JKAU: Comp. IT. Sci., **Vol. 13** No. 2, pp: 1 – 15 (2024 A.D.)
DOI: 10.4197/Comp.13-2.1

Tasaheel-v2: Development of an Innovative Textual Analysis tool with Advanced Features

Hanen Himdi[1], Fatmah Assiri[2]
[1]*Computer Science and Artificial Intelligence Department, College of Computer Science and Engineering*

[2]*Software Engineering Department, College of Computer Science and Engineering*

University of Jeddah
Jeddah 21493, Saudi Arabia
hthimdi@uj.edu.sa, fyassiri@uj.edu.sa

Abstract*—We introduce Tasaheel-v2, an automated tool specifically developed for Arabic Natural Language Processing (NLP) and textual analysis tasks. This work is an extension to the first version, Tasaheel-v1, comprised of traditional NLP tasks including stemming, segmentation, normalization, named entity recognition, and part of speech tagging. Furthermore, it included cutting-edge analytic methods, such as specific emotion, polarity, linguistics, and domain-specific word tagging. In this new innovative version, Tasaheel-v2, we introduce additional benefiting utilities designed to provide assistance for the Arabic research community. We specifically integrate another Arabic-specific POS tagger, a sentiment analyzer, and English-to-Arabic translation functions. We leverage the utilities provided in Tasaheel to develop a machine learning model designed to identify Arabic phishing emails and provide a thorough textual analysis to capture deceptive cues used to detect phishing linguistic patterns. This tool contributes to the Arabic research domain by providing assistive NLP functions and textual analysis features all in one tool. We demonstrate the efficacy and feasibility of the tool's usage in assisting in the development of machine learning models that identify Arabic phishing emails. The models were trained on distinctive textual features of part-of-speech, emotion, linguistics, and domain-specific features. Through various experiments utilizing machine learning models of Support Vector Machine, Naive Bayes, Logistic Regression, and Random Forest. The generated models, namely the Random Forest, obtained the highest accuracy of 73%. In addition, we evaluate the models' aptitude for functioning in real-world cases as unseen data, by evaluating their performance on Arabic phishing emails introduced from another dataset. The RF model yielded an accuracy of 68%, suggesting a promising prospect for the detection of phishing emails. Added to that, we provide a comprehensive textual analysis to discern the distinctive language patterns that could assist in the development of models that identify phishing emails. This work demonstrates the effectiveness of Tasaheel to fill the gap and offer innovative solutions for Arabic NLP.

Keywords*—*Natural Language Processing (NLP), Textual Analysis, Tool Development, Sentiment Analysis, Translation, Machine Learning.

I. INTRODUCTION

Natural Language Processing (NLP) is an artificial intelligence approach that allows computers to understand and respond to human language. It plays a crucial role in so many fields, such as healthcare, security, and industries. NLP for Arabic language refers to the use of computational methods in Arabic script to comprehend and interpret human language. The morphological complexity of the Arabic language, which includes root-based word creation and widespread usage of derivational and inflectional morphemes, presents difficulties for Arabic NLP. However, NLP in Arabic has been the focus of research recently, despite these obstacles. It has been used in a wide range of applications, such as text classification, opinion mining, author identification, and sentiment analysis. While sentiment analysis is the process of locating and extracting subjective information from text using machine learning and NLP, it is extremely useful since it helps understand what the general public thinks or feels about certain subjects, goods, or services. It became one of the most important study areas, with applications in a variety of industries, including politics, business, tourism, education, and health. However, due to the dearth of available Arabic datasets and NLP tools employed to develop innovative models, the need to build such supportive tools is a necessity.

As there are many Arabic NLP tools available[1, 2], they are scattered on different platforms with some offering single NLP tasks [3]. Nowadays, with the advances in the artificial intelligence field, research on Arabic projects focuses on building strong models that tackle various issues such as fake news detection [4], phishing emails detection [5], and, recently, detect GPT generated text [6]. One of the main objectives of these supportive tools is to provide comprehensive assistance in compiling useful models.

In an effort to achieve this objective, we previously developed the Tasaheel-v1 [7] tool that provides NLP utilities such as stemming, normalization, tokenization, and POS tagging from various open-source packages and tools. The tool also included emotion, polarity, linguistics, and domain-specific word tagging for a thorough textual analysis. In this study, we offer a novel extension to the previous development of Tsaheel-v1, as we integrate more useful functions in terms of POS tagging, sentiment analysis, and translation from English to Arabic options. Tasaheel's aim is to provide an extensive Arabic toolbox for NLP tasks and text analysis that can be customized for different types of analytical studies [7].To demonstrate the facilities of Tasaheel, we employed its utilities to assist in developing innovative machine learning models that detect Arabic phishing emails. The textual features extracted through this tool were employed to train the machine learning models and provide a thorough textual analysis that demonstrates the linguistic patterns in both phishing and non-phishing emails.

The remainder of this work is structured as follows: Section 2 covers Arabic NLP tools, Section 3 explains Tasaheel-v2 new integrated features in detail, and Section 4 shows case studies of utilizing the new integrated features in Tassheel-v2 and provides a discussion of results. Lastly, we conclude the paper in Section 5.

II. Related Work

In NLP, textual analysis is a fundamental task. It encompasses various subtasks such as sentiment analysis, named entity recognition, part-of-speech tagging, and more. With the proliferation of Arabic text on digital platforms, the need for effective textual analysis tools for Arabic has grown substantially. This literature review aims to provide an overview of existing single and multiple-task textual analysis tools designed specifically for Arabic.

A. *Single Task Tools for Arabic Textual Analysis*

A comprehensive framework for sentiment analysis tailored specifically for Standard Arabic text that is referred to as SentiArabic was introduced in a study by Eskander [8]. This tool is a lightweight lexicon-based sentiment analyzer that operates efficiently without the need for computationally intensive processes. Instead, it utilizes morphological lookup to establish connections with the lexicon, thus circumventing the necessity for intricate morphological analyses. Sentiment assignment within SentiArabic is facilitated through a streamlined decision tree approach based on polarity scores, as opposed to more complex machine learning techniques reliant on lexical data. Notably, special consideration is given to effectively handling negations. In sum, SentiArabic utilizes lexicon-based approaches along with machine learning techniques to classify Arabic text into positive, negative, or neutral sentiment categories. When evaluated against a blind test set, the efficacy of SentiArabic and its performance metrics achieved an F-score of 76.5 %. This achievement itself surpasses previously reported benchmarks. Also, the result obtained from the study shows that SentiArabic achieved a notable absolute improvement of 3.0% when compared with state-of-the-art systems employing deep learning methodologies.

A machine learning framework tool referred to as YAMCHA that provides basic morphological analysis, tokenization, and stemming for Arabic text was introduced by [3]. In their study, the authors leverage the YAMCHA (a tool that helps in understanding the structure of Arabic words by breaking them down into their root forms and respective affixes) by integrating Support Vector Machines (SVM) as its core algorithm. It is important to note that SVM (renowned for its robust classification capabilities) benefits from its utilization of a subset of training data, thereby demanding a substantial corpus of annotated data evaluated at the Part-of Speech (POS) level for effective system training. Using a dataset comprising 100,039 words and further partitioned into distinct training and testing sets comprising 64,608 and 35,431 words, the authors employed a comprehensive tag set encompassing 48 morphological categories that were meticulously executed for both training and testing phases. Moreover, in pursuit of optimizing automatic POS tagging performance, the YAMCHA system underwent iterative training cycles, systematically modifying the range of linguistic information incorporated during each training session. Subsequent evaluations involved subjecting newly introduced texts to rigorous testing protocols. Remarkably, the attainment of the lowest observed error rate, standing at 11.4%, was associated with a training approach wherein the preceding word of the target term was considered without regard to its POS tag. In hindsight, this outcome underscores the meticulous refinement efforts undertaken to enhance the system efficacy of YAMCHA.

