Recognition of Arabic Air-Written Letters: Machine Learning, Convolutional Neural Networks, and Optical Character Recognition (OCR) techniques

Khalid M.O. Nahar *, Izzat Alsmadi, Rabia Al Mamlook *, Ahmad Nasayreh, Hasan Gharabieh, Ali Saeed Almuflih, Fahad Alasim

Posted Date: 27 September 2023

doi: 10.20944/preprints202309.1806.v1

Keywords: Arabic air writing recognition, Machine Learning, OCR, recognition, Deep Learning

Article

Recognition of Arabic Air-Written Letters: Machine Learning, Convolutional Neural Networks, and Optical Character Recognition (OCR) Techniques

Abstract: It is a challenging problem that air-written Arabic letters has received a lot of attention in the past decades when compared to commonly spoken languages like English languages. To fill this gap, we propose a strong model that brings together machine learning (ML) and optical character recognition (OCR) methods. The model applied several ML methods, (i.e., Neural Networks (NN), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM), with deep convolutional neural networks (CNNs) such as VGG16, VGG19, and SqueezeNet for effective feature extraction. Our study utilizes the AHAWP dataset, which consists of varied writing styles and variations in hand signs, to train and evaluate the model. Preprocessing systems are applied to improve data quality by reduction bias. Besides, OCR methods are combined into our model to sequestrate individual letters from continuous air-written gestures and refine recognition results. Results of this study show that the proposed model has achieved the extreme accuracy of 88.8% using NN with VGG16. This study presents an inclusive approach that combines ML, deep CNNs, and OCR methods to address the issue of Arabic in air writing recognition research.

Keywords: Arabic air writing recognition; machine learning; OCR; recognition; deep learning

1. Introduction

Advances in information technology have reshaped how humans interact with machines and programs, as well as how they communicate within their environment and language. The concept of writing in the air has emerged as a mode of communication between humans and their intelligent applications and devices, aligning with the flexibility of human mobility and surroundings [1]. This air-writing modality finds utility across diverse fields, including human-robot communication, children's education, aiding individuals with sensory challenges, and even in the realm of meta-verses [2]. The recognition of air-written Arabic letters introduces a tapestry of intricate challenges necessitating inventive solutions. Diverging from traditional written scripts, the fluid nature of air-written gestures introduces layers of complexity that warrant specialized approaches. The dynamic essence of air-written letters transcends the confined spatial boundaries of conventional paper or screens, engendering a rich spectrum of letter formations and trajectories. Unraveling and deciphering these ever-evolving spatial-temporal patterns emerges as a formidable hurdle. Further intricacy arises from the plethora of writing styles and hand sign variations that individuals employ. This myriad of expressions compounds the intricacies tied to precise interpretation and recognition of air-written Arabic letters. Furthermore, the task of demarcating distinct individual letters within the continuum of air-written gestures mandates meticulous segmentation techniques for unwavering recognition accuracy. Amplifying the challenge is the absence of predetermined reference points or demarcated start and end strokes in the realm of air-writing, necessitating ingenious algorithms capable of impeccably detecting letters despite the absence of conventional structural cues.

While the recognition of air-written English letters has undergone thorough exploration through a range of machine learning techniques, including LSTM [3], 2D-CNN , Faster RCNN [4], and 3D Reset [5], a notable gap exists in the realm of recognizing air-written Arabic letters. Despite the extensive research on English counterparts, no study has ventured into the domain of deciphering air-written Arabic letters. To surmount this complex landscape, our approach orchestrates a holistic methodology harmonizing the strengths of machine learning (ML) methods, deep convolutional neural networks (CNNs) [6], and the finesse of optical character recognition (OCR) techniques [7]. OCR, a process that transforms images of handwritten or typewritten text into machine-readable text,

 doi:10.20944/preprints202309.1806.v1

forms a key component of our strategy. This aligns with broader endeavors to process handwritten text from electronic devices, such as paper forms, invoices, and legal documents. However, grappling with a blend of machine-generated and human-written text poses distinct challenges, particularly in the public or governmental sectors. Effective handling and storage of such diverse inputs remain an ongoing concern, wherein OCR plays a pivotal role by converting text images into machine-readable text data.

The driving force behind this research is rooted in the inherent capability of air-writing recognition to bridge the divide between genuine human gestures and digital interfaces. As intelligent systems weave more intricately into our daily lives and the demand for seamless interaction burgeons, the capacity to decode and comprehend air-written letters assumes a pivotal significance. Moreover, the allure of Arabic script, epitomized by its intricate elegance, confers an additional layer of allure to this quest, impelling us to embark on a profound exploration within this uncharted territory. This research efforts to make a meaningful contribution by not only teaching Arabic to non-native speakers but also by providing interactive training for young learners to master the art of drawing Arabic letters. The intention is to foster an engaging experience with technology that stimulates curiosity and ignites creativity. Moreover, this technology carries the potential to assist individuals with speech challenges. Many individuals who struggle with speech are still capable of writing, and this application could be harnessed to track hand movements and convert them into synthesized human speech. It opens new avenues of communication and expression for those facing obstacles in traditional verbal interaction. The main objective of this research is to construct an intricate model that seamlessly intertwines machine learning and optical character recognition techniques. This model is particularly engineered to not only accurately identify individual Arabic characters, but also entire words formed in the air. Through achieving this ambitious target, the research has the potential to significantly enhance Arabic language education and to foster more inclusive and efficient modes of communication.

The rest of the paper is constituted as follows: Section 2 requires the existing literature on air-written Arabic letter recognition. Section 3 presents a methodology for the development and evaluation of the recognition model. Section 4 give to the results and analysis of the recognition model. Section 5 concludes the paper by discussing the implications of the research findings.

2. Related work

Researchers have shown considerable interest in the field of air writing recognition, extending beyond just numerical digits and symbols to encompass various languages. In this segment, we delve into prior investigations that have delved into the realm of air writing recognition, specifically focusing on numbers, symbols, and linguistic diversity.

2.1. Air Writing with Numbers and Symbols

A particularly noteworthy investigation [2] utilized radar-based methodologies to track the trajectories of hand motions while individuals engaged in the act of air-writing numerical digits. A multi-stream convolutional neural network (MS-CNN), coupled with continuous wave radar frequency, was enlisted to distinguish numbers ranging from 0 to 9. Impressively, this model achieved an impressive accuracy of 95% when tested with numerals air-written by 12 volunteers, shedding light on the potential of radar-based approaches in the realm of air-writing recognition.

Expanding beyond numbers, the realm of air writing encompasses hand-drawn symbols, holding significance across various applications such as encrypted codes and authentication systems. Another study [8] introduced an algorithm to extract the trajectories traced during air writing, accompanied by the development of deep CNN networks that encompass both 1D-CNN and 2D-CNN architectures for deciphering hand-drawn symbols. Diligent parameter optimization contributed to achieving a high discrimination rate of 99% using their CNN models. The dataset employed in their investigation covered numeric symbols (0-9) drawn in both clockwise and counterclockwise directions, along with a set of 16 directional symbols. The models underwent training and testing on well-segmented datasets, utilizing K-fold cross-validation to determine the

 doi:10.20944/preprints202309.1806.v1

optimal K value. A recognition rate of 5 yielded the most favorable results, underscoring the effectiveness of deep CNNs in symbol recognition.

Given the increasing ubiquity of smart devices, the realm of gesture-based communication has gained prominence in interactions with these devices. Addressing this trend, another study [3] directed their attention towards enhancing hand gesture communication through air writing. They devised a system for recognizing air-typed gestures, combining 3D trajectories with a fusion of long-term memory (LSTM) and convolutional neural networks (CNN). Preprocessing techniques such as normalization and root point translation were applied to the trajectory data. The evaluation was conducted using a dataset consisting of 2100 numerals, gathered by the authors themselves, resulting in an impressive accuracy rate of 99.32%.

While these studies represent significant progress in the realm of air-writing recognition concerning numbers and symbols, the exploration of air-writing recognition within the Arabic language remains relatively uncharted. This study takes on the challenge of bridging this gap by developing a model carefully tailored to distinguish air-written Arabic letters. Through a synergy of machine learning techniques and optical character recognition, our objective is to establish a benchmark of precise and robust recognition performance within the domain of Arabic air writing.

2.2. Exploring Air-Written Letters

Deciphering air-written letters presents a unique set of challenges, particularly when considering gesture recognition and handwriting analysis. Various research endeavors have delved into methods aimed at identifying and deconstructing handwritten or typed letters in the air. These efforts focus on achieving accurate segmentation of words and recognition of individual characters within air writing. A notable attempt by [9] involved employing hashing techniques and a CNN model to fragment letters, resulting in an impressive 92.91% accuracy on the NIST dataset. This achievement showcases promising strides in accurately recognizing fragmented letters. Similarly, in a study conducted by [1], a 2D-CNN model excelled in identifying letters and numbers written in the air, outperforming alternative methods. This highlights the efficacy of deep learning techniques in the realm of recognizing air-written letters and numbers. Another research initiative led by [4] compiled a dataset featuring hand movement videos in diverse settings and devised a recognition system for air-written letters. To train their dataset, they harnessed pre-trained models like Single Shot Detector (SSD) and Faster RCNN, achieving remarkable accuracies of up to 99%. Their work aimed to enhance the diversity and performance of prior datasets.

Within the domain of optical character recognition (OCR), [10] implemented a CNN and RNN-based OCR system for recognizing diacritics and Ottoman font in Arabic script. This effort yielded remarkable outcomes, with a validation accuracy of 98%, a word recognition rate (WRR) of 95%, and a character recognition rate (CRR) of 99% on the test dataset. Furthermore, endeavors have been directed towards recognizing handwritten Arabic letters in specific contexts. [11] introduced a segmentation algorithm for handwritten Urdu script lines, achieving accuracies of 96.7% for handwritten text and 98.3% for printed text. [12] proposed a technique grounded in the Faster RCNN framework for detecting and segmenting hand postures during air typing initiation, leading to a 96.11%-character recognition accuracy. Additional studies have tackled the recognition of Arabic letters in varied scenarios. [13] presented an optical system for character recognition that utilized the learning vector algorithm and classification techniques to discern well-written and poorly written Arabic letters, with recognition rates of 86.95% and 54.55%, respectively. [14] constructed a comprehensive OCR system relying on CNN and SVM, achieving a recognition rate of 99.25% for Arabic letters and numbers. [15] compared diverse classifiers, including SVM, kNN, Naïve Bayes, ANN, and ELM, to classify an array of gestures, attaining the highest accuracy of 96.95% with SVM.

