applied sciences

MDPI

Review

A Survey of OCR in Arabic Language: Applications, Techniques, and Challenges

Safiullah Faizullah [1,*], Muhammad Sohaib Ayub [2], Sajid Hussain [2] and Muhammad Asad Khan [3]

[1] Department of Computer Science, Islamic University, Madinah 42351, Saudi Arabia
[2] Department of Computer Science, Lahore University of Management Sciences, Lahore 54792, Pakistan
[3] Department of Telecommunication, Hazara University, Mansehra 21120, Pakistan
* Correspondence: safi@iu.edu.sa; Tel.: +1-848-239-7700

Abstract: Optical character recognition (OCR) is the process of extracting handwritten or printed text from a scanned or printed image and converting it to a machine-readable form for further data processing, such as searching or editing. Automatic text extraction using OCR helps to digitize documents for improved productivity and accessibility and for preservation of historical documents. This paper provides a survey of the current state-of-the-art applications, techniques, and challenges in Arabic OCR. We present the existing methods for each step of the complete OCR process to identify the best-performing approach for improved results. This paper follows the keyword-search method for reviewing the articles related to Arabic OCR, including the backward and forward citations of the article. In addition to state-of-art techniques, this paper identifies research gaps and presents future directions for Arabic OCR.

Keywords: optical character recognition; Arabic OCR; preprocessing; segmentation; classification; postprocessing

Citation: Faizullah, S.; Ayub, M.S.; Hussain, S.; Khan, M.A. A Survey of OCR in Arabic Language: Applications, Techniques, and Challenges. *Appl. Sci.* **2023**, *13*, 4584. https://doi.org/10.3390/app13074584

Academic Editor: Antonio Fernández-Caballero

Received: 26 February 2023
Revised: 28 March 2023
Accepted: 3 April 2023
Published: 4 April 2023

1. Introduction

Optical character recognition (OCR) enables the recognition of text characters from digital images, scanned documents, and video streams. OCR software analyses the image of text and converts it into machine-encoded text, which can then be edited, searched, and indexed. OCR can be used for a wide range of applications, including document scanning, automated indexing, and form processing. Further, OCR software can be integrated into various systems, such as document management systems, workflow systems, and mobile apps. There are some challenges to OCR systems, such as the writing style, text size, and quality of the document (handwritten, printed, or scanned), which cause challenges while implementing OCR [1], and a big challenge also comes while implementing OCR in hardware systems, which helps in many regards, such as a 'Quran Read Pen' that helps blind and illiterate people to read Quran [2].

1.1. Types of OCR

There are different types of OCR systems depending on the language and writing mode of the images. For example, the documents can be handwritten, printed, or scanned, and can contain one or more languages. Therefore, OCR systems can be categorized as unilingual or multilingual based on language. A unilingual OCR system can recognize only one language, and the Arabic OCR model is an example of a unilingual OCR system. On the other hand, some OCR systems perform recognition and extraction tasks for multiple languages; these are called multilingual OCR systems.

OCR systems can be categorized into offline and online OCR systems, as shown in Figure 1. An offline OCR system is a type of OCR system where the input documents are presented in scanned, printed, and handwritten formats [3]. These OCR systems provide

online services that can be used for various purposes such as mail sorting, bank cheque reading, signature verification, utility bill processing, and insurance applications. Digital pens help blind or illiterate people by reading text in audio form. Many online recognition systems are implemented in different fields such as number-plate recognition [4]. Similarly, an online OCR system is capable of receiving and processing real-time input images. For offline recognition, multiple models are used with different datasets for different algorithms to get better recognition accuracy [5].

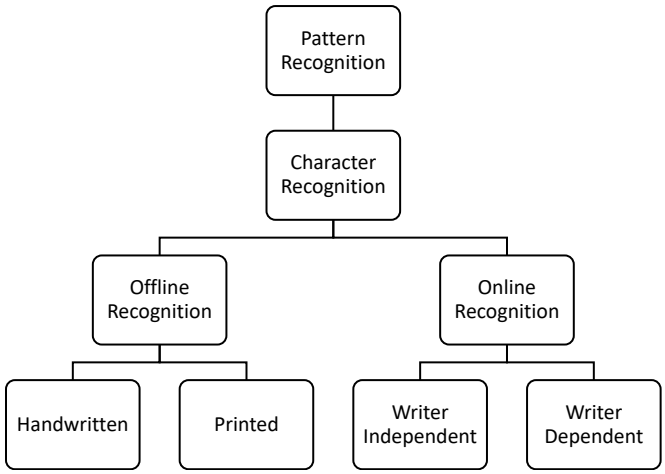

Figure 1. Types of OCR systems in Arabic and their modes of processing.

1.2. Language vs. Script

As we work on the Arabic OCR system, getting basic information about Arabic is essential. For this purpose, the concepts of language, script, and writing styles are crucial. Thus, language refers to the communication system humans use, which includes the grammar, vocabulary, and pronunciation used to convey meaning. On the other hand, the script refers to the written representation of a language, such as an alphabet or letters used to write words and sentences. A language can be written in multiple scripts and can use a script to write various languages. For example, the English letters are written in Latin script, while the Arabic language can be written in Arabic. In Arabic, the most-used styles are Naskh and Nastaleeq. Script similarities can be used to compensate for lack of availability of large amounts of training data for deep-learning-based OCR models [6].

In [5], the authors explain the basics of the Arabic language, i.e., Arabic is written from right to left and from top to bottom. It has 28 letters, which include three vowels, i.e., ا, و, and ئ. These letters change their shapes according to their usage in different words. Upper and lower case annotation does not exist. A total of 15 letters out of 28 have a point or dot above or under the letter. Arabic letters are connected from the right or left sides or both sides. However, six letters cannot be connected to their successors in a word; those letters are ءاؤرزو. There is another term called Tanween, which produces sound at the end of the words; these symbols are ًٌٍَُّ in written form. Punctuation marks have their own format in Arabic; e.g., the question mark in English is written as '?', but its shape in Arabic is '؟'.

1.3. Challenges

Some challenges are faced while designing an OCR system for the Arabic language. Arabic script uses diacritics and ligatures to indicate short vowels and certain consonant combinations, and OCR systems need to recognize and process these diacritics and ligatures correctly. The Arabic language has several types of two- and three-letter consonant combinations, i.e., shadda, sukoon, and tashkeel. An OCR system needs to recognize around 70 to 80 symbols in total for the Arabic language, including basic letters, diacritic marks, and other symbols used in the Arabic script. The Arabic script has 28 basic letters and several complex characters formed by joining multiple basic letters, and OCR systems

need to recognize and separate these complex characters. Arabic handwriting can vary greatly, making it more difficult for OCR systems to recognize the characters correctly. Arabic OCR datasets are usually smaller than those of other languages, leading to difficulty with training and fine-tuning the model. The images for OCR can be in multiple forms, i.e., computer-rendered images, scanned images, photographed images, and handwritten scans. These image types have challenges regarding the recognition rate for the OCR process.

Bafjaish et al. [7] also discussed some challenges of the Arabic OCR system. Dots come in Arabic in different places, sometimes above or below the baseline. These dots have much importance in the Arabic language; if you miss any dot somehow or during skew detection/correction, it will change the meaning of the letter or word, reducing the accuracy of the OCR model. Many of the scripts have a non-cursive style, meaning the letters present in a word have some gaps, making them easy to recognize, reducing the challenge, and making the task easy. However, the Arabic language has a cursive style , and the connectivity of letters makes text recognition more complicated. As Arabic letters are compounded to form a word, every font style shows a different level of ligature in words.

1.4. Applications

OCR systems are used now in many fields to make the workflow fast and accurate, so for this kind of digitization, OCR is used. In [8], the authors present a survey of the application of OCR and perform experiments for some applications. The OCR applications discussed are as follows:

- Invoice Imaging: Used in many businesses to track business records.
- Legal Industry: To digitize documents and enter the data directly into the databases, OCR is used.
- Banking: OCR is also widely used in banking services. For example, to process check payments, cheques are scanned and transferred in seconds.
- Healthcare: In healthcare, many forms, reports, and insurance applications are processed into databases and for other purposes; OCR helps to transfer all kinds of patient data.
- Captcha: Captcha is used to secure systems. A few letters, numbers, or both are used in a captcha, and the image is distorted. Humans can easily read this captcha, but not an average computer program.
- Automatic Number Recognition: It is used for surveillance systems to track vehicles' records by getting their number plates. OCR is also used to recognize the characters and numbers from the number plates.
- Handwriting Recognition: in this application of OCR, the text is extracted from handwritten documents and photographs. For this purpose, the model learns and identifies fonts and languages for better results.
- Scanned Receipts: some challenges comes while scanning receipts for extracting information from them, i.e., variations in receipt layout, noise, and distortion [9].

1.5. Brief OCR Process

Arabic OCR systems involve several complex steps, and Figure 2 shows the brief overflow of specific OCR steps to be followed. The image is first preprocessed to improve its quality and make recognization easier, which includes operations such as skew correction, noise reduction, and contrast enhancement. The text area is then broken down into individual segments of characters or words. The segmented characters are then recognized, and the best match is selected using a database of known characters. The recognized text then undergoes further processing to correct errors and improve accuracy. The final result is a machine-readable text document that software applications can edit, search, and analyze.

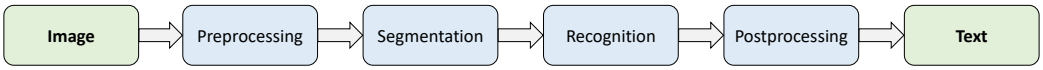

Figure 2. Brief overview of OCR process.

1.6. Goals and Outlines

This paper aims to provide an overview of the current state-of-the-art in Arabic OCR technology. It presents the main challenges and limitations of existing Arabic OCR systems. We highlight the main research trends and future directions for Arabic OCR development. We conclude by identifying the main research gaps and areas that need further study.

The rest of the paper is organized as follows: In Section 2, we review the datasets available to evaluate the Arabic OCR. In Section 3, we summarize the existing literature for each step of Arabic OCR, highlighting the main research trends and advances in the field. We conclude by summarizing the survey's main findings and highlighting the main research gaps and areas that need further study.

2. Datasets

The dataset is an important part of any OCR system to validate the results of OCR. It is especially challenging for the Arabic language because of the cursive nature of the Arabic language, diacritics, different writing styles in which each word's overall shape changes, text sizes, and other reasons. The collection of the Arabic dataset is also very limited due to the low-resource nature of the Arabic language. Previously, refs. [10,11] shared some commonly used datasets, as shown in Table 1. They presented the shared datasets for Arabic, Urdu, and Persian, both publicly available and otherwise.

2.1. Handwritten Text

Urdu and Arabic have many similarities. Writing styles are identical, and both have cursive nature as well; both start from right to left, and Urdu has about 39 to 40 letters; Arabic is similar to Urdu but has fewer characters. Urdu borrows a large vocabulary from Arabic (almost 30%). Most Urdu speakers can read Quran because of Urdu and Arabic similarities. Thus, their datasets and trained models are commonly used as well.

Table 1. Available datasets with their stats, dataset type, and mode of availability.

Dataset	Type of Content	Availability	Size of Dataset
ACTIV2 [12]	Embedded words	Public	10,415 text images
QTID [13]	Synthetic words	Private	309,720 words and 249,428 characters
IFN/ENIT [14]	Handwritten words	Public	115,000 words and 212,000 characters
AHDB [15]	Handwritten words and digits	Private	30,000 words
APTI [16]	Printed words	Public	113,284 words and 648,280 characters
HACDB [17]	Handwritten characters	Public	6600 characters and 50 writers
UPTI [18]	Printed text lines	Public	10,000 text lines
Digital Jawi [19]	Jawi paleography images	Public	168 words and 1524 characters
KHATT [20]	Handwritten text lines	Public	9327 lines, 165,890 words and 589,924 characters
ALIF [21]	Embedded text lines	Upon request	1804 words and 89,819 characters
ACTIV [22]	Embedded text lines	Public	4824 lines and 21,520 words
SmartATID [23]	Printed and handwritten pages	Public	9088 pages
Degraded historical [24]	Handwritten documents	Public	10 handwritten images and 10 printed images
Printed PAW [25]	Printed subwords	Upon request	415,280 unique words and 550,000 sub words
Checks [26]	Handwritten subwords and digits	Private	29,498 subwords and 15,148 digits
Numeral [27]	Handwritten digits	Public	21,120 digits and 44 writers
Forms [28]	Handwritten characters	Private	15,800 characters and 500 writers
KAFD [29]	Printed pages and lines	Public	28,767 pages and 644,006 lines
AHDBIFTR [30]	Handwritten images	Public	497 word images and 5 writers
ARABASE [31]	Handwritten text	Public	47,000 words and 500 free Arabic sentences
CEDAR [32]	Handwritten pages	Private	20,000 words, 10 writers, and 100 documents
CENPARMI [26]	Handwritten subwords and digits	Public	6000 digit images

Shafi and Zia [33] surveyed automatic Urdu text recognition techniques and described the algorithms, techniques, datasets, challenges, and future directions for Urdu OCR. Additionally, [34] reviewed the availability of datasets and suggested more training data to address the unique challenges of OCR systems.

