

Received July 06, 2020, accepted July 17, 2020, date of publication xxxx 00, 0000, date of current version July 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.33854

# Manuscripts Image Retrieval Using Deep Learning Incorporating a Variety of Fusion Levels

**Manal M. Khayyat<sup>1,2</sup> and Lamiaa A. Elrefaei<sup>1,3</sup> (Senior Member, IEEE)**

<sup>1</sup> Computer Science Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, 21589, Saudi Arabia

<sup>2</sup> Computer Science Department, Deanship of Preparatory Year of the Joint Medical Track, Umm Al-Qura University, Makkah, 21955, Saudi Arabia

<sup>3</sup> Electrical Engineering Department, Faculty of Engineering at Shoubra, Benha University, Cairo, 11629, Egypt

Corresponding author: Manal M. Khayyat (e-mail: mkhayyat0012@stu.kau.edu.sa).

This research was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under grant No. (DG-047-611-1440).

**ABSTRACT** The instantaneous search and retrieval of the most relevant images to a specific query image is a desirable application for all digital libraries. The automatic extraction and classification according to the most distinguishable features, is a crucial step to detect the similarities among images successfully. This study introduces a novel approach that utilizes a fusion model for classifying and retrieving historical Arabic manuscripts' images. To accomplish our goal, the images are first classified according to their extracted deep learning visual features utilizing a pre-trained convolutional neural network. Then, the texts written in the manuscripts' images are extracted and pre-processed to classify the images according to their textual features using an optimized bidirectional LSTM deep learning model with attention and batch normalization layers. Finally, both the visual and textual deep learning models are fused at three different fusion-levels named: decision-level, features-level, and score-level. The score-level fusion model resulted in a considerable improvement of each model used individually. Extensive experimentation and evaluation of the proposed fusion method on the collected ancient Arabic manuscripts dataset proved its robustness against other state-of-the-art methods recording 99% classification accuracy and 98% mean accuracy on the top-10 image retrieval.

**INDEX TERMS** Image retrieval, fusion methods, similarity measurement, Deep Learning (DL), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM).

## I. INTRODUCTION

Within recent years, there has been a lot of attempts focusing on retrieving images. Image retrieval is a significant field within Computer vision. It allows the huge number of daily created digital images to be available to tremendous online users [1]. Studies have addressed image retrieval in many various ways. Some studies focused purely on classifying and extracting the images' visual features using the Convolutional Neural Networks (CNN). While other studies focused on segmenting and extracting the textual features from text-images such as the structural and the statistical features.

Recently, many researchers implemented deep learning technology for image retrieval applications and confessed its rigidity and robustness, especially when dealing with large datasets. Because the deep learning technique simulates the humans' brains in visualizing and distinguishing between the

images [2]. Thus, the trend in most recent research papers is toward deep learning technology instead of machine learning and other classical algorithms. For instance, a well-known deep learning model used in Natural Language Processing (NLP) is the Recurrent Neural Networks (RNN). It is named recurrent because it executes sequential elements in an identical manner, whereas the generated output is depending on the preceding execution [3]. RNN has been used excessively in NLP applications because they inspect sequential information. Thus, they can achieve great results in predicting the next words in a sentence. There is also the Long Short-Term Memory (LSTM), which presents a special type of RNN. They were initially proposed by Hochreiter & Schmidhuber [4], and later on, they were modified and enhanced by many other researchers. LSTM differs from RNN in that they can refer longer than RNN. Therefore, the

LSTM can connect large gaps and present their related information [3].

Even though the deep learning technique proved its rigidity in solving many problems, it is crucial to carefully choose the correct type of the deep neural network according to the problem to be solved. For example, the LSTM deep neural networks work well with extracting the text-based features, but they don't perform well with the visual-based features. Whereas, the CNN deep neural networks are very powerful in extracting and processing the images visual-based features. Alayba et al. [5] admit that CNN can extract the best visual features from images, while RNN-LSTM can learn sequential data. Moreover, Al-Muzaini et al. [6] claim that the CNN-RNN integrated methods guarantee the successful retrieval of images.

### A. CHALLENGES ASSOCIATED WITH THE HISTORICAL ARABIC TEXT-BASED IMAGES

Many image retrieval systems employ handwritten text-images as their inputs. Thus, training the machine to recognize and process handwritten texts has been an essential concern for many researchers. They need to preprocess and segment the text into paragraphs, lines, sentences, words, or even characters. Because after the segmentation process, it is much easier to understand and to handle the segmented parts of the document for further analysis.

There are many traditional methods for text-based image retrieval, such as word spotting, text queries, segmentation, and Optical Characters Recognition (OCR). However, Alaei, et al. [7] believe that the traditional optical character recognizer based method is inadequate for document image retrieval due to its requirements to high computational cost, vulnerability to images resolution, and due to its dependency on the used language. Moreover, Yahia [8] admits that using the classical OCR for the retrieval is unsatisfactory, especially when retrieving the texts from historical Arabic manuscripts. That is due to the special characteristics in the ancient Arabic manuscripts that make the retrieval process more complicated. The historical Arabic manuscripts, in particular, posing additional challenges due to their degraded quality of blotched papers, faded inks, and lots of non-textual noise added within the main text. Al-Ayyoub et al. [2] state that the automatic manipulation and understanding of the Arabic language is challenging due to its distinct characteristics. In addition, Al-Jawfi [9] admits that the handwritten Arabic language has characteristics that make recognizing it harder than other languages. For instance, the fact that Arabic words are written from right to left, and their dots might appear above or below the letters, also the dots are ranging from 0-3, makes the recognition of the Arabic language more challenging. Even though the Arabic alphabets are consisting of only 28 letters, most of them come in four different forms depending on their position

within the word, which increase the alphabet patterns from 28 to around 60.

### B. RESEARCH SIGNIFICANCE AND CONTRIBUTIONS

It is important to process and being able to search low-quality faded inks historic images and not only the high-quality computer printed and recent images. The Arabic handwritten manuscripts are valuable piece of historical information that reflect the education, culture, society, and tradition during specific time periods. They also highlight the developments in the language through time. Retrieving images from the same manuscript is a desirable application for electronic libraries that require retrieving specific book literature and knowledge.

Considering that, the historical Arabic manuscripts are text-based images. Thus, it is crucial to extract the text from the images and to retrieve images according to their extracted textual features. However, because the historical Arabic manuscripts' images are also not purely textual. Instead, they include handwritten signatures, drawings, figures, tables, side-notes, etc [8]. Therefore, it is necessary to consider the non-textual parts within the historical Arabic manuscripts images and to retrieve images according to their extracted visual features [9].

In this study, we focus on retrieving the most relevant images to a user query image according to its automatically extracted deep learning visual and textual features. Contributions of this paper are summarized as follows:

1. Develop visual model by transfer learning from four pre-trained deep CNNs.
2. Develop textual model through optimizing the classical LSTM deep learning model.
3. "To the best of our knowledge", we are the first study that fuse both visual and textual deep learning models at different levels and using different rules to retrieve the Arabic manuscripts' images accurately.

Rest of the paper is organized as following: Section II discusses the literature review and handles the classification of the previous researches to image retrieval techniques. Section III explains the proposed method, which consists of three main steps: visual-based image retrieval, text-based image retrieval, and image retrieval using the fusion model. Section IV highlights the experiments accomplished to find the most accurate model and distance metric for measuring the similarity scores and discusses their results. It ends by comparing the results of the proposed method with the existing state-of-the-art methods. Finally, section V summarizes and concludes the paper.

## II. LITERATURE REVIEW

Image retrieval-based system, which is also known as Content-Based Image Retrieval (CBIR) is a technique to classify and retrieve queried images based on some specifications. Many studies implemented the image retrieval system, some of them based on extracting the visual

features. While other studies based on extracting the textual features. There were also some efforts implemented to fuse multiple features into one fusion model.

The extracted features from the images could be local such as color, shape, texture, etc., which are known as the hand-crafted features. In contrast, there are global high-level features of images, such as their activities or objects, which are known as the deep learning features

The literature review section begins by reviewing the classification of image retrieval techniques by many researchers. Afterward, we divide the image retrieval techniques into three main subsections: visual-based features, textual-based features, and fusion-based image retrieval. Both the visual and the textual subsections are divided further into hand-crafted features and deep learning features.

## A. CLASSIFICATION OF IMAGE RETRIEVAL TECHNIQUES

There were several efforts to classify image retrieval techniques into main categories. Aslandogan and Yu [10] classified image retrieval techniques into two broad categories, as either visual features or non-visual features. However, Marshall & Gunasekaran [1] customized the techniques further by classifying them into six main categories as following: 1) content, 2) text, 3) annotation, 4) semantic, 5) sketch, and 6) query-based. They were considering that the content-based image retrieval techniques are methods that leverage the low-level visual features presented in the images. On the other hand, the text-based image retrieval techniques are based on segmenting the images and extracting the text included in the images. Annotation-based image retrieval techniques are based on labeling the images to search and match specific keywords with the required label of the annotated images to retrieve them. While the semantic-based image retrieval techniques tend to concentrate on the semantic knowledge presented in the multimedia images such as their regions. Whereas, the sketch-based image retrieval are the techniques that capture the drawings to retrieve images.

In regard of the annotated images techniques, there were many studies to do the annotation automatically instead of manually. For example, Saleem et al. [11] used the Markovian Semantic Indexing (MSI) method to do the annotation automatically. Furthermore, Kumar et al. [12] employed the weighted nearest-neighbor, as well as the multi-class classification techniques, to annotate medical images. Hare et al. [13] utilized a semantic-space method using linear algebra to do the annotation softly through training.

Reddy & Sreedhar [14] classified image retrieval techniques into five major categories as following: 1) text, 2) content, 3) semantic, 4) fusion multi-model, and 5) relevance feedback image retrieval techniques. The authors admit that many efforts were trying to integrate more than one image

retrieval approach to generate a fusion multi-model that can increase the accuracy of the output. Moreover, they state that there is another approach for image retrieval called relevance feedback, which is mainly based on the interventions and feedback from the interacted users with the retrieval system to amend and retouch the retrieved images.

Methods in relevance feedback technique include delta mean algorithm, standard deviation, variance, query point movement, Bayesian framework, support vector machine, biased discriminant analysis, and kullback-leibler distance.

Rui et al. [15] employed the relevance feedback method to successfully retrieve the images. They experimented their model on (384) texture images from the “MIT” media lab dataset, and they were able to increase the recorded precision using the feedback case, more than without employing any feedback method.

According to Wadhai & Kawathekar [16], the visual-based image retrieval techniques can be further classified by the local low-level features that are extracted from the images’ contents. Therefore, they summarized the utilized techniques in each type of extracting the local features as following: 1) color features, 2) texture features, and 3) shape features. The techniques used for color features are conventional color histogram, fuzzy color histogram, and color correlogram. On the other hand, the techniques used for the texture features are the steerable pyramid, contour let transform, and complex directional filter bank. Finally, the techniques used for the shape features are Fourier descriptor, moment invariants, and directional histograms.

Ahmed & Barskar [17] had a different classification to image retrieval techniques. They believe that based on the state-of-the-art researches, the image retrieval techniques are divided into three categories as following: 1) retrieval based on synthetic outlines, 2) retrieval based on visual contents, and 3) retrieval based on the semantic textual features included within the images. Dureja & Pahwa [18] classified image retrieval techniques into three other distinct categories. They believe that the techniques to image retrieval started with depending on the visual features only to retrieve images, and then developed into using the distance metric learning; until it reached using deep learning technology to retrieve images. They state that the deep learning techniques that leverage the CNN layers to extract the images features automatically are currently the best techniques for retrieving images successfully.

After reviewing current existing techniques for image retrieval, we focus on the Content-Based Images Retrieval (CBIR) systems that employ pure visual-based images as input to the image retrieval system. Afterward, we investigate the techniques used with text-based images for classifying and retrieving the text in general, as well as the techniques used for Arabic manuscripts image retrieval in particular. That is because our dataset consists of Arabic manuscripts, which are text-based images.

## 1) VISUAL-BASED HAND-CRAFTED FEATURES

Bagasi and Elrefaei [19] compared between retrieving similar images using the Speeded-up Robust Feature (SURF) combined with the Sum of Squared Differences (SSD) measurement technique, and between retrieving images using the Binary Robust Invariant Scalable Keypoints (BRISK) combined with the Hamming Distance (HD) measurement technique. They concluded that SURF local-based visual features extraction technique outperforms the BRISK because it reached 61% accuracy, while BRISK achieved only 37% accuracy.

Chen et al. [20] proposed using an interactive feature learning model to harmonize the contour descriptors with the interior region of 3D images. The authors used the “ETH” 3D images dataset to evaluate their proposed model. They measured the similarity between the query object and the 3D objects stored in the dataset through two main steps. First, they used a greedy search algorithm. Second, they implemented three bipartite graph matching algorithms to reach the perfect match between each bipartite graph pairs. The authors believe that applying the ranking method to the proposed model would improve its retrieval performance. For example, the Manhattan distance would enhance its performance up-to 21.5% if ranking method applied on the “ETH” 3D images dataset.

Raju et al. [21] proposed improving the performance of retrieved images by utilizing more than one distance method for measuring the similarity between the query image and the stored images in the dataset. They recommended using a content descriptor to extract the visual features from their collected dataset according to both images' colors and edges. Afterward, they utilized a mini-max method to obtain the most accurate similarity measurement with the query image. The method combines the measurements from three different histogram distance methods, which are: Euclidean, Cosine, and Histogram intersection. They concluded that the Euclidean distance measurement accomplished more accurate results than the Cosine and the Histogram intersection distance measurements. That is because it accomplished an 89% accurate precision rate.

Beecks et al. [22] investigated the rigidity of utilizing signature-based similarity measurement to correctly and rapidly retrieve content-based images from large datasets. They suggested feature signature representation of the original images to extract three local features from them, which are the color, position, and texture features. The authors utilized various distance signature-based methods such as the earth mover's distance, the perceptually modified Hausdorff distance, and the signature quadratic form distance to measure the correlation between the two feature signatures. They concluded that there is no direct correlation between the stability and the retrieval performance because the highest average precision stability was recorded by the “Copydays” dataset as 0.763. In contrast, its retrieval

performance was not that good comparing it to other datasets.

## 2) VISUAL-BASED DEEP LEARNING FEATURES

Radenovic et al. [23] proposed using a VGG deep CNN combined with a generalized mean pooling layer for generating the images features vectors. Then, they used a Siamese model for matching the images and retrieve similar images to a query image. They used datasets that include 3D objects. Afterward, the Euclidean function was employed to compare their trained input images with the queried image. The authors evaluated their proposed method by computing the mean Average Precision (mAP) and recorded 91.9% using both the “Oxford5k” and the “Paris6k” datasets.

Zhou & Jia [24] trained two Siamese multi-layer perceptron networks to verify the similarities among sketched images and 3D images from their employed gallery. They extracted the features using a Sketch-Based Local Binary Pattern (SBLBP) algorithm. Then, the authors measured the similarity through computing the intersection between two viewpoints distances and divide the result by the images' unions. Finally, the authors recorded 60% precision on the “NTU” dataset and 39% precision on the “3D sketched” images dataset.

Seddati et al. [25] used the ResNet101 deep CNN to retrieve images successfully. Their proposed approach is based on the Multi-Scale Regional Maximum Activation of Convolutions (MS-RMAC) descriptor, which is utilizing the resulting fully connected CNN to extract images features. The authors measured the similarity using the K-Nearest-Neighbor (K-NN) algorithm to find-out the nearest four images to the original queried image. They evaluated their model by computing the mAP and accomplished 72.3% using the “Oxford5k” dataset, 87.1% using the “Paris6k” dataset, and 94% using the “INRIA Holidays” dataset.

Koch et al. [26] developed a deep Siamese neural network to recognize the handwritten digits within the “Omniglot” Dataset. The authors measured the similarity among the query digit and all other digits stored in the dataset using the Manhattan distance metric and recorded a final accuracy of 93.42%.

Ge et al. [27] experimented the effect of changing the feature vector size on the performance of the image retrieval system. They used three different pre-trained CNNs named: VGGM, VGG16, and GoogLeNet; to transfer learning into the “PatternNet” dataset. The Euclidean distance metric used to measure the distances among the images. The authors concluded that the highest achieved mAP was using the GoogLeNet pre-trained deep neural network with (832) features dimension since it reached 65.98%.

