tokenizer in some configurations for consistency. As shown in Table 3, AraBERT with its default tokenizer achieved the highest overall accuracy (92.4%) and F1-score (91.8%), owing to its contextual subword embeddings and pre-training alignment. However, when combined with tools like CAMeL or MADAMIRA, AraBERT still maintained strong performance, suggesting some resilience to input variation. In contrast, the SVM classifier demonstrated greater sensitivity to the quality of token-level input, with CAMeL yielding the best performance (87.2%) and a noticeable drop when no pre-processing was applied (81.8%). Farasa offered a favorable trade-off between accuracy and speed for both models. These results underscore that transformer models can partially mitigate pre-processing deficiencies through learned representations. However, classical models such as SVM rely more heavily on explicit linguistic structure and benefit substantially from high-quality pre-processing.

Table 3. Sentiment Analysis Performance on ASTD Dataset: SVM vs. AraBERT with Various Pre-Processing Tools

Pre-processing Tool	Accuracy (SVM)	F1-Score (SVM)	Accuracy (AraBERT)	F1-Score (AraBERT)	Remarks
AraBERT default tokenizer	—	—	**92.4%**	**91.8%**	Best overall: subword-aware contextual embedding
CAMeL Tools	87.2%	86.1%	91.2%	90.5%	Effective across both models, especially for dialects
MADAMIRA	86.7%	85.4%	90.8%	89.9%	High linguistic precision, slower inference
Farasa	85.3%	84.2%	89.7%	88.6%	Fastest, competitive performance
No Pre-processing (Raw)	81.8%	80.5%	88.2%	87.0%	Significant drop for SVM; minor for AraBERT due to subwords

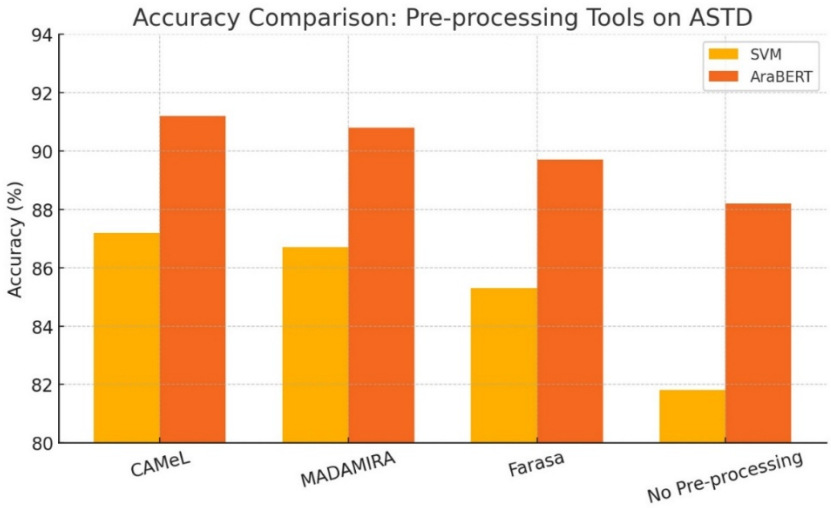

Figure 1. A comparison of classification accuracy for different Arabic pre-processing tools applied in machine learning and deep learning models

In addition to performance metrics, practical considerations such as runtime efficiency and computational complexity are critical in selecting pre-processing tools for Arabic NLP. As illustrated in Table 4, tools such as Farasa offer lightweight, high-throughput performance suitable for real-time applications. At the same time, MADAMIRA provides more linguistically grounded processing at the expense of speed, CAMeL Tools, offering modular, deep learning–based functionality that scales reasonably well. The AraBERT tokenizer, while tightly integrated into transformer pipelines and capable of subword-aware processing, requires greater computational resources and is less flexible outside deep learning contexts. These comparisons emphasize the need to balance linguistic accuracy with efficiency based on task, domain, and deployment environment.

