Recognizing Authors from Scans of Polish Handwritten Text Using the ResNet Architecture

Miłosz Poruba Jakub Machura

June 3, 2024

This document is a report of the project for the Artificial Intelligence course, themed "Artificial neural network classifier trained by backpropagation for polish handwritten text author recognition". The code is available and ready to run in this notebook at kaggle.com.

1 Introduction

The task of recognizing authors from handwritten text involves identifying individual writing styles from physical or digital manuscripts. Our project explores this challenge using the Polish Handwritten Text (HPT) dataset and employs a deep learning approach based on the ResNet50 architecture. This report delves into the methodology, implementation, and findings of using convolutional neural networks (CNNs) for authorship recognition on the HPT dataset.

1.1 The HPT dataset

The HPT dataset comprises scanned handwritten texts from eight authors. Each author's writings are scanned in bitmap format. The dataset includes metadata of word boundaries for each scan.

2 Preprocessing

To evaluate how well the model generalizes, the data is split into train and test sets. The test set is composed by randomly selecting two whole scans from each author. This ensures that the model is supposed to learn how to recognise the author from scans that it has not seen before. Learning to recognise authors from such reduced sets of scans is a demanding task, because after the split there are only a few scans left in some categories.

2.1 Preprocessing approaches

Two primary approaches are considered for segmenting the scanned images:

1. `Word Segmentation` – This approach focuses on isolating individual words from scanned images. Each word is treated as a separate input unit for the neural network. The preprocessing includes:
    - Identifying word boundaries using metadata files that accompany the images.
    - Extracting and resizing words to standard dimensions suitable for input into the CNN.
2. `Tile Segmentation` – In contrast to word segmentation, tile segmentation involves dividing each image into tiles of fixed count per scan, regardless of the word boundaries. This method might capture parts of adjacent words or spaces, providing a different kind of feature set for the network to learn from. We split each scan in square tiles, with a constant number of tiles in the horizontal dimension. Empty tiles (that exceed a 94 percent threshhold of white pixels) are rejected.

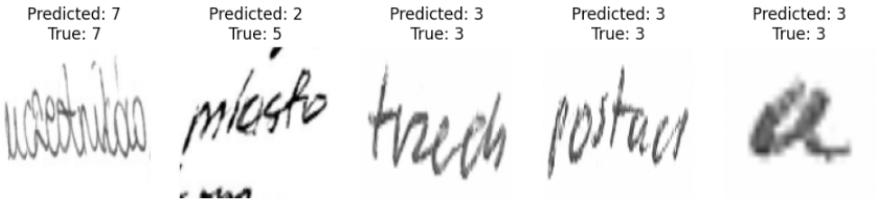

Figure 1: Word method training examples

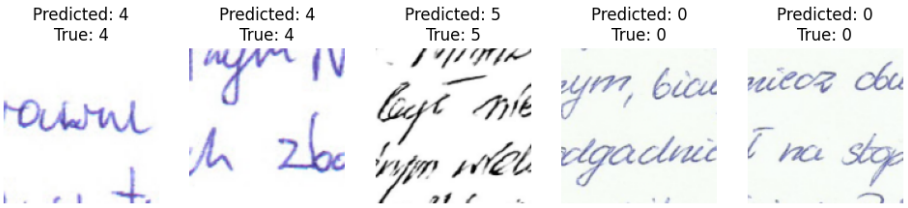

Figure 2: Tile method training examples

Category	Word segmentation	Tile Segmentation
Author 1	1908	662
Author 2	1676	495
Author 3	1754	469
Author 4	1714	214
Author 5	1699	653
Author 6	2190	542
Author 7	1473	443
Author 8	1687	780
Total	14101	4258

Table 1: The number of training examples by category and preprocessing method

3 The model

3.1 Model Architecture

The project replicates the ResNet architecture from this article, a variant of the CNN that is particularly effective in image recognition tasks due to its deep residual learning framework. The model creation is written in a way that allows easy customization of the number of parameter layers. The dataloaders transform each image, regardless of the method (tile or word), to a size of 128x128.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 3: Building block structure from the ResNet article. In this project, we experiment with a 50-layer and 101-layer version.