Ong et al. [28] proposed using the architecture of the pre-trained VGG16 deep learning model to develop a Siamese neural network for image retrieval. The authors employed two image datasets named Oxford and Paris to evaluate their model. They computed the similarities between the images



using the Euclidean distance metric and recorded 81.5% and 82.5% mAP on the “Oxford” and “Paris” datasets, respectively.

Qiu et al. [29] recommended using the ResNet pre-trained model to develop the Siamese deep neural network. They used the “TUM” dataset and created two customized datasets to show the model both positive and negative samples from the original dataset images. The researchers utilized the Euclidean distance method to measure the similarities and accomplished 87.7% precision.

Chaudhuri et al. [30] used the Siamese Graph Convolution Network (SGCN) to extract the visual features from the images and retrieve similar images to a query image. They begin by segmenting the images of both the “UC-Merced” and the “PatternNet” datasets. Afterward, they entered the segmented graphs of the images into the Siamese model to measure the similarity among the images using the Euclidean distance method. The authors reached 69.89% mAP using the “UC-Merced” dataset and, 81.79% mAP using the “PatternNet” dataset.

Wiggers et al. [31] proposed using the architecture of the AlexNet pre-trained deep learning model to build the Siamese model. They used the “Tobacco800” dataset to experiment their model. They reached 0.944 as the highest recorded mAP on the retrieved top-5 similar images using a feature vector of size (4096) combined with the Euclidean distance metric. Ioffe and Szegedy [32] experimented classifying labels from the “MNIST” dataset using the Inception pre-trained deep learning model, with and without the addition of the batch normalization layer. They concluded that without the batch normalization layer, the Inception classification model recorded 72.2% validation accuracy. And after adding the batch normalization layer, the validation accuracy raised up-to 74.8% in fewer time steps, which proves the effectiveness of using the batch normalization for both increasing the accuracy rate and for minimizing the training time.

### 3) TEXTUAL-BASED HAND-CRAFTED FEATURES

Al-Maadeed et al. [33] recommended a system for Arabic handwritten recognition that employs a web-based interface to search the text and perform the retrieval. Using the SQL server, the authors built their database of Arabic manuscripts. For the testing purpose, the authors used the “Ibn Sina” database, which contains old Arabic manuscripts. They developed the interface using Microsoft Visual Studio and Asp.Net. The authors believe that the segmentation phase in processing historical Arabic manuscripts is very challenging and that there is a significant need to convert manual written manuscripts into digital versions to simplify the process of searching and visualizing them.

Adam et al. [34] used ancient Arabic manuscripts to discover and test a good algorithm for manuscripts' age and authors detection. They utilized the “KERTAS” dataset, which consists of more than 2000 images of high-quality

scanned ancient Arabic manuscripts. To tackle the features extraction problem, the authors employed two techniques. First, is the sparse representation-based technique, which uses normalization to choose the nearest sub-space of the manuscript being assisted. Second, is the handwriting style-based features. The authors resized the images looking for the best size to discover manuscripts' features. They started with 12×12 pixels. Then, they increased the sizes to 25×25, 50×50, 100×100, 200×200 and till 250×250 pixels. They concluded that with reducing the size of the images, most of the features become unclear, which minimizes the chances to find the right matching manuscript. Similarly, increasing images size dramatically might cause the same un-clarity in visualizing images features. Eventually, they concluded that the most accurate size for visualizing images is 50×50 pixels and recorded 94.77% accuracy using the sparse representation-based manuscript age detection algorithm.

Asi et al. [35] experimented both the local and the global features of ancient Arabic manuscripts. Two datasets were used, which are: WAHD and KHATT. For the preprocessing purpose, the authors cropped the background of the scanner from original images and then segmented the main text. Regarding the local features, they were captured utilizing the Modified Contour Based Feature (M-CBF). On the other hand, the global features were recognized utilizing the “globalizing local key point descriptors”. Afterward, the similarity between the query manuscript and the manuscript stored in the dataset was measured using both the Cosine and the Chi-square distance metrics. The authors recorded 0.346 accuracy for the top-10 retrieved images using the M-CBF method.

Dinges et al. [36] collected hand-written Arabic texts in a user-friendly system to use them for testing their new segmentation-based approach for Arabic text recognition. The authors synthesized Arabic words vocabularies and called it “IESK-arDB SynWords”. They preprocessed the collected dataset through segmenting its words into individual characters. Afterward, the authors classified the segmented words using both the Support Vector Machine (SVM) and the Active Shape Model (ASM) classifiers. In addition, the authors used the Levenstein distance metric for measuring the similarities. They recommended implementing the work using C/C++ instead of using Matlab. Moreover, the authors expect that implementing the system on the GPU or the FPGA would expedite the work. They approximate from 0.1 to 0.2 seconds would process faster, leveraging the proposed implementation methods.

Al-Yahya [37] implemented both unsupervised clusterings, as well as the supervised classification for image retrieval. She applied a quantitative analysis called stylometric analysis on the linguistic features of the Arabic text to simplify the process of the automatic genre discovery. King Saud University Corpus of Classical Arabic texts (KSUCCA) used as the dataset for this study. The textual features extracted using the Most Frequent Words (MFWs)

algorithm. Afterward, four distance measures used, which are the Classic delta, Eder's delta, Argamon's linear delta, and the Canberra distance. The author concluded that using the unsupervised clustering, Argamon's linear delta was the most qualified distance measure reaching 85.5% accuracy of the clustering text genre. However, regarding the supervised classification, the Classic delta classifier provided the best results of 80%. On general attributive success, Eder's delta achieved the highest accurate results, with 75% accuracy in recognizing the text genre.

Yahia [8] recommends integrating both the Content-Based Image Retrieval (CBIR) techniques with the Latent Semantic Indexing (LSI) approach to facilitate the indexing of the historical Arabic manuscripts and eventually retrieving them. The used dataset was only two pre-scanned ancient Arabic manuscripts named: "Sahih Al-Bukhari" (صحيح البخاري) and "Mawaqeat Al-Haj wa Al-Umra" (مواقيت الحج والعمرة). The author constructed latent semantic indexing by computing the values of four local features as following: 1) concentric circle features, 2) angular line features, 3) rectangular region features, and 4) circular polar grid features. The similarities were measured utilizing the singular value decomposition. The author concluded that the most accurate set is the circular polar grid, with 78.8% recall.

Aghbari and Brook [38] introduced an approach for segmenting and retrieving ancient Arabic manuscripts. The authors used a hardcopy dataset called a Historical Arabic Handwritten (HAH) manuscript for their study by scanning it to convert it into a digital copy. Then, they preprocessed the scanned manuscripts through four steps as following: 1) binarization, 2) noise removal, 3) smoothing, and 4) thinning. After preprocessing the images of the manuscript, they segmented into words, and then each word segmented into its connected parts. The features then extracted from the connected parts by recognizing both the structural and statistical features. In addition, the feedforward technique of multi-language processing neural network used to classify the feature vectors. For the testing scenario, the authors used one historical Arabic manuscript named "كشف اللثام عن وجه الإسلام". The authors concluded that their model reached a fluctuated accuracy between 75% and 99%.

Al-Dmour & Fraij [39] suggested to segment the handwritten Arabic texts into lines using the Horizontal Projection Profile (HPP) technique and then, segment the lines further into words using the Gap Metrics (GM) technique. They used Arabic Handwritten Data-Base (AHDB) to conduct their study. They preprocessed their AHDB dataset images through filtering the text images and binarizing them. Afterward, they investigated the best clustering algorithm to extract words correctly from the historical Arabic manuscripts. Hence, they used the word spotting technique utilizing the Fuzzy C-Means (FCM) algorithm, which is distance based soft clustering method. Then, they measured the similarity between extracted words by computing the spacing between them using the Euclidean

distance metric. The authors test four different clustering algorithms looking for the best method to extract words within historical Arabic manuscripts. The first two clustering algorithms are K-Means (KM) and Fuzzy C-Means (FCM), which are based on the distance. While, the third clustering algorithm is the Gaussian Mixture Model (GMM), which is based on the probability. The last tested clustering algorithm is the Density Based Spatial Clustering of Applications with Noise (DBSCAN), which is based on the density. The authors concluded that FCM is the best clustering algorithm because it recorded 84.8% overall extraction rate.

Othman [40] proposed a model to recognize and analyze the Arabic text. He manually scanned and collected 120 ancient Arabic images to evaluate his model. He preprocessed his manually scanned Arabic images by converting the RGB-images into grey-scale version. Afterward, he converted the grey-scale images into binary-images using the adaptive binarization method. In addition, he removed the noise and all unrelated parts from the documents. In addition, he extracted the features from the ancient Arabic images utilizing the word spotting technique through the Bag of Word Fragments (BoWFs) method. Then, he measured the similarity among the query image and the rest of the images using two static formulas, which are the Histogram Intersection (HI) and the Earth Movers Distance (EMD). Finally, the author evaluated his BoWFs word spotting technique through computing both the precision and recall of the resulted output images. He was able to reach 89.60% precision rate and 50% recall. However, the author said that the output results might be biased because the query image is being counted within the output results. While, this weakness can be overcome through eliminating the query image from the output computation.

Snoussi et al. [41] recommended using the Outer Isothetic Cover (OIC) to segment the digital handwritten ancient Arabic manuscripts images into text lines. The dataset used is 100 handwritten Arabic images taken from form the KHATT Arabic text dataset. The (OIC) segmentation technique is based on Transparent Neural Network (TNN). Afterward, they evaluated their proposed method for segmenting handwritten Arabic text manually utilizing the "Test" subset within KHATT dataset. They reached that their model recorded 74% successful text extraction rate.

El Makhfi [42] proposed recognizing the written words within the Arabic manuscripts' images through extracting the handcrafted features. The authors started by preprocessing the input query image through converting it into grey-scale version then, segmenting it into lines and words. The textual features are then extracted from the Arabic manuscripts' images using the Speeded Up Robust Features (SURF) method. The similarity is then measured between the user entered query image and all the other images saved in the database by comparing the points of interest and computing the distance using the Euclidean or the Mahalanobis metrics. The authors evaluated their

proposed method using the HADARA80P dataset and recorded 95.27% recognition accuracy.

#### 4) TEXTUAL-BASED DEEP LEARNING FEATURES

Peng et al. [43] proposed projecting images into binary codes utilizing the RNN deep reinforcement learning technique. Because the utilized technique can track previously recorded errors, as well as, it can map each image to its similar binary code. The authors used Pytorch's deep learning package to evaluate their framework. Their model recorded 0.842 mAP when tested on both the "CIFAR10" and the "NUS-WIDE" datasets. While it recorded an average of 0.808 mAP when tested on the "MIRFLICKR" dataset.

Qian et al. [44] experimented the use of three deep learning models for classifying and measuring the similarities among articles. The experimented models were the Gated Recurrent Unit (GRU), the LSTM, and the Siamese deep learning model. The authors used two textual datasets to evaluate their models named: "Reuters\_50\_50" and "Gutenberg" datasets. After classifying the articles using the three proposed models, the authors measured the similarities utilizing the Cosine distance metric. They concluded that the Siamese deep learning model outperformed the other two experimented models since it recorded the highest accuracy as 99.8% using both tested datasets.

You et al. [45] recommended using the bidirectional LSTM layer within their proposed tree-based AttentionXML deep learning model for text classification. They used six datasets for evaluating their model named: EUR-Lex, Wiki10-31K, AmazonCat-13K, Amazon-670K, Wiki-500K, and Amazon-3M. The authors computed the precision to evaluate their model, and the highest precision they reached equals 95.92% using the "AmazonCat-13K" dataset.

Elnagar et al. [46] used two Arabic textual datasets named: SANAD and NADiA to experiment the effectiveness of deep learning models for Arabic text classification. They tested both the CNN and the RNN models and reached the highest classification accuracy as 96.94% when they added the attention layer after the RNN. Moreover, Liu and Guo [47] employed two attention layers after the main LSTM layer to attain the bidirectional approach for text classification. They concatenated both forward and backward representation of sentences to get the final comprehensive feature map. The authors evaluated their proposed model using seven different datasets named: MR, IMDB, SST-1, SST-2, Subj, RT-2K, and TREC. The highest accuracy they achieved equals 97.2%.

Du et al. [48] introduced a Convolutional Recurrent Attention Network (CRAN) for text classification. The authors used five datasets named: MR, SST-1, SST-2, Subj, and IMDB for evaluating their classification model. The highest recorded accuracy was 94.1% using the "Subj" dataset. Liu et al. [49] developed a bidirectional RNN attention deep learning model for sentence classification.

The final classification layer includes Softmax activation function. The authors tested their model on eight benchmark datasets and recorded 94% accuracy. Yang et al. [50] proposed a Hierarchical Attention Network (HAN) for text classification. The author's model consists of two encoders, one for encoding the words and the second for encoding the sentences, as well as, two attention layers for focusing on both the words and the sentences. They experimented their model on six different datasets that categorized as 80% for the training subset and 20% divided equally between both the testing and the validation subsets. The authors recorded the accuracy to evaluate their model and reached 75.8% as the highest accuracy using the "Yahoo answer" dataset.

Gao et al. [51] improved the text classification model that was initially proposed by [50] through adding a convolutional layer to it. Thus, it became a Hierarchical Convolutional Attention Network (HCAN). The advantage of adding the convolutional layer is to generate the embeddings' matrices for updating the attention weights. The authors evaluated their model on four datasets named: Yelp Reviews 2016, Amazon Reviews Sentiment, Amazon Reviews Category, and Pubmed. They computed the accuracy, and the highest result they could achieve equals 89.9% using the "Amazon Category" dataset.

#### 5) FUSION-BASED IMAGE RETRIEVAL

Potrus et al. [52] proposed a hybrid handwritten Arabic text recognition technique. They leveraged a manually collected dataset consisting of 4500 Arabic words to conduct the study. Moreover, they use another dataset that contains 7851 Arabic words known as (ADAB) dataset to evaluate their model. They preprocessed the written text by removing its noise and all unrelated non-textual parts included within the images. After preprocessing the images, the authors segmented and recognized the Arabic letters online using an integrated Genetic Algorithm (GA) with a Harmony Search (HS) algorithm. First, the dataset is segmented using a dominant point detection and afterward, the segmented text are recognized using the GA-HS fused model. The authors use the harmony search character algorithm to be able to measure the similarity among the searched Arabic characters and the characters stored in the dataset. Finally, the authors evaluate their proposed GA-HS fusion model for online Arabic characters recognition using two datasets. The model recorded 93.6% successful recognition rate when tested on the first dataset, which is collected manually. While, the second dataset was called ADAB, the model recorded 94.68%-96.33% successful recognition rate when tested on the ADAB dataset. However, the authors claim that the search process takes very long to find-out the relevant characters because its uses the harmony search algorithm, which is based on the stochastic process.

Wang et al. [53] recommended a fusion between CNN and the RNN to accurately classify and retrieve multiple-label images. The use of RNN empowered the framework because

of its ability to memorize the dependency occurrence of labels. They computed the similarity measurement using the greedy algorithm, but it didn't perform well due to its dependency on the initially predicted labels. Thus, they used the beam search algorithm to be able to predict the nearest labels in an image successfully. For the evaluation purpose, the authors used the Caffe deep learning framework to experiment with their CNN-RNN fusion model on three datasets. They calculated the mAP and reached 84%.

Sharif et al. [54] integrated both the Scale-Invariant Feature Transform (SIFT) local features descriptor with the (BRISK) features descriptor to retrieve similar images. They measured the similarities among retrieved images utilizing the Histogram intersection distance metric and evaluated their fusion model by calculating the mAP on different image datasets. They reached 78.14% mAP on the "Corel-1.5K" dataset and 57.37% mAP on the "Corel-5K" dataset.

Guo et al. [55] proposed a CRAN hybrid CNN-RNN deep learning model with an attention layer for improved text classification. The model evaluated in both English and Chinese languages. In addition, it evaluated on Chemistry, Physics, and Mathematics subjects. The highest F-Score the authors recorded equals 86.19% using the Chinese language dataset. Koesdwiady et al. [56] utilized the decision-level fusion model to investigate the relationship between traffic flow and weather conditions. They used the "San Francisco Bay Area" traffic and weather data to evaluate their fusion model. Initially, they extracted the weather features using the Deep Belief Network (DBN) to predict the traffic flow according to the extracted features. Then, they computed the Mean Average Error (MAE) and recorded 0.0487 using the traffic data only, 0.2192 using the weather data only, and 0.0405 using both fused predictions.