Table 4. Runtime and Computational Characteristics of Arabic Pre-processing Tools

Tool	Avg. Speed (words/sec)	Memory Footprint	Implement ation Type	Complexity Level	Suitability
AraBERT Tokenizer	~1,000	High (GPU preferred)	Deep learning/su bword	High	Deep learning pipelines: robust, but costly
CAMeL Tools	~1,200	Medium–High	Deep learning (modular)	Medium–High	Dialect-aware applications with GPU access
MADAMIRA	~350	Medium	Statistical	High	High linguistic precision, slower in practice
Farasa	**~3,000**	Low	Rule-based/light ML	Low–Medium	Best for real-time and embedded systems
No Pre-processing	n/a	Very Low	—	None	Minimal load, but lowest accuracy in ML models

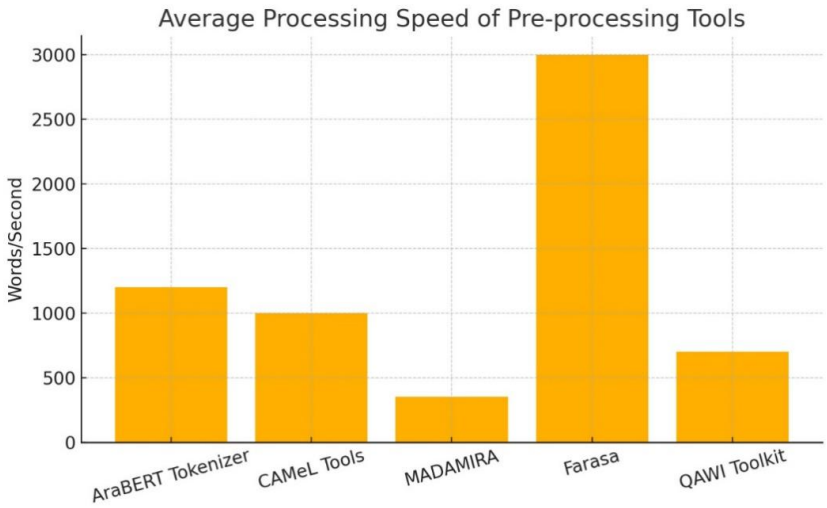

Figure 3. A general overview of the average speed and computational performance of various Arabic text pre-processing tools

Figures 2 and 3 provide a visual summary of various Arabic pre-processing tools' comparative performance and efficiency. Figure 2 illustrates the classification accuracy achieved by both SVM and AraBERT models on the ASTD sentiment analysis dataset. It shows that AraBERT consistently outperforms SVM across all pre-processing configurations, with the highest accuracy obtained using its native tokenizer. Figure 3 compares each tool's average processing speed (in words per second), highlighting Farasa as the fastest option by a wide margin. MADAMIRA is the slowest due to its computationally intensive morphological analysis. These figures underscore the trade-off between linguistic richness and computational efficiency in Arabic NLP pipelines.

This section demonstrates how Arabic preprocessing tools continue to play a central role in NLP workflows, especially in classical machine learning systems where model performance is tightly coupled to input feature quality. It also provides a clear, comparative perspective for readers choosing between deep learning pipelines and more interpretable, resource-efficient methods.

6. Challenges in Arabic Pre-Processing

Despite notable advancements, Arabic pre-processing presents significant challenges due to linguistic and sociolinguistic complexities. These include diverse dialects, morphological richness, orthographic inconsistencies, and the growing presence of multilingual and informal content. This section outlines the major issues that hinder the development of generalized, scalable, and linguistically sensitive pre-processing systems.

6.1 Dialect Identification and Processing

One of the foremost challenges is the accurate identification and processing of Arabic dialects, which deviate significantly from MSA in lexicon, morphology, syntax, and phonology. The absence of standardized orthographies for dialectal Arabic often leads to inconsistent spelling, even within the same dialect community (AlYami & AlZaidy, 2020). Furthermore, most existing linguistic resources are heavily skewed toward MSA, resulting in a persistent scarcity of annotated corpora for dialectal varieties. This lack of representation limits the training and evaluation of dialect-aware systems. Additionally, cross-dialectal variability poses a challenge to generalization, as tools trained on one dialect frequently underperform when applied to others, necessitating transfer learning and domain adaptation techniques.

