APPLICATION OF SENTIMENT ANALYSIS TO LANGUAGE LEARNING

SEMINAR REPORT

Submitted in the partial fulfilment of the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

of

APJ Abdul Kalam Technological University

by

GEORGY JOSE VILAVINAL

November, 2018

Department of Computer Engineering

College of Engineering, Chengannur, Kerala -689121

Phone: (0479) 2454125, 2451424; Fax: (0479) 2451424

COLLEGE OF ENGINEERING, CHENGANNUR
KERALA

Department of Computer Engineering

CERTIFICATE

This is to certify that the seminar entitled

APPLICATION OF SENTIMENT ANALYSIS TO LANGUAGE LEARNING

Submitted by

GEORGY JOSE VILAVINAL

is a bonafide record of the work done by him.

Co-ordinator **Head of The Department**

ACKNOWLEDGEMENT

I am greatly indebted to God Almighty for being the guiding light throughout with his abundant grace and blessings that strengthened me to do this endeavour with confidence.

I express my heartfelt gratitude towards **Dr. Jacob Thomas V.**, Principal, College of Engineering Chengannur for extending all the facilities required for doing my seminar. I would also like to thank **Dr. Smitha Dharan**, Head, Department of Computer Engineering, for providing constant support and encouragement.

Now I extend my sincere thanks to my seminar co-ordinators **Mr. Gopakumar G.**, Associate Professor in Computer Engineering and **Ms. Meera Varma** , Assistant Professor in Computer Engineering for guiding me in my work and providing timely advices and valuable suggestions.

Last but not the least, I extend my heartfelt gratitude to my parents and friends for their support and assistance.

ABSTRACT

With the growing use of biometric authentication systems in the recent years, spoof fingerprint detection has become increasingly important. In this study, we use Convolutional Neural Networks (CNN) for fingerprint liveness detection. Our system is evaluated on the datasets used in The Liveness Detection Competition of years 2009, 2011 and 2013, which comprise almost 50,000 real and fake fingerprints images. Dataset Augmentation is used to increase the classifiers performance, not only for deep architectures but also for shallow ones. We also report good accuracy on very small training sets (400 samples) using these large pre-trained networks. Our best model achieves an overall rate of 97.1% of correctly classified samples - a relative improvement of 16% in test error when compared with the best previously published results

Contents

1 INTRODUCTION **1**

2 LITERATURE REVIEW **2**

3 IMPLEMENTATION STAGES **5**

3.1 Data Pre-Processing 6

3.2 CNN Modelling 7

3.2.1 Training 9

3.2.2 Testing 10

3.2.3 Prediction accuracy 10

4 EVALUATION **11**

5 CONCLUSIONS **12**

References **13**

6 SCREENSHOTS **15**

7 APPENDIX **17**

7.1 Code 17

List of Figures

31 Proposed model architecture 5

32 Examples of real and fake fingerprint images 6

33 single layer convolutional network 7

34 Fully Connected Network 9

61 Training Dataset 15

62 Output 15

63 Training Result 16

64 Predicted Accuracy 16

1 INTRODUCTION

The basic aim of biometrics is to automatically discriminate subjects in a reliable manner for a target application based on one or more signals derived from physical or behavioral traits, such as fingerprint, face, iris, voice, palm, or handwritten signature. Biometric technology presents several advantages over classical security methods based on either some information(PIN, Password, etc.) or physical devices (key, card, etc.) [2]. However, providing to the sensor a fake physical biometric can be an easy way to overtake the systems security. Fingerprints, in particular, can be easily spoofed from common materials, such as gelatin, silicone, and wood glue [2]. Therefore, a safe fingerprint system must correctly distinguish a spoof from an authentic finger (Figure 1). Different fingerprint liveness detection algorithms have been proposed [3], [4], [5], and they can be broadly divided into two approaches: hardware and software. In the hardware approach, a specific device is added to the sensor in order to detect particular properties of a living trait such as blood pressure [6], skin distortion [7], or odor [8]. In the software approach, which is used in this study, fake traits are detected once the sample has been acquired with a standard sensor.

The features used to distinguish between real and fake fingers are extracted from the image of the fingerprint. There are techniques such as those in [2] and [9], in which the features used in the classifier are based on specific fingerprint measurements, such as ridge strength, continuity, and clarity.Convolutional Neural Networks were used to detect false vs real fingerprints. Pre-trained CNNs can yield state-of-the-art results on benchmark datasets without requiring architecture or hyper parameter selection. We also showed that these models have good accuracy on very small training sets (˜400 samples). Additionally, no task-specific hand-engineered technique was used as in classical computer vision approaches. Despite the differences between images acquired from different sensors, we show that training a single classifier using all datasets helps to improve accuracy and robustness. This suggests that the effort required to design a liveness detection system (such as hyper-parameters fine tuning) can be significantly reduced if different datasets (and acquiring devices) are combined during the training of a single classifier. Additionally, the pre-trained networks showed stronger generalization capabilities in cross-dataset experiments than CNN with random weights and the classic LBP pipeline.

2 LITERATURE REVIEW

Fingerprint Recognition Using Genetic Algorithm and Neural Network *Purneet Kaur , Jaspreet Kaur*[10] In this paper,feature selection method is based on Genetic algorithm. The fingerprint image is enhanced before extracting features. Enhancement method included histogram equilization, binarization, morphological operations, it has made a great improvement of recognition accuracy for recognition method. The combination of both genetic algorithm and neural network techniques provided the better and efficient method for fingerprint biometric. Experimental results show that the presented method has the better recognition accuracy compared with the previous fuzzy logic based recognition methods.

