Multimedia Tools and Applications (2022) 81:15395–15417
https://doi.org/10.1007/s11042-022-12572-1

Deep reinforcement learning approach for manuscripts image classification and retrieval

Manal M. Khayyat [1] · Lamiaa A. Elrefaei [2]

Received: 9 March 2021 / Revised: 17 June 2021 / Accepted: 31 January 2022 /
Published online: 28 February 2022

Abstract
The automatic classification and retrieval of images is a challenging task, especially when dealing with low-quality and faded inks images, such as the historical manuscripts. Therefore, in this study we develop a reinforcement learning agent that is capable of interacting with an environment including historical Arabic manuscript images and retrieve the most similar images to a query image. First, the deep visual features of the images are extracted utilizing the pre-trained VGG19 convolutional neural network. Then, the associated deep textual features of the images are also extracted utilizing the attentional BiLSTM deep learning model. Both features are fused using the concatenation merge layer and hashed to reduce the dimensionality among the fused feature vectors for better image classification and retrieval. The proposed method tested on a manually collected dataset and recorded a promising high accuracy proved that the computer vision could be better performing than the humans' vision.

Keywords Reinforcement learning · Image retrieval · Deep features fusion · Locality-sensitive hashing

1 Introduction

Deep learning technique includes stacked convolutional neural layers that are able to learn and predict by their own after seeing a huge number of data. Thus, the deep learning learns from training then, expands the gained knowledge on new unseen data.

✉ Manal M. Khayyat
mmkhayat@uqu.edu.sa

Lamiaa A. Elrefaei
lamia.alrefaai@feng.bu.edu.eg

[1] Computer Science Department, Deanship of Preparatory Year of the Joint Medical Track, Umm Al-Qura University, Makkah, Saudi Arabia

[2] Electrical Engineering Department, Faculty of Engineering at Shoubra, Benha University, Cairo, Egypt

On the other hand, the reinforcement learning is a cutting-edge technique in the machine learning world. The primary deep reinforcement network consists of an agent and an environment. The interaction begins from the environment; it sends the state to the agent. Then, the agent takes action based on the assigned state. Afterward, the environment rewards the agent according to the chosen action, as illustrated in Fig. 1.

The reinforcement network communicates an agent with a continuous dynamic environment looking for the optimal behavior. Thus, the reinforcement learning learns from trial-and-error actions taken by the agent [10].

Figure 2 illustrates the difference between the deep learning approach and the reinforcement learning approach.

The reinforcement learning has the advantage of learning from its historic feedback of the performed actions while trying to fine-tune the learning hyperparameters. Therefore, the final made decision by a reinforcement model is taken after a sufficient number of iterations and through reinforcing the machine to learn from the successfully taken actions [16]. The reinforcement learning is entirely autonomous in its learning, and it requires fewer data to learn than the deep learning technique. The technique has been widely adopted in electronic games such as the Atari, chess, and in developing robots [29].

A classical reinforcement learning method is using handcrafted states, such as temperature, speed, etc. [24]. However, the deep learning technology could be merged with the reinforcement learning technique to reach a Deep Reinforcement Learning (DRL) model.

Reinforcement learning can be on-policy or off-policy. The main difference between these two types of learning is that the on-policy learning methods learn the value of the policy being taken by the agent. In contrast, the off-policy learning methods don't consider the agents' taken actions while learning the policy's optimal value. An example of on-policy learning is the R-Learning, while the Q-Learning is off-policy.

Contributions of the paper present in; "to the best of our knowledge" being the first study that develops an enhanced deep reinforcement learning model to retrieve the Arabic manuscript images. The model is based on the features-level fusion of both the visual and textual models. The fused features are then hashed to reduces the dimensionalities among them and to make the environment more stable through removing the randomness existed in the search process.

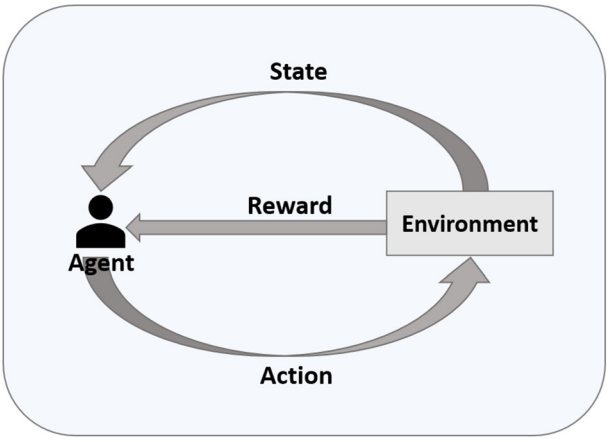

Fig. 1 The basic architecture of the reinforcement network

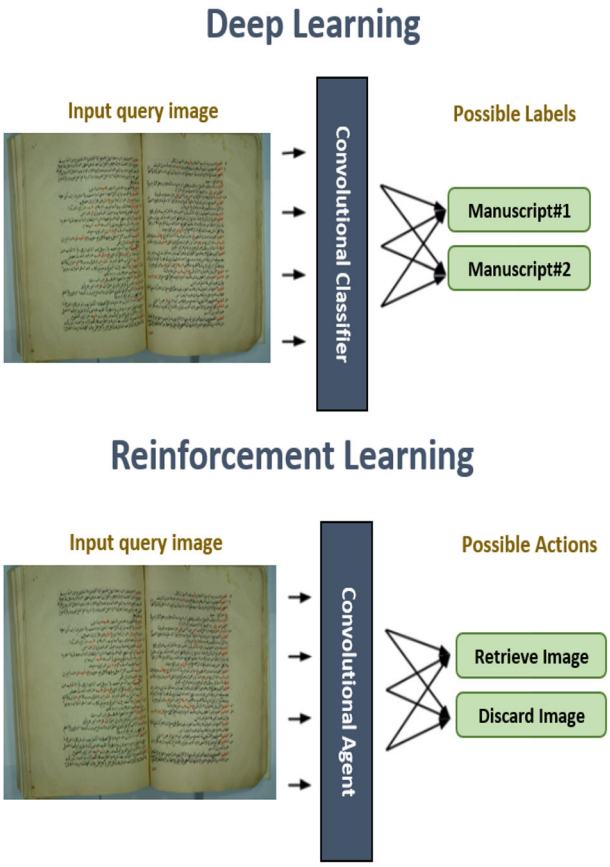

Fig. 2 Difference between deep learning and reinforcement learning

The rest of the paper is organized as follows: Section 2 discusses the literature review. Section 3 explains the proposed model for automatic image classification and retrieval. Section 4 discusses the experiments and test results. While section 5 concludes the paper and presents some ideas for the future work.

2 Literature review

Many efforts accomplished for retrieving the images of the Arabic manuscripts successfully. Some studies used the static handcrafted features to retrieve the images of the Arabic manuscripts, such as Yahia [31], who recommended using the latent semantic indexing for retrieving the images of the Arabic manuscripts. He started by preprocessing the textual-images through smoothing and converting them into a grey-scale version. Afterward, he segmented the images into lines and words and measured the similarities among them using the singular value decomposition method. The author recorded 78.8% recall. Similarly, El-Makhfi [8] recommended words spotting using the speeded-up robust features' technique. The author measured the similarity among the images using the Euclidean and the Mahalanobis distance methods. He measured the recognition accuracy and reached 95.27%.

Othman [19] utilized the bag of word fragments algorithm to recognize the handcrafted features and spot the Arabic words. He binarized the colored images and removed all the non-textual parts form them to be able to segment them successfully. Then, he measured the similarities using both the histogram intersection and the earth movers distance metrices. The author computed both the precision and the recall and reached 89.60% precision and 50% recall. Al-Maadeed et al. [3] proposed using the optical shape recognition method to spot the words within the ancient Arabic manuscripts images. The author developed a web-based search system and built the database using the SQL language. He reached that the manual handwritten images must be converted into digital form to be able to search and retrieve them.