Due to their similarities, both languages have some datasets available. The authors of [35] presented a dataset of handwritten Urdu numerals. In [11], the authors proposed an Urdu Nastaliq Handwritten Dataset (UNHD), which is written by 500 writers on A4-size paper and is available on request (https://www.kaggle.com/datasets/drsaadbinahmed/unhd-dataset, accessed on 28 March 2023). Khosrobeigi et al. [36] also presented a Persian language dataset; this dataset is collected from different Persian-language new websites, and the description of the dataset is shown in Table 2; this dataset is split into 80% for training and 20% for testing purpose.

Table 2. Example Persian dataset collected from different news websites.

Description	Stats
Total text lines of dataset	4,000,000
Total words	15,000,000
Unique words	200,000
Text lines per image	70
Total used fonts (with sizes)	11 fonts (sizes:12, 14, and 18)

There are some datasets available that are used for handwritten text recognition of Urdu, and, as we know, Urdu and Arabic use the same vocabulary and alphabet as well. Therefore, we can use Urdu datasets as well and achieve good results. For this purpose, ref. [37] presents some datasets of handwritten Urdu text recognition, which give outstanding results; the dataset descriptions and their availability are also shown in Table 3.

Table 3. Sample handwritten Urdu datasets.

	UPTI	CALAM	UNHD
Total writers	250	725	500
Text lines	60,000	3043	10,000
Words	240,000	46,664	187,200
Characters	970,650	101,181	312,000
Availability	Private	Private	Public

Naz et al. [38] summarized the state-of-the-art in OCR research for Urdu-like cursive scripts, concentrating on Nastaliq and Naskh scripts in the Urdu, Pushto, and Sindhi languages. The study discusses the quirks of these scripts as well as the text-picture databases that are readily accessible. Three categories have been established: printed, handwritten, and internet character recognition. The database is discussed, which includes 60,329 isolated digits, 12,914 strings, 1705 symbols, 14,890 isolated characters, and 318 different patterns of dates.

Alghamdi and Teahan [39] discussed the most commonly used datasets for training and evaluation of OCR systems for printed Arabic script, including the IFN/ENIT Arabic handwritten dataset, the "Handwriting Arabic Corpus" (HAC) dataset, and the RIMES dataset containing a large collection of printed and handwritten documents. The authors provide an overview of the available datasets and emphasize the importance of high-quality datasets for improving the accuracy of OCR systems.

Publicly available scanned image datasets are tested by [40], e.g., the WATAN and APTI datasets with extensive vocabularies. The datasets are split into a training set and a testing set, where training data contain 282,000 word images and 1,200,000 characters images while testing 5500 words, and 100,500 characters are used. The trained model

achieves an overall accuracy of 97.94%. As there are many challenges in Arabic optical character recognition (AOCR), [41] surveys various approaches and methods to detect and reduce errors.

In [42], the authors proposed a system synthesizing Arabic handwritten words and text pages to generate comprehensive databases for training and validating OCR systems. In the database, vocabulary of the 50,000 most-common Arabic words are used for error correction.

2.2. Printed Arabic

In Arabic OCR, printed, handwritten, and historical documents are used. To process printed Arabic documents, [43] presents top-down, bottom-up, and hybrid approaches and discusses the phases of preprocessing, segmentation, feature extraction, and classification.

An efficient, font-independent word and character segmentation algorithm for printed Arabic documents is proposed in [44]. Profile projection is used for the font-independent technique. Interquartile Range (IQR) is used for word segmentation. For character segmentation, two approaches are used, i.e., the holistic approach (segmentation-free approach) and the analytical approach, which is a segmentation-based approach, and the process followed for this purpose is shown in Figure 3. For this purpose, ATPI is the dataset used to make a font-independent OCR system that achieves 97.51% accuracy.

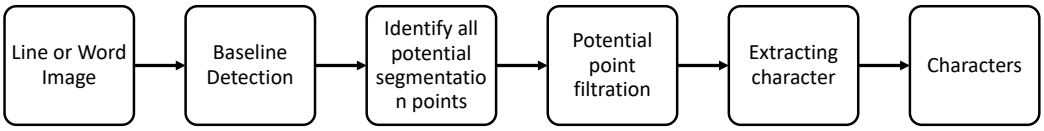

Figure 3. Character segmentation stages in order to recognize characters with maximum accuracy.

In [45], the authors propose a thinning algorithm in the preprocessing stage. A new chain code representation technique is proposed using an agent-based model for feature extraction from non-dotted Arabic text images. A character segmentation technique based on the extracted features is also introduced. A compression-based method is applied to recognize Arabic text in the classification stage. The system was tested on a public dataset and produced an accuracy of 77.3%.

The authors of [46] demonstrate the effective use of unsupervised algorithms for writer attribution of historical scanned documents and forensic document analysis. Some distinct handwriting styles differ in various ways, including character size, stroke width, loops, ductus, slant angles, and cursive ligatures. Additionally covered are prior efforts on labeled data that provide excellent accuracy rates utilizing the Hidden Markov Model (HMM), Support Vector Machine (SVM), and semi-supervised Recurrent Neural Networks (RNN).

Transformer-based models are a type of deep learning method to deal with sequential data [47]. Several metrics are used to evaluate the performance of the proposed method; those metrics are character error rate (CER) and word error rate (WER). Furthermore, results show that the proposed method improves the recognition rate of historical documents.

The data generated by IoT devices such as the Quran Read Pen, which helps to read Quran specifically to illiterate or blind people [48], is shared via the Quranic Text Image Dataset (QTID). It contains 309,720 images of words and 2,494,428 characters taken from the Quran, which uses the sequence-to-sequence model and CNN and achieves a high recognition rate. The character recognition rate (CRR) with and without diacritics is about 97.60% and 97.05%, respectively, and the overall recognition rate of this model is 99.48%, while the CNN model gives the CRR with and without diacritics of about 98.90% and 98.51%, respectively.

Feature extraction and classification techniques are used for character segmentation in ancient manuscripts for their preservation and information extraction [49].

2.3. Scanned Documents/Receipts

Information extraction from scanned documents is difficult compared to regular documents because of the rough layout and low resolution. Preprocessing involves processing the scanned document successfully, as information extraction from the scanned documents/receipts is the key perspective. ICDAR [50] presents a competition wherein 1000 scanned receipts are used to extract information; this competition includes some tasks such as text recognition, layout analysis, and information extraction.

2.4. Quranic Text

As Quran is a Holy Book, it is recited worldwide, and everyone wants to recite it correctly without any mistakes. Bashir et al. [51] review the Quranic NLP techniques, approaches used, tools, and datasets, and recitation via speech-recognition method. The techniques used in the paper are text preprocessing, text matching, clustering, classification, and speech processing. Quranic NLP work includes grammatical NLP analysis and semantic- and ontology-based technologies using BLSTM. The model took recitation of different reciters for training purposes, and a feature widely used for speech recognition, named mel-frequency cepstral coefficients (MFCCs), gives a 99.89% recognition rate for 3 s of recitation, which is far better than all of the other techniques.

3. OCR Process

The OCR process refers to identifying and converting printed or handwritten text characters into machine-encoded text. It typically involves several steps, including preprocessing, segmentation, recognition, and postprocessing. During preprocessing, the input image is cleaned up and enhanced to improve the quality of recognition. Segmentation involves breaking the image into individual or groups of letters or characters. Feature extraction is the process of identifying and extracting the relevant features of each character, such as its shape, size, and orientation. In recognition, the characters are classified by comparing them to a set of known characters, and the best match is selected as the recognized character. Finally, postprocessing of recognized text is performed to remove errors from text and improve accuracy and overall results of OCR. OCR recognition accuracy can vary depending on several factors, such as the quality of the input image, the font type and size, and the language being recognized. The flow of the overall OCR process is shown in Figure 4. We have provided a high-level description of the various techniques and methods involved in the OCR process in Table 4.

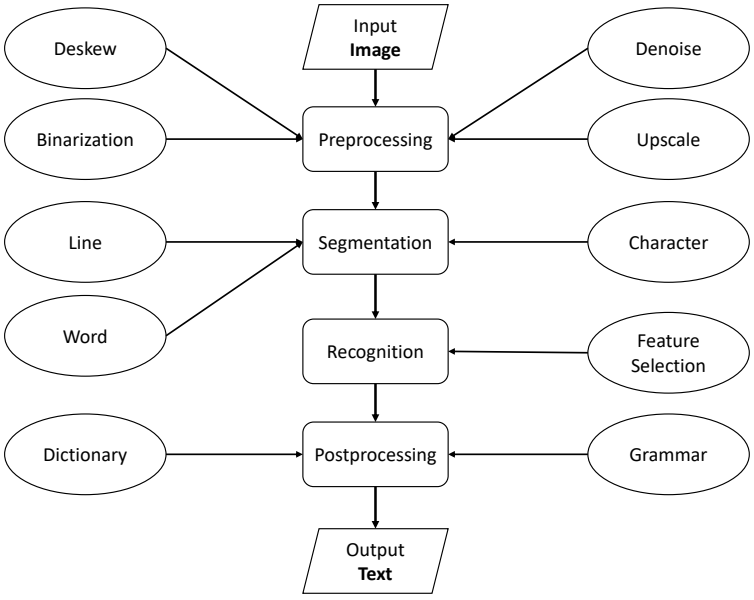

Figure 4. The flow of the OCR process along with OCR phases and methods involved.

Table 4. Comparison of techniques applicable in various OCR methods.

Technique	Method	Brief Description
Preprocessing	Binarization	Transforms the input image into a binary format
	Keystone Correction	Aligns the distortion on the edges of the image
	Skew correction	Corrects the angle of the rotated text
	Denoising	Filters-out the extra-noisy pixels from the image
	Dilation	Restores an eroded image by cleaning it up
	Erosion	Removes object boundaries and unwanted parts in images
	Thinning	Reduces thickness of objects by removing boundary pixels
	Upscaling	Enhances the resolution of the image
Segmentation	Line	Image is divided into lines for line-by-line processing
	Word	Each line is divided into words using spacing methods
	Character	Image of each word is divided into individual characters
Recognition	Template Matching	Matches an input image with predefined characters
	Feature Extraction	Extracts features and classifies image using learning algorithm
	Neural Networks	Uses interconnected neurons to predict text from image
	Deep Learning	Uses neural networks with many layers to learn patterns
	Decision Trees	Builds tree-like structure with decisions and consequences
	SVM	Constructs hyperplane separating image into different classes
	Naive Bayes	Uses Bayes' theorem to classify an input image
	Random Forest	Builds multiple decision trees, combining their outputs
	CNN	Uses deep learning with convolutional layers to classify image
	RNN	Neural network for processing sequences (characters in OCR)
	*k*NN	Classifies image based on *k*-nearest neighbors' majority class
	HT	Detects lines, circles, and edges from image for text extraction
	HOG	Computes image gradients in histograms and extract features
	HMM	Models transition probabilities of text for accurate recognition
	Profile Projection	Extracts character features using projection onto 1D axis
Postprocessing	Spell-check	Error correction, text enhancement, and restoration
	Contextual Analysis	Analyses the surrounding words based on specific context
	Confidence Scoring	Assigns scores to words—higher score means more accurate
	Language Model	Uses large corpus of text to guess best word in context
Evaluation	Character Error Rate	Percentage of characters incorrectly predicted
	Word Error Rate	Percentage of words incorrectly predicted
	Recognition Rate	Percentage of characters/words correctly recognized

3.1. Preprocessing

The formatting issues in images can have a negative impact on the accuracy of OCR models. Examples of these issues include problems related to image orientation or color correction. To improve the accuracy of these models during the training phase, image preprocessing techniques are commonly used. These techniques may involve resizing, grayscale conversion, skew correction, and/or enhancing the resolution of the image.