3.2 Training process

The specific implementation steps include:

`Custom Data Loaders` – used to batch and shuffle the data.

`Training and Validation` – The model is trained using a cross-entropy loss function and Adam optimizer. The training process is monitored for both accuracy and loss metrics to ensure proper learning without overfitting.

`Results and Evaluation` – The effectiveness of the model is evaluated based on its ability to correctly identify the author of a given piece of handwritten text. The results are assessed using accuracy metrics and loss curves to compare the performance across different epochs.

4 Experiments

4.1 Word method experiments

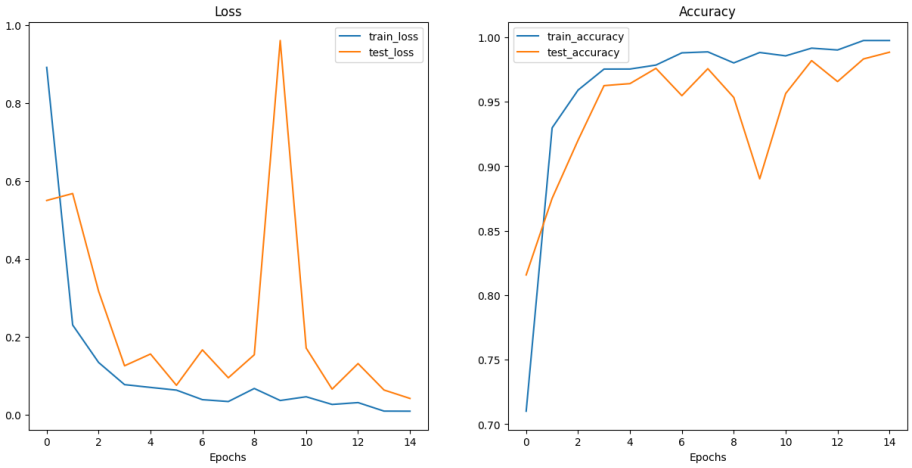

Figure 4: RESNET50, No epochs 15, Batch size 128, Learning rate 5e-5

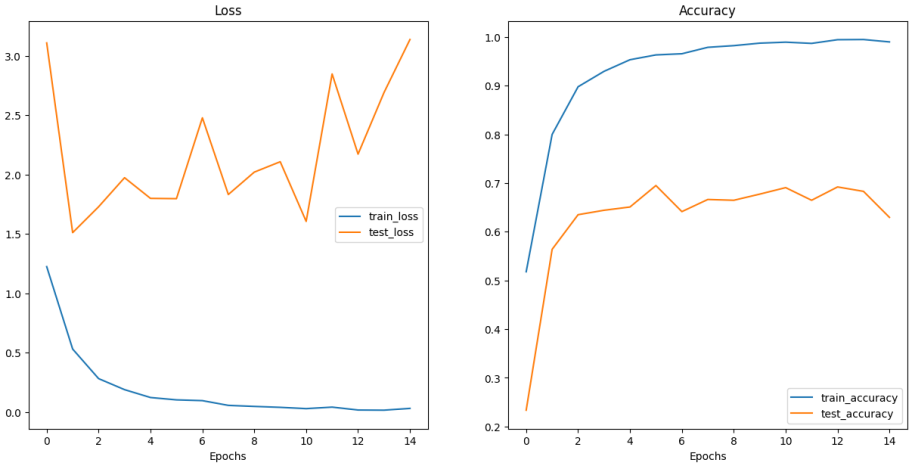

Figure 5: RESNET101, No epochs 15, Batch size 128, Learning rate 1e-5

4.2 Tile method experiments

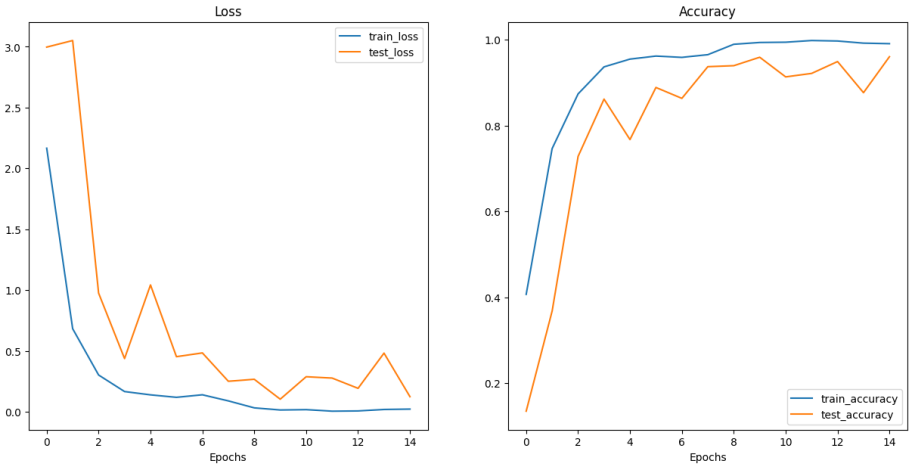

Figure 6: RESNET50, No epochs 15, Batch size 128, Learning rate 5e-5

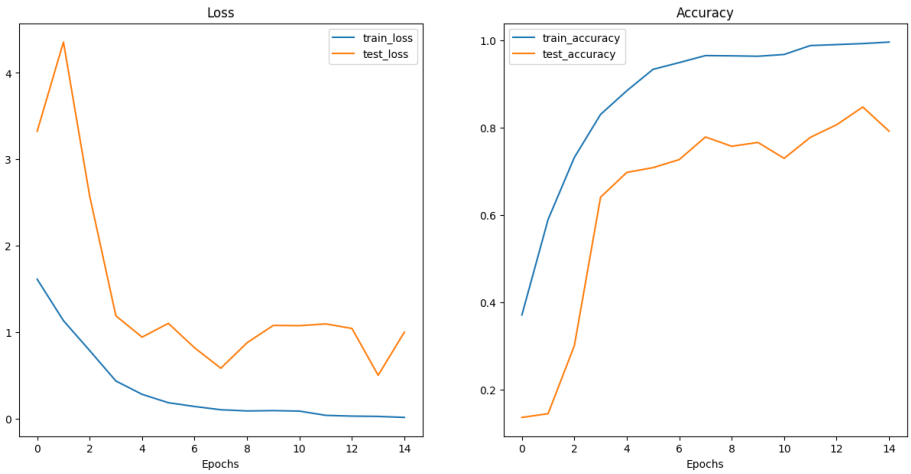

Figure 7: RESNET101, No epochs 15, Batch size 128, Learning rate 1e-5

4.3 Comparison of methods

In terms of architecture, Resnet50 is the preferred choice. It overfits much less than Resnet101, as visible above. To mitigate overfitting in the bigger model, practices such as dropout may be applied.

Both word and tile method provided similar results, with the word method achieving test accuracy as high as 98 percent, and with the tile method reaching 96 percent accuracy on the test set. It worth to notice that training the tile method achieved this with about one third of the number of training examples, compared to the word method. This resulted in training being about three times as fast.

5 Conclusion

The tile method, which is much lighter and achieves similar results to the word method. Another important factor is the lack of necessity to label the word boundaries. This is a demanding task that occurs in the word method.

Both the low amount of training examples and ease of preprocessing may be key when up-scaling the model presented in this report. Therefore, our preferred method of preprocessing the scans is the tile method. Further hyperparameter tuning may result in outperforming the word method even in terms of accuracy.

In conclusion, training a Resnet model on the HPT dataset resulted in achieving high accuracy on both the training and test set (99%+, 96%+), using a simple preprocessing approach. We believe that such model is ready to be scaled up to recognise many more authors on a much larger dataset.