Multi-Character OCR: Challenges and Solutions

Nathaniel Liu (nliu26), Chandler Nelson (cnelso71), Jakub Piwowarczyk (jpiwowa2)

1 Introduction

In the ever-increasing technological world, there has been an increase in the number of digital devices that interpret an analog world. In order to process written text as characters, Optical Character Recognition (OCR) attempts to process an image of some characters and match it to a predetermined list of symbols in an alphabet.
For our final project, we attempted to use preliminary machine methods in order to properly classify not only single characters, but also a generated multi-character string given some characters. After training, we were able to make a model that recognized the majority of sequences with enough epochs.

2 Problem Definition and Importance

Patents for OCR date back to the late 1920s-early 1930s, and attempts to transform text itself into data date back to 1956 [1]. Having the data converted into something searchable allows for a large cataloging of text, which would allow scanned documents to be searchable without much intervention from human input. Although text can be trained by starting with individual characters, multiple character recognition also theoretically allows more redundancy when checking the text [2]. OCR is still in use today, with projects such as scanning the data from an ID badge [3] to verify the authenticity of consumable goods [4].
In our project, we decided to take a more limited scope of elementary classification methods in order to identify single characters, and then built on that to read multiple characters at once with a series of convolutional neural networks.

3 Methods

3.1 Datasets

For our project, we are specifically using the NIST Handprinted Forms and Characters Database, which contains handwritten forms from more than 3600 writers. In total, there are 810,000 character images with ground-truth classifications. Specifically, the text contains upper and lowercase letters, numerical digits, and free text fields [5]. We split up the dataset into 80% training, 10% validation, and 10% testing.

3.2 Model Architecture

3.2.1 Singe Character OCR (Random Forest)

This Single Character OCR Model consists of features being extracted from the images and then fed into a random forest. We extracted Histogram of Oriented Gradients (HOGs) and Hu Moments in order for the model to be able to understand the shape and overall orientation of the character. HOGs were calculated in 9 orientations, with 8x8 pixels per cell, 2x2 cells per block, and a L2-Hys block norm. The random forest was trained with 100 estimators.

3.2.2 Singe Character OCR (Deep-Learned)

This Single Character OCR Model consists of a convolutional neural network (CNN) for identifying more complex features within the images. A simple CNN was chosen with two convolutional layers, one max pool, and two dense layers. It was found that this relatively simple network was sufficient to obtain relatively good accuracy on the dataset. Detailed architecture specifications can be found below.

Layer	**Input Dimensions**	**Output Dimensions**
Conv2D + ReLU	(1, 32, 32)	(32, 32, 32)
Conv2D + ReLU	(32, 32, 32)	(64, 32, 32)
Max Pooling	(64, 32, 32)	(64, 16, 16)
Fully Connected (FC)	(64*16*16)	(128)
Fully Connected (FC)	(128)	(Num_Classes)

Table 1: Layer-wise Input and Output Dimensions of the Single Character OCR Model

3.2.3 Multi-Character OCR

Our Multiple-Character OCR Model consists of convolutional neural networks for feature extraction with bidirectional Long Short-Term Memory layers for sequence modeling due to its ability to capture spatial and temporal patterns [6]. The architecture ends with a fully connected layer for character classification.

Layer	**Input Dimensions**	**Output Dimensions**
Input	(Batch, 1, 32, 256)	(Batch, 1, 32, 256)
Conv2D + BatchNorm + ReLU	(Batch, 1, 32, 256)	(Batch, 64, 16, 128)
Pooling	(Batch, 64, 16, 128)	(Batch, 64, 16, 128)
Conv2D + BatchNorm + ReLU	(Batch, 64, 16, 128)	(Batch, 128, 8, 64)
Pooling	(Batch, 128, 8, 64)	(Batch, 128, 8, 64)
Reshape for LSTM	(Batch, 128, 8, 64)	(Batch, 64, 128*8)
Bidirectional LSTM	(Batch, 64, 128*8)	(Batch, 64, Hidden*2)
Fully Connected (FC)	(Batch, 64, Hidden*2)	(Batch, 64, Num_Classes)

Table 2: Layer-wise Input and Output Dimensions of the Multi-Character OCR Model

3.3 Training Procedure

All training was performed on Google Collab using the A100 GPU Machines.

