IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 29, No. 1, February, 2024, pp. 9-17
Journal homepage: http://iieta.org/journals/isi

An Arabic OCR Approach Using Levenshtein Distance and CNNs

Walid Fakhet[1], Salim El Khediri[2*], Salah Zidi[1]

[1] Laboratory Hatem Bettaher (IRESCOMATH), National School of Engineers of Gabes, Gabes 6029, Tunisia
[2] Department of Information Technology, College of Computer, Qassim University, Buraydah 51452, Saudi Arabia

Corresponding Author Email: s.elkhediri@qu.edu.sa

https://doi.org/10.18280/isi.290102

Received: 24 October 2023
Revised: 24 November 2023
Accepted: 8 December 2023
Available online: 27 February 2024

Keywords:
AHCR, Convolutional Neural Network, Levenshtein distance, Jaccard distance, OCR

ABSTRACT

Arabic Handwritten Character Recognition (AHCR) systems encounter various challenges arising from the unique characteristics of the Arabic language and the limited availability of public databases. Consequently, numerous research endeavors have aimed to enhance the recognition accuracy of AHCR. In this study, we propose a solution inspired by the intricate functions of the human visual cortex and hippocampus. Our proposed system employs a segmentation method to break down Arabic characters, and a Convolutional Neural Network (CNN) is then utilized for character recognition. To recognize entire words, we employ the Levenshtein distance with a personalized database containing an extensive collection of Arabic words. Experimental results demonstrate that our system yields a word error rate ranging from 1% to 25%, contingent upon the number of accurately recognized characters.

1. INTRODUCTION

In the contemporary digital era characterized by abundant and readily accessible information, the significance of Optical Character Recognition (OCR) technology cannot be overstated. OCR has transformed the conversion of text from printed or handwritten images and scanned documents into machine-readable text, facilitating seamless digitization, enhanced search capabilities, and analytical processes. While OCR has found extensive application in languages based on the Latin script [1-3], its implementation in the context of the rich and intricate Arabic script poses distinctive challenges and opportunities.

The Arabic language stands as a rich and ancient Semitic language that has profoundly shaped the history, culture, and progress of the Middle East and North Africa [4, 5]. Boasting over 300 million native speakers [6], Arabic stands as one of the most globally spoken languages, carrying substantial significance in diverse domains such as literature, religion, science, and diplomacy [7]. Nevertheless, the intricacy of Arabic poses a formidable challenge for accurate character recognition, as illustrated in Figure 1.

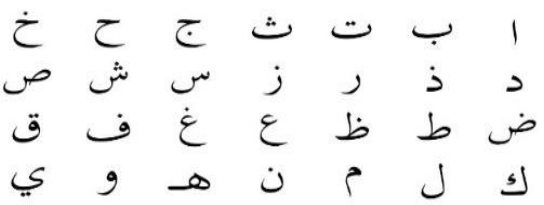

Figure 1. Arabic alphabet

Arabic OCR emerges as a solution to unlock the vast potential hidden within manuscripts, historical documents, and printed materials, offering a bridge between the traditional and the digital.

The intricate nature of the Arabic script introduces challenges that set Arabic OCR apart from its Latin counterparts [8-10]. Key challenges include cursive complexity, diacritics, ligatures, contextual variations, and style variability.

In the human brain, information about words is stored in a distributed manner across various regions associated with language processing [11-13] as shown in Figure 2. For example, the word "مدرسة" might be presented in the brain as follows:

Visual Cortex: When you see the written word "مدرسة", the visual information is processed by the visual cortex at the back of our brain. The shape and appearance of the letters "م","د","ر","س" and "ة" are recognized and processed here.

Fusiform Gyrus: The fusiform gyrus, located in the temporal lobe, is responsible for recognizing and processing visual objects, including familiar objects like cats. So, the visual representation of a cat as a whole is stored in this region.

Wernicke's Area: As mentioned earlier, Wernicke's area, located in the left hemisphere, is involved in understanding and processing language. The word "مدرسة " is associated with its meaning and semantic representation here. When we read or hear the word "مدرسة ," Wernicke's area helps us comprehend its meaning and relate it to the concept of a furry, four-legged animal.

Broca's Area: If we are thinking about saying the word " مدرسة " aloud, Broca's area comes into play. It is involved in the production of speech and helps us retrieve the appropriate sounds and motor patterns needed to say the word "مدرسة " out

loud.

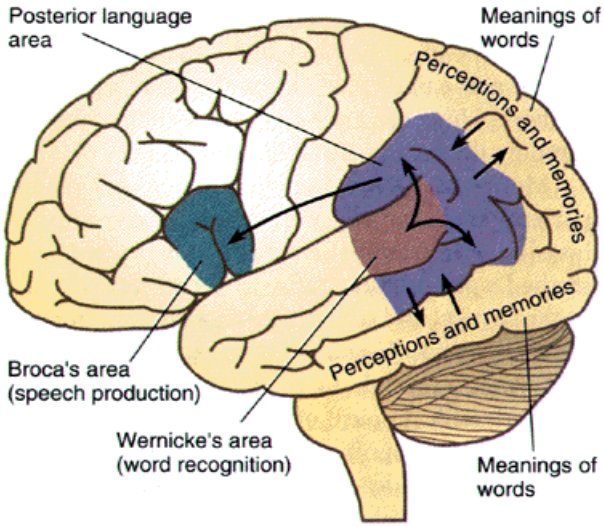

Figure 2. The human brain

Hippocampus: The hippocampus plays a vital role in forming and storing memories, including memories of words. It helps with the consolidation of new information into long-term memory. So, when we learn the word "مدرسة " for the first time, the hippocampus helps encode and store that memory.

