Facial Recognition Through Siamese CNNs

Leenane Tinashe Makurumure
University Of KwaZulu-Natal
217076701@stu.ukzn.ac.za

31 July 2020

Abstract

Facial images offer a reliable means of physical biometric identification. However, image modularities, geometric distortions, partial deformations, and obstructions could easily lead to false positives or false negatives. Due to these shortcomings, facial recognition is typically not used as a form of access control or as an authentication method, as it could lead to an imposter gaining unauthorized access or a genuine user being denied access to some resource. In this paper, we propose an improvement in facial recognition accuracy through a patch-based convolutional neural network approach for region-specific feature extraction with a Siamese architecture. Patch-based feature extraction through CNNs could enable our deep learning model to be less dependent on complete information, but maximize on the available discriminative properties and robust to geometric distortions. This could lead to more accurate identification.

1 Introduction and Background

To date, facial recognition has been successfully incorporated into simple social media applications as a means of object identification. However, it is not reliable enough for biometric identification or authentication for access control into a system or to some resource. Facial recognition could offer a relatively cheap solution to biometric authentication as illustrated in Figure 1. However, it is not very reliable, and this has kept it at the forefront of research in the past decade. There have been vast improvements due to the advancement of deep learning and faster processing of enormous data. Yet, there is still more to be done.

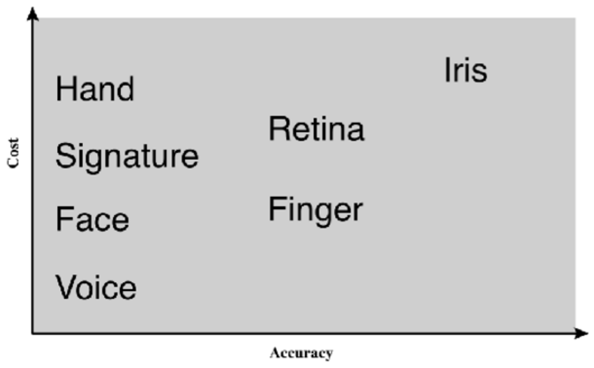


Figure 1: Cost versus accuracy of various biometric characteristics in user authentication schemes. Adapted from [14].

Ken Bodnar, an AI researcher, is quoted in an article [15] saying that AI face recognition technology is very good but not very robust. This means that the technology could misidentify someone as a genuine client (False acceptance) or an imposter (False rejection) thereby causing a breach into the system or inconveniencing legitimate clients, respectively. A study from MIT shows that facial recognition tools had significant problems when identifying people of colour [15]. Bias and potential invasion of privacy have kept facial recognition as an essential research topic.

Facial images are typically required in two different electronic tasks: verification and identification, as shown in Figure 2. Verification (authentication) performs one-to-one matching while identification is a one-to-many matching problem. In both cases, the underlying objective is the matching of a test image (known or unknown) to another image to determine if both images belong to the same person or are from different people.

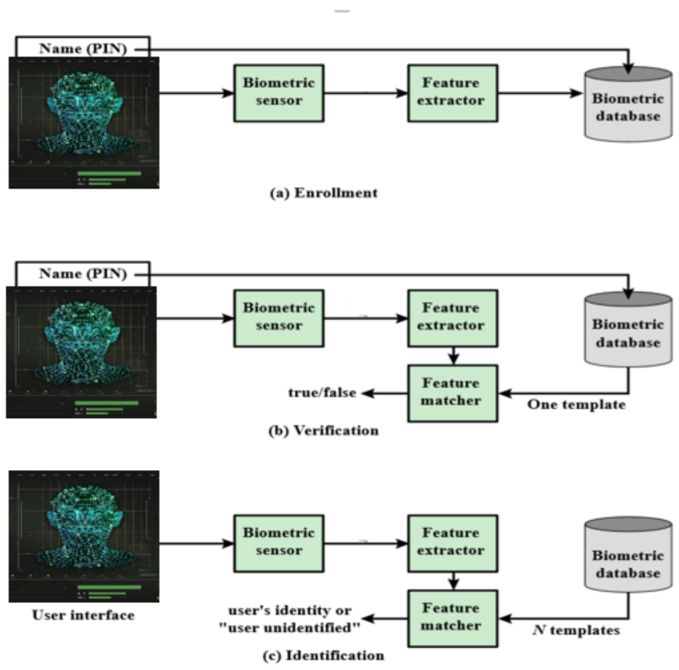


Figure 2: A Generic Biometric System. Enrolment creates an association between a user and the user's biometric characteristics. Depending on the application, user authentication either involves verifying that a claimed user is the actual user, or identifying an unknown user [14].