3.3.1 Single Character OCR (Random Forest)

This Single Character OCR model was trained to recognize single characters of text. The training dataset consisted of labeled image-text pairs, where the input images were resized to a fixed size of 32x32 pixels and were also normalized. The features were extracted from the normalized images and the random forest was trained and tested on this data.

3.3.2 Single Character OCR (Deep-Learned)

This Single Character OCR model was trained to recognize single characters of text. The training dataset consisted of labeled image-text pairs, where the input images were resized to a fixed size of 32x32 pixels and were also normalized. The effect of changing the learning rate was explored and the best model was selected.

- Batch Size: 128
- Number of Epochs: [10, 30, 50]
- Optimizer: Adam
- Learning Rate: [1e-3, 1e-4, 1e-5]
- Scheduler: None
- Loss Function: Cross Entropy Loss

The performance of the single character OCR model is evaluated using the following metrics:

- **Character Accuracy**: Measures the percentage of correctly predicted characters over the total number of characters in the target sequences.

$$\text{Character Accuracy} = \frac{\text{Number of Correct Characters}}{\text{Total Number of Characters}} \times 100$$

3.3.3 Multi-Character OCR

The Multi-Character OCR model was trained to recognize and transcribe sequences of characters from input images. The training objective minimized Connectionist Temporal Classification loss, which is well suited for sequence-to-sequence tasks[7]. The training dataset consisted of labeled image-text pairs, where the input images were resized to a fixed height of 32 pixels and a width of 256 pixels. The labels represented sequences of characters from the input images.

- Batch Size: 64
- Number of Epochs: 15

- Optimizer: Adam
- Learning Rate: 0.001
- Scheduler: StepLR
- Step Size (Scheduler): 5
- Gamma (Scheduler): 0.5
- Loss Function: CTC Loss
- Early Stopping Patience: 5 Epochs Without Improvement
- Sequence Model Hidden Size: 256
- Sequence Model Layers: 2
- Bidirectional LSTM: Enabled

The performance of the OCR model is evaluated using the following metrics:

- **Character Accuracy**: Measures the percentage of correctly predicted characters over the total number of characters in the target sequences.
$$\text{Character Accuracy} = \frac{\text{Number of Correct Characters}}{\text{Total Number of Characters}} \times 100$$
- **Sequence Accuracy**: Measures the percentage of correctly predicted sequences (entire sequences must match the target exactly).
$$\text{Sequence Accuracy} = \frac{\text{Number of Correct Sequences}}{\text{Total Number of Sequences}} \times 100$$
- **CTC Loss**: Measures the alignment-based error between the predicted and target sequences using the Connectionist Temporal Classification (CTC) loss function. It is defined as:
$$\text{CTC Loss} = -\log\left(\sum_{\pi \in \mathcal{B}^{-1}(Y)} P(\pi \mid X)\right)$$
where $\mathcal{B}^{-1}(Y)$ represents all possible paths that map to the target sequence Y, and $P(\pi \mid X)$ is the probability of path π given the input sequence X.

4 Results

4.1 Single-Character OCR (Random-Forest

The random forest model achieved an accuracy of 68.2% on the test dataset. As can be seen from Figure 1, it greatly struggled with differentiating between some upper and lower case letters. Accuracy for this model could likely be improved by adding features that consider character size and or area taken up within the input image.