B. *Multiple Task Tools for Arabic Textual Analysis*

A textual analysis tool designed for analyzing subjectivity and sentiment across diverse Arabic dialects and languages was presented in a study by [9]. This tool is referred to as SANA and offers a comprehensive and versatile lexical resource. SANA incorporates language variations from online sources like newswires, chats, Twitter, and YouTube, including Modern Standard Arabic (MSA), Egyptian Dialectal Arabic (EDA), and Levantine Dialectal Arabic (LDA), with English explanations for clarity. SANA, a sophisticated sentiment analysis tool, elevates accuracy by enriching entries with nuanced linguistic attributes like part-of-speech tags, diacritics, and gender. Its development methodology integrates meticulous manual curation, including the compilation of word lists like SIFAAT (denotes “adjectives” in Arabic extracted from Penn Arabic Treebank) and HUDA (a lexicon extracted from conversational data in Egyptian Arabic chats) with advanced automated techniques such as pointwise mutual information and machine translation. This meticulous approach guarantees comprehensive coverage and reliability, making SANA a highly effective tool across diverse linguistic contexts.

Another sophisticated system engineered for the precise task of morphological analysis and disambiguation in Arabic language processing referred to as MADAMIRA was introduced in a study by Pasha et al. [1]. This tool seamlessly merges the most efficacious attributes of its predecessors - MADA and AMIRA. It integrates various NLP tasks for Arabic text analysis, including tokenization, morphological analysis, discretization, stemming, part-of-speech tagging, and lemmatization. Also, MADAMIRA provides a unified framework for researchers and developers to perform multiple tasks seamlessly on Arabic text. Moreover, MADAMIRA represents a pinnacle of refinement in this domain, boasting a streamlined Java implementation that enhances robustness, portability, and extensibility. As demonstrated in the study, MADAMIRA achieves remarkable speed, surpassing its predecessors - MADA and AMIRA - by an order of magnitude. This advancement marks a significant milestone in Arabic language analysis, setting a new benchmark for efficiency and scholarly rigor in computational linguistics.

Taking a leaf out of MADAMIRA, a study by Darwish & Mubarak [2] introduced Farasa - an Arabic comprehensive multitask tool distinguished by its rapidity and precision. This tool is developed entirely in native Java and is free from external dependencies. As an open-source tool, the approach of Farasa is grounded in SVM rank by employing linear kernels to optimize performance. Additionally, Farasa integrates an exhaustive array of features that include the probability of stems, prefixes, and suffixes - both individually and in combination, including their presence within lexicons containing valid stems, named entities, and core stem templates. Notably, Farasa offers segmentation, normalization, name entity recognition, diacritic on words, and part-of-speech tagging.

To assist the growing demands of researchers in the intricate tasks of textual analysis and cross-corpus comparisons within Arabic contexts, a study by Nahar et al. [10] introduced the Standard Arabic Profiling (SAP) toolset. This tool is designed to meet the demand for specialized Arabic language processing, offering a refined solution aimed at simplifying the textual analysis process. It is important to note that the approach employed in crafting the tool involves three key profilers: POS, vocabulary, and readability. For example, the POS profiler provides statistical analysis of a document. The vocabulary profiler offers insights into vocabulary usage based on the Open-Source Arabic Corpus (OSAC) from CNN and BBC. On the other hand, the readability profiler assesses the

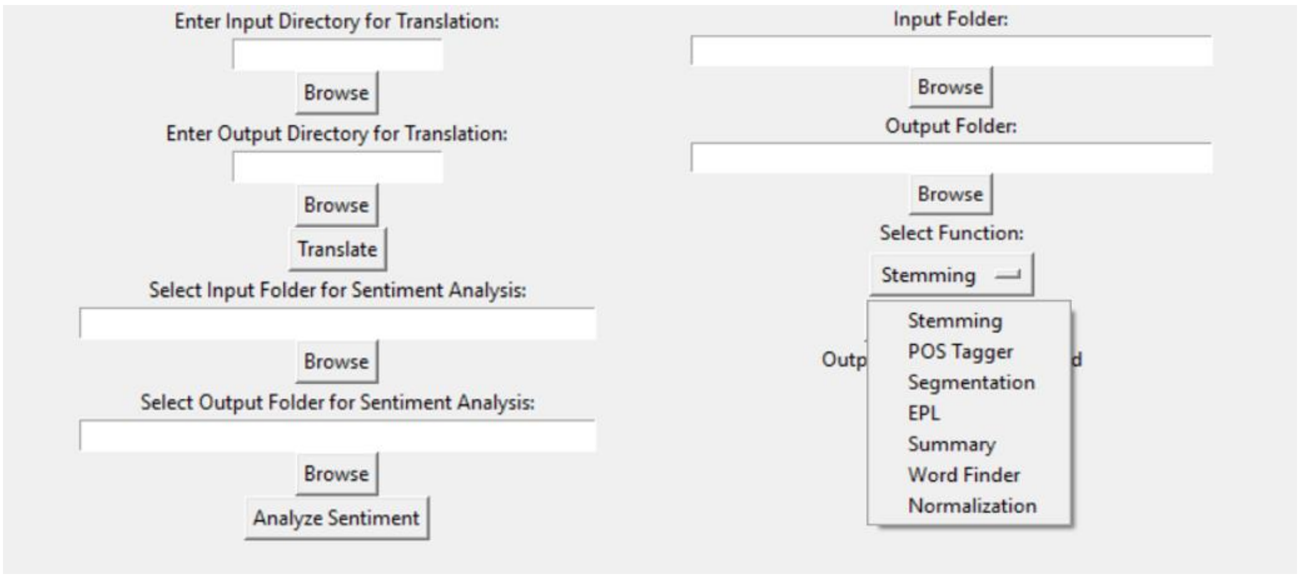


Fig. 1. Tasaheel-v2 interface

readability of the document using the Flesch Reading Ease Formula to gauge its simplicity and ambiguity.

Unlike the SAP toolset, another sophisticated open-source toolkit was introduced by [11]. Referred to as CAMeL, this tool is a Python-based ensemble of open-source utilities tailored for Arabic natural language processing. Its comprehensive suite encompasses pre-processing, morphological modeling, dialect identification, named entity recognition, and sentiment analysis functionalities. It is to be noted that CAMeL is currently in active development and continuously evolving with ongoing efforts focused on expanding its API components and command-line tools, as well as refining existing functionalities.

A monolingual Java-based framework that is meticulously designed to meet the rigorous standards of software development and tailored specifically to cater to the intricacies of the Arabic language, including both the contemporary standard Arabic and the Moroccan dialect was introduced in a study by Bouzoubaa et al. [12]. Referred to as SAFAR, this tool has evolved to encompass a diverse array of over 50 tools and resources through a decade-long process of refinement and enhancement. Notable features offered by SAFAR include text preprocessing, morphological analysis, named entity recognition (NER), sentiment analysis, machine translation, and topic modeling, including text classification, dependency parsing, and documentation support. Accessible seamlessly via its application programming interface (API) and through its intuitive web interface, this comprehensive suite of offerings positions SAFAR as a sophisticated and adaptable platform that is considered ideal for both linguistic inquiry and practical application development endeavors. In sum, SAFAR's combination of API access and web interface, along with its diverse set of tools and resources, makes it a powerful platform for Arabic language processing and analysis. Users can leverage its capabilities to perform a wide range of text-related tasks, from basic preprocessing to advanced natural language understanding and machine learning tasks.

Leading the race in Arabic NLP is Tasaheel - a tool introduced in a study by Himdi & Assiri [7]. Tasaheel is an automated Arabic textual analysis tool that supports traditional NLP tasks like stemming and POS tagging, alongside innovative features such as detailed POS tag summaries and emotion analysis. Moreover, the novelty of this tool is that it distinguishes itself by providing affix extractors for thorough textual analysis. In terms of augmenting user functionality, the tool seamlessly converts text files into Excel data, while also facilitating the precise location of specific words within designated folders.