6.2 Romanized Arabic (Arabizi)

Romanized Arabic, or Arabizi, introduces another major complication in Arabic pre-processing. Common in informal communication and social media, Arabizi represents Arabic words using Latin characters and numerals, without a universally accepted transliteration standard. This lack of standardization, region-specific variations, and creative spelling conventions make translating Arabic script non-trivial. Rule-based systems often fail to account for the context and ambiguity inherent in Arabizi usage (Brabetz, 2022).

6.3 Code-Switching and Multilinguality

The widespread use of code-switching, where speakers alternate between Arabic and other languages, such as English or French, complicates tokenization, parsing, and classification tasks. Code-switching is especially prevalent in North African and Gulf countries and poses substantial challenges to pre-processing pipelines that assume monolingual input (Nashef, 2013). Accurate language identification at the word or subword level is necessary but technically demanding, particularly in informal or noisy contexts. Moreover, the structural and grammatical differences between Arabic and embedded foreign languages further exacerbate segmentation and syntactic analysis. The lack of large-scale, annotated code-switched datasets limits the development of robust multilingual models capable of handling such linguistic fluidity.

6.4 Handling Noisy Texts

With the proliferation of Arabic content on social media and user-generated platforms, pre-processing systems must increasingly contend with noisy text. This includes typographical errors, emojis, unconventional abbreviations, and domain-specific slang, which are poorly represented in traditional newswire or formal text corpora. Most existing Arabic pre-processing tools were designed for clean MSA input and struggle to generalize to noisy, informal data (Elnagar et al., 2021). This mismatch highlights the pressing need for pre-processing systems that are resilient to noise and capable of adapting to the evolving nature of online communication.

6.5 Lack of Comprehensive Annotated Corpora

Another critical barrier to progress in Arabic pre-processing is the continued shortage of large-scale, high-quality annotated corpora. While resources for MSA have become more accessible, dialectal and domain-specific datasets remain scarce. Corpora are markedly absent in key application areas such as healthcare, finance, and education, and limited resources are available for complex NLP tasks, including semantic role labeling, discourse parsing, and pragmatic analysis. These limitations constrain the training, fine-tuning, and rigorous benchmarking of pre-processing tools across diverse contexts and use cases.

7. Advances and Future Trends

Recent developments in machine learning and data availability have sparked rapid progress in Arabic pre-processing, leading to more sophisticated, adaptable, and linguistically informed tools. This section highlights the most significant innovations and outlines key directions for future research.

7.1 Integration of Deep Learning in Pre-Processing Pipelines

Deep learning has played a pivotal role in transforming Arabic pre-processing workflows. Compared to rule-based approaches, models based on RNN, CNN, and Transformers have demonstrated superior performance in tokenization, segmentation, and morphological analysis. Context-aware architectures enable more precise handling of syntactic and lexical ambiguity, while end-to-end models reduce reliance on discrete pre-processing modules (Albalawi et al., 2021). However, these gains come at the cost of high data and computational requirements, which remain a limiting factor for many Arabic dialects and domains.

7.2 Pre-Trained Language Models for Arabic

PLMs such as AraBERT, MARBERT, and ARBERT have significantly influenced pre-processing Arabic text. These models operate at the subword level and can handle rich morphology without requiring extensive manual normalization or stemming (Alkaabi et al., 2025). Moreover, using zero-shot and few-shot capabilities, PLMs have shown strong performance in low-resource settings. In particular, masked language modeling techniques have proven effective for tasks such as automatic diacritization. Nevertheless, the performance of PLMs is highly dependent on the quality, representativeness, and diversity of their training data, and concerns over bias and domain imbalance remain.

7.3 Cross-Dialect and Cross-Lingual Transfer Learning

Transfer learning across dialects and languages has become an effective strategy to address the limited availability of dialect-specific resources. Fine-tuning MSA-trained models on small dialectal corpora has yielded promising results in POS tagging and segmentation tasks. Furthermore, multilingual PLMs such as mBERT and XLM-R allow knowledge transfer from high-resource languages to Arabic, improving performance on cross-lingual and dialectal tasks (El Mekki, et al., 2021). Techniques like meta-learning and few-shot learning also enable models to generalize quickly to new dialects with minimal supervision, thereby broadening the scope of pre-processing coverage.