Other studies extracted the automatic deep learning features from the Arabic manuscripts' images, such as [9, 12]. Elnagar et al. [9] recognized the deep learning features of two Arabic datasets using the attentional recurrent neural network. The researchers classified the Arabic text using the "Sigmoid" activation function and recorded 96.94% classification accuracy using the SANAD dataset and 88.68% using the NADiA dataset. Moreover, Khayyat and Elrefaei [12] fused the visual and the textual deep learning features presented in the ancient Arabic manuscripts' images. The authors extracted the visual features using the four different pre-trained convolutional neural networks reaching that VGG19 is the most accurate network for extracting the visual features. Then, they extracted the textual features using an enhanced attentional and bidirectional LSTM deep learning model. Both VGG19 and the attentional bidirectional LSTM deep learning model fused at three various levels, which are decision, features, and score level fusion. They concluded that the three fusion methods are generating close results. However, the score-level fusion method generated the highest image classification, and retrieval result as 0.9896% classification accuracy and 0.9819% mean average precision using the top-10 retrieval.

Concerning the image classification and retrieval in general, we found some efforts accomplished using the reinforcement learning technique. For instance, Chen et al. [7] proposed recognizing and classifying multi-labeled images using the recurrent attentional reinforcement learning. The problem they are trying to solve is considered a sequential decision task since the images are including more than one semantic label. They started their work by localizing specific regions in the input images to the model using the deep VGG16 CNN. Then, they developed an attentional LSTM deep learning model that takes the extracted features from the images' regions, as well as the preceding iterations' hidden states as inputs to recognize the semantics of the images and use them in the classification process.

The entire models developed in a deep reinforcement fashion that considers the actions as classifying the localized attentional region of images. The states are chosen according to the preceding iterations' details, and the features extracted from the images' regions using the deep VGG16 CNN. The rewards are the successful classification of the multi-labeled images. The authors used two datasets named: "PASCAL VOC" and "MSCOCO" to evaluate their proposed method. They computed the mean Average Precision (mAP) and recorded 92.0% using the PASCAL VOC dataset, and 71.1% F-score using the MSCOCO dataset.

Lin et al. [15] recommended using the Deep Q-learning Network for imbalanced data (DQNimb) to solve the image classification problem, which is formalized as a sequential decision-making problem. Because the used dataset is imbalanced, the authors employed the Imbalanced Classification Markov Decision Process (ICMDP) solution method. The basic concept of the method is to reward the classified images from the minor classes more than the images from the classes that include a large number of images, to overcome the imbalanced dataset problem. Regarding the used policy, it's an ϵ-greedy policy that utilizes a replay

memory to store the real-time experiences and use them to simplify the agent learning process. Hence, a new dataset is built from the agent's experience. The learning hyperparameters are (50,000) replay memory size, (120,000) steps to represent the interaction between the agent and the environment, (Adam) optimizer, (γ = 0.1) discount factor, and (0.00025) learning rate. After setting the learning hyperparameters, the authors used the "IMDB", "Cifar-10", "Mnist" and "Fashion-Minist" datasets to assess their proposed DQNimb. The authors recorded the highest G-mean score using the Mnist dataset as 0.991% with 1% imbalance ratio.

Zhao et al. [34] designed and implemented a deep reinforcement learning model for classifying vehicle images as an attempt to make the transportation system intelligent. They used the deep VGG CNN with a visual attention layer to focus on the important parts of the images while ignoring the trivial parts of them. Hence, the role of the agent is to find the attentional parts of the images. Afterward, the hash code of the query input image, along with all the other images in the dataset, are computed and then entered a hamming distance to measure the similarities between them. The generated similarities from the hamming distance are ranked to retrieve the top similar images to the user query image. The authors used the surveillance-nature dataset to assess their model and recorded 96.41% classification n accuracy.

Nie et al. [18] proposed using the Markov Decision Process (MDP) to predict the views of 3D images and then retrieve the images using a deep reinforcement learning-based model. The authors utilized the OpenGL tool to visualize twelve ordered pictures from each 3D image. Then, they used the extracted twelve pictures, as well as, the extracted visual features from the images using the deep FDNnet CNN as the environmental states of the deep reinforcement learning model. According to the received state from the environment, the developed model selects between three pre-designed actions, which all move the 3D image into a specific direction. The authors then computed the Euclidean distance between the query image and the rest of the images in the dataset to retrieve the most similar images. The authors evaluated their proposed method using the ModelNet40 dataset, which consists of 3D images. They reached the highest F-score using the views extraction technique as 31.12%.

Peng et al. [20] encouraged using Deep Reinforcement Learning combined with the Image Hashing method (DRLIH) to retrieve similar images. The images' hashing technique uses the hamming distance metric to match similar images with generated similar hash codes for the easier retrieval process. The authors developed a recurrent neural network and used them as an agent to hash the images into binary codes while keeping track of the previous errors to avoid them in future decisions. The agent follows the Monte-Carlo policy gradient to hash the images correctly. Three benchmark images datasets were used to evaluate the DRLIH model named: CIFAR10, NUS-WIDE, and MIRFlickr. The highest recorded mAP was using both CIFAR10 and NUS-WIDE datasets with 32bit as 84%, while the MIRFLICKR dataset recorded 81% mAP.

Wang et al. [26] recommended using the pre-trained ResNet50 CNN with the deep reinforcement learning technique to solve the videos' highlight recognition for better retrieval of videos according to users' preferences. The authors used a set of images and keywords to match the videos with the users' preferences. Hence, the videos are first segmented into a set of images. Then, the model is trained on the segmented images and keywords. Finally, the user enters a query input image to the model to retrieve the matching videos. The authors evaluated their model using four complete videos and recorded the mAP. They reached 57.24% mAP using the big bang theory, 56.93% mAP using the academy awards, 58.25% mAP using love actually, and 58.62% mAP using the BBC video.

Zhou and Agichtein [36] introduced a dynamic ranked search environment called (RLIRank). The authors proposed utilizing the deep reinforcement learning approach to retrieve the most relevant images to a user query. The retrieved documents get filtered according to the user's feedback. Three stacked LSTM layers, followed by a dense neural network, are used to define the environments states. Once the agent receives a query, then it will perform the search process. The action taken by the agent is to retrieve the ranked most relevant images to the query. The TREC 2016 and 2017 datasets were used to evaluate the proposed model and recorded 79.27% and 64.99% accuracy after the 10th iteration using the TREC 2016 and TREC 2017 datasets, respectively.

Yao et al. [32] proposed enhancing the personalized search using the reinforcement learning technique. They called their model (RLPer). It bases on the Markov Decision Process (MDP) to sequentially filter the retrieved documents according to the user's preferences. Moreover, attentional GRUs are used to rank the retrieved documents. The model is trained at the beginning according to the expert policy; then, the MDP calculated policy is more engaged to improve the training process. The relevance score between the query and the retrieved documents is computed using three main parts as following: 1) relevance with the query, 2) short-term user's interest, and 3) long-term user's interest. The authors evaluated their model using the public AOL search log dataset and recorded 59.81% mAP.

3 Proposed model

The image retrieval task requires a dynamic and interactive search system [36]. Therefore, we formalize the image retrieval problem as a sequential decision-making problem, where the agent is considered the search engine. The environment sends an image by image to the agent for retrieving the most similar images to each query image. To solve the retrieval problem, the Deep Reinforcement Q-learning Network (DRQN) is used. It involves an end-to-end ordered interaction between the environment and the agent.

Table 1 lists the main used components, along with their definitions.

From Table 1, we notice that the deep reinforcement learning approach is based on seven main components, which are explained in detail as follows:

1) **Environment**: includes the dataset, which consists of the images (I_t) along with their associated textual contexts (C_t), $D = \{(I_1, C_1), (I_2, C_2), \ldots., (I_n, C_n)\}$.

Table 1 The definitions of the DRQN components

Component	Definition
• Environment	Includes the dataset, which has the images along with their associated textual contents.
• Agent	Select and retrieve similar images in terms of features utilizing the K-NN algorithm.
• State (S_t)	The fused extracted features from the visual and the textual models.
• Action (a_t)	Array of the indices of the relevant retrieved images after computing the distance using the Cosine metric.
• Policy ( )	Boltzmann exploration policy.
• Reward (r_t)	The average computed distance of relevant retrieved images.
• Observation (o_t)	Array of the indices of the irrelevant retrieved images.

