

DocFac ID Document to Selfie Matching

Submitted in partial fulfillment of the requirements for the award of Bachelor of science degree in
Computer Science

By

SOWMIYA.K(Reg.No:38290084)

SUDHA.R(Reg.No:38290090)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119, TAMILNADU

APRIL 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119
www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SOWMIYA.K(Reg. No. 38290084)** and **SUDHA.R (Reg. No. 38290090)** who carried out the project entitled "**DocFace ID DOCUMENT TO SELFIE MATCHING**" under my supervision from November 2020 to April 2021.

Internal Guide
(Dr. P. ASHA, M.E, Ph.D.)

Head of the Department
(Dr. L. LAKSHMANAN, M.E., Ph.D.,)

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **SOWMIYA.K(Reg. No. 38290084)** hereby declare that the Project Report entitled **"DOC FACE ID DOCUMENT TO SELFIE MATCHING"** done by me under the guidance of Dr. P.ASHA ,M.E,Ph.D., is submitted in partial fulfillment of the requirements for the award of Bachelor of Science Degree in Computer Science.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D.**, Dean, School of Computing and **Dr. S. VIGNESHWARI, M.E., Ph.D.**, and **Dr. L. LAKSHMANAN, M.E., Ph.D.**, **Head of the Department**, Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. P.ASHA, M.E, Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work. I wish to express my thanks to all Teaching and Non-teaching staff members of the Completion of the project.

ABSTRACT

Numerous activities in our daily life require us to verify who we are by showing our ID documents containing face images, such as passports and driver licenses, to human operators. However, this process is slow, labor intensive and unreliable. As such, an automated system for matching ID document photographs to live face images in real time and with high accuracy is required. In this paper, we propose DocFace to meet this objective. We first show that gradient-based optimization methods converge slowly when many classes have very few samples, a characteristic of existing ID-selfie datasets. To overcome this shortcoming, we propose a method, called dynamic weight imprinting, to update the classifier weights, which allows faster convergence and more generalizable representations. Next, a pair of sibling networks with partially shared parameters are trained to learn a unified face representation with domain-specific parameters. Cross-validation on an ID-selfie dataset shows that while a publicly available general face matcher only achieves a true accept rate.

TABLE OF CONTENTS

CHAPTER N O	TITLE	PAGE.NO
	ABSTRACT	Vi
	LIST OF ABBREVIATIONS	Vii
1	INTRODUCTION	10
	1.1 AIM	10
	1.2 SYNOPSIS	10
2	LITERATURE SURVEY	12
	2.1 REVIEW ON LITERATURE WORK	12
	2.2 EXISTING SYSTEM	13
	2.3 PROPOSED SYSTEM	14
3	METHODOLOGY	15
	3.1 REQUIREMENT SPECIFICATION	15
	3.1.1 HAREWARE SPECIFICATION	15
	3.1.2 SOFTWARE SPECIFICATION	15
	3.1.3 TECHNOLOGY USED	15
	3.1.4 ALOGORITHUM	16
	3.2 ADVANTAGES AND DISADVA	16

	3.2.1 ADVANTAGES	16
	3.2.2 DISADVANTAGES	16
	3.3 SYSTEM DESIGN	16
	3.3.1 ACTIVITY DIAGRAM	17
	3.3.2 DATAFLOW DIAGRAM	18
	3.3.3 ARCHITECTURE DIAGRAM	20
	3.4 SYSTEM ALGORITHM	21
	3.5 MODULES	22
	3.5.1 MODULES EXPLANATION	22
	3.5.2 USER REGISTRATION AND AUTHENTICATION	23
	3.5.3 USER UPLOAD CERTIFICATE	23
	3.5.4 GET CERTIFICATE	23
	3.5.5 QR REQUEST AND RESPONSE FROM VERIFICATION AUTHORITY	23
4	SYSTEM FEATURE	24
	4.1 DESIGN AND IMPLEMENTATION	24
	4.1.1 CONSTRAINTS IN ANALYSIS	24
	4.1.2 CONSTRAINTS IN DESIGN	24
	4.1.3 CONSTRAINTS IN IMPLEMENTATION	25

5	RESULT AND DISCUSSION	26
6	CONCLUSION AND FUTURE WORK	28
	6.1 CONCLUSION	28
	6.2 FUTURE WORK	29
	REFERENCES	29
	APPENDIX	32
	A.SCREEN SHOT	32
	B.SOURCE CODE	37

LIST OF ABBREVIATIONS

JDK	Java Development Toolkit
DEX	Dalvik Executables
TCP	Transmission Control Protocol
IP	Internet Protocol
HTTP	Hyper Text Transfer Protocol
ADT	Android Development Tool

CHAPTER 1

INTRODUCTION

IDENTITY verification plays an important role in our daily lives. For example, access control, physical security and international border crossing require us to verify our access (security) level and our identities. A practical and common approach to this problem involves comparing an individual's live face to the face image found in his/her ID document. An application scenario of the DocFace+ system. The scans the ID document photos or reads the photo from the embedded chip and the camera takes another photo of the holder's live face (selfie). By comparing the two photos, the system decides whether the holder is indeed the owner of the ID document.

Aim:

The main aim of this project is to solve the problem of counterfeiting certificates we are proposing an digital certificate system based on block chain technology and to verify the traveler's identity using live camera, which allows faster convergence and more generalizable representations.

Synopsis:

Numerous activities in our daily life require us to verify who we are by showing our ID documents containing face images, such as passports and driver licenses, to human operators. However, this process is slow, labor intensive and unreliable. As such, an automated system for matching ID document photos to live face images (selfies) in real time and with high accuracy is required. In this paper, we propose DocFace+ to meet this objective. We first show that gradient-based optimization methods converge slowly (due to the underfitting of classifier weights) when many classes have very few samples, a characteristic of existing ID-selfie datasets. To overcome this shortcoming, to update the classifier weights, which allows faster convergence and more

generalizable representations. Next, a pair of sibling networks with partially shared parameters are trained to learn a unified face representation with domain-specific parameters. Cross-validation on an ID selfie dataset shows that while a publicly available general face matcher.

CHAPTER 2

LITERATURE SURVEY

2.1 REVIEW IN LITERATURE WORK

Xiang Xiang the recent developments of Convolutional Neural Networks, the performance of face verification methods has increased rapidly. In a typical face verification method, feature normalization is a critical step for boosting performance. This motivates us to introduce and study the effect of normalization during training. But we find this is non-trivial, despite normalization being differentiable. We identify and study four issues related to normalization through mathematical analysis, which yields understanding and helps with parameter settings. Based on this analysis we propose two strategies for training using normalized features. It is a modification of softmax loss, which optimizes cosine similarity instead of inner-product. The second is a reformulation of metric learning by introducing an agent vector for each class. We show that both strategies, and small variants, consistently improve performance by between 0.2% to 0.4% on the LFW dataset based on two models. This is significant because the performance of the two models on LFW dataset is close to saturation at over 98%.