III. TASAHEEL DEVELOPMENT

We discuss the details of developing Tasaheel's versions in two parts. In the first part, we outline the utilities in Tasaheel-v1 , and the second part thoroughly explains new innovative utilities added to form Tasaheel-v2. Fig. 1 displays the user interface of Tasaheel, which includes integrated utilities from the first and second versions. This tool adheres to legal and ethical standards by utilizing publicly available packages and tools intended for research purposes. All tools and packages integrated are properly referenced in this section. The tool can be freely offered upon request.

A. *Tasaheel-V1*

Tasaheel was developed using Python programming language, version 3.11.5. It addresses two aspects of textual analysis, which are fundamental NLP tasks and supportive functions for data analysis. Its interface is shown in Fig. 1. We briefly demonstrate the utilities included in Tasaheel-v1. and more details of the tools compilation can be found in [7].

1) *Common NLP Utilities*

Tasaheel was developed to combine several Arabic NLP functions. Some of these functions were provided by packages coded in Python and available online but were scattered in several research platforms. We aimed to collect and join several NLP packages that supported Arabic, to compile a comprehensive analysis tool. The NLP utilities available are described below and examples of their implementations are displayed in TABLE :

- **Stemming** the process of combining multiple inflected word forms into one uniform canonical form [13]. The following packages were included: Information Science Research Institute (ISRI) [14], Farasa [2], and Tashaphyne[1].
- **Segmentation** is splitting a text into comprehensible chunks, like words, phrases, or subjects [15]. Arabic text must be divided into morphemes due to its distinct morphology in order to reduce the ambiguity that results from the associated affixes. The two libraries embedded were Tashaphyne[2] and Farasa [16].
- **Normalisation** it reduces word ambiguity and removes unnecessary randomness associated with the text to unify the word variation [17]. This option provides multi normalization functions, offered by Tshaphyne [3] , which combines multiple normalization tasks in one implementation, such as the removal of stop words, and unifying unique letters. Another option is a single-normalization function, which implements a single function at an implementation, such as removing
- **Name Entity Recognition (NER)** capabilities were provided by Farasa [2] and Stanford CoreNLP [18] libraries. It entails identifying significant information in the text and categorizing it into a set of predetermined categories [19].
- **POS Tagging:** It is the process of assigning a word in a text to a certain part of speech based on both the definition it provides and the context in which it occurs [20]. The incorporated POS taggers are Farasa [2] and Stanford CoreNLP [18].

TABLE I. Examples of the existing NLP tasks output [7].

Original	تواصل السعودية جهودها الحثيثة و المستمرة في مجال الحج و العمرة و خدمة ضيوف بيت الله الحرام
Segmentation (Farasa)	تواصل ال+سعودي+ة جهود+ها ال+حثيث+ة و+ال+مستمر+ة في مجال ال+حج و+ال+عمر+ة و+خدم+ة ضيوف بيت الله ال+حرام
Stemming (ISRI)	وصل سعود جهد حث و مر في جل حج و عمر و خدم ضيف بيت الله حرم
Normalize (Tashaphyne)	تواصل السعوديه جهودها الحثيثه و المستمره في مجال الحج و العمره و خدمه ضيوف بيت الله الحرام
POS Tagging (Farasa)	تواصل/VERB السعودية/NOUN جهودها/NOUN الحثيثة/ADJ و/CONJ المستمرة/ADJ في/PREP مجال/NOUN الحج/NOUN و/CONJ العمرة/NOUN و/CONJ خدمة/NOUN ضيوف/NOUN بيت/NOUN الله/NOUN الحرام/ADJ

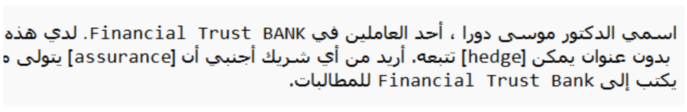
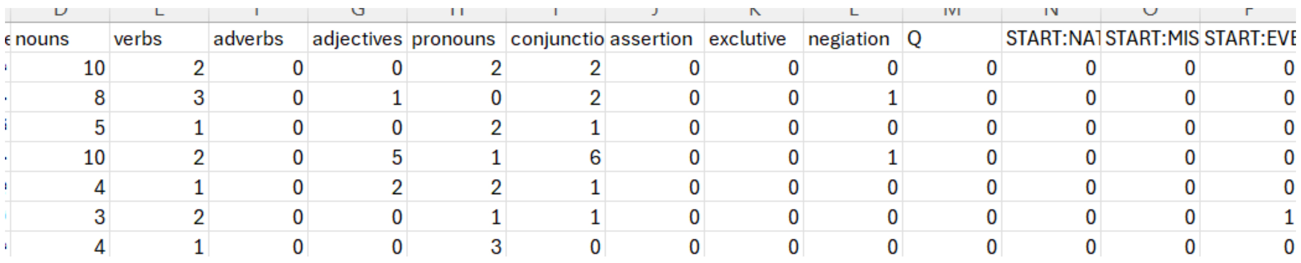

اسمي الدكتور موسى دورا ، أحد العاملين في Financial Trust BANK. لدي هذه
بدون عنوان يمكن [hedge] تتبعه. أريد من أي شريك أجنبي أن [assurance] يتولى ه
يكتب إلى Financial Trust Bank للمطالبات.

Fig. 2. Emotion, Polarity, and Linguistic tagging

2) *Emotion, Polarity, Linguistics, and Domain-Specific Word Tagger*

We developed a word tagger that tags words that fit the emotion, polarity, linguistics, and domain-

[1] https://pypi.org/project/Tashaphyne/
[2] https://pypi.org/project/Tashaphyne/
[3] https://pypi.org/project/Tashaphyne/



nouns	verbs	adverbs	adjectives	pronouns	conjunctio	assertion	exclutive	negiation	Q	START:NA	START:MIS	START:EVE
10	2	0	0	2	2	0	0	0	0	0	0	0
8	3	0	1	0	2	0	0	1	0	0	0	0
5	1	0	0	2	1	0	0	0	0	0	0	0
10	2	0	5	1	6	0	0	1	0	0	0	0
4	1	0	2	2	1	0	0	0	0	0	0	0
3	2	0	0	1	1	0	0	0	0	0	0	1
4	1	0	0	3	0	0	0	0	0	0	0	0

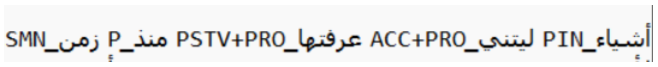
Fig. 3. Results recorded in Excel sheets

specific word categories. The tool provides an output as two files: (i) each input file displaying the matching words tagged with its category as shown in Fig. 2. (ii) A summary file for each tag matched through all the input files. Details of the wordlist compilations and the tagger's development are thoroughly explained in [7].

3) *Affix Analyzer*

To capture affixes that may pertain to the same grammatical role as the previous taggers, we provide an affix analyzer function. According to a study by [21], affixes play a significant role in changing words' meaning and, thus, the grammatical function. Extracting the affixes might be helpful for NLP projects, especially those focusing on textual analysis. We provide two options to extract affixes, i.e., to extract either prefixes or suffixes from POS tagged files.

To automate the textual analysis, we provided the option of converting all the summaries files provided from the previous taggers into Excel sheets for convenient analysis. A sample is shown in Fig. 3.