Li et al. [57] recommended increasing the accuracy of face recognition through a decision-level fusion model. The authors begin by extracting the features of faces' images using the deep CNN and using the traditional 2-Dimensional Principal Component Analysis (2DPCA) method. Then they measured the similarity scores using the Euclidean distance metric for the 2DPCA and using the Mahalanobis distance metric for the CNN method, and fused the similarity scores at the decision level. They used the "LFW" faces datasets to evaluate their fusion-model and reached 91.98% accuracy on rank-1.

Su et al. [58] employed two deep CNNs named LMCNet and MCNet to classify the environmental sounds. The developed two deep neural networks consist of four convolutional layers followed by the final classification dense layer. The authors fused the "Softmax" layers at the decision-level and evaluated the classification accuracy of their models using the "UrbanSound8K" auditory dataset. The recorded accuracy using the LMCNet equals 95.2%, while the recorded accuracy using the MCNet equals 95.3%. On the other hand, the recorded accuracy using both fused

models equal 97.2%, which is higher than the classification accuracy by each separate deep CNN.

Xie et al. [59] proposed fusing three extracted features at the decision-level to improve the classification accuracy of the lung nodules. They used the "LIDC-IDRI" dataset to experiment their proposed model. The authors begin by extracting the texture features from the dataset images using the Gray Level Co-occurrence Matrix (GLCM), as well as, extracting the shape features using the Fourier method, and finally extracting the visual features automatically from the lung medical images using the Le-Net-5 deep CNN. To assess the performance of the model, the authors computed the Area Under Curve (AUC) and accomplished 96.65%.

Liu et al. [60] developed an image retrieval system based on fusing both the low-level features extracted using the Dot-Diffused Block Truncation Coding (DDBTC) method, with the high-level features extracted using the GoogLeNet deep CNN. The authors computed the Average Precision Rate (APR) using ten different images' datasets to evaluate the performance of their proposed features-level fusion model. They recorded 98.08% as the highest reached APR using the "USPTex" dataset.

Sudha and Ramakrishna [61] compared between the different features-fusion methods found in Matlab programming application and named: left-right, right-left, down-up, and up-down features fusion. The authors begin by pre-processing the iris images within the "CASIA" dataset. Afterward, they extracted the features from the pre-processed images using six different features extraction techniques called: Discrete Wavelet Transform (DWT), Local Binary Pattern (LBP), Gabor filters, Principal Component Analysis (PCA), Support Vector Machine (SVM), and Fast Fourier Transform (FFT). Finally, the authors calculate the accuracy of each features-fusion method using various combinations from the six features extraction techniques. The authors concluded that the down-up features-level fusion method is outperforming the other methods recording 94.32% success rate using the features fused from the LBP-FFT and the LBP-DWT techniques.

Suk et al. [62] recommended fusing the features extracted from Alzheimer's disease images existed within the "ADNI" dataset to increase the accuracy of the disease diagnostic. Initially, the authors extracted the high-level features from both the Magnetic Resonance Imaging (MRI) and the Positron Emission Tomography (PET) images utilizing the Deep Boltzmann Machine (DBM). Then, they fused the extracted features from both images and computed the disease recognition accuracy. The highest accuracy they reached equals 95.35% successful recognition of Alzheimer's disease.

Zhang et al. [63] proposed a hybrid fusion model of both visual and auditory extracted features for improving the recognition of emotions in videos. The authors used the pre-trained AlexNet CNN to extract the visual features, and the 3D-CNN to extract the audio features from three different



datasets named: RML, eNTERFACE05, and BAUM-1. Afterward, the two extracted features were fused using Deep Belief Networks (DBN). Then the two DBN networks were fused again using the SVM for classifying the emotions in the videos. The highest recorded accuracy by the hybrid proposed model equals 85.97% using the “eNTERFACE05” dataset.

Xia et al. [64] recommended using a transformation-invariant deep hashing model for classifying and retrieving trademark images. They begin by augmenting the dataset's images. Then, they extracted the visual features from the images using two deep learning models named: Spatial Transformer Network (STN) and Recurrent Convolutional Network (RCN). Afterward, they fused the visual extracted features using a hashing layer to generate the binary codes of the images. The similarity between the query image and the rest of the images stored in the dataset computed utilizing the Hamming distance metric. The authors evaluated their model by calculating the mAP of the retrieved images, and they were able to record 0.449 mAP using the “NPU-TM” dataset, and 0.501 mAP using the “METU” dataset.

Vishi and Mavroeidis [65] evaluated four different score-level fusion methods called: Minimum-Score (MinS), Maximum-Score (MaxS), Simple-Sum (SS), and User-Weighting (UW) for classifying the biometric features. The authors extracted the features using both the fingerprints and the finger-veins techniques from the “SDUMLA-HMT” dataset. Then, they experimented the four score-level fusion methods combined with three various score normalization approaches named: Min-Max (MM), Z-Score (ZS), and Hyperbolic Tangent (TanH). Eventually, the authors concluded that the use of the SS score-level fusion method in conjunction with the TanH score normalization approach outperformed other methods recording 99.98% successful classification.

Lip and Ramli [66] experimented the fusion of the humans' visual images with their speech biometrics using the decision-level, features-level, and score-level fusion methods. The authors begin by extracting the visual features using the Region of Interest (ROI) and extracting the audio features using the Mel Frequency Cepstral Coefficient (MFCC) method. Afterward, they employed the SVM to classify the extracted features. The authors used the Audio-Visual digitized database, which includes the digital versions of both the speech and the images to evaluate their fusion methods. Eventually, they reached that the score-level fusion is outperforming the other two methods recording 99.95% accuracy, followed by the features-level fusion recording 99.75% accuracy, and finally, the decision-level fusion recording 99.66% accuracy.

Kaya et al. [67] fused the audio and visual features extracted from video inputs using a weighted score-level fusion model. The pre-trained VGG-Face deep learning model was used to extract the visual features, while the openSMILE model was used to extract the auditory features.

The authors evaluated their fusion model using two videos datasets named: Extended Cohn–Kanade (CK+) and MMI. They accomplished 98.47% accuracy using the “CK+” dataset, and 72.46% using the “MMI” dataset.

Kang et al. [68] proposed a score-level fusion model for face verification. The authors begin by extracting the visual features from the “LFW” datasets images using the Multi-scale Convolution Layer Blocks (MCLBs). Afterward, they improved the recognition accuracy by using the high-dimensional LBP method for the extraction of the visual features as well. Then, both extracted features joint using the SVM classifier, and the scores fused at the score-level, reaching 99.08% overall accuracy.

Bhushan and Danti [69] recommended a score-level fusion model for text classification. The authors initially pre-processed the text through stemming it utilizing four English language textual datasets named: Vehicle Wikipedia, Google newsgroup, 20 Mini newsgroup, and 20 Newsgroup. Then, they fused the generated scores from both the interval-valued classifier and the deep network classifier. The authors experimented the score-level fusion after categorizing the dataset into 60% training and 40% testing and validation, as well as, after categorizing the dataset into 40% training and 60% testing and validation. The highest recorded F-score was using the “Vehicle Wikipedia” textual dataset with the 60% training subset as 0.9773 successful text classification.

George and Routray [70] experimented identifying persons through extracting the statistical features from their eye movements. The used dataset consists of two main subsets. The first subset named (RAN), it contains white dot moving randomly on a black background. While the second subset named (TEX), it contains a white dot following a text written on the screen. Both RAN and TEX subsets were divided further into two versions: one version has only 30 minutes interval between the training and the testing data, denoted as (RAN\_30 and TEX\_30). While, the other version has a 1-year interval between the training and the testing data, denoted as (RAN\_1yr and TEX\_1yr). The authors begin by dividing the eye movement images into fixations and saccades. Afterward, they removed the noise from the eye movement images and extracted their features using the Gaussian Radial Basis Function Network (GRBFN). The highest recorded accuracy was using the RAN\_30 and TEX\_30 on the evaluation subset as 98.69% and 98.04%, respectively.

Jhansi and Reddy [71] applied a score-level fusion model for retrieving sketched-based images. They used the “TU Berlin Sketch” dataset to assess their fusion model. Initially, they extracted the visual features from the images using the Histogram of Gradient (HOG) descriptor. Then, they measured the distances among images using the Euclidean distance metric and fused the generated score distances from the HOG with the generated scores from the Gaussian Mixture Model (GMM). The authors computed the accuracy and reached 90% successful retrieval.

Jovic et al. [72] extracted the color, shape, and texture hand-crafted features from four image datasets. Afterward, they measured the distances among the query image's feature vector and all the dataset images' features vectors using the Euclidean distance metric to generate three similarity score lists. The final image retrieval is performed using one final fused and ranked similarity score list from all the three generated similarity scores lists. Three various methods tested to do the similarity scores fusion named: Inverse Rank Position (IRP), Borda Count (BC), and Leave Out (LO). The authors evaluated their fusion methods through computing the Average Retrieval Precision (ARP). Finally, they concluded that the IRP is performing better than the other two similarity scores fusion methods reaching 49.04%, 39.13%, 73.89%, and 41.23% ARP using the C-1000-A, C-1000-B, B-1776, and V-668 datasets, respectively.

Xue et al. [73] designed a fusion model that concatenates the features extracted from medical laboratory reports. The reports are text-based images that contain Chinese and Latin characters with numbers and mathematical symbols. The fusion model is based on a bidirectional LSTM model with CNN. The authors measured the distance among the extracted text using the Edit distance and recorded 98.6% precision.

From the presented literature, we realized that even though several techniques have been proposed for image retrieval, there is still a semantic gap between describing simple images with low-level features and describing large complex images with high-level features that simulate humans' perceptions. Analyzing the generated evaluation parameters, the papers worked on the image retrieval system utilizing the deep learning techniques recorded higher results than the papers that worked on the image retrieval system utilizing the handcrafted features. Moreover, we found that the fusion models included substantial details and more information about the fused data, which increased their accuracy than each individual used model. Therefore, there is a significant need to explore the ability of the fusion deep learning technology to retrieve images successfully.

Reviewing the papers worked on the Arabic image retrieval, we notice that little attention given to the Arabic manuscripts' retrieval. While more efforts accomplished in recognizing and classifying the Arabic texts. Therefore, there is a lack in the image retrieval using the Arabic manuscripts in particular, which need more efforts and considerations. Moreover, we noticed that the input data to the Arabic image retrieval system could be a complete text-image, a sub-word image, or even images containing one Arabic character. Thus, there is a need to investigate more on the retrieval of the complete Arabic manuscripts' images. Therefore, we plan to employ the deep learning features to retrieve the images of the Arabic manuscripts according to the manuscript search criteria.

### III. PROPOSED METHOD

The proposed method consists of three main steps, which are: visual-based image retrieval, text-based image retrieval, and image retrieval using the fusion model. Considering that any image retrieval system consists of two main steps, which are: the classification and the similarity measurement. Thus, each one of the three main steps are divided further into classification and similarity measurement. The classification section explains classifying the images or text according to the manuscripts' labels. While the similarity measurement section explains the method of matching the classified images or text to retrieve similar images to a user query image.

#### A. VISUAL-BASED IMAGE RETRIEVAL

Since the used handwritten ancient Arabic manuscripts' images are including visual contents, such as drawings, signatures, tables, ...etc. Thus, it is essential to extract the deep learning visual features of the images and to process them according to their extracted visual features. Figure 1 illustrates the proposed framework for the visual-based image retrieval.

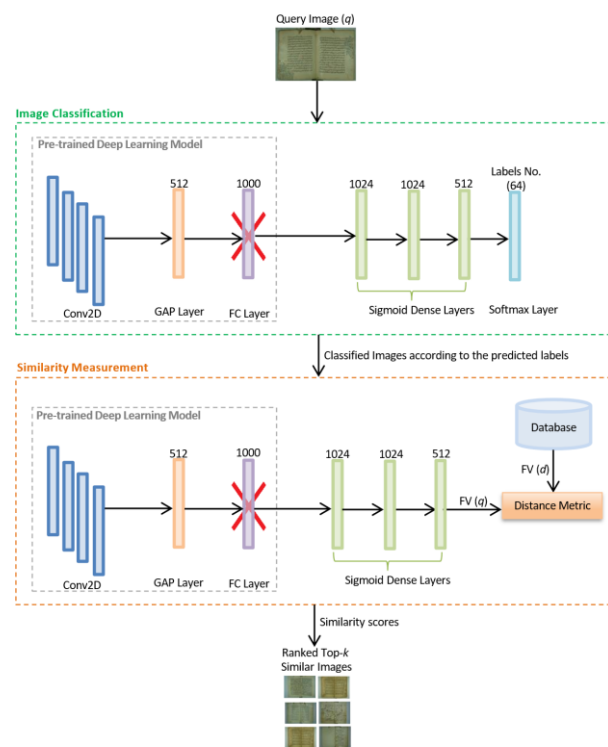


Figure 1. Proposed method for visual-based image retrieval.

To classify the images, we transfer learning from all the layers and weights of a pre-trained deep learning model. But the last fully connected layer that comes with the original model and usually has a feature vector of size (1000) should be removed and add instead of it a "Softmax" layer,

including the number of labels that we aim to classify the images according to. Considering that the used ancient Arabic manuscripts dataset includes (64) manuscripts. Thus, the number of labels in the last added “Softmax” classification dense layer, will include (64) labels. Before the final “Softmax” classification dense layer, there are three “Sigmoid” dense layers to improve the classification accuracy.

After classifying the images, the feature vector of the user query image, denoted as  $FV(q)$ , is generated utilizing the “Sigmoid” dense layer. Moreover, the features vectors of all the images in the dataset, denoted as  $FV(d)$ ; are generated and saved in a database to enter them along with the feature vector of the user query image into a distance metric to measure the distances among the images and display their similarity scores. Finally, the similarity scores are ranked to output the top- $k$  similar images to a user query image. The following subsections explain the image classification and similarity measurement in more detail.

## 1) IMAGE CLASSIFICATION

The image classification step has two phases, which are the training phase and the testing phase, as illustrated in figure 2.

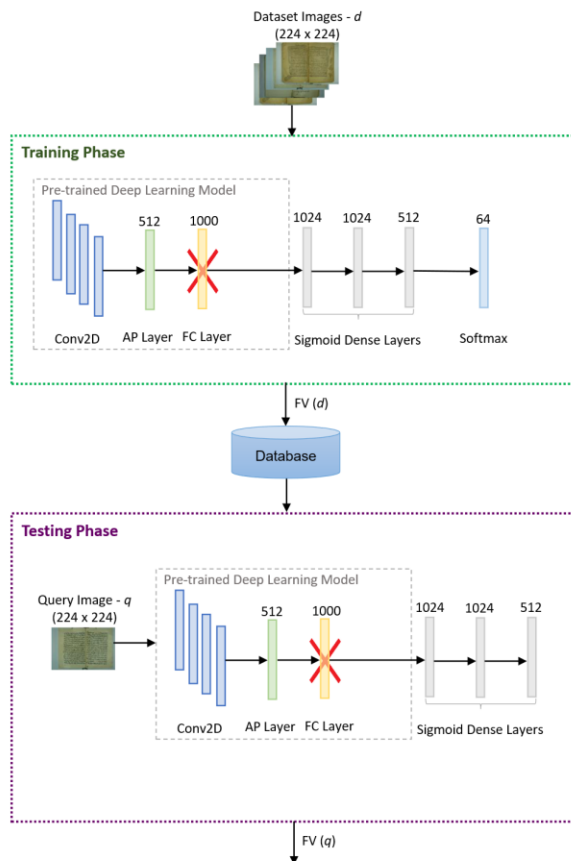


Figure 2. Training and testing of the CNN for image classification.

In the training phase, all the dataset images enter the customized deep pre-trained CNN to classify the images according to the predicted labels. After classifying the images in the training phase, the features vectors of the classified images, denoted as  $FV(d)$ , will be generated and saved in a database for faster retrieval during the testing phase.

In the testing phase, the user query image enters the deep CNN to generate its feature vector, denoted as  $FV(q)$ , utilizing the “Sigmoid” dense layer that falls just before the last fully connected layer and employed (512) dimensional feature size.

