

Human Signature Verification Using Cnn with Deployment Using Rough Django Framework

M.A.Berlin¹, Saikrishna S¹, Kummari Venkata Vamsi¹,
Nemani Jaya Rakesh¹

¹Department of CSE, R.M.D. Engineering College, Kavaraipettai, Tamil
Nadu 601206, India

mab.cse@rmd.ac.in, arunadithya11599@gmail.com, venkatavamsi1818@gmail.com,
ucs17236@rmd.ac.in

ABSTRACT

Signature verification is the process of verifying a persons identity from falsification. It is a biometric methodology and determines the processed signature is a real signature or forged one. In this paper, a signature verification system is proposed for validating any individual's signature using Convolutonal Neural Networks (CNN).

1. Introduction

Signature verification is a widely used biometric methodology in order to authenticate a person or any individual. The signature of a person is considered as a approval of the user and also treated as “seal of authentication” [R. K. Mohapatra]. Identifying a real signature from a forged one is a key objective of this paper. In this paper, the process of finding genuine signature is automated using Convolution Neural Networks.

Now a days signatures are used as formal step for making agreement or other legal documents. However, identifying and validating the genuine signature from the altered one is a difficult task [S. M. A. Navid]. Hence many approaches and algorithms are deleloped by many researches using machine learning algorithms.

Signatures are broadly classified into two types: i) offline or handwritten signature (static) and ii) online signature(dynamic) [Eman Alajrami]. Both the types of signatures differs in its viability. The offline signatures requires the physical presence of the person concerned whereas the online signatures requires the documents can be e-Signed from anywhere with an Internet-enabled device. This can be captured using webcam [K S Radhika]

So offline signatures are more difficult to verify. In this paper we have proposed a Deep Learning (DL) based offline signature verification method to prevent signature fraudulent by malicious people.

The DL method used in the proposed approach is the Convolutional Neural Network (CNN). It is predicted that the success of the obtained results will increase if the CNN method is supported by adding extra feature extraction methods and classify successfully human hand signature.

As the same person's signature may vary in different parameters, in this paper we have addressed a human signature verification system using CNN. The following sections are organized as follows: Section 2 describes the related work. Section 3 describes the proposed system in detailed manner. Section 4 discusses the experimental results and section 5 concludes the paper.

2. Related Work

The authors S. Zagoruyko et al [3], E. zgndz et al [4], E. J. Justino et al [5], and Yeung et al [6], determines the offline signatures using template matching. Template matching is used in digital image processing for identifying a part of the image that matches template image. Bailing Zhang [7] proposed Deep CNNs for determining forgery and real signatures using two phase approaches: i) writer-independent feature learning, and ii) writer dependent classification.

Daramola and T. S. Ibiyemi presented a offline signature recognition system using Discrete Cosine Transform and Hidden Markov Model [9] . The authors had the image pre-processing step for smoothening the image and it was converted into a binary image by using morphological operation. The features of the images from signature were extracted using Discrete Cosine Transform and Hidden Markov Model was used for verification. Similarities between signature images was identified using a probabilistic pattern matching technique.

However, in spite of that the same person's two different signatures may vary in the parameters such as the start and finish position of the signature, angle of inclination of the signatures, space between each letter, shape of the curve and height of each letter. Therefore determining the real signature is a serious issue and need to be addressed in an efficient way.

3. PROPOSED SYSTEM

The focus of our proposed system is to verify human signature using CNN. It is planned to design deep learning technique so that a person with lesser expertise in software should also be able to use it easily. Therefore the key objective of our proposed system are listed below:

- Verify offline signature is real or forged one.
- Large number of signatures need to be collected as samples

- Deep Learning algorithm is used to classify the forged and real signatures
- Samples of a greater number of images are collected that comprised of different signatures. We proposed a Deep Learning (DL) based offline signature verification method to prevent signature fraud by malicious people. The DL method used in the study is the Convolutional Neural Network (CNN). It is predicted that the success of the obtained results will increase if the CNN method is supported by adding extra feature extraction methods and classify successfully human hand signature.

ADVANTAGES OF PROPOSED SYSTEM

To identify the signature for the genuine and forged signatures. It is best model for deep learning technique to easily identify the human hand signature. Figure 1 shows the architecture of the proposed system.