3.1.1. Binarization and Thinning

Binarization converts a grayscale or color image into a binary image, representing each pixel as either black or white. It is an essential preprocessing step that helps to segment the text from the background and increase the contrast between the characters and the background. The objective of binarization is to transform the input image into a binary format that enhances the visibility of the characters and makes them more easily recognized by the OCR system. Various binarization techniques are used in OCR, including thresholding, adaptive thresholding, and Otsu's method.

A method for preprocessing images of historical documents for OCR and search includes image binarization, skew correction, and line segmentation. The method was tested on a dataset of historical documents. The results show that it improves the accuracy of OCR by reducing errors caused by skew and noise and can be effectively applied to historical documents of various types. The binarization step of the method converts the image into a black-and-white image to make it easier for OCR software to recognize the text [52].

An approach for binarization of non-uniformly illuminated document images to accurately recognize alphanumerical characters is presented in [53]. The proposed method combines local and global thresholding methods, i.e., Sauvola and Otsu methods to achieve robust binarization and improved performance compared to existing binarization methods. In the Sauvola binarization method, the local threshold is calculated using a Sauvola algorithm, which takes into account the local mean and standard deviation of the pixel intensities. In the Otsu binarization method, the global threshold is calculated using an Otsu algorithm, which maximizes the variance between the two classes of pixel intensities.

Thinning, also known as skeletonization, reduces the thickness of the image by deleting the boundary pixels while preserving the shape and structure. The goal of thinning is to obtain the structure of the objects in the image.

Tellache et al. [54] propose and compare different thinning algorithms for improving the performance of OCR for Arabic script. The results show that the Hybrid algorithm performed the best and improved the OCR accuracy by reducing the errors caused by variations in line thickness. The method can be effectively applied to different types of Arabic text, including handwritten and printed text. Results indicate that the Hybrid algorithm improved the OCR accuracy by reducing the errors caused by variations in line thickness. The results also show that the Hybrid algorithm can be effectively applied to different types of Arabic text, including handwritten and printed text.

3.1.2. Denoising

Unwanted changes in the intensity of an image that cover up the underlying image structure are known as noise. Noise can appear in images or documents that contain text in different ways, i.e., scanning documents, compressing files, printing documents, and noise during text recognition in the form of errors. For scanning documents, the noise introduced is in the form of changes to document quality, exposure, lighting, and blurring of the text. Noise can also be introduced during file compression, as it is used to reduce the actual size of the files, so it can add noise in the form of quality loss. Noise can also be introduced while printing the document due to the lack of printing quality of that particular machine or due to variations in ink or toner density or blurring.

Noise and image distortions significantly degrade OCR performance [55]. Noise removal is necessary for every image-processing task, and filters are used to remove unwanted variations in the image while preserving the essential details. Filters are used according to the filter behavior [56]; for example, a Gaussian filter is used to smooth an image by reducing high-frequency noise. A median filter works by replacing pixel values with the median value of the neighbouring pixels to remove impulse noise.

The authors of [57] proposed a deep learning architecture based on a convolutional neural network (CNN) for detecting and recognizing text in distorted document images of different languages. The proposed approach combines two specialized modules for text detection and recognition for automatically learning discriminative features for character recognition; it achieves outstanding performance, surpassing the best competing models by at least 13% for text detection and 7.5% for text recognition. The developed global model demonstrates a high level of robustness and significantly outperforms all other schemes in comprehensive benchmarks.

Denoising is also performed using morphological operations. Morphological operations process images according to their shapes; each pixel corresponds to its neighboring pixels. Salt-and-pepper noise is a common type of noise that appears due to random black-and-white pixels in an image, and morphological operations are used to remove such noise. Erosion and dilation are commonly used morphological operations for denoising [58]. Erosion eliminates the isolated noise pixels, and dilation fills up small, empty holes around the image caused by noise [59]. Opening removes small noisy pixels, whereas closing operations fill empty gaps in the image. The combination of opening and closing is generally used to denoise the image in the preprocessing step, as shown in Figure 5.

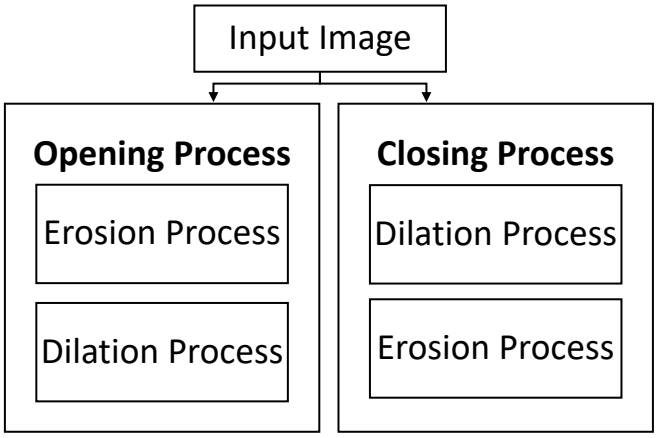

Figure 5. Opening and closing of an image.

3.1.3. Deskewing

Deskewing is the detection of rotated text in an image and computing its angle to correct its rotation [60]. It involves text-block detection, computing the angle of the rotated text, rotating the text, and correcting the image's skewness.

An adaptive deskewing method for document pictures that recognizes the image type and selects an appropriate correction technique based on image type is proposed by [61]. The text direction of the document picture is determined by the method and is used as a parameter to pick a more appropriate projection direction. The research provides many approaches for repairing various sorts of document photographs, as well as a layout-based image categorization system. The results of the experiments suggest that the algorithm is accurate and resilient, although its complexity may restrict its capacity to predict skew over a specific threshold. A voting-based deskewing method is proposed by [62]; it chooses the best deskewing algorithm based on the accuracy of skew correction for large digitization projects.

The Probabilistic Hough Transformation (PHT) method for skew detection and correction in OCR systems for scanned documents is presented in [63]. The method works in two steps: detecting lines of text and clustering them. Factors that affect OCR performance, such as skew, blur, image distortion, and noise, are addressed in the preprocessing phase, with skew being the main focus; an example is shown in Figure 6. The proposed method was tested on different datasets and showed better results than other methods used by researchers. The method calculates skew angles using the following equations:

$$n_{hieght} = (n_{width} * h)/w \tag{1}$$

where n_{height} represents normalized height, n_{width} represents normalized width, and w and h represent width and height, respectively.

$$\begin{aligned} m &= (y_2 - y_1)/(x_2 - x_1) \\ A &= arctan[(y_2 - y_1)/(x_2 - x_1)] \end{aligned} \tag{2}$$

where m represents slope/gradient of a line, and A represents the angle of each line. The proposed skew detection and correction method is used on different datasets and achieves good results compared to other methods used by researchers.

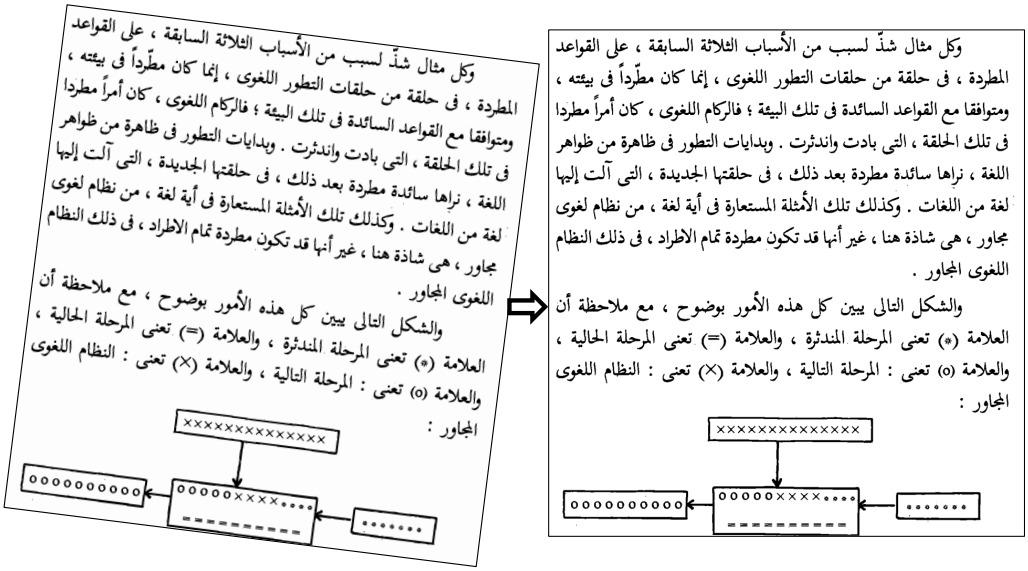

Figure 6. A skewed document (on the left) is deskewed (on the right) to achieve better OCR results.

Hough transformation (HT) is a technique to detect lines in an image and deskew the text. The authors of [7] use the HT method for skew correction. HT creates a parameter space (Hough space) in which each point in the space represents a possible line in the image. By detecting lines in the image, the method can detect the skew of the text by finding the angle at which the lines are inclined. The proposed method can detect skew angles with an accuracy of about 97% and can correct them with an average error of about 0.8°. The method was evaluated through experiments on a dataset of Quran images. The method is robust to noise, which means it can still detect and correct skew even when images have noise present. It can also be applied to images of different quality, showing consistent performance regardless of the image quality. The method was tested on a diverse set of Quran images, including images with different text sizes and levels of quality, and the results were consistent.

3.1.4. Keystone Correction

Keystone correction is used to correct slanted images. OCR algorithms work best when the text in the image is aligned with the x-axis. However, the text may be slanted due to how the original document was scanned or photographed. Keystone correction involves applying mathematical transformations to the image to correct for the slant of the text, which can be done using various algorithms, such as Hough transform or RANSAC. Distortion is aligned at the top/bottom using vertical keystone correction and left/right of the image with horizontal keystone correction. After capturing the image, there will be many technical challenges with the image that make it difficult to read the text. The morphological process and Bézier curve method resolve these challenges [64].

3.1.5. Upscaling

The process of upscaling, also known as Super Resolution, involves enhancing the resolution of the image. Random forests can also be used to upscale images [65]. They use the standard benchmarks for super-resolution and present the training and evaluation accuracy. A deep learning method for upscaling binary document images using super-resolution is the generative adversarial network (SRGAN) [66], which improves readability and OCR performance compared to traditional interpolation methods, as per evaluation metrics. The method involves training a CNN on low-resolution and high-resolution binary document images to generate high-resolution images.

3.2. Segmentation

Segmentation is an important step in OCR involving separation of an image into its constituent parts, such as lines, words, and characters, for recognition. Three types of segmentation techniques are mainly used, i.e., line, word, and character segmentation. Word segmentation, specifically, is challenging in cursive languages such as Arabic due to a lack of clear separation between characters. Traditional segmentation methods in Arabic OCR rely on rules and heuristics based on character features, but deep learning techniques such as convolutional and recurrent neural networks have shown promising results in automatic segmentation. Accurate segmentation is crucial for recognition, and improving Arabic OCR segmentation can have significant implications for document digitization, text-to-speech conversion, and language translation.

The techniques used in [67] include image processing techniques such as thresholding to convert the image into a binary image, morphological operations to perform operations such as erosion and dilation on the binary image, and connected component analysis to identify and label connected regions in the image. The authors use two databases for evaluating the performance of the proposed method. The first database is the publicly available RDI-Arabic dataset, which consists of 1000 images of Arabic text documents. The second database is a new dataset created by the authors and consists of 1000 images of Arabic text documents. The research results show that the proposed image-processing techniques can accurately segment Arabic text documents into text lines, words, and characters with a high degree of accuracy. The researchers use several evaluation metrics, such as F-score, precision, and recall, to evaluate the performance of the proposed method. The results show that the proposed method outperforms traditional methods regarding segmentation accuracy.