Hao Wang, Yitong Wang proposed Face recognition has made extraordinary progress owing to the advancement of deep convolutional neural networks (CNNs). The central task of face recognition, including face verification and identification, involves face feature discrimination. However, the traditional softmax loss of deep CNNs usually lacks the power of discrimination. To address this problem, recently several loss functions such as center loss, large margin softmax loss, and angular softmax loss have been proposed. All these improved losses share the same idea: maximizing inter-class variance and minimizing intra-class variance. In this paper, we propose a novel loss function, namely large margin cosine loss (LMCL), to realize this idea from a different perspective. More specifically, we reformulate the softmax loss as a cosine loss by L2

normalizing both features and weight vectors to remove radial variations, based on which a cosine margin term is introduced to further maximize the decision margin in the angular space. As a result, minimum intra-class variance and maximum inter-class variance are achieved by virtue of normalization and cosine decision margin maximization. We refer to our model trained with LMCL as CosFace. Extensive experimental evaluations are conducted on the most popular public-domain face recognition datasets such as MegaFace Challenge, Youtube Faces (YTF) and Labeled Face in the Wild (LFW). We achieve the state-of-the-art performance on these benchmarks, which confirms the effectiveness of our proposed approach.

Yi Sun, Yuheng Chen The key challenge of face recognition is to develop effective feature representations for reducing intra-personal variations while enlarging inter-personal differences. In this paper, we show that it can be well solved with deep learning and using both face identification and verification signals as supervision. The Deep IDentification-verification features (DeepID2) are learned with carefully designed deep convolutional networks. The face identification task increases the inter-personal variations by drawing DeepID2 features extracted from different identities apart, while the face verification task reduces the intra-personal variations by pulling DeepID2 features extracted from the same identity together, both of which are essential to face recognition. The learned DeepID2 features can be well generalized to new identities unseen in the training data. On the challenging LFW dataset, 99.15% face verification accuracy is achieved. Compared with the best previous deep learning result on LFW, the error rate has been significantly reduced by 67%.

2.1 EXISTING SYSTEM

In the existing system, Identity verification plays an important role in our daily lives. For example, access control, physical security and international border crossing require us to verify our access (security) level and our identities. to verify who we are by showing our ID documents containing face images, such as passports and driver licenses, to human operators. However, this process is slow, labor intensive and unreliable. As such, an automated system for matching ID document photos to live face images (selfies) in real time and with high accuracy is required. After verifying a traveler's identity by face comparison, the gate is automatically opened for the traveler to enter. For IDselfie matching, they are comparing a scanned or digital document photo.

2.2 PROPOSED SYSTEM

We are proposing a certificate system based on blockchain to overcome the problem. Data are stored in different nodes, and anyone who wishes to modify a particular internal datum must request that other nodes modify it simultaneously. Thus, the system is highly reliable. We developed a decentralized application and designed a certificate system based on Ethereum blockchain. This technology was selected because it is incorruptible, encrypted, and trackable and permits data synchronization. By integrating the features of blockchain, the system improves the efficiency operations at each stage. The system saves on paper, cuts management costs, prevents document frgery, and provides accurate and reliable information on digital certificates and compare user live face with verified document face.

CHAPTER 3

METHODOLOGY

3.1 REQUIREMENT SPECIFICATION

3.1.1 HARDWARE SPECIFICATION

- Hard Disk : 80GB and Above
- RAM : 4GB and Above
- Processor : P IV and Above

3.1.2 SOFTWARE SPECIFICATION

- Windows 7 and above
- JDK 1.7
- J2EE
- Tomcat 7.0
- MySQL

3.1.3 TECHNOLOGIES USED

- J2EE (JSP, Servlet), JavaScript, HTML, CSS, AJAX.
- Hibernate Framework
- MVC Pattern
- Design Pattern

3.1.4 ALOGORITHUM USED

- KNN
- RSA
- SHA-256

3.2 ADVANTAGES AND DISADVANTAGES

3.2.1 ADVANTAGES

- modification of softmax loss, which optimizes cosine similarity instead of inner-product. Avoid the need of hard sample mining. Improve performance over models without normalization.
- minimum intra-class variance and maximum inter-class variance are achieved by virtue of normalization and cosine decision margin maximization. Our approach consistently achieves the state-of-the-art results on several face benchmarks.
- optimize the triplet loss on a different space of triplets. The proxy-loss improves on state-of-art results for three standard zero-shot learning datasets, by up to 15% points.

3.2.2 DISADVANTAGES

- The problem of ID-selfie matching poses numerous challenges that are different from general face recognition. For typical unconstrained face recognition tasks, the main challenges are due to pose, illumination and expression (PIE) variations.
- The low quality of document photos due to image compression¹ and (2) the large time gap between the document issue date and the verification date remain as the primary difficulties.

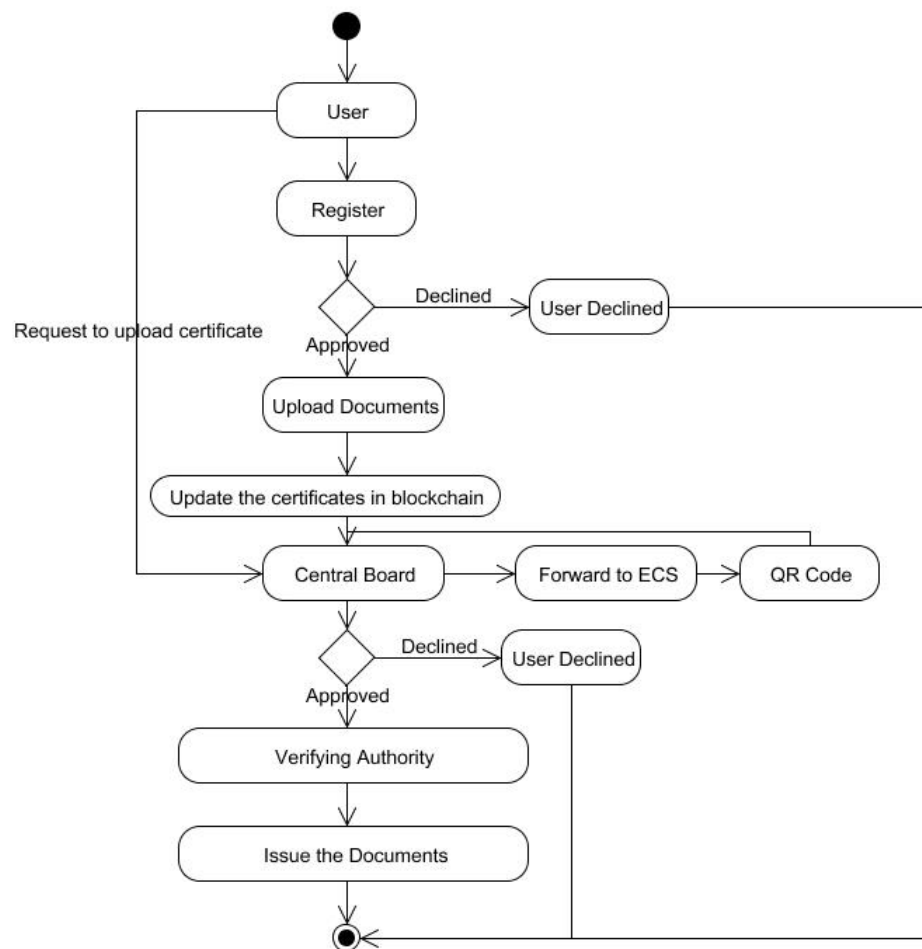
3.3 SYSTEM DESIGN

3.3.1 ACTIVITY DIAGRAM

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

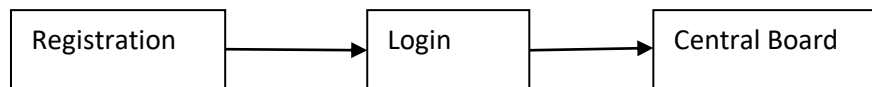
- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.