Association Areas: Various association areas in the brain help link the word "مدرسة " with related information and experiences.

The word "مدرسة " is represented by a distributed network of interconnected brain regions that work together to process its visual form, meaning, and associations. As you encounter and use the word more frequently, the connections between these brain regions strengthen, making it easier to recall and use the word "مدرسة " in various contexts.

The method presented in this article draws inspiration from the functionality of the visual cortex and the Hippocampus areas. In an effort to emulate these concepts, we sought to recognize Arabic words by employing a dictionary containing an extensive array of words, mimicking the role of the Hippocampus. To replicate the visual cortex's functionality, we implemented a segmentation method to break down a word into characters and utilized a Convolutional Neural Network (CNN) model for character recognition.The rest of this paper is structured as follows: Section 2 provides an introduction to the research background. The proposed methodology is outlined in Section 3. Experimental results are thoroughly examined in Section 4. The conclusion, along with future work, is presented in Section 5.

2. RELATED WORKS

This section is divided into two subsections: In Sec 1, we discuss some solutions for OCR segmentation. In Sec 2 we present some applied CNN models for AHCR.

2.1 Segmentation for AHCR

Segmenting characters plays a crucial role in the initial processing stages of character recognition within numerous OCR systems. As a result, various segmentation techniques are employed to divide Arabic characters effectively.

Najoua and Noureddine [14] introduced a technique based on histogram analysis. The proposed method involves several sequential steps: recognizing text lines, dividing text lines into sections of Arabic words (PAW), isolating each PAW into connected components, determining approximate boundaries of various characters within the PAW, computing the maximum count of black segments in a line of pixels, extracting primitives, and employing an error checker to identify segmentation errors. The effectiveness of this method was assessed using the Arabic fonts "Nakesh," "Bagdadi," and "Mehdi," yielding accuracy results ranging from 99% to 100%.

Another use of the histogram technique by Amin and Masini [15] where the authors proposed a segmentation method using horizontal and vertical projections. The proposed method achieved a recognition rate of 85% for the characters and 95% for the words.

Several approaches have been developed employing contour tracing to address character segmentation challenges. In Sari et al. [16], the authors utilized contour representation to identify segmentation points by applying rules to the local minima of the lower contour for each subword. The achieved recognition rate on a dataset of 100 words was 86%.

Margner [17] developed a segmentation approach relying on the contour of the primary body of words. Initially, the top contour's start and end points are identified. Subsequently, the upper contour is partitioned into sections with consistent curvatures. Finally, a horizontal line detector is employed to dismiss these lines as irrelevant for the recognition process. This segmentation method demonstrated a recognition rate of 99% for characters without dots and 96.9% for characters with dots.

In their work, Hamid and Haraty [18] applied a neuro-heuristic approach for text line segmentation in Arabic Handwritten script. The identification of potential segmentation points involved a search for topographical elements. The subdivision of these points relied on the average block width of the letters. Through this method, they attained an accuracy of 69.72%.

2.2 CNNs for AHCR classification

Lately, numerous studies have delved into the application of Convolutional Neural Networks (CNNs) in Arabic handwriting recognition. This section offers an overview of various CNN models employed in the context of recognizing Arabic handwritten characters.

In the work by El- Sawy et al. [19], a CNN model was introduced for the classification of Arabic handwritten characters. The employed CNN model comprises two convolution layers with a Rectified Linear Unit (RELU) activation function, a pooling layer with a 2x2 window size, and a fully connected layer housing 1024 neurons. The model was evaluated on their proprietary dataset (AHCD), yielding a commendable recognition rate of 94.9%.

Another use of the CNN model by Younis [20], used a CNN model with regularization parameters to prevent overfitting. The author used the AHCD and AIA9K datasets to test the proposed model, and the accuracy rate was 97.6% and 94.8% respectively. In this study, the author mentioned that the morphological similarity between the Arabic characters affected the recognition rate of the proposed model.

Elkhayati and Elkettani [21] employed a unique architecture incorporating a fully connected layer (FCL) divided into two distinct blocks: conscious blocks (CB) and unconscious blocks

(UB). These blocks, separated from each other, are fully connected to the output layer. The researchers evaluated their approach using four benchmark databases: IFHCDB, AHCD, AIA9K, and HACDB. The achieved recognition rates for these databases were 98.4%, 98.7%, 96.1%, and 95.4%, respectively.

