The Fourteenth IEEE International Conference on e-Business Engineering

A Face-Recognition Approach Using Deep Reinforcement Learning Approach for User Authentication

Ping Wang, Wen-Hui Lin*
Department of Information Management
Kun Shan University
Tainan, Taiwan
pingwang@mail.ksu.edu.tw
linwh@mail.ksu.edu.tw

Kuo-Ming Chao
Professor of Engineering and Computing, School of MIS, Coventry University, UK
k.chao@coventry.ac.uk

Chi-Chun Lo
Institute of Information Management
National Chiao Tung University
Hsinchu, Taiwan
cclo@faculty.nctu.edu.tw

Abstract—**Numerous crime-related security concerns exist in e-commerce transactions recently. User authentication for mobile payment has numerous approaches including face recognition, iris scan, and fingerprint scan to identify user's true identity by comparing the biometric features of users with patterns in the signature database. Existing studies on the face recognition problem focus mainly on the static analysis to determine the face recognition precision by examining the facial features of images with different facial expressions for users rather than the dynamic aspects where images were are often vague affected by lighting changes with different poses. Because the lighting, facial expressions, and facial details varied in the face recognition process. Consequently, it limits the effectiveness of scheme with which to determine the true identity. Accordingly, this study focused on a face recognition process under the situation of vague facial features using deep reinforcement learning (DRL) approach with convolutional neuron networks (CNNs) thru facial feature extraction, transformation, and comparison to determine the user identity for mobile payment. Specifically, the proposed authentication scheme uses back propagation algorithm to effectively improve the accuracy of face recognition using feed-forward network architecture for CNNs. Overall, the proposed scheme provided a higher precision of face recognition (100% at gamma correction γ located in [0.5, 1.6]) compared with the average precision for face image (approximately 99.5% at normal lighting γ =1) of the existing CNN schemes with ImageNet 2012 Challenge training data set.**

Keywords- Face recognition, e-commerce, Deep reinforcement learning, Convolutional neuron networks, Back propagation

I. Introduction

Mobile payment offers more convenient services for customers and sellers by using the e-commerce platform involving the employment of biometric authentication schemes to identify user's true identity by comparing the biometric features of users with patterns stored in the signature database; i.e., mobile payment generally operated under financial regulation and performed from or via a mobile device, therefore, user authentication data on the mobile devices might become targets of network attacks.

Identity theft is therefore one of the most severe threats to the security of mobile payment in online transactions of e-commerce services. A well-designed mobile transaction system can actually enable safer payments and transfers for in-store, ATM and online channels, protecting consumers, merchants, banks and payment processors from the high cost of fraud. [1] Security concerns of mobile payments may be a bottleneck of mobile payments and digital banking. Consequently, it is imperative that service providers have the means to authenticate the identity of each user for detecting fraud. To guarantee the transaction security of mobile payment, cloud service providers (CSPs) recently use biometric authentication approaches including face recognition, fingerprint, and iris scan to increase consumer trust in mobile payments and digital banking.

In performing face detection and recognition against identity theft problems, CSPs can use a static analysis to determine the face recognition precision by examining the facial features of images with different facial expressions for users. However, images of face detection in mobile payment were often vague affected by the different direction of lighting or poses. Because the lighting, facial expressions, and facial details varied in a real situation of face recognition process. As a result, such approaches are vulnerable to identity theft and additional face recognition approaches with facial feature are proposed to verify the true identity of a user.

Inspired by GoogLeNet on image classification, CSPs used the deep learning (DL) approach with deep neural networks (DNNs) to successfully increase the recognition precision for vague face image. Deep learning is a class of machine learning (MA) algorithms which use a cascade of many layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input. [2]

In 2011, deep learning feedforward networks used convolutional layers and max-pooling layers, namely convolutional neural networks (CNNs) topped by several fully connected or sparsely connected layers followed by a final classification layer. [3] Specifically, CNNs have produced excellent results in both image and speech applications from 2012. To speed up the learning of feature learning and recognition, this study proposes an improved deep feature learning algorithm incorporating the deep reinforcement learning (DRL) approach with CNNs for face recognition system (DFRS) under the situation of vague mental images. Precisely, the DRL approach uses the

978-1-5386-1412-9/17 $31.00 © 2017 IEEE
DOI 10.1109/ICEBE.2017.36

principles of the back propagation algorithm to adjust the weights with respect to the parameters of model by computing the gradient of a cost function at the output layer. The experimental results show that the DFRS provides a more high recognition precision under the situation of vague mental images for mobile payment security in avoiding the problem of identity fraud.