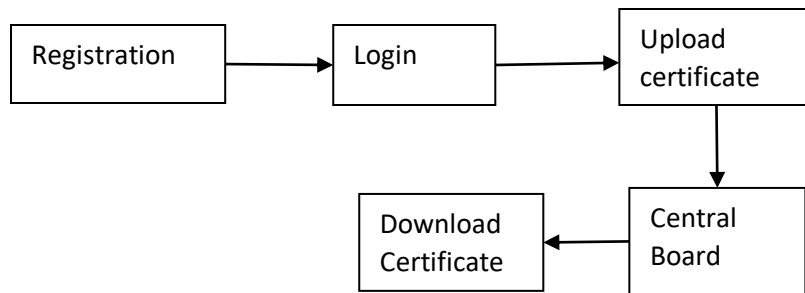
3.3.2 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview system which can later be elaborated DFDs can also be used for visualization of data processing.

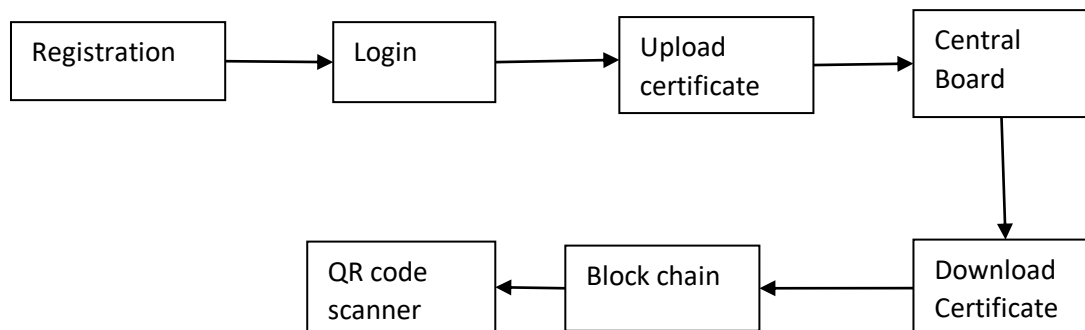
LEVEL 0



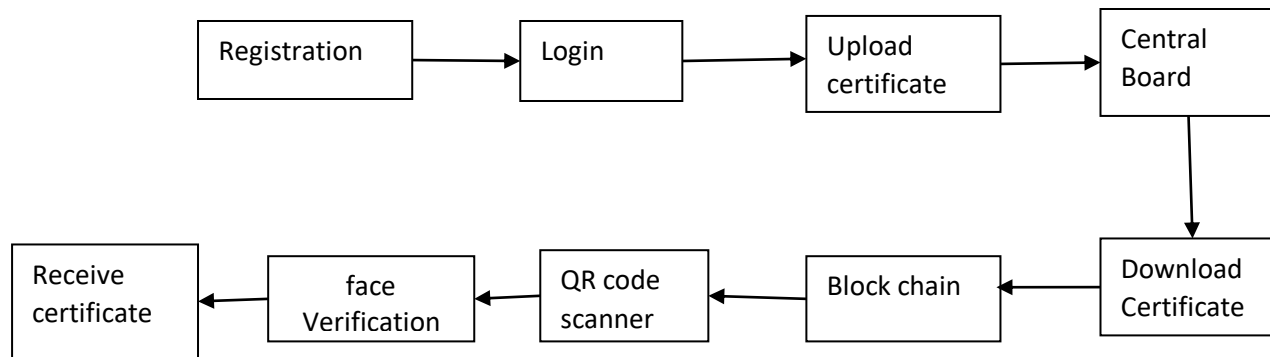
LEVEL 1



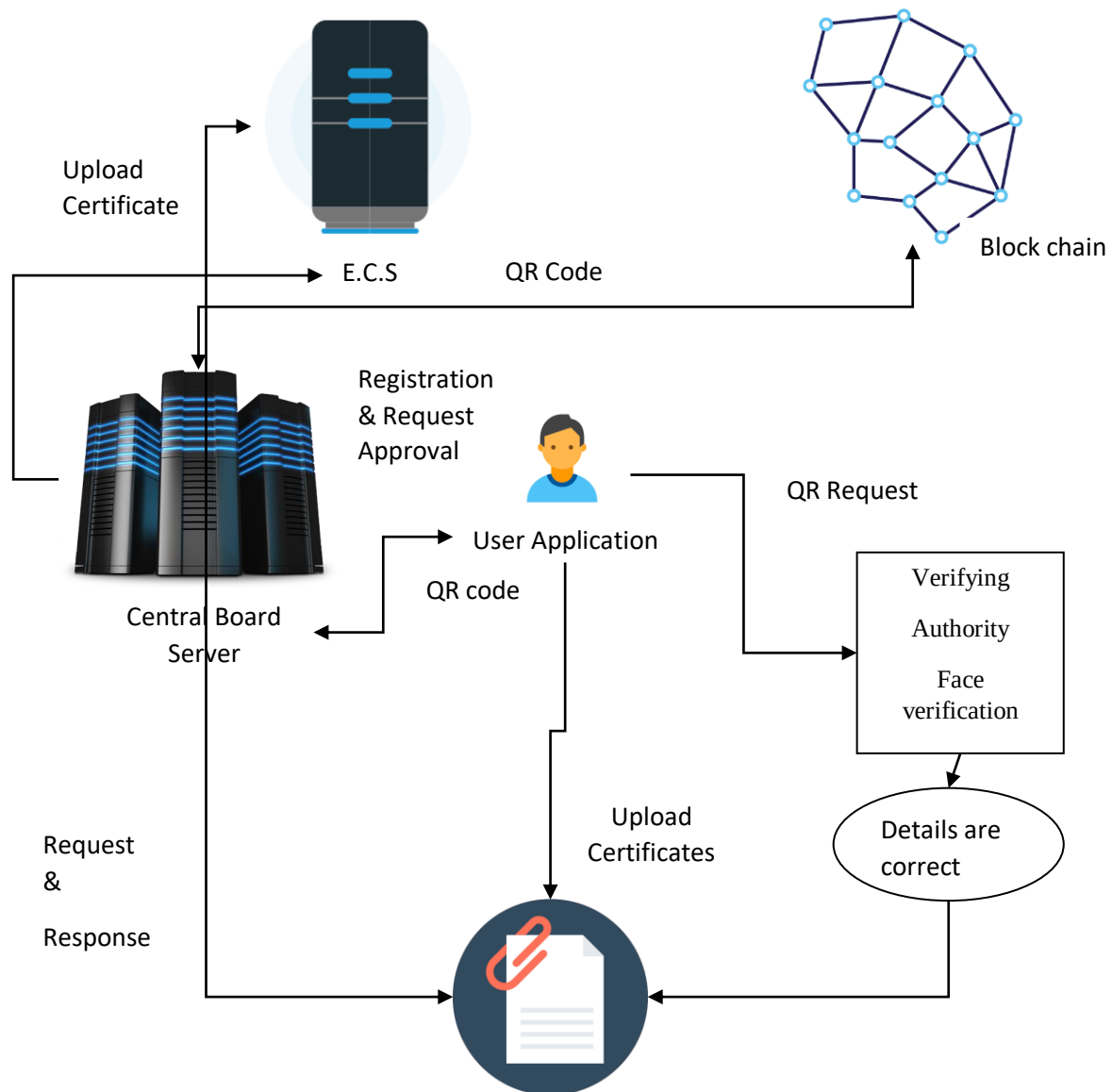
LEVEL 2



LEVEL 3



3.3.3 ARCHITECTURE DIAGRAM



3.4 SYSTEM ALOGORITHM

3.4.1.2 KNN algorithm

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

- 1. Ease to interpret output
- 2. Calculation time
- 3. Predictive Power

Let us take a few examples to place KNN in the scale:

	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.

Pseudo Code of KNN

We can implement a KNN model by following the below steps:

1. Load the data
2. Initialize the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
 1. Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
 2. Sort the calculated distances in ascending order based on distance values
 3. Get top k rows from the sorted array
 4. Get the most frequent class of these rows
 5. Return the predicted class

3.5 MODULES

- **User Registration And Authentication**
- **User Upload Certificate**
- **Get Certificate**
- **QR Request And Response From Verification Authority**

3.5.1 MODULES EXPLANATION

User Registration And Authentication

In this module user needs to registers into his application and a request will be sent to central board server for authentication. Unless the central board server approves the request user

cannot login into his account. When central board server approves the request a key will be generated and user can login into his account.

User Upload Certificate:

After user login into his account he needs to upload certificates namely pan card, aadhar card, voter id, ssc certificates to central board server. Central board server will review the certificates and accepts or decline the certificates. If central board server accepts the certificate those details will be stored in E.C.S and Blockchain. If central board server declines the certificate it won't be stored in E.C.S. or Block Chain.

Get Certificate:

If user needs a certificate he will send request to central board server. If central board server found the user details to be genuine he accepts the request and forward a request to E.C.S where all the certificates will be there. E.C.S. responds for the request and certificates will be provided to the user.

QR Request and Face verification:

If user wants to apply for any certificates he will send request to central board server and central board server will check the details and forward the request to E.C.S. E.C.S will generate the QR Code and forwarded to user via central board server. User forwards the QR code to the verifying authority and if all details are correct and face matches with live face Verifying authority will issue the document.

Chapter 4

System Features

A new optimization method for classification-based embedding learning on shallow datasets³. A new recognition system containing a pair of partially shared networks for learning unified representations from IDselfie pairs. An evaluation of COTS and public-domain face matchers showing ID-selfie matching is a non-trivial problem with different challenges from general face matching.

4.1 Design and Implementation Constraints

4.1.1 Constraints in Analysis

- ◆ Constraints as Informal Text
- ◆ Constraints as Operational Restrictions
- ◆ Constraints Integrated in Existing Model Concepts
- ◆ Constraints as a Separate Concept
- ◆ Constraints Implied by the Model Structure

4.1.2 Constraints in Design

- ◆ Determination of the Involved Classes
- ◆ Determination of the Involved Objects
- ◆ Determination of the Involved Actions
- ◆ Determination of the Require Clauses

- ◆ Global actions and Constraint Realization

4.1.3 Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object oriented methods.

CHAPTER 5

RESULT AND DISCUSSION

5.1 RESULT

A. ID Document Photo Matching