Table 1. Comparison table of Arabic Segmentation techniques

Author	Character Recognition Method	Scope of Application	Database Size	Accuracy
Najoua and Noureddine [14]	Histogram techniques	Multi-font Arabic text printed	1000 words	99%
Amin and Masini [15]	Horizontal and vertical projections and shape primitives	Multi-font Arabic text Printed	100 multi-font words	95%
Sari et al. [16]	Contour presentation		Dataset of 100 words	86%
Margner [17]	Contour detection	Printed Arabic text	4110 Printed Arabic characters	99% without dots 96.9% with dots
Hamid and Haraty [18]	Feed-forward multilayer neural networks	Arabic Handwritten Characters	Arabic handwritten-10,000 exemplars	69.72%

Table 2. Comparison table of Arabic Recognition methods

Author	Model	Database	Accuracy
El-Sawy et al. [19]	CNN	AHCD	94.9%
Younis [20]	CNN	AHCD	97.6%
		AIA9K	94.8%
		IFHCDB	98.4%
Elkhayati and Elkettani [21]	CNN	AHCD	98.7%
		AIA9K	96.1%
		HACDB	95.4%
Fakhet et al. [22]	CNN	HACDB	98%
Ahamed et al. [23]	CNN	Personal database	99.76%

In Fakhet et al. [22], a recognition system was proposed based on the similarity of Arabic characters. To build the similarity vector, a CNN model was used with the help of a confusion matrix. The model was tested with the HACDB dataset and achieved a recognition rate of 98%.

Ahamed et al. [23], the authors proposed a CNN model to recognize handwritten Arabic numbers based on the work of Ashiquzzaman and Tushar [24]. The authors used a personal handwritten Arabic numerals dataset with 72,000 images. The recognition rate achieved by the proposed model was 99.76%.

Most methods in the literature segment all words into characters then use a recognition system (CNN, SVM,) to recognize the words. These methods give more or less acceptable results despite the large number of segmentation and recognition of each character. in this study we propose a method which reduces the number of segmentation and recognition of characters.

3. PROPOSED METHOD

In this Section, the new Arabic OCR Approach is presented. we will work on the "segmentation" phases and the "classification" phase. Most of the errors that occur in an OCR exist at the level of these two phases, so we propose to reduce the number of times of segmentation and classification.

The solution we propose consists of segmenting the characters of a word and then recognizing some characters, then using a knowledge base that contains a set of words in text format, we calculate a distance between the characters recognized by the model and the words that exist in the knowledge base. To calculate the distance between words, two distances are considered: The Levenshtein distance and the Jaccard distance. In our study, we tested the two methods to get out with the best distance to use. In Table 1 and Table 2 a summary of all related works surveyed in this section.

3.1 Levenshtein distance

The Levenshtein distance [25] is a mathematical measure representing the dissimilarity between two character strings. It corresponds to the minimum number of operations required-whether deletion, insertion, or replacement-to transform one string into another. The Levenshtein distance can be viewed as a broader concept encompassing the Hamming distance (Eq. (1)).

$$lev_{a,b} = \begin{cases} \max(i,j) & if\ \min(i,j) = 0, \\ min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{ai \neq aj} \end{cases} & ,otherwise \end{cases} \quad (1)$$

The Levenshtein distance is the number of deletions, insertions, or replacements required to transform a string 'A' to the string 'B'. As shown in Figure 3, to calculate the distance between the word "ا-جم---ت" (the word to be recognized by our system) and the word "الجمعيات" (the word existing in the knowledge base) a large number of insertions, deletions and replacements are effectuated. The last number in the matrix presents the distance between the two words. The word "الجمعيات" is closer to the word that will be recognized than the word "جماعات".

3.2 Jaccard distance

Typically, the Jaccard similarity coefficient (or index) [26] is employed to assess the similarity between two sets. For sets A and B, the Jaccard index is defined as the ratio of the size of their intersection to the size of their union, as expressed in Eq. (2):

$$J(A,B) = \frac{A \cap B}{A \cup B} \quad (2)$$

These equations were used to calculate the similarity between the segmented word and all the words that exist in the knowledge base. If the similarity value is equal to 1 (0 for Eq. (1)) this means that the word searched exists in the knowledge base. Else, if the similarity value is less than 1 (less than the number of recognized characters for Eq. (1)) this means that the searched word doesn't exist in the base, so we need to recognize some other characters. To reduce the search space of words, we divided the knowledge base into multiple file texts, and each file text contains a set of words belonging to previously predefined context. The algorithmic complexity of Jaccard distance in the worst case is O(N2), where N is the number of words in a file text. For Levenshtein distance the algorithmic complexity between two words is O(|A|x|B|).

3.3 Knowledge base

The knowledge base contains a large number of words, it consists of a vocabulary base for our model. We built the knowledge base from Arabic texts collected from websites, press articles, and other existing vocabulary databases on the Internet. We have divided the Knowledge base into contexts as shown in Figure 3, each context contains a set of words that belong to this context (Table 3). The contexts used in our study are sports, religion, economics, and politics.

The purpose of dividing the knowledge base into contexts is to decrease the search space of a word in a context.

To build the knowledge base, we used JAVA code that allows us to read RSS feeds from a website. Articles read from RSS feeds are saved in text files after removing images and non-Arabic characters. We used five text files depending on the number of contexts we have chosen (Figure 4). To simplify working with files, each word is written in a separate line, so we kept only one occurrence for each word to reduce file size.

