Signature Verification using Convolutional Siamese Neural Networks

Kevin Ayuque, Poormina Haridas
New York University
New York City, NY 10012, USA
{kja306, ph1391}@nyu.edu

Abstract

In this paper we propose a new system for offline signature verification. In a real world scenario, it is difficult to obtain large data samples of original vs fake samples for any single signature and then use them to train a model for classification. It is therefore even more difficult to make a generic system that can differentiate original vs fake signatures for multiple candidate signatures. We propose a One-Shot learning method to tackle the issue using Convolutional Siamese neural networks.

1. Introduction

1.1. Motivation

A signature is a handwritten symbol that depicts an individual name, it is used as proof of identity to demonstrate intent and consent. While innocuous, it is used on cheques, contracts, diplomas, credit cards receipts, etc. Because of it's nature it is also susceptible to forgery as anyone can try to fake a signature if it is known a priori how a real signature is supposed to look. We explore how the problem of signature verification has been explored in the past, what other ideas could be implemented to improve any past approach and implement it to see how it compares.

Depending on the input format, signature verification can be of two types: (1) online and (2) offline. Online signatures are captured using an electronic writing pad together with a stylus. This can record a sequence of coordinates of the electronic pen tip while signing. Apart from the writing coordinates of the signature, these devices may also be capable of fetching the writing speed, pressure, etc., as additional information, which are used in the online verification process. On the other hand, offline signatures are usually captured by a scanner which basically produces two dimensional signature images.

While significant work has been done on online data sets, on the offline data sets good results have been achieved only using standard image processing techniques. We aim to apply and test the results of applying Computer vision methods of Convolutional Neural Networks(CNNs) to the offline data sets in this project.

1.2. Related Work

The task of signature verification has been extensively studied in the past decades [5] [7]. Some studies also include classification techniques on offline data in addition to verification [1]. Some of the most popular datasets for our task include:

- SVC 2004: First international Signature Verification Competition
- BSEC 2009: BioSecure Signature Evaluation Campaign
- SigComp 2009
- SigComp 2011
- ICDAR2015: Competition on Signature Verification and Writer Identification for On- and Off-line Skilled Forgeries
- GPDS960 Signature database

For SigComp 2011, the dataset that we use, some methods previously used are:

- SVM (Support Vector Machine)
- SVM with RANSAC (Random Sample Consensus) to remove geometry-inconsistent matches
- Edge-based directional probability distribution features and grapheme features.

From previous work, we can conclude that:

- Different systems perform better for different tasks. For example, A model that performs well on Chinese signature verification does not perform as well on Dutch signature verification.

- CNNs have not been applied to this use-case.
- Offline signature verification is significantly harder, therefore we focus our effort on the Dutch offline training and test data.

2. Approach

We use a convolutional Siamese neural network to implement a one-shot learning method over the offline signatures. A siamese network learns the difference between a pair of samples and embeds samples that are similar, closer together and, pushes samples that are different further away. A method based on MAML [3] was considered, but dropped due to complexity. We will be using the Facial Similarity code by Harshvardhan Gupta as a starting point.

2.1. Siamese Neural Networks

Siamese neural networks [4] are an interesting variation of neural networks. They do not answer the classic classification question of which class does X belong to? Instead, they answer the question, are X and Y similar? Siamese networks are trained by initializing two network towers where weights of the sister networks are held equal. Pairwise comparisons are then conducted by feeding inputs through each tower as shown in Figure 2. The networks are trained to determine whether or not the two objects it is presented with are the same or different by minimizing the distance between similar inputs and maximizing the distance between dissimilar ones. Initially they were used for online signature verification.

2.2. Data

We use data published by ICDAR 2011 Signature Verification Competition (SigComp2011) [6]. The collection contains offline and online signature samples. Signatures were either genuine: written by the reference writer, or a simulation: simulated by a writer different to the reference writer. We used the the offline data sets for Dutch signatures that consisted of PNG images, scanned at 400 dpi, RGB color. The data was divided in training and test sets having different naming conventions. Details about the number of contributing authentic authors, forgers, number of authentic reference signatures and forgeries for the Dutch signatures is provided in Table 1.