Conventional strategies use classifiers like neural networks (NN) or support vector machines (SVM) to measure similarities between two images and classify them as the same person or imposter using some threshold. However, these methods have limitations when it comes to facial recognition due to the sheer volume of data that requires processing before a decision can be made. Better results of neural networks applied to the facial recognition problems have come from siamese convolutional neural networks. Convolutional networks (CNNs) are non-linear, multi-layer neural networks that operate pixel by pixel to learn low-level features and high-level representations in a unified way.

Past research techniques tend not to perform any significant pre-processing to the image pairs and features are typically extracted from the entire facial image (global CNNs). They do not address the shortcomings that come up when the image face is disfigured or deformed. We propose an improvement to the conventional siamese CNN architecture by creating a neural network whose input is not the whole face but patches/regions of the face (patch-based siamese CNN). This technique would allow the network to specialize on a smaller area and is less likely to give a false acceptance or false rejection when the entire face is unavailable.

The remainder of this document is structured as follows: literature review, problems to be addressed, scope, methods, dataset, evaluation metrics, possible extensions and work-plan.

2 Literature Review

Facial verification studies, prior to 2014, generally compared features extracted from two faces separately before the idea of a siamese architecture. The concept of siamese architecture was first introduced by [1] who applied it to signature verification. Siamese neural network is an architecture of twin neural networks with identical weights that take different inputs and work simultaneously to compute comparable output vectors. Chopra [3] replaced the subnets of the siamese neural network with CNNs and applied it to face verification. The CNNs map input patterns into a low-dimensional target. This idea started with the PCA-based eigenface method [17] which is invariant to geometric distortions and small differences in input pairs. This drives the computing of a similarity metric between the patterns. The learned similarity metric later allows the matching of new persons from faces not seen during training. The authors in [3] trained and tested this technique on input face images from a combination of the AR dataset and the FERET dataset. They achieved verification equal error rate (EER) of 2.5% though the network partially saw subjects used for testing during training. The strength of this technique is that invariant effects do not come from previous knowledge about the task, but are learned during training. This overcomes the shortcomings of previous techniques that are sensitive to geometric transformations and distortions in the input image pairs. However, due to the complex architecture of CNNs, this system is inefficient in terms of speed.

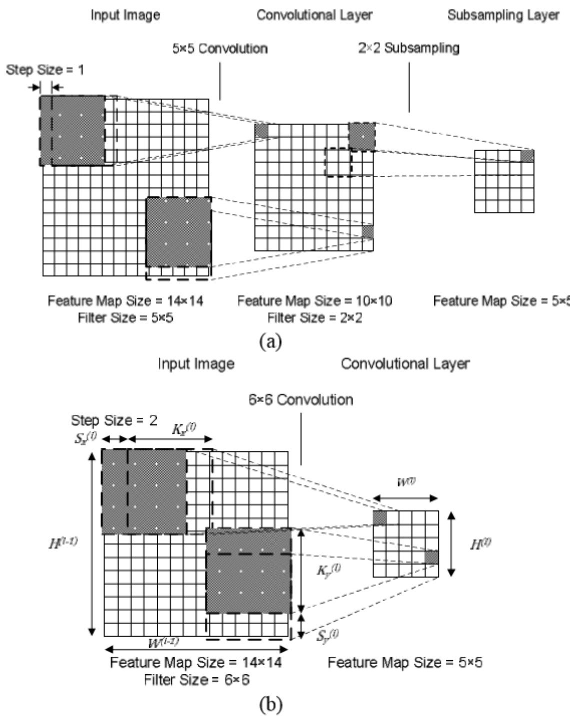


Figure 3: Convolution layer (a) Convolution (with stride of 1) followed by subsampling; (b) convolution operation (with stride of 2) [8].

