Neural Processing Letters (2024) 56:115
https://doi.org/10.1007/s11063-024-11544-w

CNN-based Methods for Offline Arabic Handwriting Recognition: A Review

Mohsine El Khayati[1] · **Ismail Kich**[2] · **Youssef Taouil**[3]

Accepted: 16 January 2024 / Published online: 19 March 2024

Abstract
Arabic Handwriting Recognition (AHR) is a complex task involving the transformation of handwritten Arabic text from image format into machine-readable data, holding immense potential across various applications. Despite its significance, AHR encounters formidable challenges due to the intricate nature of Arabic script and the diverse array of handwriting styles. In recent years, Convolutional Neural Networks (CNNs) have emerged as a pivotal and promising solution to address these challenges, demonstrating remarkable performance and offering distinct advantages. However, the dominance of CNNs in AHR lacks a dedicated comprehensive review in the existing literature. This review article aims to bridge the existing gap by providing a comprehensive analysis of CNN-based methods in AHR. It covers both segmentation and recognition tasks, delving into advancements in network architectures, databases, training strategies, and employed methods. The article offers an in-depth comparison of these methods, considering their respective strengths and limitations. The findings of this review not only contribute to the current understanding of CNN applications in AHR but also pave the way for future research directions and improved practices, thereby enriching and advancing this critical domain. The review also aims to uncover genuine challenges in the domain, providing valuable insights for researchers and practitioners.

Keywords Arabic handwriting recognition · Convolutional neural networks · Deep learning

Abbreviations

CNN	Convolutional neural network
FCN	Fully convolutional network
ANN	Artificial neural network
LSTM	Long short-term memory
BLSTM	Bidirectional long short-term memory

✉ Mohsine El Khayati
Mohsine.Elk@gmail.com

[1] Mathematics Department, Faculty of Science, University Ibn Tofail, Kenitra, Morocco

[2] Computer Science Department, Faculty of Science, University Ibn Tofail, Kenitra, Morocco

[3] Computer Engineering and Mathematics Department, Higher School of Technology, University Cadi Ayyad, Essaouira, Morocco

MDLSTM	Multidimensional long short-term memory
CTC	Connectionist temporal classification
RNN	Recurrent neural network
ML	Machine learning
DL	Deep learning
CL	Convolutional layer
FC	Fully connected layer
TL	Transfer learning
SVM	Support vector machine
KNN	K-nearest neighbors
AHR	Arabic handwriting recognition
AHCR	Arabic handwritten character recognition
AHDR	Arabic handwritten digit recognition
AHWR	Arabic handwritten word recognition
PAW	Piece of Arabic word

1 Introduction

Arabic, recognized as an official United Nations language since December 18, 1973, boasts over 400 million speakers worldwide [1]. Predominantly spoken in the 26 Arabic countries spanning the Middle East and North Africa, Arabic holds significant global influence. Its widespread use extends beyond its native speakers, as over 1 billion Muslims employ Arabic in religious practices. Moreover, the Arabic alphabet is utilized by languages like Farsi, Kurdish, Urdu, and Jawi [2]. In both printed and handwritten forms, Arabic text is prevalent, reflecting its widespread presence.

Arabic Handwriting Recognition (AHR) involves converting handwritten Arabic text from image format into a machine-readable form. This operation is crucial for various applications, including digitizing historical documents, processing bank checks automatically, and handling forms. Figure 1 illustrates some key real-world applications of AHR. With its wide-ranging uses, AHR has garnered significant interest. However, compared to other Latin languages and printed text recognition, AHR is challenging due to the distinctive characteristics of handwritten Arabic text [3]. The cursive nature, diacritics, artistic writing style, over-traces, overlaps, touching components, and vertical ligatures all contribute to the complexities of AHR.

However, recent progress in both software and hardware capabilities has significantly enhanced the feasibility of implementing robust AHR tools. Notably, Convolutional Neural Networks (CNN) have emerged as a dominant force in computer vision. CNNs, deep Feedforward Artificial Neural Networks (ANN), are renowned for their prowess in addressing complex problems, learning nonlinear mappings from high-dimensional data [4], and extracting salient invariant features [5]. In recent times, CNNs have found extensive applications in AHR, contributing to notable improvements in this domain.

Numerous surveys have explored AHR, with most of them underscoring the effectiveness and dominance of CNNs over other algorithms in recent years. Balaha et al. [2] provided a thorough examination of AHR phases, spanning from image retrieval to character classification, with a primary emphasis on character recognition. While offering a robust background on AHR phases and techniques employed in the literature, the review falls short in providing an in-depth analysis of the specific contributions within

Fig. 1 Some AHR applications

these phases. Alghyaline's review [6] comprehensively addressed printed and handwritten, offline, and online Arabic text recognition. The work highlighted the scarcity of research on online OCR systems and the lack of satisfactory commercial tools for recognizing Arabic printed text. Emphasizing a significant gap, they underscored the absence of works addressing entire scripts, as prevalent literature predominantly focuses on isolated words and characters. Distinguishing itself from other reviews, Alghyaline's work additionally discussed commercial and open-source software in this context, placing a strong emphasis on real-world applications. Alrobah and Albahli [7] focused on Deep Learning (DL) techniques, specifically for Arabic Handwritten Character Recognition (AHCR) and Arabic Handwritten Word Recognition (AHWR). Among the 37 reviewed papers in their article, 27 employed CNNs, underscoring their efficacy for AHCR. The study favored hybrid CNN models and particularly highlighted the success of models trained from scratch. However, a significant challenge highlighted in the study is the scarcity of training data in this domain. In a similar context, Ahmed et al. [8] conducted a review of DL techniques for AHR, with a specific focus on AHCR. Similarly, there is a discernible prevalence of CNNs in works published from 2016 onwards. This trend is also evident in the review conducted by Nahla and Abd-Alsabour [9].

Discussed reviews, as well as existing literature works, underscored the recent dominance of CNNs compared to other techniques in AHR, prompting the need for a dedicated literature review on CNN applications in this field. Therefore, this study aims to fill this gap by providing a focused survey of CNN-based methods applied in AHR. Our review compiles 62 carefully selected papers from major databases (IEEE Xplore, Springer, Science Direct, ACM Digital Library). The analysis considers criteria like dataset, CNN architecture, techniques, and recognition/segmentation accuracy. This concise yet comprehensive survey aims to guide the research community in this area,

offering insights that go beyond existing reviews. Additionally, this research aims to provide novice researchers with a clear understanding of the state-of-the-art results of CNNs in AHR, serving as a valuable resource for those entering the field.

The main contributions of this work are:

- **Contextualizing CNNs in AHR Literature**: While other reviews have explored DL techniques, none have exclusively focused on CNN-based methods in the context of AHR. We consolidate and analyze existing literature reviews on AHR, emphasizing the prominence of CNNs in recent studies.
- **Addressing Neglected Segmentation Phases**: Previous reviews on Machine Learning (ML) or DL for AHR often neglected segmentation phases in AHR systems. In contrast, our review comprehensively covers both recognition and segmentation phases, providing a holistic view of the challenges and advancements in the field.
- **Coverage of Recognition Aspects**: Our review spans the entire spectrum of recognition, encompassing digits, characters, and words. This broad coverage is essential for a nuanced understanding of the state-of-the-art results and emerging trends in CNN-based methods for AHR.
- **Comprehensive Presentation of AHR Field**: We present a comprehensive overview of AHR characteristics, challenges, phases, and existing databases.
- **Discussion of Open Problems and Perspectives**: Engaging in a thorough discussion, our review highlights open problems in AHR, providing valuable perspectives for field enhancement.
- **Contribution to AHR Improvement**: Recognizing the need for continued improvement in the AHR domain, our review highlights key areas where CNN-based methods have demonstrated effectiveness and suggests directions for future research.

The remainder of this article is structured as follows: Sect. 2 provides an insightful overview of Arabic handwriting, delving into its unique characteristics. Moving forward, Sect. 3 offers an extensive literature review, focusing on CNN-based methods employed in the realm of AHR. Section 4 takes a statistical approach, analyzing various aspects of the reviewed studies and their contributions. In Sect. 5, we discuss open challenges within this domain and potential future directions for research. Finally, Sect. 7 concludes the paper.

2 Arabic Handwriting Recognition

The Arabic language hosts vast archives, yet a substantial portion remains untouched by digitization. Extracting information from these original papers proves to be a delicate and time-consuming task, impeding the effective utilization of the wealth of data within these documents. A potential solution lies in scanning these archives into image format and subsequently converting them into a machine-readable format—a process known as AHR. While AHR is commonly associated with historical document recognition, its applications extend to various beneficial uses, as summarized in Fig. 1. The complex nature of the Arabic script introduces intricacies into AHR, resulting in limited satisfactory tools compared to non-cursive languages and printed text. The influence of the Arabic script extends to languages like Farsi, Urdu, and Jawi [2], all based on the Arabic script and incorporating additional characters and diacritics. Despite adaptations, these scripts share fundamental similarities with the Arabic script, making the characteristics and challenges discussed in

this section pertinent not only to Arabic but also to numerous languages sharing similar features.

2.1 Characteristics of Arabic script

The Arabic script is cursive, typically written from right to left, and comprises 28 characters, with 22 of them being fully cursive and the remaining six are semicursive (ا, د, ذ, ر, ز, و). Entirely cursive characters are linked to previous and next characters, while semi-cursive ones are only linked to the previous character. The characters' shapes change depending on their position inside the word (isolated, start, middle, or end) which represents a real issue for recognition algorithms. Figure 2 shows the 28 characters in their isolated form. A word could be composed of one or more sub-words. Sub-words are called Pieces of Arabic Word (PAW). A PAW could consist of a single character or a set of linked characters. Another distinctive feature of the Arabic script is the presence of diacritical marks, which can be points or signs. Diacritical points are crucial in determining the identity of characters, serving to provide additional information or modify the pronunciation of the character. Out of the 28 characters, 16 of them contain diacritical points positioned above or below the main body of the character. Certain characters have similar main body shapes, with the only distinction being the placement and number of diacritical points they possess. On the other hand, diacritical signs determine how the characters are pronounced, playing a role similar to vowels (A, O, E, I) in English. Pronunciation is crucial, as it can lead to a change in the meaning of the word. Arabic readers typically do not rely on diacritical signs to determine character pronunciation, as contextual information is usually sufficient. Consequently, most Arabic texts do not incorporate diacritical signs. Another prevalent feature in Arabic script is vertical ligatures. A vertical ligature is a set of vertically connected characters (usually two or three) as illustrated in Fig. 3 (d). Vertical ligatures are present either in handwriting or some printed fonts.

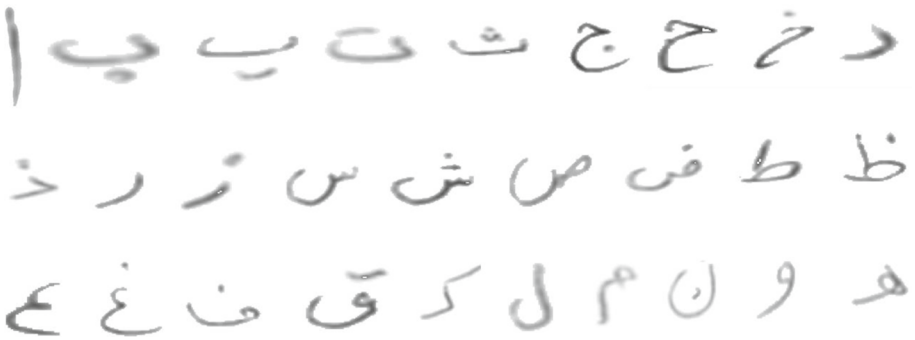

Fig. 2 The shapes of the Arabic handwritten characters in their isolated form, sourced from the IFHCDB database [10]

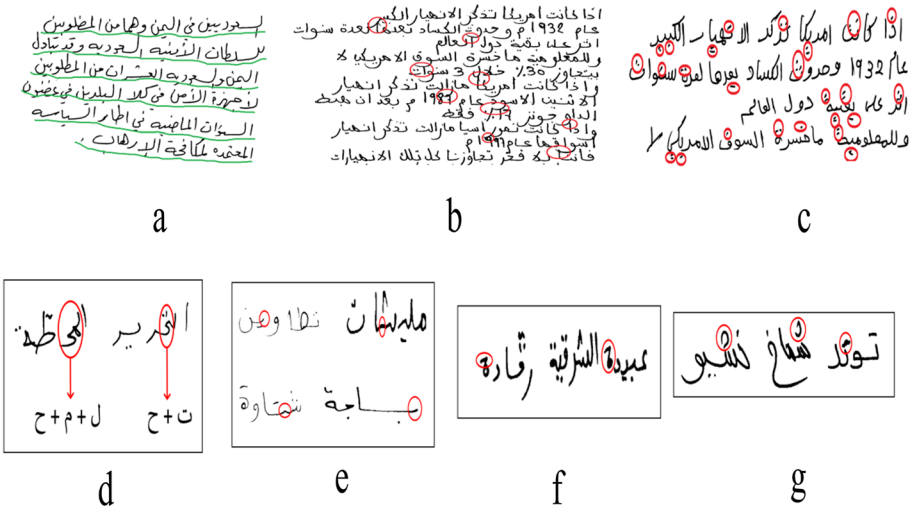

Fig. 3 Challenges in AHR: **a** inclination in text, **b** Overlaps and touches, **c** Diacritical marks, **d** vertical ligatures, **e** small/hidden ascenders, **f** touches between text and diacritical components, **g** similarity between text components and diacritical components. Source [11]

2.2 Challenges in Arabic script

Arabic handwriting poses significant challenges in recognition compared to non-cursive languages and printed text [3], introducing complexities across various phases of the recognition process. The difficulties inherent in Arabic handwriting contribute to the limited success and lack of robust tools for AHR. Key challenges include:

- **Cursive Style and Calligraphy**: The cursive nature and diverse calligraphy of Arabic writing make segmentation and recognition inherently challenging.
- **Variability in Writing Styles**: The absence of standardized guidelines results in varied writing styles among different individuals, adding a layer of complexity to recognition tasks.
- **Positional Shape Changes**: Arabic characters undergo shape variations based on their positions within a word—whether at the beginning, middle, final, or isolated—introducing complexity to the recognition process.
- **Diacritical Marks**: Diacritics present a significant challenge, with certain characters sharing identical diacritic counts, leading to reduced variability (Fig. 3 (c)). Issues such as intersecting marks (Fig. 3 (f)), variations in size, and misinterpretation of marks as isolated characters (Fig. 3 (g)) or noise further complicate recognition.
- **Overlaps and Touches**: Instances of overlaps and touches between lines, words, or characters introduce challenges in effectively extracting individual elements (Fig. 3 (b)).