Nevertheless, despite these strides, certain gaps persist in the literature concerning air-written Arabic letter recognition. A prevalent issue arises from researchers employing their own distinct datasets, which complicates model evaluation due to the lack of comparable benchmarks. Additionally, numerous studies have evaluated models on limited sample sizes, potentially inadequately reflecting real-world conditions. Further research is also warranted in the exploration

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

of advanced deep learning algorithms, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to enhance accuracy and resilience in recognizing air-written Arabic letters. While traditional machine learning techniques like SVM, Naïve Bayes, and ANN have been explored, there is room for further investigation.

The overarching objective of this research is to contribute practically to the advancement of gesture-based interaction, Arabic language education, and communication systems for individuals with disabilities. This pursuit aims to devise a scheme that facilitates effective and precise communication between humans and computer systems through air-written Arabic letters. By achieving these goals, the research seeks to enhance the identification of Arabic air-written letters by amalgamating machine learning, deep CNNs, and OCR approaches. Ultimately, the development of a reliable and accurate system capable of detecting and comprehending air-written Arabic characters across diverse applications holds the potential to enhance human-machine interaction and broaden access to learning and interacting in Arabic.

3. Arabic Air-Writing to Image Conversion and Recognition: Methodology

In this study, by identifying the hand's boundaries and turning the writing into an image, we present a model to recognize writing in the air. In our experiments, we used the AHAWP dataset [16] .Two models were created; the first relies on writing a single letter, and the second relies on writing a word in its whole. We tested extracting the crucial features from the photos for a single word using deep learning methods and training models like VGG16, VGG19, and SqueezeNet.Then, normalization is used to enhance performance, lessen sensitivity, and increase model stability. Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN) and Neural Networks (NN) are then employed as classification techniques for handwritten messages, as well as I2OCR for verification. The second model is based on predicting the completed handwritten word with the aid of I2OCR. displays the architecture of the proposed model. **Error! Reference source not found.** presentations architecture of proposed approach

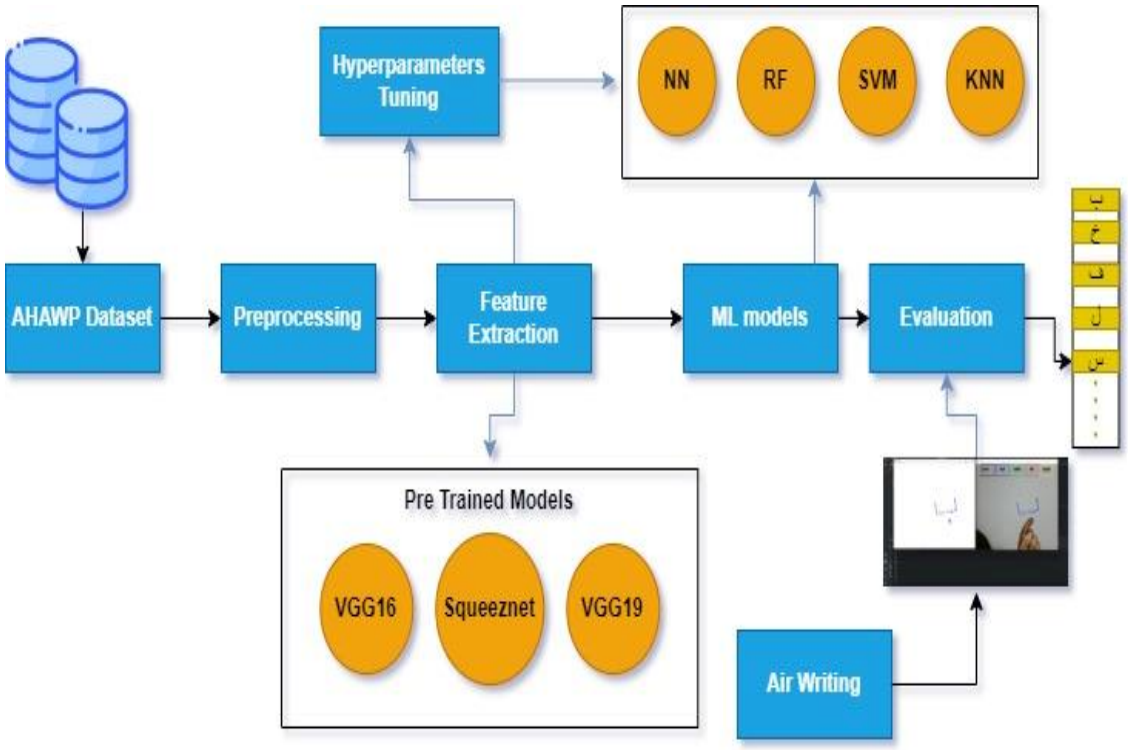


Figure 1. Architecture of Proposed Approach.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

3.1. AHAWP Dataset

In this research we utilized the AHAWP dataset [16] which consists of letters, words, and paragraphs. The dataset was gathered from 82 individuals. Includes 9,000 images of alphabets. These images display letters, in positions within words, such as at the beginning, middle or end. The dataset encompasses a total of 18 letters. To conduct our experiments, we split the dataset into a training set comprising 80% of the data and a testing set comprising the remaining 20%. This division allowed us to train our models on a portion of the data while also preserving a set for evaluating their performance. **Error! Reference source not found.** visually represents a sample, from the AHAWP dataset.

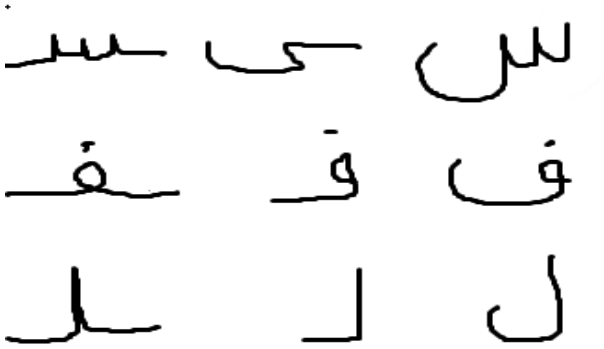


Figure 2. Sample of Dataset.

3.2. Data Pre-Processing:

Preprocessing is a step in enhancing the data sets' quality by eliminating data and preparing it for model development. In our study we focused on the processing steps for recognizing Arabic air written letters. These steps play a role in preparing the dataset for analysis and classification. Specifically, we carried out the following processing steps.

3.2.1. Image Resizing:

To ensure consistency among all samples we resized the thinned characters to a size like (224, 224). This resizing process guarantees that all character images have dimensions and facilitates feature extraction and analysis. We utilized libraries such as OpenCV or PIL to efficiently perform this image resizing operation.

3.2.2. Feature Extraction

Moving forward in our study, we delved into the task of drawing out pertinent details from the meticulously prepared character images. To accomplish this, we embraced a widely recognized method known as discrete cosine transformation (DCT), which holds a solid reputation for feature extraction. The DCT takes hold of the image by translating it into an ensemble of frequency coefficients that manage to encapsulate the crucial patterns and alterations inherent in the characters. The aim behind this feature extraction was to capture discerning cues that could serve as valuable guides in the forthcoming classification endeavor.

3.2.3. Dimensionality Reduction

With the features in hand, our next stride involved trimming down the complexity of the feature space through the art of dimensionality reduction. A reliable go-to in this realm is Principal Component Analysis (PCA) [17] , a technique celebrated for lighting the computational load and excising less informative features. In our exploration, we streamlined the feature count to a concise 99 for the VGG16, VGG19, and SqueezeNet models. This deft reduction in dimensionality crafts an avenue for streamlined analysis and the deft classification of our data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

3.2.4. Data Normalization

Concrete the way for data that's on the same playing field and setting the stage for optimal machine learning performance, we delved into the realm of data normalization. Our strategy homed in on refining our reduced feature vectors, ensuring they marched in harmony within a shared range, perhaps straddling the confines of -1 and 1. This normalization choreographed a balancing act, achieved through techniques like the tried-and-true min-max scaling or the trustworthy standardization. By taking this stride, we bid farewell to any biases stemming from dissimilar feature scales, ushering in a level ground for judicious comparisons as we transition into the eagerly awaited classification phase.

3.3. Building the Air Writing Components

The proposed method consists of several stages for building the Air Writing components, namely: (1) the development of Air Writing tools, (2) the implementation of i2OCR, (3) feature extraction, and (4) classification.

3.3.1. Air writing tools

The proposed approach expects the user to start by drawing on a canvas using hand gestures, which can be detected by a webcam and Google's Mediapipe library used for body key point detection. It initializes four arrays (bpoints, gpoints, rpoints, ypoints) to store points of different colors (e.g., blue, green, red, and yellow). These arrays are implemented as deques with a fixed maximum length of 1024. It also initializes four variables (blue index, green index, red index, and yellow index) to keep track of the current index in each array. The kernel (a morphological image processing operation for dilation and an array of color tuples) is then defined. A variable color index is initialized to store the current color index. Canvas is then constructed by creating a blank image and displaying it in a window called Paint as shown in **Error! Reference source not found.** .

