PROJECT REPORT

Writer Identification Project Report

TEAM MEMBERS:

SUBHAN 22K-4316

JUNAID ASIF 22K-4369

MUHAMMAD SADIQ 22K-4303

Writer Identification Project Report

Project Objective

The aim of this project is to identify the author of a handwritten document using deep learning. Specifically, we tackle *handwritten writer identification*, which is the task of determining the author of a handwritten document by comparing it to known handwriting samples 1 . This problem is important in domains like forensics (e.g. linking anonymous letters to suspects) 2 . By training a convolutional neural network (CNN) to classify handwriting images by writer, we attempt to automatically recognize who wrote each sample.

Datasets Used

The dataset comprises handwritten samples from over 2000 different writers. Initially, the dataset was used in its raw form, with some writers having fewer than five images. This imbalance resulted in poor generalization and low model performance.

To address this, we applied a filtering criterion and retained only those writers who had at least 5 images. After this cleaning process, the dataset was reduced to 94 writers. This made the dataset more balanced and allowed for effective training and evaluation.

Model(s) Used

We used a modified version of the ResNet-18 architecture:

i. The first convolution layer was adjusted to support grayscale input.

ii. The default linear classification head was replaced with a Cosine Similarity-based Classifier to improve separation in high-dimensional embedding space.

iii. We applied label smoothing in the loss function to avoid overconfidence and improve generalization.

iv. Data augmentation techniques such as random horizontal flips and slight rotations were used to increase sample diversity and reduce overfitting.

The model was trained using CrossEntropyLoss (with label smoothing) and the Adam optimizer.

Results

We conducted experiments on two versions of the dataset:

Configuration	Writers	Accuracy	Notes
Full Dataset (Unfiltered)	~2000+	31.73%	Many classes with too few images
Filtered Dataset (≥5 images)	94	94.07%	Balanced data, significantly better performance

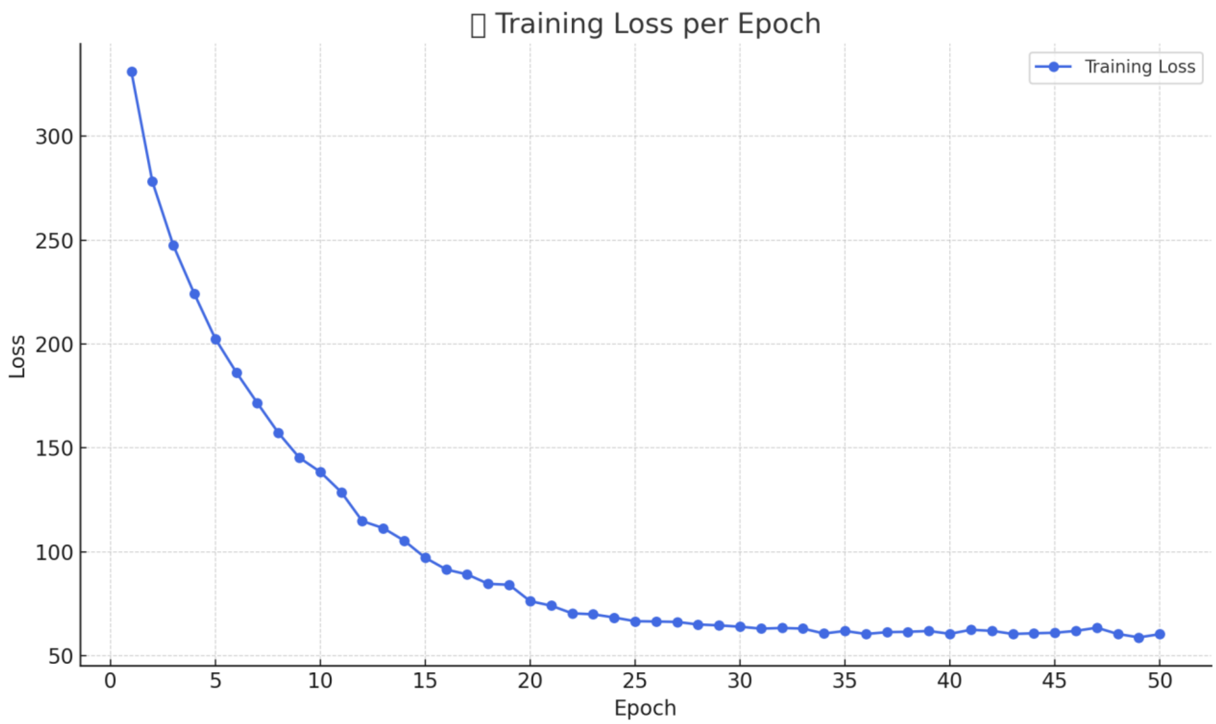
The training loss decreased steadily across 50 epochs for both experiments. However, the filtered dataset resulted in dramatically higher test accuracy due to better per-class representation and generalization.