7.4 Robust Pre-Processing for Code-Switching

Recent advances have focused on building robust models capable of handling multilingual input in response to the growing prevalence of code-switching. Neural language identification modules can detect word or subword-level language switches, enhancing segmentation and syntactic parsing in mixed-language texts. Deep learning–based transliteration systems offer improved accuracy for Arabizi conversion by leveraging contextual cues. Furthermore, joint modeling approaches that combine language identification, tokenization, and part-of-speech tagging into unified architectures represent a promising direction. However, the lack of large, annotated code-switched corpora currently hinders their development.

7.5 Ethical Considerations in Pre-Processing

As Arabic NLP tools become more widely deployed, ethical considerations in pre-processing are gaining importance. Pre-processing pipelines focusing primarily on MSA risk, introducing dialectal bias, and excluding underrepresented linguistic communities. Additionally, processing user-generated content without clear privacy safeguards raises data security and consent concerns (Arora et al., 2024). Normalization practices that erase colloquial or culturally specific language may also contribute to the marginalization of regional identities. Therefore, future research must integrate fairness, inclusivity, and transparency into the design of pre-processing systems, ensuring that they serve the full spectrum of Arabic-speaking users.

8. Conclusion

Arabic pre-processing remains a cornerstone for advancing NLP applications involving the Arabic language. This review has systematically examined the linguistic challenges inherent to Arabic, surveyed the evolution of pre-processing tools from rule-based methods to deep learning-driven architectures, and highlighted the significant influence of pre-processing choices on downstream NLP performance. Despite substantial progress, persistent issues related to dialectal diversity, orthographic ambiguity, code-switching, and informal text processing impede full automation and generalization. The field has made significant strides by integrating linguistic insights with modern machine learning innovations; however, achieving comprehensive, dialect-agnostic, and ethically responsible pre-processing solutions remains an open and critical research frontier. Future research should prioritize the development of unified, adaptable pre-processing frameworks capable of handling Modern Standard Arabic, various regional dialects, Romanized Arabic (Arabizi), and code-switched content within a single architecture. Expanding self-supervised learning techniques on massive, diverse, and representative Arabic corpora holds strong potential for reducing the reliance on annotated datasets, particularly for low-resource dialects. Moreover, establishing standardized evaluation benchmarks that include dialectal, informal, and multilingual scenarios will be essential for ensuring fair and reproducible assessments. Addressing ethical considerations such as dialect representation, bias mitigation, and user privacy must become central in future pre-processing research to create equitable and culturally sensitive Arabic NLP systems.

REFERENCES

Salloum, S. A., AlHamad, A. Q., Al-Emran, M., & Shaalan, K. (2018). A survey of Arabic text mining. Intelligent natural language processing: Trends and applications, 417-431.

Stahlberg, F. (2020). Neural machine translation: A review. Journal of Artificial Intelligence Research, 69, 343-418.

Alnawas, A., & Arici, N. (2019). Sentiment analysis of Iraqi Arabic dialect on Facebook based on distributed representations of documents. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 18(3), 1-17.

Derakhshi, M. R. F., Zafarani-Moattar, E., Al-Kabi, H. A. A., & Almarashy, A. H. J. (2024). Pclf: parallel cnn-lstm fusion model for sms spam filtering. In BIO Web of Conferences (Vol. 97, p. 00136). EDP Sciences.

Muaad, A. Y., Davanagere, H. J., Guru, D. S., Benifa, J. B., Chola, C., AlSalman, H., ... & Al-antari, M. A. (2022). Arabic document classification: performance investigation of preprocessing and representation techniques. Mathematical Problems in Engineering, 2022(1), 3720358.

Alotaiby, F., Foda, S., & Alkharashi, I. (2014). Arabic vs. English: Comparative statistical study. Arabian Journal for Science and Engineering, 39, 809-820.

Diab, M., Hacioglu, K., & Jurafsky, D. (2007). Automatic processing of modern standard Arabic text. In Arabic Computational Morphology: Knowledge-based and Empirical Methods (pp. 159-179). Dordrecht: Springer Netherlands.

Abdul-Mageed, M., Elmadany, A., & Nagoudi, E. M. B. (2020). ARBERT & MARBERT: Deep bidirectional transformers for Arabic. arXiv preprint arXiv:2101.01785.