## 2) SIMILARITY MEASUREMENT

The output predictor from the CNN model assists in predicting if the two images are similar or not, through computing the similarity scores of the dataset images. The similarity score is defined as the difference between the distances of two input images’ features vectors. If the difference is high, then they are not similar, and their generated similarity score will be closer to (0). On the other hand, if the difference between their distances is low. Then they are more similar, and their produced similarity score will be closer to (1). There are many distance metrics for calculating the difference between the images’ feature vectors. The three following commonly used distance metrics were examined, looking for the one that generates the most accurate evaluation parameters:

- Manhattan ( $L1$ )
- Euclidean ( $L2$ )
- Cosine

The static formula of the Manhattan distance metric, which is also known as the City-block distance and  $L1$  method, computes the absolute difference between the coordinates of the two input images, as illustrated in equation (1) [7].

$$L1(A, B) = \sum_{i=1}^n |A_i - B_i| \quad (1)$$

Where  $L1$  is the distance between the two features vectors,  $A$  and  $B$  are the extracted feature vector from the two input images,  $A$  is the feature vector of the query image,  $A = \{A_0, A_1, \dots, A_{n-1}\}$  and  $B$  is the feature vector of the stored image in the ancient Arabic manuscripts dataset,  $B = \{B_0, B_1, \dots, B_{n-1}\}$ ,  $n$  is the number of dimensions in each image feature vector, and  $i$  represents the  $i^{th}$  feature in the vector.

Regarding the Euclidean distance metric, which is also known as the  $L2$  method. It is the squared root form of the  $L1$  method. Its equation is presented in (2) [74].

$$L2(A, B) = \sqrt{\sum_{i=1}^n (A_i - B_i)^2} \quad (2)$$

The Cosine distance is a static similarity method that measures the Cosine angle between the two input images' feature vectors, in which the similarity score between the two images increases as much as the computed Cosine angle between their vector decreased [75]. The Cosine equation is presented in (3) [76].

$$\text{Cos}\theta(A, B) = \frac{A_i \cdot B_i}{|A_i||B_i|} = \frac{\sum_{i=1}^n A_i \cdot B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}} \quad (3)$$

After calculating the similarity scores for all the dataset images, the scores are ranked in a descending order, and the most similar top- $k$  images to a user query image are retrieved. The mean accuracy is computed to evaluate the retrieval process. Algorithm-1, explains the pseudo-code for calculating the mean accuracy per the top- $k$  retrieved images.

---

**Algorithm-1** (Calculate the retrieval mean accuracy)

---

```

mean accuracy = 0
count = 0
 $A_q$  = feature vector of the query image  $q$ 
 $B_d$  = features vectors of all the images in the dataset,
 $B_d = \{b_1, b_2, b_3, \dots, b_j\}$ 
For ( $N$ ), where  $N$ = dataset size
{ similarity array = distance metric ( $A_q, B_d$ )
  retrieve top- $k$ , where  $k \in \{10, 25, 50, 100\}$ 
  count +=1
}
 $c$  = number of correct predictions out of  $k$ 
if ( $c$  == total number of images in the manuscript)
{ accuracy = 1 }
else
{ accuracy =  $c / k$  }
mean accuracy =  $\sum_{i=1}^N \text{accuracy} / N$ 

```

---

## B. TEXT-BASED IMAGE RETRIEVAL

The Arabic manuscripts dataset includes text-based images. Thus, it is more logical to recognize the texts written inside

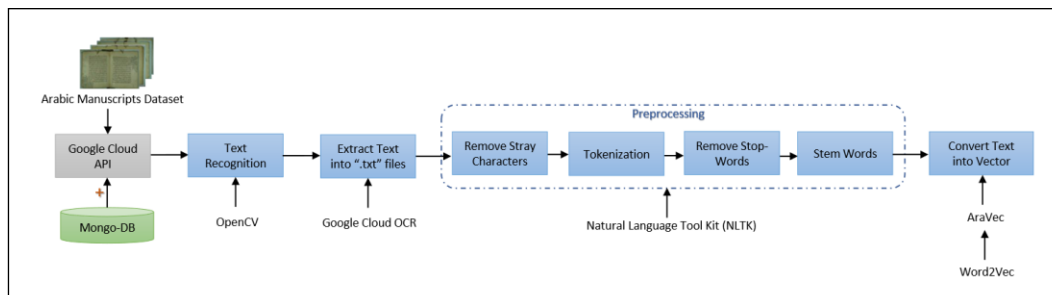
the images and extract them to be able to retrieve the images according to their textual features.

There are many challenges in recognizing and extracting the Arabic text written inside the ancient manuscripts' images because the Arabic language is morphologically complex [6]. In fact, the ancient Arabic manuscripts that are handwritten, very old, and having low-quality resolution are much more difficult to manipulate. To overcome the challenges associated with the handwritten Arabic language, we set-up the plan illustrated in figure 3 for recognizing and extracting the text.

From figure 3, we notice that the work begins by uploading the Arabic manuscripts dataset onto the "Google Cloud Platform"<sup>1</sup>. Afterward, we go through all the uploaded images in the Google Cloud and link them with the freely available "MongoDB Atlas"<sup>2</sup> database management system to facilitate manipulating our big data.

After setting up the data, the text segmentation and recognition is achieved utilizing the Open Source Computer Vision Library (CV2/OpenCV)<sup>3</sup>. It is a software package specialized in machine learning and including more than (2500) enhanced algorithms that facilitate both the physical and the organizational structures for various computer vision-based applications [77]. The text-images segmented into lines. Then, the lines segmented further into words because we need to discover the connected parts within each word that represents the Arabic characters/letters. Afterward, the layers of the images identified to be able to recognize the Arabic words written through different layers. The Maximally Stable Extremal Regions (MSER) algorithm employed to get the list of pixels set for each word. The colored images converted into grey-scale because grey-scale images process faster than RGB images. In addition, the "DetectRegions" function in MSER algorithm used to detect the regions of each word. Then, a minimum polygon around the detected regions of each identified word drawn using the "ConvexHull" function.

The advantage of being able to recognize the connected components through drawn polygons is to extract words based on their shapes instead of the lines they lay on. The



**Figure 3.** The framework of text recognition and extraction.

<sup>1</sup> <https://console.cloud.google.com>

<sup>2</sup> <https://www.mongodb.com/cloud>

<sup>3</sup> <https://opencv.org/about.html>



structure of the words in the ancient Arabic manuscripts' dataset is difficult to recognize due to the old writers' handwriting styles, as well as, the individual words are spread over multiple lines/rows and not laying on one separate line. Therefore, we had to create a two-dimensional array to detect the maximum peaks in each word and the counterpart minimum peaks. That is achieved utilizing the "local\_maxima" and the "local\_minima" functions. Hence, we can benefit from the local minima points to draw lines between the recognized lines in the original image, as illustrated in figure 4.

Figure 5 illustrates the horizontal projection profile processing on an image. The main purpose of the horizontal projection profile function is to segment images into lines/rows. It is a top-down text-lines recognition method. Keeping in mind that what interest us are the black pixels only because they are the pixels that include the words. Furthermore, we concern about the least/minimum peaks in

the image because they are the points that correspond to each line boundary. In other words, the bottom/underneath points are the lines between the black pixels

Figure 5(a) clarifies that our segmented image includes 1400 row/line and that the total number of detected black pixels in the image are more than 800 pixels. Figure 5(b) determines lines as peaks whereas the maximum peak is presented by the green color, and the minimum peaks are presented by the red color.

Similar to the horizontal projection profile, figure 6 shows the segmenting of the image into columns instead of rows. Therefore, segmenting the lines further into words since this method determines words as peaks. i.e., it represents the maximum black characters in the words as peaks. Note that there are two big deeps in the middle of the figure corresponding to the separator between the two manuscript pages because most of the scanned images in the used dataset include two images, not only one.

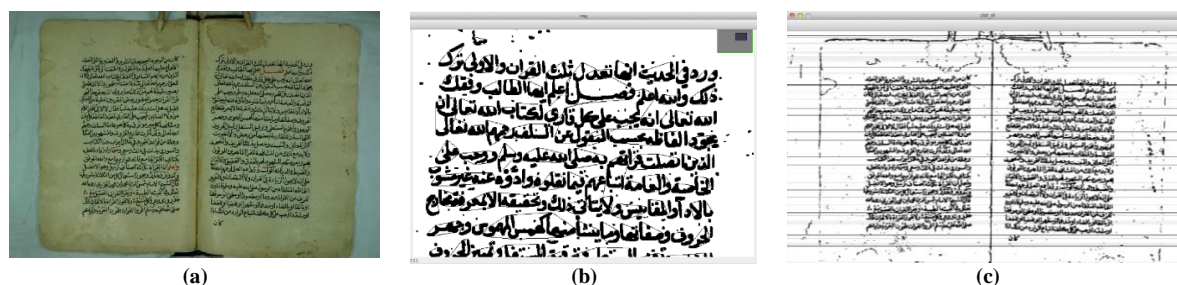


Figure 4. Text recognition. (a) Original image, (b) Drawn polygons around the words, (c) Lines display utilizing local minimum points.

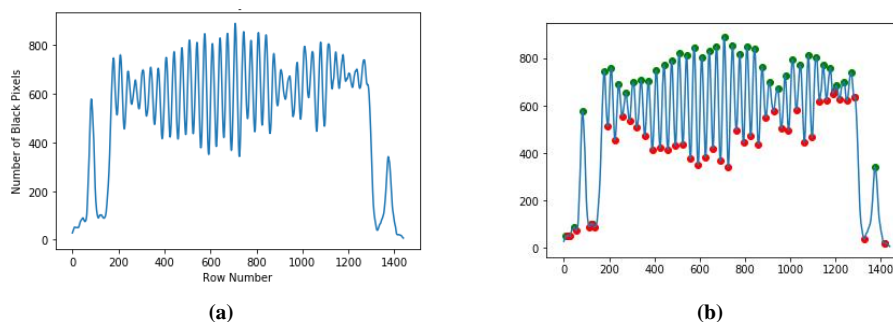


Figure 5. Horizontal projection profile. (a) Detected lines, (b) Maximum and minimum lines peaks

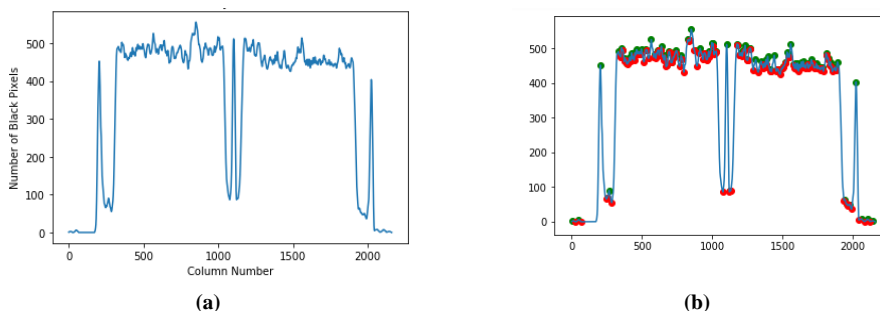


Figure 6. Vertical projection profile. (a) Detected words, (b) Maximum and minimum words peaks

The “Google” optical character recognition vision tool is used to extract the handwritten Arabic text from each segmented image correctly. The process repeated on all the dataset images and the extracted text saved in a MongoDB file associated with each image.

After extracting the text, it cleaned utilizing the Natural Language Tool Kit (NLTK 3.4.5)<sup>4</sup>, which is a NLP library, in conjunction with the “PyArabic 0.6.6”<sup>5</sup> Python library specialized in processing Arabic language text. The text cleaned through implementing the following steps:

- 1- Remove stray characters such as punctuations, special characters, digits, and non-Arabic letters.
- 2- Text tokenization, which is a process of splitting the main text into keywords (tokens) according to the existence of the white spaces between the characters to be able to process the tokens further individually.
- 3- Eliminate stop words; there is a list of (243) Arabic stop words within NLTK corpus. Thus, we used the list to clean the dataset’s text.
- 4- Stem words using the “ISRIStemmer” algorithm. Words stemming is a technique that retrieves the root of each word. It helps reduce the size of the used text while keeping it in its original base form. Hence, it can be used for querying and retrieving images including similar stemmed text.

To convert the cleaned tokenized text into feature vector, the “AraVec3.0” tool under “Word2Vec” words embedding tool is used. “AraVec” is a free pre-trained word embedding tool that is specialized in Arabic text processing. The recent version of “AraVec” has been trained on two main sources of Arabic text, which are Twitter and Wikipedia [78]. For our dataset, we employed the unigram model trained on Twitter with (100) vector size.

There are two types of converting text into a vector: 1) one hot representation and 2) distributed representation. The one-hot encoding of text converts it into a binary vector that includes only 0’s and 1’s. On the other hand, the distributed representation of text converts it into different numbers. They can be positive or negative numbers, and this is the type used in “AraVec” tool. The distributed representation of words considers the frequencies of mutual words that occurred within the text. The mutual information is one of the machine learning algorithms to classify text. It diagnoses the frequent occurrence between two words in a text. The advantage of using the mutual information algorithm over other algorithms is that it increases the classification accuracy [79].

“AraVec” word embedding tool has its dictionary of Arabic words. Thus, before converting the raw text into a distributed vector, the tool compares the text with its dictionary, and if the word is not included within the dictionary. This means that it might be extracted in a wrong

way, and hence it will not be converted into vector. This approach ensures that all converted text is accurate.

Following algorithm-2 describes the approach followed to convert the raw texts into features vectors.

---

#### Algorithm-2 (Convert text into feature vector)

---

Download the “Twitter-CBOW unigram model”<sup>6</sup> version of Aravec tool.

Let  $N$  be the size of the dataset.

$M$  is the set of manuscripts in the dataset,  
 $M = \{m_1, m_2, m, \dots, m_i\}$ .

$W$  are the words in the manuscripts’ images,  
 $W = \{w_1, w_2, w_3, \dots, w_j\}$ .

$V$  are the words in the AraVec vocabulary,  
 $V = \{v_1, v_2, v_3, \dots, v_k\}$ .

For ( $N$ )

{ 1- Load the cleaned text files from the dataset

If ( $w \in v$ )

{Then the extracted word is correct Arabic word exist within the AraVec vocabularies dictionary}

else

{Then the extracted word is incorrect and will be ignored}

- 2- Calculate the frequency of correct mutual words using the following Pointwise Mutual Information (PMI) equation [79]:

$$PMI(w, m) = \log \frac{\text{count}(w, m) * N}{\text{count}(w) * \text{count}(m)} \quad (4)$$

- 3- Generate a feature vector for each correct frequent word using the Aravec model.

- 4- Write the generated feature vector for each image into a “pickle”, which is a special Python format file used to save and process data easily.

- 5- Append vectors of all images together into one compressed vector, including each image id along with its feature vector.

- 6- Save the one complete vector in a “numpy” array on the hard disk.

}

---

#### 1) TEXT CLASSIFICATION

After extracting and preprocessing the text, the generated textual features vectors are passed into Long Short Term Memory (LSTM) deep learning model to classify the extracted textual contents according to specific labels. The LSTM is a special type of the RNN that solves the vanishing gradient problem existed in the RNN through adding memory blocks that are capable of learning long-dependent

<sup>4</sup> <https://www.nltk.org/>

<sup>5</sup> <https://pypi.org/project/PyArabic/>

<sup>6</sup> <https://github.com/bakrianoo/aravec/blob/master/README.md>

correlations. The LSTM deep learning model includes hidden state at time step ( $t$ ) denoted as ( $h_t$ ), as well as, it takes textual content input denoted as ( $x_t$ ). The input along with the hidden state pass through three LSTM gates named: input ( $i$ ), forget ( $f$ ), and output ( $o$ ) to update the weights ( $W$ ) at a cell activation vector ( $c$ ). The mathematical representation of the LSTM is presented in the following equations from (5) to (10) [47]:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (5)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (6)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (7)$$

$$\tilde{c}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (8)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c}_t \quad (9)$$

$$h_t = o_t \otimes \tanh(c_t) \quad (10)$$

Where  $b$  refers to the bias vector at the input, forget, and output gates, as well as at the cell activation vector. The  $\tilde{c}_t$  refers to the new updated memory cell at time step  $t$ , and the  $t-1$  indicates the previous forgotten memory cell. The symbol  $\otimes$  represents the element-wise multiplication operation.

Both  $\sigma$  and  $\tanh$  refers to the Sigmoid and the Hyperbolic Tangent activation functions, respectively, and their computations are presented in equations (11) and (12) [80]:

$$\sigma(x) = \frac{1}{(1 + e^{-x})} \quad (11)$$

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad (12)$$

Table 1 illustrates the architecture of the initial used LSTM deep learning model.

