Project Analysis: Handwriting Writer Identification

1. Project Objective

This project aims to identify the writer of handwritten text using deep learning techniques. Specifically, it appears to be using two different approaches:

- A ResNet-based classification model to directly identify writers
- A Siamese network to determine if two handwriting samples are from the same writer

2. Datasets Used

The project uses the [IAM Handwriting Dataset], specifically a subset containing the top 50 writers. This dataset includes:

- Handwriting samples from various writers
- A mapping of form IDs to writer IDs
- The data is preprocessed to focus on the top 10 writers with the most samples

3. Models Used

ResNet Classification Model

- Base architecture: ResNet50 with pre-trained ImageNet weights
- Transfer learning approach with the base model frozen
- Additional layers:
    - Global Average Pooling
    - Dropout (0.5)
    - Dense layer (128 units)
    - Output layer with softmax activation

Siamese Network

- Custom CNN architecture with:
    - Multiple Conv2D layers (32, 64, 128 filters)
    - BatchNormalization layers
    - MaxPooling layers
    - Dense layer (128 units)
- Uses contrastive loss function to compare image pairs
- Binary output indicating whether two samples are from the same writer

4. Results

ResNet Classification Model

- Training accuracy: ~94.7%
- Validation accuracy: ~95.8%
- Loss decreased steadily from ~1.45 to ~0.18
- Confusion matrix shows strong diagonal values, indicating good classification
- Best results seen after 15 epochs

```
Epoch 1/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 362s 8s/step - accuracy: 0.3952 - loss: 2.1408 - val_accuracy: 0.7688 - val_loss: 0.8142 - learning_rate: 0.0010
Epoch 2/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 392s 8s/step - accuracy: 0.7314 - loss: 0.8220 - val_accuracy: 0.8552 - val_loss: 0.4476 - learning_rate: 0.0010
Epoch 3/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 334s 7s/step - accuracy: 0.7987 - loss: 0.6200 - val_accuracy: 0.8774 - val_loss: 0.3402 - learning_rate: 0.0010
Epoch 4/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 391s 8s/step - accuracy: 0.8594 - loss: 0.4267 - val_accuracy: 0.9053 - val_loss: 0.2802 - learning_rate: 0.0010
Epoch 5/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 346s 8s/step - accuracy: 0.9003 - loss: 0.3098 - val_accuracy: 0.9109 - val_loss: 0.2670 - learning_rate: 0.0010
Epoch 6/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 379s 8s/step - accuracy: 0.9279 - loss: 0.2294 - val_accuracy: 0.9276 - val_loss: 0.2358 - learning_rate: 0.0010
Epoch 7/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 409s 8s/step - accuracy: 0.9351 - loss: 0.2161 - val_accuracy: 0.9331 - val_loss: 0.2318 - learning_rate: 0.0010
Epoch 8/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 360s 8s/step - accuracy: 0.9202 - loss: 0.2454 - val_accuracy: 0.9164 - val_loss: 0.2411 - learning_rate: 0.0010
Epoch 9/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 342s 8s/step - accuracy: 0.9369 - loss: 0.1737 - val_accuracy: 0.9471 - val_loss: 0.2041 - learning_rate: 0.0010
Epoch 10/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 383s 8s/step - accuracy: 0.9463 - loss: 0.1589 - val_accuracy: 0.9359 - val_loss: 0.2051 - learning_rate: 0.0010
Epoch 11/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 385s 8s/step - accuracy: 0.9591 - loss: 0.1341 - val_accuracy: 0.9387 - val_loss: 0.2019 - learning_rate: 0.0010
Epoch 12/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 381s 8s/step - accuracy: 0.9666 - loss: 0.1110 - val_accuracy: 0.9415 - val_loss: 0.2250 - learning_rate: 0.0010
Epoch 13/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 363s 7s/step - accuracy: 0.9573 - loss: 0.1346 - val_accuracy: 0.9387 - val_loss: 0.2113 - learning_rate: 0.0010
Epoch 14/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 381s 7s/step - accuracy: 0.9602 - loss: 0.1168 - val_accuracy: 0.9415 - val_loss: 0.1998 - learning_rate: 5.0000e-04
Epoch 15/15
45/45 ━━━━━━━━━━━━━━━━━━━━ 398s 8s/step - accuracy: 0.9578 - loss: 0.1440 - val_accuracy: 0.9582 - val_loss: 0.1863 - learning_rate: 5.0000e-04
<keras.src.callbacks.history.History at 0x78ef69248c10>
```