Urdu language characters are the super-set of Arabic language characters, and certain challenges are faced when performing segmentation of Urdu-like cursive languages [68]. Arabic is mainly written in Naskh, while Urdu is written in Nastaliq style. There are some challenges in the segmentation of Urdu script, i.e., cursive nature, difficult fonts such as Nastaliq, and letters changing their shape into different forms as required in the word; and cue points are hard to find in the Naskh or Nastaliq style. That is why segmentation is challenging, and character segmentation has been considered difficult in previous research. Researchers mostly used the projection profile method for segmentation, which can perform a vertical projection of the given text. However, in Arabic or Urdu, text writing starts from right to left and top to bottom, so vertical and horizontal projection is required. Analytical approaches are difficult and give the wrong character recognition results, but explicit and implicit recognition systems also give better accuracy. At the same time, holistic approaches are considered best by researchers for better accuracy with the correct recognition. Furthermore, there are better approaches than segmentation-free approaches for large vocabularies. For a reasonable accuracy rate, segmentation should perform well using the approaches that are correct and appropriate for segmentation.

Thorat et al. [69] presented a survey to discuss the methods used by previous researchers. It discussed OCR systems, tools, applications, phases, and methods. There are two types of documents, i.e., unilingual and multilingual documents. Some OCR systems are discussed, i.e., Google Docs OCR, Tesseract, ABBYY FineReader, Transym, and I2OCR, which help to provide services and help to extract text from different types of documents with different languages, whether they are unilingual or multilingual. Multilingual documents contain text from multiple languages, and the techniques used are binarization, layout analysis, page segmentation, preprocessing, feature extraction, classification, and recognition. Some approaches are used for the segmentation-free approach and HMM. There are some applications where OCR systems are used to ease use and increase work productivity in healthcare, education, banking, insurance, automatic exam paper checking, bills and invoices, newspapers, and comics. Some phases are discussed to process the document to get better accuracy, including image acquisition, preprocessing, segmentation,

classification and recognition, and postprocessing. Moreover, methods used by previous researchers are matrix matching, fuzzy logic, structural analysis, and neural networks.

3.2.1. Line Segmentation

In line segmentation, the skew-corrected image is divided into lines. Line segmentation is an important step in OCR as it allows the separation of the image into individual lines so that the OCR system can process them one-by-one and improve the recognition of the text in the image. Connected component analysis, project profile, and machine learning-based approaches can be used to perform line segmentation. Once the lines of text have been segmented, the OCR system can process each line individually, which can improve the accuracy of the character recognition process.

A method for line segmentation of printed Arabic text with diacritics using a divide-and-conquer algorithm is presented in [70]. It breaks the image of printed text into smaller blocks, applies image processing techniques to extract the text lines, and then applies a set of heuristic rules to remove false positives and adjust the segments as necessary. It uses image processing techniques such as thresholding, morphological operations, and connected component analysis. The research results show that the proposed method can accurately segment printed Arabic text with diacritics into text lines with a high degree of accuracy. The research uses several evaluation metrics such as F-score, precision, recall, and F-Measure to evaluate the performance of the proposed method.

Brodic et al. [71,72] proposed a basic standardized test framework for evaluating the quality of text line segmentation algorithms in OCR systems for accurate handwritten text recognition. Their proposed framework includes experiments for measuring the accuracy of text line segmentation, skew rate, and reference text line evaluation.

3.2.2. Word segmentation

After line segmentation, each word from the line is segmented by dividing the line of the text into individual words. Several techniques are used for word segmentation, i.e., the Spacing method involves using the spaces between words to segment the text into words. The dictionary-based method uses a dictionary of words to match against the text and segments the text into words based on the matches. Character-based methods use a combination of known character patterns, such as word breaks and punctuation, to segment the text into words. Deep-learning approaches use supervised learning on annotated datasets to learn the complex relationships between adjacent characters in order to infer word boundaries [73,74].

The authors of [75] present word segmentation in Arabic handwritten images using a convolutional recurrent neural network (CRNN) architecture. The authors employ a sliding-window approach for word segmentation, where each window is classified as either a word or non-word using a support vector machine (SVM) classifier. The experimental results show that the proposed CRNN architecture achieves state-of-the-art performance in Arabic handwriting word recognition, with an accuracy of 86.95% on the IFN/ENIT dataset.

Patil et al. [76] propose a semantic segmentation approach for images containing mixed text to segment the image into different regions based on their content. The segmented regions are then processed using different OCR methods that are specifically tailored to the type of text in each region.

3.2.3. Character Segmentation

Character-level segmentation is a technique used to segment an image of a single word into individual letters and characters. It is an optional step depending on the context of the OCR system that is being used. It may be unnecessary if the text has separate letters within a word, as the letters and characters can be segmented in the previous step using a threshold. However, character-level segmentation must be performed if the text has cursive handwriting or a nature where letters are joined.

A method for recognizing and transcribing text from a visual Arabic scripting news ticker from a broadcast stream is presented in [77]. The technique used in this research includes image processing techniques such as thresholding, morphological operations, and connected component analysis to segment the text from the background. It uses machine learning algorithms such as CNNs and long short-term memory (LSTM) to transcribe text. The research results show that the proposed method can accurately recognize and transcribe text from a visual Arabic scripting news ticker from a broadcast stream. The research uses several evaluation metrics, such as character error rate (CER) and word error rate (WER), to evaluate the performance of the proposed method. The results show that the proposed method outperforms traditional methods in recognition accuracy, achieving a lower CER and WER than traditional methods on the dataset used in the research.

Alginahi [78] discusses Arabic character segmentation approaches, including traditional methods such as vertical and horizontal projection, contour tracing and thinning, template matching, neural networks and HMM, holistic approaches and segmentation-free approaches, projection profile, baseline, contour tracing, graph theory, and morphology. The paper also discusses the challenges and limitations of each approach and suggests areas for future research. It also discusses the benefits and problems with character segmentation, especially in Arabic; problems are due to its cursive nature and the different shapes of each character depending on the word's appearance. Problems also occur because of datasets. Therefore, the Arabic Language Technology Center (ALTEC) have provided limited free access to a reliable dataset.

A method for segmenting characters, letters, and digits from Arabic handwritten document images using a hybrid approach is presented in [79]. The method uses image processing techniques, such as thresholding, morphological operations, and connected component analysis, to segment the text from the background and to separate the characters. In addition, machine learning algorithms, including *k*-means clustering and a Random Forest Classifier, are used to classify the segments into individual characters. The research demonstrates that the proposed method can accurately segment characters from Arabic handwritten document images with a high degree of accuracy. The method outperforms traditional methods regarding segmentation accuracy, as evidenced by several evaluation metrics, including F-score, precision, and recall. The proposed method achieves a higher F-score, precision, and recall than traditional methods on the dataset used in the research.

Morphological operators are also used for segmenting Arabic handwritten words [80]. The process involves using morphological operations, such as erosion and dilation, to extract the text from the background and segment the words. The technique used in this research is based on morphological operators, which perform operations such as erosion and dilation on binary images, to extract the text and separate the words. The method also uses image processing techniques, such as thresholding, to convert the image into a binary image and connected component analysis to identify and label connected regions corresponding to words. The research results show that the proposed method can accurately segment Arabic handwritten words with a high degree of accuracy. The research used several evaluation metrics, such as F-score, precision, and recall, to evaluate the performance of the proposed method. The results show that the proposed method outperforms traditional methods in terms of segmentation accuracy.

Some previous works have proposed segmentation-free approaches for Arabic and Urdu OCR, but they do not produce accurate results on clean text. The authors of [18] apply a machine learning model on clean Urdu and Arabic datasets, producing 91% and 86% accuracy on clean UPTI datasets. The authors of [34] review current approaches and challenges unique to Urdu OCR and suggest that future research should focus on developing more-sophisticated algorithms, improving training data, and addressing the unique challenges of the Urdu script. They also propose that integrating Urdu OCR with other technologies, such as machine learning and computer vision, would provide new opportunities for research in the field.

Handwritten digit recognition is discussed in [81]. It has various applications and is used in different fields such as postal mail sorting, bank check numbering, amount processing, and number entries of various forms such as taxes, insurance, and utility bills. There are handwritten images of 10 digits in the dataset from 0 to 9, and this dataset is taken from MNIST, which contains 60,000 and 10,000 images for training and testing purposes, respectively. Then, some phases and approaches are used to process the images to train and test the model, which helps to get better accuracy and gives a maximum recognition rate. Discussed phases and approaches are preprocessing and feature extraction, and then classification is used. In classification, machine learning approaches, i.e., Decision Tree, Support Vector Machine (SVM), and Artificial Neural Network (ANN), are used. The deep learning approach implements a Visual Geometry Group model with 16 layers (VGG16 model); this model is used for deep-learning image-classification problems. By using the proposed techniques, we get better recognition and a high accuracy rate; i.e., in decision tree 86%, SVM 91%, ANN 97%, and CNN 98.84% accuracy is achieved.

3.3. Recognition

Recognition, also called classification in some previous works, is the process of identifying and assigning a specific character or set of characters to a given input image. After preprocessing and segmentation, the OCR system compares the extracted features of each character or group of characters with a set of predefined templates or models. The OCR system creates these templates during a training phase, where a large dataset of sample images is used to teach the system how to recognize each character.

The classification/recognition process can involve several algorithms, including template matching, neural networks, and support vector machines. Template matching involves comparing the features of each character to predefined templates and selecting the template with the closest match to the recognized character. Neural networks and support vector machines use machine learning algorithms to learn and classify patterns in the data and can often achieve higher accuracy than template matching.

A survey of feature extraction and classification techniques is presented in [82]. The techniques include digitization, preprocessing, segmentation, feature extraction, and post-processing. Statistical, structural, template matching, artificial neural network, and kernel classification methods are also discussed.

An efficient feature-descriptor selection for improved Arabic handwritten word recognition is presented in [83]. The approach uses three image features, Histogram Oriented Gradient (HOG), Gabor Filter (GF), and Local Binary Pattern (LBP), for feature extraction, and trains a *k*NN algorithm to build models. The best model achieved an accuracy of 99.88%. A publicly available IFN/ENIT Arabic dataset is also introduced. The researcher use the global approach in the research, which is considered successful and in many cases is used more than the analytical approach.

Various methods and techniques are used in multilingual OCR [84], including preprocessing, binarization, segmentation, feature extraction, and recognition. Segmentation uses three different approaches, i.e., top-down, bottom-up, and hybrid approach, including page, line, and word/character level. The research also explores the challenges to and limitations of current multilingual OCR systems, such as dealing with different scripts and languages and the need for large amounts of annotated training data. The paper highlights the importance of multilingual OCR for applications such as digital libraries, document archiving, and machine translation. Overall, the paper provides a comprehensive overview of the current state of multilingual OCR research and its potential applications. The paper highlights the importance of multilingual OCR for various applications such as digital libraries, document archiving, and machine translation.The authors of [85] discuss the ongoing design of a tool for automatically extracting knowledge and cataloging documents in Arabic, Persian, and Azerbaijani using OCR, text processing, and information-extraction techniques.

A method using a combination of CNN and RNN for recognizing Arabic text in natural scene images is presented in [86]. The method utilizes an attention mechanism to focus on the most relevant parts of the image and improve text recognition accuracy. Two datasets are used in this research for training purposes, i.e., ACTIV and ALIF. ACTIV contains 21520 images of text lines, while ALIF contains 6532 images of text lines; both datasets are extracted from different news channels. Testing is done on the Arabic natural scene text dataset (ANST) the authors created. The proposed method is evaluated on this dataset, which shows that it outperformed the state-of-the-art methods for Arabic text recognition in natural scene images with an accuracy of 92.4%. The model uses CNN to extract features from the image and RNN to process the features and generate the text recognition output. At the same time, the attention mechanism improves the accuracy of recognition (https://tc11.cvc.uab.es/datasets/type/11, accessed on 18 March 2023).

Language detection, document categorization, and region of interest (RoI) identification with KERAS and TensorFlow are used to perform manuscript analysis and recognition in OCR systems [87]. The RoI includes tables, titles, paragraphs, figures, and lists. The deep-learning-trained model uses bounding-box regression to identify the target and serves as a reference point for object detection. The system integrates the fast gradient sign method (FGSM) and uses deep learning to recognize multilingual systems. It also investigates page segmentation methods to enhance accuracy. The system performs well against adversarial attacks on Arabic manuscripts and achieves an accuracy of 99%. It uses Hierarchical Agglomerate Clustering (HAC) to group objects in clusters based on similarity/relation. The research aims to improve preprocessing and identify parameters to enhance the accuracy of page segmentation methods.