- **Text Inclination**: Inclined text leads to overlaps and touches between lines, words, or characters, impeding effective extraction (Fig. 3 (a)).
- **Ascending and Descending Characters**: The abundance of ascending and descending characters in Arabic contributes to instances of touching and overlapping within the text (Fig. 3 (b)).
- **Vertical Ligatures**: The presence of vertical ligatures adds complexity to segmentation and recognition operations Fig. 3 (d)).

2.3 AHR phases

While Optical Character Recognition (OCR) broadly deals with recognizing printed/handwriting text, AHR is a specialized subfield that specifically addresses the challenges presented by handwritten Arabic script. The architectural framework of AHR systems closely aligns with OCR systems, encompassing five fundamental phases: Pre-processing, Segmentation, Feature Extraction, Recognition, and Post-processing. Figure 4 visually illustrates the general pipeline employed in an AHR system.

Each phase involves a series of steps; for instance, the segmentation phase entails the extraction of document objects, including text and figures, as well as line, word, and character segmentation. It's worth noting that certain steps or phases may be omitted or combined based on the adopted approach. For instance, segmentation-free approaches recognize words as single units, leading to the exclusion of the character segmentation step [12]. The interrelation among these phases is crucial, as the outcomes of each phase significantly impact subsequent phases and, consequently, the overall performance of the system.

In the **preprocessing** phase, the image undergoes various treatments to prepare it for the subsequent stages of the recognition process. Numerous procedures are applied, including noise suppression, binarization, normalization, baseline detection, line straightening, contour tracing, and more.

Segmentation involves the extraction of document components or sub-components, including paragraphs, lines, words, PAWs, and characters. Character segmentation is widely acknowledged as one of the most challenging issues in AHR.

Feature extraction involves the detection and encoding of the most representative features from the output of the previous phase. This step generates a vector of characteristics that encapsulates the identity of the queried sample.

Classification can take on a supervised or unsupervised approach. In supervised classification, the goal is to label an unknown item with the appropriate class from the dataset based on its features. To achieve this, the classifier must undergo training on previously annotated data.

The post-processing phase employs lexical (dictionary), syntactic (grammar), or semantic (meaning) methods to enhance the recognition rate and reduce classification errors.

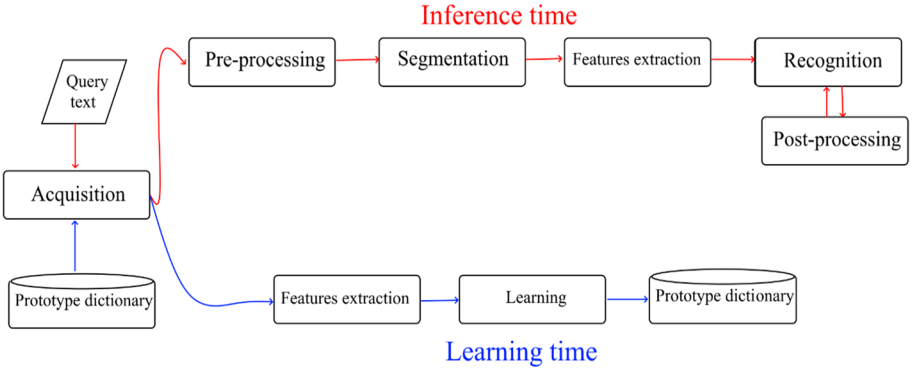

Fig. 4 The general pipeline of an AHR system

2.4 Benchmark handwritten Arabic databases.

New DL techniques are data-hungry, necessitating large datasets for optimal performance. The intricate nature of the Arabic script, coupled with the scarcity of datasets, has positioned this research area in its early stages compared to other domains. While there have been some efforts to establish databases in this context, significant work is still required to create standard benchmark databases that can adequately support the requirements of emerging DL techniques. Table 1 provides a summary of the most commonly used databases for Arabic handwriting.

3 CNN-based approaches for AHR

Since 2016, CNNs have gained prominence in AHR, outperforming traditional techniques, various machine learning approaches, and other deep learning models. Notably, CNNs have predominantly been applied to AHCR, with comparatively fewer applications in Arabic Handwritten Digit Recognition (AHDR) and AHWR. Certain aspects such as pre-processing, layout analysis, and segmentation have not received widespread attention with CNNs. Generally, there is a discernible scarcity of works that offer holistic solutions for AHR, contributing to the lack of robust tools in this domain. This section provides an extensive literature review, focusing on the applications of CNNs across major AHR tasks, encompassing layout analysis, line and character segmentation, as well as AHCR, AHDR, and AHWR.

3.1 Segmentation

Segmentation, the process of partitioning an image into discernible entities like words, subwords, paragraphs, characters, or lines, is relatively less challenging in printed documents than in handwritten ones [26] (refer to sec 2.2). Following the extraction of document objects like figures and tables, the initial segmentation step involves line segmentation, followed by word and character segmentation. While most works have extensively utilized CNN-based approaches for line segmentation, there is a noticeable scarcity of research

Table 1 Benchmark Arabic handwriting databases

Database	Size and content	Context	Availability
IFN/ENIT [13]	26.459 handwritten Tunisian town/village names	Word recognition Character segmentation	Yes, for non-commercial work
IFHCDB [10]	52.380 handwritten isolated characters and 17.740 digits	Character recognition Digit recognition	Yes, for non-commercial work
AHCD [14]	16.800 characters	Character recognition	Yes
AIA9k [15]	8737 characters	Character recognition	Yes
HACDB [16]	6600 characters	Character recognition	Yes
KHATT [17]	2000 random paragraphs and 2000 fixed paragraphs	Line segmentation Word segmentation Character segmentation Entire AHR	Yes
HAPD [18]	125×10 manuscript pages	Lines segmentation	Yes
IESK-arDB [19]	4000 words and 6000 characters	Word recognition Character recognition	Yes
CENPARMI [20]	29.498 sub-words, 15.175 digits, 2.499 amount	PAW recognition Digit recognition	Yes
CEDAR	100 text pages, each consisting of 150–200 words	Entire AHR	No
Arabic handwritten 1.0	5000 manuscript pages	Entire AHR	No
Hijja [21]	47,434 characters written by 591 participants	Character recognition	Yes
HMBD [22]	54,115 of the handwritten Arabic characters	Character recognition	Yes
AHDB[23]	10,080 forms representing 96-word classes + texts written by a hundred participant	Word recognition	Yes
ADBase [24]	70,000 digits written by 700 writers	Digit recognition	Yes
HODA [25]	102,352 digits extracted from 12,000 registration forms	Digit recognition	Yes

focusing on other critical stages, such as layout, word, character, and PAW segmentation. This section offers a review of CNN-based methods applied for segmentation issues, with a summarized overview available in Table 2.

In the realm of segmentation, researchers have extensively explored the Fully Convolutional Network (FCN) [27], another CNN extension, and its derivative models, including dilated FCN, U-Net [28], RU-Net [29], and ARU-Net [30]. Initially proposed by Long et al. [27] for semantic segmentation tasks, FCN stands out by exclusively incorporating Convolutional Layers (CLs) while excluding Fully Connected layers (FCs), thereby reducing complexity due to a vast number of trainable parameters in FCs. FCN, however, faced resolution challenges, prompting Noh et al. [31] to introduce a deconvolutional network to refine output resolution. The FCN architecture comprises an encoder for downsampling input, akin to a conventional CNN, and a decoder for upsampling input and making predictions. U-Net, an improved version of FCN, symmetrically combines local and global information through skip connections between downsampling and upsampling paths, effectively addressing the vanishing gradient issue [28]. RU-Net extends the U-Net by incorporating a residual structure (R), while ARU-Net integrates both the attention mechanism (A) and residual connections (R). In R2U-Net [32], Residual connections (R) and Recurrent CLs (2) are integrated, and AR2U-Net [33] further includes the Attention mechanism (A). Notably, in the domain of AHR, FCN and its derivatives have found substantial application, particularly in the context of line segmentation [29, 30].

In the work of Barakat and El-Sana [34], FCN was employed for the layout analysis of complex historical Arabic documents, with the goal of extracting side text and main text from non-binarized documents. The FCN's encoder part comprises a succession of 5 blocks akin to VGG-16 architecture. In the decoder part, two architectures, FCN-32 and FCN-8, were tested and FCN-8 was selected for its superior segmentation precision. The experiments were conducted on a dataset consisting of 38 documents from 7 different historical Arabic books. Notably, the proposed FCN architecture achieved a 95% F-measure for the main text and 80% for the side text.

the same FCN architecture was explored in the work of Barakat et al.[35] to extract text lines from complex historical documents. Given FCN's reliance on labeled data, the authors introduced line mask labeling to connect characters belonging to the same line, thereby generating labeled data for FCN training. Post-processing was employed to rectify predicted disconnected lines. The experiments involved 30 documents from the Islamic Heritage Project dataset (IHP), yielding an 80% F-Measure.

For the same goal, Mechi et al. [36] utilized an adaptive U-Net architecture. The authors introduced an optimization to the contracting path in U-Net by modifying the filter succession from (64, 64, 128, 256, 512, 1024) to (32, 64, 128, 512). This optimization effectively reduced the number of parameters in the network, addressing issues of overfitting and loss values during training. The adaptive U-Net achieved a training loss of 0.058, significantly outperforming the classical U-Net with a loss of 0.127. The experiments were conducted on READ, cBAD, DIVA-HisDB, and a local dataset, yielding F-measures ranging from 76 to 79% on the utilized datasets.

The adaptive U-Net was integrated with traditional analysis techniques in another study conducted by the same authors [37], to boost performance. The U-Net was utilized to identify the primary area covering the text core known as X-height. Subsequently, a modified Run Length Smoothing Algorithm (RLSA) in combination with topological structural analysis was applied to detect entire lines. The X-heights identified by U-Net and the extracted foreground pixels by RLSA were then merged. Finally, ascenders and descenders were identified to pinpoint the ultimate text lines. The hybrid approach

Table 2 A review of CNN-based methods for Arabic handwriting segmentation

Authors	Task	Model	Database	Results
Barakat and El-Sana [34]	Layout analysis	FCN	38 document images from 7 different historical Arabic books	95% F-measure for the main text 80% F-measure for side text
Barakat et al. [35]	Line segmentation	FCN	IHP	80% F-Measure
Mechi et al. [36]	Lines segmentation	Adaptative U-Net	cBAD	79% F-Measure
			DIVA-HisDB	76% F-Measure
			ANT (Arabic)	76% F-Measure
Mechi et al. [37]	Lines segmentation	U-Net-RLSA	RASM	88.42% Accuracy
			ANT (Arabic)	95.08% Accuracy
Neche et al. [29]	Word segmentation	CNN-BLSTM-CTC	KHATT	80.1% Accuracy
	Line segmentation	RU-Net		96.1% Accuracy
Gruning et al. [30]	Lines segmentation	ARU-Net + clustering	cBAD	92.2% Accuracy
Gader and Echi [33]	Line segmentation	AR2U-Net	BADAM	93.7% Precision
Elkhayati and Elkettani [39]	Lines segmentation	CNN + handcrafted features + multi-agent system	KHATT	99.6% F-Measure
			HAPD	99.8% F-Measure
Elkhayati et al. [40]	Graphemes segmentation	CNN + handcrafted features + mathematical morphology	IFN/ENIT	97.35% Accuracy
	Diacritics extraction	CNN + handcrafted features	IFN/ENIT	99,74% Accuracy
Barakat et al. [41]	Line segmentation	Siamese CNN	VML-AHTE	93.95% Accuracy 93.78% F-Measure
			ICFHR 2010	97.72% Accuracy 97.63% F-Measure

Works are sorted by task to facilitate comparison

demonstrated significant improvement over the U-Net alone, attaining an accuracy of 88.42% on the RASM dataset and 95.08% accuracy on the ANT (Arabic) dataset.

Neche et al. [29] utilized RU-Net, an extension of U-Net incorporating residual modules, for the task of text line segmentation. In their approach, text images were treated as a three-class problem, classifying pixels into background, paragraph, and text line x-height. RU-Net generated a map of three classes, which was then subjected to post-processing to extract baselines. The effectiveness of this method was evident, achieving an accuracy of 96.1% on the KHATT database. It is important to note that KHATT, unlike the previously discussed databases, is not a historical database, and it presents relatively less difficulty in text line segmentation.