The remainder of this paper is organised as follows. Section II reviews previous studies in this field. Section III introduces the proposed face recognition model based on the DRL approach associated with CNN architecture. A performance analysis of the results is presented in Section VI. Finally, Section V provides the conclusions of this study.

II. Relate Work

This section reviews the use of three important issues, namely framework design of CNNs, CNN for face recognition applications and Q-learning algorithm.

A. *Framework Design of CNNs*

The convolutional neural network (CNN) is a type of deep neural network which learns the variations from the dataset without any prior knowledge. More concisely, the CNNs are biologically-inspired variants of multi-layer perceptrons (MLPs) representing by a class of feedforward neural network. [3] Generally, a CNN consists of an image input and a classification output layer, as well as multiple hidden layers. The hidden layers are either convolutional, rectified linear unit (ReLU), pooling, or fully connected which are illustrated as follows. [3,6]

Image Input

Image input defines the size of the input images of a convolutional neural network and contains the raw pixel values of the images. Inspired by Girshick et al. [4], "recognition using regions" paradigm is used in following image preprocess to generate category independent region proposals and extracts features from the regions. [5]

Convolutional Layer

A convolutional layer consists of neurons that connect to sub-regions of the input images or the outputs of the layer before it. A convolutional layer learns the features localized by these regions while scanning through an image. For each region, computes a dot product of the weights and the input, and then adds a bias term. A set of weights that are applied to a region in the image is called a 'filter'. Furthermore, the filter moves along the input image vertically and horizontally, repeating the same computation for each region, i.e., convolving the input. The number of weights used for a filter is h*w*c, where h is the height, and w is the width of the filter size, and c is the number of channels in the input. [6]

The number of filters determines the number of channels in the output of a convolutional layer. As a filter moves along the input, the filter uses the same set of weights and bias for the convolution, forming a feature map. Hence, the number of feature maps a convolutional layer has is equal to the number of filters. Each feature map has a different set of weights and a bias. The results from the neurons of a convolutional layer usually pass through some form of nonlinearity. In practice, the number of convolutional layers depends on the amount and complexity of the image data. The results from the neurons of a convolutional layer usually pass through some form of nonlinearity. Generally, CNNs uses rectified linear units (ReLU) function for this purpose where each convolutional layer can be followed by ReLU layer or a pooling layer.

ReLU Layer

In general, a convolutional layer is usually followed by a nonlinear activation function. It performs a threshold operation to each element, where any input value less than zero is set to zero, i.e.,

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0 & x < 0 \end{cases}, \quad (1)$$

Pooling Layers

Pooling layers follow the convolutional layers for down-sampling, hence, reducing the number of connections to a fully-connected layer. Thus, pooling layers decrease the number of parameters to be learned in the following layers and help reduce overfitting.

Fully Connected Layer (FCLs)

The convolutional layers are followed by one or more fully connected layers. As suggests, all neurons in a fully connected layer connect to the neurons in the previous layer. FCL combines all of the features (sub-region information) learned by the previous layers across the image to identify the larger patterns. For classification problems, the last fully connected layer *synthesizes* them to classify the images.

Classification Layers

The softmax function is used in various multiclass classification methods. The softmax function is also known as the normalized *exponential* (as shown in Eq.2) and can be considered the multi-class generalization of the logistic sigmoid function. [7] For classification problems, a softmax layer and then a classification layer must follow the final fully connected layer. The output unit activation function is the softmax function: In mathematics, the softmax function is a generalization of the logistic function that compresses a k-dimensional vector of *r* arbitrary real values to a k-dimensional vector $y_r(x)$ of real values in the range [0, 1] that add up to 1. The function is given by

$$y_r(x) = \frac{\exp(a_r(x))}{\sum_{j=1}^{k} \exp(a_j(x))}, \quad \text{for } j = 1, \ldots, k \quad (2)$$

where a_r represents the conditional probability of the given class r, $0 \leq y_r(x) \leq 1$. In practice, the softmax function highlights the largest values and suppresses values which are significantly below the maximum value. The output of the softmax function can be used to represent a categorical distribution, i.e., a probability distribution over K different possible outcomes. More concisely, the softmax function is the output unit activation function after the last fully connected layer for multi-class classification problems as