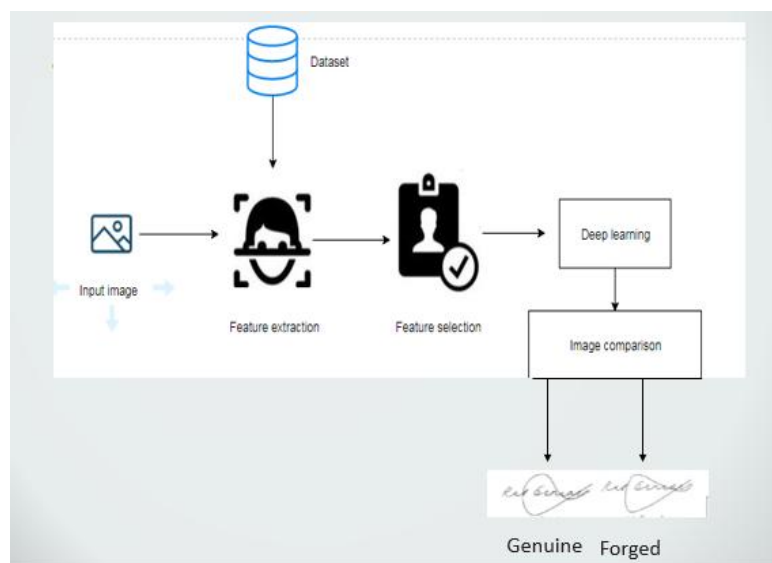


Figure 1: Architecture of the proposed system.

The proposed system comprises the following steps:

1. An image signature is given as input from dataset for feature extraction.
2. Train the data using AlexNET and LENET
3. Deploying the Model in Django Framework and Predicting Output

The entire project mainly consists of 4 modules, which are

Step1: Import the Given Image from Dataset and Training Manual CNN

We have to import our data set using keras preprocessing image data generator function also we create size, rescale, range, zoom range, horizontal flip. Then we import our image dataset from folder through the data generator function. Here we set train, test, and validation also we set target size, batch size and class-mode from this function we have to train.

TO TRAIN THE MODULE BY USING ALEXNET

To train our dataset using classifier and fit generator function also we make training steps per epoch's then total number of epochs, validation data and validation steps using this data we can train our dataset. Training the module using Alexnet CNN.

TO TRAIN THE MODULE BY USING LENET

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

A. INPUT LAYER

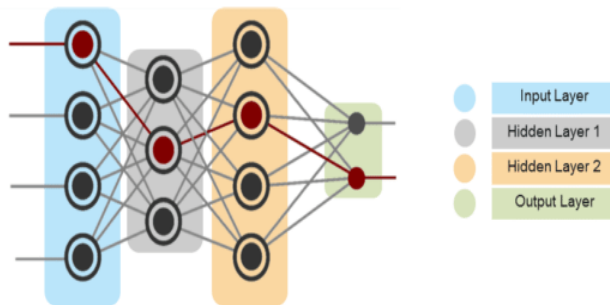
Input layer in CNN contain image data. Image data is represented by three dimensional matrixes. It needs to reshape it into a single column. Suppose you have image of dimension $28 \times 28 = 784$, it needs to convert it into 784×1 before feeding into input.

B. CONVO LAYER

Convo layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. First of all, a part of image is connected to Convo layer to perform convolution operation as we saw earlier and calculating the dot product between receptive fields (it is a local region of the input image that has the same size as that of filter) and the filter. Result of the operation is single integer of the output volume. Then the filter over the next receptive field of the same input image by a Stride and do the same operation again. It will repeat the same process again and again until it goes through the whole image. The output will be the input for the next layer

C. POOLING LAYER

Pooling layer is used to reduce the spatial volume of input image after convolution. It is used between two convolution layers. If it applies FC after Convo layer without applying pooling or max pooling, then it will be computationally expensive. So, the max pooling is only way to reduce the spatial volume of input image. It has applied max pooling in single depth slice with Stride of 2. It can observe the 4 x 4-dimension input is reducing to 2 x 2 dimensions.



D. FULLY CONNECTED LAYER (FCL)

Fully connected layer involves weights, biases, and neurons. It connects neurons in one layer to neurons in another layer. It is used to classify images between different categories by training.

E. SOFTMAX/LOGISTIC LAYER

Softmax or Logistic layer is the last layer of CNN. It resides at the end of FC layer. Logistic is used for binary classification and softmax is for multi-classification.

F. OUTPUT LAYER

Output layer contains the label which is in the form of one-hot encoded. Now you have a good understanding of CNN

DEPLOYING THE MODEL IN DJANGO FRAMEWORK AND PREDICTING OUTPUT

In this module the trained deep learning model is converted into hierarchical data format file (.h5 file) which is then deployed in our Django framework for providing better user interface and predicting the output whether the given signature is real or forged.