Antoun, W., Baly, F., & Hajj, H. (2020). AraGPT2: Pre-trained transformer for Arabic language generation. arXiv preprint arXiv:2012.15520.

Elnagar, A., Yagi, S. M., Nassif, A. B., Shahin, I., & Salloum, S. A. (2021). Systematic literature review of dialectal Arabic: identification and detection. IEEE Access, 9, 31010-31042.

Guellil, I., Saâdane, H., Azouaou, F., Gueni, B., & Nouvel, D. (2021). Arabic natural language processing: An overview. Journal of King Saud University-Computer and Information Sciences, 33(5), 497-507.

Nafea, A. A., Muayad, M. S., Majeed, R. R., Ali, A., Bashaddadh, O. M., Khalaf, M. A., ... & Steiti, A. (2024). A Brief Review on Preprocessing Text in Arabic Language Dataset: Techniques and Challenges. Babylonian Journal of Artificial Intelligence, 2024, 46-53.

Issa, I. (2023). Morphological complexity in Arabic spelling and its implication for cognitive processing. Journal of Psycholinguistic Research, 52(1), 331-357.

Elnagar, A., Yagi, S. M., Nassif, A. B., Shahin, I., & Salloum, S. A. (2021). Systematic literature review of dialectal Arabic: identification and detection. IEEE Access, 9, 31010-31042.

Boumaraf, A., Bekal, S., & Macoir, J. (2022). The Orthographic ambiguity of the Arabic Graphic System: evidence from a case of Central Agraphia affecting the two routes of Spelling. Behavioural Neurology, 2022(1), 8078607.

Allehaiby, W. H. (2013). Arabizi: An Analysis of the Romanization of the Arabic Script from a Sociolinguistic Perspective. Arab World English Journal, 4(3).

Bentahila, A., & Davies, E. E. (1995). Patterns of code-switching and patterns of language contact. Lingua, 96(2-3), 75-93.

Ghomri, T., & Souadkia, M. (2020). An analytical study of word-order patterns in Standard Arabic simple sentence. RUDN Journal of Language Studies, Semiotics and Semantics, 11(1), 78-91.

Al-Kabbi, H. A., Feizi-Derakhshi, M. R., & Pashazadeh, S. (2024). A Hierarchical Two-Level Feature Fusion Approach for SMS Spam Filtering. Intelligent Automation & Soft Computing, 39(4).

Attia, M. (2007, June). Arabic tokenization system. In Proceedings of the 2007 workshop on computational approaches to semitic languages: Common issues and resources (pp. 65-72).

Chennafi, M. E., Bedlaoui, H., Dahou, A., & Al-qaness, M. A. (2022). Arabic aspect-based sentiment classification using Seq2Seq dialect normalization and transformers. Knowledge, 2(3), 388-401.

Zeroual, I., & Lakhouaja, A. (2017, April). Arabic information retrieval: Stemming or lemmatization?. In 2017 Intelligent Systems and Computer Vision (ISCV) (pp. 1-6). IEEE.

Namly, D., Bouzoubaa, K., El Jihad, A., & Aouragh, S. L. (2020). Improving Arabic lemmatization through a lemmas database and a machine-learning technique. Recent Advances in NLP: The Case of Arabic Language, 81-100.

Kaur, J., & Buttar, P. K. (2018). A systematic review on stopword removal algorithms. International Journal on Future Revolution in Computer Science & Communication Engineering, 4(4), 207-210.

Zeroual, I., Lakhouaja, A., & Belahbib, R. (2017). Towards a standard Part of Speech tagset for the Arabic language. Journal of King Saud University-Computer and Information Sciences, 29(2), 171-178.

Pasha, A., Al-Badrashiny, M., Diab, M. T., El Kholy, A., Eskander, R., Habash, N., ... & Roth, R. (2014, May). Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In Lrec (Vol. 14, No. 2014, pp. 1094-1101).

Abdelali, A., Darwish, K., Durrani, N., & Mubarak, H. (2016, June). Farasa: A fast and furious segmenter for arabic. In Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Demonstrations (pp. 11-16).

Green, S., & Manning, C. D. (2010, August). Better Arabic parsing: Baselines, evaluations, and analysis. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010) (pp. 394-402).

