2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON)
29 November - 1 December 2019, Dhaka, Bangladesh

Signature Verification Using Convolutional Neural Network

Shayekh Mohiuddin Ahmed Navid
Dept. of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
shayekh.navi@northsouth.edu

Shamima Haque Priya
Dept. of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
Shamima.priya@northsouth.edu

Nabiul Hoque Khandakar
Dept. of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
Nabiul.khandakar@northsouth.edu

Zannatul Ferdous
Dept. of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
zannatul@ieee.org

Akm Bahalul Haque
Dept. of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
*Bahalul.haque@northsouth.edu

***Abstract*— Signatures are widely used to validate the authentication of an individual. A robust method is still awaited that can correctly certify the authenticity of a signature. The proposed solution provided in this paper is going to help individuals to distinguish signatures for determining whether a signature is forged or genuine. In our system, we aimed to automate the process of signature verification using Convolutional Neural Networks. Our model is constructed on top of a pre-trained Convolutional Neural Network called the VGG-19. We evaluated our model on widely accredited signature datasets with a multitude of genuine signature samples sourced from ICDAR[3], CEDAR[1] and Kaggle[2]; achieving accuracies of 100%, 88%, and 94.44% respectively. Our analysis shows that our proposed model can classify the signature if they do not closely resemble the genuine signature.**

Keywords*— Convolutional neural network (CNN); Signature verification; Computer Vision, Fine-tuning, Classification.

I. INTRODUCTION

Signatures have been used for decades to verify the identity of an individual. In today's society, signatures are used as a formal and crucial step in an agreement thus the validity of the signature is questioned when any legal issue arises. Validating such a task involves a different number of difficulties. Supposedly an individual's signature may alter daily, or their signature might completely change over a while and in some cases, forgeries attempted are indistinguishable from the original signatures.). Our system aims to determine whether any given signature is valid or forged. To test this hypothesis we used different convolution of neural networks (CNNs)[6] to attain signature verification using machine learning algorithms. In recent times, Deep Learning techniques have been demonstrated to be quite effective in carrying out tasks of this sort. CNNs[6] in particular are widely accredited in the field of machine learning to yield a commendable performance in classifying batches of images. In a very crude sense, it accomplishes the assignment by passing on an image onto a network of layers (where the convolutions take place) that learns the nuances and extracts important features. To tie things together, relevant images are passed through a trained model which then authenticates the validity of the signatures, in turn, minimizing the likelihood of fraud by retaining time and eliminating any kind of human intervention during the verification process that could cause errors.

In this paper, we proposed a model that could eventually lead to the identification of an altered signature using CNNs[6] along with the use of a pre-trained VGG-19[7] model and some fully connected layers to classify a given image. We achieved this by merging various datasets from different sources to build a robust model that can detect forged signatures.

II. BACKGROUND

A. Machine Learning for Signature Verification

In this work on Machine Learning for Signature Verification [8] the author proposed implementation of how simple models could be used to identify and distinguish between forged and real signatures. Because each individual develops distinctive pen motion practices that serve to depict his or her signature, signatures are used for identification. Therefore, two algorithms are at the core of any automatic signature verification scheme: one for extracting characteristics and the other for determining the similarities of two characteristics based on characteristics. Characteristics are components capturing the uniqueness. A completely distinct set of characteristics is used by automatic signature verification techniques outlined in the literature. Wavelet descriptors, projection distribution functions, expanded shadow code, and geometric features are types of characteristics used for signature verification.

B. SigNet: Convolutional Siamese Network for Writer Independent One Signature Verification

Signature verification [4] is one of biometrics and documentation's most difficult tasks. Nonetheless, like other verification issues, it requires the model to pick up very critical and minute details between genuine and forged signatures, because some particular types of deformation could only vary from the actual signature. In writer-independent situations, this verification job is even difficult, which is undeniably fiscal in realistic instances.

In this report, the scholars presented a demonstration with a convolutional Siamese organize for an offline writer-

978-1-7281-5852-5/19/$31.00 ©2019 IEEE

independent signature confirmation task. Siamese systems are twin systems with shared weights that can be prepared to memorize an identical picture highlight. Typically done by uncovering the organized pattern to a match of comparative and divergent perceptions and minimizing the Euclidean separate between comparative sets whereas at the same time maximizing the remove between disparate sets. Since clump preparing, a neural organize regularly requires pictures of the same measurements, but the signature pictures they consider run from 153* 258 to 819* 1137 in particular measurements. Utilizing bilinear insertion, they resize all pictures to a set estimate of 155* 220. They alter the pictures in this way so that the foundation pixels have values. In expansion, they normalize each picture by isolating the pixel values with the standard deviation of the picture pixel values in a dataset.