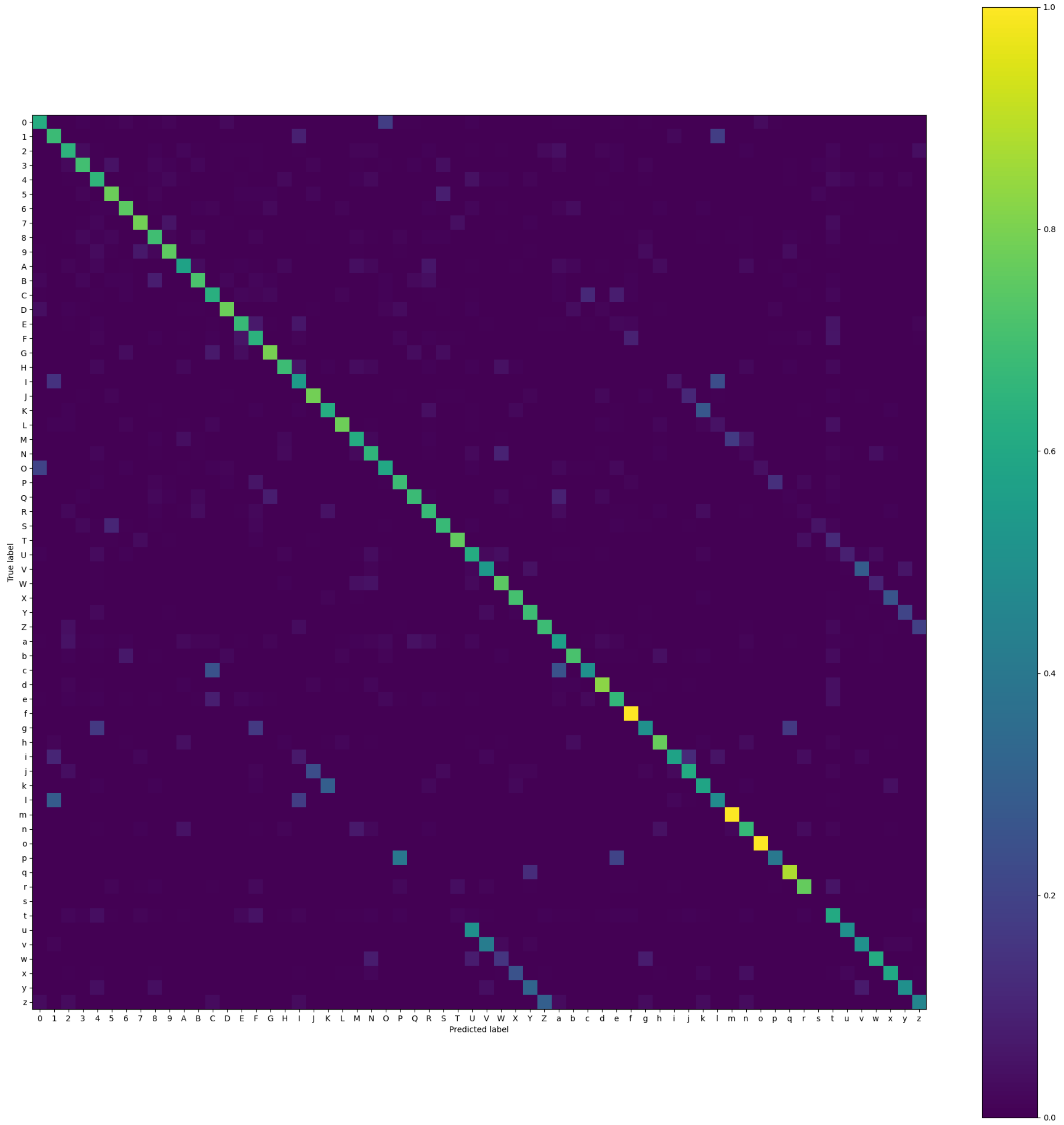

Figure 1: Confusion matrix for the random forest single character OCR.

4.2 Single-Character OCR (Deep-Learned)

The CNN-based model with a 1e-4 learning rate achieved an accuracy of 88.2% on the test dataset. As shown in Figure 2, all models were able to reach training and validation accuracies of around 90%. The model with the 1e-2 learning rate started to over-train, whereas the model with the 1e-05 started to plateau before reaching the same accuracies as the other models. This is why the model with a 1e-04 learning rate was chosen.