Table 3. Sample of words from the database

Sports	Religion	Economics	Politics
الرياضية	الله	المنشآت	الوطن
المنتخبات	رسول	لمشروع	المؤسسات
البطولة	الجلالة	المجالات	الاجراءات
الدوري	صَلّى	المؤسسين	للمنظمات

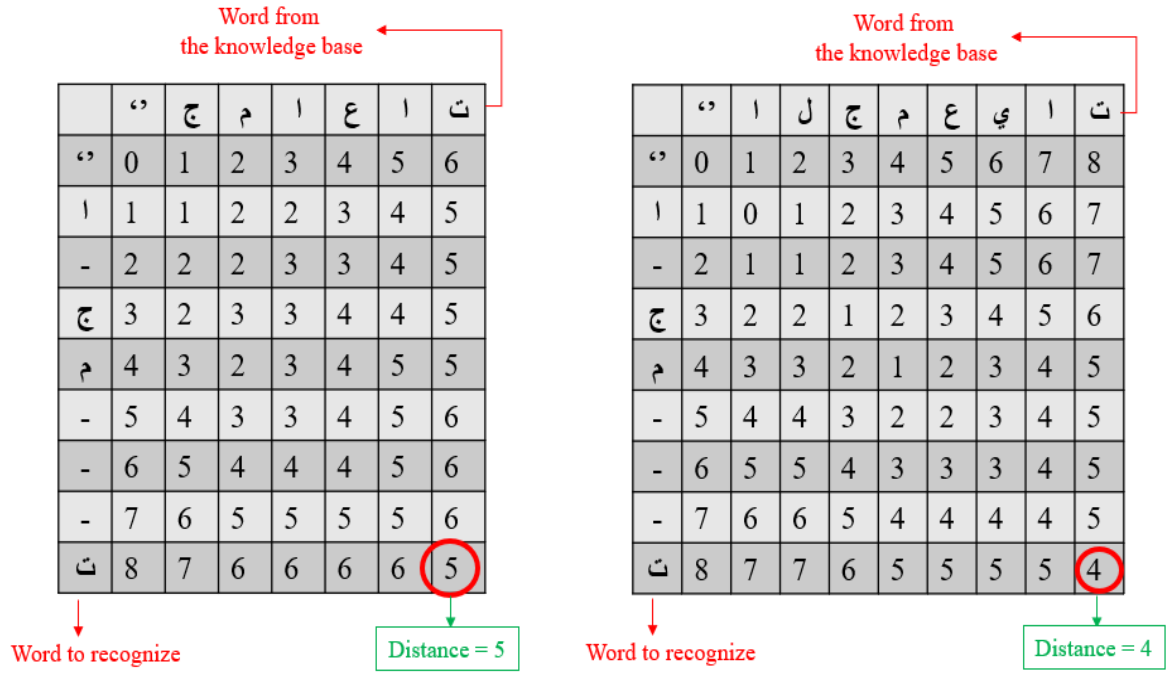

Figure 3. Example of Levenshtein distance calculation

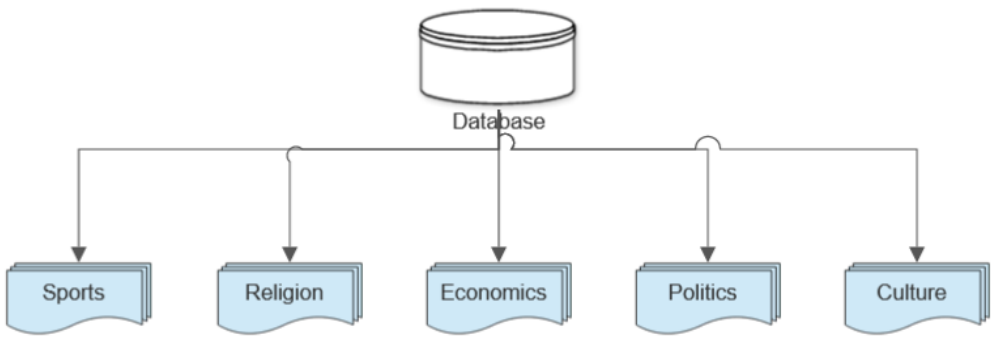

Figure 4. Structure of the new database

3.4 Presentation of the approach

3.4.1 Segmentation

Sometimes we can read a word despite the characters not being in the correct positions, so we ask the question of how our brain can read these types of words and not pay attention that the characters are not in their true positions. The explanation that we came up with, is that our brain does not recognize all the characters of a word but only recognizes a few characters as Figure 5 illustrates, then using a dictionary (a knowledge base) that presents the vocabulary learned by a person during his life, it compares the characters recognized with the words that exist in the vocabulary base to find the appropriate word (Figure 6).

This research introduces a novel algorithm designed for the recognition of Arabic texts. The algorithm's steps are delineated in the following pseudo-code:

(1) Input word image

(2) Segment the word into characters

(3) Recognize some characters

(4) Find the closet word to the recognized characters by applying a distance (Levenshtein, Jaccard)

(5) Return the result

The proposed algorithm starts by segmenting the image containing a word into characters, we used the method in the work [14] to segment the words. After segmentation, we randomly chose a number of characters (less than the length of the word) to recognize them, we used a CNN model to recognize the characters, so we used the AHDC database to train this model. Then, a distance is used to calculate the similarity between the characters obtained in the segmentation phase and the words that exist in the vocabulary base. Finally, we display the word that has the greatest similarity measure, if the similarity measure is not so sufficient, we can go back to step 3 and recognize another character to improve the similarity measure. Figure 7 shows an example application of the algorithm to an Arabic word.