To the best of our knowledge, the first study on IDselfie matching is attributed to Starovoitov et al. Assuming all face images are frontal faces without large expression variations, the authors first localize the eyes with Hough Transform. Based on eye locations, the face region is cropped and gradient maps are computed as feature maps. The algorithm is similar to a general constrained face matcher, except it is developed for a document photo dataset. Bourlai et al. considered ID-selfie matching as a comparison between degraded face images, i.e., scanned document photos, and high quality live face images.

B. Deep Face Recognition

With the success of deep neural networks in the ImageNet competition [16], virtually all of the ongoing research in face recognition now utilizes deep neural networks to learn face representation. Taigman et al. first proposed the application of deep neural networks to learn face representation. They designed an 8-layer convolutional neural network and trained it with a Softmax loss function, a standard loss function for classification problems. They used the outputs of the bottleneck layer as face representation and achieved state-of-the-art performance at that time in 2014. Considering that Softmax loss only encourages large inter-class variations but does not constrain intra-class variations, Sun et al. later proposed to train networks with both a classification signal (Softmax loss) and a metric learning signal (Contrastive loss). About the same time, Schroff et al. [19] proposed a metric learning loss function, named

triplet loss, boosting the state-of-the-art performance on the standard LFW protocol . Liu et al. first bridged the gap between classification loss functions and metric learning methods with the Angular-Softmax (A-Softmax) loss function, a modified Softmax loss that classifies samples based on angular distances. Wang et al. recently proposed the Additive Margin Softmax (AM-Softmax), which learns to increase the angular discriminability with an additive margin and is shown to be more robust than Softmax.

C.Heterogeneous Face

Recognition Heterogeneous face recognition (HFR) is an emerging topic that has become popular in the past few years . It usually refers to face recognition between two different modalities, including visible spectrum images (VIS), near infrared images (NIR) , thermal infrared images , composite sketches , etc. ID-selfie matching can be considered to be a special case of HFR since the images to be matched come from two different sources. Therefore, the techniques used in HFR could be potentially helpful for the ID-selfie problem. Most methods for HFR can be categorized into two types: synthesis-based methods and discriminant feature based methods. Synthesis-based methods aim to transform the images from one modality into another so that general intra modality face recognition systems can be applied .

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In this paper, we propose a new face recognition system, named DocFace+, for matching ID document photos to selfies. The transfer learning technique is used, where a base model for unconstrained face recognition is fine-tuned on a private ID-selfie dataset. A pair of sibling networks with shared high-level modules are used to model domain-specific parameters. Based on our observation of the weight-shift problem of classification-based embedding learning loss functions on shallow datasets, we propose an alternative optimization method, called dynamic weight imprinting (DWI) and a variant of AM Softmax, DIAM-Softmax. Experiments show that the proposed method not only helps the loss converge much faster but also leads to better generalization performance. A comparison with static weight imprinting methods confirms that DWI is capable of capturing the global distribution of embeddings accurately. Different sampling methods are studied for mini-batch construction and we find that a balanced sampling between the two domains is most helpful for learning generalizable features. We compare the proposed system with existing general face recognition systems on our private dataset and see a significant improvement with our system, indicating the necessity of domain-specific modeling of ID-selfie data. Finally, we compare the performance of different matchers on the Public-IvS dataset and find that although this dataset is similar to a general face dataset, our system still generalizes well.

