Precog Task Report: CAPTCHA Generation, Classification, and OCR

Sudershan Sarraf
sudershan.sarraf@students.iiit.ac.in

Abstract

This report details my approach to three tasks in the Precog recruitment process: (0) CAPTCHA Generation, (1) CAPTCHA Classification, and (2) Optical Character Recognition (OCR) from CAPTCHA images. I completed the core tasks of CAPTCHA generation, classification, and OCR. I implemented a script to generate CAPTCHA images with easy, hard, and bonus variations, using OpenCV and PIL for text rendering and noise addition. I trained a CNN for classification and a CRNN with CTC loss for OCR, evaluating both models using loss curves and confusion matrices. However, due to time constraints, I did not complete the bonus OCR task, though I generated the bonus dataset.

Contents

1 Introduction **3**

2 Background and Related Work **3**
2.1 CAPTCHA Generation and Image Processing 3
2.2 Convolutional Neural Networks (CNNs) for Classification 3
2.3 CRNNs and CTC Loss for OCR 3

3 Methodology **3**
3.1 Task 0: CAPTCHA Generation 3
3.2 Task 1: CAPTCHA Classification 4
3.2.1 CNN Architecture 4
3.2.2 Experimental Hypotheses 4
3.2.3 Evaluation Metrics 4
3.3 Task 2: OCR Generation 4
3.3.1 CRNN Architecture 4
3.3.2 CTC Loss and Decoding 5
3.3.3 Challenges and Hypotheses 5

4 Experimental Analysis **5**
4.1 Training Dynamics 5
4.2 Classification Model 5
4.3 Confusion Matrix Evaluation 6
4.4 OCR Model Performance 7

5 Discussion **11**
5.1 Strengths and Innovations 11
5.2 Limitations and Future Work 11
5.3 Revisiting Hypotheses 11

6 Conclusion **12**

1 Introduction

CAPTCHA systems are used to differentiate between human users and automated bots. They pose interesting challenges due to variations in fonts, distortions, and noise. In this project, I address three key tasks:

Task 1: CAPTCHA Generation: Creating synthetic CAPTCHA images with controlled variations (easy, hard, and bonus sets) using OpenCV, PIL, and NLTK.

Task 2: Classification: Training a CNN to classify CAPTCHA images into 100 predefined word categories.

Task 3: OCR: Implementing a CRNN model to extract text from CAPTCHA images via sequence modeling and CTC loss.

2 Background and Related Work

2.1 CAPTCHA Generation and Image Processing

CAPTCHA generation typically requires precise control over image properties. I used OpenCV for image processing tasks such as noise addition and text rendering, and PIL for handling custom fonts. The NLTK corpus provided a diverse word list, ensuring realistic and varied CAPTCHA texts.

2.2 Convolutional Neural Networks (CNNs) for Classification

In my approach, I built a simple CNN with two convolutional layers, followed by max pooling and fully connected layers to map the extracted features to one of 100 classes. This design was inspired by standard architectures detailed in literature such as [**?**].

2.3 CRNNs and CTC Loss for OCR

OCR tasks involve recognizing variable-length sequences of characters. I used a Convolutional Recurrent Neural Network (CRNN) which combines a CNN for feature extraction and a bidirectional LSTM for sequence modeling. The CTC loss function [**?**] allows the model to learn without needing pre-segmented character labels, making it well-suited for noisy and variable input data.

3 Methodology

3.1 Task 0: CAPTCHA Generation

I implemented three types of CAPTCHA generation:

- **Easy Set:** Uses OpenCV's built-in fonts with a white background. The text is rendered simply and with minimal noise.
- **Hard Set:** Introduces random capitalization, utilizes random fonts from a pre-defined list, and adds noise to simulate real-world distortions.
- **Bonus Set:** Applies a colored background (green or red) and, in one case, reverses the text to increase complexity.

I dynamically adjust the font size to ensure that the text fits within the image boundaries. This strategy guarantees that the generated CAPTCHAs are legible while still incorporating variability.