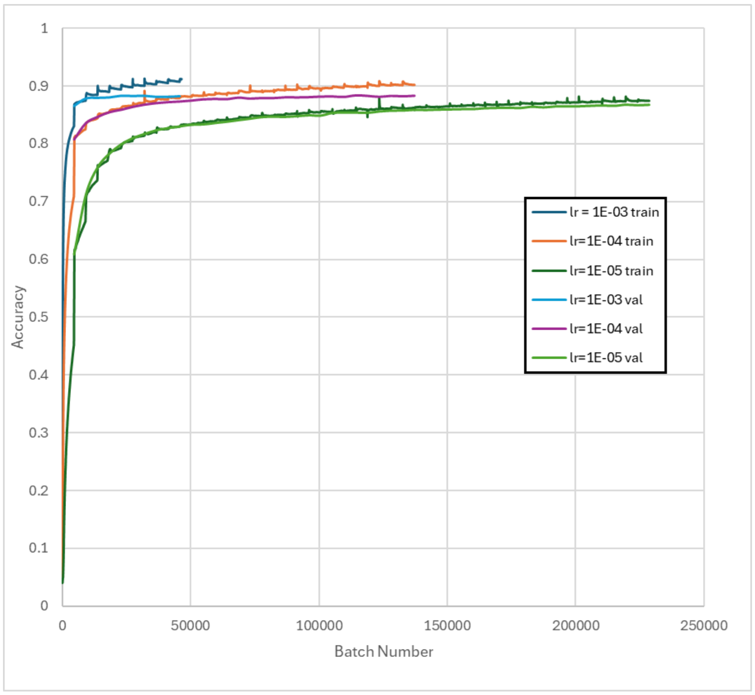

Figure 2: Training and validation accuracy over the course of training for different learning rates.

As can be seen from the confusion matrix in Figure 3, the model did an excellent job identifying almost all characters. There were a few characters that it found difficult such as ”8” and ”g” due to their similar shapes, however, the performance was still excellent.

4.3 Multi-Character OCR

As shown in Figure 4, the model's performance on character accuracy consistently improved during training, reaching over 87.76% on the test set, and over 80% on the train and validation sets.

As depicted in Figure 5, the model's performance on sequence accuracy showed a steady upward trend during earlier epochs but ultimately plateaued at under 60% on both the train and validation sets. Additionally, the model scored a 52.58% sequence accuracy on the test set.

Throughout training, both the validation loss and training loss steadily decreased, as shown in Figure 6. The model achieved a test loss of 0.3549.

5 Challenges Encountered

- **Data Preparation**: Generating diverse multi-character datasets required significant preprocessing.

- **Training Duration**: Training the OCR models for high accuracy demanded extended computational time. This was partially addressed in the multi-character OCR using