Expanding on this achievement, the authors extended their work to address word segmentation by employing a model that integrates CNN and BLSTM-CTC (Bidirectional Long Short-Term Memory followed by Connectionist Temporal Classification). Initially, representative features were extracted from text line images using a conventional CNN architecture. Subsequently, these features were inputted into a BLSTM-CTC network, which is used to map between transcription and text line images. The implemented approach demonstrated a promising accuracy of 80.1%, prompting the authors to assert the feasibility of addressing both line and word segmentation without depending on lexicons or language-specific resources.

Following a similar approach, Grüning et al. [30] applied ARU-Net to extract text lines from historical documents. They trained ARU-Net from scratch using a limited set of labeled images with data augmentation. The model classified pixels into three classes: baseline, separator, and others. Subsequently, the authors employed bottom-up clustering and various image processing techniques to extract baselines. Notably, their method demonstrated effectiveness in handling challenging cases, such as curved lines, arbitrarily oriented text lines, and complex layouts. The approach achieved a 92.2% F-measure on cBAD, showcasing significant improvement compared to the adaptive U-Net in [36].

An AR2U-Net-based approach was proposed by Gader and Echi [33] for extracting lines from diverse document types, ranging from historical to contemporary documents. They adapted AR2U-Net for pixel-wise classification, distinguishing between foreground and background pixels, and subsequently identifying text lines based on the concept of text-line masks. The method achieved a precision of 93.2% on the BADAM dataset [38]. Authors reported that this dataset poses intricate challenges, including curved, skewed, and arbitrarily oriented text lines, making it more complex than KHATT and cBAD. The success of AR2U-Net in addressing such complexities was evident in these findings.

Diverging from FCN and its derivatives, Elkhayati and Elkettani [39] introduced a directed CNN architecture embedded within a multi-agent system designed for text line segmentation. In this approach, a group of interactive agents traverses the document from the right to the left edge in the interline space, with each agent assigned to separate two adjacent lines. The CNN model functions as an assistant, supporting agents in decision-making, especially in challenging situations. To train the CNN model, a locally constructed database was utilized, specifically tailored to capture intricate scenarios encountered by the agents. The model's attention was directed using externally handcrafted representative features, integrated into the flattened layer with a certain frequency. The approach demonstrated exceptional performance, achieving a 99.6% F1-measure on KHATT and 99.8% on HAPD, surpassing a version of the approach without CNNs in both databases. The results highlight the substantial value added by the CNN model in this particular context.

This same directed model has also been shown to be effective for grapheme segmentation in another study conducted by Elkhayati et al. [40]. This model was integrated with

two mathematical morphology operations—dilation and erosion—for enhanced performance. Employed across various tasks such as over-trace extraction, diacritics suppression, and graphemes extraction, the directed CNN consistently outperformed the standard CNN architecture, showcasing the effectiveness of the implemented attention mechanism. The directed model exhibited notable improvements, achieving a 1% higher accuracy in diacritics extraction compared to a standard CNN, along with a 2% enhancement in both overtrace detection and grapheme extraction accuracies.

Barakat et al. [41] introduced an unsupervised method for text line extraction using a Siamese CNN architecture. In this approach, two identical branches of the network, resembling the AlexNet architecture, are employed to predict the similarity between patches from two given documents. Each branch takes an image patch as input and produces a feature representation of the patch. The outputs from the two branches are concatenated and fed through three FCs. Following training, CNN is utilized to extract features from patches, which are then thresholded to form blob lines. The Energy Minimization algorithm is subsequently applied to extract pixel labels. The method was evaluated on three datasets: VML-AHTE, ICDAR 2017, and ICFHR 2010, achieving F-measures ranging from 93 to 97%.

3.2 Character recognition

Various CNN-based approaches have been explored for AHCR in the literature, including simple CNN models, deep models, hybrid models, Transfer Learning (TL), and ensemble learning. Researchers have also delved into refinements like hyperparameter fine-tuning, structural modifications, and the integration of techniques such as dropout and data augmentation. This section provides a focused exploration of CNN-based strategies for AHCR, extending to related languages like Urdu, Farsi, and Jawi. Table 3 succinctly captures the essence of these methods, offering insights into their applications and performances.

3.2.1 Simple custom-designed architectures

Numerous studies in the literature have opted for the utilization of simple custom-designed architectures, with many of them based on the architecture of LeNet [42], often incorporating adjustments in FCs and CLs. These proposed architectures are typically shallow, offering the advantage of reduced complexity compared to more recent and complex designs.

One of the earliest and simplest architectures was proposed by El-sawy et al. [14], featuring 2 CLs and 2 FCs. The architecture achieved an accuracy of 94.9% on the AHCD database. This achievement is particularly noteworthy as it represents one of the pioneering works that initially applied CNNs for AHCR. Notably, the AHCD database, proposed in this work, has become one of the most widely used databases in the field of AHCR.

In an attempt to boost performance, Najdat et al. [43] extended the architecture of El-sawy et al. [14] by adding more layers (4 CLs and 3 FCs) and introducing batch normalization and dropout layers. Several hyperparameters were fine-tuned, resulting in an improved accuracy of 97.2%, surpassing the 94.9% accuracy reported in [14].

Aljarrah et al. [44] experimented with a model featuring more CLs (6 CLs) and applied data augmentation techniques to the database. With data augmentation, the model achieved a recognition rate of 97.7%, a slight improvement over the 97.2% accuracy without data

Table 3 Review of CNN-based models for AHCR

Authors	Model	Database	Accuracy (%)
El-sawy et al. [14]	Custom-designed CNN	AHCD	94.9
Najdat et al. [43]	Custom-designed CNN	AHCD	97.2
Aljerrah et al. [44]	Custom-designed CNN	AHCD	97.2
	Custom-designed CNN + data augmentation		97.7
Altwaijry and Al-Turaiki [21]	Custom-designed CNN	AHCD	97
		Hijja	88
Younis [45]	Custom-designed CNN	AHCD	94.7
		AIA9K	94.8
Boufenar et al. [46]	Custom-designed CNN	OIHACDB	97.32
Elkhayati and Elkettani [11]	Directed custom-designed CNN	IFHCDB	98.4
		AHCD	98.7
		AIA9K	96.1
		HACDB	95.4
Wagaa et al. [48]	Custom-designed CNN + Data augmentation	AHCD	98.48
		Hijja	91.24
Almansari et al.[49]	Custom-designed CNN	AHCD	95.27
Alrehali et al. [50]	Custom-designed CNN	Dataset 1 (2240 samples representing 56 classes)	74.29
		Dataset 2 (1000 samples representing 10 classes)	84.67
		Dataset 3 (2000 samples representing 10 classes)	88.20
Bouchriha et al. [60]	Custom-designed CNN	Hijja	95
Mudhsh and Almodfer [51]	VGG-based architecture	HACDB	97.32
De Sousa [52]	VGG-16 + Ensemble learning	AHCD	98.42
Alyahya et al. [53]	ResNet-18 + Ensemble learning	AHCD	98.30
Taani and Ahmed [54]	ResNet-152	AHCD	99.55
		AIA9K	99.05
Balaha et al. [22]	Deep custom-designed CNN	HMBD	95.4
		AIA9K	99

Table 3 (continued)

Authors	Model	Database	Accuracy (%)
Elleuch et al. [55]	CNN-SVM	HACDB	93.4
	CNN-SVM with dropout	HACDB	94.2
Ali et al. [61]	CNN-SVM with dropout	HACDB	99.85
		AHCD	99.71
Shams et al. [56]	CNN-SVM	AHCD + 840 additional images	95.07
Alrobah and Albahli [57]	CNN-SVM	Hijja	96.3
	CNN-XGBoost		95.7
	CNN-FC		89.7
Elkhayati and Elkettani [58]	CNN-computational geometry algorithms	IFHCDB	97.4
Husnain et al. [59]	CNN + geometrical features	Local database	96.4
Naz et al.[62]	CNN-MDLSTM	UPTI	98.12
Safarzadeh and Jafarzadeh [63]	CNN-BLSTM-CTC	Characters from different sources	95
Boufenar et al. [64]	AlexNet + TL (training from scratch)	OIHACDB-40	100
	AlexNet + TL (partial fine-tuning)		82.38
	AlexNet + TL (global fine-tuning)		98.86
	AlexNet + TL (training from scratch)	AHCD	99.98
	AlexNet + TL (partial fine-tuning)		82.53
	AlexNet + TL (global fine-tuning)		96.78
Balaha et al. [65]	CNN-5	HMBD	91.96
	VGG16 + TL		92.88
	VGG19 + TL		90.91
	MobileNetV2 + TL		78.91
Arif and Poruran [66]	OCR-AlexNet + TL	IFHCDB	96.3
	OCR-GoogleNet + TL	IFHCDB	94.7

augmentation. However, the increased number of data and layers did not show a significant difference compared to the model in [43] with fewer layers and no data augmentation.

The effect of adding more FCs was investigated by Altwaijry and Al-Turaiki [21], utilizing 4 FCs instead of 2 or 3. To mitigate potential overfitting due to the higher number of parameters, authors applied 80% dropout after FCs. The resulting CNN model reported 97% accuracy on AHCD and 88% on Hijja (a newly constructed database). However, this model did not notably enhance results on AHCD despite the increased number of layers.

Younis [45] investigated the impact of incorporating batch normalization, ReLU, and dropout layers after each CL. The model achieved accuracies of 94.8% and 94.7% on AIA9K and AHCD datasets, respectively, which were comparable to the results reported by El-sawy et al. [14]. These findings demonstrated no positive effect of dropout after CLs.

In another study, Boufenar and Batouch [46] conducted an in-depth investigation into the effect of dropout after CLs as well as FCs. The authors tested different keep probability parameters for dropout layers to find the optimal values. The best results were achieved with a dropout value of 0.2 in CLs and 0.5 in FCs. Additionally, the study explored the impact of the number of neurons within FCs. Experiments on the OIHACDB database [47], revealed that the best-reported accuracy (97.32%) was obtained using 1500 neurons in FCs with the discussed dropout values. This underscores the importance of fine-tuning dropout parameters for optimal performance.

In their work, Elkhayati and Elkettani [11] introduced a dropout-like method to decrease the number of parameters in FCs without compromising performance. This approach involved a two-block strategy at the FC level to reduce connectivity. The first block, consisting of 75% of the neurons, processed relevant features (25% of the features), while the second block, with 25% of the neurons, handled irrelevant features (75% of the features). Feature selection was performed at the flattened layer using a method termed virtual max-pooling. This strategic modification resulted in a 45% reduction in parameters compared to the standard architecture, with enhanced accuracies ranging from 0.1% to 3.3% across four databases: IFHCDB, AHCD, AIA9K, and HACDB.

Hyperparameter fine-tuning is a critical aspect of training CNNs, as various parameters significantly impact model effectiveness. Parameters such as learning rate, weight decay, dropout value, and filter size all play a role in shaping the model's performance, and even small changes can lead to substantial differences in results. In a study conducted by Wagaa et al. [48], experiments were carried out to assess the effectiveness of different optimization and data augmentation techniques, as well as the impact of noise injection on a custom-designed CNN. The performance of optimizers varied widely, ranging from 60 to 99%, with the Nadam optimizer yielding the best results. Rotation and shifting techniques in data augmentation achieved the highest reported accuracies, reaching 91.24% on Hijja and 98.48% on AHCD. The study highlighted the significant influence of optimization and data augmentation techniques on performance. The results also demonstrated that noise injection negatively impacts model performance, leading to a substantial decrease in accuracy for all injected noise types.

In the study conducted by Almansari and Hashim [49], various hyperparameters were tested on a simple CNN architecture to achieve optimal performance. The authors experimented with different batch sizes, training epochs, filter sizes, and dropout values. The best testing accuracy obtained was 95.27% on AHCD, achieved with a batch size of 32, a filter size of 5×5, a dropout value set to 0.2, and 25 training epochs. The findings suggest that the use of a 5×5 filter is better than the commonly used 3×3 filter, opening new avenues for research on this issue. Additionally, it contrasts the idea that more training epochs bring better results, as 25 epochs were enough for the model to converge, and further training led to a decrease in performance.

In a distinct study, Alrehali et al. [50] experimented with a CNN model featuring 3 CLs and 3 FCs for AHCR from historical documents. The model was trained and tested on three locally constructed databases with varying character shapes, achieving accuracies ranging between 74.29% and 88.20%. While the obtained results might seem comparatively lower than those of other models, it is essential to note that AHCR from historical text is inherently more challenging compared to standard text.

3.2.2 Deep architectures

Against simple custom-designed models, several studies have attempted to employ deeper CNN architectures, often based on well-known models in the literature such as VGGNet, AlexNet, and ResNet. Authors typically either apply the architecture as is to the problem, only adjusting the output layer to match the number of classes for AHCR, or make modifications within the architecture by adding or removing certain layers. Deep models show promise for enhanced performance; however, the deeper the network, the more data it requires for better generalization, prompting researchers to adopt certain solutions such as regularization and data augmentation techniques.

Aiming for improved performance with reduced complexity, Mushsh and Almodfer [51] modified the VGGNet architecture. VGGNet, known for its higher complexity as one of the deepest networks, underwent adjustments by reducing the size of the input image, the number of filters in CLs, and the number of neurons in FCs. These modifications resulted in a significant reduction in the number of parameters, decreasing from 138 to 2 million. To address overfitting, the authors incorporated dropout and data augmentation techniques, underscoring their importance in enhancing the model's performance. The adapted model achieved 99.57% on ADBase for digits and 97.32% on HACDB for characters.