SMN_زمن P_منذ PSTV+PRO_عرفتها ACC+PRO_ليتني PIN_أشياء

Fig. 4. POS tagged text by APL

B. *Tasaheel-V2 Integrated New Features*

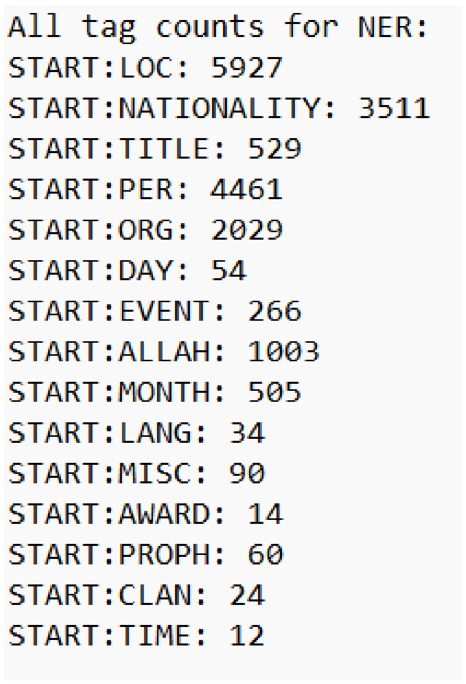
As we previously described the utilities included in Tasaheelv1, we introduce the utilities integrated in this version, Tasaheelv2. Studies on the Arabic language sometimes face a scarcity of datasets [22] in comparison to the wide range of obtainable datasets in English. Therefore, some studies such as [23] translate an available English dataset into Arabic using translation tools. For convenience, we propose a solution to this issue by configuring a translation utility that translates an input text from English to Arabic in little time. Moreover, we upgraded the polarity word tagging in the previous version of Tasaheel, by integrating a more advanced sentiment analyzer. Further, we expand the POS tagger options for the user by integrating an advanced Arabic-specific tagger. The following section explains the utilities in detail:

1) *POS and NER Tagger*

In this version, we added an Arabic-specific POS tagger called Arabic Linguistic Pipeline (APL). ALP is a novel linguistic pipeline designed for the processing of text in Modern Standard Arabic for natural language processing purposes. It addresses typical natural language processing tasks such as word tokenization, POS tagging, lemmatization, base chunking, and Named Entity Recognition (NER) by interpreting them as a unitary sequence labeling job [24]. This tool offers unique named entity recognition according to specific categories such as religion, event, country, title, organization, day, and month. Additionally, it offers POS tags that are meticulously crafted in accordance with Arabic grammar. Specifically, its tags correspond to the grammatical rules of gender (masculine and feminine) and number (singular, dual, and multiple) in the Arabic language. According to a comparative study by [25], the ALP tagger outperformed other POS taggers such as Farasa, Camel, and Madamira in evaluating data derived from several books that contain diverse narrative styles. The choice of adding this tagger is to provide wide selective options for the users accustomed to their needs. Fig. 4 demonstrates a sample of a POS and NER-tagged text with APL.

Similar to the taggers offered in Tasaheel-v1, the output generated through this option is two files: (i) each word tagged in a single file (ii) a summary file for the number of occurrences of each tag in all the input files. Fig. 5 displays a sample of a summary file that displays the number of occurrences of each tag across all the input files.

2) *Translation*



```
All tag counts for NER:
START:LOC: 5927
START:NATIONALITY: 3511
START:TITLE: 529
START:PER: 4461
START:ORG: 2029
START:DAY: 54
START:EVENT: 266
START:ALLAH: 1003
START:MONTH: 505
START:LANG: 34
START:MISC: 90
START:AWARD: 14
START:PROPH: 60
START:CLAN: 24
START:TIME: 12
```

Fig. 5. NER summary file

We integrate MarianMTModel translator as a translator option. It is an effective, free, and open-source framework for Neural Machine Translation [26], coded entirely in C++ with very few requirements. Marian is a specialized sequence-to-sequence model developed exclusively for machine translation. The system employs an encoder-decoder architecture that incorporates attention processes to provide translations across different languages. Marian possesses the exceptional ability to undergo training in numerous languages concurrently, rendering it a very adaptable model for jobs involving translation across multiple languages. The primary developers of this project are the team at Microsoft Translator. Several academic institutions, including the University of Edinburgh and previously the Adam Mickiewicz University in Poznan, as well as business entities, actively contribute to its development.

The script uses the **MarianTokenizer** from the **transformers** library for text translation, specifically employing the “HelsinkiNLP/opus-mt-en-ar” model for English to Arabic translation. This choice is significant for several reasons:

- **Accuracy:** MarianMT models are known for their high accuracy in machine translation, benefiting from extensive training on large datasets. It achieved a higher METEOR (Metric for Evaluation of Translation with Explicit Ordering) compared to the T5 [27].
- **Efficiency:** These models are optimized for performance, enabling the script to translate texts quickly, which is crucial for processing multiple files or large volumes of text.
- **Flexibility:** The script’s design to handle text in chunks addresses the model’s sequence length limitations, ensuring comprehensive translation without loss of content.
- **Speed:** According to [28] translated text at high speed compared to other translators such as Google [29].

3) *Translation Example*

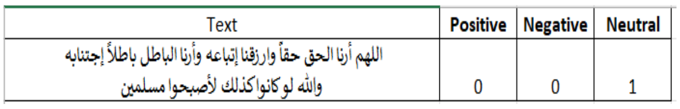
Before Translation:
Once upon a time, in a bustling city, a lonely old man discovered a forgotten library filled with magical books that brought his cherished memories to life. With each page turned, he found companionship in the characters and solace in the stories, reminding him that adventures never truly end.

After Translation:

في يوم من الأيام، في مدينة مزدحمة، اكتشف رجل مسن وحيد مكتبة منسية مليئة بالكتب السحرية التي أعادت إحياء ذكرياته العزيزة. مع كل صفحة يقلبها، وجد رفقة في الشخصيات وعزاء في القصص، لتذكره أن المغامرات لا تنتهي أبدًا

4) *Sentiment Analysis*

Sentiment Analysis is a computational technique for identifying the polarity (positive, negative, or neutral) of a text. The sentiment analyzer option is constructed by integrating the “camel_tools.sentiment,” part of the CAMeL Tools by utilizing extensive pre-trained language models [30]. It was developed by employing HuggingFace’s Transformers [31] to fine-tune multilingual BERT (mBERT) [32] and AraBERT [33] for the aim of Arabic sentiment analysis. The fine-tuning included adding a fully linked linear layer with a softmax activation function to the final hidden state. To enhance the presentation of the data's sentiments, we included the functionality to present all input statements with their corresponding sentiment scores in a single Excel sheet. The analyzer labels an Arabic statement as positive, negative, or neutral, by indicating ‘1’ to its label, as seen in Fig. 6.



Text	Positive	Negative	Neutral
اللهم أرنا الحق حقاً وارزقنا إتباعه وأرنا الباطل باطلاً إجتنابه والله لو كانوا كذلك لأصبحوا مسلمين	0	0	1

Fig. 6. Sentiment analysis result camel

IV. Case Study

With respect to the aforementioned features, we have combined the features of Tasaheel-v1 and Tasaheel-v2 to develop Tasaheel, a comprehensive functioning tool for NLP tasks. Its features are also utilized to assist in the preparation of datasets needed to train machine learning models that detect Arabic phishing emails. It would also be applied to extract viable cues to assist in the detection of deceptive text found in phishing emails.

A. *Leveraging Tasaheel for Compiling Models to Detect Phishing Emails*

We assess the dependability of Tasaheel-v2’s integrated utilities by analyzing its performance in support of compiling an ML model that classifies Arabic phishing emails. Phishing emails pose a substantial menace to individuals and companies on a global scale. These fraudulent emails have the intention of deceiving users into revealing confidential information or engaging in dangerous activities. In fact, according to the Gulf Business new paper “There were almost a million phishing attempts detected by Kaspersky in Saudi Arabia in the second quarter of 2020, the cybersecurity company says in a new report. Kaspersky’s spam and phishing report for Q2 says its systems detected 973,061 phishing attacks in the kingdom in the three months. The UAE followed with 617,347, Egypt had 492,532, Oman 193,379, Qatar 128,356, Kuwait 106,245, and Bahrain 67,581.” [4] The staggering statistics show the importance of tackling this problem. Ensuring the detection and prevention of phishing emails is essential for protecting personal and financial security. Recently, textual analysis combined with machine learning techniques has recently emerged as a possible method to address this increasing threat, by analyzing distinctive linguistic markers dominant in phishing emails’ context. We follow the main steps of compiling an ML model in terms of dataset, preprocessing, feature extraction, training, and testing and evaluation. We will demonstrate the use of Tasaheel-v2 functions in detail for each step.