Fingerprint Recognition and Matching *Aliyu Tukur* [11] The reliability of any automatic fingerprint verification system strongly relies on the precision obtained in the minutia extraction process. A number of factors damage the correct location of minutia. Among them, poor image quality is the one with most influence. The proposed minutiae matching algorithm is capable of finding the correspondences between minutiae without resorting to exhaustive research. However, there is a scope of further improvement in terms of efficiency and accuracy which can be achieved by improving the hardware to capture the image or by improving the image enhancement techniques.

Fingerprint identification and recognition using backpropagation neural network *Adrian Lim Hooi Jinl, Ai Chekima, Jamal Ahmad Dargham, and Liau Chung Fan*[12] In this paper, to enhance the system constructed features extraction has to be implemented inside this system. Here features extracted from the fingerprint image can be classified into local features and global features, thus increasing the accuracy of further fingerprint identification and recognition process. After the image has been processed it would then be fed into the back propagation neural network as input in order to train the network. After training, the neural network is ready to perform the identification and recognition operations (matching process). A neural network has been necessarily developed to identify and recognize the core part of the fingerprint images.

Fingerprint Spoof Detection Using Contrast Enhancement and Convolutional Neural Networks*Han-Ul Jang, Hak-Yeol Choi, Dongkyu Kim, Jeongho Son,and Heung-Kyu Lee* [13] In this paper, we propose a technique to detect fingerprint spoof using contrast enhancement

and CNN.Here uses histogram equalization as a contrast enhancement technique to improve the recognition rate of fingerprint images and detects fake fingerprints by judging whether or not the sub-block of fingerprint image is forged through CNNs. The proposed CNNs is composed of 6 weight layers and totalizing the results. The experimental results show that the average accuracy is 99.8

Fingerprint Liveness Detection based on Weber Local Image Descriptor*Diego Gragnaniello, Giovanni Poggi, Carlo Sansone and Luisa Verdoliva* [14]In this paper, we investigate the use of a local discriminative feature space for fingerprint liveness detection. In particular, we rely on the Weber Local Descriptor (WLD), which is a powerful and robust descriptor recently proposed for texture classification. Inspired by Weber's law, it consists of two components, differential excitation and orientation, evaluated for each pixel of the image. Joint histograms of these components are then processed to build the discriminative features used to train a linear kernel SVM classifier. Experimental results with different databases and different sensors show WLD to perform favorably compared to the state-of-the-art methods in fingerprint liveness detection. In addition, by combining WLD with LPQ (Local Phase Quantization) results further improve significantly.

Fingerprint Spoof Buster*Tarang Chugh, Kai Cao, and Anil K. Jain,*[15] A robust and accurate method for fingerprint spoof detection is critical to ensure the reliability and security of the fingerprint authentication systems. Here utilized fingerprint domain knowledge by extracting local patches centered and aligned using minutiae in the input fingerprint image for training MobileNet-v1 CNN models. The local patch based approach provides salient cues to differentiate spoof fingerprints from live fingerprints. The proposed approach is able to achieve a significant reduction in the error rates for intra-sensor (63as well as cross-dataset scenarios (29on public domain LivDet datasets.

A Study of Biometric Approach Using Fingerprint Recognition *Ravi Subban and Dattatreya P. Mankame*[16] This paper presented the related works and performance analysis for fingerprint biometric. The performance evaluation is done on surveyed works with different parameters and existing methods. Biometrics presents obvious advantages over password and token-based security. The survey study various issues related to uni-modal biometric systems is discussed. The security and privacy concerns that biometric authentication raises need to be

addressed. It is surveyed that automatic fingerprint recognition is the best candidate biometric technology for explosives security from an analysis of the requirements: security, usability, ruggedness, size, form factor, privacy and operational temperature range.

Fingerprint recognition using neural network*W. F. Leung, S. H. Leung, W. H. Lau and Andrew Luk* [17]This paper describes a neural network based approach for automated fingerprint recognition. Minutiae are extracted from the fingerprint image via a multilayer perceptron (MLP) classifier with one hidden layer. The backpropagation learning technique is used for its training. Selected features are represented in a special way such that they are simultaneously invariant under shift, rotation and scaling. Simulation results are obtained with good detection ratio and low failure rate. The proposed method is found to be reliable for system with a small set of fingerprint data.

Fingerprint Detection and Authentication using feature extraction based on Minutiae *Hriday Goyal, Gaurav Verma, Chetan Arora*[18]. This paper deals with the fingerprint detection and Authentication using feature extraction based on minutiae. Several concepts of image processing like image enhancement, image segmentation, image binarization and morphology are used in it. Various algorithms are developed to complete the above-mentioned task and for minutiae matching. The calculations are equipped for discovering correspondences between information minutiae design and put away minutiae design without depending on thorough inquiry. Execution of the created framework is then assessed on a database with fingerprints from various individuals.

Fingerprint liveness detection based on binarized statistical image feature with sampling from Gaussian distribution*Qiaoqiao Li,Patrick P. K. Chan* [19]In this paper,we proposed a revised method by collecting features based on the points which conform to Gaussian distribution to detect spoof fingerprint attacks in fingerprint biometric systems.This method obtains the texture characters from a single image by using binarized statistical image feature descriptor. A number of pixels are sampled according to the two-dimension Gaussian distribution. The features are extracted from the patch centered with each sampled pixel.