$$p(c_r \mid x,\theta) = \frac{p(x_r,\theta \mid c_r)p(c_r)}{\sum_{j=1}^{k} p(x_j,\theta \mid c_j)p(c_j)} = \frac{\exp(a_r \mid x,\theta)}{\sum_{j=1}^{k}\exp(a_j \mid x,\theta)}, \quad (3)$$

where $0 \le P(c_r \mid x,\theta) \le 1$ and $\sum_{j=1}^{k} p(c_j \mid x_j,\theta) = 1$. Moreover $a_r = \ln(p(x_r,\theta \mid c_r)p(c_r))$, $p(x_r,\theta \mid c_r)$ represents the conditional probability of the same given class r and $p(c_r)$ is the class prior probability.

B. Convolutional Neural Networks for face recognition applications

CNNs are used for multi - class classification which classifies an image into one of N identity classes in the use of face recognition, as shown in Fig.1. The user authentication problem is most frequently discussed regarding machine learning (ML) techniques for feature extraction and training in e-commerce transaction security. Many classification approaches incorporate layer-wise CNNs with classification algorithms for face detection and recognition [3-6,8-10]. Face recognition is to classify one of N identity classes with recognized facial features by using a test set of distinct images including the positives and negatives in learning process to measure the prediction accuracy of face recognition.

The advantages of CNNs using it are: 1) it can be applied to deal with a large amount of training data, 2) CNNs can automatically learn features to capture complex visual variations by leveraging a large amount of training data, 3) In testing phase, CNNs can be easily parallelized on GPU cores for acceleration and thousands of CPU cores have been used to get the result with a less time, and 4) by classifying training images into a large amount of identities, the last hidden layers (full-connected layers) of deep neural networks would form rich identity-related features[6].

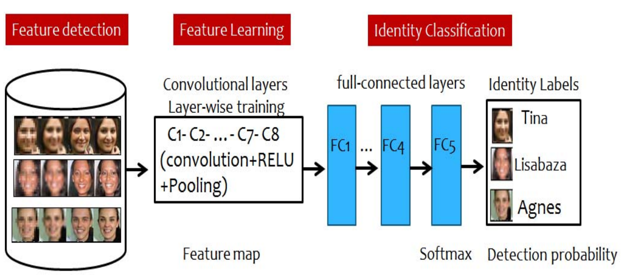

FIGURE 1. FEATURE LEARNING AND RECOGNITION BASED ON CNNS.

In the case of CNNs, the classification is used to determine the error of the network and then adjust the network to minimize it, such as DeepFace developed by Facebook. It identifies human faces in digital images by employing a nine-layer neural net with over 120 million connection weights associated with four-million training images uploaded by Facebook users.[11] The average success rate of facial recognition system was approximately 97%, compared to 85% for the FBI's Next Generation Identification system. [12]

Numerous face-recognition techniques have been used in analysis algorithms for achieving high authentication precision, for example, the global precision of face recognition was over 99.63% with labelled faces in the wild (LFW) using 200,000,000 images for Google deep learning results in 2007. Later on, a new approach of deep learning using both face identification and verification was proposed in Deep IDentification (DeepID) [13] to develop effective feature representations for reducing intra-personal variations along with layer-wise CNNs. In further, DeepID2 nets[15]was developed by using 25 face regions were trained where each network taking a particular face region as input. Later on, learning discriminative deep face representation was developed through large-scale face identity classification (face identification) in DeepID2 [14], and DeepID2+[15]. Consequently, 99.15%, 99.45% face verification accuracy was achieved with labelled faces in the LFW dataset, respectively. Recently, DeepID3[11] was improved by Sun et al., where two architectures are rebuilt from stacked convolution and inception layers. As a result, an ensemble of the proposed two architectures achieves 99.53% LFW face verification accuracy.

Practically, a network designer has to decide design issues including the number of layers of each type, the ordering of layers, and the hyper parameters for each type of layer in constructing a CNN. [16] As a result, over-exploration can lead to slow convergence times In traditional reinforcement learning settings, yet over-exploitation can lead to convergence to local minima [17]. Thus, the present study uses the DRL algorithm incorporating the agents to solve the problem of vague mental image by revising feed-forward network architecture for CNNs.