The environment is created using the "gym" reinforcement learning library. Once, a new environment is initiated, then it has access to the dataset images and their associated textual contents. After creating the environment, build an environment's constructor. At each time step (t), the constructor shuffles all the train images and select one image randomly. The visual features of the selected image are extracted using the pre-trained VGG19 convolutional neural network. Then, the textual contents of the selected image are recalled for extracting the textual features from them using the BiLSTM deep learning model. Both extracted visual and textual features are fused into one features-level fusion model using the concatenate merge layer from the "Keras" library. Table 2 illustrates the architecture of the features-level fusion model.

From Table 2, we notice that the fusion model accepts visual and textual inputs. Then, the model takes the extracted visual features from the VGG19 deep learning model 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 into 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 dropout and the batch normalization layers are added after the concatenation layer to improve the fusion model classification accuracy, followed by the final "Softmax" classification 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.

The features-level fusion model developed using the "Concatenate" merge layer from the "Keras" deep learning library only. Because the other different types of fusion models such as, the decision-level fusion model and the score-level fusion model, require the inputs to the merge layer to be of the same size. While, unlike the other fusion models, the features-level fusion model able of using features vectors 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. Equation (1) illustrates the mathematical representation of the used concatenate layer.

$$Concatenate = \quad VCL, TCL \tag{1}$$

Where VCL represents the visual classified labels and TCL represents the textual classified labels.

Table 2 Architecture of the features-level fusion model

Layer	Output Shape	Parameters Number	Connected to
Input_1	(224, 224, 3)	0	NA
Input_2	(500, 100)	0	NA
Model_1	(512)	22,124,096	Input_1[0][0]
Model_2	(128)	5,085,108	Input_2[0][0]
Concatenate_1	(640)	0	Model_1 [0, 10] Model_2 [0, 10]
Droupout_1	(640)	0	Concatenate_1[0][0]
Batch_Normalization_3	(640)	2560	Dropout_1[0][0]
Dense_1	(64)	41,024	Batch_Normalization_3[0][0]

The features-level fusion model complied using the "Adam" optimizer with the "categorical crossentropy" loss function, utilizing (0.01) learning rate and then, trained using (10) learning cycles. The generated evaluation results are summarized in Table 9 of [12] research study.

The fused features are then sent to the agent. The following algorithm illustrates the approach of sending images from the environment to the agent.

Algorithm 1 (Send Images to the Agent)

Let *n* be the total number of train images
Create an environment using "gym"
Build Environment's Constructor
Create a Search Session
For (*n*)
{

1. Shuffle images and select one image at time step *t* randomly.
2. Extract the visual features from the selected images using VGG19.
3. Take the corresponding textual content of the selected image.
4. Extract the textual features from the retrieved contents using BiLSTM deep learning model.
5. Fuse both extracted features using the concatenate layer.
6. Send the fused features to the search agent.

}

2) **Agent**: works as a search engine that selects and retrieves the most similar images to the sent image's features from the environment. Hence, the agent has access to the fused features being sent. Once the agent receives states from the environment, it takes actions as correct as possible utilizing the KNN algorithm to classify and measure the distances using the Cosine distance metric, trying to reach the optimal relevant list of images. Each search episode ends when the agent's goal is reached, which is finding the top-k similar images correctly.
3) **State (S_t)**: All the images in the manually collected dataset are shuffled and augmented using the "ImageDataGenerator" library in "Keras" as explained in [13]. At each discrete time step (t), a random image is chosen to extract its both visual and textual features. Regarding the visual features, they are extracted using the VGG19 deep learning model. The textual features associated with each image are also extracted using the BiLSTM deep learning model. Afterward, the features are fused into one feature vector to represent a newly updated state that is ready to be received by the search agent.

4) **Action (a_t)**: compute the distance between the entered query image and each retrieved image using the Cosine distance metric associated with the KNN algorithm. Considering that the goal is to maximize the cumulative rewards to reach the optimal action-value function (Q^*), which is computed as illustrated in Eq. (2) [24]:

$$Q^*(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_a Q(s_t, a_t) - Q(s_t, a_t) \right] \tag{2}$$

Where $Q(s,\ a)$ is the initial action-value function, α refers to the learning step size, r is the environment's reward based on the chosen action, γ refers to the discount factor $0 \leq \gamma \leq 1$, s is the newly updated state sent from the environment, and a is the new updated action taken by the agent.

The following equation, clarifies the computation of the initial action-value function [34]:

$$Q(s_t, a_t) = r_t + \gamma \max_a Q(s_{t+1}, \boldsymbol{a}) \tag{3}$$

Where $\boldsymbol{a}$ is the set of all actions that the agent can choose from. The agent's actions are connected to the dynamic environment according to its computed distance to the query image. If the computed distance is less than the threshold point then, the image as is set as relevant image and its index is added to the action array because it belongs to the same manuscript as the query image. Otherwise, the image is set to irrelevant image and its index is added to an observation array. Note that, all taken decisions are saved in an experience memory dataset to help the agent in taking its future actions. Hence, the agent learns from trial and error.

5) **Policy (π)**: the Boltzmann exploration policy[1] is used to direct the agent with a set of rules to follow. The policy works with a discrete action space. Hence, it receives the predicted actions from the agent as probabilities and converts them into distributions representing the true action that will be applied to the environment.

The implementation of the Boltzmann exploration policy on an action at a time is illustrated in Eq. (4) [24]:

$$\pi_t(a) = \frac{e^{P_t(a)}}{\sum_{i=1}^{n} e^{P_t(i)}} \tag{4}$$

Where P_t is the computed probability by the "Softmax" clasification function, and n is the total number of images in the dataset.

6) **Reward (r_t)**: according to the action taken by the agent, which is the list of all relevant images to the user query image, the average of the relevant images distances is returned as a reward to the agent's action.

The reward is the average of a simple numerical value that can be computed to help the agent learns better. In this study, we used the average of the computed distances between the query image and the relevant retrieved images because the KNN algorithm is depending on the

[1] https://nervanasystems.github.io/coach/_modules/rl_coach/exploration_policies/boltzmann.html#Boltzmann

concept of measuring the distances between an image and all the nearest neighboring images in a dataset. Considering that as much as the distance is small, as much as the retrieved image is similar to the query image and vice versa. Hence, we can make sure that the model is working successfully and retrieving similar images to the query image by computing the average of the distances of the retrieved images and making sure that it is less than the threshold point, which is set to (0.2). The threshold is a simple constant value used to ensure that the retrieved images are relevant to the query image.

The value of the threshold point equals to (0.2) because it is the optimal threshold value reached after calculating the average of the total predicted positive images from the test subset, which includes only 15% from the entire dataset images. The total number of images in the entire dataset equals (8638). Thus, 15% of the total number of images equals (1296). Considering that the total number of positively predicted images equals (310), then the optimal threshold value is (310/1296 = 0.2).

The calculations are explained in many articles such as in [2, 6, 17].

The reward is a session-based computed value that gets updated with every new episode that includes a new query image (state). Algorithm 2 explains the computation of the reward.

Algorithm 2 (Reward Computation)

Let k be the number of the total retrieved similar images' distances.
r_t is the reward from the environment to the agent at the time step t.
$\hbar = 0.2$, is the distance threshold.

For (k)
{
  If (Cosine distance = (1 – Cosine similarity) < = $\hbar$)
  {set the image as relevant image (RI)}

  else
  {add the image's index to the observation array}
}

$$r_t = \frac{\sum_{i=0}^{k} distances\ (RI)}{k} \tag{5}$$

7) **Observation (o_t)**: the indices of all images that are irrelevant (not from the same manuscript as the query image) are saved in an observation array (o_t) in the experience replay memory to assist the agent in the learning process.

Figure 3 illustrates the architecture of the proposed model.