B. *Dataset*

Due to the lack of available Arabic phishing datasets, we translated an open-source English dataset of phishing emails [34] into Arabic. To balance it, we included non-phishing emails from the Enron project available from Carnegie Mellon University’s School of Computer Science[5]. Both datasets had labels such as email date, sender name, and email ID. We retrieved only the “body” label, which provides the actual content of the email, as the other labels were not pertinent to our work. We utilized the integrated Marian MT translator. Through this process, 100 phishing and 100 legitimate emails were translated into Arabic. Fig. 7 displays the original email in English and its translated version in Arabic. The dataset can be accessed freely on Github [6]. The statistics for utilized datasets are presented in Table 2.

Original
Dear User We suspect an unknown access to this account. Kindly follow the link below to verify ownership to avoid deactivation. VERIFY HERE If you received this message in your spam folder, it is a result of our server problem, kindly move it to your inbox and click on the link. We apologize for the inconvenience.
Translated
عزيزي المستخدم نشتبه في وجود وصول غير معروف إلى هذا الحساب. يرجى اتباع الرابط أدناه للتحقق من الملكية لتجنب التعطيل. تحقق هنا إذا تلقيت هذه الرسالة في مجلد البريد العشوائي، فهذا نتيجة لمشكلة في خادمنا، يرجى نقلها إلى صندوق الوارد الخاص بك والنقر على الرابط . نعتذر عن الإزعاج .

[4] https://gulfbusiness.com/saudi-arabia-led-gcc-in-number-of-phishingattacks-in-q2-kaspersky-report/

[5] http://www.cs.cmu.edu/enron/

[6] https://github.com/Hanen-Tarik

TABLE II. Dataset statistics

Class	Number of Emails	Average No. of Words
Phishing	100	87.2
Non-Phishing	100	71.7

Fig. 7. Original Email and translated version

C. *PreProcessing*

Text preprocessing is a crucial step in natural language processing (NLP) that involves cleaning and converting unstructured text input to make it ready for categorization. The standard preprocessing stages were carried out utilizing Tasaheel's integrated packages.

- **Normalization** removes repeated characters and elongated words and standardize the Arabic letters: alef, alef maqsoura, and ta-marbouta. Here, the Tashaphyne package was used to apply normalization.
- **Tokenization** tokenize the textual content into separate tokens. The Farasa tokenizer was employed for this task.
- **Elimination** of diacritics, punctuation marks, and nonalphabetic characters like emojis, hashtags, emails, and web page URLs. The customized normalization utility available in Tsaheel to remove these characters was used to perform this task.

D. *Textual Features Extraction*

We extracted eight types of POS, six emotion, nine linguistic, and seven domain-specific terms from the phishing emails dataset to identify indicators linked to phishing behavior. The extraction feature is integrated into the tool as a tagger, and the results are automatically saved in an Excel file to facilitate analysis, as shown in Fig. 8

TABLE III. Model performance metrics

Model	Class	Precision	Recall	f1-score	Accuracy
RF	phishing	0.73	0.71	0.74	0.73
	non phishing	0.71	0.75	0.72	
LR	phishing	0.68	0.64	0.67	0.68
	non phishing	0.65	0.68	0.70	
SVM	phishing	0.57	0.60	0.59	0.64
	non phishing	0.64	0.67	0.69	
NB	phishing	0.67	0.59	0.62	0.65
	non phishing	0.72	0.71	0.71	

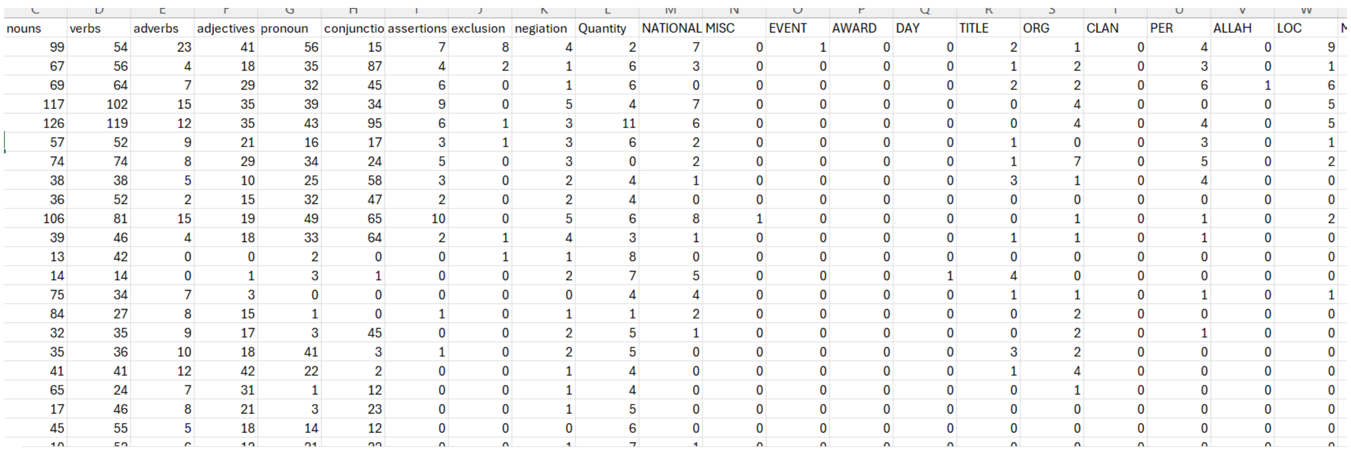
E. *Model Training and Testing*

Text Now that the dataset is clean and set. For the training phase, we applied Naive Bayes (NB), Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM) classifiers. We used the Scikit-learn library in our experiment. To validate our results, we applied 70% training 30% testing. Since the data size was small, we ran our experiment on a standard computer with 1 TB and a 7 IO dual-core processor. We report Accuracy, Precision, Recall, and F1 measures. Orange Python Toolkit version 3.31.0 was used to train and test the model. It is an open-source ML and data visualization software that enables data analysis and model evaluation options[7].

Table 3 demonstrates the results of the proposed Tasaheel-v2 in classifying Arabic phishing and non phishing emails. Despite all the classifiers showcasing reasonable performances, the RF classifier achieved the highest overall accuracy (73%). Furthermore, it achieves reasonable performance across other parameters (precision: 0.73, recall: 0.71, F1-score: 0.74), demonstrating its effectiveness in classifying both phishing and non-phishing emails in Arabic. One reason for such high performance is its ensemble nature by combining multiple decision trees, which makes RF less prone to overfitting, particularly in the presence of smaller datasets (i.e., 200 emails in our case) [35]. Additionally, RF is suitable for handling datasets with multiple features, particularly in linguistic scenarios with different POS and emotions [36].