Obeid, O., Zalmout, N., Khalifa, S., Taji, D., Oudah, M., Alhafni, B., ... & Habash, N. (2020, May). CAMeL tools: An open source python toolkit for Arabic natural language processing. In Proceedings of the twelfth language resources and evaluation conference (pp. 7022-7032).

Antoun, W., Baly, F., & Hajj, H. (2020). Arabert: Transformer-based model for arabic language understanding. arXiv preprint arXiv:2003.00104.

Al-Kaabi, H., Al-Ibraheemi, F., Jasim, A. K., & AL-Rekabi, M. (2025). Fusion-Based Hybrid Model for SMS Spam Detection Integrating Local, Sequential, and Contextual Features.

Guo, J., Tiwari, G., Droppo, J., Van Segbroeck, M., Huang, C. W., Stolcke, A., & Maas, R. (2020). Efficient minimum word error rate training of rnn-transducer for end-to-end speech recognition. arXiv preprint arXiv:2007.13802.

El-Shishtawy, T., & El-Ghannam, F. (2012). An accurate arabic root-based lemmatizer for information retrieval purposes. arXiv preprint arXiv:1203.3584.

Maamouri, M., Bies, A., Buckwalter, T., & Mekki, W. (2004). The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In Proceedings of the NEMLAR Conference on Arabic Language Resources and Tools.

Nivre, J., de Marneffe, M.-C., Ginter, F., et al. (2016). Universal Dependencies v1: A multilingual treebank collection. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016).

Mubarak, H., Darwish, K., & Oflazer, K. (2014). Arap-Tweet: A large multi-dialect Twitter corpus for gender, age, and language variety identification. In Proceedings of the EMNLP Workshop on Arabic Natural Language Processing (ANLP).

Darwish, K. (2013). Building a shallow Arabic morphological analyzer in one day. In Proceedings of the NAACL-HLT Workshop on Arabic Natural Language Processing.

El-Khair, I. A. (2016). The impact of morphological segmentation on Arabic machine translation. Journal of King Saud University - Computer and Information Sciences, 28(1), 42-49.

Habash, N., & Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005).

Nabil, M., Aly, M., & Atiya, A. (2015, September). Astd: Arabic sentiment tweets dataset. In Proceedings of the 2015 conference on empirical methods in natural language processing (pp. 2515-2519).

AlYami, R., & AlZaidy, R. (2020, March). Arabic dialect identification in social media. In 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS) (pp. 1-2). IEEE.

Brabetz, G. (2022). Arabizi: A Linguistic Manifestation of Glocalization in the Arabic Language Area. Maydan: rivista sui mondi arabi, semitici e islamici, 2, 103-129.

Nashef, H. A. (2013). , hello and bonjour: a postcolonial analysis of Arab media's use of code switching and mixing and its ramification on the identity of the self in the Arab world. International Journal of Multilingualism, 10(3), 313-330.

Elnagar, A., Yagi, S. M., Nassif, A. B., Shahin, I., & Salloum, S. A. (2021). Systematic literature review of dialectal Arabic: identification and detection. IEEE Access, 9, 31010-31042.

Albalawi, Y., Buckley, J., & Nikolov, N. S. (2021). Investigating the impact of pre-processing techniques and pre-trained word embeddings in detecting Arabic health information on social media. Journal of big Data, 8(1), 95.

Hussein Alkaabi, Fuqdan Ibraheemi, Ali Jasim et al. Arabic SMS Spam Detection Using AraBERT and Dual Feature Extraction: A Study on Modern Standard and Iraqi Dialects, 09 June 2025, PREPRINT (Version 1) available at Research Square [https://doi.org/10.21203/rs.3.rs-6832100/v1]

El Mekki, A., El Mahdaouy, A., Berrada, I., & Khoumsi, A. (2021, June). Domain adaptation for Arabic cross-domain and cross-dialect sentiment analysis from contextualized word embedding. In Proceedings of the 2021 conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 2824-2837).

Arora, S., Thota, S. R., & Gupta, S. (2024, August). Data Mining and Processing in the Age of Big Data and Artificial Intelligence-Issues, Privacy, and Ethical Considerations. In 2024 4th Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-6). IEEE.