3 IMPLEMENTATION STAGES

Three main implementation stages are:

- Pre-processing
- CNN Modelling
- prediction

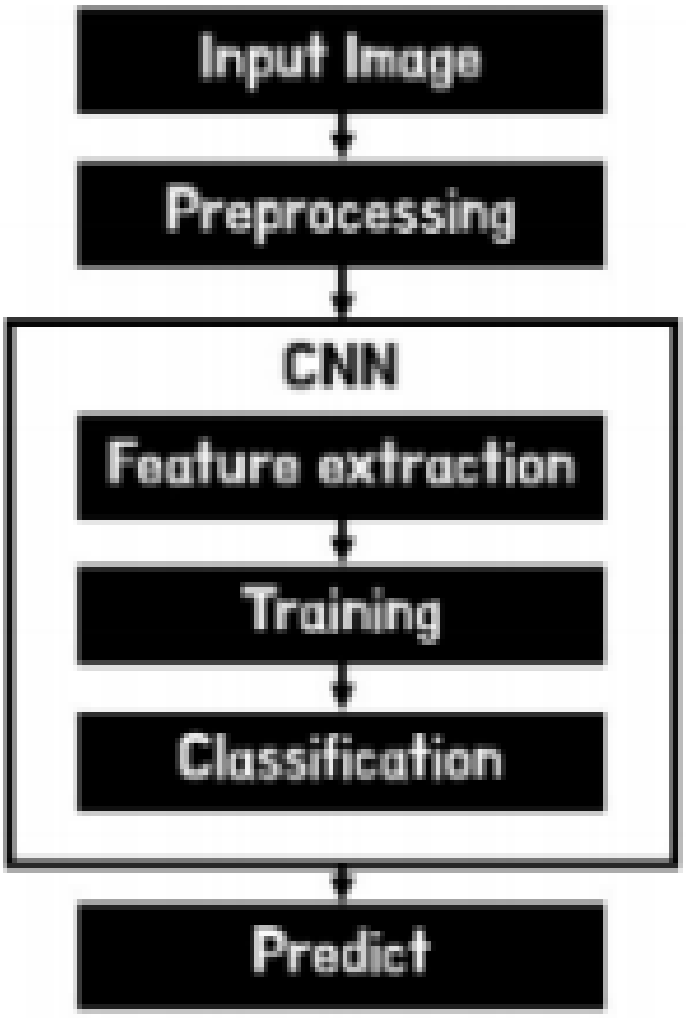

Figure 31: Proposed model architecture

3.1 Data Pre-Processing

As the first step, Collecting sample images of fake and real fingerprint from the dataset provided by Liveness Detection Competition of the year 2015.It contains both training and tesing dataset. The training data set contains almost 50,000 real and fake fingerprints images.The quality of the fingerprint image are crucial for the recoginition process. Initially we have to load both the training and testing datasets into our program.Then display the training images and its count.

The images are of different sizes. So we want to tranform the images into equal sizes.In Keras, image preprocessing task is performed by using ImageDataGenerator Class. Here perform resize, horizontalflip,shear,zooming operations.