In contrast, NB and LR exhibited similar accuracy (around 65-68%) but with a slight imbalance between classes. Table 3 showcases that NB achieved slightly better precision for phishing emails (0.67) but lower recall (0.59) compared to LR (precision: 0.65-0.68, recall: 0.64-0.68). Such results indicate that NB might better identify phishing emails but miss some phishing attempts (lower recall). Typically, NB demonstrates excellent results when dealing with independent features [37]. However, linguistic features often have some dependencies among different features that lead to a slight class imbalance in NB. In contrast, LR works best for binary classification

[7] https://orangedatamining.com/download/



nouns	verbs	adverbs	adjectives	pronoun	conjunctio	assertions	exclusion	negiation	Quantity	NATIONAL	MISC	EVENT	AWARD	DAY	TITLE	ORG	CLAN	PER	ALLAH	LOC
99	54	23	41	56	15	7	8	4	2	7	0	1	0	0	2	1	0	4	0	9
67	56	4	18	35	87	4	2	1	6	3	0	0	0	0	1	2	0	3	0	1
69	64	7	29	32	45	6	0	1	6	0	0	0	0	0	2	2	0	6	1	6
117	102	15	35	39	34	9	0	5	4	7	0	0	0	0	0	4	0	0	0	5
126	119	12	35	43	95	6	1	3	11	6	0	0	0	0	0	4	0	4	0	5
57	52	9	21	16	17	3	1	3	6	2	0	0	0	0	1	0	0	3	0	1
74	74	8	29	34	24	5	0	3	0	2	0	0	0	0	1	7	0	5	0	2
38	38	5	10	25	58	3	0	2	4	1	0	0	0	0	3	1	0	4	0	0
36	52	2	15	32	47	2	0	2	4	0	0	0	0	0	0	0	0	0	0	0
106	81	15	19	49	65	10	0	5	6	8	1	0	0	0	0	1	0	1	0	2
39	46	4	18	33	64	2	1	4	3	1	0	0	0	0	1	1	0	1	0	0
13	42	0	0	2	0	0	1	1	8	0	0	0	0	0	0	0	0	0	0	0
14	14	0	1	3	1	0	0	2	7	5	0	0	0	1	4	0	0	0	0	0
75	34	7	3	0	0	0	0	0	4	4	0	0	0	0	1	1	0	1	0	1
84	27	8	15	1	0	1	0	1	1	2	0	0	0	0	0	2	0	0	0	0
32	35	9	17	3	45	0	0	2	5	1	0	0	0	0	0	2	0	1	0	0
35	36	10	18	41	3	1	0	2	5	0	0	0	0	0	3	2	0	0	0	0
41	41	12	42	22	2	0	0	1	4	0	0	0	0	0	1	4	0	0	0	0
65	24	7	31	1	12	0	0	1	4	0	0	0	0	0	0	1	0	0	0	0
17	46	8	21	3	23	0	0	1	5	0	0	0	0	0	0	0	0	0	0	0
45	55	5	18	14	12	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0

Fig. 8. Textual features results.

tasks because it effectively defines the boundary between two classes [38]. Therefore, it shows good accuracy in classifying phishing and non phishing emails at the expense of some imbalance among other parameters.

Lastly, SVM exhibits the lowest overall accuracy of 64% and a more significant imbalance among other classes. Even though SVM achieved a decent recall for non-phishing emails (0.67), its ability to identify phishing emails was lower (precision: 0.57, recall: 0.60). As a result, SVM might struggle to distinguish phishing emails from legitimate ones. The main reason behind this performance degradation is SVM's dependency and sensitivity to data size and parameter tuning [39]. The limited dataset size in the study might have hindered SVM's ability to effectively learn a complex separation hyperplane between the classes, resulting in low performances. In summary, RF produced promising results across all ML models, as shown in Fig. 9.

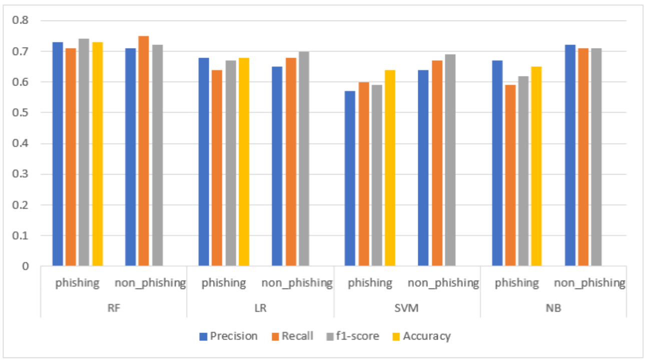


Fig. 9. Model performance visuals

F. *Evaluation of unseen data*

Aiming to ensure the validation of our model's performance in detecting any form of phishing emails, we conducted a further experiment using an unseen phishing dataset composed by [40]. This dataset is composed of various Arabic emails divided into 150 phishing and 150, non-phishing emails. It was inputted to the models as unseen data, while the compiled models were trained on the main dataset. The unseen test dataset was subjected to identical pre-processing steps, followed by the extraction of the same textual features as the main dataset. The results of the models gave a range of accuracies from 48% to 68%, The lowest accuracy pertained to SVM, and the RF produced the highest accuracy reaching 68%. Fig. 10 shows the confusion matrix of the RF model's performance on the unseen test dataset, which shows that the model successfully classified 130 non-phishing emails from 150; however, it lagged behind in terms of the phishing emails, correctly classifying only 74 out of 150 emails. This might be justified as the similarity in the textual features between both phishing and non-phishing emails causes very nuanced differences for the model to discern. The results emphasize the potential of the models to identify non-phishing emails from phishing emails, by capturing the distinctive textual features that represent linguistic cues correlated with phishing emails.

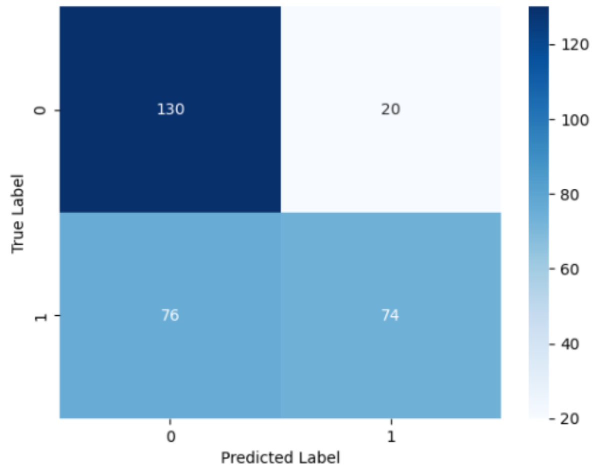


Fig. 10. Confusion matrix of the RF model's performance

G. *Leveraging Tasaheel for Textual Analysis*

Since the phishing behavior is an intentional attempt to deceive the user [41], deceiving another person requires the manipulation of language and the cautious construction of a story that appears to be true. Several studies, including those conducted by [42] and [43], have compared the word usage of genuine and fake articles to obtain a deeper understanding of the writing characteristics of deceptive text. For that, Tasaheel-v2 is used to conduct a thorough textual analysis. Therefore, the analysis can help develop strategies to detect and counteract phishing attempts. We extracted ten POS, six emotion, nine linguistic, and seven domain-specific word tags. The lexical density of each tag is calculated to provide a thorough analysis of the full dataset.

Lexical Density (L) =
(Total feature occurrences in a class / Total words in the class) × 100

Their lexical densities are calculated and displayed in Table 4.

H. *Discussion*

According to the results, we find that RF achieved the highest accuracy reaching 73% compared to the other ML models. On the other hand, distinctive patterns in the textual features revealed vital indications.