C. Q-learning algorithm

A recent application of Q-learning to deep learning, by Google DeepMind, namely 'deep reinforcement learning' or 'deep Q-networks', has been successful at playing some Atari 2600 games at expert human levels. Here, this study incorporates the principles of Q-learning and DNNs to optimize agents suited for solving face recognition for user authentication in the case of vague mental images. The problem model for Q-learning algorithm consists of an agent which has various states $s_t \in S$ and a set of actions per state A. By performing an action $a_t \in A$, the agent can move from state to state. Executing an action in a specific state provides the agent with a reward (τ_t) in the form of numerical score. The goal of the agent is to maximize its total reward. This reward is a weighted sum of the expected values of the rewards of all future steps starting from the current state, where the weight for a step from a state Δt steps into the future is calculated as $r^{\Delta t}$. [18]

The algorithm therefore has a function that calculates the Quantity of a state-action combination: [19]

$$Q : S \quad x \quad A \rightarrow R \tag{4}$$

Q-learning learns estimates of the optimal Q-values of an Markov decision process (MDP), which means that behavior can be dictated by taking actions greedily with respect to the learned Q-values. At each step *s*, choose the action *a* which maximizes the function Q. Q is the estimated utility function $Q(s_t, a_t)$, it reveals us how good an action is given a certain state. The core of the algorithm is a simple value iteration update. It assumes the old value and makes a correction based on the new information.

Before learning has started, Q-value returns an (arbitrary) fixed value, chosen by the developer. Then, each time the agent selects an action, and observes a reward () and a new state that may depend on both the previous state and the selected action, Q is updated. The core of the algorithm is a simple value iteration update as follows. It assumes the old value and makes a correction based on the new information.
A detailed algorithm for classifying the user by using Q-table learning is described as follows. [20]

1) Initialize Q-values $Q(s_t, a_t)$ arbitrarily for all state-action pairs.
2) For life or until learning is stopped.
3) Then, each time the agent selects an action (*a*) in the current state (*s*) based on current Q-value estimates ($Q(s_t, \bullet)$). A new state (s_{t+1}) that may depend on both the previous state (s_t)) and the selected action, Q is updated. Typically, lookup tables used for storing the Q-values are implemented as a matrix, i.e., $[Q(s_t, a_t)_{t=1,\ldots,T}]$.

4) update rule: select the action that has the largest Q-value.

$$Q(s_t, a_t) = \underset{old\ value}{(1-\alpha)Q(s_t, a_t)} + \alpha[(\underset{reward}{\tau_t} + \gamma . \underset{\max estimated\,value}{\max_a Q(s_{t+1}, a)} - Q(s_t, a_t)], \tag{5}$$

where α is the learning rate parameter, $\alpha \in [0,1]$. Generally, use a learning rate of 0.001 for deep feature learning. It updates the Q-value of the last state-action pair (*s*, *a*) with respect to the observed outcome state (s_{t+1}) and the reward observed for the current state (τ_t) which is a static reward value that is received by the agent for taking an action in a given state.

III. Face-Recognition Model with a Pre-trained Deep Neural Network Network

The proposed model was designed to deep feature learning based on Inception-v3 model pre-trained on a large facial image database, ImageNet to speed up face recognition on transfer learning.

Face recognition for user authentication comprises two main steps, namely the face detection and recognition. The difficulties in face detection mainly come from two aspects: 1) the large visual variations of human faces in the cluttered backgrounds; 2) the large search space of possible face positions and face sizes. Existing schemes may not be effective for faces due to small-sized faces, lighting and complex appearance variations, i.e., face recognition for unconstrained video-surveillance environments is a highly demanding task, and needs several pre-processing to be usable.

Obviously, lighting affects the image quality and CSPs might regard the different images of an individual. Inspired by Chen et al. [21], they used the face alignment to increase the recognition precision. Thus, the present study uses CNNs framework with detection window alignment preprocess for landmark calibration to improve the recognition precision of images affected by lighting in the user authentication process.

Detailed workflow from face recognition using Inception v3 model is shown in Fig. 2, which presents a function flow diagram of the proposed face-recognition scheme; Fig. 2 illustrates the three subphases in the user authentication process: i) the feature extraction phase, ii) the feature training phase, and iii) the face recognition phase.