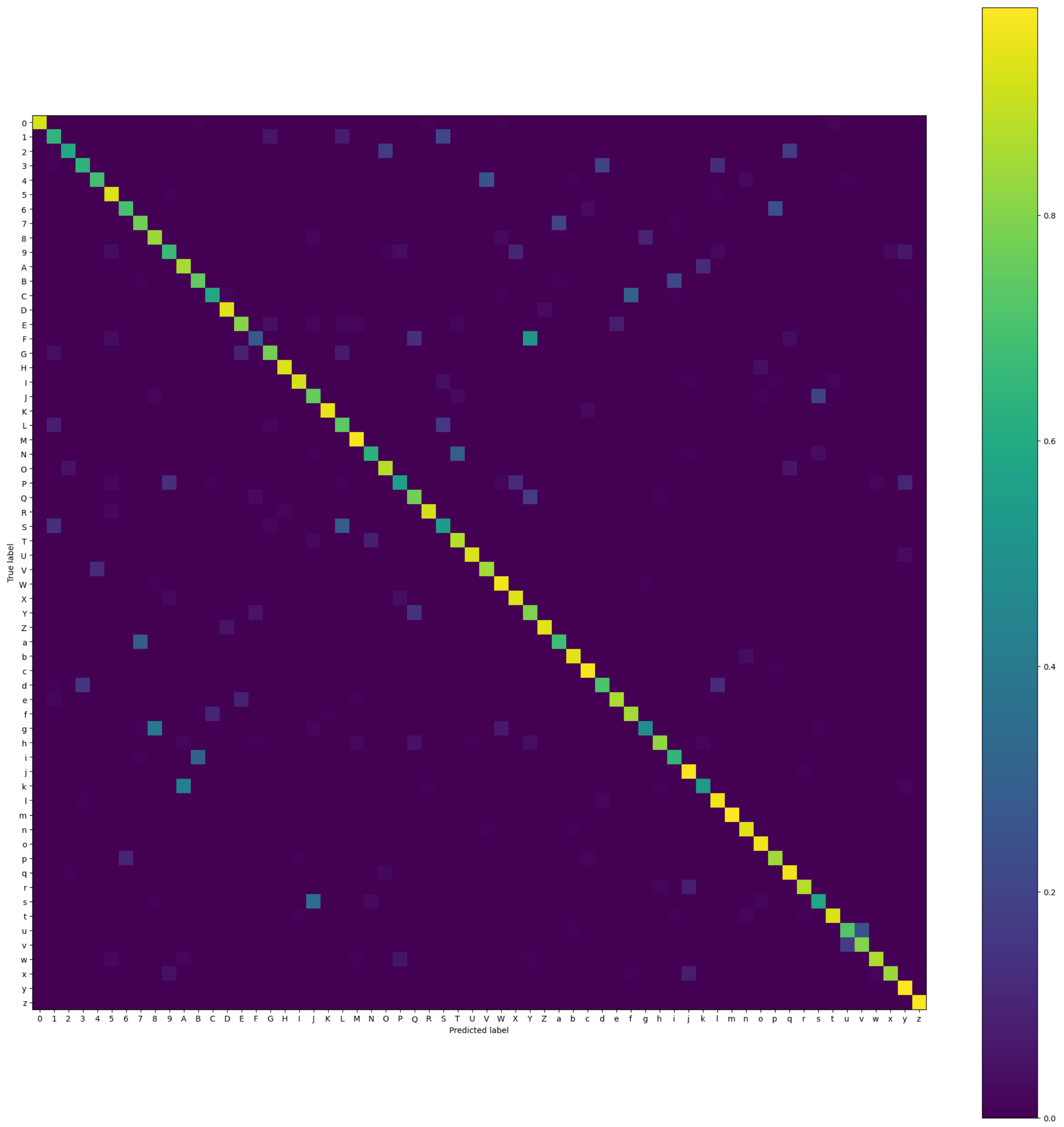

Figure 3: Confusion matrix for the cnn-based single character OCR.

Batch Normalization.

6 Implications

The results of the Multi-Character OCR model suggest that the model effectively learns character-level predictions and achieves relatively strong performance for multi-character se-