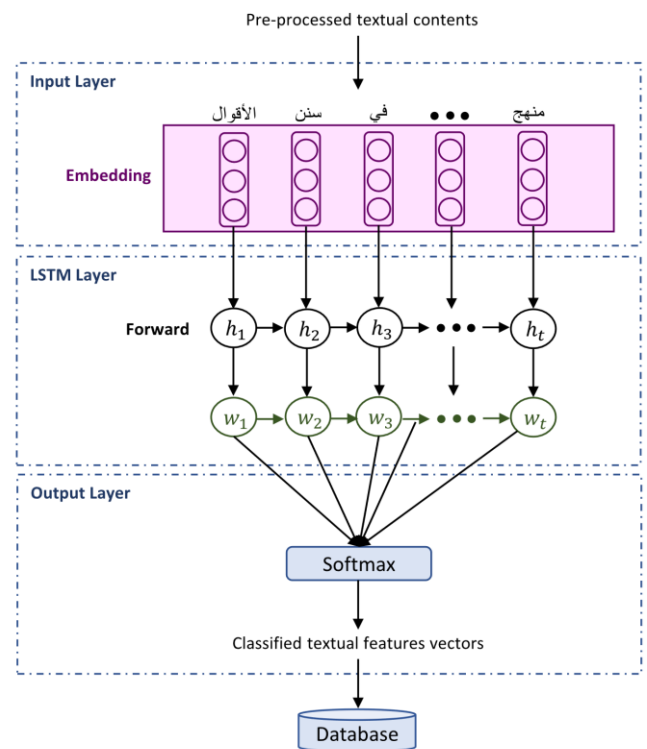
**Table 1.** The architecture of the LSTM deep learning model

Layer	Output Shape	Parameters No.
Embedding_1	(500, 100)	5000000
Spatial_dropout1d_1	(500, 100)	0
LSTM_1	(500, 64)	42240
Droupout_1	(64)	0
Dense_1	(64)	8256

From table 1, we notice that the LSTM deep learning model accepts textual contents of the shape (500, 100). The (500) refers to the maximum average of words in each input sequence or in each input sentence, while the (100) refers to the dimension of the words embedding. The textual contents are passed through the “Spatial dropout” layer to drop the

entire one-dimensional feature map. This is helpful for supporting independence among the features vectors before entering them into the LSTM layer. Afterward, the textual features vectors enter the LSTM layer that includes (500) cells or hidden states with a (64) time steps for every hidden state. Then, there is a regular “Dropout” layer to support the independence between the LSTM neural units, which avoids the overfitting problem. Followed by a final “Softmax” classification dense layer, including the (64) manuscripts’ labels that we aim to classify the text according to.

Figure 7 illustrates the architecture of the initial proposed model for text classification. For simplicity purposes, the dropout layers were eliminated from the model’s architecture.



**Figure 7.** The initial proposed model for text classification

From figure 7, we notice that the cleaned pre-processed textual contents enter the embedding layer to generate the features vectors of the text. The features vectors are then passed into the hidden states of the LSTM layer, denoted as  $\{h_1, h_2, h_3, \dots, h_t\}$  to generate the weight of each input word, denoted as  $\{w_1, w_2, w_3, \dots, w_t\}$ . The LSTM layer moves the weights in a forward manner only. The weights are then used by the Softmax dense layer to display the output classified textual features vectors according to specific labels. The features vectors are finally saved in a database for easier future access.

The architecture of the initial LSTM deep learning model optimized through implementing the following steps:

- 1- Make the LSTM layer bidirectional instead of unidirectional (BiLSTM). The bidirectional approach assists in making the model's weights flow in both forward and backward directions, which allow the model to read the inputs in two ways and increase its memory [46]. The authors in [45], [47], [49], and [73] improved their accuracy using the bi-directional technique.
- 2- Add the (Attention) layer after the LSTM layer. The attention layer is a custom layer from the "Keras" deep learning library to assist the model, focusing on the important textual contents for a more successful classification process. The "glorot\_uniform" distribution is used to initialize the attention layer's weights. These initial weights get updated with every time step through adjusting the weights exited from the preceding LSTM layer according to the more significant words. Many studies handling the sequence classification of words to labels, tend to employ the attention layer to improve the performance of their classification models, such as the studies done in [45- 51]. With the attention layer, the LSTM deep learning model focuses on every time step hidden state and updates its weights utilizing the "TanH" activation function followed by the "Softmax" function [47]. This technique stimulates the humans' brains in capturing and focusing the important words only and keep memorizing them for future actions. The mathematical representation of the attention layer is presented in equation (13) [81]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) * V \quad (13)$$

Where Q, K, and V are three real-number metrics for word embeddings representing the Query (Q), the Key (K), and the Value (V). According to the similarities between the text entries, the  $QK^T$  is generated, representing the new weighted matrix. The weighted matrix is then scaled using the  $\sqrt{d_k}$  dimensional factor and multiplied by the value matrix to obtain the adjusted attention's weights.

- 3- Add Batch Normalization (BN) layer before the final classification dense layer. Adding the batch normalization to the deep learning model has several advantages, one of them is expediting the training process, and more importantly, is increasing the validation accuracy [80]. With every training batch, the batch normalization layer normalizes the weights coming from the previous layers to feed the updated weights to the final classification dense layer. In other words, the batch normalization layer computes both the mean and the standard deviation of the input words to preserve the mean activation value close to (0) and the

standard deviation activation value close to (1), which normalizes the weights. The mathematical representation of the batch normalization is presented in equations (14) to (17) [80]:

$$\mu_B = \frac{1}{n} \sum_{i=1}^n x_i \quad (14)$$

$$\sigma_B^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_B)^2 \quad (15)$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (16)$$

$$y_i = \gamma \hat{x}_i + \beta \quad (17)$$

Where  $B$  refers to a mini-batch including  $x_i$  activation values,  $B = \{x_1, x_2, x_3, \dots, x_n\}$ ,  $\mu_B$  refers to the mini-batch mean, and  $\sigma_B^2$  refers to the mini-batch variance,  $\hat{x}_i$  is the normalized activation value,  $\gamma$  is the learned scale parameter, and  $\beta$  is the learned shift parameter for calculating the final Batch Normalization (BN) function. Hence, the  $BN_{\gamma, \beta}$  takes the input activation value  $x_i$  and normalizes it to get the output activation value  $y_i$ .

Afterward, the normalized activation values enter the final "Softmax" classification dense layer to calculate the probabilities of the text classification. The Softmax equation for calculating the probabilities is presented in (18) [82]:

$$p(w_t | w_{t-k}, \dots, w_{t+k}) = \frac{e^{y w_t}}{\sum_{i=1}^n e^{y_i}} \quad (18)$$

Where  $p$  refers to the calculated probability,  $\{w_1, w_2, w_3, \dots, w_t\}$  refers to the training words matrix, and  $y_i$  is the normalized activation value.

Table 2 illustrates the optimized architecture of the LSTM deep learning model. The cells highlighted using the green color are for visualizing the added layers on the initial architecture.

**Table 2. Optimized architecture of the LSTM deep learning model**

Layer	Output Shape	Parameters No.
Embedding_1	(500, 100)	5000000
Spatial_dropout1d_1	(500, 100)	0
Bidirectional_LSTM_1	(500, 128)	84480
Attention_1	(128)	628
Droupout_1	(128)	0
Batch_normalization_1	(128)	512
Dense_1	(64)	8256



From table 2, we notice that after modifying the LSTM layer from unidirectional to bidirectional, the time steps for every hidden state used within the LSTM layer doubled from (64) to become (128). As well as, the number of the LSTM trainable parameters doubled from (42240) to become (84480), which emphasizes the more knowledge the model gained utilizing the bidirectional technique. Figure 8 illustrates the optimized model for text classification.

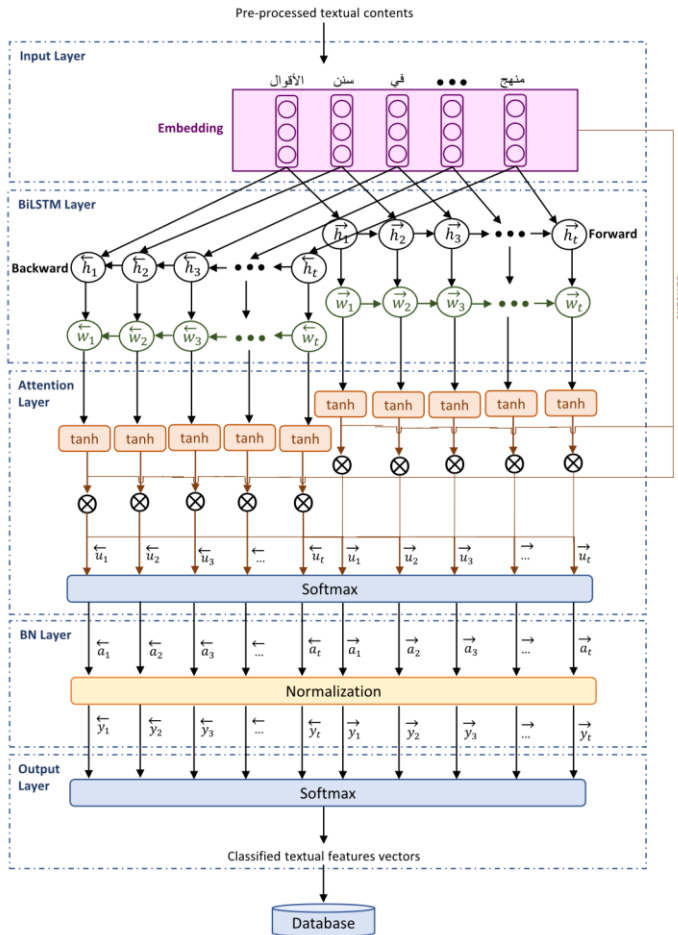


Figure 8. The optimized model for text classification

From figure 8, we notice that both the BiLSTM hidden states and weights are doubled compared with the hidden states and weights in figure 7. We also notice that the attention layer reads the textual contents again through entering it into the “TanH” activation function and then, multiply it with the output LSTM weights. Thereby, all the output weights from the BiLSTM layer denoted as  $w_t$ , get updated to  $u_t$ . The updated weights pass into the first “Softmax” activation function to calculate the attention weights denoted as  $a_t$ . Afterward, the batch normalization layer takes the attention weights and normalize them to generate the normalized weights denoted as  $y_t$ . The final adjusted normalized weights enter the second “Softmax” activation function to output the classified textual features vectors and save them in a database.

## 2) SIMILARITY MEASUREMENT

To measure the similarity among the textual contents, we calculate the distance between the features vectors of all classified text, which is already saved in the database, and the feature vector of the textual contents associated with the user-chosen query image as illustrated in figure 9.

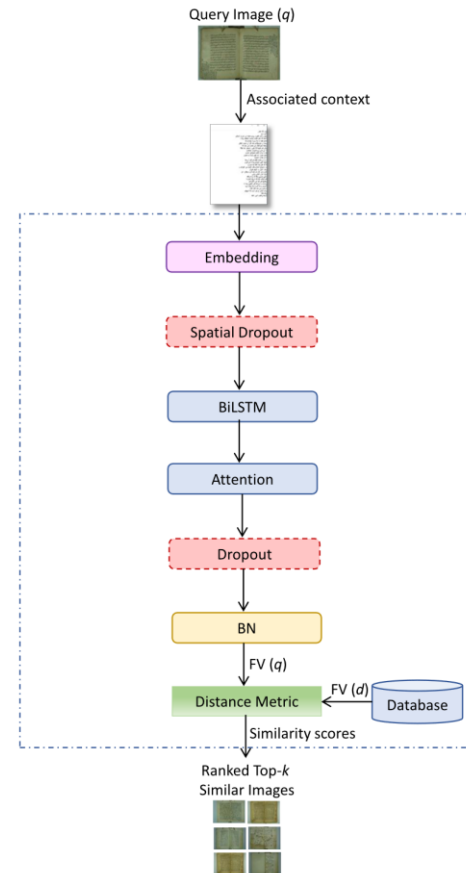


Figure 9. The similarity measurement of the classified text

From figure 9, we notice that the textual feature vector of the user query image is taken from the Batch Normalization (BN) layer. The distance is then measured between the textual feature vector of the user query image denoted as  $FV(q)$  and between all the other textual features vectors saved previously in the database denoted as  $FV(d)$ . As much as the textual contents are similar, as much as the computed similarity score will be close to (1). In contrast, as much as the textual contents are different, as much as the computed similarity score will be close to (0). The generated similarity scores are ranked in a descending order, and according to the highest measured similarities, the top- $k$  similar images to the user query image will display in the final output result.

## C. IMAGE RETRIEVAL USING FUSION MODEL

Since most of the images are neither purely visual nor completely textual. Thus, fusing both visual and textual

models into one fusion model is a crucial step for improving the performance of each model separately [60]. Therefore, we begin in this section by fusing the models into one model for image classification. Afterward, the similarity scores are measured to retrieve similar images using the best experimented fusion model for image classification.

### 1) IMAGE CLASSIFICATION

After extracting the visual features from the images of the ancient Arabic manuscripts' dataset using the pre-trained deep CNN. As well as, after extracting the textual features from the same images using the optimized BiLSTM deep learning model, we experimented the fusion of both models using three different fusion methods named: decision-level fusion, features-level fusion, and score-level fusion; looking for the most accurate fusion method that increases the classification accuracy of each used model individually.

#### 1- Decision-level fusion

The decision-level fusion method base on the predicted labels resulted from the final classification dense layers. Hence, both the inputs and the outputs from the decision-level fusion model are the classified labels. The main purpose of the decision-fusion model is to classify the images according to the new fused labels more accurately. The decision-level fusion model illustrated in figure 10 is applied to the predicted (64) manuscripts' labels from both the visual pre-trained CNN, and from the textual BiLSTM optimized deep learning model.

#### 2- Features-level fusion

The features-level fusion method base on fusing the extracted features from the deep learning models. Hence, both the inputs and the outputs from the features-level fusion model are the extracted features. The main purpose of the features-level fusion model is to obtain

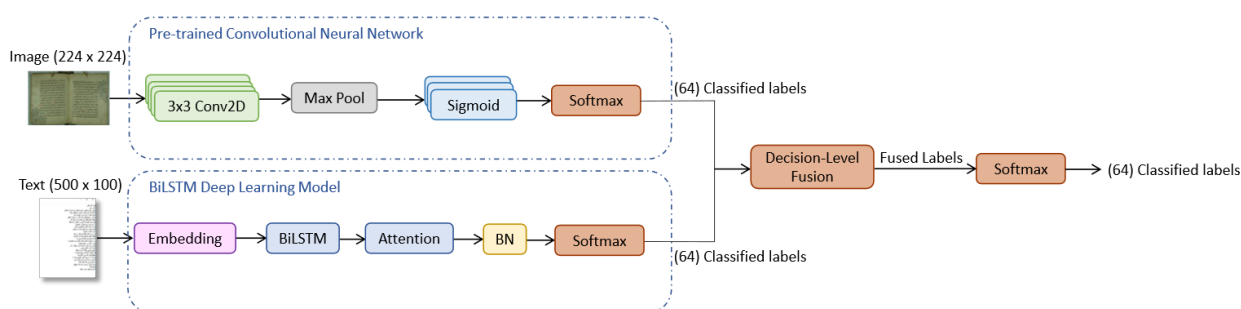


Figure 10. Decision-level fusion model

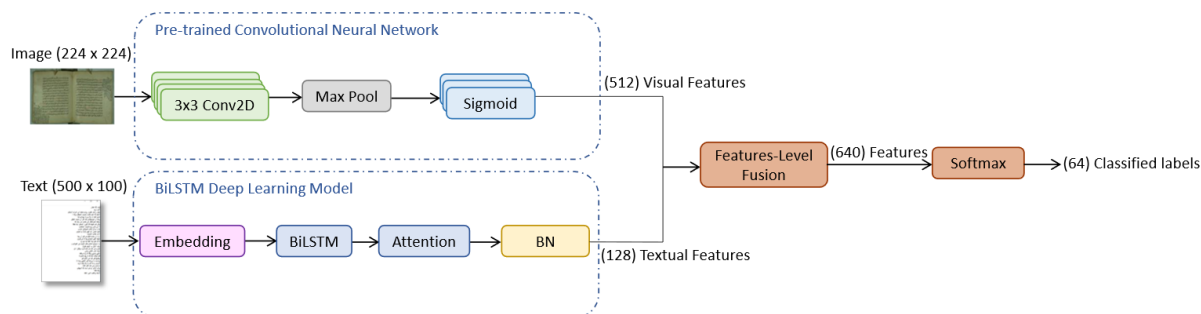


Figure 11. Features-level fusion model

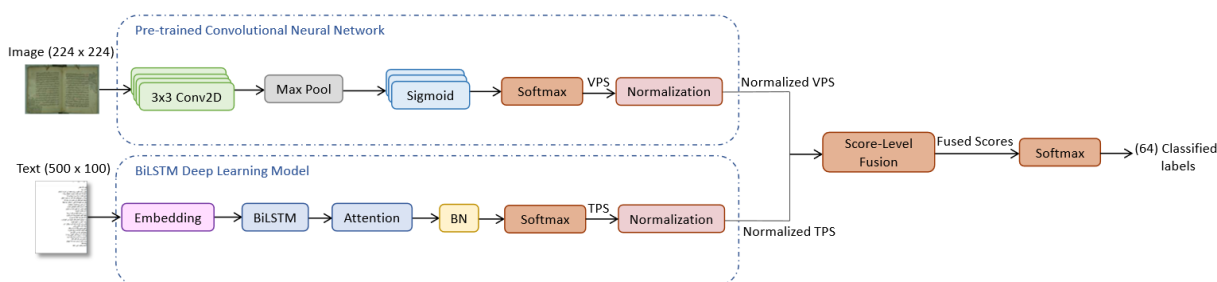


Figure 12. Score-level fusion model

new fused features that are better characterizing the images. The features-level fusion model illustrated in figure 11 is applied to the extracted visual and textual features from both the pre-trained CNN and the BiLSTM deep learning models.