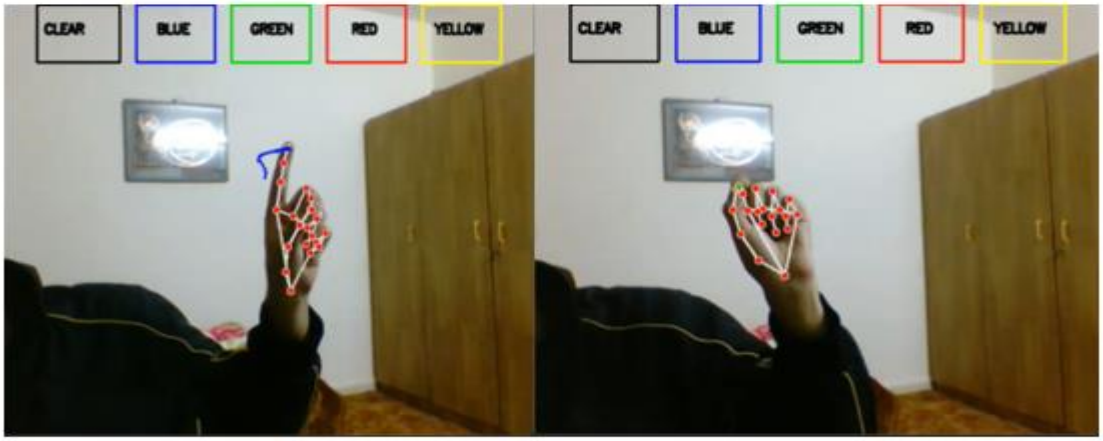


Figure 3. Example Shows Type of Writing in Left and Stopping Writing in Right.

The next step is to initialize the Media pipe hands, draw modules, and set up the webcam for capturing frames. In the main loop of the script, each frame is read from the webcam, flipped vertically, and converted to RGB color space.

UI elements are then drawn on the frame, including rectangles and text labels for the different colors and a 'CLEAR' button as shown in **Error! Reference source not found.**. The RGB frame is then passed to the Media pipe hands module to detect the hand gestures. If a hand is detected, the code draws a bounding box around the hand and gets the hand's key points (joints). The system then needs to check if the user has selected a color by clicking on one of the color buttons. If a color has been selected, the code gets the coordinates of the hand's palm and appends them to the appropriate color array. It also dilates the points in the color array to make them thicker on the canvas, start writing and in (B) side signs to stop writing. **Error! Reference source not found.** shows result of writing.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

After completing the writing, the result of the final appears on a white screen as shown in **Error! Reference source not found.** The next step is to draw the points on the canvas and display the updated canvas and frame in their respective windows. If the user clicks the 'CLEAR' button, the code clears all the points from the arrays and the canvas.

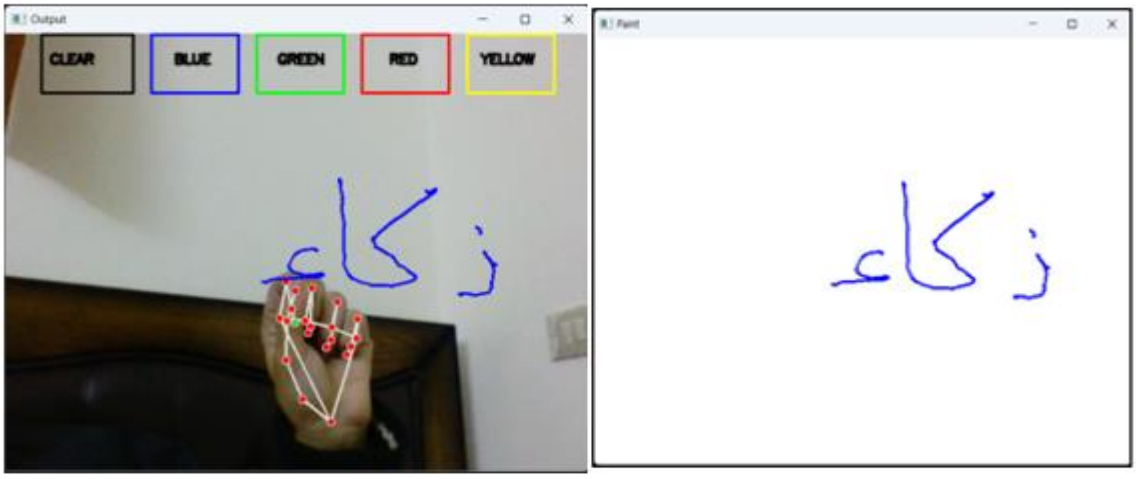


Figure 4. Result of Writing.

3.4. Optical character recognition (OCR)

In our study, we turn our attention to **Error! Reference source not found.**, where we delve into the intricate design of the Optical Character Recognition (OCR) system's underlying structure [18]. This methodology unfolds in a carefully choreographed sequence, with each stage performing a distinct role: image acquisition, pre-processing, text recognition, and post-processing. This primary stage aids as the cornerstone, involving the conversion of the image into binary data. This transformative process acts as the pivotal mechanism that empowers the OCR system's interpretive prowess.

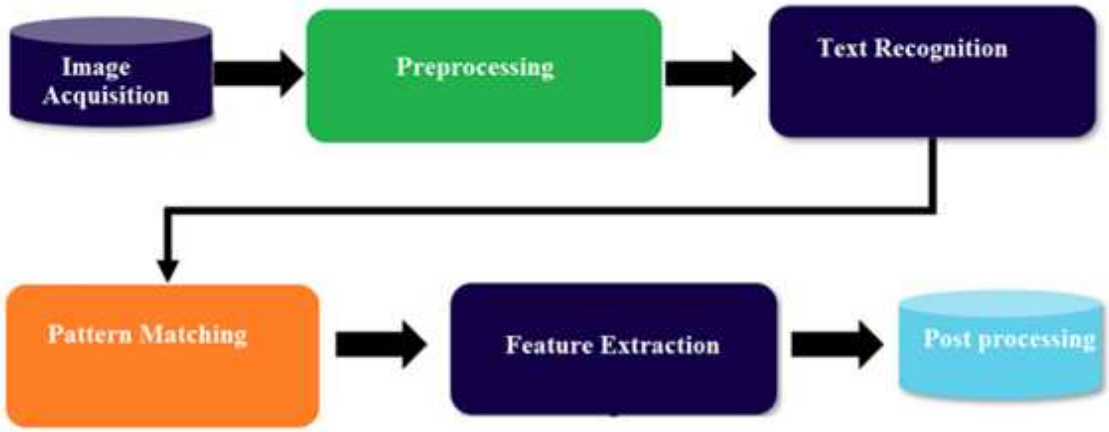


Figure 5. Architecture of the OCR.

In the subsequent phase, known as pre-processing, a suite of techniques is harnessed, encompassing image alignment, noise reduction, and language identification. These endeavors collectively aim to refine the image by eliminating imperfections and enhancing its readiness for subsequent recognition. Moving forward, the third stage, text recognition, unfolds. Here, the OCR system diligently employs a combination of pattern matching and feature extraction algorithms. This concerted effort enables the system to discern and identify the intricate characters that compose the image. As we reach the culmination of this multi-stage endeavor, the post-processing stage comes to the forefront. Within this domain, the OCR system undertakes crucial tasks, including necessary

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

rectifications and format adjustments. The overarching goal is to bestow a mantle of accuracy and uniformity upon the recognized text, ensuring its fidelity. In the context of our study, a prominent role is assumed by the freely accessible online tool, i2OCR . This tool is adept at extracting text from a diverse array of sources, spanning images to scanned documents, encompassing materials such as books, faxes, contracts, invoices, mail, passports, and ID cards. An impressive spectrum of linguistic diversity, spanning over 100 languages, can be effectively deciphered by i2OCR.

3.5. Arabic Air Writing Letter Recognition System Using Deep Convolutional Neural

This research employed Deep Convolutional Networks and Machine Learning models to achieve superior performance and accuracy compared to using them individually. The study specifically focused on combining Convolutional Neural Network (CNN) models for Arabic letter image classification with Machine Learning algorithms, such as Support Vector Machines (SVM) Neural Network (NN), Random Forest (RF) and K nearest neighbor (KNN), to enhance the overall system performance and accuracy. Three different models were applied in this study, namely VGG16, VGG19, and SqueezeNet, with the aim of assessing their effectiveness in achieving the desired outcomes. By leveraging the strengths of both Deep Convolutional Networks and Machine Learning models, this study provides a comprehensive approach to Arabic letter recognition that can have significant implications across various domains and applications as follows.

3.5.1. The VGGNet CNN architecture

The graph known as VGG19 [19] presents a refined iteration of VGG16, incorporating a thoughtful modification. This enhancement introduces two supplementary convolutional layers, each boasting 512 filters and adhering to a kernel size of (3x3), all while maintaining a stride of 1. The activation function employed in these layers remains consistent, embracing the ReLU framework. [20] A visual representation of both the VGG16 and VGG19 architectures shown in **Error! Reference source not found.** and **Error! Reference source not found.**.

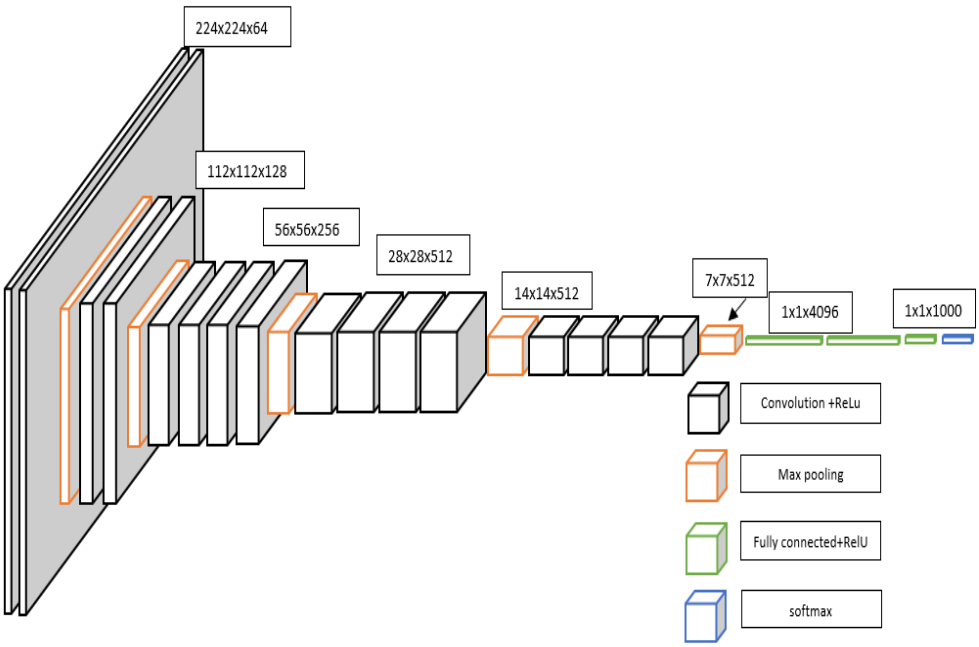


Figure 6. Architecture of VGG19.