Figure 1. Sample images from SigComp 2011 dataset. Top row: Original samples, Bottom row : Forgeries.

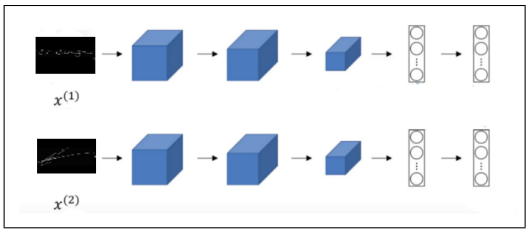

Figure 2. Siamese network for offline signature verification

	Training			Testing			
Data Type	A	G	F	A	GR	GQ	F
Offline	10	240	123	54	648	648	638

Table 1. NUMBER OF AUTHORS (A) AND NUMBER OF GENUINE (G) (REFERENCE (GR) AND QUESTIONED (GQ)) AND FORGED (F) SIGNATURES IN THE DUTCH DATA SET

2.3. Pre-processing

Batch training a neural network typically needs images of same sizes but the our dataset had varying sizes for the different signature images. We resized all the images to a fixed size $200X200$ while keeping the aspect ratio intact. We conducted experiments with inverted images (background pixels = 0) and non-inverted images separately. We did not conduct any further pre-processing to keep the aspect ratio intact and gain as much information as we could, from the images. The images are then paired to be passed into the network. Each original signature of a candidate sample is paired with other original signatures of the same candidate/class and each of it's fake samples. This takes the total f our data points to 8410 pairs of images.

2.4. Model

Our model consists of a Convolutional Siamese Neural Network. Instead of building a pair of networks to get the difference between a pair of inputs (as shown in Figure 1) we use only one model which is fed images in succession. This works as the weights are constrained to be identical for both networks. Then we calculate the loss value using both the images, and then back propagate. This saves a lot of memory without affecting the accuracy. The Adam optimizer with learning rate 0.0005 is used to update the weights.

The loss function used is Contrastive loss [2] as it aims to differentiate and not classify, like most other networks. Contrastive loss can be defined as:-

$$(1-Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}max0, m - D_w)^2$$

where D_w is defined as the euclidean distance between the outputs of the sister siamese networks. Euclidean distance is defined as:-

$$\sqrt{G_w(X_1) - G_w(X_2)^2}$$

where G_w is the output of one of the sister networks. X_1 and X_2 is the input data pair.

Intuitively, the contrastive loss function evaluates how well the network is distinguishing a given pair of images.

The network architecture is relatively simple with 3 convolutional layers of sizes $(4X3X3), (8X3X3), (8X3X3), (8X200X200, 500)$ and 2 linear layers of sizes $(500, 500), (500, 5)$.

Throughout the network, we use Rectified Linear Units (ReLU) as the activation function to the output of all the convolutional and fully connected layers and Batch normalization on all the convolutional layers. A Dropout with rate = 0.5 is applied on the last linear layer. Our entire framework is built from scratch using the Pytorch library. The training and testing was run on Prince clusters and Colab GPUs.

3. Experiments

In order to evaluate our network, we used the ICDAR 2011 [6] dataset with a few variations in our pre-processing techniques and network configurations. We inverted our images and changed the dimensions of the network to make it deeper.

3.1. Performance Evaluation

A threshold d is used on the distance measure $D(xi, xj)$ output by the network to decide whether the signature pair (i, j) belongs to similar or dissimilar classes. We denote the signature pairs (i, j) with the same identity as S_{real}, whereas all pairs of different identities as S_{fake}. Then, we can define the set of all true positives (TP) at d as:-

$TP(d) = (i, j)S_{real}, withD(x_i, x_j)d$

The set of all false rejections (FR) at d can be defined as:-

$FR(d) = (i, j)$ is predicted as S_{fake}, when actually $(i, j)S_{real}$ with $D(x_i, x_j) > d$

Finally the set of all false positives (FP) at d can be defined as:-

$FP(d) = (i, j)$ is predicted as S_{real}, when it actually $(i, j)S_{fake}$ with $D(x_i, x_j)d$