VGGNet was investigated within an ensemble learning framework by De Sousa [52]. The ensemble consisted of four models: i) VGG12, a VGG16-based architecture, ii) VGG12 with data augmentation, iii) REGU, an experimentally adapted new architecture, and iv) REGU with data augmentation. The predictions from these architectures were combined to form an averaged ensemble classifier, ENS4. Although no significant differences were observed when comparing the four architectures individually, models with data augmentation showed a slight enhancement compared to those without. ENS4 demonstrated superior performance, achieving remarkable accuracies of 99.74% for MADBase digits and 98.42% on AHCD for characters, underscoring the effectiveness of the ensemble learning strategy.

The concept of ensemble learning was explored by Alyahya et al. [53]. The authors examined two architectures: a standard ResNet-18 and a modified version of ResNet-18 with dropout layers after CLs. The ensemble model was tested with both 1 FC and 2 FCs and data augmentation techniques were applied to the AHCD dataset. The highest accuracy, 98.3%, was achieved by the standard ResNet-18, while the ensemble model reached 98% accuracy with both 1 and 2 FCs. Interestingly, in contrast to the findings of [52], ensemble learning did not contribute to performance improvement in this context. Regarding dropout, the results align with [45], confirming that adding dropout after all CLs does not contribute to the improvement of the performance.

Taani and Ahmed [54] introduced modifications to the standard ResNet-152 architecture, adjusting layers, parameters, activation functions, and regularization techniques. The model demonstrated impressive accuracy, achieving 99.8% on MadBase for digits, 99.05% on AIA9K, and 99.55% on AHCD for characters. The model outperformed the ResNet-18

used by [53], underscoring the impact of depth on results. It's important to note that this improvement may be attributed to the specific adjustments made by the authors, as they did not make a direct comparison between the standard architecture and their modified version.

The impact of depth in CNNs was also explored by Balaha et al. [22]. The authors conducted a comparison between two custom-designed CNN architectures: HMB1, characterized by higher complexity (more layers and parameters), and HMB2, a shallower model. The two architectures underwent experimentation with various optimizers and weight initializers. HMB1 outperformed HMB2, underscoring the superiority of deep models. The study also highlighted the effectiveness of data augmentation in improving testing accuracy and reducing overfitting. Reported accuracies reached 100% on CMATER (for digits), 99% on AIA9K (characters), and 95.4% on HMBD, a newly proposed dataset.

3.2.3 Hybrid architectures

Hybrid methods have proven to be effective in the context of AHCR, often combining CNN with ML techniques, mostly Support Vector Machine (SVM). CNN serves as a feature extractor, while other ML techniques are employed for classification. The idea behind these hybrids is that CNN is excellent at feature extraction, surpassing handcrafted methods, while SVM, for instance, proves to be robust in classification, outperforming FCs.

In Elleuch et al.'s model [55], SVM is employed after FCs, utilizing CLs for feature extraction and a combination of FC and SVM with an RBF kernel for classification. The input flows through the feature extraction part, FC, and then SVM. The study explored the impact of dropout in FCs, experimentally demonstrating its significance, with the Dropout-based model achieving higher accuracies, with 94.17% on HACDB for characters and 92.95% on IFN/ENIT for words.

Shams et al. [56] adopted a similar strategy, integrating SVM after the final FC of the CNN model to enhance results. The authors simplified the problem to 13 classes by clustering similar characters through the K-means algorithm, easing the classification challenge. The model achieved a correct classification rate of 95.07% on AHCD.

Alrobah and Albahli [57] explored the hybridization of CNN with two ML algorithms, SVM, and XGBoost. Through experimentation with various optimizers and weight initializers to optimize hyperparameters, the study demonstrated that SVM and XGBoost, outperform traditional FCs as classifiers. Moreover, the CNN-SVM hybrid model yielded slightly better results than CNN-XGBoost. Overall, the authors reported an 8% improvement in accuracy compared to [21] on the Hijja dataset, utilizing the same CNN architecture. This finding underscores the superiority of SVM over traditional FCs.

In the study of Elkhayati and Elkettani [58], CNN was hybridized with two computational geometry algorithms, namely the Relative Neighborhood Graph and Gabriel's Graph. These algorithms function as a filtering layer during inference, effectively reducing potential classes for a query item. The output of the filtering layer is intersected with the results of the softmax layer at the end of the CNN to determine the predicted class. The filtering layer relies on the feature vector extracted by the CLs and intervenes only during inference. This hybridization resulted in a 3% improvement in accuracy for a simple custom-designed CNN on the IFHCDB database. These findings suggest the potential use of other computational geometry algorithms in this context, expanding beyond the commonly used K-Nearest Neighbors (KNN) algorithm in literature.

Hybrid models in AHCR also explore the combination of handcrafted extraction features with machine-learned features, as proposed by Husnain et al. [59], through the

incorporation of geometrical feature extraction before the CNN model. In this approach, geometrical features are initially extracted and then combined with pixel-based data before being input into the CNN model. The authors developed a database for Urdu handwritten characters and numerals, reducing the number of classes from 28 to 10 by merging characters with similar shapes. The model achieved a character recognition accuracy of 96.04% and a numeral recognition accuracy of 98.3%. The authors did not conduct a comparative experiment to assess the impact of geometrical features on the results, leaving uncertainty about whether this approach had a positive effect compared to a standalone CNN.

Naz et al. [62] explored the hybridization of two neural networks, combining CNN and the Multidimensional Long Short-Term Memory (MDLSTM). CNN is tasked with extracting low-level translational invariant features from the input data. Subsequently, these extracted features are forwarded to the MDLSTM, which further captures high-level features and handles the classification task. The proposed hybrid approach achieved a recognition accuracy of 98.12% on the UPTI dataset [67].

3.2.4 Transfer learning

Transfer Learning (TL) offers an effective response to data scarcity [68, 69] by leveraging pre-trained models from related tasks, reducing training time and the need to build models from scratch. Recently, the number of works using TL in AHR has started to grow. Researchers in this field have utilized pre-trained CNN models, often trained on extensive datasets like ImageNet, as a starting point, then fine-tuned the models on Arabic handwriting databases.

Boufenar et al. [64] attempted to study the effect of TL in the context of AHCR. For this purpose, they investigated three learning strategies: (i) Learning from scratch, (ii) CNN as a feature extractor, and (iii) fine-tuning CNN. The strategies were tested on an AlexNet-based architecture consisting of 8 layers. Surprisingly, the "training from scratch" approach demonstrated superior performance compared to the TL strategies, achieving accuracy rates near 100% under certain conditions. The study results raise questions about the effectiveness of TL in AHCR, particularly as there was a significant difference favoring training from scratch over the two TL approaches.

On the other hand, in the study conducted by Balaha et al. [65], TL demonstrated effectiveness when applied within three well-known architectures: VGG16, VGG19, and MobileNetV2. VGG16 outperformed the other two models, achieving an accuracy of 92.88% on the HMBD database. Furthermore, this TL-enhanced model surpassed the performance of 14 different custom-designed CNN architectures without TL. However, the authors acknowledged that this improvement came at the expense of complexity, as VGG16 proved to be more intricate than the custom-designed architectures.

Arif and Poruran [66] employed TL in two architectures referred to as OCR-AlexNet and OCR-GoogleNet, adapted from the original architectures of AlexNet and GoogleNet. In the case of OCR-AlexNet, the initial layer weights of AlexNet were retained, while adjustments were made to the final three layers to tailor them to the new task. In the OCR-GoogleNet model, only four inception modules from GoogleNet were utilized, and fine-tuning was performed on the last two layers. The OCR-AlexNet and OCR-GoogleNet achieved average accuracies of 96.3% and 94.7%, respectively, on the IFHCDB database. Importantly, these results were found to be statistically significant when compared to the performance of other traditional methods that were examined in their experiments (Table 3).

3.3 Digit recognition

Recognizing digits poses a slightly lesser challenge compared to characters, primarily due to the reduced number of classes (10 instead of 28), diminished inter-class variability, and the inherent complexities associated with characters (refer to Sect. 2). This disparity accounts for the higher recognition accuracies achieved in AHDR compared to AHCR. Notably, methodologies applied for AHCR are also extended to AHDR, encompassing simple CNNs, deep CNNs, hybrid models, TL, and ensemble learning. For a succinct overview of CNN-based methods in AHDR, refer to Table 4.

Ashiquzzaman and Tuchar [70] conducted a comparative analysis between handcrafted feature extraction and machine-learned features. In their study, two models were evaluated: a traditional MLP with hand-crafted features and a CNN with the same MLP structure for classification. The assessment was carried out on the CMATERDB 3.3.1 database, revealing that CNN outperformed the MLP, achieving a 4.1% increase in accuracy. These findings confirm the efficacy of CNNs in feature extraction compared to traditional handcrafted methods.

The same CNN architecture [70] underwent further improvement in another study by Ashiquzzaman et al. [71]. In this enhancement, the authors incorporated data augmentation techniques, introduced the dropout effect, and changed the activation function. Addressing the limitation of the ReLU function, which can halt learning in regions where x is negative due to its derivative being 0, the authors opted for the Exponential Linear Unit (ELU) function. ELU's derivative never becomes 0 across its curve, facilitating smooth learning. The incorporation of dropout and data augmentation proved to be impactful, resulting in a notable improvement in the model's performance, which achieved an accuracy of 99.4% on the CMATERDB 3.3.1 database. Notably, the method without data augmentation maintained a 97.4% accuracy, consistent with their results in [70], affirming that changes in the activation function had no impact, while data augmentation exerted the most significant influence on the results.

Ahmed et al. [72] introduced several adjustments to the same architecture [70], such as switching to the Adam optimizer from AdaDelta and increasing dropout by 15%. These modifications contributed to an enhanced performance of the model, achieving 98.91% accuracy on an augmented version of CMATERdb 3.3.1. However, it remains unclear whether the improvement is solely attributed to the adjustments or if the data augmentation applied to the database also played a significant role, as the authors did not conduct experiments on the original CMATERdb 3.3.1. The authors also proposed a second architecture with 1 more FC, which achieved even better accuracy (99.76%) than the modified one. The findings suggest that incorporating more FCs can lead to better performance, as previously suggested by studies such as [43] and [21]. However, it's crucial to consider that FCs contribute significantly to the overall increase of parameters due to their full connectivity. Therefore, while additional FCs may enhance performance, they also introduce increased complexity to the model. Balancing the benefits of improved accuracy with the associated complexity is an important consideration in designing effective CNN architectures.

The findings of Latif et al. [73] align with similar conclusions, emphasizing the benefit of incorporating more FCs in CNN architecture. The study compared the performance of a CNN with one FC against a configuration with two FCs in the context of recognizing multilingual handwritten digits. The evaluation covered five distinct databases, each representing a different language: Eastern Arabic, Farsi, Urdu,

Table 4 A review of CNN-based models for Arabic handwritten digits recognition

Authors	Model	Database	Accuracy
Ashiquzzaman and Tochar [70]	Custom-designed CNN	CMATERdb 3.3.1	97.40
Ashiquzzaman et al. [71]	Custom-designed CNN + data augmentation	CMATERdb 3.3.1 (augmented)	99.40%
Ahmad et al. [72]	Custom-designed CNN	CMATERdb 3.3.1 (augmented)	99.76%
Balaha et al. [22]	Custom-designed CNN	CMATERdb	100%
Latif et al. [73]	Custom-designed CNN with 1 FC	MADBase	99.30%
	Custom-designed CNN with 2 FCs		99.21%
Nanehkaran et al. [78]	Custom-designed CNN	HODA	99.45
Musdhsh et al. [51]	VGGNet-based architecture	ADBase	99.6%
Taani and Ahmed [54]	ResNet-152	MadBase	99.8%
Ahranjany et al. [74]	LeNet-5 + Ensemble learning	IFHCDB	99.01%
Nanekharan et al. [75]	Ensemble learning with Resnet18-VGG16-Xception	HODA	99.87%
		IFHCDB	98.42
		CENPARMI	97.13
De Sousa [52]	VGG16 + Ensemble learning	MADBase	99.74%
Gupta and Bag [76]	Custom-designed CNN	CMATERdb 3.3.1	96.53%
	CNN-SVM	Multilingual database	96.23%
	CNN-softmax		96.04%
Parseh et al. [77]	CNN-SVM	HODA	99.56
Husnain et al. [59]	CNN + geometrical features	Local database	98.3%
Ghofrani and Toroghi [79]	CapsuleNet-EM routing	HODA	99.87%
Safarzadeh and Jafarzadeh [63]	CNN-BLSTM-CTC	HODA	99.37%
		AHDBase	99.43%
Alkhawaldah [80]	VGG16-BLSTM	AHDBase	98.92%
		MAHDBase	97.94%
Alani [81]	RBM-CNN	CMATERDB 3.3.1	98.59

Devanagari, and Western Arabic. Except for Western Arabic, all other languages featured Arabic digits with slight variations in some numerals. As anticipated, the two-layered architecture exhibited a slight performance advantage over the single-layer CNN, achieving an average accuracy of 99.32%.

Ahranjany et al. [74] employed an ensemble learning strategy by combining the predictions of five LeNet-5 architectures. The fusion of predictions was executed through various methods, including average, maximum, minimum, and product, with extensive testing to determine the most effective approach. Before training, two strategies, boosting and rejection, were implemented. In the boosting strategy, misclassified and rejected samples were repeated in the training set, while the rejection strategy involved removing 10% of "hard-to-recognize" samples from the training set. The proposed approach achieved its highest recognition rate of 99.17% on the IFHCDB normal dataset and a 99.98% recognition rate after rejecting "hard-to-recognize" samples.