The authors in [8] improved Chopra's design in terms of computational speed and complexity by fusing the convolutional and subsampling layers of the CCNs in the model, making it a four-layer CNN architecture (an idea introduced by [12] in handwriting digit recognition). Figure 3 shows the change in the architecture of the convolutional neural subnets. These authors applied this model on the AT&T dataset and achieved an equal error rate (EER) of 3.33%. This technique was able to classify a pair of images in 0.6 milliseconds, which is significantly faster than [3]. It could also verify test subjects not seen during training.Through the past years, there have been different variants of deep convolution networks that differ in terms of the model architecture. Some popular ones are explained in the papers [9] [11] [13] [16] [6] [5] [7]. Some say the paper [10] was the pioneering publication, but [9] is regarded as the most influential paper. The architecture of the network, called AlexNet, peeved the way for CNNs. It has a relatively simple layout architecture, and it achieved a top 5 test error rate of 15.4% (top 5 error is the rate at which given an image, the model does not output the correct label with its top 5 predictions). In 2012 the model ZF Net [11] fine-tuned AlexNet to improve performance through GPUs and achieved an 11.2% error rate. In this paper, the authors clearly show how to visualize the filters and weights correctly. Most past proposed models and techniques extract features from global facial images. None of them explore the viability of patch-based feature extraction. In this paper, we will explore the viability of region-specific feature extraction with siamese CNNs.

3 Problems to Be Addressed

This research aims to design and implement techniques and experiments to evaluate the viability of patch-based siamese CNNs for facial image matching. The work should then answer the question of whether this technique of patch-based feature extraction can be more effective than global CNNs that extract features from the entire facial images when applied to images taken in real environments.

4 Scope

The latest study on siamese CNNs applied to facial recognition has shown excellent results, hence this study only adds a patch-based feature extraction approach instead of a global feature extraction approach to the already proposed and tested technique in [8] to show its viability. As in the past studies, we will not perform any other major pre-processing to our dataset to represent the real environment. We will limit our study to only feature extraction. Our model will not address other factors like facial orientation but will only focus on frontal upright facial images. We will not perform segmentation on the images to find the face in them; we assume that the model will receive only facial images. In actual practice, these pre-processing functions are performed by other parts of the technology before being passed into our proposed feature extraction model, and there has also been a great deal of research on them.

5 Methods

We will model a siamese CNN similar to the one proposed in [8]. This will form the controlled variable of our experiment such that when we feed patches instead of global facial images to the model, we will be able to conclude that our approach is viable, with empirical evidence. The major component of this task is creating the actual siamese CNN. We will use potential tools like OpenCV, Matlab, Numpy, and Keras to provide implemented functions for image processing and CNNs.

5.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) or ConvNets have had a significant advancement in image analysis due to their specialization characteristics by detecting patterns and making sense of them. The main three layers of a CNN that enable this specialization are: the Convolution layer, pooling/subsampling layer, and fully-connected layer.

The convolution layer of the network receives an input image and outputs a stack of filtered images (feature maps) to the next layer using the convolution operator. The number of output feature maps is determined by what we have set as the number of filters. This layer detects different features using different filters (edges, shapes, texture, objects, etc.). A filter is simply a small matrix in which we determine the dimensions. The values of the filter are initially randomized, and will be learned during training. The deeper the network goes, the more sophisticated the filter becomes such that rather than detecting edges or shapes, they may be able to detect specific objects like eyes and nose.

The general equation for spatial filtering (correlation) is:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{t} w(s,t) f(x+s, y+t) \tag{1}$$

where x and y are varied so that each pixel in w visits every pixel in f. This can be expressed as equation 2 given the original image pixels (Figure 4) and filter (Figure 5) shown.

a	b	c
d	**e**	f
g	h	i

Figure 4: Original image Pixels

r	s	t
u	**v**	w
x	y	z

Figure 5: Filter

$$e_{(processed)} = v*e+r*a+s*b+t*c+u*d+w*f+x*g+y*h+z*i \quad (2)$$

Convolution works the same except the filter is first rotated by 180°. The general equation for the convolution operator is therefore given by:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{t} w(s,t) f(x-s, y-t) \quad (3)$$