From Fig. 3, we notice that the developed model to retrieve the images consists of six main steps as following:

1. Start from the environment, which has the dataset images and its related contexts. At each discrete time step, the environment enters a random shuffled image into the pre-trained deep VGG19 CNN to extract the visual features from it. Moreover, the corresponding

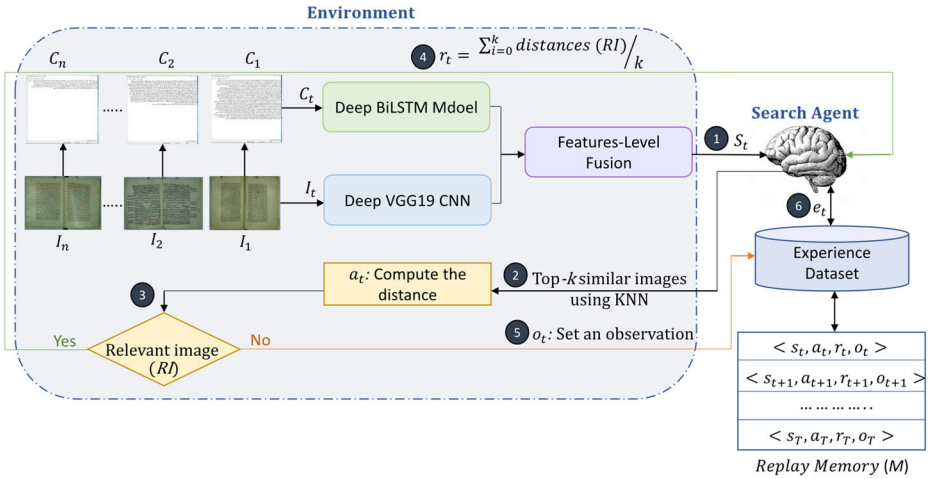

Fig. 3 Framework of the proposed DRL model for image retrieval

textual features of the image are extracted using an attentional BiLSTM deep learning model. The features from both the visual and textual models are fused using the concatenation merge layer to generate the state (S_t).

2. Once the agent receives a state from the environment, it follows the policy rules and checks the previous saved experience as an attempt to take the correct action. To create the agent's experience dataset, we store all the previously taken actions as an experience in a replay memory. Note that, the first action is taken with no previous saved experience. i.e., The $e_t(a) = 0, \forall\ a$. However, as much as the agent is trained, as much as it builds a better experience depending on the previously taken actions. i.e. $e_{t+1}(a) = e_t(a) + \alpha\ (r_t)\ \pi_t(a)$.
3. After the agent retrieves the top-k similar images, the distance is computed between the query image and the rest of the retrieved images to assign the proper reward to the agent's taken action.
4. If the retrieved images by the agent are from the same manuscript as the user's query image, then the agent gets a reward equals to the average of distances of the relevant retrieved images.
5. In contrast, if the retrieved image is from a different manuscript than the user's query image, then the environment saves the retrieved image's index in an observation array.
6. The detailed information of the reached decisions are saved in the replay memory (M) to assist the agent in learning more efficiently through time.

The architecture in Fig. 3 is chosen according to the main components used in the common basic architecture for any deep reinforcement learning model. As illustrated in Fig. 1, the basic architecture for all deep reinforcement learning models is that there is a communication between an agent and the environment, where the environment sends a state to the agent to be able to respond with an action according to the sent state. Afterward, the environment rewards the agent on the chosen action to assist the agent to learn better whether the chosen action is wrong or right. Hence, the agent tries to utilize the reward to reach the optimal action.

The used collected dataset consists of textual-based images. Therefore, we had to employ two different deep learning models. The first model is VGG19 deep learning model to extract the visual features from the images. The VGG19 model is selected because we tested many

deep learning models in previous studies such as the study in [13] and the VGG19 model generated high promising results in classifying the historical manuscripts images according to the extracted visual features. On the other hand, the second model for extracting the textual features is BiLSTM deep learning model because it showed in [12] high results in classifying the images according to their extracted textual features.

After extracting both visual and textual features from the images successfully, we had to concatenate them using one fusion model. Thus, we used the features-level fusion model, which represent the final state to the search agent. The agent then applies the KNN algorithm to be able to take the correct action, which is retrieving the most similar images to each query image. The environment rewards the agent with the average of the retrieved relevant images' distances. This reward assists the agent in learning better and take more correct actions in the future because the entire information are saved in an experience memory dataset.

However, after experimenting the initial developed model, we reached that its generated results were not very high. Thus, we enhanced it by hashing the generated fused features, as illustrated in Fig. 4, and explained in the next subsection 3.1.

3.1 Model enhancement

Hashing the features reduces the dimensionalities among them, which assist in improving the clustering of the images belonging to the same manuscript. The hash function is one of the most popular solutions for approximating the nearest neighbor search [28]. Because it makes the environment more stable by removing the randomness in the search process. The hash function was suggested by many researchers to improve the image retrieval accuracy [5, 11, 21, 23, 25, 27, 30, 33, 35]. In addition, the authors of paper [34] and paper [20] recommended combining the deep reinforcement learning technique with the hashing technique to increase the image retrieval accuracy. Therefore, the initially proposed method for the image retrieval increased one step.

From Fig. 4, we notice that the retrieval steps became seven instead of six. The added step is to take the fused visual and textual features of the images and hash them. Then, send the generated reduced hashed features to the search agent. This step removes the randomness existed in the environment and assisted the agent in clustering and identifying similar images more accurately. The proposed hashing function is the Locality Sensitive Hashing (LSH) Forest method. The author in [4]

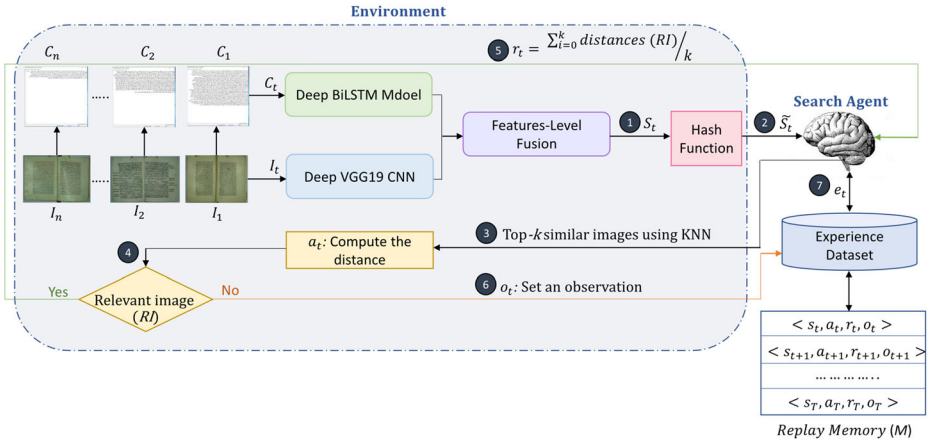

Fig. 4 Enhanced DRL model including the hash function for image retrieval

recommended using LSH Forest to hash textual data as it improves the retrieval accuracy. The LSH Forest is a data-independent, unsupervised learning method that clusters the images in groups according to their similarities. It receives the fused features as input and converts them into binary hash codes after reducing the dimensionalities among them. Thus, the images from the same manuscript will have similar hash codes. The mathematical representation of the LSH Forest method utilizing the min hash codes is presented in Eq. (6) [28]:

$$\widetilde{S}_t = argmin_{g=1}^{2^G} h_g^T . S_t \quad (6)$$

Where $\widetilde{S}_t$ is the new binary hashed features generated from the original fused features S_t. 2^G is the size of the randomly generated projection hyperplane h = $\{h_1, h_2, h_3, \ldots, h_{2^G}\}$.

4 Experiments and tests results

The used hardware is "ABS Battelbox" machine with Ubuntu 16.04 OS and Nvidia Gefore RTX 2080 GPU.

The dataset used for the experiment is from [13]. It includes 64 historical Arabic manuscripts' having a total of 8638 images, as illustrated in Table 3. The images are having both visual and textual contents, such as drawings, tables, figures, and texts.

To extract the textual contents from the images, the images uploaded into the "Google" cloud platform then, the Google optical character recognition tool is used to extract the Arabic handwritten text from the images. Hence, each image in the dataset is associated with a ".txt" file, including its textual contents. The initial extracted text tokenized and cleaned using the "natural language tool kit",[2] which removed the stray characters, non-Arabic letters, digits, and stop words. After cleaning the text, it converted into a feature vector using the "AraVec"[3] word-embedding tool and saved it in a database.

The entire dataset was divided into 70% training subset, 15% for the validation subset, and 15% for the testing subset according to the study in [14].

In the reinforcement learning solutions, we need to build or re-use an environment to hold our dataset. Thus, we minimized the coding part for developing the environment in our algorithm through re-using the available "gym[4]"library.

Gym is a ready and easy-to-use library to develop the reinforcement learning environment and link it with the domain model as explained in Fig. 5 from [22].