3.4.2 CNN architecture

The depicted CNN model in Figure 8 proposes three convolutional layers aimed at extracting pertinent features from the image. The initial convolutional layer employs a 3×3 kernel, yielding a feature map size of 32×32, succeeded by a max pooling layer resulting in a feature map of 16×16. This ensures effective low-level feature extraction. The second convolutional layer encompasses a feature map of 16×16 and a max pooling layer with a 2×2 size, producing a feature map of 8×8. Finally, the third convolutional layer incorporates feature maps of 8×8, followed by a max pooling layer yielding a feature map size of 4×4. To mitigate overfitting, a dropout of 50% was introduced.

Since the position of the characters is important in the Levenshtein distance, we keep the position of the characters by using the number of characters obtained during the segmentation phase.

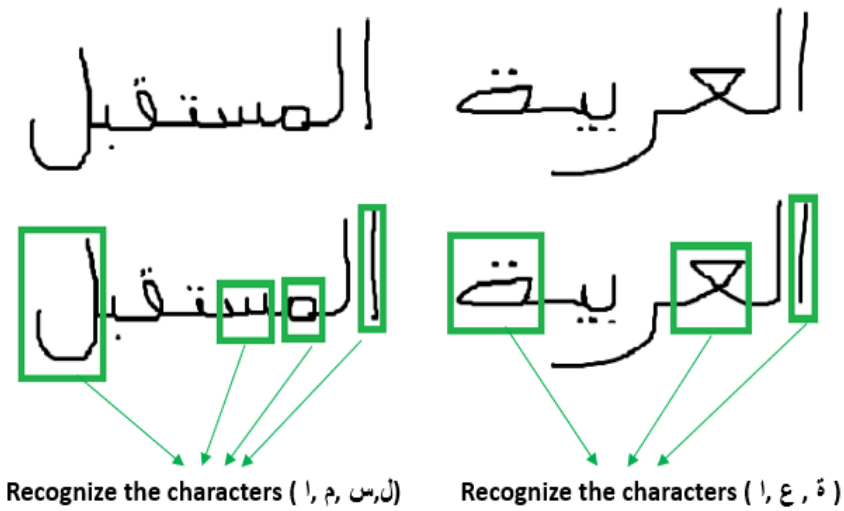

Figure 5. Our Perception of how the human brain recognizes words

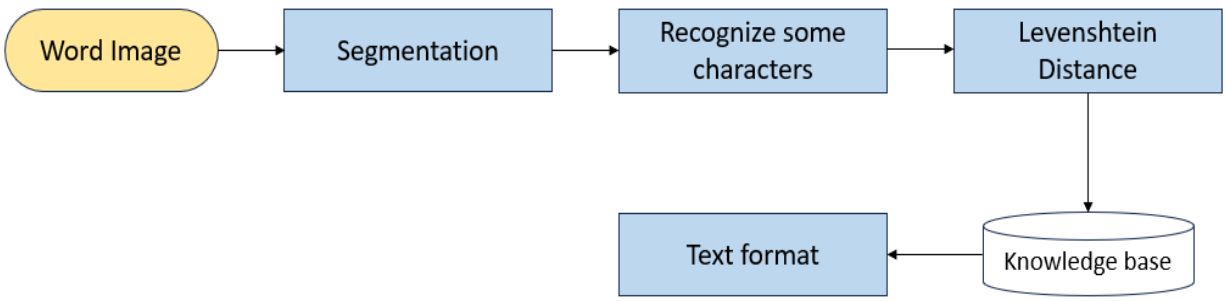

Figure 6. The proposed CNN architecture

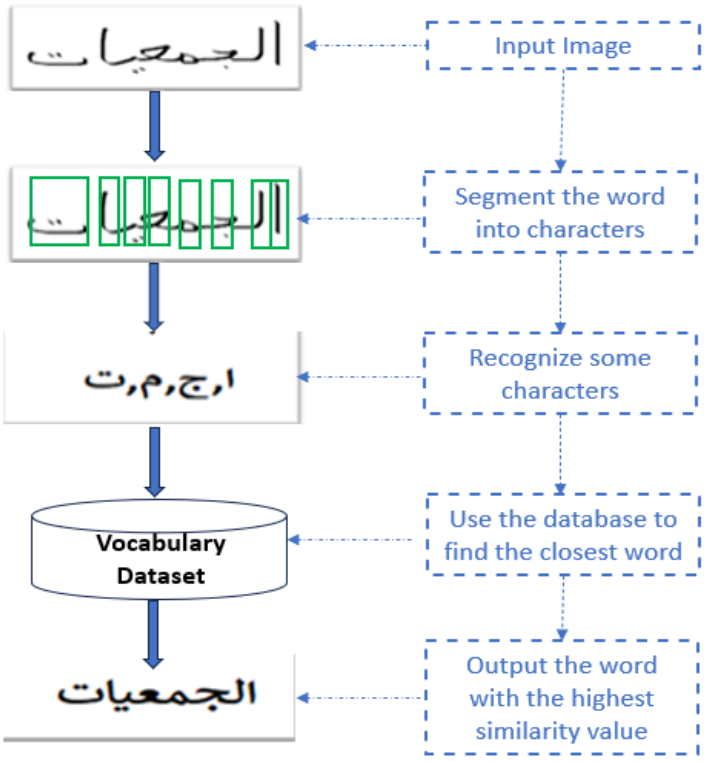

Figure 7. Sample of the proposed system

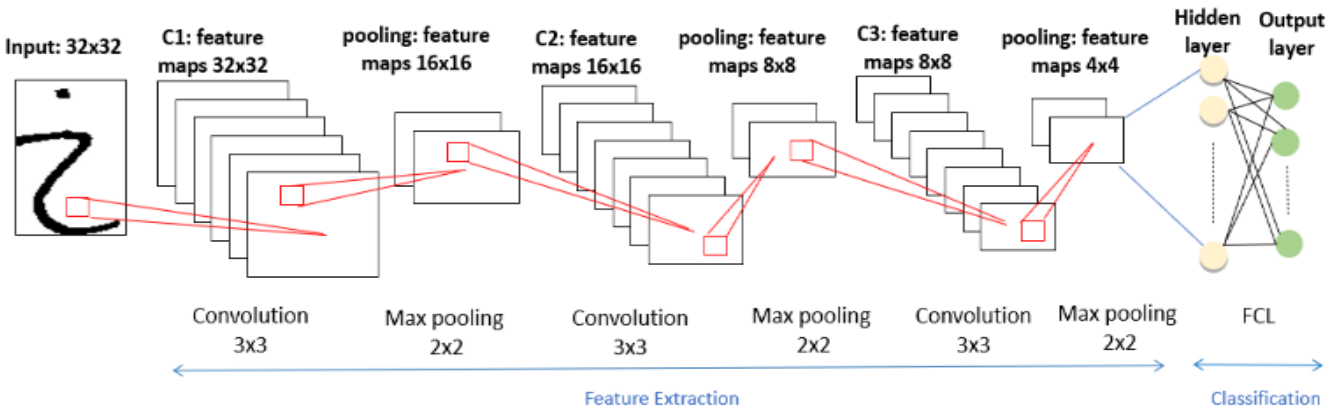

Figure 8. The proposed system

4. EXPERIMENT

4.1 Experimental settings

The system we developed uses the AHCD dataset to train the CNN model to recognize Arabic characters. The AHCD database is a database that contains 13622 images divided into a test part and a training part. Thus, our system uses the vocabulary base that we have created. The database is built from 5 text files that present the 5 contexts that we choose to work with.

The experiments are performed on an Intel® Core™ I5-10300H processor and 16GB RAM. The operating system is WINDOWS 10 64-bit.

4.2 Experimental results

In our experiments, we employed the Word Error Rate (WER) to assess the performance of our system, calculated using the Eq. (3):

$$WER = \frac{m}{n} \tag{3}$$

where, m is the total number of words recognized correctly by our system and n is the total number of words existing in a golden file to evaluate the system performance.

To evaluate our system, we used several parameters as presented in Table 4 we tested the impact of the number and the position of the recognized characters on the result. We see that the number of characters affects the result, so the position of the recognized characters can affect the result. The result demonstrates that the Levenshtein distance gives better results than the Jaccard distance.