TABLE IV. POS, linguistic, emotion, domain-specific lexical densities

POS	**Non-Phishing**	**Phishing**
Nouns	23.86	30.6
Proper Nouns	10.87	18.75
Verbs	3.91	4.46
Conjunction	2.78	3.41
Pronoun (All)	3.10	4.06
Pronouns(Singular)	33.16	28.27
Pronouns (Plural)	5.44	6.07
Adverb	0.23	0.26
Prepositions	8.14	7.51
Adjectives	4.85	4.98
Linguistics		
Assurance	1.14	0.79
Negators	1.55	0.69
Opposite	0.19	0.03
Justification	0.18	0.22
Exception	0.23	0.19
Illustration	0.12	0.05
Hedge	0.52	0.20
Order	0.01	0.01
Intensifiers	0.59	0.62
Emotions		
Joy	0.93	0.78
Sad	0.17	0.25
Anger	0.24	0.02
Fear	0.14	0.03
Surprise	0.08	0.11
Disgust	0.04	0.02
Domain-Specific		
Religious	0.19	0.32
Organization	0.27	0.44
Day	0.11	0.02
Month	0.09	0.02
Nationality	0.0	0.10
Title	0.0	0.20
Quantifier	5.78	3.17

First, we find that in terms of POS, the largest proportion of POS tags comes from nouns. Interestingly, the findings also show that not only there is a huge increase in nouns, but there is an increase in 'conjunctions' as well to support these nouns. A study by Kapusta et al. [44] reported similar observations wherein 'nouns' comprised a higher percentage of the themes they discovered while conducting their experimental study. With further analysis of the noun types, we found that proper nouns in phishing emails were highly used compared to non-phishing emails. What these findings suggest could in part be explained by the fact that phishing email writers tend to integrate their ‘deceptive’ statements with made-up proper nouns in the form of people and governmental key figures, to convince the receiver of the legitimacy of the sender, by stating ‘countries’, ‘organizations’ and ‘key figures’ to support their narratives. In fact, verbs, adverbs, and adjectives, although slightly more highly used in phishing emails, are also indicators that their writers tend to support their deceptive writings by integrating explanatory and descriptive made-up events. Many of which contained intensifiers to attract the reader to a specific point. The terminologies used to explain events include temporal elements like days, months, and quantifiers for more description.

Another interesting insight is the use of pronouns. We found that singular pronouns are less frequently used in phishing emails representing an overall value of 28.27% than in non-phishing representing an overall value of 33.16%. This notion was supported in an extant study conducted by [45], highlighting that liars tend to dissociate themselves from their made-up statements in by feeling the guilt of lying or due to their lack of support for their lie by authentic facts.

As most phishing emails contain descriptive events, we find that they also contain a large number of emotions. When it concerns emotions, we find that sad, disgust, and surprise are the most dominant emotional terms used in phishing emails. On the other hand, non-phishing emails frequently included emotions of joy, anger, and fear. There was no indication of the discrepancies between emotions other than the diversity of topics discussed in the emails.

Another interesting finding is the difference between the uses of justification terms. According to the central route of persuasion, it emerges from a thorough assessment of the presenting arguments and message qualities, which necessitates a significant amount of effort and cognition. On the contrary, the peripheral path of persuasion includes linking concepts or establishing assumptions that are unconnected to the logic and quality of the presented information. This method could be referred to as heuristic because it does not guarantee to be optimal or even adequate in achieving its goal of locating accurate or trustworthy information. The peripheral route requires minimal energy and mental capacity [46]. This result is consistent with what we discovered when analyzing the linguistic word choices in non-phishing and phishing emails. On the contrary, we found that fake news relies more on linguistic terms such as places (geographical location) and times (events in time) to support the made-up events in their effort to persuade through a peripheral route.

V. Conclusion

In this study, we have introduced Tasaheel-v2, which expands the options for the users as it includes sentiment analysis and translation services from English to Arabic. We integrated these features with the ones of Tasaheel-v1, which were stemming, tokenization, normalization, POS, emotion, polarity, linguistic, and domain-specific tagging. Both versions integrated to form a Tasaheel to provide great support for researchers interested in the Arabic research domain. The tool’s distinctive feature resides in the fact that it offers a range of cutting-edge, open-source NLP packages and tools, all conveniently consolidated into one tool for convenient use. Moreover, we demonstrate the effectiveness and feasibility of the tool’s usage in assisting to compile machine learning models that detect Arabic phishing emails, which are a scarcely investigated topic due to the limitations of such supportive tools. In this study, the developed models achieved 73% accuracy, namely the RF model. Further, we test the models’ potential for employment in real-world cases, by testing their performance on unseen instances of Arabic phishing emails. The RF model achieved the highest accuracy of 68%, indicating a promising direction for phishing email detection. Additionally, we offer an extensive textual analysis of the dataset containing phishing emails in order to identify deceptive linguistic patterns that might be useful in the detection of such emails. Our goal for future initiatives is to provide advanced tools that offer convenient solutions for Arabic computational linguistics.

References

[1] A. Pasha, M. Al-Badrashiny, M. T. Diab, A. El Kholy, R. Eskander, N. Habash, M. Pooleery, O. Rambow, and R. Roth "MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of arabic.," *in Proceedings of the Ninth International Conference on Language Resources and Evaluation LREC*, vol. 14, pp. 1094–1101, 2014.

[2] K. Darwish and H. Mubarak, "Farasa: A new fast and accurate arabic word segmenter," *in Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 1070–1074, 2016.

[3] A. Elnily and A. Abdelghany, "Automatic POS tagging of Arabic words using the YAMCHA machine learning tool," *in 2022 20th International Conference on Language Engineering (ESOLEC)*, vol. 20, pp. 72–77, IEEE, 2022.

[4] H. Himdi, G. Weir, F. Assiri, and H. Al-Barhamtoshy, "Arabic fake news detection based on textual analysis," *Arabian Journal for Science and Engineering*, vol. 47, no. 8, pp. 10453– 10469, 2022.

[5] A. Zamir, H. U. Khan, T. Iqbal, N. Yousaf, F. Aslam, A. Anjum, and M. Hamdani, "Phishing web site detection using diverse machine learning algorithms," *The Electronic Library*, vol. 38, no. 1, pp. 65–80, 2020.

[6] J. Wu, S. Yang, R. Zhan, Y. Yuan, D. F. Wong, and L. S. Chao, "A survey on LLM-gernerated text detection: Necessity, methods, and future directions," *arXiv preprint arXiv:2310.14724*, 2023.

[7] H. Himdi and F. Assiri, "Tasaheel: An Arabic automative textual analysis tool-all in one," *IEEE Access*, 2023.

[8] R. Eskander, "SentiArabic: A sentiment analyzer for standard Arabic," *in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[9] M. Abdul-Mageed and M. T. Diab, "Sana: A large scale multi-genre, multi-dialect lexicon for Arabic *subjectivity and sentiment analysis.," in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2014)*, pp. 1162–1169, 2014.

[10] K. Nahar, A. Al Eroud, M. Barahoush, and A. Al-Akhras, "SAP: standard Arabic profiling toolset for textual analysis," *International Journal of Machine Learning and Computing*, vol. 9, no. 2, pp. 222–229, 2019.

[11] O. Obeid, N. Zalmout, S. Khalifa, D. Taji, M. Oudah, B. Alhafni, G. Inoue, F. Eryani, A. Erdmann, and N. Habash, "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, 2020.

[12] K. Bouzoubaa, Y. Jaafar, D. Namly, R. Tachicart, R. Tajmout, H. Khamar, H. Jaafar, L. Aouragh, and A. Yousfi, "A description and demonstration of safar framework," *in Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pp. 127–134, 2021.

[13] M. Mustafa, A. S. Eldeen, S. Bani-Ahmad, A. O. Elfaki, et al., "A comparative survey on Arabic stemming: approaches and challenges," *Intelligent Information Management*, vol. 9, no. 02, p. 39, 2017.

[14] D. H. Abd, W. Khan, K. A. Thamer, and A. J. Hussain, "Arabic light stemmer based on ISRI stemmer," *in International Conference on Intelligent Computing*, pp. 32–45, Springer, 2021.

[15] I. Pak and P. L. Teh, "Text segmentation techniques: A critical review," *Innovative Computing, Optimization and Its Applications: Modelling and Simulations*, pp. 167–181, 2018.

[16] K. Darwish and H. Mubarak, "Farasa: A new fast and accurate Arabic word segmenter," *in Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), (Portoro˘z, Slovenia)*, pp. 1070–1074, European Language Resources Association (ELRA), May 2016.

[17] S. Patro and K. K. Sahu, "Normalization: A preprocessing stage," *arXiv preprint arXiv:1503.06462*, 2015.

[18] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit*," in Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, (Baltimore, Maryland)*, pp. 55–60, Association for Computational Linguistics, June 2014.