Ensemble learning was explored differently by Nanekharan et al. [75], where three different deep models—VGG16, ResNet18, and Xception—were utilized, and their predictions were fed into a meta-classifier following the principles of bagging ensemble learning. The ensemble approach outperformed the individual models, achieving an average recognition accuracy of 97.65% on HODA, 98.42% on IFHCDB, and 97.13% on CENPARMI datasets. However, a notable drawback of this method is its computational expense due to the utilization of multiple deep models. While the results align with those obtained in [74], favoring ensemble models over individual ones, the approach in [74] outperformed this one even with smaller models, possibly due to the boosting strategies employed and the exploration of different ensemble methods.

Gupta and Bag [76] conducted a comprehensive comparison involving ML classifiers (SVM, MLP, and Random Forest), deep CNNs with TL (AlexNet, VGG-16, ResNet-18, and DenseNet-121), a custom-designed CNN with softmax, and a hybrid CNN-SVM model. Their evaluation was performed on a multilingual handwritten numerals dataset, consolidating shapes from eight script types, including Arabic, into a unified database with 10 classes. Remarkably, the hybrid CNN-SVM model surpassed all counterparts with an accuracy of 96.23%, closely followed by CNN-softmax at 96.04%. Notably, ML methods exhibited unsatisfactory results, confirming that features learned through CNNs outperform handcrafted ones. This conclusion is supported by the comparison, where SVM with handcrafted features reported less favorable results compared to SVM with CNN. The unexpectedly superior performance of a shallow CNN architecture compared to deeper ones raised intriguing questions about the efficacy of deep models as well as TL. Despite these considerations, the work overall affirmed the feasibility of multilingual digit recognition through CNN-based approaches.

The effectiveness of the CNN-SVM hybrid approach was further corroborated by the results of Parseh et al. [77]. In this study, SVM was exclusively utilized during the inference phase, while training involved a conventional custom-designed CNN with FCs. During testing, the FCs were substituted with SVM. The comparison revealed a slight performance improvement using the SVM classifier over FCs, achieving an accuracy of 99.56% on the HODA dataset.

In contrast to the findings in [76], the work of Nanehkaran et al. [78] revealed that traditional SVM with handcrafted features could achieve comparable results to CNN. The study conducted a comparative analysis of three ML classifiers (SVM, KNN, and ANN) and two DL classifiers (CNN and AutoEncoder) for Arabic/Persian digits. While CNN emerged as the top performer with a 99.45% accuracy on the HODA dataset, SVM closely followed with 99.3% accuracy. These results underscore the ongoing superiority of CNN

in computer vision tasks but highlight also the effectiveness of SVM in classification. CNN exhibited a shorter inference time compared to traditional ML techniques, which is an additional advantage for its account. The AutoEncoder had the fastest inference time; however, it was less effective compared to other algorithms.

Unlike these findings, Ghofrani and Toroghi [79] showcased the effectiveness of AutoEncoders in AHDR, specifically CapsuleNet. This marked the first application of CapsuleNet in this domain, with the authors justifying their choice based on the equivariance property inherent in CapsuleNets, providing increased resistance against transformations. CapsuleNet comprises two main parts: the encoder and decoder. Each part consists of three layers, with CL constituting the initial layer of the encoder, while the decoder exclusively features FCs. Impressively, the model achieved a remarkable 99.87% accuracy on the HODA dataset, surpassing the performance of the CNN model proposed in [78]. Furthermore, the authors highlighted that the proposed model exhibited robustness against noise.

The combination of two types of ANNs has demonstrated effectiveness in the work of Safarzadeh and Jafarzadeh [63]. The authors utilized a CNN-BLSTM-CTC hybrid model for AHCR, AHDR, and AHWR. This same combination was used by [29] for word segmentation (refer to Sect. 3.1.) but with differences in the architecture. The model begins with ten stacked convolutional layers generating 512 feature vectors of size 16 (Table 4). This feature matrix (512×16) is then fed into BLSTM as a sequence, then to CTC. The architecture achieved notable accuracies on various datasets: 99.43% on AHDBase for digits, 99.37% on HODA for digits, 95% for characters from various sources, and up to 93% accuracy on IFN/ENIT for words.

In contrast to the common approach of integrating CNN and BLSTM sequentially, Alkhawaldah [80] introduced an innovative method by incorporating LSTM layers between CLs and FCs within VGG-16 and GoogleNet architectures. The authors also leveraged a TL strategy, attempting to combine the strengths of TL in data-limited fields, CNNs for extracting local features, and LSTMs for capturing long-term dependencies. The proposed hybrid model yielded satisfactory results, achieving 98.92% accuracy on AHDBase and 97.94% on MAHDBase.

In the work by Alani [81], a novel approach was introduced by combining CNN with an unsupervised ANN called the Restricted Boltzmann Machine (RBM). The goal was to enhance the model's capacity to extract relevant features, leveraging RBM's known capability to extract highly useful features from raw data. The hybrid method involved a two-stage training process. Initially, RBM was trained on the dataset to extract relevant features. Subsequently, the extracted features, represented in the form of a 28×28 matrix, were utilized to train a small, custom-designed CNN. This combined approach reached a 98.59% accuracy on the CMATERDB 3.3.1 dataset, demonstrating superior performance compared to both the standalone CNN used in a previous study [70] and the combination of the RBM-SVM classifier. This outcome underscores the effectiveness of leveraging diverse neural network architectures for enhanced feature extraction.

3.4 Word recognition

AHR approaches can be broadly categorized into two main classes: segmentation-based approaches and segmentation-free approaches. In segmentation-based approaches, also known as analytic approaches, the word undergoes segmentation into characters before proceeding to the recognition phase. Conversely, in segmentation-free approaches, also referred to as holistic approaches, the entire word is recognized as a single unit [12].

Holistic approaches are commonly employed in scenarios with a limited vocabulary. However, for tasks involving an unlimited vocabulary, analytic approaches become necessary. This section delves into works that utilize holistic approaches, recognizing words as single entities. In the literature, numerous authors have successfully employed CNN-based methods for Arabic Handwritten Word Recognition (AHWR), often building upon the methodologies discussed in the character and digit recognition sections. Table 5 provides a comprehensive summary of CNN-based methods specifically tailored for AHWR.

In Mustapha and Khalafallah 's work [82], a standard CNN architecture with 4 CLs and 4 FCs proved effective for AHWR, achieving an impressive 99.14% accuracy on the SUST-ARG [121] database, consisting of Arabic names.

Lamsaf et al. [83] explored a more streamlined CNN architecture with 3 CLs and 1 FC. Before recognition, essential pre-processing steps, including binarization, normalization, and rotation, were implemented. A sizable dataset of 40,320 Arabic handwritten words was expanded from AHDB [23] through data augmentation. The model demonstrated a recognition rate of 96.76% on the newly augmented dataset. We propose potential improvement through experimental adjustments or the adoption of a deeper architecture. This suggestion is based on the observation that data augmentation was applied, indicating a correlation between network depth and data quantity. Deep models have consistently proven effective in handling large volumes of data.

El-melegy et al. [84] confirmed this hypothesis by achieving similar results using the original AHDB, which is 10 times smaller than the augmented database used in [83]. They conducted extensive experiments, particularly focusing on hyperparameters, specifically the L2 regularization parameter, to address overfitting. Moreover, they extended the training epochs from 50 to 500. These adjustments led to a notable improvement with an accuracy of 96.58%, surpassing the 82% obtained with default hyperparameters. We suggest that incorporating data augmentation into these findings could further enhance the results.

Deeper models have found practical applications in AHWR. For instance, Ghanim et al. [85] conducted experiments with six different deep CNN models, including AlexNet, VGG16, GoogleNet, ResNet50, ResNext, and DenseNet, along with SVM, on the IFN/ENIT database. The classification proceeded by two stages: matching and ranking. Matching encompassed representing IFN/ENIT as an intricate search tree-like model with interrelated clusters, each containing database classes sharing similar regional and geometric features. The ranking approach aimed to streamline complexity by minimizing the number of database classes involved in the final classification. Surprisingly, AlexNet achieved the highest accuracy of 95.6% outperforming other deeper models. This challenges the notion that deeper networks are better. However, it is crucial to consider the database size, as extensive models require more data to perform well.

Almodfer et al. [86] subjected AlexNet to structural and hyperparameter modifications, specifically investigating the impact of dropout and activation functions. The systematic application of dropout at various positions in four distinct AlexNet architectures revealed consistent and significant improvements in classification accuracy with incremental dropout values. Regarding activation functions, a comparison between ReLU and TanH showed no significant advantage of one over the other. The results showcased enhanced recognition accuracies, with the network featuring the maximum dropout layers achieving the best outcomes: 92.13% with ReLU and 92.55% with TanH on the IFN/ENIT database. Interestingly, it's noted that the standard AlexNet used in [85] yielded better results than the version in this study, indicating that the modifications made to the AlexNet architecture harmed performance. However, the comparison between the two works is not objective

Table 5 Review of CNN-based models for AHWR

Authors	Model	Database	Results
Mustapha and Khalafallah [82]	Custom-designed CNN	SUST-ARG names database	99.14% Accuracy
Lamsaf et al. [83]	Custom-designed CNN	Augmented AHDB	96.76% Accuracy
El-melegy et al. [84]	Custom-designed CNN with no data augmentation	AHDB	96.8% Accuracy
	Custom-designed CNN with data augmentation		97.8% Accuracy
Ghanim et al. [85]	AlexNet	IFN/ENIT	95.6% Accuracy
Almodfer et al. [86]	AlexNet with ReLU	IFN/ENIT	92.13% Accuracy
	AlexNet with Tanh		92.55% Accuracy
Ali et al. [61]	CNN-SVM	IFN/ENIT	98.58% Accuracy
		AHDB [23]	99.00% Accuracy
Elleuch et al. [55]	CNN-SVM	IFN/ENIT	92.95% Accuracy
Ali et al. [87]	Custom-designed CNN	AHDB	97.7%
	CNN-SVM-KNN		98.4%
Amrouch et al. [88]	CNN-HMM	IFN/ENIT	89.23% Accuracy
Maalaj et al. [89]	CNN-BLSTM-CTC	IFN/ENIT	92.21% Accuracy
Safarzadeh and Jafarzadeh [63]	CNN-BLSTM-CTC	IFN/ENIT	93% Accuracy
Khemiri et al. [90]	DBN-CNN	IFN/ENIT	95.20% Accuracy
Khosravi and Chalechale[91]	CNN	Iranshahr	89.04% Accuracy
	AutoEncoder-CNN		91.09% Accuracy
Awni et al. [92]	ResNet18 with Adam	IFN/ENIT	7.21% Error rate
	ResNet18 with SGD		9.41% Error rate
	ResNet18 with RMSProp		11.57% Error rate
	Ensemble learning with ResNet18		6.63% Error rate
Almodfer et al. [94]	Ensemble learning with custom-designed CNN	IFN/ENIT	8.5% Error rate
Poznanski and Wolf [95]	Multi-branch VGG-16	IFN/ENIT	97.07% Accuracy

due to differences in methodology. The work in [85] involved two stages before the classification phase, while in this work, classification was performed directly.

Similar to AHCR and AHDR, Hybrid models have taken place in AHWR. In the study of Ali and Mallaiah [61], CNN and SVM were combined, following [55] [56] (see Sect. 3.2.3). SVM was chosen over other ML classifiers experimentally, providing the best accuracy in comparisons. For feature extraction, two deep neural network models, Spatial Pyramid Pooling (SPPnet) and Network in Network (NIN), were integrated Simultaneously. Cross-entropy and max-margin minimum classification error (M3CE) were merged to enhance results. Results demonstrated the effectiveness of dropout as well as the capacity of M3CE to develop cross-entropy. Overall, the proposed model yielded favorable results, achieving 99.96% on AHDB for words, 96.5% on IFN/ENIT for words, 97.42% on AHCD for characters, and 97.85% on HACDB for characters.

Ali and Suresha [87] replaced the FCs of CNN with a fusion of KNN and SVM for classification. While CNN achieved the best individual performance on AHDB and IFN/ENIT datasets, the proposed fusion model surpassed the performance of the three individual algorithms. The fusion model achieved 98.7% training accuracy and 98.4% testing accuracy on AHDB.

Amrouch et al. [88] replaced the second FC of the LeNet-5 architecture with the Hidden Markov Model (HMM) classifier. The output from the flattened layer is directed to the HMM for classification, leveraging HMM's aptitude for capturing the dynamics of Arabic handwriting, as emphasized by the authors. The proposed model achieved an accuracy of 88.95% on IFN/ENIT, marking a 1.02% improvement compared to HMM with handcrafted features. In their comparison, the authors emphasized the impact of feature extraction using CNN over hand-crafted methods. Notably, a comparative analysis between the standard CNN with FCs and CNN-HMM was omitted, leaving unexplored the rationale behind substituting FCs with HMM.

RNNs or LSTMs have been combined with CNN in the context of AHR by many authors [63, 67, 80], aiming to exploit their ability to learn long-term dependencies in temporal or sequenced data. LSTMs are particularly useful in the context of words compared to characters and digits, as words contain a sequence of characters and PAWs. In this context, Maalaj and Kherallah [89] adopted a CNN-BLSTM model in the same manner as [63]. With the incorporation of data augmentation, the model achieved a notable accuracy of 92.21% on IFN/ENIT. While the authors did not conduct a direct comparison between individual CNN and BLSTM models to justify their hybridization, they reported that their method outperformed literature approaches based on both handcrafted and machine-learned features.