6.2 FUTURE WORK

This application has more advantages future work of this project to increases the high quality of dataset. The classification-based loss functions are able to capture global information and thus achieve state-of-the art performances on general face recognition problems while the metric learning loss functions are shown to be effective on very large datasets .

REFERENCES

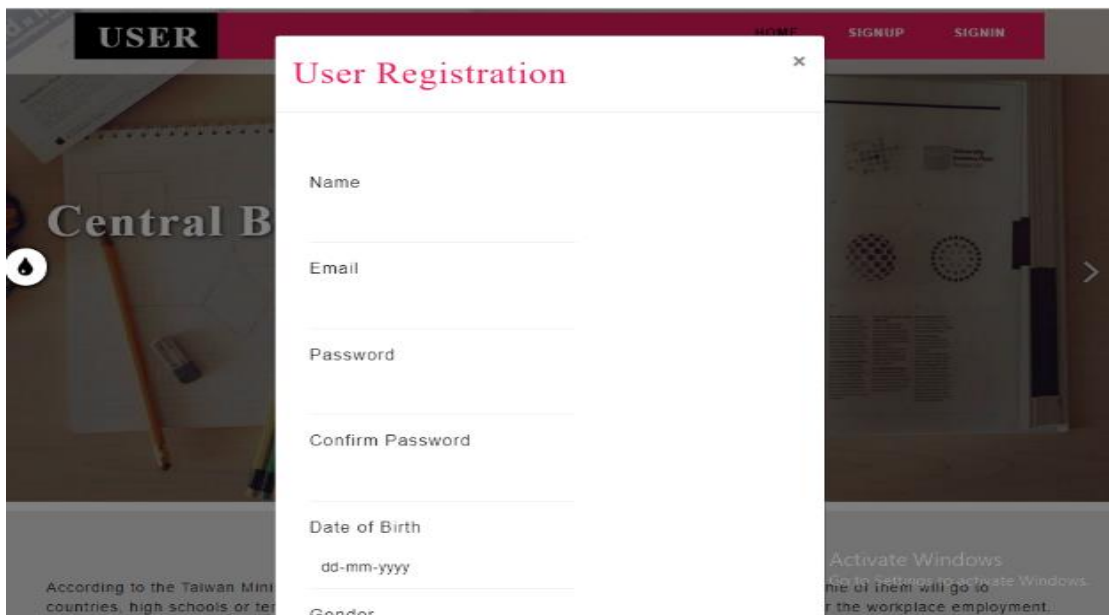
- [1] D. White, R. I. Kemp, R. Jenkins, M. Matheson, and A. M. Burton, “Passport officers’ errors in face matching,” *PLoS ONE*, vol. 9, no. 8, 2014, Art. no. e103510.
- [2] Wikipedia. (2018). *Australia Smartgate*. [Online]. Available: <https://en.wikipedia.org/wiki/SmartGate>
- [3] Wikipedia. (2018). *Epassport Gates*. [Online]. Available: https://en.wikipedia.org/wiki/EPassport_gates
- [4] U.S. Customs and Border Protection. (2018). *Automated Passport Control (APC)*. [Online]. Available: <https://www.cbp.gov/travel/uscitizens/apc>
- [5] Xinjiang Heng An Perimeter Security Equipment Company. (2018). *What Is ID-Person Matching?* [Online]. Available: http://www.xjhazj.com/xjhazj/vip_doc/8380983.html
- [6] Jumio. (2018). *Netverify ID Verification*. [Online]. Available: <https://www.jumio.com/trusted-identity/netverify>
- [7] Mitek. (2018). *Mitek ID Verification*. [Online]. Available: <https://www.miteksystems.com/mobile-verify>
- [8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled faces in the wild: A database for studying face recognition in unconstrained environments,” Univ. Massachusetts, Amherst, MA, USA, Rep. 07-49, Oct. 2007 V. Starovoitov, D. Samal, and B. Sankur, “Matching of faces in camera images and document photographs,” in *Proc. ICASSP*, 2000, pp.

- 2349–2352. [10] T. Bourlai, A. Ross, and A. Jain, “On matching digital face images against scanned passport photos,” in *Proc. IEEE Int. Conf. Biometrics Identity Security (BIDS)*, 2009, pp. 1–10.
- [11] T. Bourlai, A. Ross, and A. K. Jain, “Restoring degraded face images: A case study in matching faxed, printed, and scanned photos,” *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 2, pp. 371–384, Jun. 2011.
- [12] V. V. Starovoitov, D. I. Samal, and D. V. Briliuk, “Three approaches for face recognition,” in *Proc. Int. Conf. Pattern Recognit. Image Anal.*, 2002, pp. 707–711.
- [13] Y. Shi and A. K. Jain, “DocFace: Matching ID document photos to selfies,” in *Proc. BTAS*, 2018, pp. 1–8.
- [14] X. Zhu *et al.*, “Large-scale bisample learning on ID vs. spot face recognition,” *arXiv:1806.03018*, 2018.
- [15] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, “VGGFace2: A dataset for recognising faces across pose and age,” in *Proc. IEEE Conf. Autom. Face Gesture Recognit.*, 2018, pp. 1–8.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012, pp. 1–9.
- [17] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the gap to human-level performance in face verification,” in *Proc. CVPR*, 2014, pp. 1701–1708.
- [18] Y. Sun, Y. Chen, X. Wang, and X. Tang, “Deep learning face representation by joint identification-verification,” in *Proc. NIPS*, 2014, pp. 1988–1996.
- [19] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in *Proc. CVPR*, 2015, pp. 815–823.
- [20] I. Masi, A. T. Tr`ân, T. Hassner, J. T. Leksut, and G. Medioni, “Do we really need to collect millions of faces for effective face recognition?” in *Proc. ECCV*, 2016, pp. 212–220.
- [21] W. Liu *et al.*, “SphereFace: Deep hypersphere embedding for face recognition,” in *Proc. CVPR*, 2017, pp. 212–220.
- [22] A. Hasnat, J. Bohné, J. Milgram, S. Gentric, and L. Chen, “DeepVisage: Making face recognition simple yet with powerful generalization skills,” in *Proc. ICCV Workshops*, 2017, pp. 1682–1691.