Siamese neural network is a network engineering lesson that regularly has two indistinguishable sub-networks. With the same parameters and shared weights, the twin CNNs [6] have the same setup. The upgrading parameter is reflected in both sub-networks. This system has been successfully utilized in pitifully observed metric learning and confronts confirmation in arrange to decrease dimensionality. A misfortune work at the best, which calculates a similitude metric including the Euclidean separate between the include representations on each side of the Siamese arrange, is joined to these subsystems.

C. Handwritten Forgery Detection Using Convolutional Neural Networks

In this paper, the author[5] proposes a method that includes CNNs[6] to differentiate between signatures. Here the author stated that the type of forgery discovered in this paper was Simulation Forgery, Random Forgery, Tracing Forgery and Optical Transfer Forgery.

The method that he proposed includes steps such as preprocessing of the data, converting it to grayscale, noise addition with salt and pepper noise and then removal of those noises, conversion of the image from grayscale to bitmap and finally resizing the dataset. The author went on to attain 98.23% accuracy using ConvNets of 3 layers of size 32,64 and 128 respectively along with Max-Pooling layers. Lastly, he passed it onto a fully connected layer. The train: test split of the dataset he used to attain this accuracy was 8:2.

III. EXPERIMENTAL SETUP

A. The Proposed Model

With the purpose of this study, the design principles of the proposed method are given below.

i. Provide better precision in distinguishing forged signatures.
ii. Reduce the decrease in precision due to the time difference.

When a particular image of a signature is subjected to our model, the model uses the weights and biases that learned during the training phase to conclude of a signature is forged or real.

For our model, we have used a pre-trained model with a VGG-19[7] architecture, the VGG-19[7], is connected to the 256 CNN[6] layer, then to the 128 CNN layer and then to 64 CNN layer. The CNN-64 layer was then connected to a fully connected layer of 512 outputs- then to 256 outputs- then to 128 output-then to a dense layer of 2 outputs. The parameters of each of the layers are given below.

Table 1. DESCRIPTION OF MODEL LAYERS

Model Layers	Number of Filters	Kernel Size	Strides	Activation
VGG-19(Without Final Layer)				
ConvNet	256	2	1	Relu
ConvNet	128	3	1	Relu
ConvNet	64	3	1	Relu
Dense	512			Relu
Dense	256			Relu
Dense	128			Sigmoid
Dense	2			Softmax

Figure 1. Summary of the Model

Adding more layers, was unsuccessful as it decreased the accuracy. Stride signifies the distance between the application of filters for the convolution and pooling operations. The last layer with 2 neurons, has a softmax activation function. We used Rectified Linear Units (ReLU) throughout the network as the activation function for the production of all convolutional and fully linked layers, except for 128 dense layer. The Rectified Linear Unit is often used in deep learning designs for the activation function. If it gets any adverse input, the feature returns 0 but returns that value for any beneficial value x. So it can be written as

$$F(x) := max(0,x) \qquad (1)$$

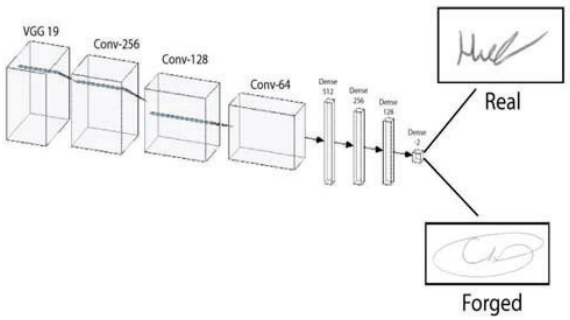

Figure 2. Proposed Model

B. VGG-19 Architecture

For this specific use case, our pre-trained model of choice was the VGG-19[7] which is a pre-trained neural

network model for transfer learning made available in popular neural network APIs such as Keras. The VGG network architecture is the brainchild of the Visual Geometry Group, a team from the University of Oxford and made its inception by securing the runner-up spot at the ILSVRC 2014 competition. The model consists of 19 layers and can classify up to 1000 classes. The breakthrough it made over its predecessors is by opting for multiple 3X3 sized small filters over some much larger 11X11 kernel filters, not to mention that a 3X3 sized filter helps with retaining finer details of the image. The stacking of these smaller filters yields better results as it increases the depth of the network and allows the network to pick up more intricate features of the input image at a lower cost for a given receptive field. This sort of stack arrangement of small kernel filters into blocks/modules are multiplied and are found throughout the network, in turn, accounting for over some 138 million parameters during its operation. It is for the aforementioned advantages, a network as such has been our choice when it comes to a task such as signature verification where extracting minute details and patterns in a given signature becomes essential while distinguishing between a forged and a real signature.