This can be expressed as equation 4 given the original image pixels and filter shown above.

$$e_{(processed)} = v*e+z*a+y*b+x*c+w*d+u*f+t*g+s*h+r*i \quad (4)$$

The pooling/subsampling layer, shrinks the stack of feature maps by down sampling the features so that the model learns fewer parameters during training, reducing the chance of over-fitting. This is done by stepping through each of the filtered images from the convolution layer with a filter of a particular window (usually two) and by a particular stride (usually two). Equation 5 and 6 gives the width and height of the resulting images after pooling. Max pooling is a typical filtering operation used for this layer. It works by taking a maximum value from each window, and this works better than average pooling.

$$output_w = Image_w - Filter_w + 1 \tag{5}$$

$$output_h = Image_h - Filter_h + 1 \tag{6}$$

The pooling/subsampling layer usually includes an activation function, or may be put as a separate layer. We will use the Rectified linear activation function (ReL). The function steps through every pixel in a given image, returning it directly if it is positive. Otherwise, it will return zero as shown in Figure 6. We will use this function, instead of the sigmoid or hyperbolic tangent activation function, to avoid the vanishing gradient problem.

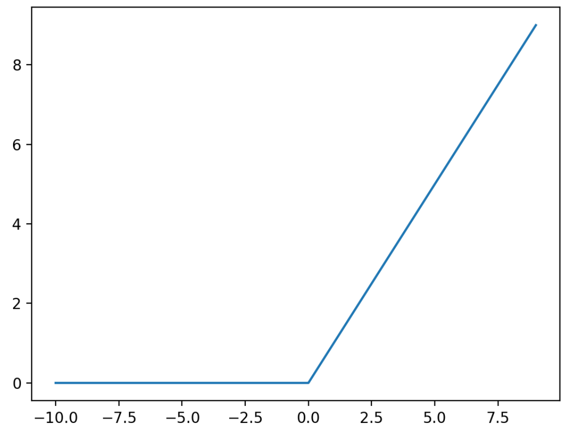


Figure 6: Rectified Linear Activation for Negative and Positive Inputs [2].

The last layer of the network is the fully-connected layer. This layer takes a list of flattened feature values from the last pooling layer into a 1D-feature vector. The rational of this architecture is that the convolution layer provides a low dimension, invariant feature space, and a fully connected layer learning a non-linear function in that space. The learning happens through back-propagation and gradient descent. The model learns features (filter values) in the convolution layer and weights in the fully-connected layer. We are at liberty to select hyperparameters for the model. Some are mentioned above, others include how many of each type of layers and in what order. A typical structure of CNN is shown in Figure 7.

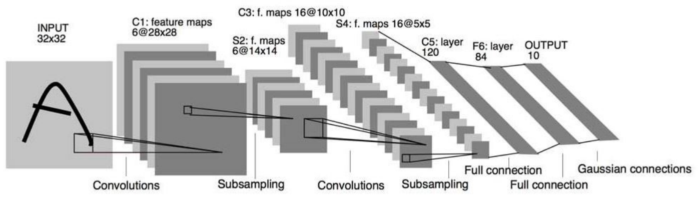


Figure 7: Architecture of LeNet-5, a Convolutional Neural Network [10].

5.2 Siamese Architecture

The siamese architecture is a network of two identical neural networks that share weights. It receives two inputs and returns a similarity measure that tells us how similar the two inputs are. The two neural networks are replaced with convolutional neural networks and applied to facial recognition. The CNNs get us two features that give us a similarity measure by taking the element-wise absolute difference. Through this, we can deduce if they are genuine pairs or imposter pairs. A distance function is learned between the two vector representations produced by the same neural networks, such that two equal faces would have similar feature vectors, a small absolute difference, and a high similarity score. In contrast, two different faces would have different feature vectors, a high absolute difference and, a low similarity score. During training, we propagate and update the model parameters so that the conditions above are satisfied. We will use the L1 Norm (Manhattan Distance) to calculate the similarity measure (E_w) shown in Figure 8.