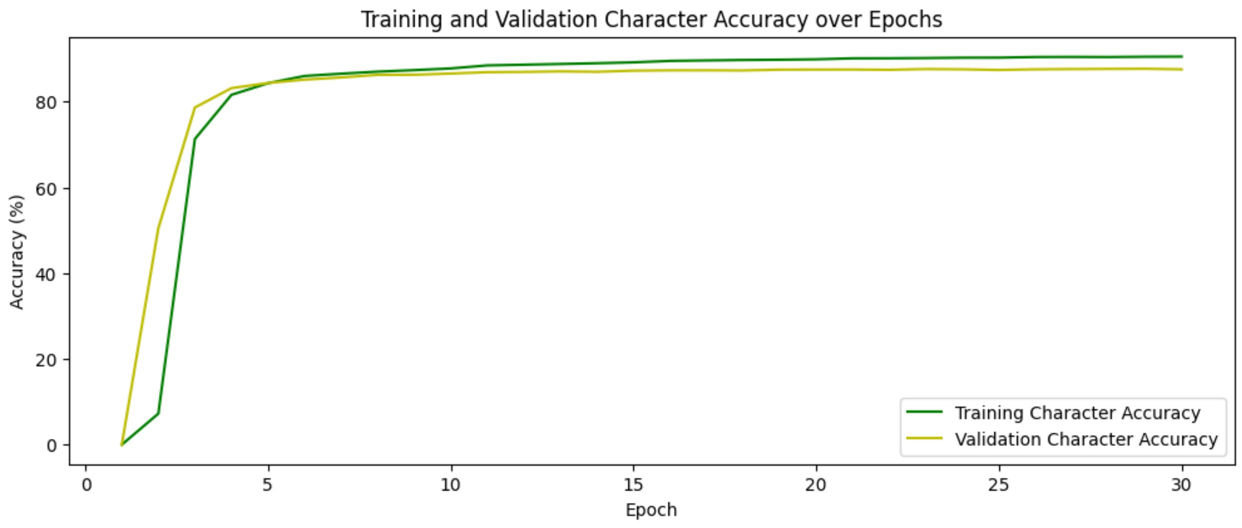

Figure 4: Training and validation character accuracy over epochs.

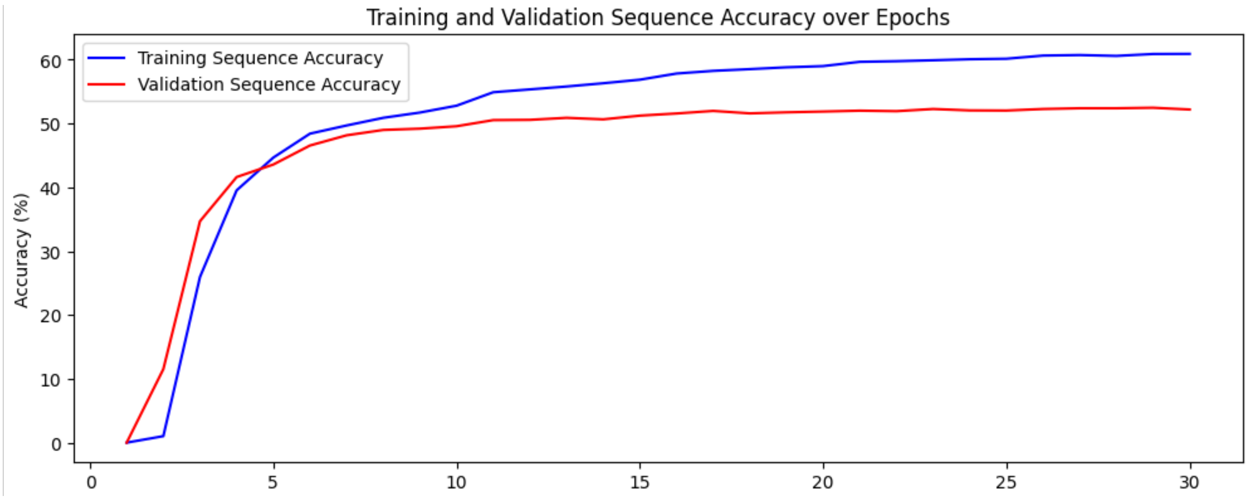

Figure 5: Training and validation sequence accuracy over epochs.

quences. However, the disparity between character accuracy and sequence accuracy suggests challenges in modeling dependencies across longer sequences. The lower sequence accuracy suggests that architectural enhancements, such as attention mechanisms or transformer-based models, are needed.