SYSTEM FRAMEWORK

Training from Scratch

To train a deep network from scratch, you gather a very large labeled data set and design a network architecture that will learn the features and model. This is good for new applications, or applications that will have a large number of output categories. This is a less common approach because with the large amount of data and rate of learning, these networks typically take days or weeks to train.

Transfer Learning

To train a deep network from scratch, you gather a very large labeled data set and design a network architecture that will learn the features and model. This is good for new applications, or applications that will have a large number of output categories. This is a less common approach because with the large amount of data and rate of learning, these networks typically take days or weeks to train.

Most deep learning applications use the transfer learning approach, a process that involves fine-tuning a pretrained model. You start with an existing network, such as AlexNet or GoogLeNet, and feed in new data containing previously unknown classes. After making some tweaks to the network, you can now perform a new task, such as categorizing only dogs or cats instead of 1000 different objects. This also has the advantage of needing much less data (processing thousands of images, rather than millions), so computation time drops to minutes or hours.

Feature extraction:

A slightly less common, more specialized approach to deep learning is to use the network as a **feature extractor**. Since all the layers are tasked with learning certain features from images, we can pull these features out of the network at any time during the training process. These features can then be used as input to a machine learning model such as support vector machines (SVM)

4. RESULT AND DISCUSSION

The current software provides the exact result of whether the signature is a real one or forged one

with higher accuracy (about 95%) compared to the previous algorithm. CNN model has proved be the best model when it comes to deep learning. Moreover in cases where there is a necessity to observe, analyze and extract information or perform classifications from an image, of the 3 branches of deep learning (CNN or ANN or RNN) CNN is the undeniably best option to rely on. Figure 1 shows Actual signature and Fake signature:

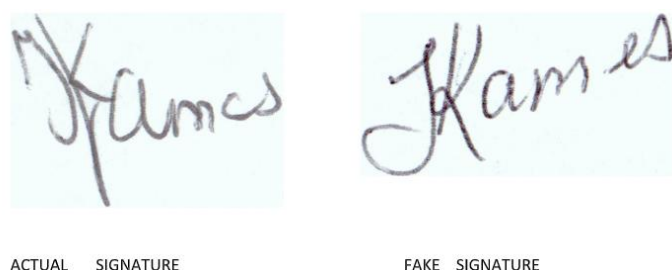


Fig. 1: Actual and Fake signature

Figure 2 shows the samples of signatures used for training



Fig.2: Training set

```

In [6]: from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, EarlyStopping

In [7]: from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Convolution2D
        from tensorflow.keras.layers import MaxPooling2D
        from tensorflow.keras.layers import Flatten
        from tensorflow.keras.layers import Dense
        import warnings
        warnings.filterwarnings('ignore')

In [8]: Classifier=Sequential()

In [9]: Classifier.add(Convolution2D(32,3,3,input_shape=(100,100,3),activation='relu'))
        Classifier.add(MaxPooling2D(pool_size=(2,2)))
        Classifier.add(Convolution2D(128,3,3,activation='relu'))
        Classifier.add(MaxPooling2D(pool_size=(2,2)))
        Classifier.add(Flatten())
        Classifier.add(Dense(256, activation='relu'))
        Classifier.add(Dense(10, activation='softmax'))

        Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['accuracy'])
        Classifier.summary()

Model: "sequential_1"
Layer (type) Output Shape Param #
-----
conv2d_2 (Conv2D) (None, 32, 32, 32) 896
max_pooling2d_2 (MaxPooling2D) (None, 16, 16, 32) 0
conv2d_3 (Conv2D) (None, 5, 5, 128) 36992
max_pooling2d_3 (MaxPooling2D) (None, 2, 2, 128) 0
flatten_1 (Flatten) (None, 512) 0
    
```