Siamese Network

- Training accuracy: ~75.7%
- Validation accuracy: ~73.2%
- Loss decreased steadily from ~1.44 to ~0.15
- Best results seen after 5 epochs

5. Discussion on Results

The ResNet classification model performs very well:

- High accuracy (over 95%) on validation data
- The training curves show a healthy learning pattern with no obvious overfitting
- The confusion matrix shows strong performance across all writer classes
- The model demonstrates ability to correctly identify writers from their handwriting

The training implementation includes several best practices:

- Early stopping and learning rate reduction callbacks
- Data preprocessing including normalization
- Class balancing considerations

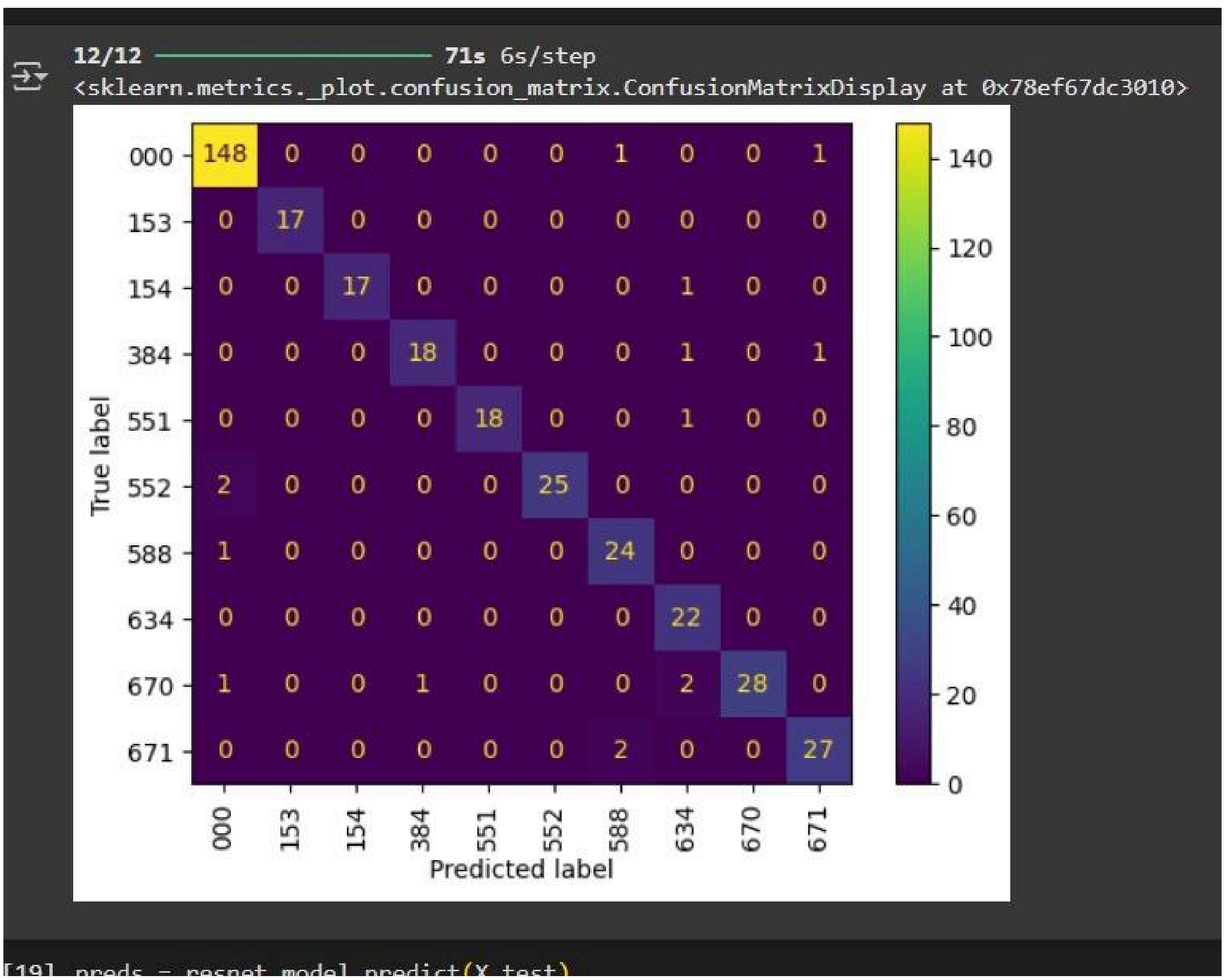
- Visualization of training progress