Before we could use this network, the top classification layer was dropped and replaced with a dense layer of 2 in order to use it for our purpose. No other adjustments were made to the base architecture of the VGG-19.

C. Dataset Preparation and Augmentation

In the case of Forged Signature Verification, we looked into multiple datasets. One of the datasets was found in Kaggle[2], another one was taken from ICDAR[3] competition seemed to be good enough for our use. Other than that we used one more dataset[1] from an open-source and which proved to be quiet useful. All the images were on RGB color space and the format was.PNG. All the datasets are combined to form the dataset that we will be using in our classification problem. The overall dataset will have 5380 Images with which we will be evaluating our performance.

To preprocess the images, the images of the signatures were de-noised using the OpenCV library. The number of data samples present in the datasets is presented below:
ICDAR[3]: Total number of signatures: 2020 signatures.
Kaggle Dataset[2]: Total number of signatures: 720 signatures.
CEDAR[1]: Total number of signatures: 2640 signature.
For our experiment, we split all the datasets into an 8:2 ratio for training and testing respectively. For training the model all the signatures from the training portion of the dataset were merged.

For testing our model the test data were kept separately to validate our model on unseen data for each of the datasets.

Table 2
DATA AUGMENTATION

Data	Augmentation
Train	Shear = 0.2 ; Rescale = 1/255 ; Zoom = 0.2 and Rotation = 40 Degree
Test	Rescale = 1/255

Figure 3. Data Augmentation Strategies

D. Transfer Learning and Model Training

Eventually, we used VGG-19[7] architecture to train our model in two ways. VGG-19[7] is a large neural network with a huge number of learnable parameters. Preparing such an enormous network from start successfully needs a big dataset and access to critical computational assets. In any case, by leveraging likenesses between particular picture datasets, this issue can be maintained a strategic distance from. Particularly, the low-level highlights learned by the primary few layers of a network will be roughly the same for most sensible picture classification errands notwithstanding of the dataset. This infers we will initialize our neural organize with parameter values learned from another dataset and anticipate the values for the network's, to begin with, layers to operate well without preparing. This is known as transfer learning.

The layers in the VGG-19[7] were frozen and the weights of the layers were distributed on the rest of the network for the layers connecting from Conv256 to Conv128 layers and dense128 to dense2 layers. The model was first trained for 10 epochs in this phase.

Later, weights were unfrozen from the pre-trained model and then trained from one end to the other. This time it was used as the optimizer with a learning pace of 0.0001 and momentum of 0.9, stochastic gradient descent. The model has been taught for 20 more epochs in this stage.

Throughout the whole training, binary cross-entropy was used as the loss function. The loss function is defined as such:

$$H_y0(y) := -1/N \, X(y_i^0 \log(y_i) + (1 - y_i^0)\log(1 - y_i)) \qquad (2)$$

where y is the label, y_i is the predicted probability of the point of being a certain class for all N points.

Classification of Forged and Genuine Signatures has to deal with every minute change in details of patterns and vibrations. For our proposed method, we tried different methods of tackling this problem. For the first 10 Iterations, the weights from the base layer of the VGG-19[7], were used. For the next 10 iterations, the layers of the VGG-19[7] were unfrozen and the model was trained again after which the accuracy increased significantly. The model was trained for more than 30 epochs to achieve the best results.

IV. RESULTS AND ANALYSIS

After the completion of the training of our model, it was tested with real and forged signatures from the three datasets that were kept separate for a fair evaluation of the performance of the model. The model performed very well on ICDAR[3] and the dataset from Kaggle[2] and gave us really good results as compared to other state of the art methods; on the CEDAR[1] dataset we achieved an adequate result.