Figure 9 presents the evolution of the word error rate compared to the number of recognized characters; we see that

the Levenshtein distance gives better results than the Jaccard distance when the number of recognized characters is bigger.

Figure 10 illustrates the error rate progression in the rule-based system, the hybrid system [27], and our proposed method utilizing Levenshtein distance. The comparison clearly demonstrates that the new method exhibits a lower error rate than the alternative systems. Despite using only "L/2" characters (L: length of word) in our model to minimize the recognized characters, it outperforms both the rule-based and hybrid models. In our proposed method, we achieved an error rate of 14.1% with 1000 words, and this error rate decreases to 4% when the number of words is increased to 1500. The table provides a clear overview, indicating that our system performs increasingly better as the number of words in the database grows, maintaining an error rate between 6% and 1.2%.

Also in this research, we have studied the impact of bad segmentation which is a very important step in our system. In Table 5 we have presented the two types of segmentation faults that can occur during the process of segmentation, under-segmentation, and over-segmentation.

An error segmentation can cause characters to be misrecognized. Segmentation faults can affect the performance of our system, but we can overcome this problem by increasing the number of characters to recognize. From the experiments we have carried out, we can say that recognizing half of the characters of a word can help have an excellent result even if there are segmentation problems.

Our model basically depends on the number of words present in the database. If the word to be recognized does not exist in the database, the model will not be able to correctly recognize the word but will return the closest match.

Table 4. Samples of words recognized by our system

N°	Word	Recognized Characters	Number of Characters	Word to Recognize	Distance	Output
1	الجمعيات	ت ,ا ,ع ,ج ,ل ,ا	6	ات-ع-الج	Livenshtein	الجمعيات
2	الجمعيات	ت ,ا ,ع ,ج ,ل ,ا	6	ات-ع-الج	Jaccard	الجمعيات
3	الجمعيات	ت ,ا ,ع ,ج ,ا	5	ات-ع-ج-ا	Livenshtein	الجمعيات
4	الجمعيات	ت ,ا ,ع ,ج ,ا	5	ات-ع-ج-ا	Jaccard	الجمعيات
5	الجمعيات	ت ,ع ,ج ,ا	4	ت--ع-ج-ا	Livenshtein	الجمعيات
6	الجمعيات	ت ,ع ,ج ,ا	4	ت--ع-ج-ا	Jaccard	الجماعات
7	الرياضية	ض ,ر ,ا	3	ض--ر-ا--	Livenshtein	الرياضية
8	الرياضية	ض ,ر ,ا	3	ض--ر-ا--	Jaccard	الرياضية
9	الرياضية	ة ,ر ,ا	3	ة---ر-ا	Livenshtein	ارضية
10	الرياضية	ة ,ر ,ا	3	ة---ر-ا	Jaccard	ارضية

