Deep Convolutional Siamese Neural Networks for Handwriting Authorship Verification

William Du
Michael Fang
Margaret Shen

Overview

Determining the **authorship** of a written text is a significant task in the realm of forensics, signature verification, and literary history. There has not yet been a study on the accuracy of forensic handwriting analysis, but the field is surrounded by much skepticism. In 1993, the **siamese neural network** was developed to tackle the signature verification problem. Since then, there have been **no published studies** using siamese networks for **general handwriting verification**. There has also been research into handwriting author classification using CNN architectures, resulting in accuracies of up to 97%. However, our inputs may include authors which the training set has not seen, making this a problem beyond the scope of supervised author classification.

Problem Statement

Given two pieces of handwritten text, our goal is to identify with certainty whether or not the same author wrote both pieces of text. We will use **convolutional siamese neural networks** to encode the images into a feature space and determine whether they are written by the same author. Success of an architecture will be measured by accuracy because the data is constructed with a 50/50 split.

Data

The **IAM Handwriting Database** is composed of 1539 pages of scanned text written by 657 different writers. These are segmented into 6685 sentences, 13353 lines, and 115320 words. We use separate writers for our training, validation, and test sets. We construct our datasets to have equal numbers of positive and negative examples.

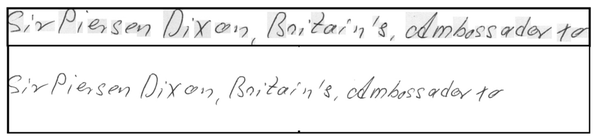


Figure 1. Sample line before and after cropping, padding and thresholding.

Architecture

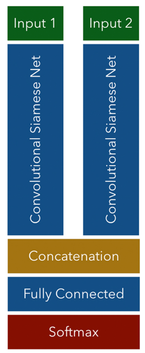


Figure 2: In our network, two inputs are taken and evaluated for a score. The two inputs are fed into Siamese convolutional networks that translate each image into latent encoding space. These latent encodings are then concatenated, along with the square of the differences, and used to produce class probabilities, in this case whether or not the two inputs are duplicates.

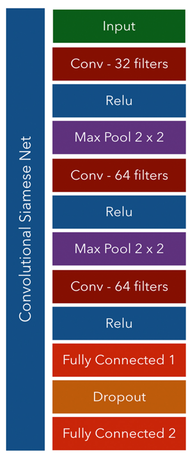


Figure 3: The baseline architecture for each network in the siamese net. When training on words, the convolution filter sizes are all 3 by 3. When training on lines, the convolution filter sizes are 10 by 10, 8 by 8, and 4 by 4 respectively. These expanded filter sizes for lines were mainly used to manage memory usage for larger images. The fully connected layers have 400 and 200 units respectively.

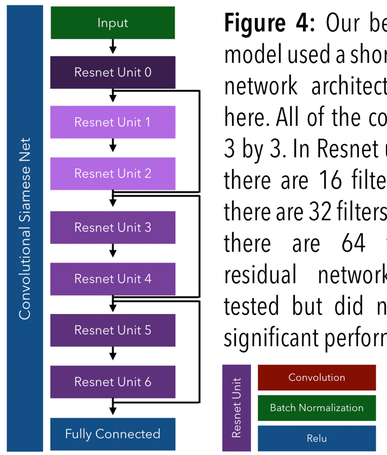


Figure 4: Our best performing model used a shortened residual network architecture displayed here. All of the convolutions are 3 by 3. In Resnet unit 0, 1 and 2 there are 16 filters, in 3 and 4 there are 32 filters and in 5 and 6 there are 64 filters. Larger residual networks were also tested but did not result in a significant performance boost.

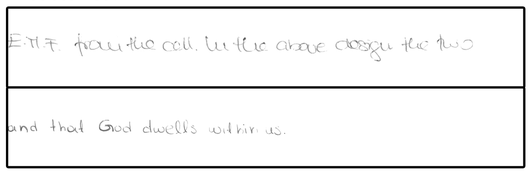
Explorations

Model	Words	Train Size	Val Accuracy
Baseline	Words	10k	60.25%
Baseline	Lines	10k	72.4%
Tiny Resnet 1 Repetition	Lines	10k	91.46%
Medium Resnet 3 Repetition	Lines	10k	89.3%
VGG13	Words	10k	58.0%

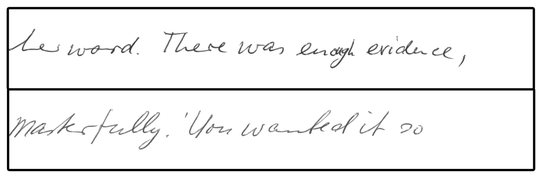
Figure 5. Accuracy for each of the models explored.

Error Analysis

Below are a few representative samples of test inputs that were misclassified.



False Positive: It is difficult to tell, but these two samples are by different authors. The style is very similar for both: loopy, rounded letters. The darkness of the ink is also very similar.



False Negative: These two samples are by the same author. It was probably misclassified because the sample on the left is

Another common type of false negative is a pair of samples by the same author in which one of the samples contains a single word (end of a paragraph) and only **white space** following. This means that the model is probably looking at amount of white space as a feature.

Results

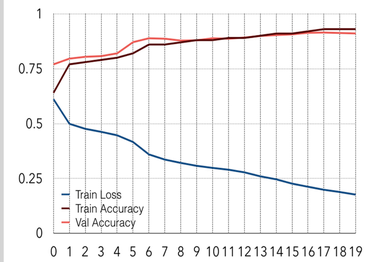


Figure 6. Training loss and accuracy as well as validation accuracy for the best model trained over all epochs. Early stopping was used to pick the best model. The best test set performance was 91.15%.

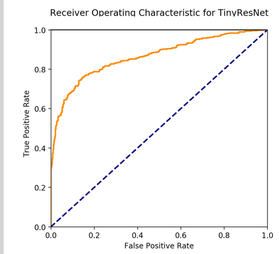


Figure 7. This curve diagrams the relation between specificity and sensitivity of our best model. Because the curve is moderately sloped on top, it demonstrates that we have a lower specificity than sensitivity.

Future Directions

- Using datasets that are more representative of the true distributions
- **Ensembling** our models to break above 95% accuracy
- Trying Inception v3 **GoogLenet** as the encoding network
- **Visualizing** saliency maps and filters to understand what type of features are being used to describe handwriting

Conclusion

We succeeded in training a model that accurately classifies lines 90% of the time. Our **TinyResNet** outperformed our initial baseline model, suggesting that having a large number of layers with small filters works better than having fewer layers with larger kernels. Furthermore, our model performed better on **lines versus words**, perhaps due to the simple fact that each line contains more information about the writer's style than a word. We would have liked to train on entire pages as well, but our model ran into memory limitations.