- [23] F. Wang, J. Cheng, W. Liu, and H. Liu, “Additive margin softmax for face verification,” *IEEE Signal Process. Lett.*, vol. 25, no. 7, pp. 926–930, Jul. 2018.
- [24] W. Liu, Y. Wen, Z. Yu, and M. Yang, “Large-margin softmax loss for convolutional neural networks,” in *Proc. ICML*, 2016, pp. 507–516.
- [25] W. Liu *et al.*, “Deep hyperspherical learning,” in *Proc. NIPS*, 2017, pp. 1–11.
- [26] S. Z. Li and A. Jain, *Encyclopedia of Biometrics*. New York, NY, USA: Springer, 2015.
- [27] S. Z. Li, R. Chu, S. Liao, and L. Zhang, “Illumination invariant face recognition using near-infrared images,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 627–639, Apr. 2007.
- [28] J. Choi, S. Hu, S. S. Young, and L. S. Davis, “Thermal to visible face recognition,” in *Proc. SPIE DSS-DS107 Biometric Technol. Human Identification IX*, 2012, pp. 1–10.
- [29] X. Tang and X. Wang, “Face photo recognition using sketch,” in *Proc. ICIP*, 2002, p. 1.
- [30] X. Wang and X. Tang, “Face photo-sketch synthesis and recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 1955–1967, Nov. 2009.

APPENDIX

A. SCREEN SHOT



The screenshot shows a web application interface. At the top, there is a dark red navigation bar with the word "USER" in white. To the right of "USER" are links for "HOME", "SIGNUP", and "SIGNIN". A white modal window titled "User Registration" is centered on the screen. It contains several input fields: "Name", "Email", "Password", "Confirm Password", "Date of Birth" (with a placeholder "dd-mm-yyyy"), and "Gender". The background of the page is a dark image of a desk with a pencil and a piece of paper that says "Central B". At the bottom of the page, there is a Windows activation watermark that reads "Activate Windows. Go to Settings to activate Windows. Some of them will go to the workplace employment."

Fig 6.1 login page

User needs to registers into his application and a request will be sent to central board server for authentication. Unless the central board server approves the request user cannot login into his account. When central board server approves the request a key will be generated and user can login into his account.

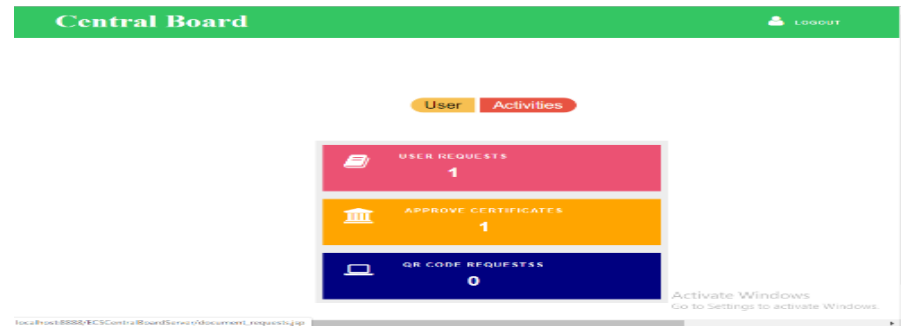


Fig 6.2 request page

. When central board server approves the request a key will be generated and user can login into his account.

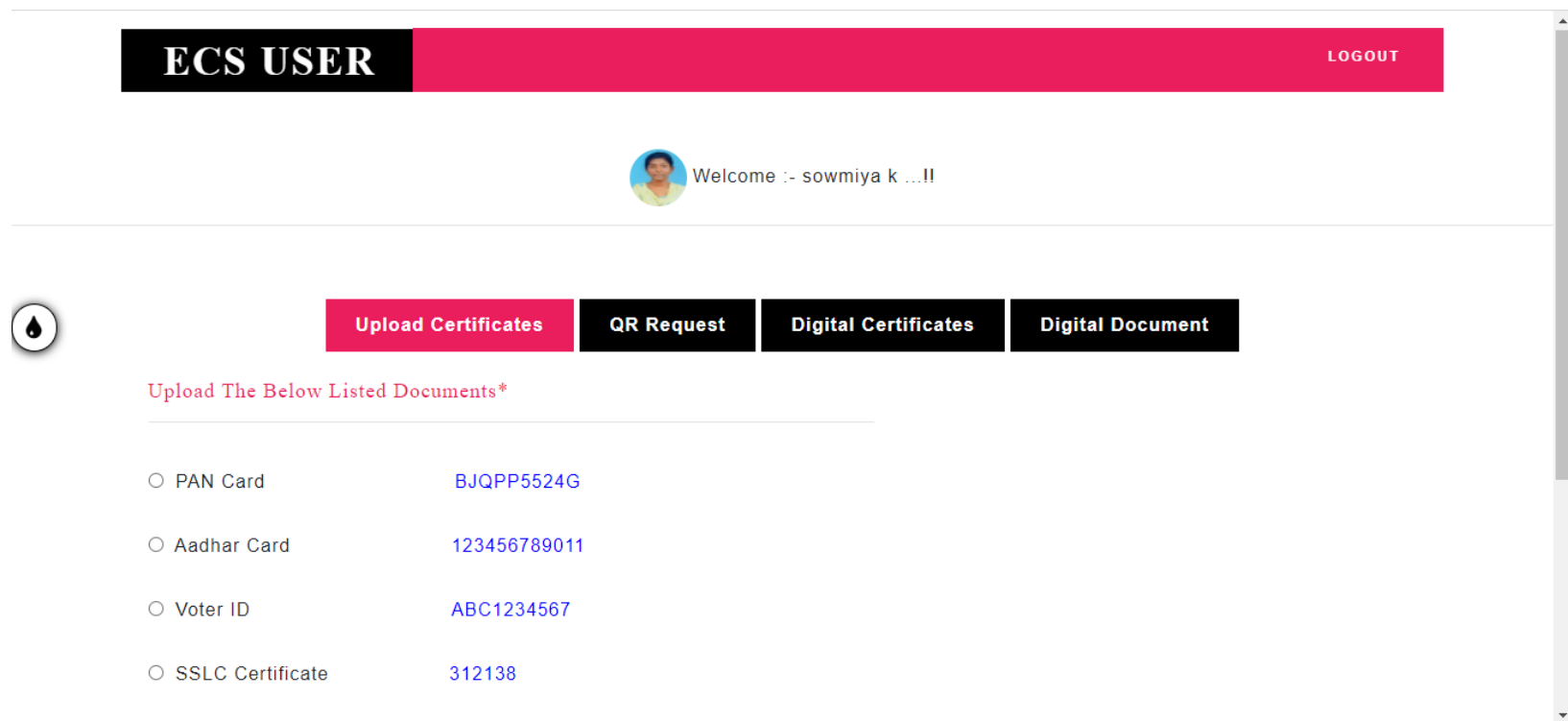


Fig 6.3

. After user login into his account he needs to upload certificates namely pan card, aadhar card, voter id, ssc certificates to central board server. Central board server will review the certificates and accepts or decline the certificates. If central board server accepts the certificate those details will be stored in E.C.S and Blockchain. If central board server declines the certificate it won't be stored in E.C.S. or Block Chain. If user needs a certificate he will send request to central board server.

The screenshot displays the 'Central Board' web application interface. At the top, a green header bar contains the text 'Central Board' on the left, and 'HOME' and 'LOGOUT' (with a user icon) on the right. Below the header, there are two tabs: 'Documents' (highlighted in orange) and 'Verification' (highlighted in red). The main content area is titled 'Document Details' and lists four documents for verification:

Document Type	Document ID	Email
PAN CARD	CKJPB8531J	gts@gmail.com
AADHAR CARD	29053495978690	
VOTER ID	VTR7844611	
SSLC CERTIFICATE	145121	

At the bottom of the document list, there are two buttons: 'Accept' (green) and 'Decline' (purple). A watermark 'Activate Windows Go to Settings to activate Windows.' is visible in the bottom right corner of the interface.