[19] S. Naseer, M. M. Ghafoor, S. bin Khalid Alvi, A. Kiran, S. U. Rahmand, G. Murtazae, and G. Murtaza, "Named entity recognition (NER) in NLP techniques, tools accuracy and performance.," *Pakistan Journal of Multidisciplinary Research*, vol. 2, no. 2, pp. 293–308, 2021.

[20] D. Kumawat and V. Jain, "POS tagging approaches: A comparison," *International Journal of Computer Applications*, vol. 118, no. 6, 2015.

[21] Z. K. Igaab and I. Kareem, "Affixation in english and arabic: A contrastive study," *English Language and Literature Studies*, vol. 8, no. 1, pp. 92–103, 2018.

[22] S. AL-Sarayreh, A. Mohamed, and K. Shaalan, "Challenges and solutions for Arabic natural language processing in social media," *in International conference on Variability of the Sun and sun-like stars: from asteroseismology to space weather*, pp. 293–302, Springer, 2023.

[23] S. Alharthi, R. Siddiq, and H. Alghamdi, "Detecting Arabic fake reviews in e-commerce platforms using machine and deep learning approaches," *Journal of King Abdulaziz University Computing and Information Technology Sciences*, vol. 11, p. 27–34, Sep. 2022.

[24] A. A. Freihat, G. Bella, M. Abbas, H. Mubarak, and F. Giunchiglia, "ALP: An Arabic linguistic pipeline," *in Analysis and Application of Natural Language and Speech Processing*, pp. 67–99, Springer, 2022.

[25] R. Alluhaibi, T. Alfraidi, M. A. Abdeen, and A. Yatimi, "A comparative study of Arabic part of speech taggers using literary text samples from saudi novels," *Information*, vol. 12, no. 12, p. 523, 2021.

[26] M. Junczys-Dowmunt, R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. F. Aji, N. Bogoychev, et al., "Marian: Fast neural machine translation in C++," *arXiv preprint arXiv:1804.00344*, 2018.

[27] H. Jivane, "Evaluating machine translation models with T5 and Marian," *Medium*, Aug. 30, 2023. [Online]. Available: https://medium.com/@mleinbayarea/evaluating-machine-translation-models-with-t5-and-marian-52ad3651f1a8. [Accessed: Aug. 31, 2024].

[28] M. Junczys-Dowmunt, T. Dwojak, and H. Hoang, "Is neural machine translation ready for deployment? a case study on 30 translation directions," *in Proceedings of the 13th International Conference on Spoken Language Translation*, Dec. 8-9 2016.

[29] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[30] O. Obeid, N. Zalmout, S. Khalifa, D. Taji, M. Oudah, B. Alhafni, G. Inoue, F. Eryani, A. Erdmann, and N. Habash, "CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, Marseille, France, 2020, pp. 7022–7032. European Language Resources Association.

[31] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and others, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[33] W. Antoun, F. Baly, and H. Hajj, "AraBERT: Transformer-based model for Arabic language understanding," *arXiv preprint arXiv:2003.00104*, 2020.

[34] D. Radev, "CLAIR collection of fraud email," *ACL Data and Code Repository*, ADCR2008T001, 2008. [Online]. Available: http://aclweb.org/aclwiki. [Accessed: Aug. 31, 2024].

[35] M. Gashler, C. Giraud-Carrier, and T. Martinez, "Decision tree ensemble: Small heterogeneous is better than large homogeneous*," in 2008 Seventh international conference on machine learning and applications*, pp. 900–905, IEEE, 2008.

[36] M. Zakariah et al., "Classification of large datasets using random forest algorithm in various applications: Survey," *International Journal of Engineering and Innovative Technology (IJJEIT)*, vol. 4, no. 3, 2014.

[37] M. Norallahi and H. Seyed Kaboli, “Urban flood hazard mapping using machine learning models: GARP, RF, MaxEnt and NB,” *Natural Hazards*, vol. 106, pp. 119–137, 2021.

[38] S. H. Ebenuwa, M. S. Sharif, M. Alazab, and A. Al-Nemrat, “Variance ranking attributes selection techniques for binary classification problem in imbalance data,” *IEEE Access*, vol. 7, pp. 24649–24666, 2019.

[39] A. C. Lorena and A. C. De Carvalho, “Evolutionary tuning of SVM parameter values in multiclass problems,” *Neurocomputing*, vol. 71, no. 16-18, pp. 3326–3334, 2008.

[40] S. Salloum, T. Gaber, S. Vadera, and K. Shaalan, “A New English/Arabic Parallel Corpus for Phishing Emails,” *ACM Transactions on Asian and Low-Resource Language Information Processing,* vol. 22, no. 7, Article 201, pp. 1–17, Jul. 2023.

[41] J. S. Tucker and H. S. Friedman, “Chapter 17 - emotion, personality, and health,” *in Handbook of Emotion, Adult Development, and Aging,* pp. 307–326, Academic Press, 1996.

[42] J. Kapusta and J. Obonya, “Improvement of misleading and fake news classification for flective languages by morphological group analysis,” *Informatics,* vol. 7, p. 4, MDPI, 2020.

[43] B. Horne and S. Adali, “This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news,” *in Proceedings of the international AAAI conference on web and social media,* vol. 11, pp. 759–766, 2017.

[44] J. Kapusta, P. Hajek, M. Munk, and L. Benko, “Comparison of fake and real news based on morphological analysis,” *Procedia Computer Science*, vol. 171, pp. 2285–2293, 2020.

[45] M. L. Knapp, R. P. Hart, and H. S. Dennis, “An exploration of deception as a communication construct,” *Human communication research,* vol. 1, no. 1, pp. 15–29, 1974.

[46] P. Bertelson, P. Eelen, and G. d’Ydewalle, *International Perspectives On Psychological Science, II: The State of the Art*. Taylor & Francis, 2013.

عنوان الورقة العلمية بالعربي

حنين طارق حمدي[1] ، فاطمه يوسف عسيري [2]

[1]قسم علوم الحاسب والذكاء الاصطناعي، كلية علوم وهندسة الحاسب، جامعة جدة، المملكة العربية السعودية

[2] قسم هندسة البرمجيات، كلية علوم وهندسة الحاسب، جامعة جدة، جدة، المملكة العربية السعودية

المستخلص: نقدم Tasaheel–v2، أداة آلية تم تطويرها خصيصًا لمعالجة اللغة الطبيعية العربية (NLP) ومهام التحليل النصي. يُعد هذا العمل امتدادًا للإصدار الأول، Tasaheel–v1، الذي شمل مهام NLP التقليدية مثل التجذير، التقسيم، التوحيد، التعرف على الكيانات المسماة، وتحديد أجزاء الكلام. بالإضافة إلى ذلك، تضمن طرق تحليل متقدمة مثل تحديد العواطف المحددة، والتوجهات، والتحليل اللغوي، ووضع العلامات على الكلمات حسب المجال. في هذا الإصدار المبتكر الجديد، Tasaheel–v2، نقدم أدوات إضافية تهدف إلى تقديم المساعدة للمجتمع البحثي العربي. قمنا بدمج مُصنف آخر لأجزاء الكلام خاص باللغة العربية، ومحلل للمشاعر، ووظائف الترجمة من الإنجليزية إلى العربية. نستفيد من الأدوات المقدمة في Tasaheel لتطوير نموذج تعلم آلي مصمم لتحديد رسائل البريد الإلكتروني الاحتيالية باللغة العربية وتقديم تحليل نصي شامل لالتقاط الإشارات المخادعة المستخدمة في اكتشاف الأنماط اللغوية للاحتيال. تسهم هذه الأداة في المجال البحثي العربي من خلال تقديم وظائف معالجة اللغة الطبيعية وميزات التحليل النصي في أداة واحدة.

الكلمات المفتاحية. : معالجة اللغة الطبيعية (NLP)، التحليل النصي، تطوير الأدوات، تحليل المشاعر، الترجمة، التعلم الآلي.