Utilizing the "gym" library, we customized our environment and trained the "DQNAgent" on the "Boltzmann" exploration policy successfully. This is all done under "Keras-rl" toolkit. Note that, "Keras-rl" toolkit is used to develop the policy value to action. From "rl.agents.dqn", we imported the "DQNAgent". Moreover, from the "rl.policy", we imported the "BoltzmannQPloicy", and from the "rl.memory", we imported the "SequantialMemory".

The "K-NN", as well as, the "LSH Forest" similarity search algorithms combined with the "Cosine" distance metric with a (reward value function = 10) are employed to compute the distances among the top-10 similar images to the user's query image and retrieve the most relevant images.

Regarding the learning hyperparameters, we set the min_hash_match to (4). Then, the deep reinforcement learning model was trained for (190 episodes), including (100 interaction steps)

[2] http://www.nltk.org/
[3] https://github.com/bakrianoo/aravec/blob/master/README.md
[4] https://gym.openai.com/

Table 3 Historical Arabic manuscripts dataset

Manuscript ID	Arabic Title	Author English Name	Time Period in Hijri	Calligraphy	No. of Images
1	تيسير الوصول إلى جامع الأصول	Abu Theyaa Abdulrahman Bin Ali Bin Mohammed	1004	Al-Nask	191
2	شرح الجامع الصغير	Al-Manawe	0	Al-Hur	42
3	منتخب كنز العمال	Alshaikah Hosam Aldin	0	Al-Hur	293
4	قطعة من شرح معاني الآثار	Abu Jafar Altahawe Almasri	0	Al-Nask	80
5	الأعمال الموجبة	Mohammed Alshaibi	1135	Al-Thulth	12
6	الهداية في علم الرواية	Shams Aldin Mohammed Bin Aljazri	1305	Al-Farsi	16
7	مسند الإمام أحمد بن حنبل رواية ابنه عبد الله	Alimam Ahmed Bin Hanbal	0	Al-Hur	309
8	مرقاة المفاتيح على مشكاة المفاتيح	Sidi Ali Alqari	1180	Al-Hur	313
9	الجامع الصغير	Abdulrahman Bin Abibakr Alsayoti	1233	Al-Hur	277
10	شرح صحيح مسلم بن الحجاج	Abu Zakariya Mohe Aldin Alnawawi	1075	Al-Nask	162
11	مشكاة المصابيح	Wali Aldin Altbrizi	1033	Al-Nask	260
12	مشكاة المصابيح	Wali Aldin Altbrizi	1183	Al-Nask	264
13	رياض الصالحين	Abu Zakariya Mohe Aldin Alnawawi	0	Al-Thulth	114
14	ذكر أسباب إصلاح البيوت	Aqeel Bin Omar	0	Al-Reqaa	16
15	قطعة من صحيح البخاري	Mohammed Bin Ismail Albukhari	1232	Al-Hur	114
16	شرح الأربعين، المسمى الفتح المبين	Ibn Hajar Alhythami	1335	Al-Hur	101
17	منتخب كنز العمال	Alshaikah Hosam Aldin	0	Al-Hur	292
18	الزواجر في الكبائر	Ibn Hajar Alhythami	0	Al-Nask	16
19	ثبت الأمير	Mohammed Bin Mohammed Alamer	1307	Al-Reqaa	30
20	مسانيد	Shihab Aldin Ahmed Ibn Mohammed Alhythami	1246	Al-Nask	139
21	العقد الفريد لبيان الراجح في جواز التقليد	Hasan Alshernulaly	1384	Al-Nask	27
22	الرحيق المختوم شرح قلائد المنظوم	Mohammed Afandi Abbdin	1305	Al-Thulth	44
23	الأجوبة المكية على الأسئلة الحفظية	Mohammed Maki Bin Azoz Altonisy	0	Al-Diwani	10
24	خزانة الروايات	Hekma Alhindi	0	Al-Nask	49
25	نظم الفوايد شرح المقاصد	Hasan Alshernulaly	1096	Al-Thulth	216
26	ملتقى الأبحر	Ibrahim Bin Mohammed Alhalabi	1064	Al-Diwani	158
27	المربع في حكم العقد على المذاهب الأربع	Abdulmoati Alsimlawy	1306	Al-Hur	6
28	تحفة التحرير	Hasan Alshernulaly	1064	Al-Nask	7
29	مقدمة عن الصلاة و شروطها	Hathar Bin Ahmed	1284	Al-Thulth	25
30	كنز الدقائق	Hafez Aldin Alnsfy	0	Al-Nask	104
31	عمدة الحكام ومرجع القضاة في الأحكام	Moheb Aldin Alhamawy	1243	Al-Hur	65
32	إجادة الجدة بمنع القصر في طريق جدة	Taj Aldin Bin Ahmed Aldahan	1310	Al-Farsi	11
33	القول البليغ في حكم التبليغ	Ahmed Bin Mohammed Alhamawy	1066	Al-Thulth	9
34	رسالة في القنوت في النوازل		1171	Al-Diwani	8

Table 3 (continued)

Manuscript ID	Arabic Title	Author English Name	Time Period in Hijri	Calligraphy	No. of Images
		Taj Aldin Bin Ahmed Aldahan			
35	أحكام الناطفي	Ahmed Bin Mohammed Alnatefy	1037	Al-Thulth	34
36	الفوائد الزينية في مذهب الحنفية	Zain Bin Nejam	1239	Al-Thulth	50
37	العناية على شرح الهداية	Akmal Aldin	1334	Al-Reqaa	274
38	العناية على شرح الهداية	Akmal Aldin	1334	Al-Reqaa	214
39	الدرة المنيفة على مذهب أبي حنيفة	Omar Bin Omar Alzahri Aldafri Alhanafy	1197	Al-Nask	40
40	درر الحكام شرح غرر الأحكام	Mala Khasro	0	Al-Hur	82
41	شرح التسهيل	Badr Aldin	0	Al-Farsi	71
42	الفوائد الشافية في إعراب الكافية	Hussain Bin Ibrahim	1209	Al-Reqaa	254
43	شرح الآجرومية	Ali Alnubity	1233	Al-Hur	179
44	تعليق الدرة الشنوانية على شرح الآجرومية	Alshenwany	1019	Al-Thulth	129
45	تعليق الفواصل على إعراب العوامل	Hassan Bin Ahmed Zaini Zadah	1165	Al-Thulth	82
46	شرح شذور الذهب	Ibn Hesham Alnahwi	1233	Al-Nask	140
47	لطائف الإعراب في شرح قواعد الاعراب	Haj Baba Ibn Othman Althrsiwi	1086	Al-Reqaa	119
48	حاشية على متن السمرقندية	Ahmed Bin Zaini Dahlan	1283	Al-Reqaa	9
49	مقامات الحريري	Qasem Alhariry	1064	Al-Thulth	132
50	حلية الكميت	Mohammed Alnawajy Almasri	0	Al-Nask	138
51	ريحانة الألبا وزهرة الحياة الدنيا	Ahmed Bin Mohammed Alkhafagy	1330	Al-Farsi	271
52	شرح الرسالة العضدية	Yousef Alhanafi Alshafei	1168	Al-Reqaa	6
53	تفسير الخطيب الشربيني	Mohammed Alkhatib Alsherbini	0	Al-Reqaa	280
54	حاشية على شرح الكافي	Mohi Aldin Altaljy	1135	Al-Reqaa	96
55	الاشاعة لاشراط الساعة	Mohammed Bin Abdulrasol	1368	Al-Nask	98
56	شرح الصدور في شرح حال الموتى في القبور	Jalal Aldin Alsayoti	0	Al-Thulth	103
57	الأشباه والنظائر الفقهية	Abdulwahab Alshearani	1160	Al-Diwani	297
58	تبيين الحقائق شرح كنز الدقائق	Alzailai	0	Al-Reqaa	283
59	تبيين الحقائق شرح كنز الدقائق	Alzailai	0	Al-Reqaa	381
60	تبيين الحقائق شرح كنز الدقائق	Alzailai	0	Al-Reqaa	141
61	تبيين الحقائق شرح كنز الدقائق	Alzailai	0	Al-Reqaa	175
62	تبيين الحقائق شرح كنز الدقائق	Alzailai	1132	Al-Reqaa	373
63	فتح القدير	Othman Ibn Mohammed	1233	Al-Reqaa	170
64	أسباب الإختيار	Abduallah Bin Alnasqi	1053	Al-Nask	237

between the agent and the environment per episode. The environment seed used to initialize the complete environment per episode was set to (123), while the agent warmup steps set to (30). The model's render mode was initialized to "human" to keep the learning more stable. Finally, we compiled the deep reinforcement learning model using (Adam) optimizer with a (0.001) learning rate.