Table 5. Sample of words with segmentation error

N°	Word	Error Type	Character Not Segmented /Misrecognized	Output
1	تمهيدا	Under-segmentation	ي not segmented	تمهيدا
2	الدكتور	Misrecognition	د recognized as ذ تrecognized as ن	الدكتور
3	المواطنون	Under-segmentation	م not segmented	المواطنون
4	المتوقع	Over-segmentation	تsegmented as two charactersن	المتوقع
5	للاستفسار	Over-segmentation	سsegmented as two characters ن	للاستنفار
6	بالاضافة	Misrecognition	ةrecognized as ه	بالاضافة

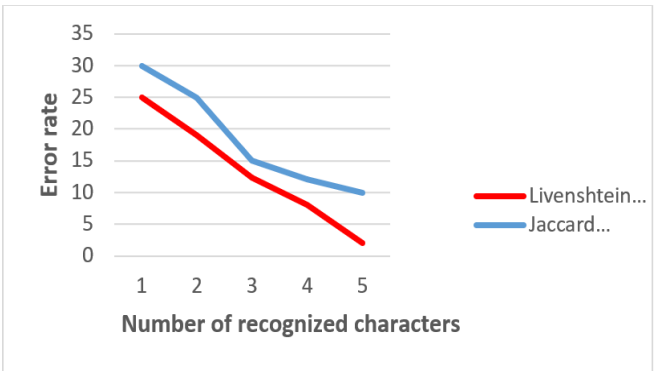

Figure 9. The growth of error rate

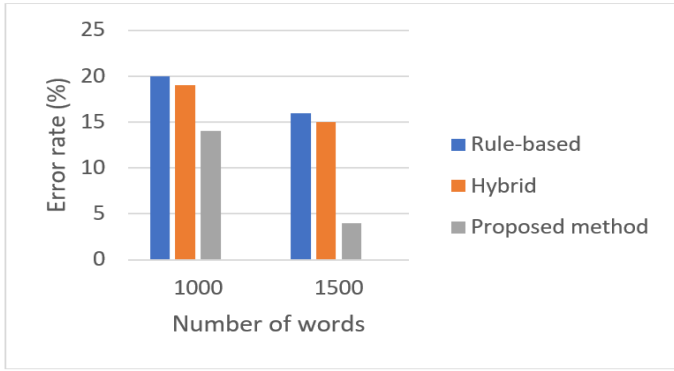

Figure 10. The growth of error rate

5. CONCLUSIONS

In this article, we introduced a novel method for recognizing Arabic handwritten characters by leveraging the Levenshtein distance in conjunction with a knowledge dataset. We made a comparison between the Jaccard distance and the Levenshtein distance to come up with a better distance to use in our approach. The idea of this approach was inspired by the functionality of the visual cortex and the hippocampus in the human brain. To test the new approach, we developed our personal Arabic database which contains a huge number of words from multiple disciplines (sports, culture, economics, religion…).

The experimental results indicate that our proposed system outperforms the current state-of-the-art approach. The recognition error rate ranged between 1% and 25%, contingent

upon the number of characters accurately identified.

In future works, we plan to imitate the functionality of the other areas of the human brain like the Wernicke and the Boca's areas to see the impact on the recognition rate, also we will enrich our personal dataset with new words in order to make it a benchmark dataset.

REFERENCES

[1] Breuel, T.M., Ul-Hasan, A., Al-Azawi, M.A., Shafait, F. (2013). High-performance OCR for printed English and Fraktur using LSTM networks. In 2013 12th International Conference on Document Analysis and Recognition, IEEE, Washington, DC, USA, pp. 683-687. https://doi.org/10.1109/ICDAR.2013.140

[2] Afli, H., Qui, Z., Way, A., Sheridan, P. (2016). Using SMT for OCR error correction of historical texts. Tenth International Conference on Language Resources and Evaluation (LREC 2016), Portorož, Slovenia, pp. 962–966

[3] Kolak, O., Resnik, P. (2005). OCR post-processing for low density languages. In Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 867-874.

[4] Blau, J., Blau, Y. (1981). The renaissance of Modern Hebrew and Modern Standard Arabic: Parallels and differences in the revival of two Semitic languages, Vol. 18. Univ of California Press.

[5] Hetzron, R. (Ed.). (1997). The Semitic Languages. Taylor & Francis.

[6] Horesh, U., Cotter, W.M. (2016). Current research on linguistic variation in the Arabic-speaking world. Language and Linguistics Compass, 10(8): 370-381. https://doi.org/10.1111/lnc3.12202

[7] Julian, G. (2020). What are the most spoken languages in the world. Retrieved May, 31(2020): 38. https://www.fluentin3months.com/most-spoken-languages/.