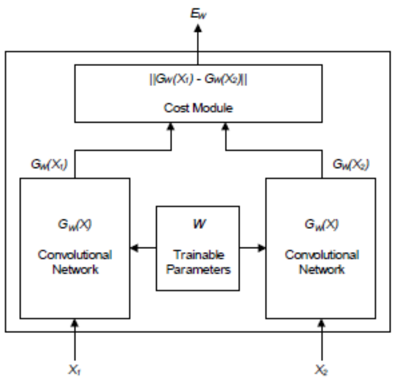


Figure 8: Siamese Architecture [8].

5.3 Loss Function

A loss function is used to calculate the model error. Its job is to represent all aspects of the model into a single number, and improvements on that number are a sign of a better model. We will apply a contrastive loss function from [chopra] shown in equation 7.

$$L(W, Y, X_1, X_2) = (1 - Y)L_G(E_W) + YL_1(E_W) \tag{7}$$

$$L(W, Y, X_1, X_2) = (1 - Y)\frac{2}{Q}(E_W)^2 + 2Qe^{\frac{-2.7726E_W}{Q}} \tag{8}$$

where Q denotes the total number of neuron outputs in the fully-connected layer.

6 Evaluation and Validation

We hope to evaluate and validate that patch-based features learned by the model are more effective than those based on complete facial images. We expect that training the model on a smaller area (patches) will enable a deep learning model to be less dependent on complete information but maximize the available discriminative properties, which can lead to accurate identification when some parts of the face are unavailable.

6.1 Dataset

We will train and test the proposed technique on the AT&T dataset [4]. The dataset consists of 40 different subjects, each with ten different gray-scale facial images taken against a dark homogeneous background with subjects in an upright and frontal position (with tolerance for some side movement). The images vary in facial expressions (open eyes/closed eyes/smiling/not smiling) and facial details (glasses/ no glasses) and lighting. We will partition the dataset into two disjoint sets: one for training the model and the other for testing.

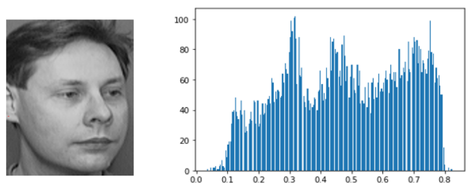


Figure 9: Sample 1

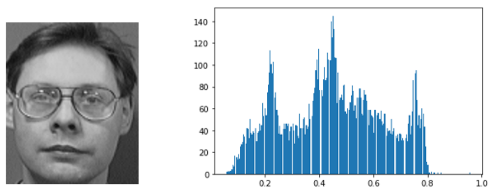


Figure 10: Sample 2

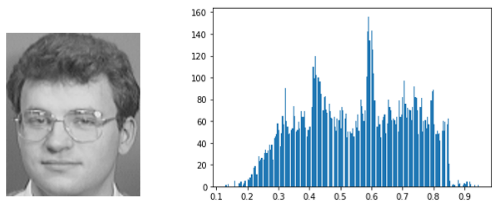


Figure 11: Sample 3

Figure 12: Sample images from the AT&T Dataset with corresponding Image Histograms

To a computer, the task of image matching is not a straight forward one since two sample images of the same person (sample 1 and sample 2), as shown in Figure 12, can have different gray level distributions. An imposter image (sample 3) can have similar grey level distributions to a genuine sample.

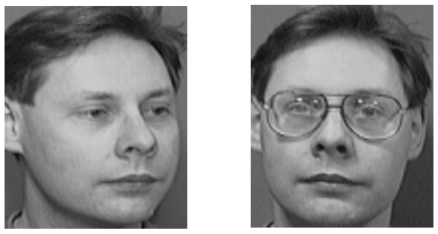


Figure 13: Genuine Pair

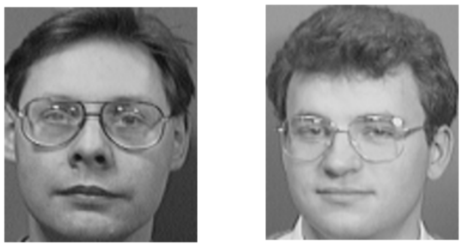


Figure 14: Imposter Pair

Figure 15: Genuine and Imposter Pairs

The model should be able to distinguish Figure 13 as a genuine pair and Figure 14 as an imposter pair.