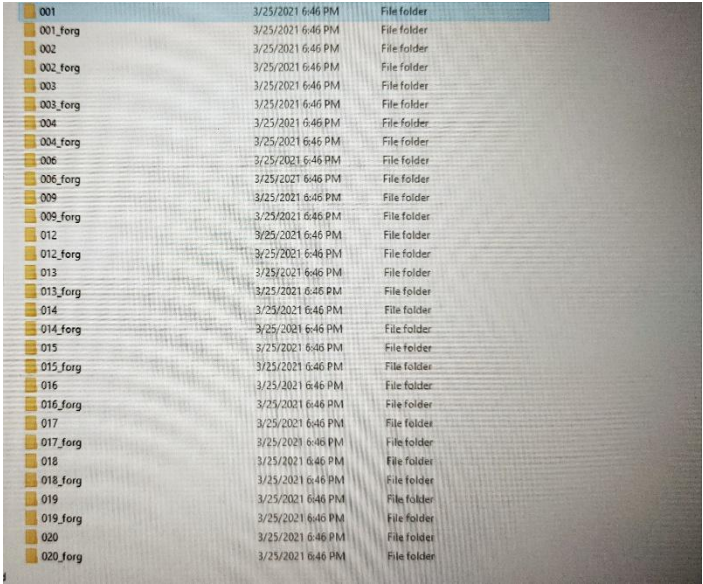
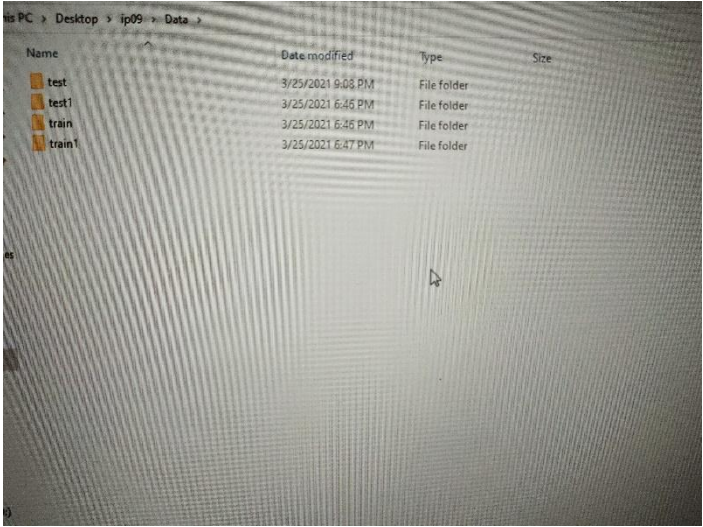
Deep learning is constantly proving itself to be the future tech in terms of the way it is going to revolutionize the other industries such as healthcare, banking sector, agriculture sector etc.

```

Package requirements 'logitech==3.2.7, Django==3.0.5, Pillow==7.1.1, sqlalchemy==0.9.1' are not satisfied
Install requirements Ignore requirements

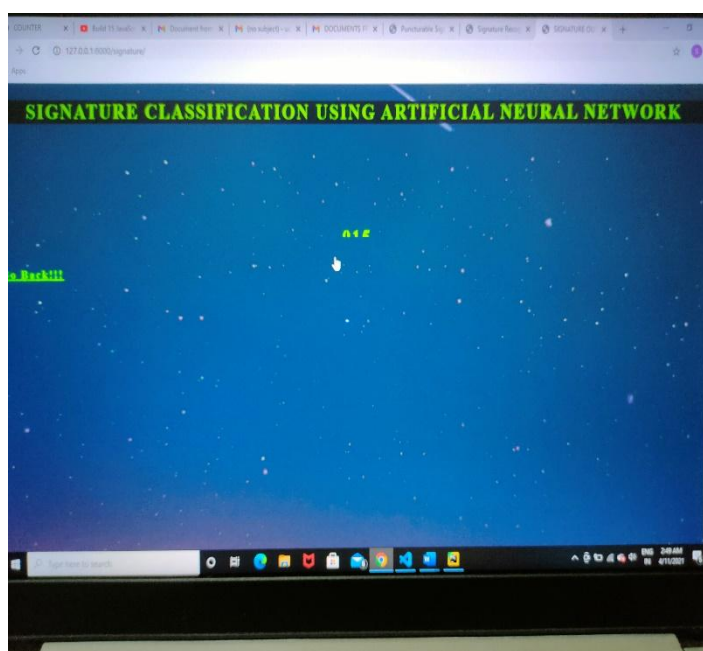
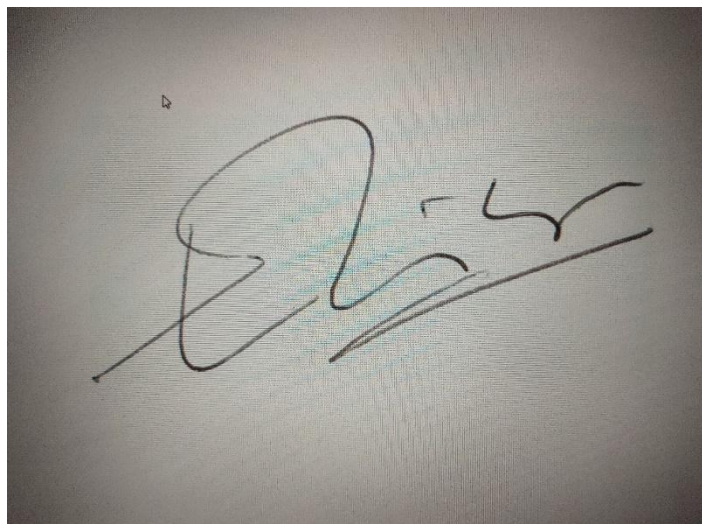
1 from django.shortcuts import render
2
3 from django.http import HttpResponseRedirect
4 from django.urls import reverse_lazy
5 from django.views.generic import TemplateView
6 from employee.forms import EmployeeForm
7
8 from django.views.generic import DetailView
9 from employee.models import Employee
10
11
12 class EmployeeImage(TemplateView):
13     form = EmployeeForm
14     template_name = 'emp_image.html'
15
16     def post(self, request, *args, **kwargs):
17         form = EmployeeForm(request.POST, request.FILES)
18
19         if form.is_valid():
20             obj = form.save()
21
22             return HttpResponseRedirect(reverse_lazy('emp_image_display', kwargs={'pk': obj.id}))
23
24         context = self.get_context_data(form=form)
25         return self.render_to_response(context)
26
27     def get(self, request, *args, **kwargs):
28         return self.post(request, *args, **kwargs)
29
30
31 class EmpImageDisplay(DetailView):
32     model = Employee
33     template_name = 'emp_image_detail.html'
34     queryset = EmployeeImage.objects.all()
    
```