### 3- Score-level fusion

The score-level fusion method accepts the probability scores from the visual and from the textual models' classifiers. Then it fuses the scores into one fusion model using a rule to better classify the images according to the fused scores. The classifiers' probability scores are heterogeneous because one classifier is visually based, while the other is textually based. Thus, it is necessary to normalize the scores before performing the fusion to keep them within a similar numerical scale [83].

The score-level fusion model illustrated in figure 12 is applied to the generated Visual Probability Scores (VPS) from the Softmax classifier of the pre-trained CNN. As well as, on the generated Textual Probability Scores (TPS) from the Softmax classifier of the optimized BiLSTM deep learning model. According to Chitroub [83], the scores obtained from the classifiers' probabilities are including the substantial details of the input data.

From figure 10, we notice that the fusion model accepts two inputs. The first input of shape (224 x 224) colored pixels representing the images. While the second input of shape (500 x 100) representing the textual contents. Then, the model takes the predicted outputs from both final "Softmax" classification dense layers in the visual model and in the textual model to merge them in one decision-level fusion model that predicts the labels of both models. The outputs from the decision-level fusion model are the fused (64) manuscripts classified labels.

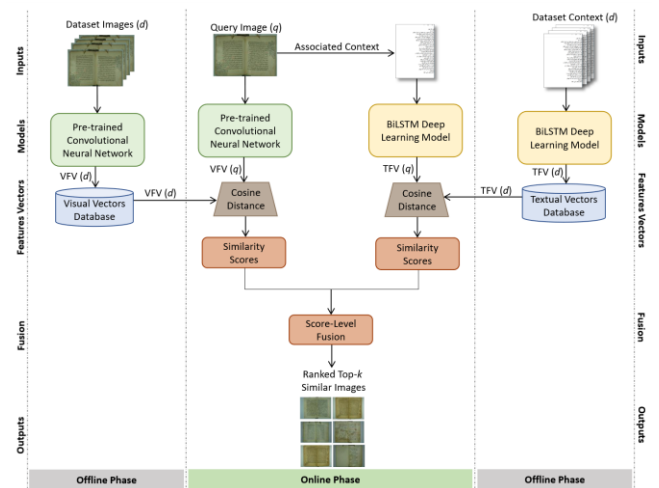
From figure 11, we notice that the fusion model takes the extracted visual features from the pre-trained CNN with (512) dimensions. As well as, it takes the extracted textual features from the optimized BiLSTM deep learning model with (128) dimensions to concatenate them in one feature-level fusion model. Therefore, the output shape from the concatenation layer equals (512 visual feature vector + 128 textual feature vector = 640 fused feature vector). The "Softmax" classification layer is added after the concatenation layer to predict the labels of the fused features from both the visual and the textual models. The outputs from the features-level fusion model are the classified (64) manuscripts labels according to the fused features vectors.

From figure 12, we notice that the score-level fusion model accepts visual and textual inputs to generate the VPS from the CNN classifier and the TPS from the BiLSTM classifier. These probabilities are generated using the "model.predict()" built-in function from "Keras" deep learning library to predict the probabilities of the inputs to the models. Afterward, we normalize the generated probability scores using the "TanH" score normalization function. Then, we fuse the normalized scores using one score-level fusion model. To do the fusion,

we should apply some rules to the scores, such as the max rule, min rule, or sum rule. The fused scores are then entered into a final "Softmax" layer to get the (64) classified labels according to the score-level fusion model.

### 2) SIMILARITY MEASUREMENT

After classifying the images, the top-k similar images to a user query image are retrieved through fusing the similarity scores from both the visual and the textual models, as illustrated in figure 13.



**Figure 13. The similarity measurement approach using the score-level fusion model**

From figure 13, we notice that the image retrieval step is performed in three phases. Two phases are accomplished offline. The first offline phase generates the Visual Features Vectors from the entire dataset images using the deep pre-trained CNN, denoted as  $VfV(d)$ , and saves them in a database. Similarly, the second offline phase generates the Textual Features Vectors from the entire dataset context using the optimized BiLSTM deep learning model, denoted as  $TFV(d)$ , and saves them in a database.

During the online phase, the user enters a query image. Then, its associated text is retrieved to get both the visual and the textual features vectors of the query image. Afterward, the model generates two different similarity scores, one from the visual model and the other from the textual model. The scores are fused using the sum-rule score-level fusion model to obtain one fused similarity score from both visual and textual models. The final fused similarity scores are then ranked in a descending order to retrieve the top-k similar images from the ranked list.

## IV. EXPERIMENTS AND TESTS RESULTS

This section begins by explaining the settings used for conducting the experiments. Then, we experiment the image classification and retrieval according to the images' deep visual features. In addition, we experiment the image

classification and retrieval according to the deep textual features. Finally, we test various fusion methods for fusing both the visual and the textual models into one model to reach the optimized image retrieval system.

### A. EXPERIMENTS' SETTINGS

The testing machine is "ABS Battelbox" PC, including Ubuntu 16.04 Operating System and Intel Core i7-9700K 3.60 GHz with (8) core processors and Nvidia Gefore RTX 2080. Regarding the used software to run the experiments, we utilized Python (3.7.4) programming language with the Jupyter notebook web application interface. Both the CUDA (10.0.130) toolkit and the cuDNN version (7) deep neural network libraries were utilized to run the code on the GPU instead of running it on the CPU. The link to the complete code is added to github<sup>7</sup>.

After setting the hardware and software for the experiments, we used the same dataset in [84]. It consists of (8638) images belonging to (64) ancient Arabic manuscripts collected from the "wqf"<sup>8</sup> website. The manuscripts written using six different types of Arabic calligraphies. Some of the historical Arabic manuscripts written within the time periods between (1004) and (1384) "Hijri" date. While, the time periods of other manuscripts in the dataset were known. They might be written before the "Hijra" of the Prophet "Mohammed" where the Islamic Hijri date gets started, or the writer might didn't indicate the date. The number of images within each manuscript is ranging from (6) to (381) colored images. Thus, the "weighted categorical cross-entropy" algorithm used to weight the classes including the minimum number of images more than the classes including large significant number of images. This algorithm is used to fix the unbalanced images within the dataset. Because the unbalanced ratio of images may result in an accuracy paradox problem where the generated results could be biased and overfitted to the manuscript including the largest number of images [85]. Moreover, an offline data augmentation was implemented to generate new modified versions from the original images with a random rotation, zooming, and shearing. This augmentation technique helps in increasing the training portion of the dataset, as well as, it generalizes the model on the images. Most of the images were having (2160 x 1440) pixels size. But, we resized them into (224 x 224) pixels to prepare them for entering the deep learning models.

According to the results reached in [86] regarding the best categorization of the dataset, we categorized the entire dataset into 70% training, and 30% divided equally between the testing and the validation subsets.

### B. IMAGE CLASSIFICATION AND RETRIEVAL ACCORDING TO THE VISUAL FEATURES

The dataset images classified according to the (64) manuscripts using four pre-trained deep learning models as following:

- MobileNet\_V1\_100\_244
- ResNet\_V2\_50
- DenseNet\_201
- VGG\_19

The used learning hyperparameters were inspired by the study done in [84] that is focusing on tuning the learning hyperparameters for the visual deep convolutional neural networks. Therefore, we used (1e-6) learning rate with a final "Softmax" classification dense layer, including "Adam" optimizer and "weighted categorical cross-entropy" loss. Three "Sigmoid" dense layers added before the final "Softmax" classification dense layer as they increase the classification accuracy. The number of neurons on the penultimate layer equals (512) neurons. The model trained using (10) learning cycles/ epochs and the batch size for training the model is set to (64).

To evaluate the four experimented deep learning models for image classification, the Validation Accuracy (VA), Average Precision (AP), Average Recall (AR), and Average F-Score (AF) computed after running the model ten learning cycles.

Table 3 summarizes the evaluation of the four pre-trained deep learning models for image classification. The highest generated metrics were highlighted in bold, while the lowest parameters highlighted using italics for easier visualization of the results.

From table 3, we notice that both the MobileNet\_100 and the ResNet\_50 deep learning models accomplished around 97% successful classification of the images according to the manuscripts. However, the VGG\_19 pre-trained deep learning model outperformed the other three deep learning models in classifying the dataset images because it recorded 98% successful classification. In contrast, the DenseNet\_201 deep learning model was the worst model for classifying the images.

**Table 3. Image classification according to the manuscript**

Evaluation Parameter	MobileNet_100	ResNet_50	DenseNet_201	VGG_19
Validation Accuracy	0.9744	0.9752	<i>0.4679</i>	<b>0.9768</b>
Average Precision	0.9782	0.9733	<i>0.4954</i>	<b>0.9836</b>
Average Recall	0.9756	0.9724	<i>0.4802</i>	<b>0.9811</b>
Average F-Score	0.9762	0.9719	<i>0.4265</i>	<b>0.9817</b>

<sup>7</sup>[https://github.com/ManalKhayyat/Arabic\\_Manuscripts\\_Image\\_Retrieval.git](https://github.com/ManalKhayyat/Arabic_Manuscripts_Image_Retrieval.git)

<sup>8</sup> <http://wqf.me/?p=15619>



Therefore, we decided to use the deep pre-trained VGG\_19 CNN for measuring the similarities among images. The “Sigmoid” dense layer with (512) feature size used to generate the images features vectors. The feature vector of the user query image along with the saved features vectors of all other images in the dataset used to measure the similarity among them utilizing three different distance metrics. Then, the model retrieved the top- $k$  similar images from the entire dataset.

Table 4 lists the calculated mean accuracy for each top- $k$  from the entire dataset utilizing the deep VGG\_19 CNN.

**Table 4. Retrieval mean accuracy per  $k$  similar images**

Metric	Top-10	Top-25	Top-50	Top-100
Manhattan	97.2707%	96.4376%	95.9357%	95.5366%
Euclidean	97.2699%	96.4504%	<b>95.9522%</b>	95.5637%
Cosine	<b>97.2751%</b>	<b>96.4512%</b>	95.9481%	<b>95.5640%</b>

From table 4, we conclude that as much as the number of  $k$  images is smaller, as much as the model records higher retrieval mean accuracy. The three tested distance metrics were all performing well in measuring the similarities, and their generated values were close to each other. However, the generated mean accuracies by the Manhattan distance metric were the lowest compared with the other two distance metrics. In contrast, the mean accuracies generated using the Cosine distance metric were the highest. Therefore, we decided to use the VGG19 classification model combined with the Cosine distance metric in the image retrieval system according to the visual features.

### C. IMAGE CLASSIFICATION AND RETRIEVAL ACCORDING TO THE TEXTUAL FEATURES

The initial LSTM deep learning model compiled using “Adam” optimizer with the “weighted categorical cross-entropy” loss function. The learning rate equals (1e-6), the dropout rate equals (0.5) and the spatial dropout rate equals (0.3). The final classification dense layer includes “Softmax” activation function with (64) labels. The batch size for training the model is set to (128), then the model trained using (10) learning cycles on both non-cleaned and cleaned textual data and its generated evaluation parameters summarized in table 5.

**Table 5. LSTM classification according to the manuscript**

Evaluation Parameter	Non-Cleaned Text	Cleaned Text
Validation Accuracy	0.3573	0.5720
Average Precision	0.2662	0.4815
Average Recall	0.2662	0.4815
Average F-Score	0.2662	0.4815

<sup>9</sup> [https://keras.io/api/layers/merging\\_layers/](https://keras.io/api/layers/merging_layers/)

From table 5, we notice that the generated evaluation parameters by the initial LSTM deep learning model on both non-cleaned and cleaned text were not satisfying. Thus, we optimized the classification model through making the LSTM layer bidirectional (BiLSTM), and through adding both the “Attention” and the “Batch Normalization” (BN) layers. Table 6 summarizes the evaluation of the optimized BiLSTM deep learning model on both non-cleaned and cleaned textual data.

**Table 6. Optimized BiLSTM classification according to the manuscript**

Evaluation Parameter	BiLSTM + Attention	BiLSTM + Attention + BN
	Non-Cleaned Text	
Validation Accuracy	0.5107	0.6333
Average Precision	0.5304	0.6452
Average Recall	0.5304	0.6498
Average F-Score	0.5304	0.6475
	Cleaned Text	
Validation Accuracy	0.7249	0.8381
Average Precision	0.7454	0.8546
Average Recall	0.7454	0.8484
Average F-Score	0.7454	0.8515

We notice that the classification accuracy of non-cleaned text improved from 36% (as illustrated in table 5) to 63% (as illustrated in table 6) after optimizing the LSTM deep learning model. Similarly, the validation accuracy of the cleaned text improved from 57% to become 84%, which is a successful and satisfying result. Moreover, we notice that the cleaned textual data are generating higher evaluation parameters than the non-cleaned textual data. Therefore, we will use the optimized BiLSTM architecture, including both the attention and the batch normalization layers, to classify the cleaned textual data for measuring the similarities among texts.

Table 7 illustrates the calculated mean accuracy for each top- $k$  texts from the entire dataset utilizing the optimized BiLSTM deep learning model.

**Table 7. Retrieval mean accuracy per  $k$  similar text**

Metric	Top-10	Top-25	Top-50	Top-100
Manhattan	63.7192%	57.9827%	53.9149%	49.7222%
Euclidean	64.0321%	58.6654%	54.9090%	<b>51.0960%</b>
Cosine	<b>64.7836%</b>	<b>59.2452%</b>	<b>55.1781%</b>	51.0182%

From table 7, we conclude that the Cosine distance metric is generating the highest retrieval results. Thus, we decided to use it along with the BiLSTM deep learning classification model.

### D. IMAGE CLASSIFICATION AND RETRIEVAL ACCORDING TO THE FUSION MODELS

“Keras”<sup>9</sup> deep learning library provides several built-in merge layers that can be used with the fusion model. We tested four

different merge layers with the decision-level fusion model for classifying images and texts according to the manuscripts looking for the merge layer that will increase the classification accuracy. The evaluation results of the developed decision-level fusion model are summarized in table 8.

**Table 8. Evaluation of the decision-level fusion model**

Evaluation	Concatenate	Maximum	Average	Multiply
VA	0.9470	0.9603	<b>0.9727</b>	0.9355
AP	<b>0.9741</b>	0.9468	0.9408	0.9501
AR	0.9598	<b>0.9680</b>	<b>0.9680</b>	0.9633
AF	<b>0.9670</b>	0.9573	0.9542	0.9567

From table 8, we notice that the decision-level fusion model generated close results using the four tested merge Keras built-in layers and that the results were all around 96%. The reached results are higher than the manuscript classification using the textual model, which reached 0.8381% validation accuracy. But they are not higher than the manuscript classification using the visual model, which reached 0.9768% validation accuracy. Therefore, we experimented another fusion type called features-level fusion.