Within the confines of this study, our focus zeroed in on the assessment of two distinct iterations of the VGGNet architecture: VGG16 and VGG19. The lineage of both VGG16 and VGG19 is traced back to their initial training on the extensive ImageNet dataset. This vast collection boasts a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

remarkable array of over a million meticulously annotated images, spanned across an impressive gamut of 1000 classes. Their debut marked a watershed moment in the realm of image classification, as their performance soared to the zenith of the ImageNet classification task, setting a formidable benchmark at the time of their inception.

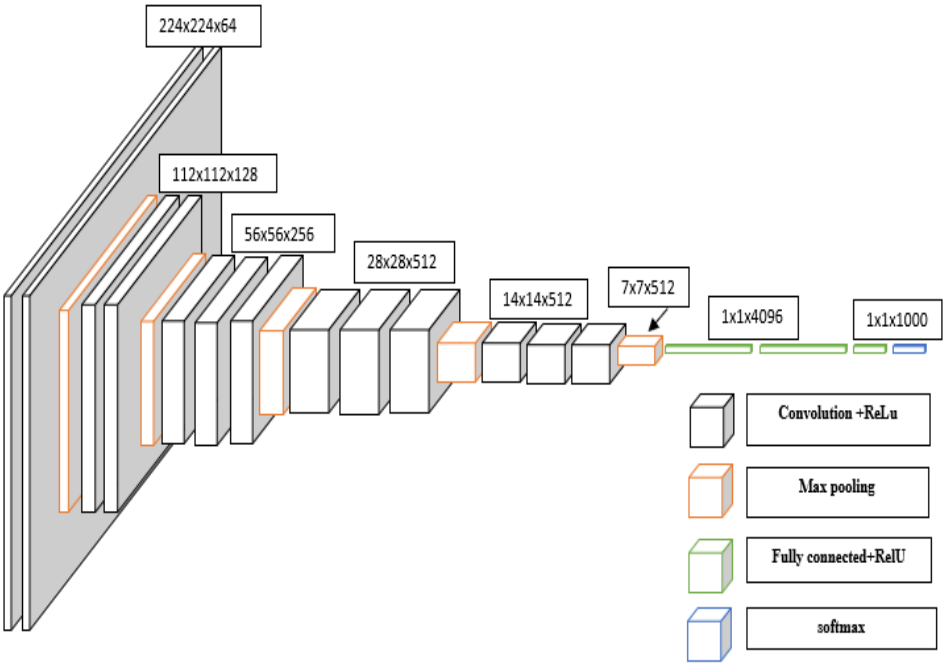


Figure 7. Architecture of VGG16.

The graph shows VGG16 as a profound convolutional neural network meticulously tailored for the domain of image classification tasks. The inception of this network commences with the input layer, primed to accommodate an image of dimensions 224x224x3. Here, the width and height unfurl across a span of 224 pixels, while the prism of color channels remains RGB. The intricate lattice of this network is woven with an array of layers, spanning the domains of convolutional, max pooling, and fully connected counterparts. The inaugural duo is comprised of convolutional layers, wherein each layer boasts a complement of 64 filters. These filters are cast over a kernel expanse of 3x3, harmoniously guided by a stride of 1. As the network forges ahead, the seventh and eighth strata don the attire of 512 filters, with the ninth and tenth layers echoing this design. In parallel, the VGG19 architecture emerges as a distinguished variant of VGG16, charting its course with an augmentation. This enhancement unfurls through the incorporation of two supplementary convolutional layers, each underpinned by a suite of 512 filters. Like kindred spirits, these layers reverberate with a kernel realm of 3x3, coupled with a stride attuned to the pulse of 1. United by the ReLU activation function, these layers amplify the expressive capacity of the architecture. Illustrative renderings of both the VGG16 and VGG19 in **Error! Reference source not found.** and **Error! Reference source not found.**, each epitomizing a formidable embodiment of convolutional mastery.

3.5.2. SqueezeNet architecture

The SqueezeNet architecture as shown in **Error! Reference source not found.**, introduce [21] , represents a convolutional neural network (CNN) tailored to excel in image classification tasks while being highly efficient. Its design incorporates multiple layers, including fully connected layers, convolutional and pooling layers, and fire modules, serving as compact and efficient building blocks to streamline the network's size. At the input layer, raw images are fed into the network, and the convolutional layers play a crucial role in extracting distinctive features from these input images. To achieve this, a combination of 3x3 and 1x1 convolutional filters is utilized. Furthermore, the pooling

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

layers come into play, effectively reducing the size of the feature maps and exercising control over the risk of overfitting. This judicious combination of architectural elements empowers SqueezeNet to deliver exceptional performance while maintaining remarkable efficiency, making it an ideal choice for a wide range of image classification applications.

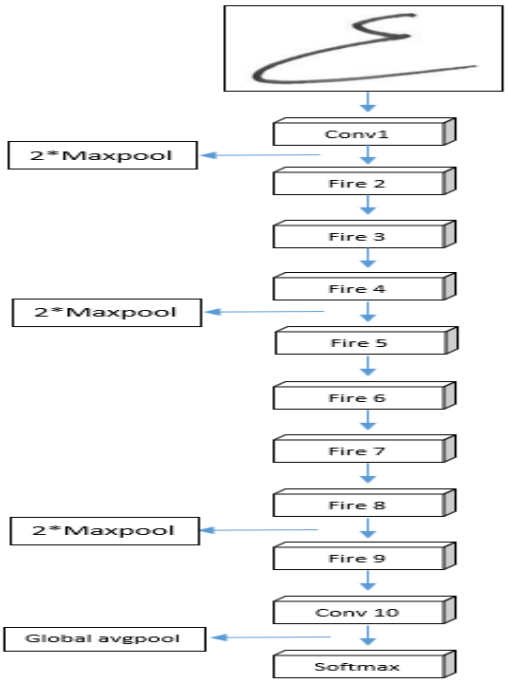


Figure 8. Architecture of SqueezeNet.

Max pooling is employed to downsize the feature maps, effectively reducing their size. The SqueezeNet architecture hinges on the pivotal fire modules, serving as its fundamental building blocks. These modules encompass a squeeze layer, which decreases the number of filters in the feature maps, and an expanded layer, which elevates the filter count using 1x1 and 3x3 convolutional layers. This ingenious design enables the network to maintain superior accuracy while being significantly smaller and faster compared to other CNNs. The ultimate classification decision is made by the fully connected layers, which comprise dense layers with a SoftMax activation function at the output layer. **Error! Reference source not found.** illustrates the architecture of SqueezeNet, demonstrating how its diverse layers and building blocks collaborate harmoniously to classify input images. Remarkably, the SqueezeNet architecture has been proven to excel in image classification tasks with remarkable efficiency. Its adept usage of fire modules, coupled with a combination of 3x3 and 1x1 convolutional filters, grants it the ability to achieve high accuracy while maintaining a compact network size, rendering it an ideal choice for scenarios where computational resources are limited.

3.6. Hyperparameters Tuning

Hyperparameter tuning plays a pivotal role in the realm of machine learning, especially when dealing with intricate models endowed with numerous parameters. Hyperparameters, being adjustable configurations, govern the learning process of a machine learning algorithm, standing apart from model parameters acquired during training. They directly influence the model's performance and its aptitude to generalize effectively. The process of hyperparameter tuning entails methodically exploring and identifying the optimal combination of hyperparameters to optimize the model's performance. The ultimate objective is to pinpoint the hyperparameter values that yield the highest accuracy, minimize error, or attain the best performance metric for a specific task. Among the various techniques for hyperparameter tuning, Grid Search and Random Search are two commonly employed methodologies.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

3.6.1. Grid Search

Grid Search serves as a fundamental technique for optimizing hyperparameters in machine learning [22] . Leveraging the power of cross-validation, this method enables us to train our models using various hyperparameter combinations, subsequently evaluating their performance to discover the most promising configurations.

Algorithm.1 Pseudo Code of Grid Search

```
Function Grid Search ():
    Hyperparameter Grid Search = Define Hyperparameter Grid Search
    Best Hyperparameters = None
    Best Performance = Select
    for Hyperparameter in Hyperparameter Grid Search
      Model = Set Hyperparameters in Model
      Performance = Evaluate Model
        if Performance > Best Performance
        Best Performance = Performance
        Best Hyperparameters = Hyperparameters
        END

    end
```

The process commences by selecting a specific set of hyperparameters for the model, which is then used in training and cross-validation. Through systematic exploration of the hyperparameter space, Grid Search identifies configurations that exhibit the highest performance during validation. These superior hyperparameter combinations are then seamlessly integrated into the model, guaranteeing optimal performance. In our implementation, we opt for K=5 in cross-validation, dividing the dataset into five subsets for thorough evaluation. Algorithm 2 provides a detailed illustration of Grid Search's operation. This sophisticated approach proves to be a powerful tool in attaining finely tuned hyperparameters, thereby significantly boosting the performance and efficiency of our machine learning model. By methodically exploring diverse hyperparameter combinations, Grid Search empowers us to pinpoint the best set of hyperparameters, leading to optimal model performance.