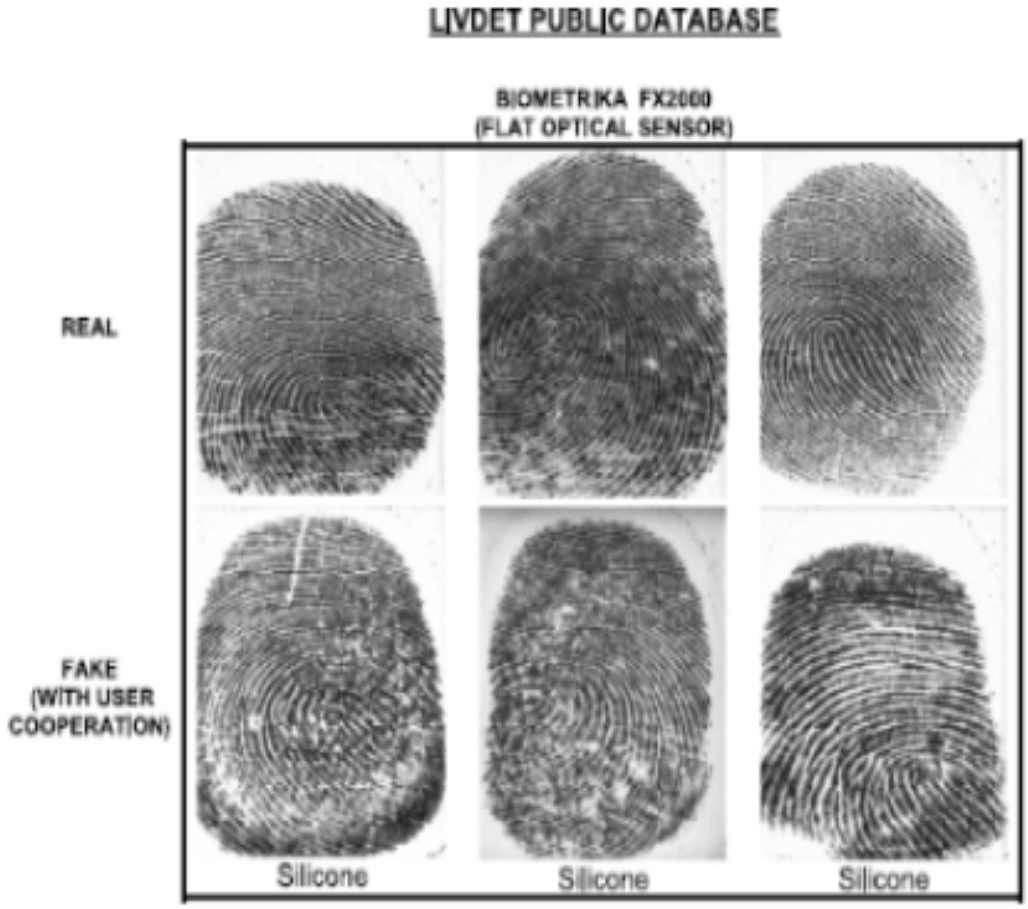

Figure 32: Examples of real and fake fingerprint images

3.2 CNN Modelling

CNN Model consist of feature extraction, training and classification tasks. Figure 3.3 illustrates the feed-forward pass of a single layer convolutional network. The input sample is convoluted with three random filters of size 5x5 (enlarged to make visualization easier), generating 3 convoluted images, which are then subject to non-linear function max(x; 0), followed by a max-pooling operation, and subsampled by a factor of 2.

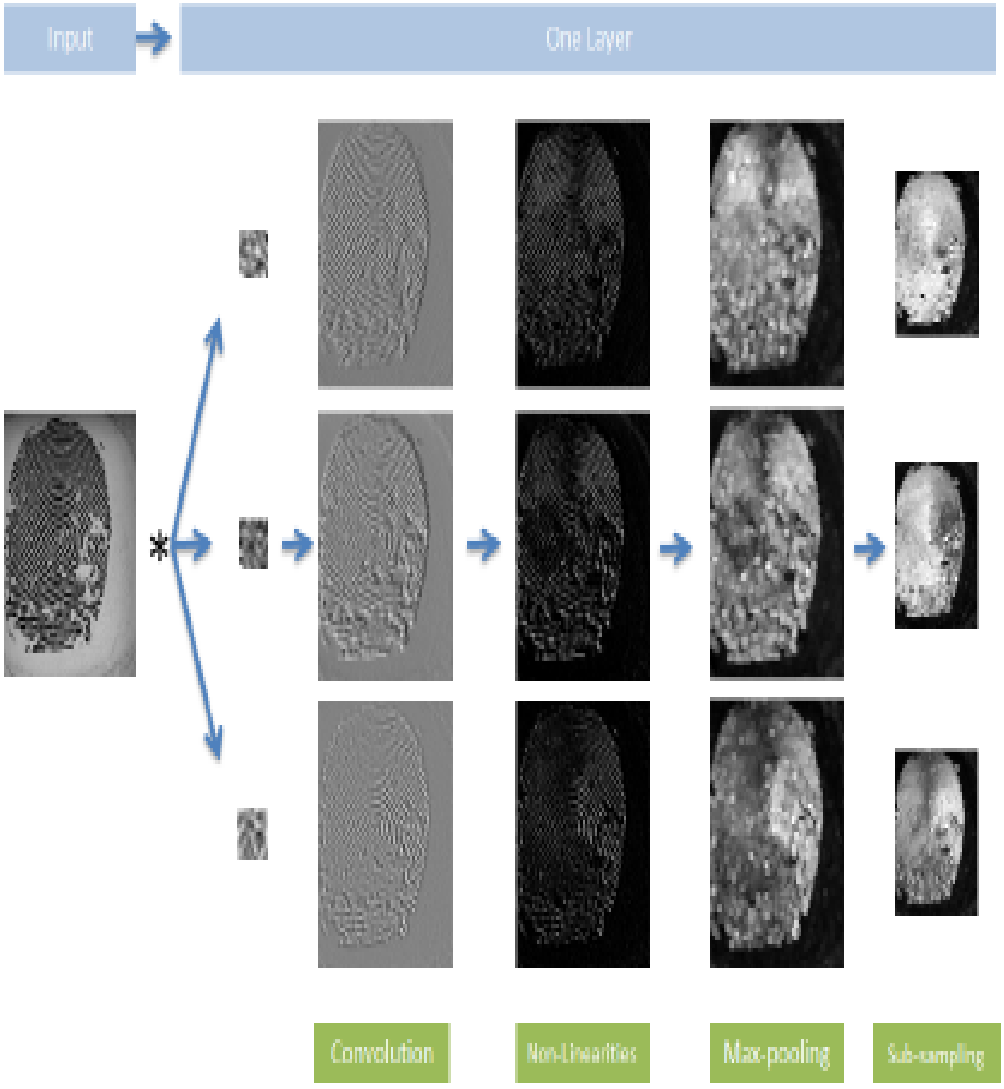

Figure 33: single layer convolutional network

Here there are three main layers

- ReLU layer
- Pooling Layer
- Fully Connected Layer

Keras is used to model this network. Keras is a deep learning library in python. The core data structure of Keras is a model, a way to organize layers. The simplest type of model is the Sequential model, a linear stack of layers. This model is initialised and the input image is pass throgh a convolution layer.ReLU is Rectified Linear Units. This layer applies the non-saturating activation function f(x)=max(0,x). It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer.