The complete code for extracting the visual and textual features from the Arabic manuscripts' images and then concatenate them into one fusion model is available on

github.[5] In addition, the code to develop the enhanced deep reinforcement learning model that utilizes the fused visual and textual features for the image retrieval is available on github.[6]

Figure 6 illustrates an example of the testing phase by the developed model.

From Fig. 6, we notice that the testing phase includes entering the user's query image to the model to retrieve the most similar images to it. Therefore, the text associated with the entered query image is recalled to extract both the visual and the textual features of the image. Then, fuse them using one features-level fusion model. The fused features are then hashed using the LSH Forest method to reduce the dimensionalities among the features and remove the environment's randomness. The hashed fused features are passed into the search agent to select and retrieve the most similar images to the user query image using the K-NN algorithm. During the agent's search process, it checks its experience replay memory, which was built during the training phase as an attempt to take the correct action as possible.

The Cosine distance is measured between the query image and each retrieved image to decide their relevancy. Considering that the distance threshold is set to (0.2), then the first four retrieved images, which are highlighted using the green color in Fig. 6, are relevant and belonging to the same manuscript as the query image. In contrast, the last retrieved image that is highlighted using the red color is irrelevant, and it is from different manuscripts than the true manuscript of the query image. Note that, the distance of the first retrieved image is (0.000), meaning that the model retrieved the exact same user's query image.

To evaluate the performance of the developed deep reinforcement learning models, we computed the generated accuracy by each model. The equation for calculating the accuracy evaluation metric presented in (7) [1]:

$$Accuracy = \frac{S_{cw}}{T_w} \tag{7}$$

Where S_{cw} represents the number of successfully retrieved images and T_w represents the total number of images.

For the image retrieval task, we computed the mean Average Precision (mAP). Hence, first the Average Precision (AP) for each query until position N is computed as in Eq. (8):

$$\mathrm{AP} = \frac{1}{GT} \sum_{i=1}^{N} \frac{TP}{i} \tag{8}$$

Where GT are the ground truth labels, and TP are the true positive labels. Afterward, the mean of the computed average precision is calculated as illustrated in Eq. (9):

$$\mathrm{mAP} = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{9}$$

[5] https://github.com/ManalKhayyat/Arabic_Manuscripts_Image_Retrieval
[6] https://github.com/Manal-Khayyat/DRL

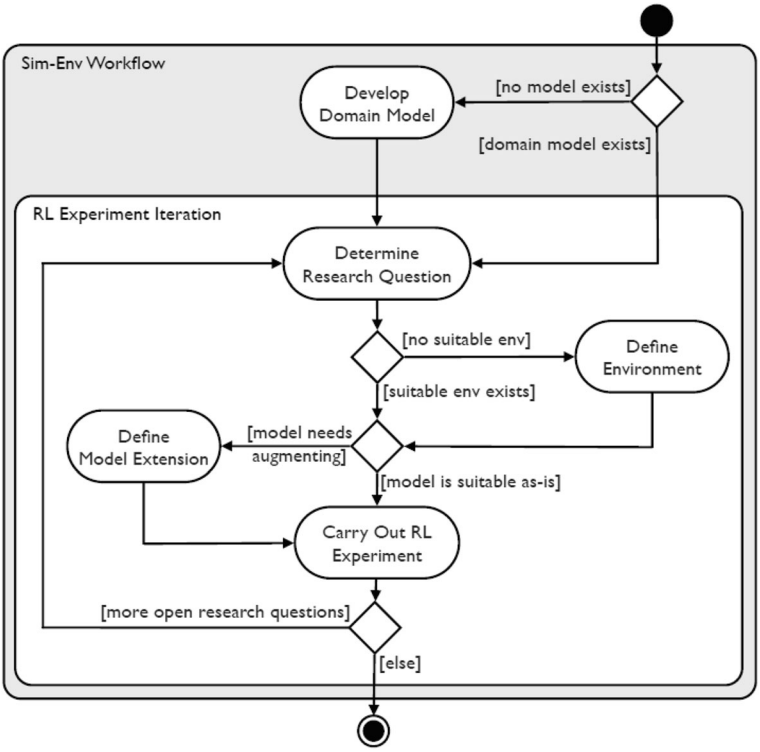

Fig. 5 Reinforcement learning environment flowchart [22]

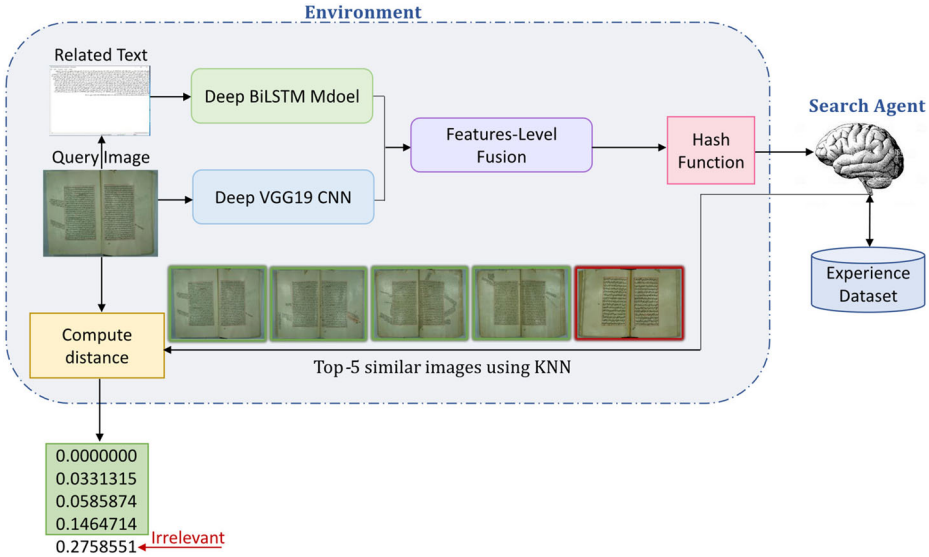

Fig. 6 Framework of the proposed DRL model for image retrieval

The results of the classification accuracy and the mAP of the successfully retrieved top-10 similar images utilizing both the initial DRQN and the enhanced DRQN model, including the hashing function, are summarized in Table 4. The same results are visualized in Fig. 7.

From Table 4 and Fig. 7, we notice that the recorded evaluation parameters are higher when we employed the hashing function to reduce the dimensionality among the fused features vectors.

The use of the K-NN algorithm alone to retrieve the similar images was including some randomness in searching the most similar images to the user query image, while combining the K-NN algorithm with the LSH Forest method eliminated the existed randomness in the environment and assisted the agent in its learning process, which resulted in more accurate images retrieval. The average retrieval time in seconds using the GPU for each retrieved top-10 similar images calculated from the entire dataset and utilizing the Cosine distance metric within the KNN algorithm equals 5.60275 s using the DRQN algorithm alone and 5.92057 s using the DRQN algorithm with the hashing function.

Figure 8 illustrates three 100% successful examples of the output results from entering an input query image into the proposed deep reinforcement learning model that includes the hashing function. From Fig. 8, we notice that the model was able to retrieve the top-5 images successfully in around five seconds, which proves that we reached a novel solution for image classification and retrieval.

Table 5 compares the proposed method with state-of-the-art methods. All the studies in Table 5 are categorized into three main categories according to the utilized approach to perform the image classification and/or retrieval as following: Word spotting, deep learning, and reinforcement learning.

From Table 5, we notice that four studies [3, 8, 19, 31] used the word spotting technique to classify and retrieve the images of the Arabic manuscript. On the other hand, only two studies [9, 12] employed automatic deep learning features to classify the Arabic manuscripts' images. However, looking at the studies that utilized the reinforcement learning approach, we were the only study that utilized the reinforcement learning to classify and retrieve the Arabic manuscripts' images, while all the other studies implemented the reinforcement learning on the English language images.