3.6.2. Random Search

Based on the experiments conducted by [23] , random search has been shown to outperform grid search in hyperparameter optimization. This method demonstrates exceptional efficiency in finding optimal models while minimizing computational time. Unlike grid search, random search explores broader areas by employing random sampling of hyperparameter combinations. Each set of hyperparameters is then evaluated, allowing the method to efficiently discover promising configurations. The key advantage of random search lies in its ability to explore hyperparameter spaces randomly, which often leads to the discovery of valuable configurations quickly. This streamlined approach presents a compelling alternative to grid search, showcasing its effectiveness in optimizing hyperparameters for machine learning models. Researchers and data scientists can benefit from adopting random search as a powerful tool to fine-tune their models and achieve enhanced performance with reduced computational overhead.

 doi:10.20944/preprints202309.1806.v1

3.7. Supervised Machine learning Models

In this paper, machine learning (ML) methods were leveraged to train and assess models dedicated to character image classification. To achieve peak performance, meticulous optimization of the model's parameters took place during its developmental phase. Through 10-fold cross-validation, the training set, comprising 80% of the original dataset, was subjected to rigorous evaluation alongside the testing set (20%). Optimal parameters yielding the highest accuracy were then incorporated into the final model. The study explored four renowned ML classification techniques, namely Support Vector Machines (SVM), Neural Networks (NN), Random Forests (RF), and K-Nearest Neighbors (KNN). Each of these methods received comprehensive scrutiny as part of the study's endeavors.

3.7.1. Support Vector Machines (SVM)

SVM, short for Support Vector Machine, stands as a widely favored machine learning algorithm suitable for both classification and regression tasks. The core concept behind SVM revolves around identifying the hyperplane that optimally separates the various classes within the dataset. The SVM architecture is relatively straightforward and comprises a few fundamental components. To begin, SVM is trained on a set of labeled data, which serves as the basis for determining the hyperplane that best segregates the different classes. In pursuit of superior performance, we employed two optimization techniques: grid search and random search, both of which helped us find the most optimal hyperparameters. The kernel, an essential part of SVM [24], plays a crucial role in mapping the data to higher dimensions, allowing for better class separation. We experimented with various kernel types, including linear, polynomial, and sigmoid kernels, evaluating their effectiveness through search-based optimization algorithms. Finally, the best kernel among these options was chosen to optimize the model's performance [25] .

3.7.2. Neural Network (NNs)

The Neural Network, also known as Artificial Neural Network (ANN), is a widely utilized machine learning model for classification tasks. This sophisticated model consists of three layers of nodes: input, hidden, and output. Each node applies a transfer function to the weighted sum of the nodes from the previous layer, along with a bias term. During the training process, the network's weights and parameters are iteratively updated using the provided dataset. Given our study's specific context involving multiple classifications among 18 classes, we employed the SoftMax activation function. This function's primary objective is to convert the network's output into predictive probabilities for each class, enhancing its ability to handle multiclass classification scenarios [26].

3.7.3. Random Forest (RF)

Random Forest (RF) is an important classifier in the field of machine learning and is known for its effectiveness and versatility. It belongs to the category of group learning methods, in which it constructs multiple decision trees by selecting random samples from a data set, and the accuracy is increased through the random construction of trees. And based on our study of the multiple classification between letters, the voting method was used, and the result is obtained based on the highest number of votes [27] .

3.7.4. K-Nearest Neighbors (KNN)

KNN, or K-Nearest Neighbors, is a versatile learning algorithm used for both classification and regression tasks [28]. Its foundation lies in assessing the similarity between data points, and it operates by determining the value of K, a positive number representing the number of nearest neighbors to consider when calculating distances. By iterating between K and new data points, the algorithm identifies the closest neighbors and assigns the category based on their votes. Choosing the appropriate value of K is crucial in KNN. A small K value may result in overfitting, where the model

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

becomes too sensitive to noise and specific data points, leading to reduced generalization. On the other hand, a large K value can cause underfitting, where the model oversimplifies and fails to capture intricate patterns in the data. In our study, we determine the class of each letter based on the majority vote of its nearest neighbors, considering their proximity. The class with the highest vote becomes the predicted class for the letter under consideration.

3.8. Evaluation of Models

When it comes to assessing the performance of both Deep Learning (DL) and traditional Machine Learning (ML) models for classification, a range of metrics comes into play, and accuracy holds a pivotal position among them. The confusion matrix emerges as a key player, furnishing vital details about both actual and predicted labels, thereby facilitating an in-depth analysis of the model's efficacy. To comprehensively evaluate the model's performance, the confusion matrix yields invaluable insights by breaking down the counts of True Negatives (TNs), True Positives (TPs), False Negatives (FNs), and False Positives (FPs). These metrics play a critical role in measuring the model's adeptness at making accurate distinctions across various classes. The accuracy is computed as depicted in equation (1):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

4. Result and discussion

4.1. Performance of Classifier of Algorithms

Error! Reference source not found. outlines the performance outcomes achieved by employing various classifiers on both CNN models (VGG19, VGG16, and SqueezeNet) and ML models (SVM, NN, RF, and KNN), employing three distinct optimization techniques: Grid Search, Random Search, and Default Parameters. The algorithms undergo thorough evaluation under different optimization approaches, namely Grid Search, Random Search, and Default Parameters. A closer examination of the performance disparities across diverse optimization strategies reveals noteworthy fluctuations in accuracy across different models and classifiers, as demonstrated in **Error! Reference source not found.**. The comparison of accuracy among ML classifiers yields intriguing insights. Notably, Grid Search yields an accuracy of 0.888 for the NN classifier when applied to the VGG16 model, closely followed by SVM with an accuracy of 0.855. However, the accuracy decreases to 0.843 when default parameters are employed.

Table 1. Performance of Classifier of Algorithms.

CNN models	ML models	Accuracy of Optimization methods		
	ML	Grid Search	Random Search	Default Parameters
VGG19	SVM	0.855	0.853	0.816
	NN	0.847	0.851	0.825
	RF	0.744	0.735	0.706
	KNN	0.727	0.727	0.692
VGG16	SVM	0.855	0.853	0.816
	NN	**0.888**	**0.847**	**0.843**
	RF	0.757	0.752	0.719
	KNN	0.751	0.751	0.699
SqueezeNet	SVM	0.819	0.799	0.770
	NN	0.825	0.823	0.813
	RF	0.729	0.724	0.695
	KNN	0.712	0.712	0.632

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

For the NN classifier, all optimization techniques yielded remarkable results, consistently achieving accuracies above 0.847 in VGG19. Both SVM and NN classifiers exhibited outstanding precision when applied to the VGG16 model. Specifically, the NN classifier attained the highest accuracy under Grid Search, registering 0.888, followed SVM at 0.855, and Default Parameters at 0.843. The comparison of performance across various optimization methods provides valuable insights into the efficacy of different approaches in fine-tuning the models for improved accuracy and precision. The NN classifier exhibited commendable performance with accuracy values of 0.888, 0.847, and 0.843 under Grid Search, Random Search, and Default Parameters, respectively. In the SqueezeNet model, the NN classifier achieved accuracies of 0.813 using default parameters, 0.823 with random searches, and 0.825. Particularly, the NN classifier consistently showcased outstanding results across all optimization techniques, maintaining accuracies surpassing 0.888. Comparing ML models, the SVM classifier consistently outperformed CNN models, particularly VGG19 and VGG16. Across every model and optimization technique for the AHAWP dataset, the neural network classification consistently outshined other classifiers. The RF and KNN classifiers generally demonstrated lower accuracy levels compared to the SVM and NN classifiers.

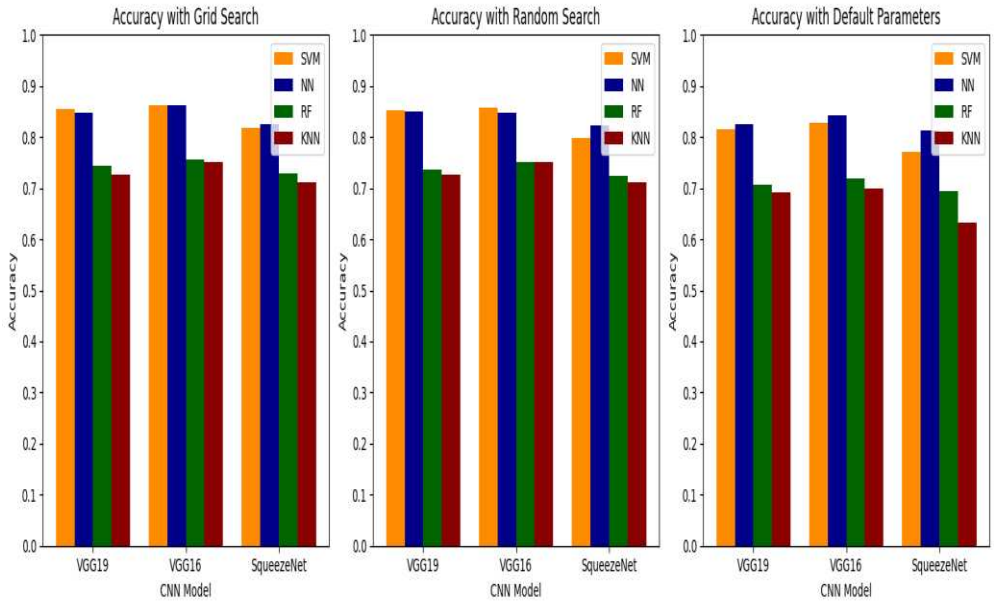


Figure 9. Comparison accuracy of the ML classifier performed.

4.2. Compared mean accuracy scores between models.

For Arabic handwritten recognition on the AHAWP dataset, the NN classifier continuously demonstrated the highest accuracy rates, reaffirming its potential in this domain. Results from the SVM classifier were also promising, particularly when used with the VGG16 model. However, future research should explore areas for improvement, as the RF and KNN classifiers exhibited lower accuracy results. The optimization approach significantly impacted accuracy, with Grid Search and Random Search consistently outperforming Default Parameters. The t-tests conducted in this investigation compared mean accuracy scores of several pairs of models, providing insights into their statistical significance. The calculated p-values indicated the likelihood of observed differences in mean accuracy being genuine, with a lower significance level indicating stronger evidence against the null hypothesis. In comparing the SVM classifier with other classifiers (NN, RF, and KNN) using the VGG16 CNN model, the p-value of 0.861 exceeded the conventional significance threshold of 0.05.