Then it pass through a pooling layer.The pooling layer makes the CNN less sensitive to small changes in the location of a feature. There are a number of ways to implement pooling, but the most effective is max pooling. Max pooling is also referred to as a downsampling layer. Max pooling uses the maximum value from each of a cluster of neurons at the prior layer. Max pooling reduces the size of feature map.To perform max pooling, imagine a window(size 2x2) sliding across the feature map. As the window moves across the map, we take the largest value in the window and discard the rest. Then it pass through a fully connected network. Neurons in a fully connected layer have full connections to allactivations in the previous layer, as seen in regular NeuralNetworks The fully-connected layer has 3 parts - an input layer, a hidden layer, and an output layer. The input layer is the output of the preceding layer, which is just an array of values.

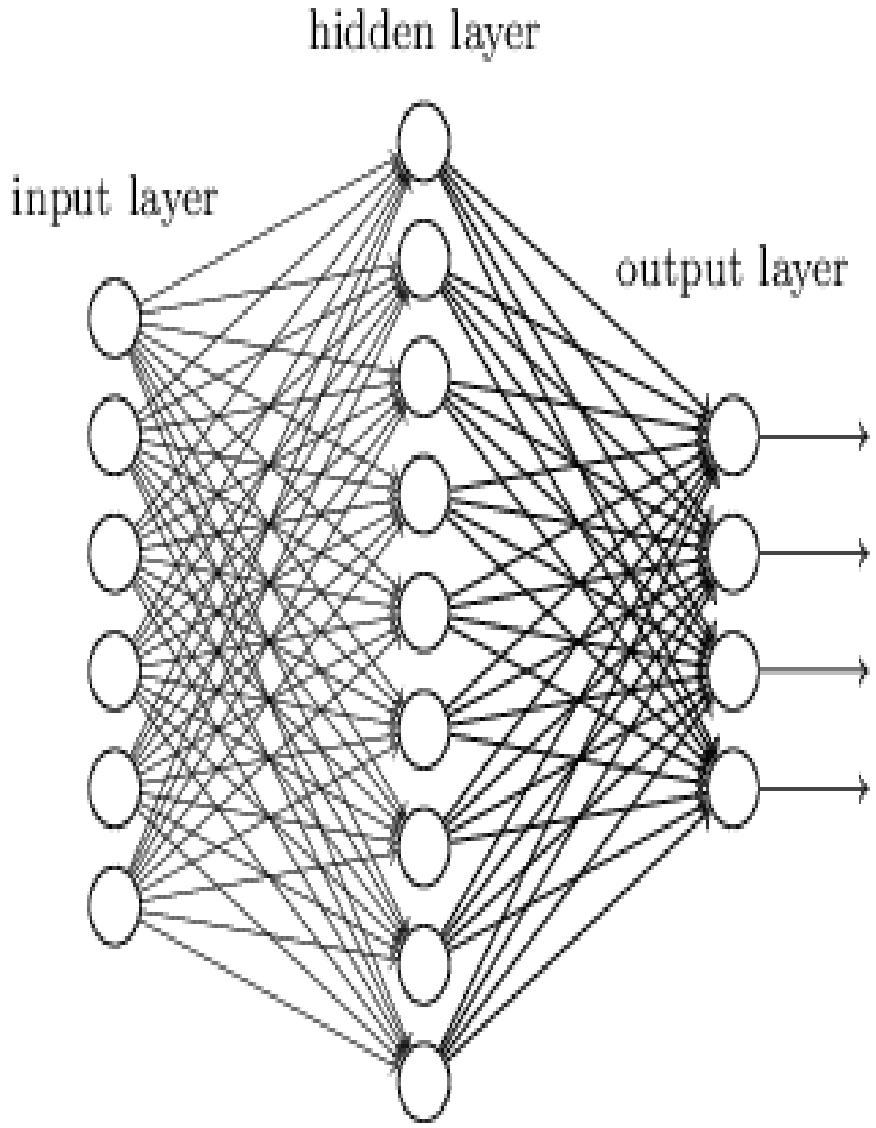

Figure 34: Fully Connected Network

3.2.1 Training

Covolutional Neural Network can play a critical role in fingerprint liveness detection. An CNN can be configured and trained to handle such variations observed in the texture of the fingerprint. Extracted features of all the images in the data set are the input of the neural network. Here AdamOptimizer (Adaptive Moment Estimation) is used to get faster convergence. AdamOptimizer is a type of Gradient Descent optimization algorithm. It is an optimization algorithm that can used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.Here take 15 epochs for training and weight updation is performed.So loss can be calculated in each Epoch. Loss will be decreased while the training progressing. Based on the weight updation, a model is generated.

3.2.2 Testing

Based on the generated model from training, testing is performed. To recognize the fingerprint take any image from the data set and fed that image to the trained network.It gives the result by showing whether it matches to real or fake fingerprint and display the result.

3.2.3 Prediction accuracy

Compute the prediction accuracy based on the number of real and fake images. Displaying the accuracy of the real and fake test dataset.

```
x = np.ones((1,leny),dtype=np.float32)
y = np.float32(y)
y = y.transpose()
acc1 = abs(x-y)
acc = np.sum(acc1)
acc = (leny-acc)/leny
acc = acc*100
```

4 EVALUATION

- Evaluation of the network using test dataset sample images.
- Here we can find out the Accuracy of the Model.
- In this, I have tested the model with 10 sample images.
- Get the result whether the sample image is a fake or real fingerprint.
- Compute the predicted accuracy.