The Siamese network provides a complementary approach that focuses on similarity between handwriting samples rather than direct classification. This is particularly useful when:

- New writers are introduced to the system
- Limited examples per writer are available
- A verification system is needed

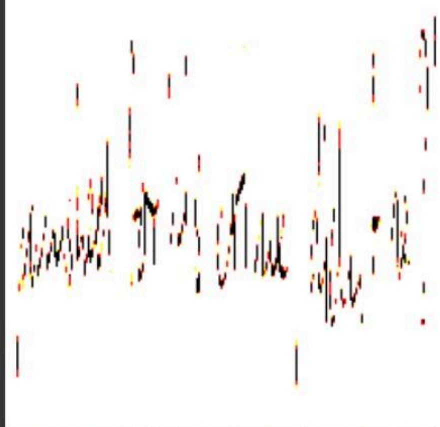
The confusion matrix shows exceptionally strong performance on writer identification, with most classes achieving perfect or near-perfect classification. The model appears particularly effective at distinguishing between different handwriting styles.

The example predictions show successful classification of test samples where the predicted label matches the true label.



```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers). Got ra
```

True: 671 | Pred: 671

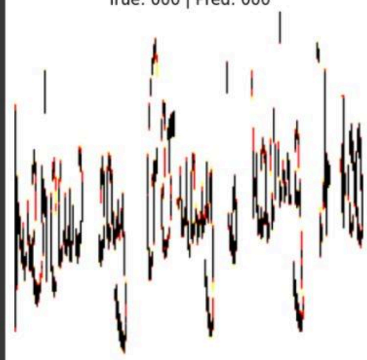


```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers). Got ra
```