A survey of various methods and techniques used for recognizing text in natural images and videos is presented in [88]. The authors discuss various challenges and propose approaches for recognizing text in the wild due to font, color, and background variations. The paper covers traditional methods as well as more-recent methods based on deep learning, such as CNN and RNN. The authors also discuss evaluation metrics and datasets used for text recognition in the wild. Overall, the paper provides an overview of the state-of-the-art in text recognition in the wild and highlights areas that need further research. Additionally, the authors provide a comprehensive review of publicly available resources on their Github repository (https://github.com/HCIILAB/Scene-Text-Recognition, accessed on 22 March 2023).

Bouchakour et al. [89] use the CNN classifier for printed Arabic OCR using a combination of texture, shape, and statistical features. Evaluation of the proposed method achieves an accuracy of 97.23%, which shows the effectiveness of combining image features with a CNN classifier. Similarly, the authors of [90] conduct experiments to analyze the impact of different hyper-parameters and network architectures on the performance of a CNN model for OCR of handwritten digits.

The authors of [91] survey OCR methodologies for Urdu fonts, such as Nastaliq and Naskh, and other similar languages having Urdu-like scripts, such as Arabic, Pashtu, and Sindhi. Moreover, the main focus of this survey is to compare all of the phases involved in OCR, i.e., Image Acquisition, Preprocessing, Segmentation, Feature Extraction, Classification, and Recognition. The Urdu Printed Text Images (UPTI) dataset is divided into training, testing, and validation. It contains about 10,000 text images; so for implementation, multidimensional LSTM-RNN is used, and it achieves an accuracy of 98%. Many stages in the phases make the OCR system better, and it is important to follow all of the stages to make a perfect system, as shown in Figure 7. In the past, researchers used multiple datasets and found a specific output such as character recognition or ligatures recognition and achieved good accuracy.

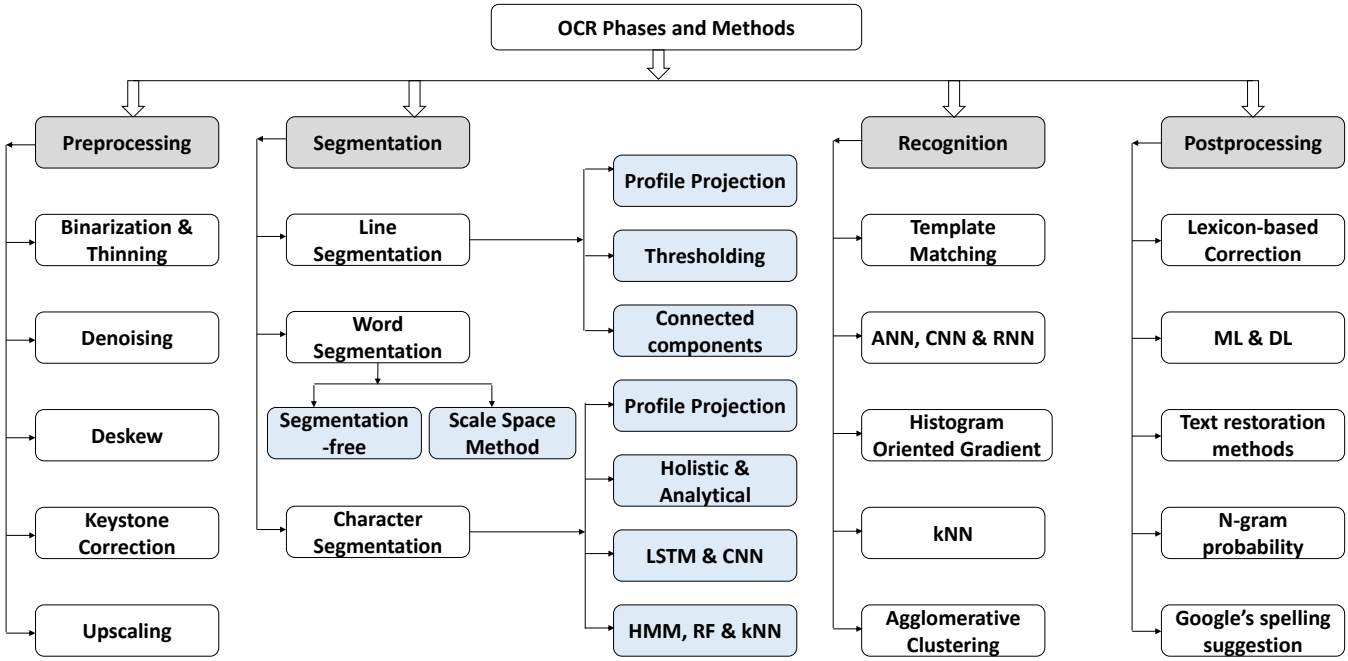

Figure 7. Processes and techniques in each phase of the OCR system.

An artificial neural network (ANN) classifier to identify the characters in a text is presented in [92]. The dataset is generated from different documents with different qualities of the result. A preprocessing step is used to eliminate noise; then, the segmentation phase is done using multiple steps, i.e., line, word, and character segmentation. A feature-extraction step for the character's image is performed to obtain features for all the pixels. The authors trained the ANN classifier on a dataset of printed Arabic text and then evaluated its performance on a separate test set. The results show that the system can accurately recognize the characters in the test set with high recognition rates. A KERAS model is used with a three-layer ANN classifier for character classification. Noise density and multi-spatial resolution matrices are used for evaluation, showing a fast, efficient, high-performance OCR system.

Mittal and Garg [93] review the various techniques used for text extraction from images and documents using OCR; they also discuss the challenges and limitations of text extraction. Reviewed papers are grouped on the basis of the types of OCR techniques that are used. The authors found that the most-used OCR techniques are based on machine learning, specifically deep learning. They also found that the OCR techniques that use multiple-recognition engines and preprocessing techniques perform better than single-recognition-engine-based techniques.

Deep learning models such as CNN, RNN, and attention-based models are discussed in [94]. The paper discusses the performance of models on different Arabic handwritten datasets, and these models show much-improved character recognition rate, word recognition rate, and overall recognition rate. The researchers also discussed challenges dealing with different handwriting styles and sizes.

The authors of [95] present a method for recognizing Arabic handwritten characters using different Holistic techniques, CNN, and deep learning models. At the same time, all of the techniques are well reviewed in this research, i.e., preprocessing, segmentation, feature extraction, recognition, and postprocessing. The authors found that using these models and techniques properly improves the recognition rate of the system by a significant margin.

In [96], automatically extracting and processing text from images is discussed. The challenges that may arise in OCR stages are also explored, as well as the general phases of an OCR system, including preprocessing, segmentation, normalization, feature extrac-

tion, classification, and postprocessing. Additionally, the paper highlights OCR's main applications and provides a review of the state-of-the-art at the time of its publication.

The approaches to processing text from documents are reviewed in [97]. In this review, text detection and text transcription are discussed. Text detection is a task whereby a process detects or finds text in a document or image. It is a difficult task but can be detected easily by box bounding and text detection as object detection. Text detection as object detection is challenging but is solved using computer vision tasks such as single-shot multi-box detectors and fast R-CNN models. Meanwhile, in text transcription, the text is extracted in editable form from the document or image of interest. Document layout analysis is the dominant part of selecting the region of interest. Then, the authors identify some datasets used in past research: ICDAR, Total-Text, CYW1500, SynthText, and the updated dataset FUNSD.

3.4. Postprocessing

Postprocessing is the final step in the OCR process; it involves improving the accuracy and quality of the recognized text. Postprocessing techniques can include various methods, such as spell checking, contextual analysis, confidence scoring, and language-model integration. Spell checking involves comparing the recognized text against a dictionary of words to identify and correct spelling errors. Contextual analysis analyses the recognized text within the context of the surrounding text to identify and correct errors that may be caused by confusion with other words. Confidence scoring assigns a confidence score to each recognized character based on the certainty of the OCR system's recognition, and characters with lower confidence scores are flagged for review or correction. The language model is also used to analyze the recognized text in the language context. Postprocessing significantly improves the accuracy and quality of the recognized text.

An overview of the different postprocessing techniques developed and applied to OCR output, including methods for correction of errors, text enhancement, and text restoration, is discussed in [98]. Various methods are used in postprocessing, including spell checking, grammar checking, lexicon-based correction, machine learning, and deep-learning-based approaches. The paper also discusses using different types of features, such as character-level, word-level, and document-level features, as well as preprocessing techniques, such as segmentation and normalization.

Several approaches have been used in the postprocessing of OCR output to improve the accuracy and completeness of the recognized text:

- Spell checking: checks the spelling of the recognized text and corrects any errors by comparing it to a dictionary [99].
- Grammar checking: checks the grammar of the recognized text and corrects any errors by comparing it to a set of grammar rules.
- Lexicon-based correction: uses a lexicon (a list of words and their possible variations) to correct errors in the recognized text by comparing it to the lexicon and suggesting alternative words where there are errors.
- Machine-learning-based approaches: uses machine learning algorithms, such as decision trees, random forests, and support vector machines, to correct errors in the recognized text.
- Deep-learning-based approaches: uses deep learning algorithms, such as CNNs and RNNs, to correct errors in the recognized text.
- Text enhancement: includes techniques to improve the recognized text's visibility, legibility, and readability, such as binarization, deskewing and smoothing of text.
- Text restoration: includes techniques to recover missing or degraded text, such as text in-painting, completion, and restoration.

The authors of [98] also present the results of various experiments and evaluations conducted to assess the performance of postprocessing techniques. These include comparisons of different methods and systems, evaluations of the effects of different types of

features and preprocessing techniques, and evaluations of the performance of systems on different types of OCR output and languages.

Doush et al. [100] present a word-context and rule-based technique for OCR postprocessing. OCR is used to obtain text from scanned documents, and the output text is not always 100% accurate. Thus, after obtaining the text, postprocessing step is required to recognize and minimize the errors. The presented research is about printed documents, it lies in an offline OCR system. Cursive nature also causes problems in this step because characters in Arabic are connected, and there are other additional things such as diacritics and different shapes of each character in different words. These things bring more complications to this step. A very small amount of work has been done on the Arabic postprocessing technique because it shows a very high character and word error rate after recognition. Therefore, it is a less attractive side for researchers. In the proposed research, an Arabic text database is prepared, available in three formats, i.e., HTML, PDF, and scanned-document images. The database has 4581 files, and there are about 8994 scanned images. Thus, from the database, 1000 images are used for training by the rule-based method, which reduces the word error rate.

Bassil and Alwani propose an algorithm using Google's online spelling suggestions [101], which helps to improve the accuracy and suggest what should come after each character to make a meaningful word according to the sentence. All of the suggestions given by Google's spelling suggestion algorithm are based on N-gram probability. The authors hybridize this method to make the proposed postprocessing technique. Therefore, the proposed hybrid postprocessing system starts with the generated OCR file (token). The language model checks each token, and if the language model does not find this token, then the error model takes this token and suggests the correct word. This token/word again goes into the language model in an attempt to find matches for the token. If it matches, then the model moves to the next word; otherwise, the error model again suggests the new token provided by Google's spelling suggestion algorithm, as shown in Figure 8.

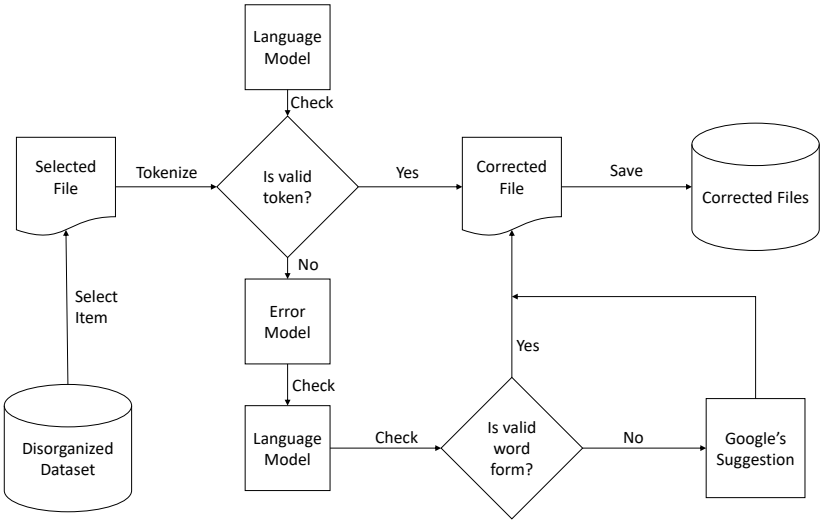

Figure 8. Hybrid postprocessing technique based on Google's spelling suggestion algorithm.