Khemiri et al. [90] hybridized CNN with another type of sequenced data learning ANN, specifically Dynamic Bayesian Network (DBN), this time utilizing the ANN for feature extraction. DBN, renowned for its adeptness in extracting representative features, was leveraged to combine the advantages of handcrafted features extracted by DBN and machine-learned features of CNN. The features extracted by DBN were subsequently convolved with the input data using CNN. The approach yielded a 95.20% accuracy on the IFN/ENIT database, which is better than several individual CNN and DBN architectures. In the literature, most works compare handcrafted features with CNN-learned features. This study illustrates the effectiveness of combining these features, particularly with sophisticated techniques like DBN. Notably, satisfactory results can also be achieved with traditional methods, as emphasized in [59] in the context of AHCR.

The same approach was adopted by Khosravi and Chalechale [91], this time employing an AutoEncoder for feature extraction in place of DBN. The encoded image in AutoEncoder is then passed through the CNN layers for further feature extraction and classification. Before this stage, the authors decomposed words into PAWs, extracted diacritical marks, and created a dictionary of main PAWs. They implemented a PAW fusion algorithm to mitigate confusion between PAWs of different words. The AutoEncoder-CNN model underwent evaluation on the Iranshahr dataset, comprising 17,000 images of handwritten names of 503 Iranian cities. The AutoEncoder-CNN achieved recognition accuracy of 91.09%, surpassing a standalone CNN by 2%, thereby demonstrating the effectiveness of the added AutoEncoder as a feature extractor.

Ensemble learning methods have also found application in AHWR. In a study by Awni et al. [92], three ResNet18 models were trained with distinct optimizers (RMSprop, Adam, and SGD). The final prediction was determined through model averaging, aggregating predictions from all trained models. The authors incorporated the cyclical learning rate [93], a learning range finder method, for dynamic learning rates instead of a fixed learning rate. The ensemble achieved superior results, exhibiting a 6.63% error rate on IFN/ENIT, surpassing the performance of individual models. Notably, ResNet with the Adam optimizer, showcasing a 7.21% error rate, outperformed other optimization techniques.

Averaging ensemble learning was also explored by Almodfer et al. [94]. In their study, three similar CNN architectures were trained separately, employing identical hyperparameters but varying image sizes (100 px, 200 px, 300 px). While CNN trained with 300 px images exhibited a notable 3% accuracy improvement, highlighting the significance of high-resolution data, it incurred longer training and inference times. The ensemble model outperformed the three individual CNNs, achieving a 2.4% lower word error rate on IFN/ENIT compared to the best-performing individual model. This aligns with the findings of Awni et al. [92].

In a concept similar to ensemble learning, Poznanski and Wolf [95] introduced a multi-branch VGG16 architecture. Unlike using one FC layer for the entire attribute architecture, the authors employed separate FCs for each attribute. This design capitalizes on CLs' ability to recognize letter appearances independently of their position, while FCs capture spatial information. Dividing the single FC layer into parts, each corresponding to a spatial section, enables specialized learning, ultimately enhancing accuracy. The model, tested on diverse databases, including IFN/ENIT for Arabic, delivered results ranging from 94.09% to 99.29% accuracy depending on the test scenario.

3.5 Highlights and discussion

CNN models have proven effective in AHR, with a primary focus on tasks like digit, character, and word recognition. For segmentation challenges, FCN, a combination of CNN and AutoEncoder, is commonly employed for its proficiency in handling the challenges of this task [11, 29, 30, 33–37]. Results vary across recognition tasks, showcasing the inherent difficulty of each problem. The accuracy levels are spanning from 96 to 100% for AHDR, 82% to 100% for AHCR, and 88% to 99% for AHWR.

The observed diversity in results within the literature can be attributed to various factors such as architecture, data quality, and training hyperparameters. Hyperparameter selection poses a significant challenge in training CNNs, typically performed experimentally, demanding considerable time and effort. Studies have shown that even subtle adjustments in hyperparameters can significantly impact results [72, 84]. This highlights the necessity for more comprehensive studies comparing hyperparameter choices specific to AHR.

The literature has explored various CNN-based methods, including simple CNN models, deep models, hybrid models, ensemble learning, and TL. Hybrid models have gained widespread adoption, demonstrating effectiveness in this domain. Authors have employed two primary hybridization approaches: i) Combining CNN with another ML or DL technique for classification and ii) Combining CNN with another handcrafted method or DL model for feature extraction. In the first approach, SVM [55–57, 61, 76, 77, 87] and LSTM [62, 63, 80, 89] have proven to be effective classifiers, surpassing the capacity of FCs. In the second approach, deep networks like DBN [81], RBM [90], and AutoEncoder [76, 79, 91] have exhibited effectiveness in extracting representative features, thereby enhancing the performance of standalone CNNs.

Another employed method is Ensemble learning [52, 53, 74, 75, 92, 94, 95], although it is less prevalent than hybrid models. In this approach, authors independently train several CNN models, employing either the same architecture or different models with distinctions in architecture, hyperparameters, or input images. The predictions from these models are then aggregated using techniques such as averaging and maximum. Experimental results consistently demonstrate that ensemble models outperform standalone models. However, it is crucial to consider the heightened complexity of these models during both training and inference times, as they demand more time and computational power than standalone models.

Deeper models [22, 51, 73, 85] have not consistently demonstrated a significant performance increase compared to shallower ones, as might be anticipated. Surprisingly, in certain cases, shallower models have outperformed their deeper counterparts. This paradox can be attributed to the pervasive challenge of data scarcity in AHR. Deep models inherently demand abundant data for robust generalization, presenting a substantial obstacle in this domain. As a pragmatic response to limited datasets, researchers have opted for shallow models, explaining the prevalent reliance on shallower CNNs in AHR [14, 43, 44, 49, 82].

The scarcity of data in AHR is evident in the literature's focus on specific databases, notably AHCD for characters, IFN/ENIT for words and CEMARTDB, HODA, and ADBase for digits. Unlike databases such as ImageNet boasting millions of samples, AHR databases typically consist of only a few thousand samples. Consequently, the current pool of AHR databases may fall short of catering to the demands of complex DL architectures.

A potential solution to this challenge is the effective use of data augmentation techniques [23, 48, 71], which have demonstrated their capacity to enhance the overall performance of models. It is noteworthy that the depth of the network correlates with the demand for more extensive data to achieve better generalization. Surprisingly, some shallower models in the literature have been applied to data-augmented datasets (large datasets), while deeper models have been applied to smaller datasets. Reassessing these practices could potentially enhance the performance of models by aligning network depth with appropriate data sizes.

A second solution to address generalization concerns in this data-limited area is the integration of regularization techniques, notably dropout [48, 49, 86]. While exceptions

exist, most experiments in the literature underscore the significance of dropout for achieving better model generalization.

A third solution in this context lies in the use of TL [64–66, 76]. TL was less widely adopted in AHR compared to other techniques, and its benefits have been limited due to the use of models pre-trained on ImageNet, a dataset distant from the specifics of AHR. To harness the potential advantages of TL, a recommendation is made to explore pre-trained models from datasets similar to those utilized in the AHR domain. This approach could provide a more promising avenue for improving performance, as demonstrated in various other application fields.

4 Survey Analysis

In this survey, a total of 130 papers were collected, out of which 62 papers were analyzed. Each of the selected papers focused on investigating CNN and its variants for AHR purposes. The majority of the selected papers were published between 2017 and 2022, reflecting the rapid growth of CNN research in the AHR field during that period. Figure 5 presents the distribution of papers per year, highlighting the increasing trend. The papers were sourced from various publishers, including IEEE, Springer, Elsevier, and others. Figure 6 illustrates that IEEE and Springer accounted for the majority of the papers. Different types of papers were considered, including journal articles, conference papers, and book chapters. As shown in Fig. 7, journal articles constituted the majority of the selected papers.

The selected papers covered four main phases of AHR: Segmentation, AHCR, AHDR, and AHWR. Figure 8 demonstrates that AHCR was the most addressed phase. A statistical analysis of the methods used in these 62 papers is provided in Fig. 9. The majority of the works utilized simple custom-designed CNN architectures as discussed in the previous paragraphs. A significant number of papers employed FCN and its variants, primarily for segmentation purposes. The same trend was observed for the hybridization between CNN and SVM. The usage of databases in the selected papers is analyzed in Fig. 10. The most commonly used databases were AHCD and IFN/ENIT, confirming what was mentioned in Sect. 3.5. Additionally, a notable number of works utilized their local datasets.

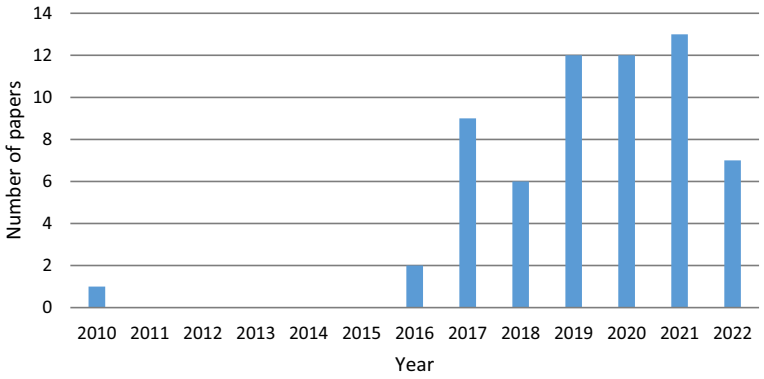

Fig. 5 The distribution of selected papers over the years

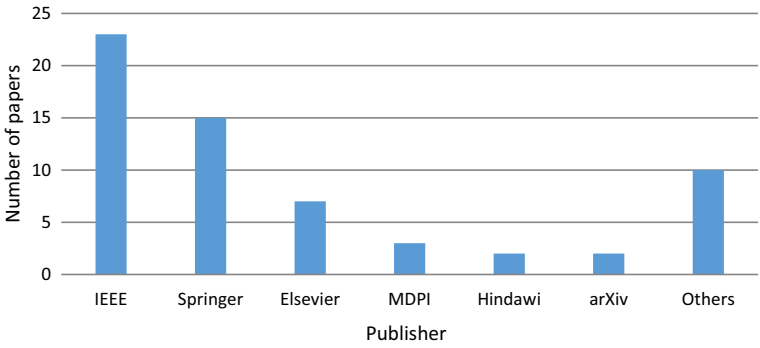

Fig. 6 Sources of the selected papers

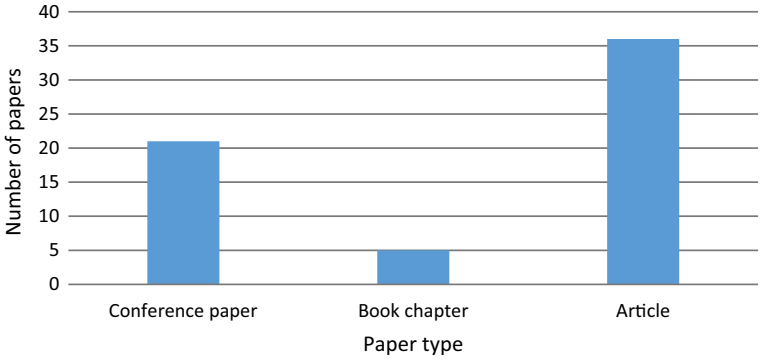

Fig. 7 Types of the selected papers

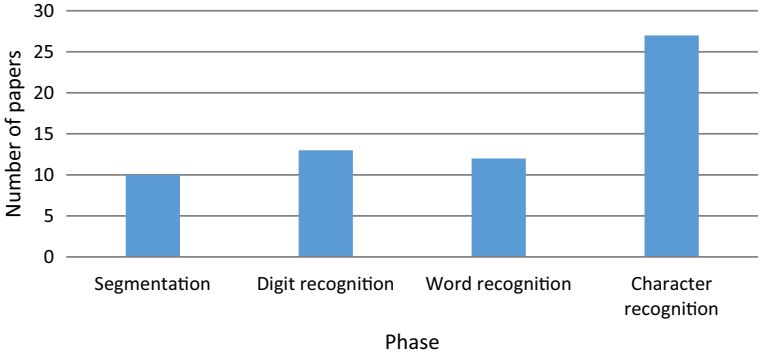

Fig. 8 Distribution of the works across different tasks

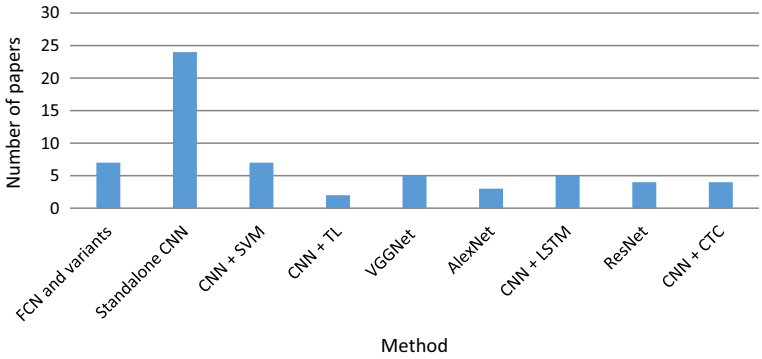

Fig. 9 Number of papers per method

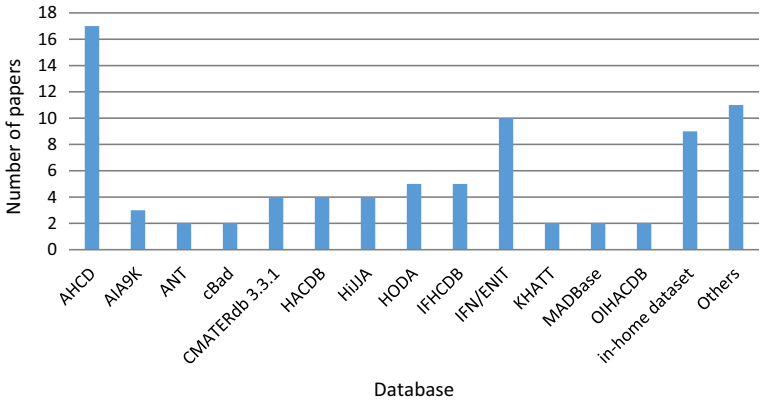

Fig. 10 Number of papers for each database

5 Open problems and future directions

Despite the promising results demonstrated by CNN-based methods in the field of AHR, which have enabled notable progress in specific tasks, several open problems and challenges persist. Some of these include:

- **The lack of specific application datasets:** This is a major issue for researchers, as the size and quality of the dataset used significantly impacts the results obtained. However, the databases and repositories available are often limited and of inadequate quality for some purposes.
- **Variation in handwriting styles**: Arabic is a cursive script, and there is significant variation in the way different individuals write. This variability makes it challenging to design models that can accurately recognize different styles of handwriting.
- **The similarity between characters**: Arabic characters can have similar shapes, especially when written in cursive. This makes it difficult for models to distinguish between different characters and can lead to recognition errors.