5 CONCLUSIONS

Convolutional Neural Networks were used to detect false vs real fingerprints. Pre-trained CNNs can yield state-of-the-art results on benchmark datasets without requiring architecture or hyper parameter selection. We also showed that these models have good accuracy on very small training sets (˜400 samples). Additionally, no task-specific hand-engineered technique was used as in classical computer vision approaches. Despite the differences between images acquired from different sensors, we show that training a single classifier using all datasets helps to improve accuracy and robustness. This suggests that the effort required to design a liveness detection system (such as hyper-parameters fine tuning) can be significantly reduced if different datasets (and acquiring devices) are combined during the training of a single classifier. Additionally, the pre-trained networks showed stronger generalization capabilities in cross-dataset experiments than CNN with random weights and the classic LBP pipeline.

References

[1] V. Mura, L. Ghiani, G. L. Marcialis, F. Roli, D. A. Yambay, and S. A. Schuckers, *"Livdet 2015 fingerprint liveness detection competition 2015."*

[2] J. Galbally, F. Alonso-Fernandez, J. Fierrez, and J. Ortega-Garcia, *"A high performance fingerprint liveness detection method based on quality related features,"* Future Generation Computer Systems, vol. 28, no. 1, pp. 311–321, 2012.

[3] Y. Chen, A. Jain, and S. Dass, *"Fingerprint deformation for spoof detection,"* in Biometric Symposium, 2005, p. 21.

[4] B. Tan and S. Schuckers, *"Comparison of ridge-and intensity-based perspiration liveness detection methods in fingerprint scanners,"* in Defense and Security Symposium. International Society for Optics and Photonics, 2006, pp. 62 020A–62 020A.

[5] P. Coli, G. L. Marcialis, and F. Roli, *"Fingerprint silicon replicas: static and dynamic features for vitality detection using an optical capture device,"* International Journal of Image and Graphics, vol. 8, no. 04, pp. 495–512, 2008.

[6] P. D. Lapsley, J. A. Lee, D. F. Pare Jr, and N. Hoffman, *"Anti-fraud biometric scanner that accurately detects blood flow,"* Apr. 7 1998, uS Patent 5,737,439.

[7] A. Antonelli, R. Cappelli, D. Maio, and D. Maltoni, *"Fake finger detection by skin distortion analysis,"* Information Forensics and Security, IEEE Transactions on, vol. 1, no. 3, pp. 360–373, 2006.

[8] Baldisserra, A. Franco, D. Maio, and D. Maltoni, *"Fake fingerprint detection by odor analysis,"* in Advances in Biometrics. Springer, 2005, pp. 265–272.

[9] A. K. Jain, Y. Chen, and M. Demirkus, *"Pores and ridges: Highresolution fingerprint matching using level 3 features,"* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 29, no. 1, pp. 15–27, 2007.

[10] Purneet Kaur , Jaspreet Kaur "Fingerprint Recognition Using Genetic Algorithm and Neural Network"

[11] Aliyu Tukur "Fingerprint Recognition and Matching"

[12] Adrian Lim Hooi Jinl, Ai Chekima, Jamal Ahmad Dargham, and Liau Chung Fan ”Fingerprint identification and recognition using backpropagation neural network”

[13] Han-Ul Jang, Hak-Yeol Choi, Dongkyu Kim, Jeongho Son,and Heung-Kyu Lee ”Fingerprint Spoof Detection Using Contrast Enhancement and Convolutional Neural Networks”

[14] Diego Gragnaniello, Giovanni Poggi, Carlo Sansone and Luisa Verdoliva”Fingerprint Liveness Detection based on Weber Local Image Descriptor”

[15] arang Chugh, Kai Cao, and Anil K. Jain, ”Fingerprint Spoof Buster”

[16] avi Subban and Dattatreya P. Mankame ”A Study of Biometric Approach Using Fingerprint Recognition”

[17] . F. Leung, S. H. Leung, W. H. Lau and Andrew Luk ”Fingerprint recognition using neural network”

[18] riday Goyal, Gaurav Verma, Chetan Arora, ”Fingerprint Detection and Authentication using feature extraction based on Minutiae”

[19] iaoqiao Li,Patrick P. K. Chan, ”Fingerprint liveness detection based on binarized statistical image feature with sampling from Gaussian distribution”

[20] https://ieeexplore.ieee.org/document/7390065/(Accessed On 18/03/18)