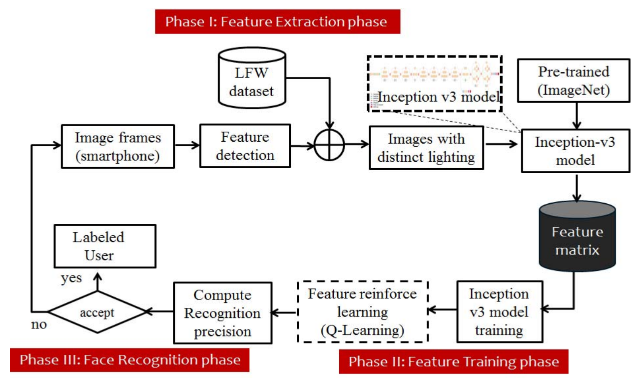

Figure 2. Face recognition using Inception-v3 model with pretrained ImageNet.

To quick prompt to classify the images, we incorporate Inception v3 model where facial features were pre-trained using the ImageNet 2012. [22] Inception-v3 model is composed of one or more convolutional layers with fully connected layers on top for matching those in typical artificial neural networks, where models try to classify entire images into 1000 classes.

In feature extraction phase, take a video from a smartphone and then split it into image frames. Image frames can be used to recover from misses from face detection including the two pre-processing tasks for images: i) the lighting processing is used as photometric normalization to increase both lighting invariance and recognition performance, and detect whether an image is too blurry to be usable for recognition and ii) alignment is for geometrically normalizing the detected face so that facial features including eyes, nose, and mouth are at canonical positions in image before recognition. [23]

In feature training phase, upload the images to the cloud platform and train data using Inception v3 model that led to huge performance increases by employing a forty-eight-layer neural net to classify the images using feature matrix pre-trained from the ImageNet dataset. In face recognition phase, calculate the recognition precision for the proposed scheme and label the specific user after performing the successful recognition.

IV. Experimental Results

An experiment was performed to validate the proposed face-recognition scheme could be provided, where user faces from both ImageNet 2012 Challenge training data set and labelled faces in the LFW dataset [24] were used for making comparisons and to calculate the recognition precision of the proposed scheme. A frequently used ImageNet 2012 Challenge training data set in machine learning for pre-training an image recognition system, which is allowing developers to construct an image classifier rapidly.

The testing environment is built a PC with Intel i7-6700 processor associated with installing Ubuntu operating system 14.04 LTS 64bit, 32GB RAM, DDR4(16GB*2), SSD: 120GB, and Nvidia GeForce GTX 1080 TI Graphics Cards as the host. In the proposed project, the image analyser uses a software tools including Python2.7, TensorFlow, Opencv-python(2.4.13), Openjdk-1.8, Bazel, Nvidia-driver 375.66, Nvidia-cuda 8.0, and Nvidia cudnn 5.1 in an Ubuntu operation system to conduct the neural network configurations and perform the experiment process described in Fig.2.

A total of 250 subjects of image were downloaded from the LFW database. For each subject, the face image varied in the 24 frames taken per 15 degree rotation and a total of 6,000 training images were obtained. These images were normalised to a size of 250 × 250 pixels, were used in the following experiments. In the experiment, feature extraction involved lighting pre-process and detection window alignment.

In performing face recognition for online transactions, it is hard to determine the true identity due to the lighting varied in the face recognition process. Fig. 3 shows the images from left to right are a little dark due to dim light for gamma correction value= 0.05~0.6. Conversely the images are excessive white if the light is particularly bright where gamma correction value =1.5~6.0. A simulation of lighting varied was used to examine the recognition capability of the proposed model for vague mental images by changing gamma-value (γ =0.05~6.0) for each test image.

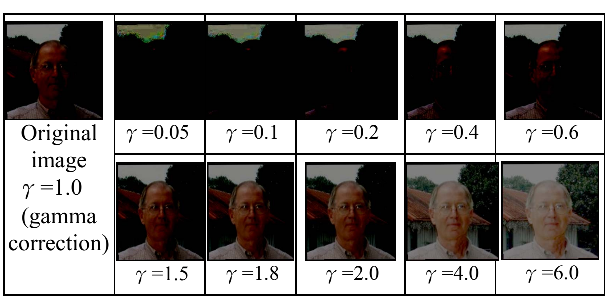

Figure 3. Images affected by a series of lighting changes

To demonstrate the robustness of the proposed method, the present study uses the Inception-v3 model pre-trained on ImageNet incorporating with Q-learning algorithm to fine-tune our model, as shown in Fig.4. Specially, the backpropagation algorithm based on Q-learning modifies the network weights so as to automatically adjust the model parameter (θ) and reduce the predicted error between the output of the network and the desired output. In other words, it works by learning an action-value function that ultimately gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. [18]