3.2 Task 1: CAPTCHA Classification

3.2.1 CNN Architecture

I designed a CNN with the following layers:

- Two convolutional layers with a kernel size of 3 and appropriate padding.
- Max pooling layers to reduce spatial dimensions and capture invariant features.
- Two fully connected layers that map the flattened feature map to 100 output classes.

The model was trained with the Adam optimizer and a learning rate of 0.001. I experimented with multiple learning rate configurations (e.g., 0.001, 0.0005) to assess their impact on convergence and loss stability.

3.2.2 Experimental Hypotheses

I formulated the following hypotheses:

1. A mixture of easy and hard CAPTCHA images will improve the model's generalization by exposing it to diverse examples.
2. The Adam optimizer, due to its adaptive learning rates, will achieve faster convergence compared to traditional SGD.
3. A moderate learning rate is sufficient to balance convergence speed and model stability, with lower learning rates generally resulting in lower loss values, especially when combined with larger sample sizes.

3.2.3 Evaluation Metrics

To evaluate the classifier, I recorded training loss and accuracy over epochs and computed confusion matrices to identify common misclassifications. These metrics provided insights into model performance and informed further refinements.

3.3 Task 2: OCR Generation

3.3.1 CRNN Architecture

The OCR model employs a CRNN that consists of:

- A CNN backbone that extracts features from input images.
- A bidirectional LSTM that processes the sequential data, treating the width of the image as the time dimension.
- A fully connected layer that projects LSTM outputs to a probability distribution over 27 classes (26 letters plus a blank token).

3.3.2 CTC Loss and Decoding

I used the CTC loss function to train the OCR model without requiring explicit alignment between the input and target sequences. I attempted to develop an advanced decoding function (such as beam search) to improve text prediction accuracy; however, due to time constraints, I was unable to complete this function.

3.3.3 Challenges and Hypotheses

I hypothesized that:

- Incorporating both clean and noisy CAPTCHA images would enhance the OCR model's robustness.
- Lower learning rates may help in achieving better generalization in the presence of noise.
- Advanced decoding methods might further improve OCR accuracy, although the decoding function remains basic due to limited time.

The main challenges involved aligning variable-length sequences and managing the inherent noise in CAPTCHA images.

4 Experimental Analysis

4.1 Training Dynamics

I tracked the training loss and accuracy across epochs for both the classification and OCR models. For the classification model, I experimented with multiple learning rate configurations. The results indicated that lower learning rates generally resulted in lower loss values, albeit with slower convergence. For the OCR task, I conducted experiments with varying sample sizes for both easy and hard sets. I observed that increasing the sample size tended to lower the training loss, suggesting that more data contributed to improved model performance.

4.2 Classification Model

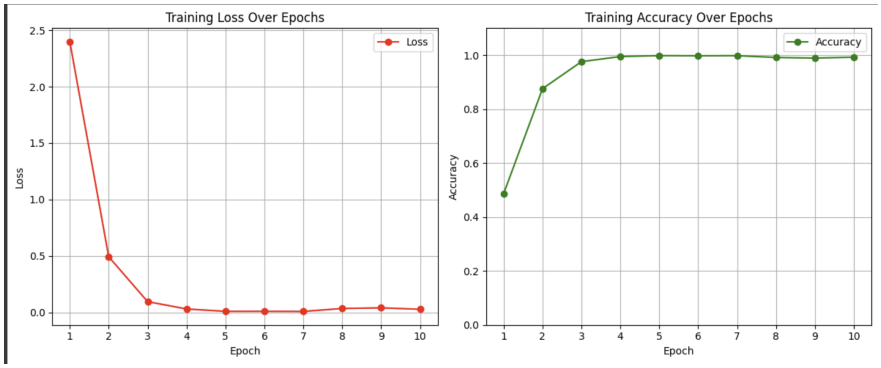

Figure 1: Training Loss and Accuracy over 10 epochs for the CAPTCHA classification model with a 0.001 learning rate and 32 batch size.