6 SCREENSHOTS

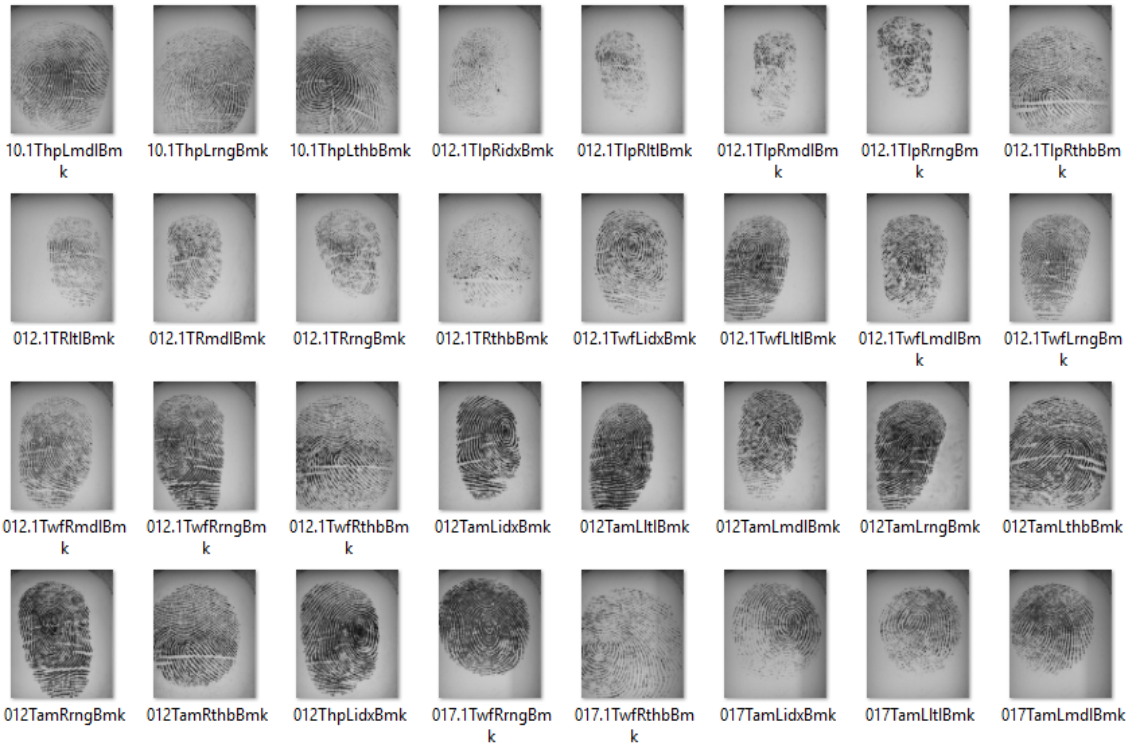

Figure 61: Training Dataset

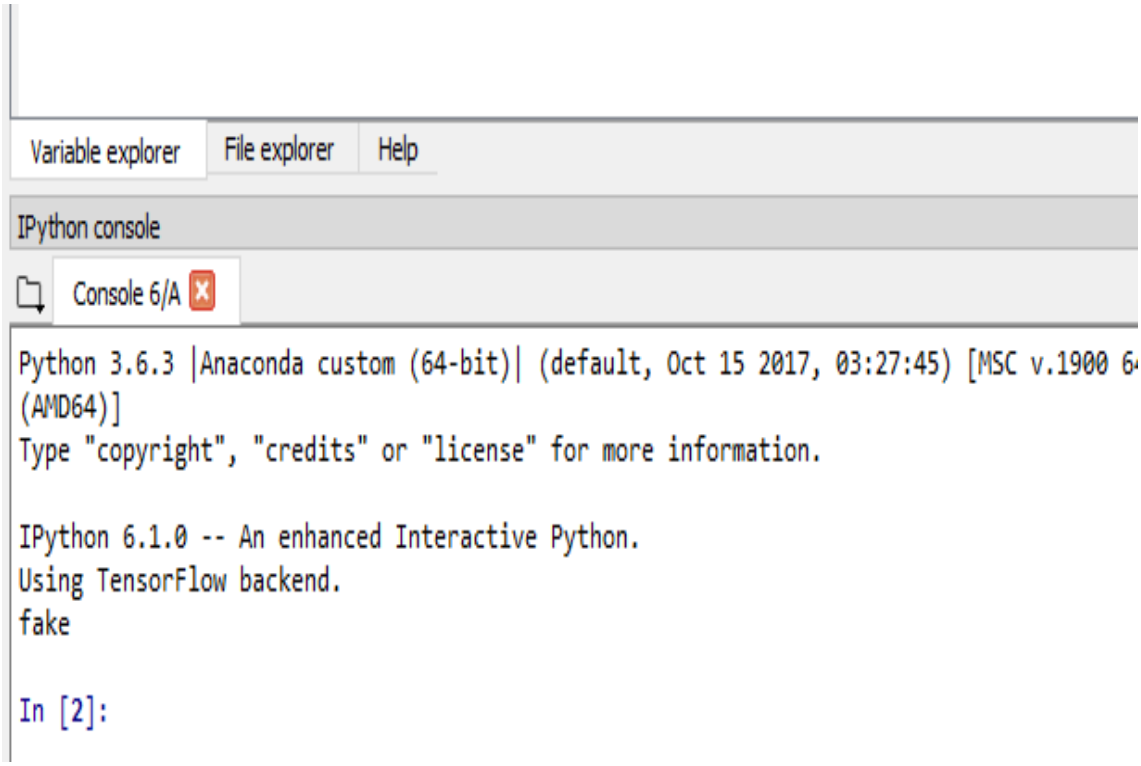

Figure 62: Output

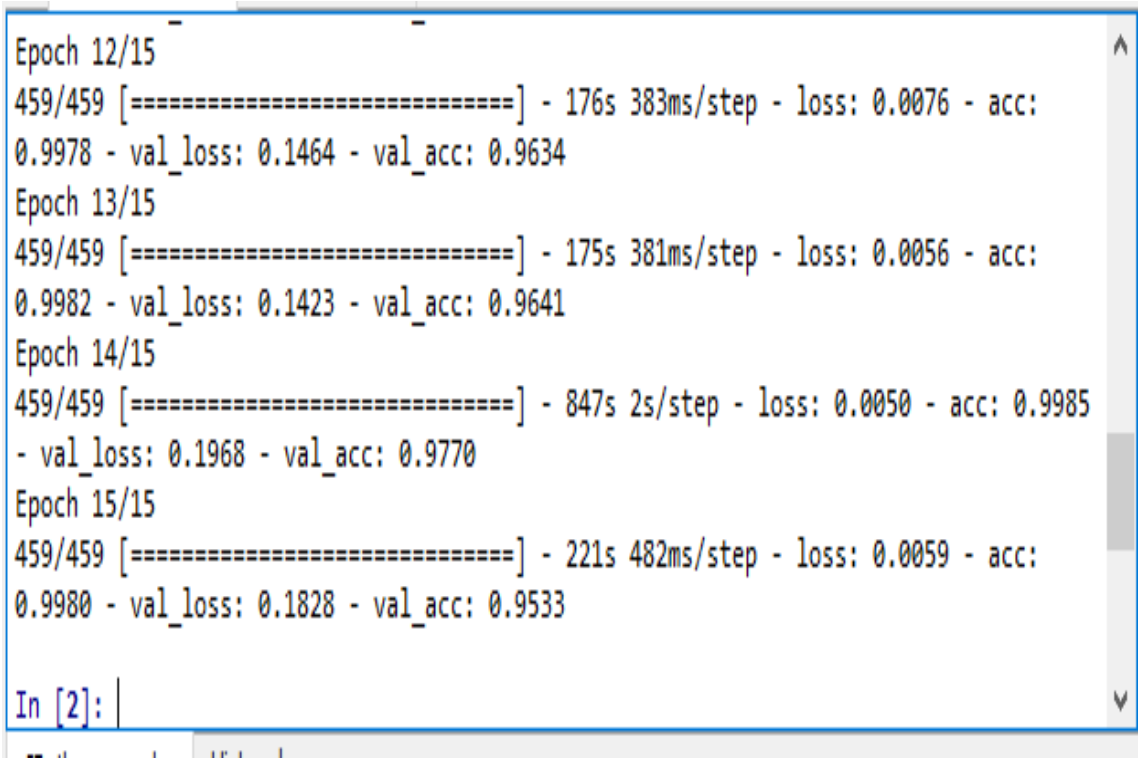

Figure 63: Training Result

```
runfile('F:/mm/CNN1_test_all.py', wdir='F:/mm')
Using TensorFlow backend.
fake
real
fake
fake
real
real
real
real
real
real
real
real
real
real
real
fake
real
real
fake
fake
fake
real
real
fake
fake
real
real
real
real
real
Prediction Accuracy is
70.7317073171
```

Figure 64: Predicted Accuracy

7 APPENDIX

7.1 Code

Code for Training

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
classifier = Sequential()
classifier.add(Conv2D(32, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))
classifier.add(Conv2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))
classifier.add(Flatten())
classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dense(units = 1, activation = 'sigmoid'))
classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
training_set = train_datagen.flow_from_directory(r'F:\mm\training_set1',
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                 class_mode = 'binary')
test_set = test_datagen.flow_from_directory(r'F:\mm\test_set',
                                            target_size = (64, 64),
                                            batch_size = 16,
                                            class_mode = 'binary')
classifier.fit_generator(training_set,
                         steps_per_epoch = 459,
                         epochs = 15,
                         validation_data = test_set,
                         validation_steps = 105)