Then the true positive rate $TPR(d)$, the true negative rate $TNR(d)$ and the false positive rate for a given signature, distance d are then defined as:-

$TPR(d) = \frac{|TP(d)|}{|S_{real}|}$,

$FRR(d) = \frac{|FR(d)|}{|(TP+FN)|}$,

Threshold	0.22	0.24	0.26	0.28	0.3	0.32	0.34
Accuracy	0.67	0.66	0.65	0.64	0.63	0.62	0.61

Table 2. Results with White background

$FPR(d) = \frac{|FP(d)|}{|S_{fake}|}$

The final accuracy is computed after generating RoC curves and measuring the accuracy at the point where $FPR = FRR$ as:-

$Accuracy = \frac{|(TP+TN)|}{|len(Testdata)|}$

Here D is a set of predefined thresholds as can be seen in Tables 2 and 3.

3.2. Results

Figure 3 shows our training loss over the epochs, which seems favourable and appears to converge very fast. Figure 4 shows a few sample results of test pairs which look promising.

Tables 2 and 3 show the accuracy for the various thresholds for a white and black background respectively. We notice that the white background gives slightly better results and therefore here the images could be sharpened for better performance.

Figures 5 and 6 show the ROC curves for a white and black background respectively (FPR vs TPR and for FRR vs FAR(=FPR). For the ROC curves, we are using the thresholds listed in Tables 2 and 3 respectively. We do not use all the values from 0 to 1 because it would be computationally expensive (running a test for each threshold takes approximately 30 minutes). The listed thresholds were were carefully chosen after testing on a small batch with good results.

Table 4 compares our model to the best and worst performing models in SigComp 2011 [5].

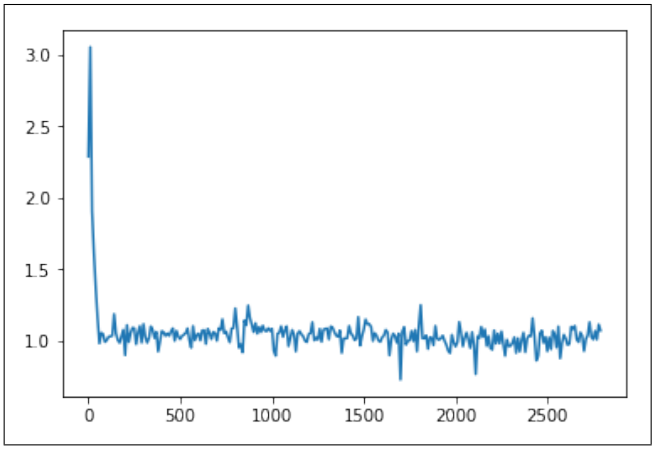

Figure 3. Training loss for 20 epochs.