Table 3. ICDAR DATASET RESULTS

Metrics	ICDAR -Dataset		
		Forged	Real
Test Accuracy	100%		
Test Loss	0.0102		
Precision		100%	100%
Recall		100%	100%
F-1 Score		100%	100%

Figure 4: Results for ICDAR Dataset

Table 4. RESULTS FOR ICDAR DATASET

Metrics	Kaggle Dataset		
		Forged	Real
Test Accuracy	94%		
Test Loss	0.1336		
Precision		97%	91%
Recall		90%	97%
F-1 Score		94%	94%

Figure 5: Results for Kaggle Dataset

TABLE 5. RESULTS FOR KAGGLE DATASET

Metrics	Overall Model
Train Accuracy	99.40%
Train Loss	0.0162

Figure 6. Results for the overall model

Our model beat the state of the art in ICDAR[3] dataset. We can assume that we beat the state of the art for the Kaggle Dataset[2] as well since the state of the art accuracy was not found. We achieved adequate results on CEDAR[1] dataset.

The real signatures that were not classified correctly were observed to be fake signatures with the naked eye and most of the forged signatures that were not classified correctly seem to be of an expert level of forgery, hence the outcome.

The testing of the model was carried out using test data from the split of 8:2 on each of the datasets. To make the testing fair, we tested the model with signatures that our model has never seen before. We achieved a staggering accuracy of 100% on ICDAR [3],94.44% on the Kaggle dataset [2] and 88% on CEDAR [1].

V. CONCLUSION

We have experimented with several versions in signature verification tasks where we have been able to demonstrate that convolutional neural networks do an outstanding job at verifying signatures by allowing access to examples of real and forged signatures of the same individuals whose signatures are seen at test moment during the practice. We then performed an experiment in which we tested our network on the signatures that were not seen at all during the training. In essence, for a circumstance where one might be willing to compare a legitimate signature with a plausibly-forged one by the same signatory, our devised architecture yields a set of adequate results with a strong potential for subsequent growth in the field of forged signature detection. Our model can be used by deploying it into portable platforms such as smartphones and raspberry pies for inferencing if a particular image of a signature is forged or real.

Figure 7. Real Signatures Classified as Forged

Figure 8. Forged Signatures Classified as Real

When talking about our drawbacks or bottlenecks we faced during experimentation, computing resource limitations became a real hurdle as it impeded us from possibly training the data on a much bigger network and for considerably more number of epochs which technically ought to have a positive effect on our results. In addition, our inability to work with a bigger dataset (which in our case was restricted to several thousand samples) with more signature samples per person is something that hinders the accuracy of our model without question, given that we were successful in congregating authentic open-source signature data from the internet which is actually a very difficult task to carry out. To be more specific, despite our technique on the model being promising in the literature on signature verification, it directed very satisfiable results on the Kaggle [2] & the

ICDAR [3] datasets, leaving out CEDAR [1] to be the dataset on which the model performed otherwise. In our pursuit of achieving superior results in the near future, we would need to acquire such resources to further enhance the performance of the aforementioned architecture on not only the CEDAR [1], but for any set of signatures being put to the test.

REFERENCES

[1] CEDAR Signature Verification", Cedar.buffalo.edu, 2019. [Online]. URL: https://cedar.buffalo.edu/signature/.

[2] URL: https://www.kaggle.com/divyanshrai/ Handwritten-signatures.

[3] "Datasets - TC11", Iapr-tc11.org, 2019. [Online]. Available: http://www.iapr-tc11.org/mediawiki/index.php?title= Datasets..

[4] Sounak Dey, Anjan Dutta, J. Ignacio Toledo, Suman K. Ghosh, JosepLlad´os, and Umapada Pal. Signet: Convolutional Siamese network for writer independent. Offline signature verification. CoRR, abs/1707.02131, 2017. URL http://arxiv.org/abs/1707.02131.

[5] S Jerome Gideon, Anurag Kandulna, Aron Abhishek Kujur, A Diana, and KumudhaRaimond. Handwritten signature forgery detection using convolutional neural networks. *Procedia Computer Science*, 143:978 – 987, 2018. ISSN 1877-0509. doi: https://doi.org/10.1016/j. procs.2018.10.336.URLhttp://www.sciencedirect.com/ science/article/pii/S1877050918320301

[6] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks.*CoRR*,abs/1511.08458,2015. URL http://arxiv.org/abs/1511.08458.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014

[8] Harish Srinivasan, Sargur N. Srihari, and Matthew J. Beal. Machine learning for signature verification. In Prem K. Kalra and Shmuel Peleg, editors, *Computer Vision, Graphics and Image Processing*, pages 761–775, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-54068302-5.