6.2 Evaluation Metrics

Our evaluation metrics for the proposed technique will be accuracy, which we will get from the calculated equal error rate (EER) derived from a combination of false acceptance rate (FAR) and false rejection rate (FRR) on our test dataset. FAR is the likelihood of the model incorrectly accepting an imposter sample, equation 8. FRR is the likelihood of the model incorrectly rejecting a genuine sample, equation 9. EER is the value when FAR is equal to FRR. The lower the EER the higher the accuracy of the model.

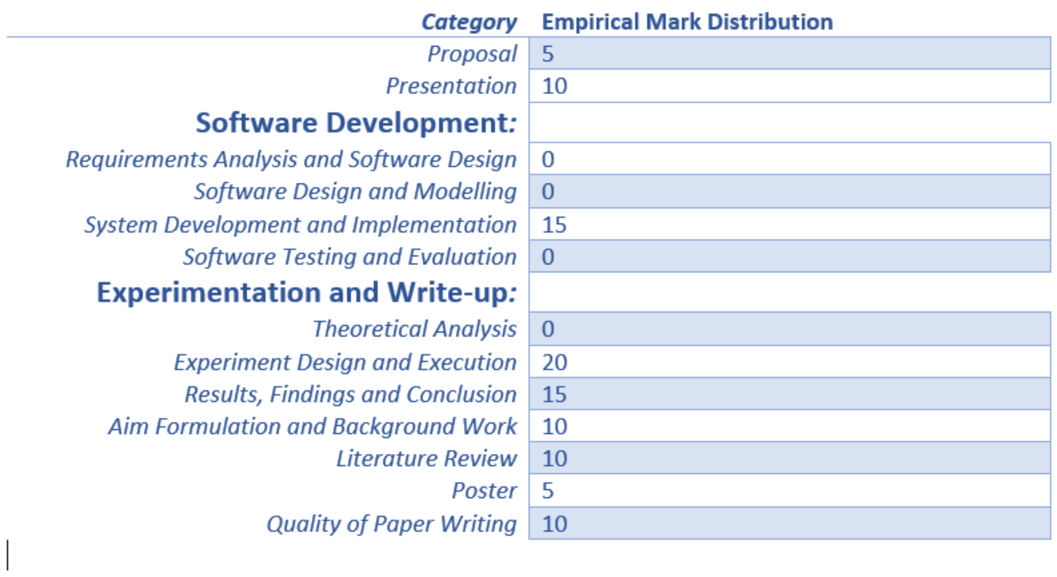
$$FAR = \frac{No.of falseacceptances}{No.of identification attempts} \tag{9}$$

$$FAR = \frac{No.of falserejections}{No.of identification attempts} \tag{10}$$

7 Possible Extensions

The siamese architecture can be implemented such that the two CNNs run parallel to each other to speed up performance. Pre-processing techniques like segmentation can be performed onto the sample images before feeding them into the model to enhance feature extraction.