The authors of [102] present a corpus-based technique for improving the performance of an Arabic OCR system. The method involves using a large corpus of texts in Arabic to train the OCR system and improve its recognition accuracy. The corpus of texts is preprocessed to ensure that it is suitable for OCR training, and then it is used to train the OCR system. The performance of the OCR system is then evaluated using a set of test images, and the recognition accuracy is compared to that of traditional methods. The study also shows that using a larger corpus of texts leads to better performance of the OCR system. This phase helps to provide better recognition or reduce the word error rate and character error rate. The collection of a large corpus of texts in Arabic is used to train the OCR system

and then preprocess the corpus to ensure that it is suitable for OCR training. The OCR system is trained using the preprocessed corpus, and the OCR system's performance is evaluated using a set of test images. The research shows that the corpus-based technique improves the recognition accuracy of the OCR system by a significant margin compared to traditional methods. It also shows that using a larger corpus of texts leads to better performance of the OCR system.

3.5. Evaluation

Evaluation measures the accuracy and quality of the recognized text output. OCR evaluation typically involves comparing the recognized text to the original input image or document and calculating various performance metrics to assess the quality of the OCR output. Some common metrics used in OCR evaluation include character error rate (CER), word error rate (WER), recall, precision, and F1 score. OCR evaluation can be performed using various methods depending on the goal of the application, such as manual inspection, crowdsourcing, or automated evaluation software. Evaluation is an important step in OCR development and deployment, as it allows developers and users to assess the accuracy and quality of the OCR output and make improvements or adjustments to the OCR system as needed.

An Arabic OCR evaluation tool is discussed in [103]. The Arabic language is difficult to recognize due to its characters'/alphabets' behavior in different words. Arabic OCR's accuracy could be higher because of improper evaluation of performance metrics. Different tools and software have been introduced to help find the performance and accuracy of the applied algorithms. The introduced software is built specially to check Arabic OCR; it checks the performance based on objectives, i.e., accuracy and evaluation metrics. These tools briefly describe the text in characters with or without dots, baseline, and diacritics and the class in which the text lies. These tools include Tesseract, easy OCR, Paddle-Paddle OCR, and PyMuPdf. Recognition rate (RR), character error rate (CER), and word error rate (WER) are the evaluation measures used by these programs to evaluate OCR output.

Kiessling et al. [104] discuss various open-source tools for Arabic OCR systems containing Tesseract (https://github.com/tesseract-ocr/tesseract, accessed on 24 February 2023), OCRad (https://www.gnu.org/software/ocrad/, accessed on 24 February 2023), and GOCR (https://jocr.sourceforge.net/, accessed on 24 February 2023). These tools are evaluated using an Arabic dataset, and Tesseract gives better recognition results. However, it is slower than other evaluated tools. These tools give better accuracy for high-resolution documents, and the accuracy gets worse as the quality of documents decreases.

The authors of [105] provide an overview of existing tools and metrics used to evaluate the OCR system in previous research. Their paper covers traditional evaluation techniques and discusses their pros and cons. It also discusses evaluation metrics such as character error rate, word error rate, and recognition rate. The detailed review also discusses the many challenges involved in evaluating the performance of the OCR system.

The performances of generative and discriminative recognition models for offline Arabic handwritten recognition are compared using generatively trained hidden Markov modeling (HMM), discriminatively trained conditional random fields (CRF), and discriminatively trained hidden-state CRF (HCRF) in [106]. The study presents recognition outcomes for words and letters and assesses the efficiency of all three strategies using the Arabic IFN/ENIT dataset.

Singh et al. [107] discussed offline handwritten word recognition in Devanagari. A holistic-based approach is used in this approach, wherein a word is considered a single entity, and the approach processes it further for extraction and recognition. A class of 50 words recognizes every word based on a feature vector set, uniform zoning, diagonal, centroid, and feature-based. The proposed system uses gradient-boosted algorithms to enhance the performance; furthermore, some other classifiers are used for this purpose, i.e., *k*NN and Random Forest Classifier. Thus, the overall achieved accuracy is 94.53%. The authors generate the dataset during the implementation, which is available on request. The

paper also provided some information on previous research in this area, where researchers used Hidden Markov Model, Support Vector Machine, and Multi-Layer perceptron classifiers. The authors also highlighted the importance of the availability of quality datasets for the improved performance of OCR techniques.

The impact of OCR quality on the accuracy of short text classification tasks is presented in [108]. A multi-class classification of short text is introduced. For this, the authors propose a dataset of beauty product images that contains 27,500 entries of labeled brand data and generate results based on targeting specific brands. The authors also show that preprocessing techniques such as text normalization and noise reduction can improve the performance of the classification model on low-quality OCR text.

In addition to papers improving specific processes involved in OCR, some previous papers present a combination of various OCR techniques for overall improved process accuracy. OCR4all [109] is an open-source OCR software that combines state-of-the-art OCR components and continuous model training into a comprehensive workflow for processing historical printings and provides a user-friendly GUI and extensive configuration capabilities. The software outperforms commercial tools on moderate layouts. It achieves excellent character error rates (CER) on very complex early printed books, making it a valuable tool for non-technical users and providing an architecture allowing easy integration of newly developed tools.

3.6. Summary of Presented Techniques

This survey aims to provide a comprehensive literature review of the various techniques involved in Arabic OCR and to provide valuable insights into the current state-of-the-art in Arabic OCR. To achieve this goal, we analyzed a range of research papers and articles that focused on different Arabic OCR techniques and methods. Our findings are summarized in Table 5, which presents a concise overview of the methods employed in the reviewed papers. The table outlines the different OCR techniques, including preprocessing, segmentation, recognition, and postprocessing, along with their performance evaluations in terms of accuracy using various types of printed, scanned, and handwritten datasets. This survey is useful for researchers and practitioners who are interested in Arabic OCR systems. By comparing and contrasting the different techniques in the complete pipeline of the Arabic OCR, they can choose the most appropriate one for their specific task. To further aid researchers and practitioners, Table 6 presents various methods commonly employed in OCR systems and their respective advantages and disadvantages.

Table 5. A brief tabular outline of the described papers with the proposed OCR techniques and their performance evaluations.

OCR Techniques	OCR Tasks					Accuracy
	Preprocessing	Segmentation	Recognition	Postprocessing	Evaluation	
Ahmad et al. [63]	✓	✓	✓			99.3% (Scanned)
Bafjaish et al. [7]	✓	✓	✓			90% (Scanned)
Karthick et al. [59]	✓	✓	✓	✓		87.4% (Handwritten), 90% (Scanned)
Abdo et al. [67]	✓	✓	✓		✓	94.1% (Printed)
Qaroush et al. [70]	✓	✓			✓	11% (Segmentation)
Tayyab et al. [77]	✓	✓	✓		✓	98.36% (Scanned)
Alginahi [78]		✓	✓			93.65% (Handwritten), 86.14% (Scanned)
Verma and Ali [82]	✓	✓				No Recognition
Hamida et al. [83]	✓		✓		✓	99.88% (Handwritten)
Butt et al. [86]	✓		✓		✓	87% (Scanned)
Nguyen et al. [98]				✓	✓	No Recognition
Doush et al. [100]				✓	✓	No Recognition
Neudecker et al. [105]					✓	No Recognition
Vitman et al. [108]	✓		✓		✓	83.5% (High-quality), 58.4%(Low-quality)

Table 6. Pros and cons of various recognition methods for Arabic OCR.

Method	Pros	Cons
Template matching	Simple and easy to implement	Limited accuracy, sensitive to noise and variations in text
Deep learning	High accuracy, can handle variations in text	Requires large amounts of training data, computationally expensive
*k*NN	For small datasets, takes less training time and make predictions quickly	Sensitive to noisy or irrelevant features
RNN	For processing large sequential data and can learn term dependencies	Computationally expensive and sensitive to overfitting
Hough Transformation	Robust to noise and can detect lines and circles at any orientation	Computationally expensive when dealing with large images
Histogram Oriented Gradient	Extracts features such as edge orientation and texture, and is computed quickly	Ineffective at detecting finer details and is sensitive to variations in lighting and contrast
Hidden Markov Model	Models complex patterns and can be trained on large/sequential datasets	Computationally expensive to train and sensitive to model parameters
Profile Projection	Extracts features from images, such as character width and spacing	Sensitive to variations in lighting and contrast.
Random Forest	Relatively easy to train and can handle noisy or missing data	Does not perform well on highly imbalanced or sparse datasets
SVM	Used for classification tasks and can handle high-dimensional data	Computationally expensive non-linear kernels require hyperparameter tuning
Hybrid approaches	Combines the strengths of multiple methods	More complex and difficult to implement

4. Discussion and Conclusions

We surveyed that researchers use multiple approaches and datasets to get better recognition rates. The literature review suggests that a proper process needs to be followed, which includes preprocessing, segmentation (text-area detection and line, word, and character segmentation), recognition, and postprocessing. We discussed the pros and cons of each technique discussed in this survey. For example, segmentation-based approaches give better results than segmentation-free approaches, and vertical/horizontal projection produces good results for word and character segmentation. However, OCR results depend upon a good dataset as well. Some datasets are available for the Arabic OCR, but only a few are publicly available. Postprocessing and dataset availability require more attention from researchers. For postprocessing, if Google spelling checker-like algorithms are implemented and improved, then this stage can perform very well, enhancing the overall result of the OCR system. We need a publicly available dataset with an extensive vocabulary of printed and handwritten text (characters and words) for the dataset.

In conclusion, we presented a survey of the state-of-the-art Arabic OCR, which has come a long way in recent years, with several approaches and techniques developed to improve its accuracy and performance. However, many challenges still need to be addressed, including dealing with the variability and complexity of the Arabic script and the large number of dialects and variations in the language. Despite these challenges, the potential benefits of Arabic OCR are clear, and researchers and developers are working hard to continue to improve and refine the technology. We will likely see even more accurate and reliable Arabic OCR systems with continued research and development.