The p-values as shown in **Error! Reference source not found.** for the comparisons between SVM and RF and SVM and KNN are less than 0.05 which shows that the accuracy achieved by the SVM classifier differs significantly the p-values from the t-tests give us important information about the statistical significance of the variations in mean accuracy between the models.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

Table 2. Comparison of ML T-test and P-value.

T-test	P-value
SVM vs NN	0. 86123788
SVM vs RF	0.00280216
SVM vs KNN	0.00495305

These results add to our comprehension of the related Figure 10 results allow us to conclude that the SVM classifier performs similarly to the NN classifier, but that when utilizing the VGG16 CNN model, it greatly surpasses the RF and KNN classifiers when it comes to of accuracy. The Tukey's Honestly Significant Difference (HSD) test results yield significant insights regarding the pairwise differences in mean accuracy among the models being examined as shown in **Error! Reference source not found.**. Comparing the KNN model to the NN model, we observe a mean difference of 0.1173. The associated p-value is 0.0006, which falls below the significant threshold of 0.05. Consequently, we can conclude that there is a statistically significant difference in mean accuracy between the KNN and NN models. Specifically, the KNN model exhibits a higher mean accuracy compared to the NN model.

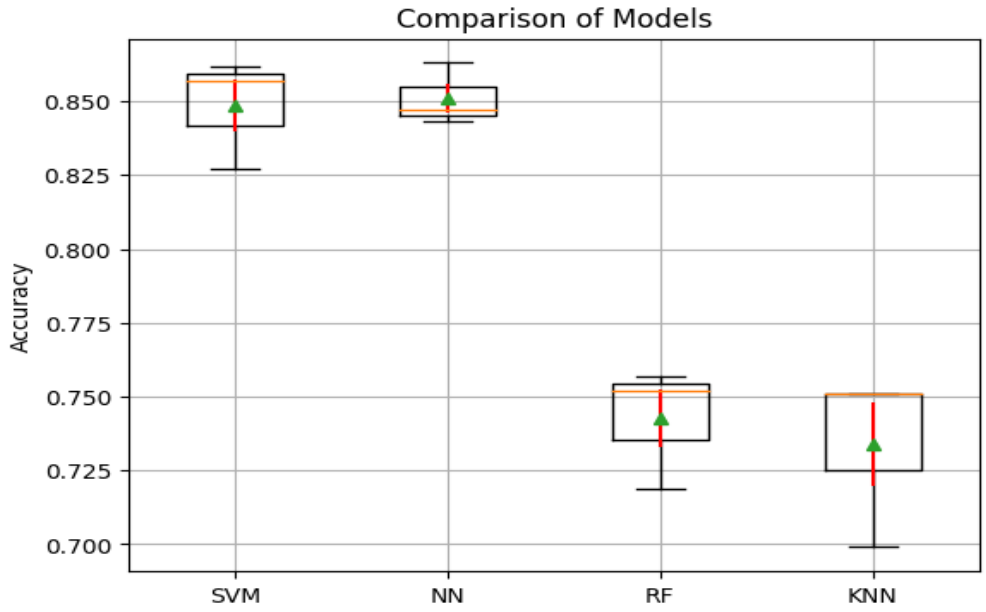


Figure 10. Comparison Box plot of ML models.

Table 3. The Tukey's Honestly Significant Difference (HSD) test results.

Group1	Group2	Mean diff	p-adj	lower	upper	reject
KNN	NN	0.1173	0.0006	0.0619	0.1728	True
KNN	RF	0.09	0.009519	-0.0464	0.0644	True
KNN	SVM	0.115	0.0007	0.0596	0.1704	True
NN	RF	-0.1083	0.0011	-0.1638	-0.0529	True
NN	**SVM**	**-0.0023**	**0.999**	**-0.0578**	**0.531**	**False**
RF	SVM	0.106	0.0013	0.0506	0.1614	True

However, when comparison the KNN and RF models, we discover a mean difference of 0.09 with a p-value of 0.009519. As the p-value goes above 0.05, we lack enough evidence to support a significant difference in mean accuracy among the KNN and RF models.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

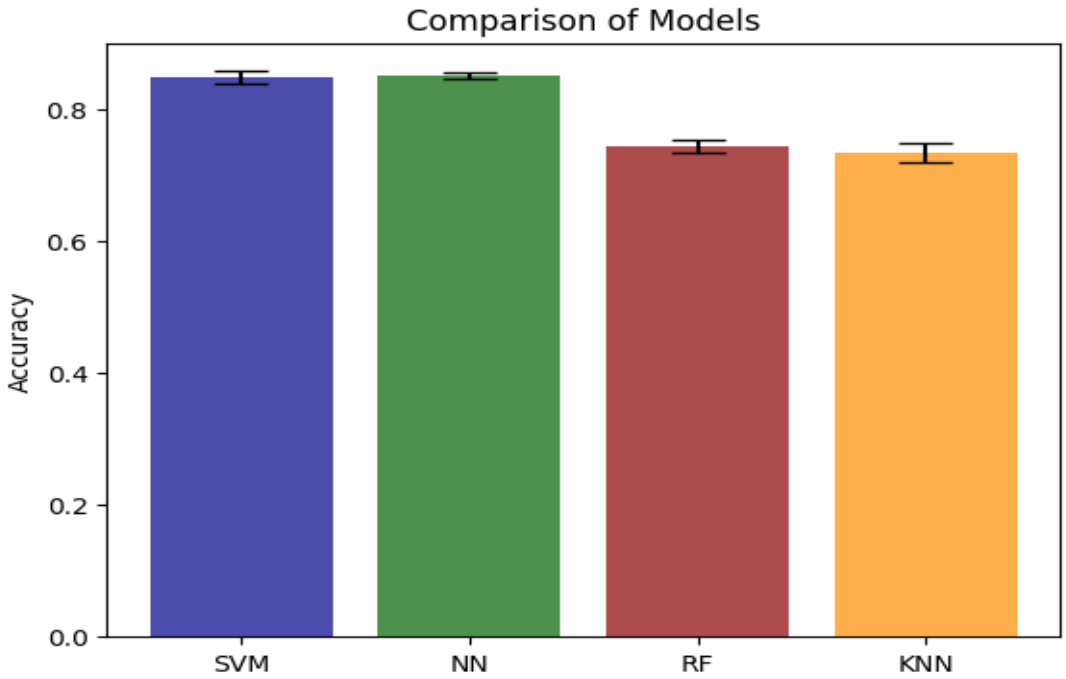
Additionally, upon closer examination of the KNN and SVM models, a mean difference of 0.115 with a p-value of 0.0007 becomes evident. Given that the p-value is below the significance threshold of 0.05, it indicates a statistically significant divergence in mean accuracy between the KNN and SVM models. Specifically, the KNN model showcases a notably higher mean accuracy in comparison to the SVM model.

Turning our attention to the contrast between the NN and RF models, a mean difference of -0.1083, denoted by the negative sign, emerges. This suggests that the NN model displays a slightly lower mean accuracy compared to the RF model. The corresponding p-value of 0.0011 further emphasizes a significant dissimilarity in mean accuracy between these two models, with the RF model demonstrating superior performance.

Conversely, the mean difference between the NN and SVM models is -0.0023, a value proximate to zero. The associated p-value of 0.999 implies that the mean accuracy of the NN and SVM models is essentially indistinguishable. As such, any observed fluctuations in accuracy between these models are more likely attributable to chance than a substantive performance variance.

Shifting our focus to the mean variance analysis between the RF and SVM models, a mean variance of 0.106 is identified, accompanied by a p-value of 0.0013. This p-value corroborates a significant discrepancy in mean accuracy. The outcomes of the Tukey's HSD test illuminate the nuances of the dissimilarities among the evaluated models in terms of mean accuracy. **Error! Reference source not found.** graphically elucidates the superiority of the NN model in mean accuracy, surpassing the other models. However, a discernible discrepancy in mean accuracy between the NN and SVM models is not discerned.

When juxtaposed with the alternative models, the NN model, the top performer, demonstrates enhanced accuracy in accurately classifying air-written letters as shown in **Error! Reference source not found.**. This suggests that the model possesses a greater capacity to adeptly detect and assign appropriate labels to the letters. **Error! Reference source not found.** illustrates the confusion matrix for the NN models, definitively identifying the NN model as the most proficient contender. The confusion matrix provides insightful revelations into the model's predictions, empowering us to gauge the precision of classifying air-written letters. Through the confusion matrix, we can meticulously assess the model's classification accuracy, identify any errors or zones of uncertainty, and decode the matrix where anticipated letter labels align with the columns, while actual letter labels align with the rows.



Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

Figure 11. Comparison Accuracy of Models.