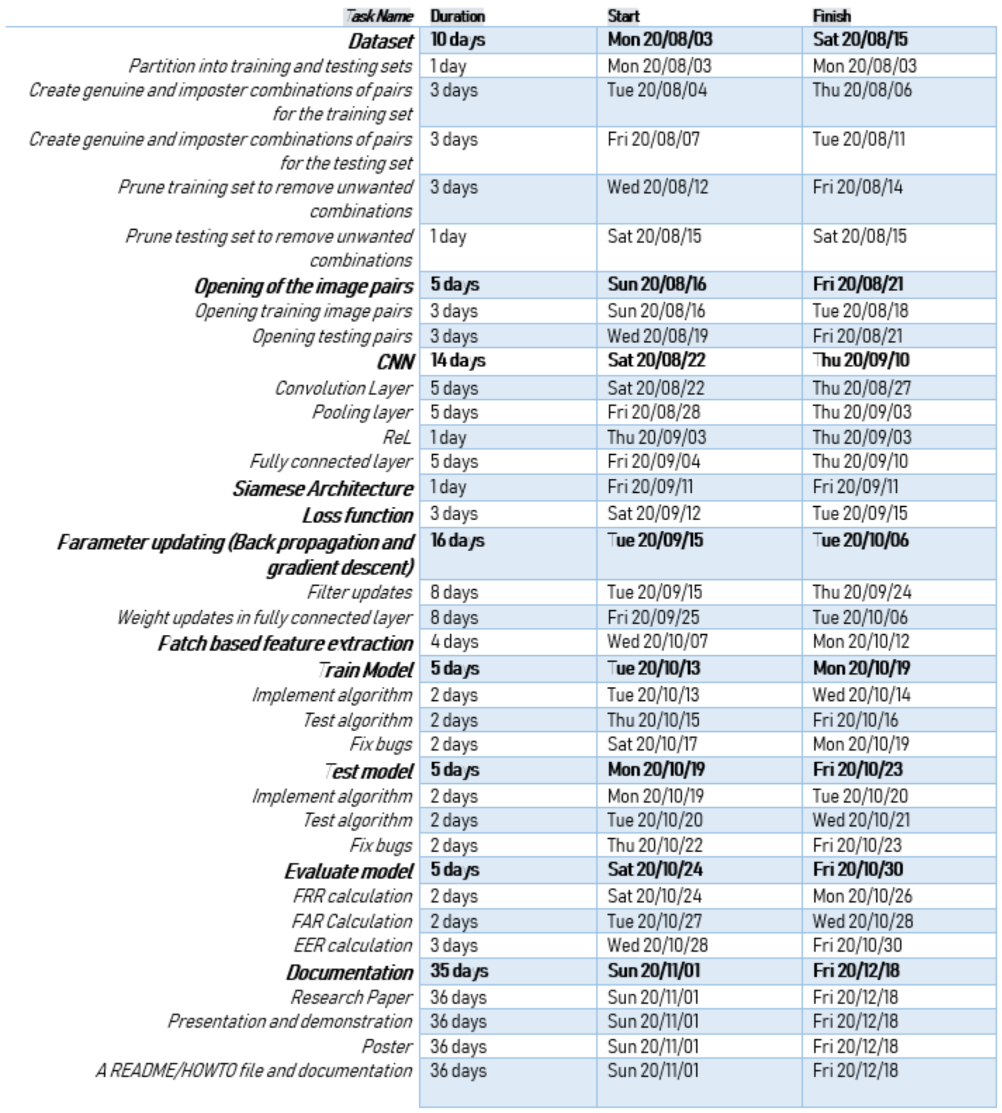
8 Marking guide



Category	Empirical Mark Distribution
Proposal	5
Presentation	10
Software Development:	
Requirements Analysis and Software Design	0
Software Design and Modelling	0
System Development and Implementation	15
Software Testing and Evaluation	0
Experimentation and Write-up:	
Theoretical Analysis	0
Experiment Design and Execution	20
Results, Findings and Conclusion	15
Aim Formulation and Background Work	10
Literature Review	10
Poster	5
Quality of Paper Writing	10