Moreover, we notice that our proposed model, which is combining the hashing function with the fused deep learning features, recorded high results comparing it with the previously recorded results by the other papers. Even the two papers that recommended combining the deep reinforcement technique with the hashing technique [20, 34] recorded lower results than our recorded results. That is because we fused the visual and textual features into one features-level fusion model that is better categorizing the manuscripts' image. Then, we hashed the fused features to remove the randomness from the environment and facilitate grouping the images from the same manuscript efficiently, which assists the agent in its learning process. Combining the reinforcement learning agent with the fused hashed features in the proposed method, enabled the model to visualize and distinguish between the Arabic manuscripts' images successfully.

Table 4 Evaluation of the DRL models according to the manuscript

Evaluation Parameter	DRQN	DRQN + Hashing
Classification Accuracy	0.9218495	0.9686520
Top-10 mAP	0.8963009	0.9431034

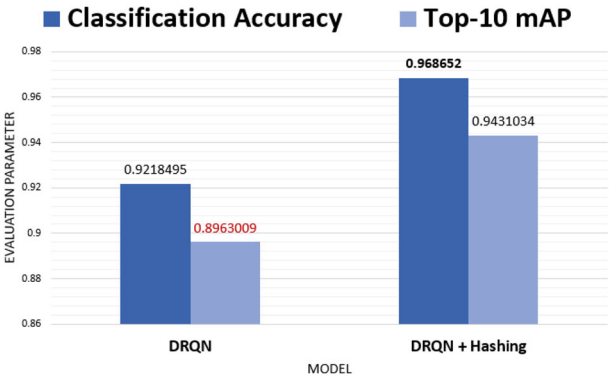

Fig. 7 Evaluation of the DRL model according to the manuscript

5 Conclusion and future work

This study introduces a novel approach for image classification and retrieval using a deep reinforcement learning model that communicates a search agent with the environment to retrieve the most similar images to a user-query image. The initially developed model was utilizing the K-nearest neighboring algorithm to do the classification and retrieval of the received fused visual and textual features from the environment. Adding the hash function enhanced the model's performance by reducing the dimensionalities among the received fused features, as well as, it removed the randomness existed in the environment, which assisted the agent in its learning process and increased the accuracy recording 96.87% classification accuracy and 94.31% mean average precision on the top-10 image retrieval.

Even though we did many previous similar studies for image retrieval from ancient Arabic manuscripts, such as [12–14]. The research in [13] was focusing on similar image retrieval according to the same author using four different deep learning models. On the other hand, the

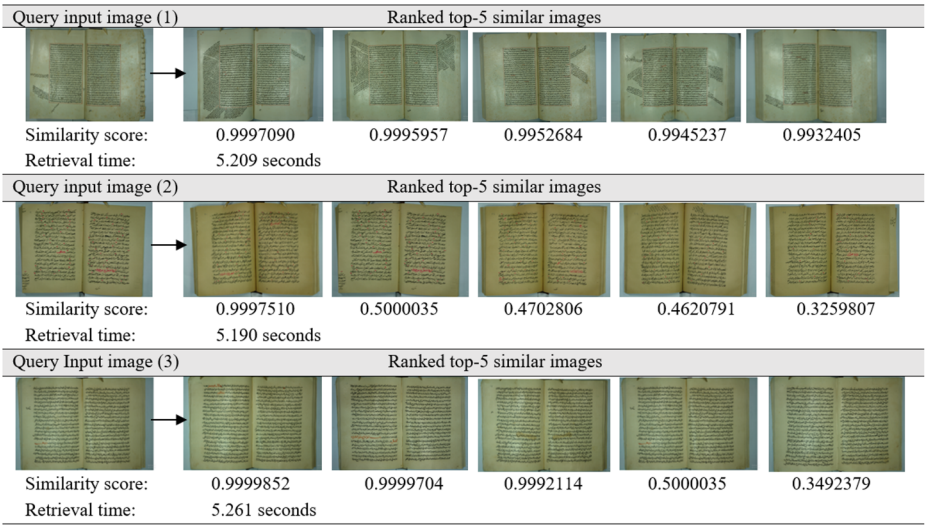

Fig. 8 Example of successfully retrieved similar images from the enhanced DRQN model including the hashing function

Table 5 Comparison with state-of-the-art methods for image classification and/or retrieval

Approach	Reference# (Year)	Problem Domain	Language	Classification/ Retrieval Technique	Dataset	Similarity Measure	Results
Word spotting	[31] (2011)	Retrieve manuscripts' images	Arabic	Laten semantic indexing	Two scanned Arabic manuscripts	Singular value decomposition	78.8% recall.
	[8] (2019)	Retrieve manuscripts' images	Arabic	Speeded up robust features	HADARA80P	Euclidean and Mahalanobis distance	95.27% accuracy.
	[19] (2015)	Retrieve manuscripts' images	Arabic	Bag of word fragments	120 scanned manuscripts' images	Histogram intersection and the earth movers distance	89.60% precision and 50% recall.
	[3] (2017)	Retrieve manuscripts' images	Arabic	Optical shape recognition	The "Ibn Sina" historical Arabic manuscript.	Web-based search SQL database.	The handwritten manuscripts images should be converted into digital form to be able to automatically search and retrieve them.
Deep Learning	[9] (2020)	Classify the Arabic text	Arabic	Attentional recurrent neural network	SANAD and NADiA databases	Sigmoid activation function	96.94% accuracy using SANAD, and 88.68% using NADiA database.
	[12] (2020)	Retrieve similar images according to manuscript	Arabic	Score level fusion of VGG19 and BiLSTM	8638 manually collected manuscripts' images	Cosine distance	98.96% classification accuracy and 98.19% mean average precision on the top-10 image retrieval.
	[13] (2020)	Retrieve images according to the author	Arabic	MobileNet_V1 ResNet_V2_50 DenseNet_201 VGG_19	8638 manually collected manuscripts' images	Sigmoid + Softmax activation functions	95.59% accuracy 96.23% accuracy 95.83% accuracy 95.91% accuracy
	[14] (2020)	Retrieve images according to handwriting style	Arabic	Transfer learning from MobileNet_-V1_100_244	2653 manually collected manuscripts' images	Softmax activation function	95.83% accuracy
Reinforcement Learning	[7] (2017)		English		PASCAL VOC and MSCOCO	hybrid loss function	

Table 5 (continued)

Approach	Reference# (Year)	Problem Domain	Language	Classification/ Retrieval Technique	Dataset	Similarity Measure	Results
		Classify general images using their textual labels		Recurrent attentional deep reinforcement learning			92.0% mAP using the PASCAL VOC dataset, and 71.1% F-score using the MSCOCO dataset.
	[15] (2020)	Classification of general images	English	DQNimb model	IMDB, Cifar-10, Mnist and Fashion-Minist	Softmax activation function	99.1% highest recorded G-mean score with 1% imbalance ratio using the Mnist dataset.
	[34] (2017)	Classification of vehicle images	English	Visual attentional reinforcement learning	Surveillance-nature dataset	Hamming distance	96.41% classification accuracy.
	[18] (2019)	Retrieve 3D images	English	Deep reinforcement learning	ModelNet40 dataset	Euclidean distance	31.12% F-score using the views extraction technique.
	[20] (2018)	Classify general images using their textual labels	English	RNN deep reinforcement learning model (DRLIH)	CIFAR10, NUS-WIDE, and MIRFlickr	Euclidean distance	84% mAP using both CIFAR10 and NUS-WIDE. 81% mAP using MIRFLICKR dataset.
	[26] (2020)	Retrieve video images	English	Deep reinforcement learning	The big bang theory, academy awards, love actually, and BBC	Earth Mover's Distance	58.62% highest recorded mAP using the BBC video.
	[36] (2020)	Retrieve ranked search images	English	RLIRank ranking deep reinforcement search	TREC 2016 and 2017 datasets	Softmax activation function	79.27% and 64.99% accuracy after the 10th iteration using the TREC 2016 and TREC 2017 datasets, respectively.
	[32] (2020)	Retrieve ranked personalized images	English	RLPer personalized deep reinforcement search	public AOL search log dataset	Softmax activation function	59.81% mAP
	Proposed method	Classify and retrieve the top-k similar images according to the manuscript label	Arabic	DRQN	Manually collected ancient Arabic manuscripts	Cosine distance	92.18% classification accuracy and 89.63% for the top-10 image retrieval
				DRQN with hash function			96.87% classification accuracy and 94.31% for the top-10 image retrieval

research in [14] was also employing deep learning to retrieve similar images but, according to the same handwriting style. This study and the research in [12] are having the same domain because they are both focusing on similar image retrieval according to the manuscript's label. However, the research in [12] was using the deep learning fusion models technique. While this study is novel because it is presenting a new enhanced technique for image retrieval utilizing a deep reinforcement learning model that learns from trial and error. Table 5 highlights the difference between this study and all previous studies in the field.