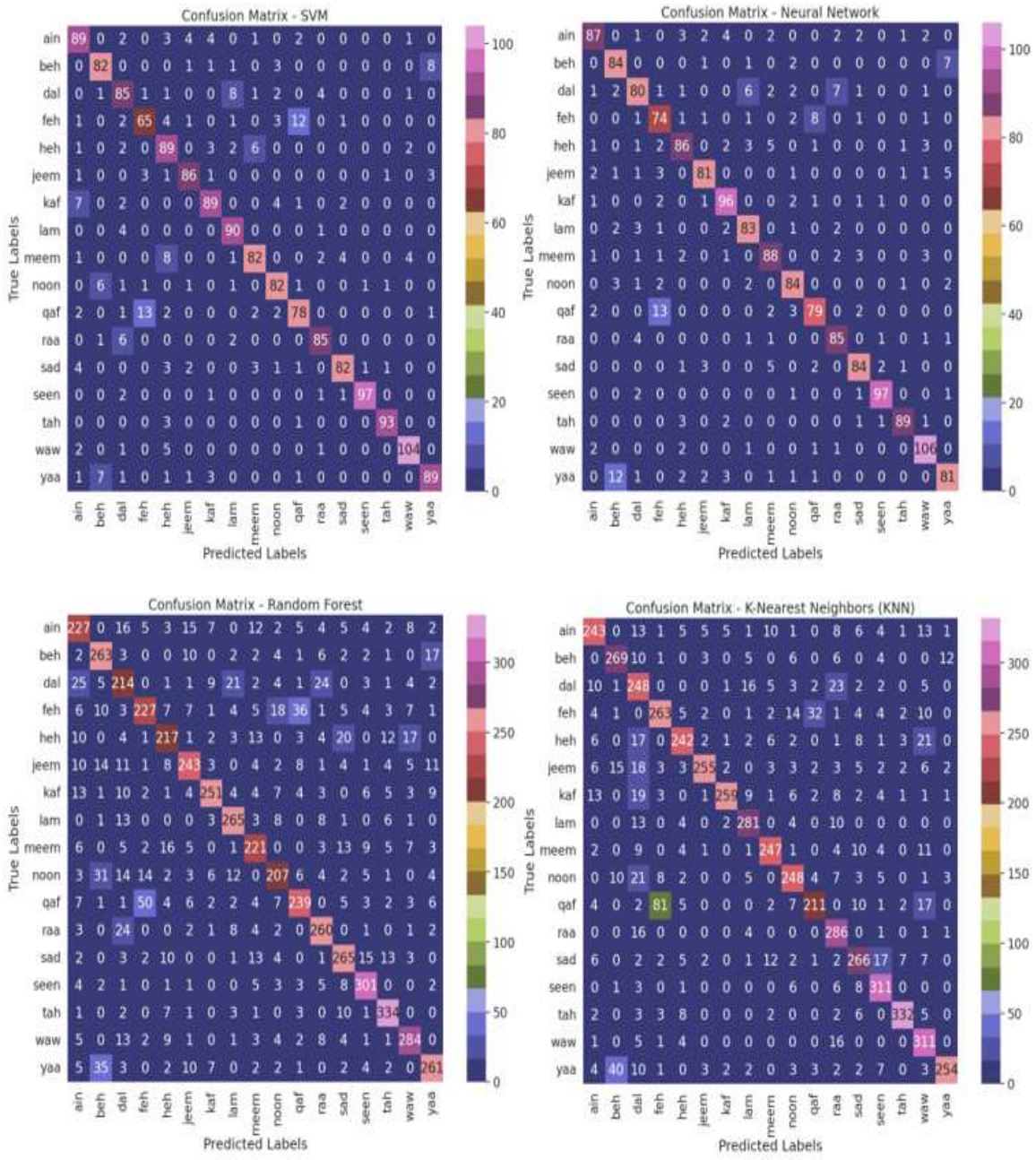


Figure 12. Confusion Matrix for All Models.

4.3. Sample of lettering writing by Experimental Setup

The aim of the experiment was to develop a model that combines machine learning (ML) with optical character recognition (OCR) methods to accurately identify Arabic letters written in the air. For the experimental setup, air writing was recorded using a laptop equipped with a Core i5 10th generation camera. The Python coding language and PyCharm development environment were employed for the implementation. To enhance computing speed, a NVIDIA GeForce GT graphics card was utilized. **Error! Reference source not found.** showcases a sample of the air-written lettering captured during the experiment.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

Figure 13. Sample of lettering writing.

4.4. Validation of our model

Validation of our model involved an elaborate process specifically designed for air-written Arabic letters in **Error! Reference source not found.**. The procedure commenced with capturing the letter as an image, which was then saved for further processing. To ensure optimal performance, the image underwent a series of preprocessing steps aimed at enhancing clarity and eliminating any unwanted artifacts. Among these steps were resizing the image and employing smoothing techniques to refine the letter's edges and visibility. Once the preprocessing phase was complete, the letter image was fed into a machine learning algorithm that had been meticulously trained on a dataset containing images of Arabic letters. This algorithm was purpose-built to leverage the knowledge gained from the training dataset's patterns and features, enabling it to predict and classify the input letter effectively. During the validation phase, the model employed a trained algorithm to make precise predictions for the air-written letter. By comparing the input letter with the patterns and characteristics it had internalized during training, the model skillfully identified its corresponding Arabic character. This prediction process proved invaluable in accurately recognizing and classifying the air-written letters, showcasing the robustness and efficacy of our developed approach.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

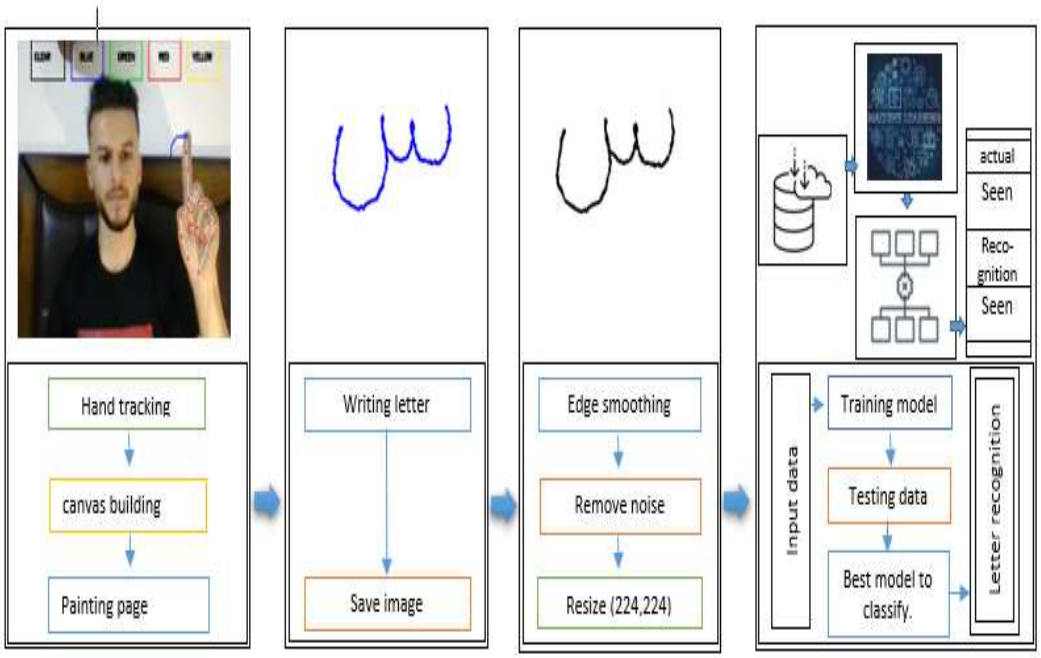


Figure 14. validation process for the air-written Arabic letters.

The validation results were evaluated based on the accuracy of the model's predictions as shown in **Error! Reference source not found.**. The accuracy represented the percentage of correctly identified letters out of the total validation set. This metric served as an indicator of the model's performance and its ability to accurately recognize air-written Arabic letters. The validation process played a crucial role in assessing the reliability and effectiveness of the developed model. It allowed us to verify the model's ability to accurately recognize individual Arabic characters based on air writing input. By comparing the model's predictions with the ground truth labels, we could determine the level of accuracy achieved and identify any areas for improvement. Our model's performance in recognizing air-written Arabic letters was assessed using predicted letter pairs. The true/false category was verified based on whether the model correctly detected the letter or not. The results **Error! Reference source not found.** . indicate that the model achieved a high level of accuracy in some cases, correctly predicting the letters Beh (ب), Ain (ع), Heh (ه), Jeem (ج), Kaf (ك), Meem (م), Noon (ن), Raa (ر), Sad (ص), Seen (س), and

Table 4. Actual and Prediction of our Model.

Actual	Prediction	True/False
Beh (ب)	Beh (ب)	T
Dal(د)	Ain(ع)	F
Ain(ع)	Ain(ع)	T
Feh(ف)	Qaf(ق)	F
Heh(ه)	Heh(ه)	T
Jeem (ج)	Jeem (ج)	T
Kaf (ك)	Kaf (ك)	T
Lam (ل)	Dal(د)	F
Meem(م)	Meem(م)	T
Noon(ن)	Noon(ن)	T
Qaf(ق)	Feh(ف)	F
Raa(ر)	Raa(ر)	T
Sad(ص)	Sad(ص)	T
Seen(س)	Seen(س)	T

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 September 2023 doi:10.20944/preprints202309.1806.v1

Tah(ت)	Tah(ت)	T
Waw(و)	Heh(ه)	F
Yaa(ي)	Beh (ب)	F

Tah (ت). However, there were instances where the model made incorrect predictions, such as misclassifying Dal (د) as Ain (ع), Feh (ف) as Qaf (ق), Lam (ل) as Dal (د), and Waw (و) as Heh (ه).

4.5. Comparison of the proposed model ***with Preview Work.***

Error! Reference source not found. offers a thorough comparison of the approaches and findings from earlier studies in the field of air writing recognition. In our study, we have suggested a model that combines machine learning (ML) and optical character recognition (OCR) methods to recognize Arabic air writing specially. Even if our model's accuracy might not be as high as some of the earlier research included in the table, it's crucial to consider the particulars of the Arabic script and its difficulties. Our study stands out as the first to concentrate especially on the recognition of Arabic air writing. This demonstrates its substantial impact on the industry. Although an 88% accuracy rate may seem lower in comparison to research focusing on English or other. The accuracy levels presented in the table should be interpreted with caution, as direct comparisons may not be appropriate due to variations in datasets, techniques, and language-specific characteristics. Each study examines a different language and writing system, necessitating unique methodologies and considerations. Despite these differences, our model's effectiveness in identifying Arabic air-written letters is evident through its impressive performance, demonstrating its value within the context of Arabic script.

Table 5. Comparison of the proposed model with Preview Work.