- **Preprocessing challenges**: Preprocessing is an important step in AHR, as it helps to enhance the quality of the input images and improve the accuracy of the models. However, preprocessing can be challenging in AHR due to the complex nature of the script.
- **Recognizing handwritten text in historical Arabic papers:** This presents an additional challenge due to the substantial differences between ancient and modern writing styles. The existing datasets used for training handwritten recognition models are primarily based on modern Arabic, rendering them ineffective for accurately recognizing historical Arabic papers.
- **The inconsistency of the Arabic language**, particularly concerning similar stroke characters and dots, can be lost or deleted due to low resolution or preprocessing. This issue is particularly pertinent for characters with diacritical marks, which require more attention and investigation.

Addressing these challenges will require further research and development in the field of AHR, including the development of new techniques for preprocessing, feature extraction, and architecture design, as well as the creation of larger and more diverse labeled datasets. There are several potential directions for the future application of CNNs in AHR. Here are some possibilities:

- **Improving recognition accuracy**: Despite the promising results achieved by CNN-based models in AHR, there is still room for improving their recognition accuracy. Future research could explore ways to enhance the training process by optimizing hyperparameters and developing more effective loss functions.
- **Hybrid state-of-the-art models** have demonstrated significant improvements in AHR compared to standard models. These hybrid approaches involve combining various ML and DL methods to achieve better performance. Therefore, future research should prioritize the design and development of novel hybrid approaches in AHR.
- **Incorporating additional features**: CNNs have proven effective in recognizing the spatial features of Arabic handwriting. However, incorporating additional features, such as temporal information, could enhance overall accuracy. For instance, the combination of RNNs with CNNs has demonstrated robustness. Further exploration in this avenue could be beneficial to capture the temporal dynamics of Arabic handwriting.
- **Multi-language recognition**: Most CNN-based models for AHR are designed to recognize Arabic text only. However, the demand for recognition of multiple languages is increasing. Future research could focus on developing CNN-based models that can recognize multiple languages, including Arabic, English, and other languages.
- **Handling noise and variability**: Arabic handwriting is characterized by variability and noise. CNN-based models can be sensitive to these factors, leading to decreased recognition accuracy. Future research could investigate ways to handle the noise and variability in Arabic handwriting to improve recognition accuracy.
- **Recognition of diacritics**: Recognizing diacritics is important for accurate AHR, but it is a challenging task due to the small size of the marks and their placement above or below letters.
- **Real-time recognition**: CNN-based models are effective in offline AHR. However, the demand for real-time recognition is increasing, especially in applications such as mobile devices. Future research could focus on developing CNN-based models that can achieve real-time AHR.
- **TL and domain adaptation**: TL and domain adaptation are effective in improving recognition accuracy in various computer vision tasks. Future research could explore

the potential of TL and domain adaptation in improving the recognition accuracy of CNN-based models for AHR. By exploring the potential of TL in AHR, researchers can assess its effectiveness in improving recognition accuracy and efficiency.

- **Interpretability**: CNNs are often considered "black boxes" as it can be difficult to understand how they reach their decisions. Future research could focus on developing more interpretable CNN-based models for AHR to provide insights into how the models make their recognition decisions.

Overall, these are some of the open problems and future directions that could help improve AHR. Addressing these challenges could lead to more accurate and robust AHR systems that can be used in a variety of applications, from document digitization to automated handwriting analysis.

6 Conclusion

In recent times, CNNs have dominated AHR, showing promising results and offering hope for overcoming challenges. In this article, we comprehensively reviewed CNN-based methods applied to AHR, shedding light on their strengths and weaknesses. We also explored the AHR landscape, outlining characteristics, challenges, and future directions. The article aimed to guide the research community and assist researchers, both experienced and novice, in navigating AHR complexities and discovering effective techniques.

The reviewed studies highlighted the effectiveness of various methods, each with its distinct advantages and weaknesses. Ensemble learning and hybrid models, notably, have shown significant effectiveness over standalone models. Shallow standalone models were more widely adopted than deep ones in the literature, reflecting a response to limited data availability in this domain. Additionally, data augmentation and dropout techniques demonstrated their potential to enhance model generalization. TL, a promising technique, has not been extensively explored, suggesting the need for further investigation and improved practices that could yield better results.

Despite the advancements made, several challenges in AHR persist and need to be addressed. We recommend focusing investigations on two main aspects: addressing Arabic script-related issues and refining model design. Regarding Arabic script challenges, priorities include constructing larger real-world datasets and addressing issues like noise, variability, and skewness in Arabic handwriting. While recognizing isolated characters, words, and digits is valuable for specific applications, other phases, particularly segmentation, are pivotal in AHR. Therefore, more attention is required for these tasks to develop comprehensive and robust tools for AHR, addressing a significant concern in this domain. Concerning model design, and based on the conclusions extracted from this review, we suggest further exploration of using SVM or LSTM networks instead of FCs. Additionally, investigating the use of other Deep ANNs for feature extraction alongside CNN could enhance model performance. The integration of data augmentation and dropout techniques is also recommended for future works to improve model generalization and performance.

With the continuous advancements in DL techniques, the availability of larger and more diverse datasets, and the development of novel network architectures, CNNs are anticipated to maintain a crucial role in enhancing the accuracy and efficiency of AHR systems. These advancements hold significant implications for various applications, including document

analysis, text recognition, and language processing. Ultimately, these developments contribute to the broader field of Arabic language technology, fostering innovation and improving the capabilities of automated systems in handling Arabic handwritten content.

Author contributions M.E. wrote the main manuscript text. I.K. and Y.T. reviewed and formatted the manuscript

Funding No funding was received to assist with the preparation of this manuscript.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Mohd KN, Adnan AHM, Yusof AA, et al (2019) Teaching arabic language to malaysian university students using education technologies based on education 4.0 Principles. SSRN, pp 38–51
2. Balaha HM, Ali HA, Badawy M (2021) Automatic recognition of handwritten Arabic characters: a comprehensive review. Neural Comput & Applic 33:3011–3034. https://doi.org/10.1007/s00521-020-05137-6
3. Amin A (1998) Off-line Arabic character recognition: the state of the art. Pattern Recogn 31:517–530. https://doi.org/10.1016/S0031-3203(97)00084-8
4. Maitra DS, Bhattacharya U, Parui SK (2015) CNN based common approach to handwritten character recognition of multiple scripts. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 1021–1025
5. Shin H, Roth HR, Gao M et al (2016) Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE Trans Med Imaging 35:1285–1298. https://doi.org/10.1109/TMI.2016.2528162
6. Alghyaline S (2022) Arabic optical character recognition: a review. CMES 135:1825–1861. https://doi.org/10.32604/cmes.2022.024555
7. Alrobah N, Albahli S (2022) Arabic handwritten recognition using deep learning: a survey. Arab J Sci Eng 47:9943–9963. https://doi.org/10.1007/s13369-021-06363-3
8. Ahmed R, Dashtipour K, Gogate M et al (2020) Offline arabic handwriting recognition using deep machine learning: a review of recent advances. In: Ren J, Hussain A, Zhao H et al (eds) Advances in brain inspired cognitive systems. Springer, Cham, pp 457–468
9. Nahla Ibrahim Youssef NA-A (2022) A review on arabic handwriting recognition. J Southw Jiaotong University, 57
10. Mozaffari S, Faez K, Faradji F, et al (2006) A comprehensive isolated Farsi/Arabic character database for handwritten Ocr research. In: Proceedings of the 10th international workshop on frontiers in. La Baule, France, pp. 385–389
11. Elkhayati M, Elkettani Y (2022) UnCNN: a new directed CNN model for isolated Arabic handwritten characters recognition. Arab J Sci Eng. https://doi.org/10.1007/s13369-022-06652-5
12. Khorsheed MS (2007) Offline recognition of omnifont Arabic text using the HMM ToolKit (HTK). Pattern Recogn Lett 28:1563–1571. https://doi.org/10.1016/j.patrec.2007.03.014

13. Pechwitz M, Snoussi Maddouri S, Märgner V, et al (2002) IFN/ENIT-database of handwritten Arabic. In: the 7th Colloque International Francophone sur l'Ecrit et le Document , CIFED. Hammamet, Tunis, pp 129–136
14. Elsawy A, Loey M, El-Bakry H (2017) Arabic handwritten characters recognition using convolutional neural network. WSEAS Trans Comput Res 5:11–19
15. Torki M, Hussein ME, Elsallamy A, et al (2014) Window-based descriptors for arabic handwritten alphabet recognition: a comparative study on a novel dataset. arXiv:14113519
16. Lawgali A, Angelova M, Bouridane A (2013) HACDB: Handwritten Arabic characters database for automatic character recognition. In: European Workshop on Visual Information Processing (EUVIP). pp 255–259
17. Mahmoud SA, Ahmad I, Al-Khatib WG et al (2014) KHATT: an open Arabic offline handwritten text database. Pattern Recogn 47:1096–1112. https://doi.org/10.1016/j.patcog.2013.08.009
18. Kumar J, Kang L, Doermann D, Abd-Almageed W (2011) Segmentation of Handwritten Textlines in Presence of Touching Components. In: 2011 International Conference on Document Analysis and Recognition. IEEE, Beijing, China, pp. 109–113
19. Elzobi M, Al-Hamadi A, Al Aghbari Z, Dings L (2013) IESK-ArDB: a database for handwritten Arabic and an optimized topological segmentation approach. IJDAR 16:295–308. https://doi.org/10.1007/s10032-012-0190-z
20. Suen CY, Nadal C, Legault R et al (1992) Computer recognition of unconstrained handwritten numerals. Proc IEEE 80:1162–1180. https://doi.org/10.1109/5.156477
21. Altwaijry N, Al-Turaiki I (2021) Arabic handwriting recognition system using convolutional neural network. Neural Comput & Applic 33:2249–2261. https://doi.org/10.1007/s00521-020-05070-8
22. Balaha HM, Ali HA, Saraya M, Badawy M (2021) A new Arabic handwritten character recognition deep learning system (AHCR-DLS). Neural Comput & Applic 33:6325–6367. https://doi.org/10.1007/s00521-020-05397-2
23. Al-Ma'adeed S, Elliman D, Higgins CA (2002) A data base for Arabic handwritten text recognition research. In: Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition. pp 485–489
24. Abdleazeem S, El-Sherif E (2008) Arabic handwritten digit recognition. IJDAR 11:127–141. https://doi.org/10.1007/s10032-008-0073-5
25. Khosravi H, Kabir E (2007) Introducing a very large dataset of handwritten Farsi digits and a study on their varieties. Patt Recogn Lett 28:1133–1141. https://doi.org/10.1016/j.patrec.2006.12.022
26. Shi Z, Setlur S, Govindaraju V (2009) A steerable directional local profile technique for extraction of handwritten arabic text lines. In: 2009 10th International conference on document analysis and recognition. IEEE, Barcelona, Spain, pp 176–180
27. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR). Pp. 3431–3440
28. Ronneberger O, Fischer P, Brox T (2015) U-Net: convolutional networks for biomedical image segmentation. In: Navab N, Hornegger J, Wells WM, Frangi AF (eds) Medical image computing and computer-assisted intervention – MICCAI 2015. Springer International Publishing, Cham, pp 234–241
29. Neche C, Belaid A, Kacem-Echi A (2019) Arabic handwritten documents segmentation into text-lines and words using deep learning. In: 2019 International conference on document analysis and recognition workshops (ICDARW). pp 19–24
30. Grüning T, Leifert G, Strauß T et al (2019) A two-stage method for text line detection in historical documents. IJDAR 22:285–302. https://doi.org/10.1007/s10032-019-00332-1
31. Noh H, Hong S, Han B (2015) Learning deconvolution network for semantic segmentation. In: 2015 IEEE international conference on computer vision (ICCV). pp. 1520–1528
32. Alom MZ, Hasan M, Yakopcic C, et al (2018) Recurrent residual convolutional neural network based on U-Net (R2U-Net) for medical image segmentation
33. Aïcha Gader TB, Echi AK (2020) Unconstrained handwritten Arabic text-lines segmentation based on AR2U-Net. In: 2020 17th international conference on frontiers in handwriting recognition (ICFHR). pp. 349–354
34. Barakat BK, El-Sana J (2018) Binarization free layout analysis for arabic historical documents using fully convolutional networks. In: 2018 IEEE 2nd International workshop on arabic and derived script analysis and recognition (ASAR). pp. 151–155
35. Barakat B, Droby A, Kassis M, El-Sana J (2021) Text line segmentation for challenging handwritten document images using fully convolutional network
36. Mechi O, Mehri M, Ingold R, Essoukri Ben Amara N (2019) Text line segmentation in historical document images using an adaptive U-net architecture. In: 2019 International conference on document analysis and recognition (ICDAR). pp. 369–374