The developed deep reinforcement learning model can extract both visual and textual features from historical low-quality Arabic manuscripts images and retrieve the most similar images to a query image instantly and successfully. Thus, we believe that the model can also work successfully on modern high-quality images.

Future work includes an analysis of time and space efficiency for the proposed method. We also aim to extend our approach and computing skills to 3D multimedia data.

Funding The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4400271DSR02)

Declarations

Competing interests The authors declare that there is no conflict of interest.

References

1. Al Aghbari Z, Brook S (2009) Word stretching for effective segmentation and classification of historical arabic handwritten documents. Proc. 2009 3rd Int. Conf. Res. Challenges Inf. Sci. RCIS 2009:217–224
2. Alam M (2020) "k-Nearest Neighbors (kNN) for anomaly detection". [Online] *towards data science*. Available at: https://towardsdatascience.com/k-nearest-neighbors-knn-for-anomaly-detection-fdf8ee160d13 (accessed 03 Nov 2020)
3. Al-Maadeed S, Issawi F, and Bouridan A (2017). "Word Retrieval System for Ancient Arabic Manuscripts, " in *Proc. 9th IEEE-GCC Conf. Exhib. (GCCCE)*, pp. 1–5,
4. Bawa M, Condie T, Ganesan P (2005) LSH Forest : Self-Tuning Indexes for Similarity Search. Int World Wide Web Conf Committ (IW3C2):1–10
5. Bozas KK and Izquierdo E (2012). "Large Scale Sketch Based Image Retrieval Using Patch Hashing," *Springer-Verlag Berlin Heidelb.*. 210–211
6. Brownlee J (2020) "A Gentle Introduction to Threshold-Moving for Imbalanced Classification", [Online] *Machine Learning Mastery*. Available at: https://machinelearningmastery.com/threshold-moving-for-imbalanced-classification/ (accessed 03 Nov 2020)
7. Chen T, Wang Z, Li G, Lin L (2017) Recurr Attent Reinforcement learning for multi-label image recognition. SenseTime Gr Ltd:6730–6737
8. El-Makhfi N (2019) A word spotting method for Arabic manuscripts based on speeded up robust features technique. Adv Sci Technol Eng Syst J 4(6):99–107
9. Elnagar A, Al-Debsi R, Einea O (2020) Arabic text classification using deep learning models. Inf Process Manag 57:102–121
10. Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: a survey. J Artif Intell Res 4:237–285
11. Kekre HB and Mishra D (2011) "Content based image retrieval using weighted hamming distance image hash value. 284–285
12. Khayyat M, Elrefaei L (2020) Manuscripts image retrieval using deep learning incorporating a variety of fusion levels. IEEE Access J 8:136460–136486
13. Khayyat M, Elrefaei L (2020) Towards author recognition of ancient arabic manuscripts using deep learning: a transfer learning approach. Int J Comput Digital Syst (IJCDS) 9(4):1–18

14. Khayyat M and Elrefaei L(2020) "A Deep Learning Based Prediction of Arabic Manuscripts Handwriting Style," Accepted Manuscript for publication in the International Arab Journal of Information Technology, *(IAJIT)*. 17(5)
15. Lin E, Chen Q, Qi X (2020) Deep reinforcement learning for imbalanced classification. Appl Intell 50: 2488–2502
16. Marr B (2018) "Artificial Intelligence: What's The Difference Between Deep Learning And Reinforcement Learning?," [Online] *Forbes*. Available at: https://www.forbes.com/sites/bernardmarr/2018/10/22/artificial-intelligence-whats-the-difference-between-deep-learning-and-reinforcement-learning/#6e010ff7271e (accessed 05 Jun 2020).
17. Mysiak K (2020) "Classification Metrics & Thresholds Explained", [Online] *towards data science*. Available at: https://towardsdatascience.com/classification-metrics-thresholds-explained-caff18ad2747 (accessed 03 Nov 2020)
18. Nie W, Wang W, Liu A, and Chen C (2019) "Characteristic Views Extraction Modal Based-On Deep Reinforcement Learning For 3d Model Retrieval," IEEE Int. Conf. Image Process. 2389–2393
19. Othman RAA (2015) "Arabic Manuscripts Analysis and Retrieval," Ph.D. dissertation, *Dept. Inf. Comput. Sci., King Fahd Univ. Petroleum Minerals, Dhahran, Saudi Arabia*, pp. 1–199
20. Peng Y, Zhang J, and Ye Z (2018) "Deep Reinforcement Learning for Image Hashing," arXiv:1802.02904 2: 1–12. [Online]. Available: http://arxiv.org/abs/1802.02904.
21. Saritha RR, Paul V, Kumar PG (2019) Content based image retrieval using deep learning process. Cluster Comput. 22:4187–4200
22. Schuderer A, Bromuri S, and Van Eekelen M (2021) "Sim-Env: Decoupling OpenAI Gym Environments from Simulation Models". 1: 1–17
23. Shi X, Sapkota M, Xing F, Liu F, Cui L, Yang L (2018) Pairwise based Deep Ranking Hashing For Histopathology Image Classification and Retrieval. Pattern Recognit. 81:14–22
24. Sutton RS, Barto AG (2015) Reinforcement learning : an introduction. Cambridge, MA, USA: MIT Press:1–338
25. Varga D, Sziranyi T (2016) Fast content-based image retrieval using convolutional neural network and hash function. IEEE Int Conf Syst, Man, Cybernetics - SMC:2636–2640
26. Wang H, Wang K, Wu Y, Wang Z, Zou L (2020) User preference-aware video highlight detection via deep reinforcement learning. Multimed Tools Appl 79:15015–15024
27. Wang J, Kumar S, Chang SF (2010) Semi-Supervised Hashing for Scalable Image Retrieval. IEEE Comp Soc Conf Comp Vision Pattern Recogn:3424–3431
28. Wang J, Shen HT, Song J, and Ji J (2014) "Hashing for Similarity Search : A Survey," *arxiv:1408.2927*, pp. 1–29, . [Online]. Available at: https://arxiv.org/abs/1408.2927 (accessed 19 Feb 2020).
29. Wiering M, and Otterlo MV (2012) "Reinforcement learning: state-of-the-art," *Springer-Verlag Berlin Heidelberg,* vol, 12.
30. Xia R, Pan Y, Lai H, Liu C, Yan S (2014) Supervised hashing for image retrieval via image representation learning. Proceed Twenty-Eighth AAAI Conf Artifi Intell:2156–2162
31. Yahia MHN (2011) "Content-Based Retrieval of Arabic Historical Manuscripts Using Latent Semantic Indexing," Ph.D. dissertation, *Dept. Inf. Comput. Sci., King Fahd Univ. Petroleum Minerals, Dhahran, Saudi Arabia*, pp. 1–98..
32. Yao J, Dou Z, Xu J, and Wen J (2020) "RLPer : A Reinforcement Learning Model for Personalized Search, " *In Proceedings of TheWeb Conference* 2020 *(WWW'20), ACM, New York, NY, USA*, pp. 2298–2308
33. Zhang R, Lin L, Zhang R, Zuo W, Zhang L (2015) Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. IEEE Trans Image Process 24(12):4766–4779
34. Zhao D, Chen Y, Lv L (2017) Deep reinforcement learning with visual attention for vehicle classification. IEEE Trans Cogn Dev Syst 9(4):356–367
35. Zhao F, Huang Y, Wang L, Tan T (2015) Deep semantic ranking based hashing for multi-label image retrieval. IEEE Xplore:1556–1564
36. Zhou J and Agichtein E (2020). "RLIRank : Learning to Rank with Reinforcement Learning for Dynamic Search," In Proceedings of TheWeb Conference 2020 (WWW'20), ACM, New York, NY, USA, pp. 2842–2848