[8] Eviatar, Z., Ibrahim, R. (2014). Why is it hard to read Arabic?. Handbook of Arabic literacy: Insights and perspectives, 77-96. https://doi.org/10.1007/978-94-017-8545-7_4

[9] AbdelRaouf, A., Higgins, C.A., Khalil, M. (2008). A database for Arabic printed character recognition. In Image Analysis and Recognition: 5th International Conference, ICIAR 2008, Póvoa de Varzim, Portugal, Proceedings Springer Berlin Heidelberg, 5: 567-578. https://doi.org/10.1007/978-3-540-69812-8_56

[10] Kanoun, S., Slimane, F., Guesmi, H., Ingold, R., Alimi, A.M., Hennebert, J. (2009). Affixal approach versus analytical approach for off-line Arabic decomposable vocabulary recognition. In 2009 10th International Conference on Document Analysis and Recognition, IEEE, Barcelona, Spain, pp. 661-665. https://doi.org/10.1109/ICDAR.2009.264

[11] Stowe, L.A., Haverkort, M., Zwarts, F. (2005). Rethinking the neurological basis of language. Lingua, 115(7): 997-1042. https://doi.org/10.1016/j.lingua.2004.01.013

[12] Middlebrooks, E.H., Yagmurlu, K., Szaflarski, J.P., Rahman, M., Bozkurt, B. (2017). A contemporary framework of language processing in the human brain in the context of preoperative and intraoperative language mapping. Neuroradiology, 59: 69-87. https://doi.org/10.1007/s00234-016-1772-0

[13] Duff, M.C., Brown-Schmidt, S. (2012). The hippocampus and the flexible use and processing of language. Frontiers in Human Neuroscience, 6: 69. https://doi.org/10.3389/fnhum.2012.00069

[14] Najoua, B.A., Noureddine, E. (1995). A robust approach for Arabic printed character segmentation. In Proceedings of 3rd International Conference on Document Analysis and Recognition, Montreal, QC, Canada, 2: 865-868. https://doi.org/10.1109/ICDAR.1995.602038

[15] Amin, A., Masini, G. (1986). Machine recognition of multifont printed Arabic texts. Proceedings of the 8th International Conference on Pattern Recognition, pp. 392-295.

[16] Sari, T., Souici, L., Sellami, M. (2002). Off-line handwritten Arabic character segmentation algorithm: ACSA. In Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, IEEE, Niagra-on-the-Lake, ON, Canada, pp. 452-457. https://doi.org/10.1109/IWFHR.2002.1030952

[17] Margner, V. (1992). SARAT-A system for the recognition of Arabic printed text. In 11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems, IEEE, Computer Society, 1: 561-562. https://doi.ieeecomputersociety.org/10.1109/ICPR.1992.201841

[18] Hamid, A., Haraty, R. (2001). A neuro-heuristic approach for segmenting handwritten Arabic text. In Proceedings ACS/IEEE International Conference on Computer Systems and Applications, IEEE, Beirut, Lebanon, pp. 110-113. https://doi.org/10.1109/AICCSA.2001.933960

[19] El-Sawy, A., Loey, M., El-Bakry, H. (2017). Arabic handwritten characters recognition using convolutional neural network. WSEAS Transactions on Computer Research, 5(1): 11-19.

[20] Younis, K.S. (2017). Arabic hand-written character recognition based on deep convolutional neural networks. Jordanian Journal of Computers and Information Technology, 3(3).

[21] Elkhayati, M., Elkettani, Y. (2022). UnCNN: A new directed CNN model for isolated Arabic handwritten characters recognition. Arabian Journal for Science and Engineering, 47(8): 10667-10688. https://doi.org/10.1007/s13369-022-06652-5

[22] Fakhet, W., El Khediri, S., Zidi, S. (2022). Guided classification for Arabic Characters handwritten Recognition. In 2022 IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA), Abu Dhabi, United Arab Emirates, IEEE, pp. 1-6. https://doi.org/10.1109/AICCSA56895.2022.10017668

[23] Ahamed, P., Kundu, S., Khan, T., Bhateja, V., Sarkar, R., Mollah, A.F. (2020). Handwritten Arabic numerals recognition using convolutional neural network. Journal of Ambient Intelligence and Humanized Computing, 11: 5445-5457. https://doi.org/10.1007/s12652-020-01901-7

[24] Ashiquzzaman, A., Tushar, A.K. (2017). Handwritten Arabic numeral recognition using deep learning neural networks. In 2017 IEEE International Conference on

Imaging, Vision & Pattern Recognition (icIVPR), Dhaka, Bangladesh, IEEE, pp. 1-4. https://doi.org/10.1109/ICIVPR.2017.7890866
[25] Yujian, L., Bo, L. (2007). A normalized Levenshtein distance metric. IEEE Transactions On Pattern Analysis And Machine Intelligence, 29(6): 1091-1095. https://doi.org/10.1109/TPAMI.2007.1078
[26] Niwattanakul, S., Singthongchai, J., Naenudorn, E., Wanapu, S. (2013). Using of Jaccard coefficient for keywords similarity. In Proceedings of The International Multiconference of Engineers and Computer Scientists, Hong Kong, 1(6): 380-384.
[27] Doush, I.A., Alkhateeb, F., Gharaibeh, A.H. (2018). A novel Arabic OCR post-processing using rule-based and word context techniques. International Journal on Document Analysis and Recognition (IJDAR), 21: 77-89. https://doi.org/10.1007/s10032-018-0297-y

NOMENCLATURE

AHCR	Arabic Handwritten Characters Recognition
OCR	Optical Characters Recognition
CNN	Convolutional Neural Networks
SVM	Support Vector Machine
AHCD	Arabic Handwritten Characters Dataset
HACDB	Handwritten Arabic Characters DataBase
AIA9K	Arabic Isolated Alphabets 9K