Paper	Languages	Method	Result
[1]	Air writing English	2D-CNN	accuracy: 91.24%
[3]	Air-writing English	LSTM	accuracy: 99.32%
[4]	English	Faster RCNN	accuracy: 94%
[5]	Air writing Korean and English	3D ResNet	Character error rate (CER): Korean: 33.16% English: 29.24%
[12]	Air-writing English	Faster RCNN	mean accuracy: 96.11 %
[29]	Air writing English	-	error rate: 0.8%
[30]	Air writing English	MS-CNN	accuracy: 95%
[31]	Air writing Hindi	PointNet	recognition rate: >97%
Our Model	**Air writing Arabic**	**Hybrid Model VGG16+NN**	**Accuracy :88%**

Our research addresses a significant gap in the field of air writing identification by focusing on Arabic script, which paves the way for future advancements and practical applications. This highlights the importance of devising tailored strategies that address the specific challenges posed by various writing systems and languages. Moreover, our study contributes not only to the knowledge base of Arabic air writing but also establishes itself as the first investigation into Arabic air writing recognition. While previous studies have primarily focused on English or other languages, we recognized the importance of understanding the unique complexities and characters within Arabic script. By exploring this specific context, our study enriches the knowledge of air writing recognition techniques for Arabic. Our model effectively combines Machine Learning (ML) and Optical Character Recognition (OCR) methods to accurately recognize Arabic air-written letters, offering a comprehensive solution that leverages the strengths of both approaches. As pioneers in Arabic air writing recognition, our research lays a solid foundation for future investigations in this area. This opens new avenues for researchers to explore additional methods and techniques that can further enhance recognition accuracy in Arabic air writing. As we move forward, these findings will

undoubtedly spur further advancements and innovations in the realm of air writing recognition for Arabic and other languages.

5. Conclusions and future work

This paper presented a novel approach for recognizing air-written Arabic letters using Machine Learning (ML) and DNNs, and OCR methods models. The results demonstrated that the NN method, along with the features extracted from VGG19, achieved high accuracy and low error rates in recognizing Arabic letters. Additionally, the study explored the use of optical character recognition as an alternative method for identifying air-written letters, showing comparable outcomes to ML models. Notably, our model performed exceptionally well on the AHAWP dataset and exhibited promising results when compared to other models. This research contributes to the expanding body of knowledge in Arabic handwriting recognition and machine learning, offering potential applications in education and aiding individuals with physical disabilities in communication. Future studies should consider investigating hybrid ML and their purpose in the recognition of air-written Arabic letters. The utilized model can discover practical use in several applications involving the recognition of children's Arabic air writing.

Funding: This research was funded by the Deanship of Scientific Research, King Khalid University, Kingdom of Saudi Arabia, under grant number KKU-IFP2-H-14. The authors extend their appreciation to the Ministry of Education in KSA for funding this research work through the project number KKU-IFP2-H-14.

Reference

1. F. Al Abir, M. Al Siam, A. Sayeed, M. A. M. Hasan, and J. Shin, "Deep Learning Based Air-Writing Recognition with the Choice of Proper Interpolation Technique," *Sensors (Basel).*, vol. 21, no. 24, pp. 1–15, 2021, doi: 10.3390/s21248407.
2. S. Ahmed, W. Kim, J. Park, and S. H. Cho, "Radar-Based Air-Writing Gesture Recognition Using a Novel Multistream CNN Approach," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23869–23880, 2022, doi: 10.1109/JIOT.2022.3189395.
3. S. Imtiaz and N. Kim, "Neural Network and Depth Sensor," 2020.
4. A. K. Choudhary, N. Dua, B. Phogat, and S. U. Saoji, "Air Canvas Application Using Opencv and Numpy in Python," *Int. Res. J. Eng. Technol.*, vol. 08, no. 08, pp. 1761–1765, 2021.
5. U. H. Kim, Y. Hwang, S. K. Lee, and J. H. Kim, "Writing in the Air: Unconstrained Text Recognition From Finger Movement Using Spatio-Temporal Convolution," *IEEE Trans. Artif. Intell.*, 2022, doi: 10.1109/TAI.2022.3212981.
6. L. Bouchriha, A. Zrigui, S. Mansouri, S. Berchech, and S. Omrani, "Arabic Handwritten Character Recognition Based on Convolution Neural Networks," *Commun. Comput. Inf. Sci.*, vol. 1653 CCIS, no. 8, pp. 286–293, 2022, doi: 10.1007/978-3-031-16210-7_23.
7. N. Altwaijry and I. Al-Turaiki, "Arabic handwriting recognition system using convolutional neural network," *Neural Comput. Appl.*, vol. 33, no. 7, pp. 2249–2261, 2021, doi: 10.1007/s00521-020-05070-8.
8. C. H. Hsieh, Y. S. Lo, J. Y. Chen, and S. K. Tang, "Air-Writing Recognition Based on Deep Convolutional Neural Networks," *IEEE Access*, vol. 9, pp. 142827–142836, 2021, doi: 10.1109/ACCESS.2021.3121093.
9. M. Khandokar, I and Hasan, M and Ernawan, F and Islam, S and Kabir, "Handwritten character recognition using convolutional neural network," *J. Phys. Conf. Ser.*, vol. 1918, p. 042152, 2021, doi: 10.1088/1742-6596/1918/4/042152.
10. M. Mohd, F. Qamar, I. Al-Sheikh, and R. Salah, "Quranic optical text recognition using deep learning models," *IEEE Access*, vol. 9, pp. 38318–38330, 2021, doi: 10.1109/ACCESS.2021.3064019.
11. S. Malik *et al.*, "An efficient skewed line segmentation technique for cursive script OCR," *Sci. Program.*, vol. 2020, 2020, doi: 10.1155/2020/8866041.
12. S. Mukherjee, S. A. Ahmed, D. P. Dogra, S. Kar, and P. P. Roy, "Fingertip detection and tracking for recognition of air-writing in videos," *Expert Syst. Appl.*, vol. 136, pp. 217–229, 2019, doi: 10.1016/j.eswa.2019.06.034.
13. M. A. Fadeel, "Off-line optical character recognition system for arabic handwritten text," *J. Pure Appl. Sci.*, vol. 18, no. 4, pp. 41–46, 2019.
14. G. Sokar, E. E. Hemayed, and M. Rehan, "A Generic OCR Using Deep Siamese Convolution Neural Networks," *2018 IEEE 9th Annu. Inf. Technol. Electron. Mob. Commun. Conf. IEMCON 2018*, pp. 1238–1244, 2019, doi: 10.1109/IEMCON.2018.8614784.

15. S. Misra, J. Singha, and R. H. Laskar, "Vision-based hand gesture recognition of alphabets, numbers, arithmetic operators and ASCII characters in order to develop a virtual text-entry interface system," *Neural Comput. Appl.*, vol. 29, no. 8, pp. 117–135, 2018, doi: 10.1007/s00521-017-2838-6.
16. M. A. Khan, "Arabic handwritten alphabets, words and paragraphs per user (AHAWP) dataset," *Data Br.*, vol. 41, p. 107947, 2022, doi: 10.1016/j.dib.2022.107947.
17. H. Abdi and L. J. Williams, "Principal component analysis. wiley interdisciplinary reviews: computational statistics," *Wiley Interdisplinary Rev. Comput. Stat.*, pp. 1–47, 2010.
18. "I2OCR", [Online]. Available: https://www.i2ocr.com/
19. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.
20. A. S. Jaradat *et al.*, "Automated Monkeypox Skin Lesion Detection Using Deep Learning and Transfer Learning Techniques," *Int. J. Environ. Res. Public Health*, vol. 20, no. 5, 2023, doi: 10.3390/ijerph20054422.
21. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," pp. 1–13, 2016, [Online]. Available: http://arxiv.org/abs/1602.07360
22. B. H. Shekar and G. Dagnew, "Grid search-based hyperparameter tuning and classification of microarray cancer data," *2019 2nd Int. Conf. Adv. Comput. Commun. Paradig. ICACCP 2019*, no. February, pp. 1–8, 2019, doi: 10.1109/ICACCP.2019.8882943.
23. J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.
24. R. E. Al Mamlook, A. Nasayreh, H. Gharaibeh, and S. Shrestha, "Classification Of Cancer Genome Atlas Glioblastoma Multiform (TCGA-GBM) Using Machine Learning Method," *IEEE Int. Conf. Electro Inf. Technol.*, vol. 2023-May, no. July, pp. 265–270, 2023, doi: 10.1109/eIT57321.2023.10187283.
25. S. V. N. Vishwanathan and M. N. Murty, "SSVM: A simple SVM algorithm," *Proc. Int. Jt. Conf. Neural Networks*, vol. 3, no. 1, pp. 2393–2398, 2002, doi: 10.1109/ijcnn.2002.1007516.
26. N. Kriegeskorte and T. Golan, "Neural network models and deep learning," *Curr. Biol.*, vol. 29, no. 7, pp. R231–R236, 2019, doi: 10.1016/j.cub.2019.02.034.
27. G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, no. 2, pp. 197–227, 2016, doi: 10.1007/s11749-016-0481-7.
28. S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning k for kNN Classification," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 3, 2017, doi: 10.1145/2990508.
29. M. Chen, G. AlRegib, and B. H. Juang, "Air-Writing Recognition - Part I: Modeling and Recognition of Characters, Words, and Connecting Motions," *IEEE Trans. Human-Machine Syst.*, vol. 46, no. 3, pp. 403–413, 2016, doi: 10.1109/THMS.2015.2492598.
30. S. Ahmed, W. Kim, J. Park, and S. H. Cho, "Radar Based Air-Writing Gesture Recognition Using a Novel Multi-Stream CNN Approach," *IEEE Internet Things J.*, vol. PP, no. 8, p. 1, 2022, doi: 10.1109/JIOT.2022.3189395.
31. J. K. Sharma, "Highly Accurate Trimesh and PointNet based algorithm for Gesture and Hindi air writing recognition Highly Accurate Trimesh and PointNet," no. Lmc, pp. 0–17, 2023.