Handwriting Generation and Fraud Detection

December 16, 2020

1 Rishabh Bhonsle (17217) (EECS)

2 Prithvi Poddar (17191) (EECS)

3 Sourav Yadav (17285) (EECS)

report_2

December 16, 2020

1 Report for Fraud Detection

The aim of this project is to classify pairs of signature images as authentic or forged.

2 Overview

During this new tech era, Handwritten signatures still hold the utmost level of authenticity in our personal and legal lifes. People sign cheques, authorize documents and contracts, validate credit card transactions and verify many other things with their signatures. A signature can only be accepted if it is from the intended person. The probability of two signatures drawn by the same person being the exact same is very less. So, detecting a forgery becomes a challenging task. In this project, we try to build a classifier for mechanized detection of forged signatures. The work presented is still premature(due to time restraints) and requires subsequent thorough remodelling.

3 Data

To our luck we found an already cleaned and well ordered dataset for our project on Kaggle Fraud Detection Dataset. All the data are extracted from ICDAR 2011 Signature Dataset and organized perfectly for user usage.

The dataset contains the signatures of dutch users both genuine and fraud.

In the dataset the directory number says the name of the user and it's classified into two: Genuine with its own user number and fraud with the user number + "_forg".

The dataset is divided as follows :

test(directory) - Constitutes all the test images using the nomenclature defined above. train(director) - Contains all the images with nomenclature defined above, used for testing purposes. train_data.csv - First column represents the original signature by a user, the next column contains either one of the original signatures or forged one of the same user. Ending column is a binary with 1 if both signatures are original and 0 if the second one is forged. test_data.csv - Same format as train_data.csv. Used for testing purposes.

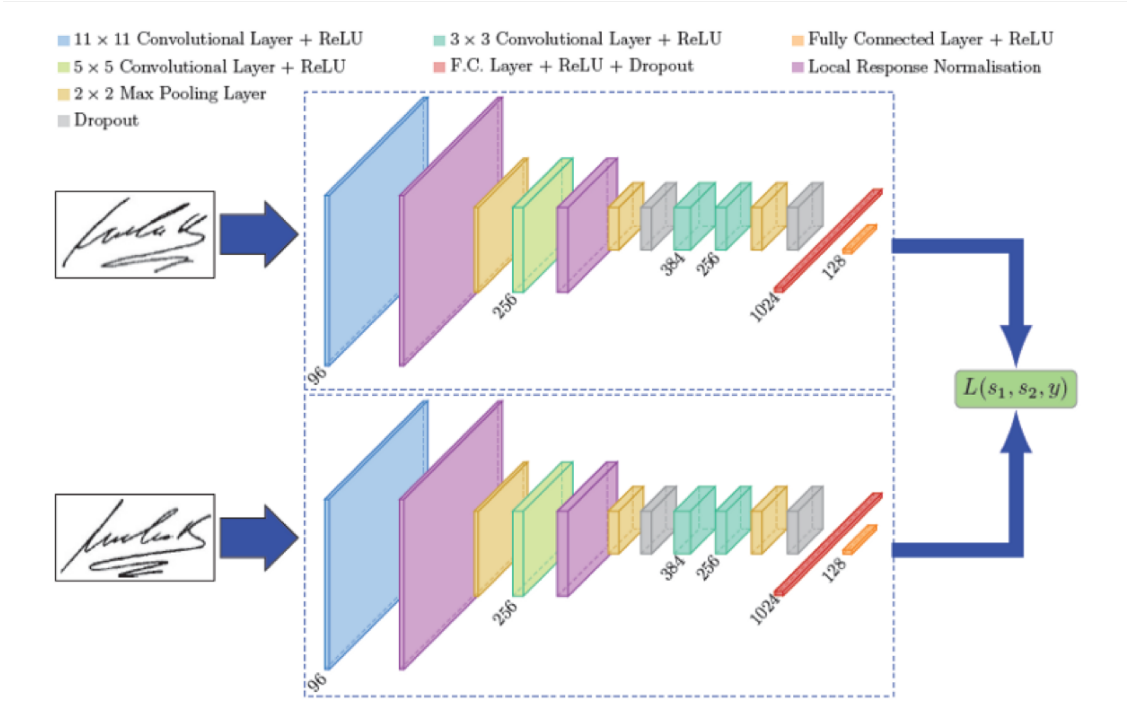
The testing set is approximately one fourth the size of the training set. The respective shapes are as follow: Training Set - (23205, 3) Testing Set - (5747, 3)

4 Approach

Since we're labelling pairs of images as forged or original, it makes sense to use a Siamese Network(sometimes called twin network) with contrastive loss function.

4.1 Siamese Network

In the modern Deep Learning Era, Neural Networks are getting good at almost every task, but these rely on more data to perform well. But for some tasks like Signature Verification we don't have millions of images in our arsenal, to solve these kinds of tasks we have a new type of neural networks Siamese Network. They learn from very little data, the ability to learn from a modest dataset has made them more popular in recent times.



A Siamese Network is a class of neural network architectures that contain two or more identical subnetworks(in this project 2 networks are used). 'Identical' here means, they have the same configuration with the same parameters and weights. Also, parameter updating is mirrored across both the sub-networks. It is used to find the similarity of the inputs by comparing its feature vectors, so these networks are used in many applications.

Traditionally, a neural network learns to predict multiple classes. This poses a problem when we need to add/remove new classes to data. In this case, we have to update the neural networks and retrain it on the whole dataset. SNNs, on the other hand, learn a similarity

function. Thus, we can train it to see if two images are the same. This enables us to classify new classes of data without training the network again.

There can be some downsides to Siamese Networks. Since SNNs involve quadratic pairs to learn from it is generally slower than normal classification. Also, it won't output the probabilities of the prediction, but the distance from each class.

4.2 Contrastive Loss

Loss function here used is Contrastive Loss, it is a popular loss function used highly nowadays. It is distance based as opposed to more conventional error-prediction losses. This loss is used to learn embeddings in which two similar points have a low Euclidean distance and two dissimilar points have large Euclidean distance. Defined as,

$(1 - Y)^{1/2} D_w^2 + (Y)^{1/2} (max(0, m - D_w))^2$

where D_w is just the Euclidean distance.

5 Results

Classification error is around one in every four signature image pairs.