Fig 6.4

. If central board server found the user details to be genuine he accepts the request and forward a request to E.C.S where all the certificates will be there. E.C.S. responds for the request and certificates will be provided to the user.

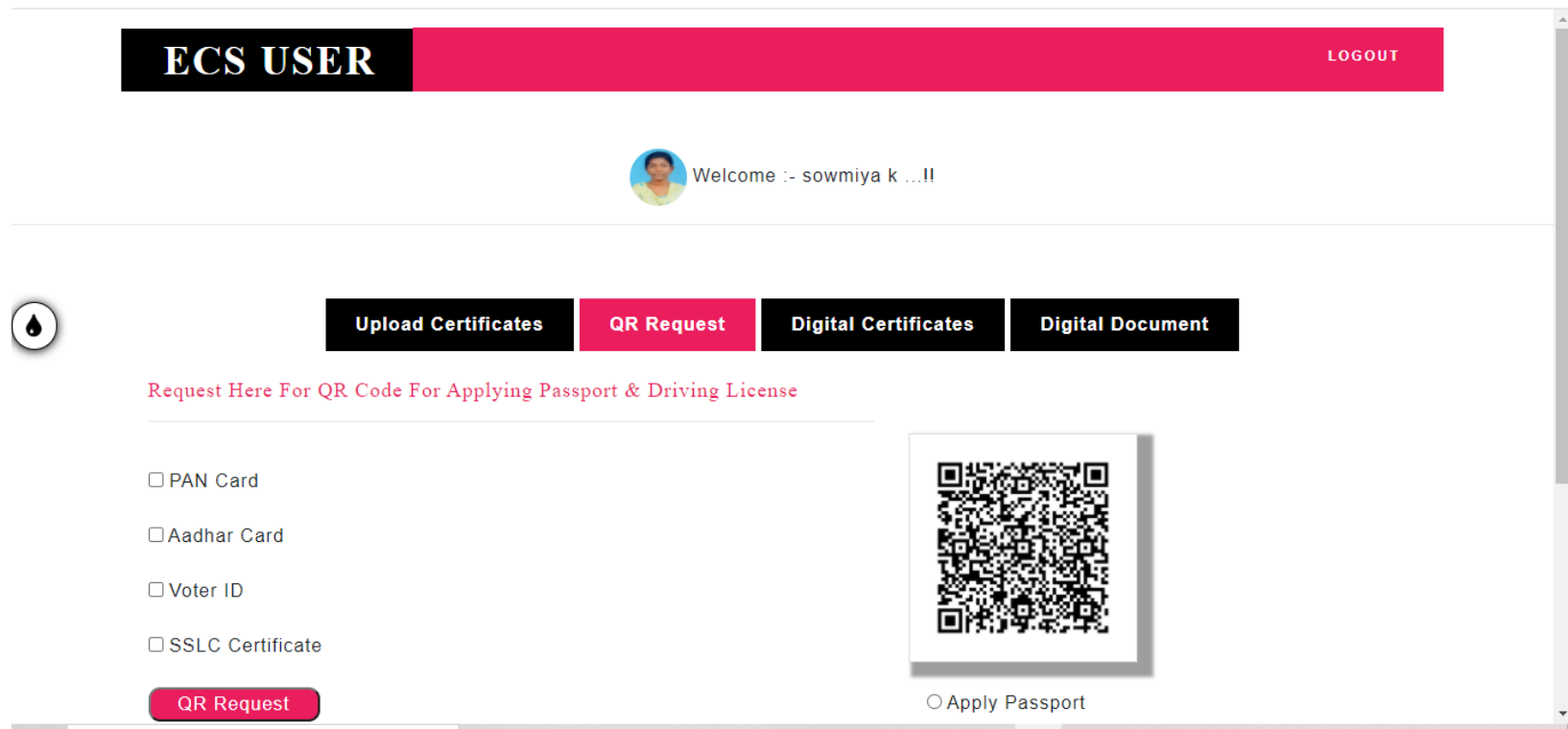


Fig 6.5

. If user wants to apply for any certificates he will send request to central board server and central board server will check the details and forward the request to E.C.S. E.C.S will generate the QR Code and forwarded to user via central board server. User forwards the QR code to the verifying authority and if all details are correct and face matches with live face Verifying authority will issue the document.



Fig 6.6 Comparing the the live face and id document face

B.SOURCE CODE

Login

```
import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import service.UserDao;

import com.http.servlet.HttpServlet;

public class Login extends HttpServlet {

    public void doPost(HttpServletRequest request, HttpServletResponse response)

        throws ServletException, IOException {
```

```

response.setContentType("text/html");

PrintWriter out = response.getWriter();

//-----

 UserDao userDao = new UserDao();

String email = request.getParameter("email");

String password = request.getParameter("password");

String result = userDao.userLogin(email, password);

if (!result.equalsIgnoreCase("no")) {

    System.out.println("----->>>>"+result);

    String temp[] = result.split(",");

    if (temp[1].equalsIgnoreCase("pending")) {

        request.setAttribute("msglog", "Your request is in pending...");

        RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

        rd.forward(request, response);

    }else if(temp[1].equalsIgnoreCase("declined")){

        request.setAttribute("msglog", "Your request is rejected");

        RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

```

```
        rd.forward(request, response);
    } else {

        HttpSession session=request.getSession(true);

        session.setAttribute("image", temp[2]);

        session.setAttribute("email", email);

        session.setAttribute("name", temp[0]);

        RequestDispatcher rd = request.getRequestDispatcher("menu.jsp");

        rd.forward(request, response);

    }

    } else {

        request.setAttribute("msglog", "Invalid email or password");

        RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

        rd.forward(request, response);

    }

    out.close();}

}
```

Userdao

```
import java.util.List;

import hibernate.util.HibernateUtil;

import org.hibernate.Criteria;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.criterion.Restrictions;

import model.CertificatesPojo;

import model.QrPojo;

import model.UserPojo;


public class UserDao implements MethodInterface {

    @Override

    public boolean saveUser(UserPojo userPojo) {

        boolean result = false;

        Session session = HibernateUtil.openSession();

        Transaction tx = null;
```

```
try {  
    tx = session.beginTransaction();  
    tx.begin();  
    Criteria cr = session.createCriteria(UserPojo.class);  
    cr.add(Restrictions.eq("email", userPojo.getEmail()));  
    List<UserPojo> list = cr.list();  
    if (list.isEmpty()) {  
        session.save(userPojo);  
        result = true;  
    }  
    tx.commit();  
  
} catch (Exception e) {  
    if (tx != null) {  
        tx.rollback();  
    }  
    e.printStackTrace();  
}
```

```
    } finally {  
        session.close();  
    }  
    return result;  
}
```