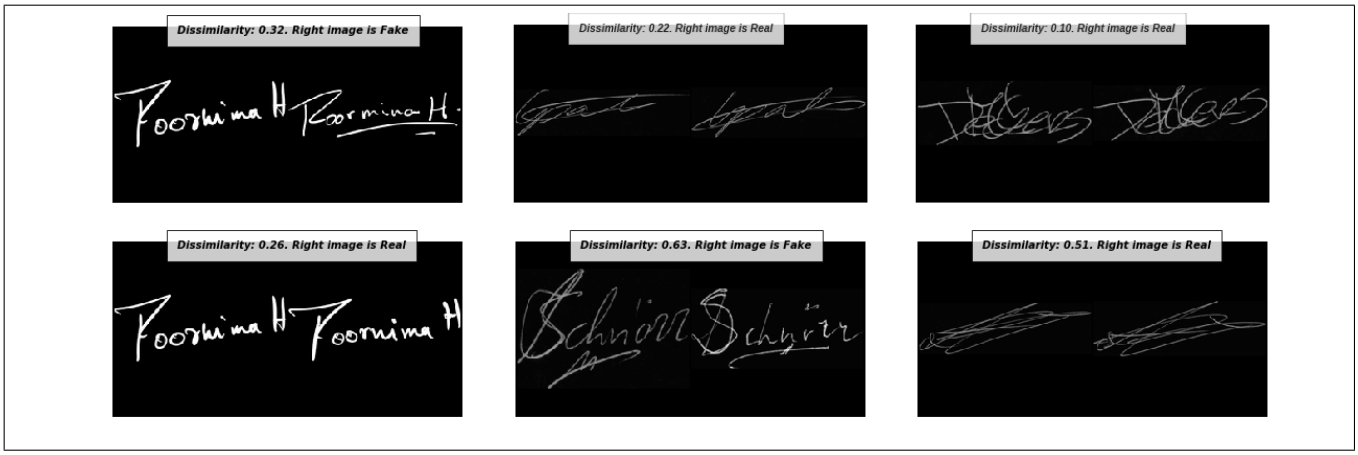

Figure 4. Test samples with the calculated dissimilarity and real label. (The first column is not a part of the actual testing datasets)

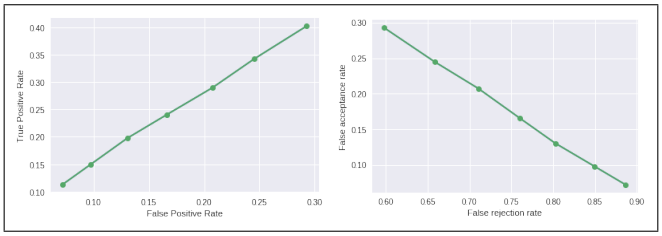

Figure 5. ROC curves for training with White background.

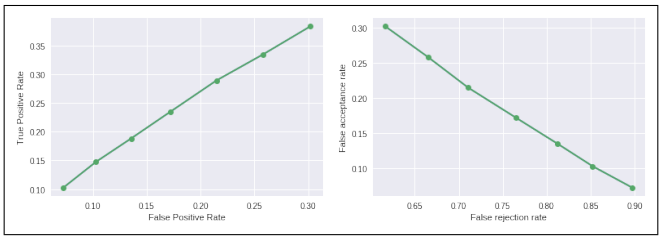

Figure 6. ROC curves for training with Black background.

Threshold	0.22	0.24	0.26	0.28	0.3	0.32	0.34
Accuracy	0.66	0.66	0.65	0.64	0.62	0.61	0.59

Table 3. Results with black background

Metric	Our	Best	Worst
Accuracy	0.67	0.97	0.71
TPR	0.11	–	–
FPR / FAR	0.072	0.219	0.287
FNR / FRR	0.89	0.247	0.291

Table 4. Results for the Dutch Offline Dataset

4. Discussion

In this project we applied the concept of CNNs to offline data sets of signatures. Although our accuracy is lower, it should be noted that none of the SigComp models have used a CNN based approach. The approach and the idea of using Siamese network for this task seems natural and promising. Following are the future steps that can be taken to improve the results:

1. During pre-processing stage, the images can be made sharper and affine transformation can be used to appropriately position signatures captured in different angles. This will significantly help bring down the false negative rates.

2. Increase the depth of the layers which will increase the complexity of the model but will help learn finer features of each signature. This will help the overall accuracy.

The improved methods can be applied on other data sets to see the quality of the results on different languages as well. This will make the model more generic and therefore more useful.

References

[1] M. Adamski and K. Saeed. Online signature classification and its verification system. In *2008 7th Computer Information Systems and Industrial Management Applications*, pages 189–194.

[2] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

[3] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks.

[4] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. page 8.

[5] F. Leclerc and R. Plamondon. Automatic signature verification: The state of the art - 1989-1993. *IJPRAI*, 8:643–660, 06 1994.

[6] M. Liwicki, M. I. Malik, C. E. Van Den Heuvel, X. Chen, C. Berger, R. Stoel, M. Blumenstein, and B. Found. Signature verification competition for online and offline skilled forgeries (sigcomp2011). In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1480–1484. IEEE, 2011.

[7] R. Plamondon and G. Lorette. Automatic signature verification and writer identificationthe state of the art. *Pattern recognition*, 22(2):107–131, 1989.