The features-level fusion model developed using the “Concatenate” merge layer from the “Keras” deep learning library only. Because all the other types of merge layers require the inputs to the merge layer to be of the same size. While, unlike the previous tested decision-level fusion model, the features vectors are of different sizes. Thus, we used the “Concatenate” merge layer in conjunction with the developed features-level fusion model for classifying both images and text according to the manuscripts. The model evaluation results are summarized in table 9.

**Table 9. Evaluation of the features-level fusion model**

Evaluation Parameter	Concatenate
Validation Accuracy	0.9836
Average Precision	0.9745
Average Recall	0.9836
Average F-Score	0.9790

From table 9, we notice that the features-level fusion model generated better results than the decision-level fusion model. That is because the evaluation parameters of the features-level fusion model were above 97%. Whereas, the evaluation parameters using the decision-level fusion model were around 96%.

The score-level fusion method with three different rules was also tested, and its generated results summarized in table 10.

**Table 10. Evaluation of the score-level fusion model**

Evaluation	Min Rule	Max Rule	Sum Rule
VA	0.9837	0.9804	<b>0.9896</b>
AP	0.9885	<b>0.9956</b>	0.9906

AR	0.9906	0.9904	<b>0.9959</b>
AF	0.9895	0.9930	<b>0.9916</b>

From table 10, we notice that the score-level fusion model using the simple-sum rule generated higher results than using the min and max rules. Furthermore, we notice that the evaluation parameters are higher than the parameters generated using the decision-level and the features-level fusion models. Because the evaluation parameters of the score-level fusion model were all above 98%, which is higher than the classification using the images visual and textual models separately.

After extensive experimentation of multiple fusion methods, we reached that the best performing fusion method is the simple sum score-level fusion. That is because it recorded the highest classification accuracy according to the manuscripts. Therefore, we decided to use it for fusing the classification of images.

Table 11 highlights the evaluation parameters of the image classification using the deep VGG\_19 CNN, the optimized BiLSTM deep learning model, and the score-level fusion model.

**Table 11. Image classification using visual, textual, and fusion models**

Evaluation	Visual	Textual	Fusion
VA	0.9768	0.8381	<b>0.9896</b>
AP	0.9836	0.8546	<b>0.9906</b>
AR	0.9811	0.8484	<b>0.9959</b>
AF	0.9817	0.8515	<b>0.9916</b>

From table 11, we conclude that the score-level fusion of both the visual and the textual models improved the classification accuracy more than using each model separately. That is because the fusion model has more information about the images than each single model, which increased its ability to identify the images and classifying them.

Table 12 summarizes the computed mean accuracy using the Cosine distance metric per the retrieved top-k similar images using the VGG19 deep CNN, optimized BiLSTM deep learning model, and using the score-level fusion model.

**Table 12. Mean accuracy per k similar images using visual, textual, and fusion models**

Top-k	Visual	Textual	Fusion
Top-10	97.2751%	64.7836%	<b>98.1819%</b>
Top-25	96.4512%	59.2452%	<b>97.3695%</b>
Top-50	95.9481%	55.1781%	<b>97.0304%</b>
Top-100	95.5640%	51.0182%	<b>96.7362%</b>

From table 12, we notice that the textual model was including the lowest accuracies for image retrieval. In contrast,














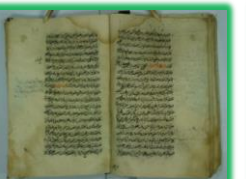


the score-level fusion model was including the highest accuracies for retrieving images.

After deciding to use the score-level fusion model of the deep pre-trained VGG19 CNN and the optimized BiLSTM deep learning model, we illustrate in table 13 an example of the output results from the same input query image using the visual VGG19 CNN, the textual BiLSTM deep learning model, and the proposed score-level fusion model. The similarity score, along with the manuscript id of each retrieved image is displayed under it. The correctly retrieved images are highlighted using the green color, while the images retrieved from different manuscripts than the query image are highlighted using the red color.

From table 13 we notice that the score-level fusion model was able to 100% successfully retrieve the top-5 similar images to the query input image. In addition, we notice that there is a variation in the generated similarity scores and that they are in a descending order, which indicates the ability of the model to visualize and distinguish between the Arabic manuscripts' images successfully.

Table 14 compares theoretically the proposed method with the existing state-of-the-art methods worked on the Arabic manuscripts. The papers divided according to the features as either handcrafted or deep learning features. The comparison is relative because the datasets used to conduct the studies and the employed classification techniques are different. Moreover, some papers worked on the classification only.

**Table 13.** Query input image and its ranked top-5 similar output images

<p style="text-align: center;"><b>Input Image</b></p>  <p style="text-align: center;">Manuscript ID: 26</p>				
<p style="text-align: center;"><b>Retrieval from the visual-based model</b></p> <p style="text-align: center;">Retrieval time: 0.279 seconds    Top-5 matching rate: 80%</p>				
				
Similarity score: 1.0000000 Manuscript ID: (26)	0.9879376 (26)	0.9793003 (26)	0.9744199 (26)	0.9727287 (16)
<p style="text-align: center;"><b>Retrieval from the textual-based model</b></p> <p style="text-align: center;">Retrieval time: 0.221 seconds    Top-5 matching rate: 60%</p>				
				
Similarity score: 0.9597840 Manuscript ID: (26)	0.9593968 (41)	0.9573525 (26)	0.9567745 (35)	0.9564679 (26)
<p style="text-align: center;"><b>Retrieval from the fusion model</b></p> <p style="text-align: center;">Retrieval time: 0.456 seconds    Top-5 matching rate: 100%</p>				
				
Similarity score: 0.9759797 Manuscript ID: (26)	0.9759594 (26)	0.9759564 (26)	0.9759060 (26)	0.9758681 (26)

**Table 14. Relative comparison with the state-of-the-arts methods addressed the Arabic manuscripts recognition/retrieval**

	Reference# (Year)	Problem Domain	Input Data	Feature Extraction	Classification	Dataset		Result
						Name	Size	
<b>Handcrafted Features</b>	[8] (2011)	Retrieve Arabic manuscript images using the textual features	Subword images	Word spotting through LSI	Singular Value Decomposition (SVD) algorithm	Sahih Al-Bukhari and Mawaqet Al-Haj wa Al-Umra	Two pre-scanned ancient Arabic manuscripts	Recall: 78.8% using the circular polar grid features set
	[9] (2009)	Recognize handwritten Arabic letters	Image containing one Arabic character	LeNet CNN	Error signal function	Manually collected Arabic texts with different handwriting styles.	758 segmented Arabic characters.	There is a need to handle the non-textual parts within manuscripts as images not as text
	[19] (2019)	Arabic manuscript images retrieval using visual features	Complete image	SURF and BRISK CBIR	Hamming distance and Sum of square distance	Manually collected Arabic manuscripts	1670 images	61% overall accuracy using SURF technique and 37% using BRISK
	[33] (2017)	Retrieve Arabic manuscript images using text search	Arabic text word	Optical Shape Recognition (OSR)	Index pages in XML file	Ibn Sina" database	51 historical manuscripts images	NA
	[34] (2018)	Retrieve Arabic manuscript images using the textual features	Textual contents	Run length, edge direction, and edge hinge	*ML: K-nearest neighbor	KERTAS	2505 images	94.77% accuracy with predefined folds and 42.31% accuracy with random train/test split using (50×50) size
	[35] (2017)	Retrieve Arabic manuscript images using the textual features	Textual contents	Modified contour-based feature and local key point descriptors	Cosine or Chi-square distance metric	IHP and KHATT ancient Arabic manuscripts	IHP contains 2313 images and KHATT contains 4000 images	88.9% and 73% classification accuracy using KHATT and IHP datasets respectively. 34.6% retrieval accuracy of the top10 images using M-CBF
	[36] (2016)	Arabic texts recognition	Handwritten text image	SVMs or ASMs	Levenshtein distance	IESK-arDB SynWords manually collected dataset	50,000 famous Arabic words vocabularies.	Recall: 65.1279% Precision: 64.716%
	[38] (2009)	Classify Arabic manuscript images	Word binary image	Feedforward technique of multi-language processing neural network	Error signal function	One historical Arabic manuscript named "كشف اللثام عن وجه الإسلام".	27 images	89.3% average accuracy.
	[39] (2014)	Arabic texts extraction	Handwritten Arabic text image	Word spotting through FCM	Euclidean distance	Arabic Handwritten Data-Base	25 images	Extraction rate: 84.8%
	[40] (2015)	Retrieve Arabic manuscript images using textual features	Arabic word image	Word spotting through BoWFs	Histogram intersection and Earth movers distance	Manually scan and collect 120 ancient Arabic images	120 manually scanned ancient Arabic images.	Recall: 50% Precision: 89.60%



	[41] (2016)	Arabic texts extraction	Subset of historical text images	OIC transparent neural network	Compute the upper and lower limits of diacritic point	KHATT Arabic dataset	100 handwritten Arabic text.	Extraction rate: 74%
	[42] (2019)	Retrieve Arabic manuscript images using textual features	Image contains one Arabic word	SURF	Compare points of interest and compute the distance using the Euclidean or the Mahalanobis metrics.	HADARA80P	20 images	95.27% recognition accuracy.
	[52] (2014)	Arabic characters recognition	Binary Arabic word	GA-HS fused model	Harmony search character algorithm	Manually collected dataset consisting of 4500 Arabic words and ADAB dataset	4500 Arabic words (24,960 individual characters) to conduct the study.	Recognition rate: 93.6% using manual collected dataset Recognition rate: 94.68%-96.33% using ADAB
<b>Deep Learning Features</b>	[46] (2020)	Classify Arabic text	Arabic articles	Attention-GRU	Sigmoid and binary cross entropy	SANAD Arabic dataset	8836 articles.	96.94% classification accuracy.
						NADiA Arabic datasets	485k articles.	88.68% classification accuracy.
	Proposed method	Retrieve Arabic manuscript images using both visual and textual features	Text-based image	Pretrained VGG19 CNN	Softmax dense layer	Collected ancient Arabic manuscripts	8638 images	97.68% classification accuracy and 97.28% mean accuracy on the top-10 image retrieval.
				Attentional BiLSTM				83.81% classification accuracy and 64.78% mean accuracy on the top-10 image retrieval.
				Score-level fusion of VGG19 and BiLSTM				<b>98.96%</b> classification accuracy and <b>98.19%</b> mean accuracy on the top-10 image retrieval.

While other papers worked on both the classification and retrieval.

From table 14, we notice that seven papers [8], [19], [33], [34], [35] [40], and [42] addressed the retrieval of the Arabic manuscript' images. However, all of them used the handcrafted features.

For the classification, we found authors used traditional approaches such as the index pages in XML files, as well as, they used classical neural networks. Regarding the used datasets of Arabic manuscripts, there are Sahih Al-Bukhari, Mawaqet Al-Haj wa Al-Umra, KHATT dataset, Ibn Sina database, ADAB, SANAD, NADiA, and manually collected Arabic manuscripts.

Some of the researchers indicated the size of the used data as the number of manuscripts images. While, other researchers indicated the number of the Arabic characters within the manuscripts images. The highest recorded recognition rate was using the SANAD Arabic dataset in [46] as 96.94%. However, we recorded 98.96% recognition and classification accuracy using the fusion model. Furthermore, we notice that one study [52] used fusion model but, it utilized handcrafted features. Therefore, we admit that we reached a novel

approach that proves its success, among other existing methods.

## V. CONCLUSION

This study aims to classify and retrieve the top- $k$  similar images to a user query image. We begin by transfer learning from four deep pre-trained CNNs named: MobileNetV1, DenseNet201, ResNet50, and VGG19 to classify and to extract the visual features from the dataset images. We found-out that the deep pre-trained VGG19 CNN is more rigid in its performance than the other tested pre-trained CNNs.

Moreover, we developed an LSTM deep learning model for classifying and for extracting the textual features from the images of the Arabic manuscripts. The initial results from the LSTM deep learning model were not satisfying. Thus, we optimized it through using the bidirectional technique, as well as, through adding the attention and the batch normalization layers. After classifying the images, we experimented three different distance metrics looking for the most accurate method for measuring the similarity scores. The three-distance metrics were Manhattan, Euclidean, and Cosine. We reached that the Cosine distance metric is the most accurate method in computing the similarity scores.

Then, we experimented fusing both the visual and the textual models using the decision-level, the features-level, and the score-level fusion methods. We concluded that the score-level fusion method using the simple-sum rule of both the VGG19 CNN and the optimized BiLSTM deep learning model is outperforming the other experimented fusion methods, and it's generating higher evaluation parameters than each used model separately.

The proposed fusion model evaluated by computing the accuracy, precision, recall, and the F-score. In addition, we compared our method with state-of-the-art methods. Finally, we admit the success of the proposed method in classifying and retrieving the most similar images to a user query image.

## ACKNOWLEDGMENTS

The authors acknowledge with thanks the Deanship of Scientific Research (DSR) technical and financial support for funding the project.

## REFERENCES

- [1] A.M. Marshall, and S. Gunasekaran, "A Survey on Image Retrieval Methods," *CIET-ECE DEPT Conf.*, pp. 1–16, 2014.
- [2] M. Al-Ayyoub, A. Nuseir, K. Alsmearat, Y. Jararweh, and B. Gupta, "Deep learning for Arabic NLP: A survey," *J. Comput. Sci.*, vol. 26, pp. 522–531, 2018.
- [3] V. Nigam, "Understanding Neural Networks. From neuron to RNN, CNN, and Deep Learning," Lect, 11-Sep-2018. [Online] *Towards Data Science*. Available at: <https://towardsdatascience.com/understanding-neural-networks-from-neuron-to-rnn-cnn-and-deep-learning-cd88e90e0a90> [Accessed 17 Apr. 2019].
- [4] S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEMORY," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [5] A. M. Alayba, V. Palade, M. England, and R. Iqbal, "A Combined CNN and LSTM Model for Arabic Sentiment Analysis," *Springer International Publishing*, vol. 12, pp. 179–191, 2018.
- [6] H. A. Al-Muzaini, T. N. Al-Yahya, and H. Benhidour, "Automatic Arabic Image Captioning using RNN-LSTM-Based Language Model and CNN," (*IJACSA*) *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 6, pp. 67–73, 2018.
- [7] F. Alaei, A. Alaei, U. Pal, and M. Blumenstein, "A Comparative Study of Different Texture Features for Document Image Retrieval," *Expert Syst. Appl.*, vol. 121, pp. 97–114, 2018.
- [8] M. H. N. Yahia, "Content-Based Retrieval of Arabic Historical Manuscripts Using Latent Semantic Indexing," *King Fahd university of Petroleum and Minerals*, pp. 01–98, 2011.
- [9] R. Al-Jawfi, "Handwriting Arabic Character Recognition LeNet Using Neural Network", *International Arab Journal of Information Technology*, vol. 6, no. 3, pp. 304–309, 2009.
- [10] Y. A. Aslandogan and C. T. Yu, "Techniques and Systems for Image and Video Retrieval," *World Wide Web Internet Web Inf. Syst.*, vol. 11, no. 1, pp. 1–19, 1999.
- [11] M. Saleem, R. Senthikumar, and T. S. Prakash, "Image Retrieval System by Automatic Annotation," vol. 1, no. 8, pp. 286–290, 2014.
- [12] A. Kumar et al., "Adapting content-based image retrieval techniques for the semantic annotation of medical images," *Comput. Med. Imaging Graph.*, vol. 49, pp. 37–45, 2016.
- [13] J. S. Hare, P. H. Lewis, P. G. B. Enser, and C. J. Sandom, "Mind the Gap: Another look at the problem of the semantic gap in image retrieval," *Multimed. Content Anal. Manag. Retr.*, p. 607–309, 2006.
- [14] K. S. Reddy and K. Sreedhar, "Image Retrieval Techniques: A Survey," *Int. J. Eng. Technol.*, vol. 7, no. 1.2, p. 215, 2018.
- [15] Y. Rui, T. S. Huang, and S. Mehrotra, "Content-Based Image Retrieval With Relevance Feedback in MARs," pp. 815–818, 2002.
- [16] S. A. Wadhwa and S. S. Kawathekar, "Techniques of Content Based Image Retrieval: A Review," *IOSR Journal of Computer Engineering (IOSR-JCE)*, pp. 75–79, n.d.
- [17] G. F. Ahmed and R. Barskar, "A Study on Different Image Retrieval Techniques in Image Processing," *Journal of Computing and Engineering (IJSCE)*, vol. 1, no. 4, pp. 247–251, 2011.
- [18] A. Dureja and P. Pahwa, "Image retrieval techniques: a survey," *International Journal of Engineering & Technology*, vol. 7, no. 2, p. 215–219, 2018.
- [19] B. Bagasi, and L. Elrefaei, "Arabic Manuscript Content Based Image Retrieval: A Comparison between SURF and BRISK Local Features," *Int. J. Comput. Digit. Syst.*, vol. 7, no. 6, pp. 355–364, 2019.
- [20] F. Chen, B. Li, and L. Li, "3D object retrieval with graph-based collaborative feature learning," *J. Vis. Commun. Image Represent.*, vol. 58, pp. 261–268, 2019.
- [21] S. Raju, K. Sreelatha, and S. Kumari, "An Efficient Approach to Improve Retrieval Rate in Content Based Image Retrieval Using MPEG-7 Features," *Adv. Intell. Syst. Comput.*, vol. 248, pp. 337–347, 2014.
- [22] C. Beecks, S. Kirchhoff, and T. Seidl, "On stability of signature-based similarity measures for content-based image retrieval," *Multimed. Tools Appl.*, vol. 71, no. 1, pp. 349–362, 2014.
- [23] F. Radenovic, G. Tolias, and O. Chum, "Fine-tuning CNN Image Retrieval with No Human Annotation," *Tpami.*, vol. 2, pp. 1–14, 2018.
- [24] W. Zhou, and J. Jia, "A Learning Framework for Shape Retrieval Based on Multilayer Perceptrons," *Pattern Recognit. Lett.*, vol. 117, pp. 119–130, 2019.
- [25] O. Seddati, S. Dupont, S. Mahmoudi, and M. Parian, "Towards Good Practices for Image Retrieval Based on CNN Features," *Proc. - 2017 IEEE Int. Conf. Comput. Vis. Work. ICCVW*, pp. 1246–1255, 2018.
- [26] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese Neural Networks for One-shot Image Recognition," *Proc. 32 Nd Int. Conf. Mach. Learn. Lille, Fr. JMLR W&CP*, vol. 37, pp. 1–8, 2015.
- [27] Y. Ge, S. Jiang, Q. Xu, C. Jiang, and F. Ye, "Exploiting representations from pre-trained convolutional neural networks for high-resolution remote sensing image retrieval," vol. 77, pp. 17489–17515, 2018.