classifier.save('my_model2.h5')
```

Code for Testing

```
import numpy as np
import cv2
from keras.preprocessing import image
from keras.models import load_model

model = load_model('my_model2.h5')

from tkinter.filedialog import askopenfilename
img_path = askopenfilename()

img = cv2.imread(img_path)
cv2.imshow("Input",img)
cv2.waitKey(100)


test_image = image.load_img(img_path, target_size = (64, 64))
test_image1 = image.img_to_array(test_image)
test_image2 = np.expand_dims(test_image1, axis = 0)
result = model.predict(test_image2)
#training_set.class_indices
if result[0][0] == 1:
  prediction = 'real'
else:
  prediction = 'fake'



print(prediction)
```

Prediction computing code

```
import numpy as np
import glob
from keras.preprocessing import image
from keras.models import load_model
model = load_model('my_model2.h5')
db = r"F:\mm\test_set\real\*.png"
names = []
y = []
for i in glob.glob(db):
    names.append(i)
    test_image = image.load_img(i, target_size = (64, 64))
    test_image1 = image.img_to_array(test_image)
    test_image2 = np.expand_dims(test_image1, axis = 0)
    result = model.predict(test_image2)
    #training_set.class_indices
    clas = result[0][0]
    if clas == 1:
      prediction = 'real'
      y.append(clas)
    else:
      prediction = 'fake'
      y.append(clas)
    print(prediction)
leny = len(y)
# use if fake folder  x = np.zeros((1,leny),dtype=np.float32)
# use if real folder
x = np.ones((1,leny),dtype=np.float32)
y = np.float32(y)
y = y.transpose()
acc1 = abs(x-y)
acc = np.sum(acc1)
acc = (leny-acc)/leny
acc = acc*100
print("Prediction Accuracy is")
print(acc)
```