True: 000 | Pred: 000

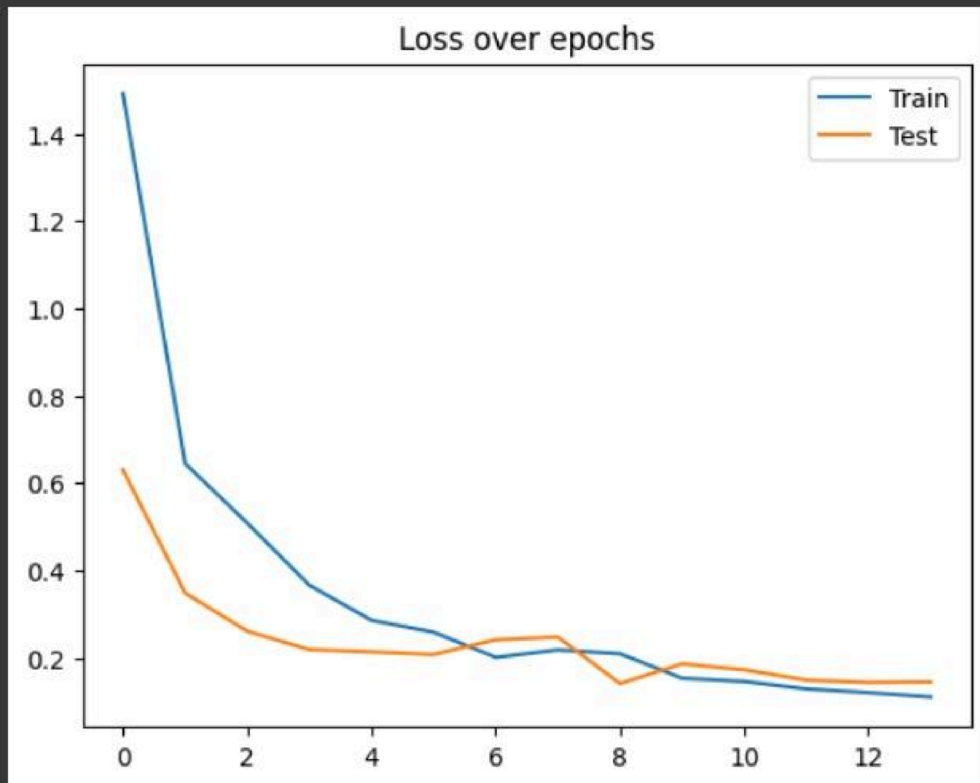
```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers). Got range
```

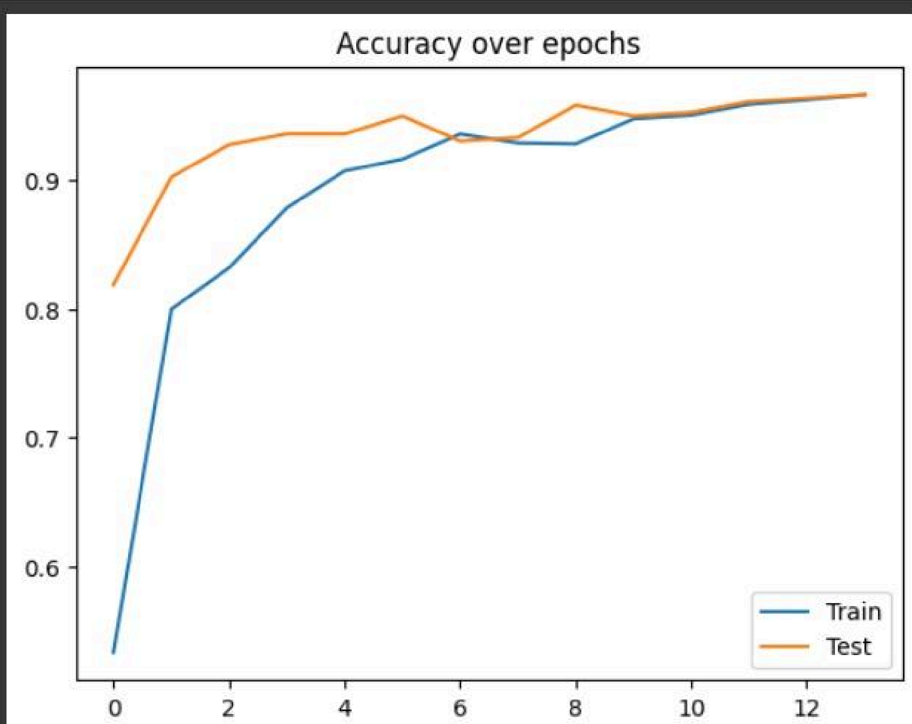
True: 000 | Pred: 000



```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers). Got range [-73.68..151.061].
```

True: 671 | Pred: 671


Loss over epochs
Train
Test
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
2
4
6
8
10
12


Accuracy over epochs
0.9
0.8
0.7
0.6
Train
Test
0
2
4
6
8
10
12