37. Mechi O, Mehri M, Ingold R, Ben Amara NE (2021) Combining deep and Ad-hoc solutions to localize text lines in ancient arabic document images. In: 2020 25th International conference on pattern recognition (ICPR). pp 7759–7766
38. Kiessling B, Ezra DSB, Miller MT (2019) BADAM: a public dataset for baseline detection in Arabic-script manuscripts. In: Proceedings of the 5th international workshop on historical document imaging and processing. Association for Computing Machinery, New York, NY, USA, pp. 13–18
39. Elkhayati M, Elkettani Y (2021) Arabic handwritten text line segmentation using a multi-agent system and a directed CNN. In: The Fifth International conference on intelligent computing in data sciences. Fez, Morocco
40. Elkhayati M, Elkettani Y, Mourchid M (2022) Segmentation of handwritten Arabic graphemes using a directed convolutional neural network and mathematical morphology operations. Pattern Recogn 122:108288. https://doi.org/10.1016/j.patcog.2021.108288
41. Barakat BK, Droby A, Alasam R, et al (2020) Unsupervised deep learning for text line segmentation
42. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324. https://doi.org/10.1109/5.726791
43. Najadat HM, Alshboul AA, Alabed AF (2019) Arabic handwritten characters recognition using convolutional neural network. In: 2019 10th international conference on information and communication systems (ICICS). IEEE, Irbid, Jordan, pp. 147–151
44. AlJarrah MN, Zyout MM, Duwairi R (2021) Arabic handwritten characters recognition using convolutional neural network. In: 2021 12th International conference on information and communication systems (ICICS). Pp. 182–188
45. Younis-Khaled S (2017) Arabic hand-written character recognition based on deep convolutional neural networks. JJCIT 3:186. https://doi.org/10.5455/jjcit.71-1498142206
46. Boufenar C, Batouche M (2017) Investigation on deep learning for off-line handwritten Arabic Character Recognition using Theano research platform. In: 2017 Intelligent Systems and Computer Vision (ISCV). pp. 1–6
47. Boufenar C, Batouche M, Schoenauer M (2018) An artificial immune system for offline isolated handwritten arabic character recognition. Evol Syst 9:25–41. https://doi.org/10.1007/s12530-016-9169-1
48. Wagaa N, Kallel H, Mellouli N (2022) Improved Arabic alphabet characters classification using convolutional neural networks (CNN). Comput Intell Neurosci 2022:e9965426. https://doi.org/10.1155/2022/9965426
49. Almansari OA, Hashim NNWN (2019) Recognition of isolated handwritten Arabic characters. In: 2019 7th International conference on mechatronics engineering (ICOM). pp 1–5
50. Alrehali B, Alsaedi N, Alahmadi H, Abid N (2020) Historical arabic manuscripts text recognition using convolutional neural network. In: 2020 6th conference on data science and machine learning applications (CDMA). pp 37–42
51. Mudhsh MA, Almodfer R (2017) Arabic handwritten alphanumeric character recognition using very deep neural network. Information 8:105. https://doi.org/10.3390/info8030105
52. Palatnik de Sousa I (2018) Convolutional ensembles for Arabic handwritten character and digit recognition. PeerJ Computer Science 4:e167. https://doi.org/10.7717/peerj-cs.167
53. Alyahya H, Ismail MMB, Al-Salman A (2020) Deep ensemble neural networks for recognizing isolated Arabic handwritten characters. TIPCV 6:68–79. https://doi.org/10.19101/TIPCV.2020.618051
54. Taani A, Ahmad S (2021) Recognition of Arabic handwritten characters using residual neural networks. JJCIT. https://doi.org/10.5455/jjcit.71-1615204606
55. Elleuch M, Maalej R, Kherallah M (2016) A new design based-SVM of the CNN classifier architecture with dropout for offline arabic handwritten recognition. Procedia Computer Science 80:1712–1723. https://doi.org/10.1016/j.procs.2016.05.512
56. Shams M, Elsonbaty AA, ElSawy WZ (2020) Arabic handwritten character recognition based on convolution neural networks and support vector machine. Int J Adv Comput Sci Appl (IJACSA). https://doi.org/10.14569/IJACSA.2020.0110819
57. Alrobah N, Albahli S (2021) A hybrid deep model for recognizing arabic handwritten characters. IEEE Access 9:87058–87069. https://doi.org/10.1109/ACCESS.2021.3087647
58. Elkhayati M, Elkettani Y (2020) Towards directing convolutional neural networks using computational geometry algorithms: application to handwritten arabic character recognition. Adv Sci Technol Eng Syst J 5:137–147. https://doi.org/10.25046/aj050519
59. Husnain M, Saad Missen MM, Mumtaz S et al (2019) Recognition of urdu handwritten characters using convolutional neural network. Appl Sci 9:2758. https://doi.org/10.3390/app9132758

60. Bouchriha L, Zrigui A, Mansouri S et al (2022) Arabic handwritten character recognition based on convolution neural networks. In: Bădică C, Treur J, Benslimane D et al (eds) Advances in computational collective intelligence. Springer, Cham, pp 286–293
61. Ali AAA, Mallaiah S (2022) Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout. J King Saud Univer - Comput Inform Sci 34:3294–3300. https://doi.org/10.1016/j.jksuci.2021.01.012
62. Naz S, Umar AI, Ahmad R et al (2017) Urdu Nastaliq recognition using convolutional–recursive deep learning. Neurocomputing 243:80–87. https://doi.org/10.1016/j.neucom.2017.02.081
63. Safarzadeh VM, Jafarzadeh P (2020) Offline persian handwriting recognition with CNN and RNN-CTC. In: 2020 25th international computer conference, computer society of Iran (CSICC). pp. 1–10
64. Boufenar C, Kerboua A, Batouche M (2018) Investigation on deep learning for off-line handwritten Arabic character recognition. Cogn Syst Res 50:180–195. https://doi.org/10.1016/j.cogsys.2017.11.002
65. Balaha HM, Ali HA, Youssef EK et al (2021) Recognizing arabic handwritten characters using deep learning and genetic algorithms. Multimed Tools Appl 80:32473–32509. https://doi.org/10.1007/s11042-021-11185-4
66. Ma KO, Poruran S (2020) OCR-Nets: variants of pre-trained CNN for Urdu handwritten character recognition via transfer learning. Procedia Comput Sci 171:2294–2301. https://doi.org/10.1016/j.procs.2020.04.248
67. Sabbour N, Shafait F (2013) A segmentation-free approach to Arabic and Urdu OCR. In: Zanibbi R, Coüasnon B (eds) Burlingame. California, USA, p 86580N
68. Chuanqi T, Fuchun S, Tao K, et al (2018) A survey on deep transfer learning. In: Artificial Neural Networks and Machine Learning – ICANN 2018, Věra Kůrková, Prof. Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, Ilias Maglogiannis. Springer, p 207
69. Weiss K, Khoshgoftaar TM, Wang D (2016) A survey of transfer learning. Journal of Big Data 3:9. https://doi.org/10.1186/s40537-016-0043-6
70. Ashiquzzaman A, Tushar AK (2017) Handwritten Arabic numeral recognition using deep learning neural networks. In: 2017 IEEE International conference on imaging, vision & pattern recognition (icIVPR). pp 1–4
71. Ashiquzzaman A, Tushar AK, Rahman A, Mohsin F (2019) An efficient recognition method for handwritten arabic numerals Using CNN with data augmentation and dropout. In: Balas VE, Sharma N, Chakrabarti A (eds) Data management, analytics and innovation. Springer, Singapore, pp 299–309
72. Ahamed P, Kundu S, Khan T et al (2020) Handwritten Arabic numerals recognition using convolutional neural network. J Ambient Intell Human Comput 11:5445–5457. https://doi.org/10.1007/s12652-020-01901-7
73. Latif G, Alghazo J, Alzubaidi L, et al (2018) Deep Convolutional Neural Network for Recognition of Unified Multi-Language Handwritten Numerals. In: 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR). pp. 90–95
74. Ahranjany SS, Razzazi F, Ghassemian MH (2010) A very high accuracy handwritten character recognition system for Farsi/Arabic digits using Convolutional Neural Networks. In: 2010 IEEE fifth international conference on bio-inspired computing: theories and applications (BIC-TA). pp. 1585–1592
75. Nanehkaran YA, Chen J, Salimi S, Zhang D (2021) A pragmatic convolutional bagging ensemble learning for recognition of Farsi handwritten digits. J Supercomput 77:13474–13493. https://doi.org/10.1007/s11227-021-03822-4
76. Gupta D, Bag S (2021) CNN-based multilingual handwritten numeral recognition: a fusion-free approach. Expert Syst Appl 165:113784. https://doi.org/10.1016/j.eswa.2020.113784
77. Parseh M, Rahmanimanesh M, Keshavarzi P (2020) Persian handwritten digit recognition using combination of convolutional neural network and support vector machine methods. IAJIT 17:572–578. https://doi.org/10.34028/iajit/17/4/16
78. Nanehkaran YA, Zhang D, Salimi S et al (2021) Analysis and comparison of machine learning classifiers and deep neural networks techniques for recognition of Farsi handwritten digits. J Supercomput 77:3193–3222. https://doi.org/10.1007/s11227-020-03388-7
79. Ghofrani A, Toroghi RM (2019) Capsule-Based Persian/Arabic Robust Handwritten Digit Recognition Using EM Routing. In: 2019 4th International Conference on Pattern Recognition and Image Analysis (IPRIA). Pp. 168–172
80. Alkhawaldeh RS (2021) Arabic (Indian) digit handwritten recognition using recurrent transfer deep architecture. Soft Comput 25:3131–3141. https://doi.org/10.1007/s00500-020-05368-8

81. Alani AA (2017) Arabic handwritten digit recognition based on restricted boltzmann machine and convolutional neural networks. Information 8:142. https://doi.org/10.3390/info8040142
82. Mustafa ME, Khalafallah M (2020) A deep learning approach for handwritten Arabic names recognition. IJACSA. https://doi.org/10.14569/IJACSA.2020.0110183
83. Lamsaf A, Kerroum MA, Boulaknadel S, Fakhri Y (2022) Recognition of Arabic handwritten words using convolutional neural network. Indonesian J Electr Eng Comput Sci 26:1148–1155. https://doi.org/10.11591/ijeecs.v26.i2.pp1148-1155
84. El-Melegy M, Abdelbaset A, Abdel-Hakim A, El-Sayed G (2019) Recognition of Arabic handwritten literal amounts using deep convolutional neural networks. In: Morales A, Fierrez J, Sánchez JS, Ribeiro B (eds) Pattern Recognition and Image Analysis. Springer, Cham, pp 169–176
85. Ghanim TM, Khalil MI, Abbas HM (2020) Comparative study on deep convolution neural networks DCNN-based offline arabic handwriting recognition. IEEE Access 8:95465–95482. https://doi.org/10.1109/ACCESS.2020.2994290
86. Almodfer R, Xiong S, Mudhsh M, Duan P (2017) Enhancing AlexNet for Arabic handwritten words recognition using incremental dropout. In: 2017 IEEE 29th international conference on tools with artificial intelligence (ICTAI). pp 663–669
87. Ali AAA, M S (2019) Arabic handwritten character recognition using machine learning approaches. In: 2019 Fifth international conference on image information processing (ICIIP). pp 187–192
88. Amrouch M, Rabi M, Es-Saady Y (2018) Convolutional feature learning and CNN based HMM for arabic handwriting recognition. In: Mansouri A, El Moataz A, Nouboud F, Mammass D (eds) Image and signal processing. Springer, Cham, pp 265–274
89. Maalej R, Kherallah M (2018) Convolutional neural network and BLSTM for offline arabic handwriting recognition. In: 2018 International Arab conference on information technology (ACIT). pp 1–6
90. Khémiri A, Echi AK, Elloumi M (2019) Bayesian versus convolutional networks for Arabic handwriting recognition. Arab J Sci Eng 44:9301–9319. https://doi.org/10.1007/s13369-019-03939-y
91. Khosravi S, Chalechale A (2022) Recognition of Persian/Arabic handwritten words using a combination of convolutional neural networks and autoencoder (AECNN). Math Probl Eng 2022:e4241016. https://doi.org/10.1155/2022/4241016
92. Awni M, Khalil MI, Abbas HM (2019) Deep-learning ensemble for offline Arabic handwritten words recognition. In: 2019 14th international conference on computer engineering and systems (ICCES). pp. 40–45
93. Smith LN (2017) Cyclical learning rates for training neural networks. In: 2017 IEEE winter conference on applications of computer vision (WACV). pp. 464–472
94. Almodfer R, Xiong S, Mudhsh M, Duan P (2017) Multi-column deep neural network for offline arabic handwriting recognition. In: Lintas A, Rovetta S, Verschure PFMJ, Villa AEP (eds) Artificial neural networks and machine learning – ICANN 2017. Springer, Cham, pp 260–267
95. Poznanski A, Wolf L (2016) CNN-N-Gram for HandwritingWord Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2305–2314