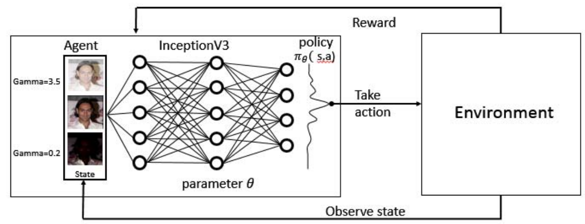

Figure 4. Face recognition using pre-trained Inception-v3 model with Q-Learning.

At initial, the mages was normalised to a size of 250 × 250 pixels used a state of input image , where s_t denotes a state at time t, a_t denotes an action of developer's label at time t , and $Q(s_t, a_t, \theta)$ denotes the corresponding output of softmax layer in the following experiments, where θ is model parameter adjusted (i.e., weighs of hidden layers). $Q^*()$ is known as the action-value function and individual $Q^*(s_t, a_t, \theta)$ are known as Q-values.

$$Q^*(s_t, a_t, \theta) = E[\tau_t + \gamma.\max_a Q^*(s_{t+1}, a\ , \theta)] \quad (6)$$

To fine-tune the weights of deep learning using CNNs, we define the loss function of Q-learning, (s_t, a_t, θ) as [24]

$$L(\theta) = E[(\tau_{t+1} + \gamma.\max_a Q(s_{t+1}, a\ , \theta) - Q(s_t, a_t, \theta))^2], \quad (7)$$

where γ is the discount factor 1 $(0 \le \gamma \le 1)$ which trades off the importance of sooner against later rewards. Here, use a stochastic gradient descent to minimize the error of a model on our training data. The gradient descent of the loss function for Eq. (7) is calculated as

$$\nabla_\theta L(\theta) = \frac{\partial L(\theta)}{\partial \theta} = E[(\tau_{t+1} + \gamma.\max_a Q(s_{t+1}, a\ , \theta) - Q(s_t, a_t, \theta))\frac{\partial Q(s_t, a_t, \theta)}{\partial \theta}], \quad (8)$$

Eq.(8) can assist the designer to find the set of weights that result in the smallest error for our model on the training data when the result of Eq.(8) is approaching 0 . [25]

The training parameters were setting to as follows: the batch number = 32, the learning rate (α) = 0.001. Considering the distance between starting state and terminal states, the parameter γ was set to 0.9 to allow positive Q-values to propagate faster and setting an immediate positive reward of 0.3 for a state. [26] Finally, the training converged at16000 iterations with loss = 1.76.

The experimental results using the pre-trained Inception V3 neural network modules with Q-table learning to fine-tune the previous model from the LFW data set is shown as in Table 1 and Fig.5. In Fig.5, the recognition precision is affected by lighting intensity, i.e., the recognition precision decreases when the light intensity is excessive dim or bright. Fig.5 shows that the average success rate of facial recognition using top-5 score where the top 5 class having the highest probability was approximately 100.0% when Gamma correction values are located in the interval [0.6, 1.5]. Overall, the proposed scheme provided a higher precision of face recognition (100% at gamma correction γ located in [0.5, 1.6]) compared with the average precision for face image (approximately 99.5% at normal lighting γ =1) of the existing CNN schemes.

TABLE 1. SUCCESS PERCENTAGE FOR THE USE OF DIFFERENT LIGHTING FOR LFW DATA SET.

Sizing of training dataset	Gamma-correction value	Number of success recognition /Number of total images(LFW)	Top-1	Top-5
250	0.05	3/250	0.012	0.036
250	0.1	6/250	0.024	0.064
250	0.2	47/250	0.0188	0.384
250	0.4	197/250	0.788	0.948
250	0.6	247/250	0.988	1
250	1	250/250	1	1
250	1.5	249/250	0.996	1
250	1.8	221/250	0.884	0.976
250	2	207/250	0.828	0.936
250	3	112/250	0.448	0.716
250	4	74/250	0.296	0.548
250	6	36/250	0.144	0.328