Author Contributions: Conceptualization, S.F., M.S.A. and S.H.; methodology, S.H., M.S.A. and M.A.K.; investigation, M.S.A. and S.H.; resources, M.S.A. and S.H.; writing—original draft preparation, M.S.A. and S.H.; writing—review and editing, S.F., M.S.A., S.H. and M.A.K.; visualization, M.S.A., S.H. and M.A.K.; supervision, S.F. and M.A.K.; project administration, S.F. and M.A.K.; funding acquisition, S.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by the Deputyship of Research & Innovation, Ministry of Education, Saudi Arabia, through project number 964. In addition, the authors would like to express their appreciation for the support provided by the Islamic University of Madinah.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available data sources were used for this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alhomed, L.S.; Jambi, K.M. A survey on the existing arabic optical character recognition and future trends. *Int. J. Adv. Res. Comput. Commun. Eng. (IJARCCE)* **2018**, *7*, 78–88.
2. Beg, A.; Ahmed, F.; Campbell, P. Hybrid OCR techniques for cursive script languages-a review and applications. In Proceedings of the International Conference on Computational Intelligence, Communication Systems and Networks, Liverpool, UK, 28–30 July 2010; pp. 101–105.
3. Djaghbellou, S.; Bouziane, A.; Attia, A.; Akhtar, Z. A Survey on Arabic Handwritten Script Recognition Systems. *Int. J. Artif. Intell. Mach. Learn. (IJAIML)* **2021**, *11*, 1–17. [CrossRef]
4. Islam, N.; Islam, Z.; Noor, N. A survey on optical character recognition system. *arXiv* **2017**, arXiv:1710.05703.
5. Rashid, D.; Kumar Gondhi, N. Scrutinization of Urdu Handwritten Text Recognition with Machine Learning Approach. In Proceedings of the International Conference on Emerging Technologies in Computer Engineering, Xiamen, China, 21–23 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 383–394.
6. Idrees, S.; Hassani, H. Exploiting Script Similarities to Compensate for the Large Amount of Data in Training Tesseract LSTM: Towards Kurdish OCR. *Appl. Sci.* **2021**, *11*, 9752. [CrossRef]
7. Bafjaish, S.S.; Azmi, M.S.; Al-Mhiqani, M.N.; Radzid, A.R.; Mahdin, H. Skew detection and correction of Mushaf Al-Quran script using hough transform. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*. [CrossRef]
8. Singh, A.; Bacchuwar, K.; Bhasin, A. A survey of OCR applications. *Int. J. Mach. Learn. Comput.* **2012**, *2*, 314. [CrossRef]
9. Antonio, J.; Putra, A.R.; Abdurrohman, H.; Tsalasa, M.S. A Survey on Scanned Receipts OCR and Information Extraction. In Proceedings of the International Conference on Document Analysis and Recognit, Jerusalem, Israel, 29–30 November 2022.
10. Al-Sheikh, I.S.; Mohd, M.; Warlina, L. A review of arabic text recognition dataset. *Asia-Pac. J. Inf. Technol. Multimed. (APJITM)* **2020**, *9*, 69–81. [CrossRef]
11. Ahmed, S.B.; Naz, S.; Swati, S.; Razzak, M.I. Handwritten Urdu character recognition using one-dimensional BLSTM classifier. *Neural Comput. Appl.* **2019**, *31*, 1143–1151. [CrossRef]
12. Zayene, O.; Masmoudi Touj, S.; Hennebert, J.; Ingold, R.; Essoukri Ben Amara, N. Open datasets and tools for arabic text detection and recognition in news video frames. *J. Imaging* **2018**, *4*, 32. [CrossRef]
13. Badry, M.; Hassan, H.; Bayomi, H.; Oakasha, H. QTID: Quran Text Image Dataset. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 385–391. [CrossRef]
14. Pechwitz, M.; Maddouri, S.S.; Märgner, V.; Ellouze, N.; Amiri, H. *IFN/ENIT-Database of Handwritten Arabic Words*; CIFED: Hammamet, Tunis, 2002; Volume 2, pp. 127–136.
15. Al-Ma'adeed, S.; Elliman, D.; Higgins, C.A. A data base for Arabic handwritten text recognition research. In Proceedings of the International workshop on frontiers in handwriting recognition, Niagara-on-the-Lake, ON, Canada, 6–8 August 2002; pp. 485–489.
16. Slimane, F.; Ingold, R.; Kanoun, S.; Alimi, A.M.; Hennebert, J. *Database and Evaluation Protocols for Arabic Printed Text Recognition*; DIUF-University of Fribourg: Fribourg, Switzerland, 2009; p. 1.
17. Lawgali, A.; Angelova, M.; Bouridane, A. HACDB: Handwritten Arabic characters database for automatic character recognition. In Proceedings of the European Workshop on Visual Information Processing (EUVIP), Paris, France, 10–12 June 2013; pp. 255–259.
18. Sabbour, N.; Shafait, F. A segmentation-free approach to Arabic and Urdu OCR. In Proceedings of the Document Recognition and Retrieval, San Jose, CA, USA, 16–20 January 2005; Volume 8658, pp. 215–226.
19. Saddami, K.; Munadi, K.; Arnia, F. A database of printed Jawi character image. In Proceedings of the International Conference on Image Information Processing (ICIIP), Waknaghat, India, 21–24 December 2015; pp. 56–59.
20. Mahmoud, S.A.; Ahmad, I.; Al-Khatib, W.G.; Alshayeb, M.; Parvez, M.T.; Märgner, V.; Fink, G.A. KHATT: An open Arabic offline handwritten text database. *Pattern Recognit.* **2014**, *47*, 1096–1112. [CrossRef]
21. Yousfi, S.; Berrani, S.A.; Garcia, C. ALIF: A dataset for Arabic embedded text recognition in TV broadcast. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 1221–1225.
22. Zayene, O.; Hennebert, J.; Touj, S.M.; Ingold, R.; Amara, N.E.B. A dataset for Arabic text detection, tracking and recognition in news videos-AcTiV. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 996–1000.
23. Chabchoub, F.; Kessentini, Y.; Kanoun, S.; Eglin, V.; Lebourgeois, F. SmartATID: A mobile captured Arabic Text Images Dataset for multi-purpose recognition tasks. In Proceedings of the International Conference on Frontiers in Handwriting Recognition (ICFHR), Hyderabad, India, 4–7 December 2016; pp. 120–125.
24. Sulaiman, A.; Omar, K.; Nasrudin, M.F. A database for degraded Arabic historical manuscripts. In Proceedings of the International Conference on Electrical Engineering and Informatics (ICEEI), Langkawi, Malaysia, 25–27 November 2017; pp. 1–6.
25. Bataineh, B. A Printed PAW Image Database of Arabic Language for Document Analysis and Recognition. *J. ICT Res. Appl.* **2017**, *11*, 200–212. [CrossRef]

26. Al-Ohali, Y.; Cheriet, M.; Suen, C. Databases for recognition of handwritten Arabic cheques. *Pattern Recognit.* **2003**, *36*, 111–121. [CrossRef]
27. Awaidah, S.M.; Mahmoud, S.A. A multiple feature/resolution scheme to Arabic (Indian) numerals recognition using hidden Markov models. *Signal Process.* **2009**, *89*, 1176–1184. [CrossRef]
28. Asiri, A.M.; Khorsheed, M.S. Automatic Processing of Handwritten Arabic Forms using Neural Networks. In Proceedings of the IEC (Prague), Prague, Czech Republic, 26–28 August 2005; pp. 313–317.
29. Luqman, H.; Mahmoud, S.A.; Awaida, S. KAFD Arabic font database. *Pattern Recognit.* **2014**, *47*, 2231–2240. [CrossRef]
30. Ramdan, J.; Omar, K.; Faidzul, M.; Mady, A. Arabic handwriting data base for text recognition. *Procedia Technol.* **2013**, *11*, 580–584. [CrossRef]
31. Amara, N.E.B.; Mazhoud, O.; Bouzrara, N.; Ellouze, N. ARABASE: A Relational Database for Arabic OCR Systems. *Int. Arab J. Inf. Technol.* **2005**, *2*, 259–266.
32. Srihari, S.; Srinivasan, H.; Babu, P.; Bhole, C. Handwritten arabic word spotting using the cedarabic document analysis system. In Proceedings of the Symposium on Document Image Understanding Technology (SDIUT-05), College Park, MD, USA, 2–4 November 2005; pp. 123–132.
33. Shafi, M.; Zia, K. Urdu character recognition: A systematic literature review. *Int. J. Appl. Pattern Recognit.* **2021**, *6*, 283–307. [CrossRef]
34. Khan, N.H.; Adnan, A. Urdu optical character recognition systems: Present contributions and future directions. *IEEE Access* **2018**, *6*, 46019–46046. [CrossRef]
35. Bhatti, A.; Arif, A.; Khalid, W.; Khan, B.; Ali, A.; Khalid, S.; Rehman, A.u. Recognition and Classification of Handwritten Urdu Numerals Using Deep Learning Techniques. *Appl. Sci.* **2023**, *13*, 1624. [CrossRef]
36. Khosrobeigi, Z.; Veisi, H.; Hoseinzade, E.; Shabanian, H. Persian Optical Character Recognition Using Deep Bidirectional Long Short-Term Memory. *Appl. Sci.* **2022**, *12*, 11760. [CrossRef]
37. Husnain, M.; Saad Missen, M.M.; Mumtaz, S.; Coustaty, M.; Luqman, M.; Ogier, J.M. Urdu handwritten text recognition: A survey. *IET Image Process.* **2020**, *14*, 2291–2300. [CrossRef]
38. Naz, S.; Hayat, K.; Razzak, M.I.; Anwar, M.W.; Madani, S.A.; Khan, S.U. The optical character recognition of Urdu-like cursive scripts. *Pattern Recognit.* **2014**, *47*, 1229–1248. [CrossRef]
39. Alghamdi, M.; Teahan, W. Printed Arabic script recognition: A survey. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 415–427. [CrossRef]
40. Osman, H.; Zaghw, K.; Hazem, M.; Elsehely, S. An Efficient Language-Independent Multi-Font OCR for Arabic Script. *arXiv* **2020**, arXiv:2009.09115.
41. Muhammad, M.; ElGhazaly, T. Handling OCR-degraded arabic text: A comprehensive survey. In Proceedings of the ISSR Conference, Turku, Finland, 27–30 June 2013.
42. Dinges, L.; Al-Hamadi, A.; Elzobi, M.; El-Etriby, S. Synthesis of common Arabic handwritings to aid optical character recognition research. *Sensors* **2016**, *16*, 346. [CrossRef]
43. Bouressace, H. A Review of Arabic Document Analysis Methods. In Proceedings of the International Conference on Pattern Analysis and Intelligent Systems (PAIS), Oum El Bouaghi, Algeria, 12–13 October 2022; pp. 1–7.
44. Qaroush, A.; Jaber, B.; Mohammad, K.; Washaha, M.; Maali, E.; Nayef, N. An efficient, font independent word and character segmentation algorithm for printed Arabic text. *J. King Saud-Univ.-Comput. Inf. Sci.* **2022**, *34*, 1330–1344. [CrossRef]
45. Al Ghamdi, M.A. A Novel Approach to Printed Arabic Optical Character Recognition. *Arab. J. Sci. Eng.* **2022**, *47*, 2219–2237. [CrossRef]
46. Majumdar, S.; Brick, A. Recognizing Handwriting Styles in a Historical Scanned Document Using Scikit-Fuzzy c-means Clustering. *arXiv* **2022**, arXiv:2210.16780.
47. Mostafa, A.; Mohamed, O.; Ashraf, A.; Elbehery, A.; Jamal, S.; Khoriba, G.; Ghoneim, A.S. OCFormer: A Transformer-Based Model For Arabic Handwritten Text Recognition. In Proceedings of the International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC), Cairo, Egypt, 26–27 May 2021; pp. 182–186.
48. Badry, M.; Hassanin, M.; Chandio, A.; Moustafa, N. Quranic script optical text recognition using deep learning in IoT systems. *CMC-Comput. Mater. Contin.* **2021**, *68*, 1847–1858. [CrossRef]
49. Moudgil, A.; Singh, S.; Gautam, V. An Overview of Recent Trends in OCR Systems for Manuscripts. In *Cyber Intelligence and Information Retrieval*; Springer: Berlin, Germany, 2022; pp. 525–533.
50. Huang, Z.; Chen, K.; He, J.; Bai, X.; Karatzas, D.; Lu, S.; Jawahar, C. Icdar2019 competition on scanned receipt ocr and information extraction. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019; pp. 1516–1520.
51. Bashir, M.H.; Azmi, A.M.; Nawaz, H.; Zaghouani, W.; Diab, M.; Al-Fuqaha, A.; Qadir, J. Arabic natural language processing for Qur'anic research: A systematic review. *Artif. Intell. Rev.* **2022**. [CrossRef]
52. Gupta, M.R.; Jacobson, N.P.; Garcia, E.K. OCR binarization and image pre-processing for searching historical documents. *Pattern Recognit.* **2007**, *40*, 389–397. [CrossRef]
53. Michalak, H.; Okarma, K. Robust combined binarization method of non-uniformly illuminated document images for alphanumerical character recognition. *Sensors* **2020**, *20*, 2914. [CrossRef] [PubMed]
54. Tellache, M.; Sid-Ahmed, M.; Abaza, B. Thinning algorithms for Arabic OCR. In Proceedings of the Pacific Rim Conference on Communications Computers and Signal Processing, Victoria, BC, Canada, 19–21 May 1993; Volume 1, pp. 248–251.