Figure 16: Empirical mark distribution

9 Work-plan



Task Name	Duration	Start	Finish
Dataset	**10 days**	**Mon 20/08/03**	**Sat 20/08/15**
Partition into training and testing sets	1 day	Mon 20/08/03	Mon 20/08/03
Create genuine and imposter combinations of pairs for the training set	3 days	Tue 20/08/04	Thu 20/08/06
Create genuine and imposter combinations of pairs for the testing set	3 days	Fri 20/08/07	Tue 20/08/11
Prune training set to remove unwanted combinations	3 days	Wed 20/08/12	Fri 20/08/14
Prune testing set to remove unwanted combinations	1 day	Sat 20/08/15	Sat 20/08/15
Opening of the image pairs	**5 days**	**Sun 20/08/16**	**Fri 20/08/21**
Opening training image pairs	3 days	Sun 20/08/16	Tue 20/08/18
Opening testing pairs	3 days	Wed 20/08/19	Fri 20/08/21
CNN	**14 days**	**Sat 20/08/22**	**Thu 20/09/10**
Convolution Layer	5 days	Sat 20/08/22	Thu 20/08/27
Pooling layer	5 days	Fri 20/08/28	Thu 20/09/03
ReL	1 day	Thu 20/09/03	Thu 20/09/03
Fully connected layer	5 days	Fri 20/09/04	Thu 20/09/10
Siamese Architecture	1 day	Fri 20/09/11	Fri 20/09/11
Loss function	3 days	Sat 20/09/12	Tue 20/09/15
Farameter updating (Back propagation and gradient descent)	**16 days**	**Tue 20/09/15**	**Tue 20/10/06**
Filter updates	8 days	Tue 20/09/15	Thu 20/09/24
Weight updates in fully connected layer	8 days	Fri 20/09/25	Tue 20/10/06
Fatch based feature extraction	4 days	Wed 20/10/07	Mon 20/10/12
Train Model	**5 days**	**Tue 20/10/13**	**Mon 20/10/19**
Implement algorithm	2 days	Tue 20/10/13	Wed 20/10/14
Test algorithm	2 days	Thu 20/10/15	Fri 20/10/16
Fix bugs	2 days	Sat 20/10/17	Mon 20/10/19
Test model	**5 days**	**Mon 20/10/19**	**Fri 20/10/23**
Implement algorithm	2 days	Mon 20/10/19	Tue 20/10/20
Test algorithm	2 days	Tue 20/10/20	Wed 20/10/21
Fix bugs	2 days	Thu 20/10/22	Fri 20/10/23
Evaluate model	**5 days**	**Sat 20/10/24**	**Fri 20/10/30**
FRR calculation	2 days	Sat 20/10/24	Mon 20/10/26
FAR Calculation	2 days	Tue 20/10/27	Wed 20/10/28
EER calculation	3 days	Wed 20/10/28	Fri 20/10/30
Documentation	**35 days**	**Sun 20/11/01**	**Fri 20/12/18**
Research Paper	36 days	Sun 20/11/01	Fri 20/12/18
Presentation and demonstration	36 days	Sun 20/11/01	Fri 20/12/18
Poster	36 days	Sun 20/11/01	Fri 20/12/18
A README/HOWTO file and documentation	36 days	Sun 20/11/01	Fri 20/12/18

Figure 17: Project Plan showing milestones(bold) and respective tasks to achieve them

References

[1] Jane Bromley et al. “Signature Verification using a Siamese Time Delay Neural Network”. In: *Int.]. Pattern Recognit. Artzf Intell* 7 (1993).

[2] Jason Brownlee. “A gentle introduction to the rectified linear unit (relu)”. In: *Machine Learning Mastery. https://machinelearningmastery.com/rectified-linear-activation-function-fordeep-learning-neural-networks* (2019).

[3] Sumit Chopra, Raia Hadsell, and Yann LeCun. “Learning a similarity metric discriminatively, with application to face verification”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 539–546.

[4] Andy Hopper FREng. “The ORL face database”. In: *AT&T (Olivetti) Research Laboratory Cambridge, “ht tp://www. uk. research. att. com/facedatabase. html* 199.2 (1992).

[5] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[6] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[7] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. “Spatial transformer networks”. In: *Advances in neural information processing systems*. 2015, pp. 2017–2025.

[8] Mohamed Khalil-Hani and Liew Shan Sung. “A convolutional neural network approach for face verification”. In: *2014 International Conference on High Performance Computing & Simulation (HPCS)*. IEEE. 2014, pp. 707–714.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[10] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[11] D Matthew and R Fergus. "Visualizing and understanding convolutional neural networks". In: *Proceedings of the 13th European Conference Computer Vision and Pattern Recognition, Zurich, Switzerland.* 2014, pp. 6–12.

[12] Patrice Y Simard, David Steinkraus, John C Platt, et al. "Best practices for convolutional neural networks applied to visual document analysis." In: *Icdar*. Vol. 3. 2003. 2003.

[13] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[14] William Stallings et al. *Computer security: principles and practice.* Pearson Education Upper Saddle River, NJ, USA, 2012.

[15] Michal Strahilevitz. "Facial Recognition Bans: What Do They Mean For AI (Artificial Intelligence)?" In: (2020).

[16] Christian Szegedy et al. "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2015, pp. 1–9.

[17] Matthew Turk and Alex Pentland. "Eigenfaces for recognition". In: *Journal of cognitive neuroscience* 3.1 (1991), pp. 71–86.