7 Future Work

Although we achieved relatively successful classification on the datasets, the next step would be to use a noisier dataset to improve the robustness of our model. Additionally, while a CNN was used for feature extraction, exploring transformer-based architectures for enhanced sequence modeling could yield further improvements. Finally, it would be interesting to expand to a different language with a more complex system of characters, more than the 62

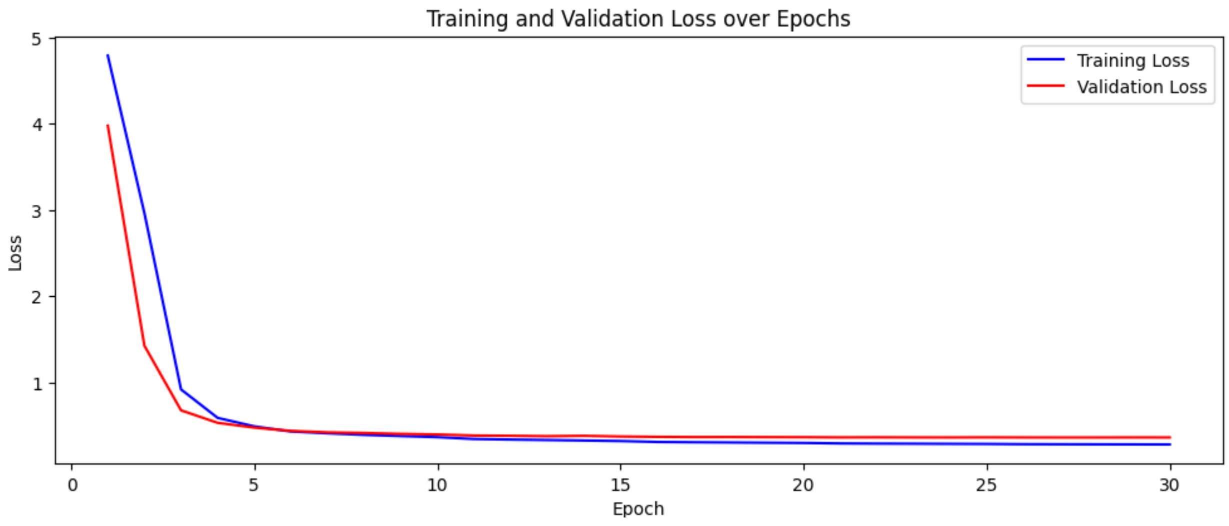

Figure 6: Training and validation loss over epochs.

used within our model.

References

1 Shunji Mori, Ching Y. Suen, and Kazuhiko Yamamoto, ”Historical review of OCR research and development,” *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1029-1057, July 1992. Available: `https://doi.org/10.1109/5.156468`.

2 Lawrence O’Gorman, ”The document spectrum for page layout analysis,” *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 2422, pp. 6-16, 1995. Available: `https://doi.org/10.1117/12.373511`.

3 W. Cavalcante, I. G. Torné, L. Camelo, R. Fernandes, A. Printes, and H. Bragança, ”An ID Badge Information Extractor Based on Object Detection and Optical Character Recognition,” *IEEE Access*, vol. 12, pp. 152559-152567, Oct. 2024. Available: `https://doi.org/10.1109/ACCESS.2024.3471449`.

4 Z. Yang, Y. Li, and X. Wang, ”CLCRNet: An Optical Character Recognition Network for Cigarette Laser Code,” *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1-11, 2024. Available: `https://doi.org/10.1109/TIM.2024.3041234`.

5 National Institute of Standards and Technology, ”NIST Special Database 19,” [Online]. Available: `http://doi.org/10.18434/T4H01C`.

6 Marjani M, Mahdianpari M, Mohammadimanesh F. CNN-BiLSTM: A Novel Deep Learning Model for Near-Real-Time Daily Wildfire Spread Prediction. *Remote Sensing*. 2024; 16(8):1467. Available: `https://doi.org/10.3390/rs16081467`.

7 A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, pp. 369-376, June 2006. Available: https://www.cs.toronto.edu/~graves/icml_2006.pdf.