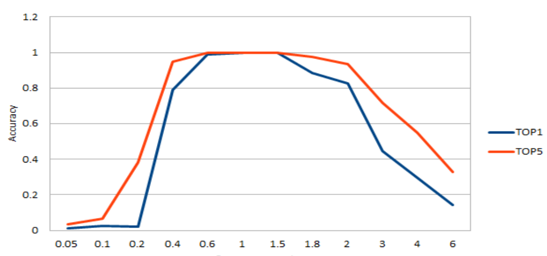

FIGURE 5. FACE RECOGNITION PRECISION USING INCEPTION-V3 MODEL WITH PRE-TRAINED IMAGENET

V. CONCLUSIONS

This paper has presented a face recognition model that uses a pre-training CNNs-based classifier to enhance the recognition precision of DFRS under the situation of vague mental images. Importantly, the proposed approach incorporates the Q-table learning to fine-tune the previous model in different types of lightings. Overall, the results have shown that the model proposed in this study can enhance the recognition precision compared with the average precision for face image of the existing CNN schemes.

ACKNOWLEDGES

This work was supported jointly by the Ministry of Science and Technology of Taiwan under Grant Nos. MOST 106-2632-E-168-001, MOST 106-3114-E-492-00 and MOST 106-2410–H -168-002.

REFERENCES

[1] BioID, BioID mobile payment be recognized the way you are, https://www.bioid.com/Solutions/Mobile-Payment

[2] Wikipedia, Deep learning, https://en.wikipedia.org/wiki/Deep_learning

[3] Wikipedia, Convolutional neural networks, Wikipedia, convolutional neural networks, https://en.wikipedia.org/wiki/ convolutional neural networks

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv preprint arXiv:1311.2524, 2013.

[5] H. Li, , et al. "A Convolutional Neural Network Cascade for Face Detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015

[6] The MathWorks, Inc, Specify Layers of Convolutional Neural Network, https://www.mathworks.com/help/nnet/ug/layers-of-a-convolutional-neural-network.html

[7] C. M. Bishop, Pattern Recognition and Machine Learning. Springer, New York, NY, 2006.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge, 2014.

[9] F., S. Sudhakar, M. Saberian, and L.J. Li, Multi-view face detection using deep convolutional neural networks." arXiv preprint arXiv:1502.02766 (2015).

[10] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2012

[11] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the gap to human-level performance in face verification. In Proc. CVPR, 2014.

[12] Y. Sun, D Liang, X. Wang, X. Tang, Deepid3: Face recognition with very deep neural networks, arXiv preprint arXiv:1502.00873, 2015

[13] Y. Sun, X.Wang, and X. Tang. Hybrid deep learning for face verification. In Proc. ICCV, 2013.

[14] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In Proc. CVPR, 2014.

[15] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. arXiv: 1412.1265, 2014.

[16] B. Baker, O. Gupta, N. Naik & R. Raskar, Designing Neural Network Architectures using Reinforcement Learning Media Laboratory, MIT, arXiv:1611.02167, 2016.

[17] L. P. Kaelbling, M. L Littman, and A. W Moore. Reinforcement learning: A survey. Journal of Artificial Intelligence Research, 4:237–285, 1996.

[18] Wikipedia, Q-learning, https://en.wikipedia.org/wiki/Q-learning

[19] BURLAP, Tutorial: Creating a Planning and Learning Algorithm, http://burlap.cs.brown.edu/tutorials/cpl/p3.html

[20] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. Joint cascade face detection and alignment. In Computer Vision–ECCV 2014. 2014.

[21] J. Zhang, S. Shan, M. Kan, and X. Chen. Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In Computer Vision–ECCV 2014.

[22] Stanford Vision Lab, the ImageNet 2012 Challenge training data set, http://www.image-net.org/challenges/LSVRC/2012/

[23] N. Vu, S. Schwab, P. Bouges, X. Naturel, C. Blanc, T. Chateau L. Trassoudaine, Face Recognition for Video Security Applications, VESALIS 2Université Blaise.

[24] University of Massachusetts, Labeled faces in the LFW, http://vis-www.cs. umass.edu/lfw/

[25] D. Plaut and G. Hinton. Learning sets of filters using back-propagation. Computer Speech & Language. 2(1), pp. 35-61, 1987.

[26] V. Mnih, K. Kavukcuoglu, D. Silver and et al., Human-level control through deep reinforcement learning, Nature. 2015 Feb 26;518(7540):529-33.