55. Mohsenzadegan, K.; Tavakkoli, V.; Kyamakya, K. Deep Neural Network Concept for a Blind Enhancement of Document-Images in the Presence of Multiple Distortions. *Appl. Sci.* **2022**, *12*, 9601. [CrossRef]
56. Mahmud, J.U.; Raihan, M.F.; Rahman, C.M. A complete OCR system for continuous Bengali characters. In Proceedings of the Conference on Convergent Technologies for Asia-Pacific Region (TENCON), Bangalore, India, 15–17 October 2003; Volume 4, pp. 1372–1376.
57. Mohsenzadegan, K.; Tavakkoli, V.; Kyamakya, K. A Smart Visual Sensing Concept Involving Deep Learning for a Robust Optical Character Recognition under Hard Real-World Conditions. *Sensors* **2022**, *22*, 6025. [CrossRef]
58. Nashwan, F.M.; Rashwan, M.A.; Al-Barhamtoshy, H.M.; Abdou, S.M.; Moussa, A.M. A holistic technique for an Arabic OCR system. *J. Imaging* **2017**, *4*, 6. [CrossRef]
59. Karthick, K.; Ravindrakumar, K.; Francis, R.; Ilankannan, S. Steps involved in text recognition and recent research in OCR; a study. *Int. J. Recent Technol. Eng.* **2019**, *8*, 2277–3878.
60. Cao, Y.; Wang, S.; Li, H. Skew detection and correction in document images based on straight-line fitting. *Pattern Recognit. Lett.* **2003**, *24*, 1871–1879. [CrossRef]
61. Bao, W.; Yang, C.; Wen, S.; Zeng, M.; Guo, J.; Zhong, J.; Xu, X. A Novel Adaptive Deskewing Algorithm for Document Images. *Sensors* **2022**, *22*, 7944. [CrossRef]
62. Boiangiu, C.A.; Dinu, O.A.; Popescu, C.; Constantin, N.; Petrescu, C. Voting-based document image skew detection. *Appl. Sci.* **2020**, *10*, 2236. [CrossRef]
63. Ahmad, R.; Naz, S.; Razzak, I. Efficient skew detection and correction in scanned document images through clustering of probabilistic hough transforms. *Pattern Recognit. Lett.* **2021**, *152*, 93–99. [CrossRef]
64. Li, Y.; Zou, F.; Yang, S.; Liu, H.; Ding, Y.; Zhu, K. Research on Improving OCR Recognition Based on Bending Correction. In Proceedings of the International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 11–13 December 2020; Volume 9, pp. 833–837.
65. Schulter, S.; Leistner, C.; Bischof, H. Fast and accurate image upscaling with super-resolution forests. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3791–3799.
66. Pandey, R.K.; Vignesh, K.; Ramakrishnan, A. Binary document image super resolution for improved readability and OCR performance. *arXiv* **2018**, arXiv:1812.02475.
67. Abdo, H.A.; Abdu, A.; Manza, R.R.; Bawiskar, S. An approach to analysis of Arabic text documents into text lines, words, and characters. *Indones. J. Electr. Eng. Comput. Sci.* **2022**, *26*, 754–763. [CrossRef]
68. Naz, S.; Umar, A.I.; Shirazi, S.H.; Ahmed, S.B.; Razzak, M.I.; Siddiqi, I. Segmentation techniques for recognition of Arabic-like scripts: A comprehensive survey. *Educ. Inf. Technol.* **2016**, *21*, 1225–1241. [CrossRef]
69. Thorat, C.; Bhat, A.; Sawant, P.; Bartakke, I.; Shirsath, S. A Detailed Review on Text Extraction Using Optical Character Recognition. *ICT Anal. Appl.* **2022**, 719–728.. [CrossRef]
70. Qaroush, A.; Awad, A.; Hanani, A.; Mohammad, K.; Jaber, B.; Hasheesh, A. Learning-free, divide and conquer text-line extraction algorithm for printed Arabic text with diacritics. *J. King Saud-Univ.-Comput. Inf. Sci.* **2022**, *34*, 7699–7709. [CrossRef]
71. Brodic, D.; Milivojevic, D.R.; Milivojevic, Z.N. An approach to a comprehensive test framework for analysis and evaluation of text line segmentation algorithms. *Sensors* **2011**, *11*, 8782–8812. [CrossRef]
72. Brodić, D.; Milivojević, D.R.; Milivojević, Z. Basic test framework for the evaluation of text line segmentation and text parameter extraction. *Sensors* **2010**, *10*, 5263–5279. [CrossRef]
73. Reisswig, C.; Katti, A.R.; Spinaci, M.; Höhne, J. Chargrid-OCR: End-to-end trainable optical character recognition through semantic segmentation and object detection. In Proceedings of the Workshop on Document Intelligence at NeurIPS 2019, Vancouver, BC, Canada, 14 December 2019.
74. Agarwal, M.; Hassan, F.; Pandey, G.; Ghosh, S. Handwriting recognition using deep learning. In *Emerging Trends in Data Driven Computing and Communications: Proceedings of DDCIoT 2021*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 67–81.
75. Boualam, M.; Elfakir, Y.; Khaissidi, G.; Mrabti, M. Arabic handwriting word recognition based on convolutional recurrent neural network. In Proceedings of the 6th International Conference on Wireless Technologies, Embedded, and Intelligent Systems (WITS 2020), Fez, Morocco, 14–16 October 2020; Springer: Berlin/Heidelberg, Germany, 2022; pp. 877–885.
76. Patil, S.; Varadarajan, V.; Mahadevkar, S.; Athawade, R.; Maheshwari, L.; Kumbhare, S.; Garg, Y.; Dharrao, D.; Kamat, P.; Kotecha, K. Enhancing Optical Character Recognition on Images with Mixed Text Using Semantic Segmentation. *J. Sens. Actuator Netw.* **2022**, *11*, 63. [CrossRef]
77. Tayyab, M.; Hussain, A.; Alshara, M.A.; Khan, S.; Alotaibi, R.M.; Baig, A.R. Recognition of Visual Arabic Scripting News Ticker from Broadcast Stream. *IEEE Access* **2022**, *10*, 59189–59204. [CrossRef]
78. Alginahi, Y.M. A survey on Arabic character segmentation. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2013**, *16*, 105–126. [CrossRef]
79. Boraik, O.A.; Ravikumar, M.; Saif, M.A.N. Characters Segmentation from Arabic Handwritten Document Images: Hybrid Approach. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 395–403. [CrossRef]
80. AbdAllah, N.; Viriri, S. Off-Line Arabic Handwritten Words Segmentation using Morphological Operators. *arXiv* **2021**, arXiv:2101.02797.
81. Jabde, M.; Patil, C.; Mali, S.; Vibhute, A. Comparative Study of Machine Learning and Deep Learning Classifiers on Handwritten Numeral Recognition. In Proceedings of the International Symposium on Intelligent Informatics, Trivandrum, India, 31 August–2 September 2022.

82. Verma, R.; Ali, J. A-survey of feature extraction and classification techniques in OCR systems. *Int. J. Comput. Appl. Inf. Technol.* **2012**, *1*, 1–3.
83. Hamida, S.; El Gannour, O.; Cherradi, B.; Ouajji, H.; Raihani, A. Efficient feature descriptor selection for improved Arabic handwritten words recognition. *Int. J. Electr. Comput. Eng.* **2022**, *12*. [CrossRef]
84. Peng, X.; Cao, H.; Setlur, S.; Govindaraju, V.; Natarajan, P. Multilingual OCR research and applications: An overview. In Proceedings of the International Workshop on Multilingual OCR, Washington, DC, USA, 24 August 2013; pp. 1–8.
85. Bergamaschi, S.; De Nardis, S.; Martoglia, R.; Ruozzi, F.; Sala, L.; Vanzini, M.; Vigliermo, R.A. Novel perspectives for the management of multilingual and multialphabetic heritages through automatic knowledge extraction: The digitalmaktaba approach. *Sensors* **2022**, *22*, 3995. [CrossRef]
86. Butt, H.; Raza, M.R.; Ramzan, M.J.; Ali, M.J.; Haris, M. Attention-based CNN-RNN Arabic text recognition from natural scene images. *Forecasting* **2021**, *3*, 520–540. [CrossRef]
87. Al-Barhamtoshy, H.M.; Jambi, K.M.; Rashwan, M.A.; Abdou, S.M. An Arabic Manuscript Regions Detection, Recognition and Its Applications for OCRing. *Trans. Asian-Low-Resour. Lang. Inf. Process.* **2023**, *22*, 1–28. [CrossRef]
88. Chen, X.; Jin, L.; Zhu, Y.; Luo, C.; Wang, T. Text recognition in the wild: A survey. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–35. [CrossRef]
89. Bouchakour, L.; Meziani, F.; Latrache, H.; Ghribi, K.; Yahiaoui, M. Printed Arabic Characters Recognition Using Combined Features and CNN classifier. In Proceedings of the International Conference on Recent Advances in Mathematics and Informatics (ICRAMI), Tebessa, Algeria, 21–22 September 2021; pp. 1–5.
90. Ahlawat, S.; Choudhary, A.; Nayyar, A.; Singh, S.; Yoon, B. Improved handwritten digit recognition using convolutional neural networks (CNN). *Sensors* **2020**, *20*, 3344. [CrossRef]
91. Ashraf, N.; Arafat, S.Y.; Iqbal, M.J. An Analysis of Optical Character Recognition (OCR) Methods. *Int. J. Comput. Linguist. Res.* **2019**, *10*, 81. [CrossRef]
92. Al-Sadawi, B.; Hussain, A.; Ali, N.S. High-Performance Printed Arabic Optical Character Recognition System Using ANN Classifier. In Proceedings of the Palestinian International Conference on Information and Communication Technology, Gaza, Palestine, 28–29 September 2021; IEEE Computer Society: Colombia, DC, USA, 2021; pp. 1–6.
93. Mittal, R.; Garg, A. Text extraction using OCR: A systematic review. In Proceedings of the International Conference on Inventive Research in Computing Applications, Coimbatore, India, 15–17 July 2020; pp. 357–362.
94. Alrobah, N.; Albahli, S. Arabic handwritten recognition using deep learning: A survey. *Arab. J. Sci. Eng.* **2022**, *47*, 9943–9963. [CrossRef]
95. Alwaqfi, Y.M.; Mohamad, M.; Al-Taani, A.T. Generative Adversarial Network for an Improved Arabic Handwritten Characters Recognition. *Int. J. Adv. Soft Comput. Its Appl.* **2022**, *14*, 176–195. [CrossRef]
96. Hamad, K.; Mehmet, K. A detailed analysis of optical character recognition technology. *Int. J. Appl. Math. Electron. Comput.* **2016**, *1*, 244–249. [CrossRef]
97. Subramani, N.; Matton, A.; Greaves, M.; Lam, A. A survey of deep learning approaches for ocr and document understanding. *arXiv* **2020**, arXiv:2011.13534.
98. Nguyen, T.T.H.; Jatowt, A.; Coustaty, M.; Doucet, A. Survey of post-ocr processing approaches. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–37. [CrossRef]
99. Neto, A.F.d.S.; Bezerra, B.L.D.; Toselli, A.H. Towards the natural language processing as spelling correction for offline handwritten text recognition systems. *Appl. Sci.* **2020**, *10*, 7711. [CrossRef]
100. Doush, I.A.; Alkhateeb, F.; Gharaibeh, A.H. A novel Arabic OCR post-processing using rule-based and word context techniques. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2018**, *21*, 77–89. [CrossRef]
101. Bassil, Y.; Alwani, M. Ocr post-processing error correction algorithm using google online spelling suggestion. *arXiv* **2012**, arXiv:1204.0191.
102. Aliwy, A.H.; Al-Sadawi, B. Corpus-based technique for improving Arabic OCR system. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *21*, 233–241. [CrossRef]
103. Alghamdi, M.A.; Alkhazi, I.S.; Teahan, W.J. Arabic OCR evaluation tool. In Proceedings of the International conference on computer science and information technology (CSIT), Amman, Jordan, 13–14 July 2016; pp. 1–6.
104. Kiessling, B.; Kurin, G.; Miller, M.T.; Smail, K.; Miller, M. Advances and Limitations in Open Source Arabic-Script OCR: A Case Study. *Digit. Stud. Champ NumÉRique* **2021**, *11*. [CrossRef]
105. Neudecker, C.; Baierer, K.; Gerber, M.; Clausner, C.; Antonacopoulos, A.; Pletschacher, S. A survey of OCR evaluation tools and metrics. In Proceedings of the International Workshop on Historical Document Imaging and Processing, Lausanne, Switzerland, 5–10 September 2021; pp. 13–18.
106. Elzobi, M.; Al-Hamadi, A. Generative vs. Discriminative Recognition Models for Off-Line Arabic Handwriting. *Sensors* **2018**, *18*, 2786. [CrossRef]
107. Singh, S.; Garg, N.K.; Kumar, M. On the performance analysis of various features and classifiers for handwritten devanagari word recognition. *Neural Comput. Appl.* **2023**, *35*, 7509–7527. [CrossRef]
108. Vitman, O.; Kostiuk, Y.; Plachinda, P.; Zhila, A.; Sidorov, G.; Gelbukh, A. Evaluating the Impact of OCR Quality on Short Texts Classification Task. In Proceedings of the Mexican International Conference on Artificial Intelligence, Monterrey, Mexico, 24–29 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 163–177.

109. Reul, C.; Christ, D.; Hartelt, A.; Balbach, N.; Wehner, M.; Springmann, U.; Wick, C.; Grundig, C.; Büttner, A.; Puppe, F. OCR4all—An open-source tool providing a (semi-) automatic OCR workflow for historical printings. *Appl. Sci.* **2019**, *9*, 4853. [CrossRef]