@Override

```
public boolean setProfile(String email, String image) {  
    boolean result = false;  
    Session session = HibernateUtil.openSession();  
    Transaction tx = null;  
    try {  
        tx = session.beginTransaction();  
        tx.begin();  
        Criteria cr = session.createCriteria(UserPojo.class);  
        cr.add(Restrictions.eq("email", email));  
        List<UserPojo> list = cr.list();
```

```
        if (!list.isEmpty()) {  
            UserPojo userPojo = list.get(0);  
            userPojo.setImage(image);  
            session.update(userPojo);  
            result = true;  
        }  
        tx.commit();  
  
    } catch (Exception e) {  
        if (tx != null) {  
            tx.rollback();  
        }  
        e.printStackTrace();  
    } finally {  
        session.close();  
    }  
  
    return result;  
}
```

```
}
```

```
@Override
```

```
public String userLogin(String email, String password) {
```

```
    String result = "no";
```

```
    Session session = HibernateUtil.openSession();
```

```
    Transaction tx = null;
```

```
    try {
```

```
        tx = session.beginTransaction();
```

```
        tx.begin();
```

```
        Criteria cr = session.createCriteria(UserPojo.class);
```

```
        cr.add(Restrictions.eq("email", email));
```

```
        cr.add(Restrictions.eq("password", password));
```

```
        List<UserPojo> list = cr.list();
```

```
        if (!list.isEmpty()) {
```

```
            for (UserPojo userPojo : list) {
```

```
                result = userPojo.getName() + "," + userPojo.getUid() + ","
```

```

        + userPojo.getImage();
    }
}

tx.commit();

} catch (Exception e) {
    if (tx != null) {
        tx.rollback();
    }

    e.printStackTrace();
} finally {
    session.close();
}

return result;
}

```

@Override

```
public boolean saveCertificates(CertificatesPojo certificatesPojo) {  
  
    boolean result = false;  
  
    Session session = HibernateUtil.openSession();  
  
    Transaction tx = null;  
  
    try {  
  
        tx = session.beginTransaction();  
  
        tx.begin();  
  
        session.save(certificatesPojo);  
  
        result = true;  
  
        tx.commit();  
  
  
    } catch (Exception e) {  
  
        if (tx != null) {  
  
            tx.rollback();  
  
        }  
  
        e.printStackTrace();  
  
    } finally {
```

```
        session.close();  
    }  
  
    return result;  
}
```

@Override

```
public List<CertificatesPojo> loadCertificateStatus(String email) {  
  
    List<CertificatesPojo> list = null;  
  
    Session session = HibernateUtil.openSession();  
  
    Transaction tx = null;  
  
    try {  
  
        tx = session.beginTransaction();  
  
        tx.begin();  
  
        Criteria cr = session.createCriteria(CertificatesPojo.class);  
  
        cr.add(Restrictions.eq("email", email));  
  
        list = cr.list();  
  
        tx.commit();  
    }  
}
```

```
    } catch (Exception e) {  
        if (tx != null) {  
            tx.rollback();  
        }  
        e.printStackTrace();  
    } finally {  
        session.close();  
    }  
    return list;  
}
```

@Override

```
public boolean updatePanCard(String email, String panCard,  
    String panStatus, String type) {  
    boolean result = false;  
    Session session = HibernateUtil.openSession();
```

```
Transaction tx = null;

try {

    tx = session.beginTransaction();

    tx.begin();

    Criteria cr = session.createCriteria(CertificatesPojo.class);

    cr.add(Restrictions.eq("email", email));

    List<CertificatesPojo> list = cr.list();

    if (!list.isEmpty()) {

        if (type.equals("pan card")) {

            CertificatesPojo certificatesPojo = list.get(0);

            certificatesPojo.setPanCard(panCard);

            certificatesPojo.setPanStatus(panStatus);

            certificatesPojo.setStatus("pending");

            session.update(certificatesPojo);

            result = true;

        } else if (type.equals("aadhar card")) {

            CertificatesPojo certificatesPojo = list.get(0);
```

```
        certificatesPojo.setAadharCard(panCard);

        certificatesPojo.setAadharStatus(panStatus);

        certificatesPojo.setStatus("pending");

        session.update(certificatesPojo);

        result = true;
    } else if (type.equals("voter id")) {

        CertificatesPojo certificatesPojo = list.get(0);

        certificatesPojo.setVoterId(panCard);

        certificatesPojo.setVoterIDStatus(panStatus);

        certificatesPojo.setStatus("pending");

        session.update(certificatesPojo);

        result = true;
    } else if (type.equals("sslc")) {

        CertificatesPojo certificatesPojo = list.get(0);

        certificatesPojo.setSslc(panCard);

        certificatesPojo.setSslcStatus(panStatus);

        certificatesPojo.setStatus("pending");
```

```
        session.update(certificatesPojo);

        result = true;
    }

}

tx.commit();
} catch (Exception e) {
    if (tx != null) {
        tx.rollback();
    }

    e.printStackTrace();
} finally {
    session.close();
}

return result;
}
```

@Override

```
public boolean saveQR(QrPojo qrPojo) {  
  
    boolean result = false;  
  
    Session session = HibernateUtil.openSession();  
  
    Transaction tx = null;  
  
    try {  
  
        tx = session.beginTransaction();  
  
        tx.begin();  
  
        Criteria cr = session.createCriteria(QrPojo.class);  
  
        cr.add(Restrictions.eq("email", qrPojo.getEmail()));  
  
        cr.add(Restrictions.eq("certificates", qrPojo.getCertificates()));  
  
        List<QrPojo> list = cr.list();  
  
        if (list.isEmpty()) {  
  
            session.save(qrPojo);  
  
            result = true;  
  
        }  
  
        tx.commit();  
  
    }  
  
}
```

```
    } catch (Exception e) {  
        if (tx != null) {  
            tx.rollback();  
        }  
        e.printStackTrace();  
    } finally {  
        session.close();  
    }  
    return result;  
}
```

@Override

```
public List<QrPojo> loadQr() {  
    List<QrPojo> list = null;  
    Session session = HibernateUtil.openSession();  
    Transaction tx = null;  
    try {
```

```
        tx = session.beginTransaction();

        tx.begin();

        Criteria cr = session.createCriteria(QrPojo.class);

        list = cr.list();

        tx.commit();

    } catch (Exception e) {

        if (tx != null) {

            tx.rollback();

        }

        e.printStackTrace();

    } finally {

        session.close();

    }

    return list;

}
```

@Override

```
public List<QrPojo> loadQr(String email) {  
  
    List<QrPojo> list = null;  
  
    Session session = HibernateUtil.openSession();  
  
    Transaction tx = null;  
  
    try {  
  
        tx = session.beginTransaction();  
  
        tx.begin();  
  
        Criteria cr = session.createCriteria(QrPojo.class);  
  
        cr.add(Restrictions.eq("email", email));  
  
        list = cr.list();  
  
        tx.commit();  
  
  
    } catch (Exception e) {  
  
        if (tx != null) {  
  
            tx.rollback();  
  
        }  
    }  
}
```

```
        e.printStackTrace();  
    } finally {  
        session.close();  
    }  
    return list;  
}
```