5. CONCLUSION

Thus we have successfully developed a model using the AlexNet algorithm of one of the most powerful deep learning models, CNN (Convolutional Neural Network) to achieve an higher accuracy in terms of recognizing a signature and classifying it as to whether it is the corresponding person's original signature or a fake one. This can be useful in various sectors which involves collecting authentic information of the customer, employee or any other person. Some of the sectors include banking, database related fields, healthcare etc.



6. REFERENCES

1. S. M. A. Navid, S. H. Priya, N. H. Khandakar, Z. Ferdous and A. B. Haque, "Signature Verification Using Convolutional Neural Network," *2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON)*, 2019, pp. 35-39, doi: 10.1109/RAAICON48939.2019.19.
2. R. K. Mohapatra, K. Shaswat and S. Kedia, "Offline Handwritten Signature Verification using CNN inspired by Inception V1 Architecture," *2019 Fifth International Conference on Image Information Processing (ICIIP)*, 2019, pp. 263-267, doi: 10.1109/ICIIP47207.2019.8985925.
3. Eman Alajrami , Belal A. M. Ashqar , Bassem S. Abu-Nasser , Ahmed J. Khalil , Musleh M. Musleh , Alaa M. Barhoom , Samy S. Abu-Naser, "Handwritten Signature Verification using Deep Learning", *International Journal of Academic Multidisciplinary Research (IJAMR)*, Vol. 3 Issue 12, December – 2019, Pages: 39-44.
4. S. Zagoruyko, N. Komodakis "Learning to Compare Image Patches via Convolutional Neural Networks" *CVPR 2015*, arXiv:1504.03641v1

5. E. zgndz, T. entrk, and M. E. Karstlgil, Off-line signature verification and recognition by support vector machine, in European signal processing conference, EUSIPCO, 2005.
6. E. J. Justino, A. El Yacoubi, F. Bortolozzi, and R. Sabourin, An off-line signature verification system using HMM and graphometric features, in Fourth IAPR International Workshop on Document Analysis Systems (DAS), Rio de. Citeseer, 2000, pp. 211222.
7. Yeung, D.-Y., Chang, H., Xiong, Y., George, S.E., Kashi, R.S., Matsumoto, T., Rigoll, G.: SVC2004: "First International Signature Verification Competition" in ICBA 2004. LNCS, vol. 3072, pp. 16-22. Springer, Heidelberg.
8. Bailing Zhang, "Off-line signature verification and identification by pyramid histogram of oriented gradients" International Journal of Intelligent Computing and Cybernetics, 3(4):611–630, 2010.
9. S. A. Daramola and T. S. Ibiyemi, "Offline signature recognition using hidden markov model (HMM)," International journal of computer applications, vol. 10, no. 2, pp. 17-22, 2010.