- [28] E. Ong, S. Husain, and M. Bober, "Siamese Network of Deep Fisher-Vector Descriptors for Image Retrieval," pp. 1–12, 2017.
- [29] K. Qiu, Y. Ai, B. Tian, B. Wang, and D. Cao, "Siamese-ResNet : Implementing Loop Closure Detection based on Siamese Network," *IEEE Intell. Veh. Symp.*, pp. 716–721, 2018.
- [30] U. Chaudhuri, B. Banerjee, and A. Bhattacharya, "Siamese graph convolutional network for content based remote sensing image retrieval," *Comput. Vis. Image Underst.*, vol. 184, pp. 22–30, 2019.
- [31] K.L. Wiggers, A.S. Britto, L. Heutte, A.L. Koerich, and L.S. Oliveira, "Image Retrieval and Pattern Spotting using Siamese Neural Network," pp. 1–8, 2018.
- [32] S. Ioffe, and C. Szegedy, "Batch Normalization : Accelerating Deep Network Training by Reducing Internal Covariate Shift," pp. 1–11, 2015.
- [33] S. Al-Maadeed, F. Issawi, and A. Bouridan, "Word Retrieval System for Ancient Arabic Manuscripts," *2017 9th IEEE-GCC Conference and Exhibition (GCCCE)*, pp. 1–5, 2017.
- [34] K. Adam, A. Baig, S. Al-Maadeed, A. Bouridane, and S. El-Menshaw, "KERTAS: dataset for automatic dating of ancient Arabic manuscripts," *International Journal on Document Analysis and Recognition*, vol. 21, no. 4, pp. 283–290, 2018.
- [35] A. Asi, A. Abdalhaleem, D. Fecker, V. Märgner, and J. El-Sana, "On writer identification for Arabic historical manuscripts," *International Journal on Document Analysis and Recognition*, vol. 20, no. 3, pp. 173–187, 2017.
- [36] L. Dinges, A. Al-Hamadi, M. Elzobi, and S. El-Etribi, "Synthesis of common arabic handwritings to aid optical character recognition research," *Sensors (Switzerland)*, vol. 16, no. 3, pp. 1–25, 2016.
- [37] M. Al-Yahya, "Stylometric analysis of classical Arabic texts for genre detection," 2018.
- [38] Z. Aghbari, and S. Brook, "Word Stretching for Effective Segmentation and Classification of Historical Arabic Handwritten Documents," *Proceedings of the 2009 3rd International Conference on Research Challenges in Information Science, RCIS*, pp. 217–224, 2009.
- [39] A. Al-Dmour and F. Fraij, "Segmenting Arabic Handwritten Documents into Text lines and Words," vol. 6, no. 3, pp. 109–119, 2014.
- [40] R. A. A. Othman, "Arabic Manuscripts Analysis and Retrieval," *PhD Thesis, King Fahad Univerity of Petroleum and Minerals*, pp. 1–199, 2015.
- [41] S. Snoussi, F. Ghazouani, and Y. Wahabi, "Text lines Segmentation of Handwritten Arabic Script using Outer Isothetic Cover," *Graphics, Vision and Image Processing Journal, USA*, vol. 16, no. 1, pp. 47–55, 2016.
- [42] N. El Makhfi, "A Word Spotting Method for Arabic Manuscripts Based on Speeded Up Robust Features Technique," vol. 4, no. 6, pp. 99–107, 2019.
- [43] Y. Peng, J. Zhang, and Z. Ye, "Deep Reinforcement Learning for Image Hashing," *arXiv:1802.02904*, vol. 2, pp. 1–12, 2018.
- [44] C. Qian, T. He, and R. Zhang, "Deep Learning based Authorship Identification," *Report, Stanford University*, pp. 1–9, 2017.
- [45] R. You, Z. Zhang, Z. Wang, S. Dai, H. Mamitsuka, and S. Zhu, "AttentionXML : Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification," pp. 1–17, 2019.
- [46] A. Elnagar, R. Al-Debsi, and O. Einea, "Arabic text classification using deep learning models," *Inf. Process. Manag.*, vol. 57, pp. 102–121, 2020.
- [47] G. Liu, and J. Guo, "Bidirectional LSTM with attention mechanism and convolutional layer for text classificatio," *Neurocomputing.*, 2019.
- [48] J. Du, L. Gui, R. Xu, and Y. He, "A Convolutional Attention Model for Text Classification," vol. 3, pp. 183–195, 2018.
- [49] T. Liu, S. Yu, B. Xu, and H. Yin, "Recurrent networks with attention and convolutional networks for sentence representation and classification," *Appl. Intell.*, 2018.
- [50] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical Attention Networks for Document Classification," *Proc. NAACL-HLT.*, pp. 1480–1489, 2016.
- [51] S. Gao, A. Ramanathan, and G. Tourassi, "Hierarchical Convolutional Attention Networks for Text Classification," *Proc. Of the 3rd Work. Represent. Learn. NLP.*, pp. 11–23, 2018.
- [52] M. Y. Potrus, U. K. Ngah, and B. S. Ahmed, "An evolutionary harmony search algorithm with dominant point detection for recognition-based segmentation of online Arabic text recognition," *Ain Shams Eng. J.*, vol. 5, no. 4, pp. 1129–1139, 2014.
- [53] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "CNN-RNN: A Unified Framework for Multi-label Image Classification," *J. Japan Soc. Cancer Ther. 13th Cong.*, pp. 245–246, 2016.
- [54] U. Sharif, Z. Mehmood, T. Mahmood, M.A. Javid, A. Rehman, and T. Saba, "Scene analysis and search using local features and support vector machine for effective content-based image retrieval," *Artif. Intell. Rev.*, vol. 52, pp. 901–925, 2018.
- [55] L. Guo, D. Zhang, L. Wang, H. Wang, and B. Cui, "CRAN : A Hybrid CNN-RNN Attention-Based Model for Text Classification," *Springer International Publishing*, 2018.
- [56] A. Koesdwiady, R. Soua, and F. Karray, "Improving Traffic Flow Prediction With Weather Information in Connected Cars: A Deep Learning Approach," pp. 0018–9545, 2016.
- [57] J. Li, T. Qiu, C. Wen, K. Xie, and F. Wen, "Robust Face Recognition Using the Deep C2D-CNN Model Based on Decision-Level Fusion," vol. 18, pp. 1–27 (2018).
- [58] Y. Su, K. Zhang, J. Wang, and K. Madani, "Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion," vol. 19, pp. 1–15, 2019.
- [59] Y. Xie, J. Zhang, Y. Xia, M. Fulham, and Y. Zhang, "Fusing Texture, Shape and Deep Model-Learned Information at Decision Level for Automated Classification of Lung Nodules on Chest CT," *Inf. Fusion.*, pp. 1566–2535, 2017.

- [60] P. Liu, J. Guo, C. Wu, and D. Cai, "Fusion of Deep Learning and Compressed Domain features for Content Based Image Retrieval," pp. 1057–7149, 2017.
- [61] D. Sudha, and M. Ramakrishna, "Comparative Study of Features Fusion Techniques," *2017 Int. Conf. Recent Adv. Electron. Commun. Technol.*, vol. 39, pp. 235–239, 2017.
- [62] H. Suk, S. Lee, and D. Shen, "Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis," *Neuroimage.*, pp. 1053–8119, 2014.
- [63] S. Zhang, S. Zhang, T. Huang, W. Gao, and Q. Tian, "Learning Affective Features with a Hybrid Deep Model for Audio-Visual Emotion Recognition," *IEEE Trans. Circuits Syst. Video Technol.*, pp. 1051–8215, 2017.
- [64] Z. Xia, J. Lin, and X. Feng, "Trademark image retrieval via transformation-invariant deep hashing," *J. Vis. Commun. Image Represent.*, vol. 59, pp. 108–116, 2019.
- [65] K. Vishi, and V. Mavroedis, "An Evaluation of Score Level Fusion Approaches for Fingerprint and Finger-vein Biometrics," *Proc. Of the 10th Nor. Inf. Secur. Conf. (NISK 2017), Oslo, Norw.*, pp. 1–11, 2017.
- [66] C.C. Lip, and D.A. Ramli, "Comparative Study on Feature, Score and Decision Level Fusion Schemes for Robust Multibiometric Systems," *Front. Comput. Educ.*, vol. 133, pp. 941–948, 2012.
- [67] H. Kaya, F. Gürpınar, and A.A. Salah, "Video-based emotion recognition in the wild using deep transfer learning and score fusion," *Image Vis. Comput.*, vol. 65, pp. 66–75, 2017.
- [68] B.-N. Kang, Y. Kim, and D. Kim, "Deep Convolutional Neural Network using Triplets of Faces , Deep Ensemble, and Score-level Fusion for Face Recognition," *IEEE Conf. Comput. Vis. Pattern Recognit. Work., Honolulu, HI, USA*, pp. 109–116, 2017.
- [69] S.N.B. Bhushan, and A. Danti, "Classification of text documents based on score level fusion approach," *Pattern Recognit. Lett.*, pp. 0167–8655, 2017.
- [70] A. George, and A. Routray, "A Score-level Fusion Method for Eye Movement Biometrics," pp. 1–11, 2016.
- [71] Y. Jhansi, and E.S. Reddy, "A Methodology for Sketch based Image Retrieval based on Score level Fusion," *Int. J. Comput. Appl.*, vol. 109, no. 3, pp. 0975–8887, 2015.
- [72] M. Jovic, Y. Hatakeyama, F. Dong, and K. Hirota, "Image Retrieval Based on Similarity Score Fusion from Feature Similarity Ranking Lists," *Springer-Verlag Berlin Heidelb.*, pp. 461–470, 2006.
- [73] W. Xue, Q. Li, and Q. Xue, "Text Detection and Recognition for Images of Medical Laboratory Reports With a Deep Learning Approach," *IEEE Access*, vol. 8, pp. 407–416, 2020.
- [74] S.P. Rana, M. Dey, and P. Siarry, "Boosting content based image retrieval performance through integration of parametric & nonparametric approaches," *J. Vis. Commun. Image Represent.*, vol. 58, pp. 205–219, 2019.
- [75] S. Prabhakaran, "Cosine Similarity – Understanding the math and how it works (with python codes)," 22-Oct-2018 [Online] *Machine Learning Plus*. Available at: <https://www.machinelearningplus.com/nlp/cosine-similarity/> (accessed 29 Feb 2020).
- [76] P. Sitikhu, K. Pahi, P. Thapa, and S. Shakya, "A Comparison of Semantic Similarity Methods for Maximum Human Interpretability," pp. 1–4, 2019.
- [77] R. Ponakala, H.K. Adda, C.A. Kumar, K. Avula, and K.A. Sheela, "Character Recognition And Extraction of An Indian State License Plates Using K-NN," *Jawaharlal Nehru Technological University Hyderabad.*, 2016.
- [78] A. Soliman, K. Eissa, and S. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP," *Procedia Computer Science.*, vol. 117, pp. 256–265, 2017.
- [79] F.H. Khan, U. Qamar, and S. Bashir, "SentiMI : Introducing point-wise mutual information with SentiWordNet to improve sentiment polarity detection," *Appl. Soft Comput. J.*, vol. 39, pp. 140–153, 2016.
- [80] F. Schilling, "The Effect of Batch Normalization on Deep Convolutional Neural Networks," *KTH Royal Institute of Technology*, pp. 1–113, 2016.
- [81] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *31st Conf. Neural Inf. Process. Syst. (NIPS)*, 2017.
- [82] Q. Le, and T. Mikolov, "Distributed Representations of Sentences and Documents," vol. 32, pp. 1–5, 2014.
- [83] S. Chitroub, "Classifier combination and score level fusion: concepts and practical aspects," *Int. J. Image Data Fusion.*, vol. 1, no. 2, pp. 113–135, 2010.
- [84] M. Khayyat, and L. Elrefaei, "Towards Author Recognition of Ancient Arabic Manuscripts Using Deep Learning: A Transfer Learning Approach," Accepted Manuscript for publication in the *International Journal of Computing and Digital Systems, (IJCDs)*, vol. 9, no. 4, 2020.
- [85] S. Guo, Y. Liu, R. Chen, X. Sun, and X. Wang, "Improved SMOTE Algorithm to Deal with Imbalanced Activity Classes in Smart Homes," *Neural Process. Lett.*, 2018.
- [86] M. Khayyat, and L. Elrefaei, "A Deep Learning Based Prediction of Arabic Manuscripts Handwriting Style," Accepted Manuscript for publication in the *International Arab Journal of Information Technology, (IAJIT)*, vol. 17, no. 5, 2020.



**MANAL M. KHAYYAT** received the B.Sc. degree (Hons.) in Computer Science from King Abdulaziz University, Saudi Arabia, in 2007 and received M.Sc. degree of Applied Science in Quality Systems Engineering from Concordia University, Canada, in 2015. She is currently a PhD student in the Department of Computer Science at King Abdulaziz University. She worked at the IT department of Effat University, Saudi Arabia, from 2007 to 2010. Then, she worked as a lecturer at King Abdulaziz University, from 2012 to 2019 and she is currently working as a lecturer at Umm Al-Qura University, Saudi Arabia. Her research interests include



computer vision, image processing, natural language recognition, and deep learning.



**LAMIAA A. ELREFAEI** (M'12–SM'17)

received the B.Sc. degree (Hons.) in electrical engineering (electronics and telecommunications), and the M.Sc. and Ph.D. degrees in electrical engineering (electronics) from the Faculty of Engineering at Shoubra, Benha University, Egypt, in 1997, 2003, and 2008, respectively. She held a number of faculty positions at Benha University, as a Teaching Assistant, from 1998 to 2003, as an Assistant Lecturer, from 2003 to 2008, and has been a Lecturer, since 2008. She is currently an Associate Professor with the Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. Her research interests include computational intelligence, biometrics, multimedia security, wireless networks, and nano networks.