Graph of loss per epoch for unfiltered 2000+ writers dataset:

Training Loss Table:

Epoch	Training Loss
1.0	330.9981
2.0	278.0735
3.0	247.3800
4.0	224.1148
5.0	202.4938
6.0	186.3491
7.0	171.6436
8.0	157.3038
9.0	145.3284
10.0	138.5014
11.0	128.7659
12.0	114.8371
13.0	111.4543
14.0	105.4068
15.0	97.2125
16.0	91.5564
17.0	89.1958
18.0	84.6637
19.0	84.1990
20.0	76.3286
21.0	74.1647
22.0	70.4612
23.0	70.0201
24.0	68.4598

25.0	66.6794
26.0	66.5441
27.0	66.3502
28.0	65.0952
29.0	64.6833
30.0	64.0297
31.0	63.1153
32.0	63.4145
33.0	63.1432
34.0	60.7572
35.0	61.9915
36.0	60.5402
37.0	61.4156
38.0	61.5982
39.0	61.9691
40.0	60.6198
41.0	62.5544
42.0	62.0561
43.0	60.5431
44.0	60.8718
45.0	61.1190
46.0	62.0247
47.0	63.5571
48.0	60.5348
49.0	58.8951
50.0	60.4404

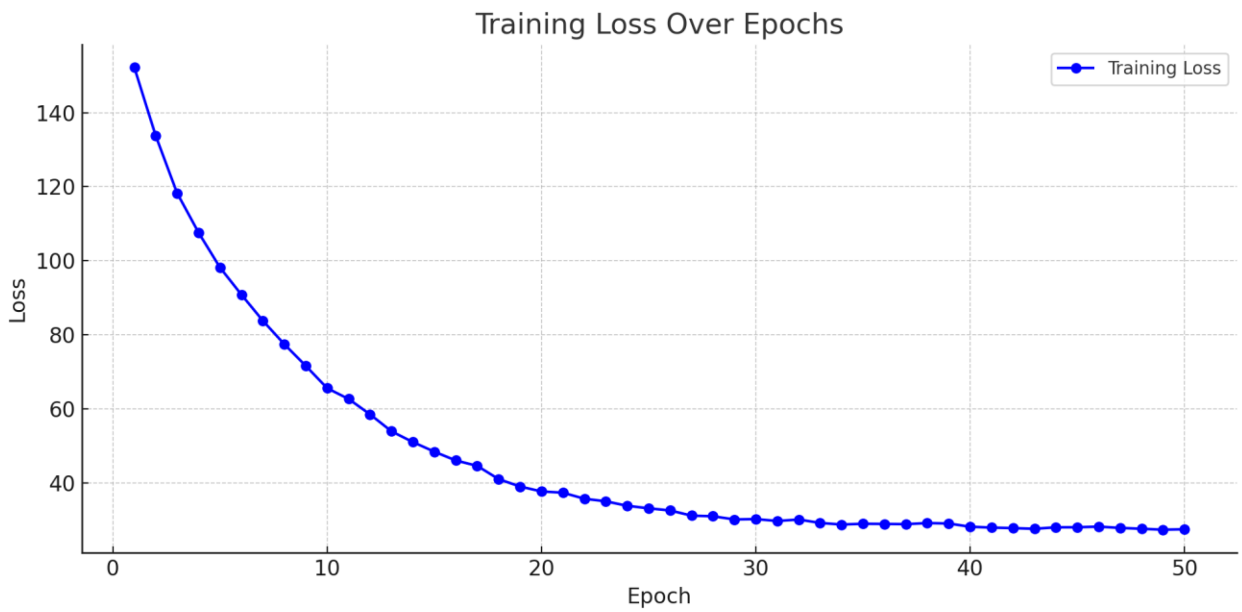


Graph of loss per epoch for filtered dataset (94 writers) :

Training Loss Table:

Epoch	Training Loss
1	152.2293
2	133.7400
3	118.2322
4	107.5771
5	98.1402
6	90.8059
7	83.7735
8	77.4287
9	71.5851
10	65.5415
11	62.6328
12	58.4772
13	53.8903
14	50.9874
15	48.3827
16	46.0183

17	44.5621
18	40.9699
19	38.9780
20	37.6482
21	37.3112
22	35.6680
23	34.9793
24	33.7629
25	33.0882
26	32.5056
27	31.1142
28	30.9407
29	30.0808
30	30.1789
31	29.6610
32	30.0566
33	29.1341
34	28.7008
35	28.9085
36	28.8656
37	28.8036
38	29.1012
39	28.9769
40	28.1066
41	27.8779
42	27.7346
43	27.5495
44	27.9538
45	27.9659
46	28.1343
47	27.7927
48	27.5545
49	27.3119
50	27.4303



Discussion on Results

Training on the unfiltered dataset with over 2000 classes led to poor performance due to extreme class imbalance — many writers had only a few examples. The model showed overfitting, as training loss decreased but test accuracy remained low (31.73%).

To address this, we filtered the dataset to include only writers with at least 5 images, which resulted in 94 well-represented classes. This cleaned version allowed the model to learn meaningful patterns and generalize better to unseen data.

The use of a cosine classifier helped the model perform well on a fine-grained classification task, while label smoothing and augmentation further boosted robustness. The final model achieved a test accuracy of 94.07%, clearly demonstrating the importance of dataset balancing and architectural adjustments in high-class-count classification problems.