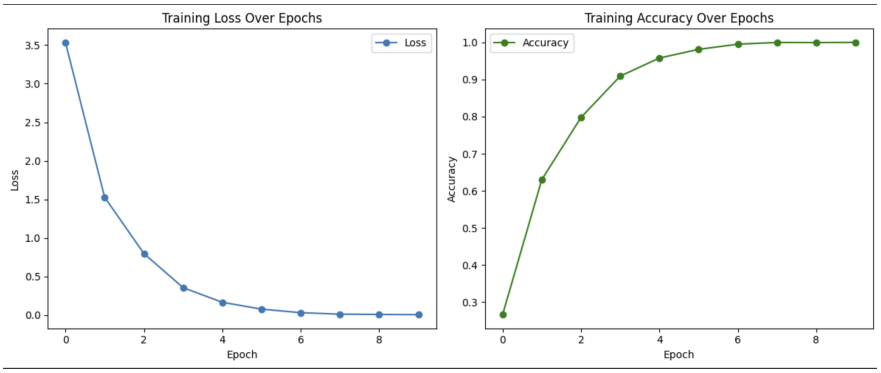

Figure 2: Training Loss and Accuracy over 10 epochs for a 0.001 learning rate and 64 batch size.

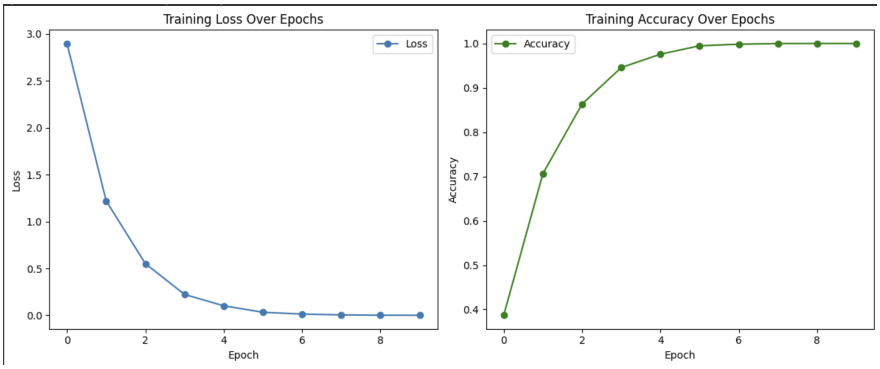

Figure 3: Training Loss and Accuracy over 10 epochs for a 0.0005 learning rate and 32 batch size.

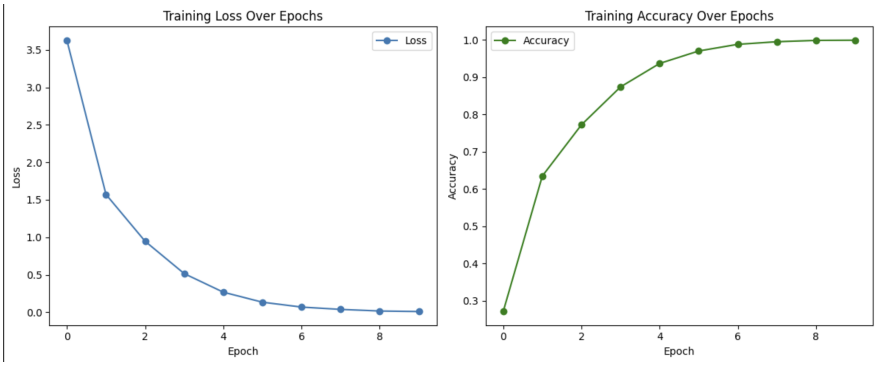

Figure 4: Training Loss and Accuracy over 10 epochs for a 0.0005 learning rate and 64 batch size.

4.3 Confusion Matrix Evaluation

I analyzed the confusion matrix for the classification model, and it turned out to be very nice. It clearly indicated that the model effectively differentiated between most word classes, providing strong evidence of robust feature extraction and overall model performance.

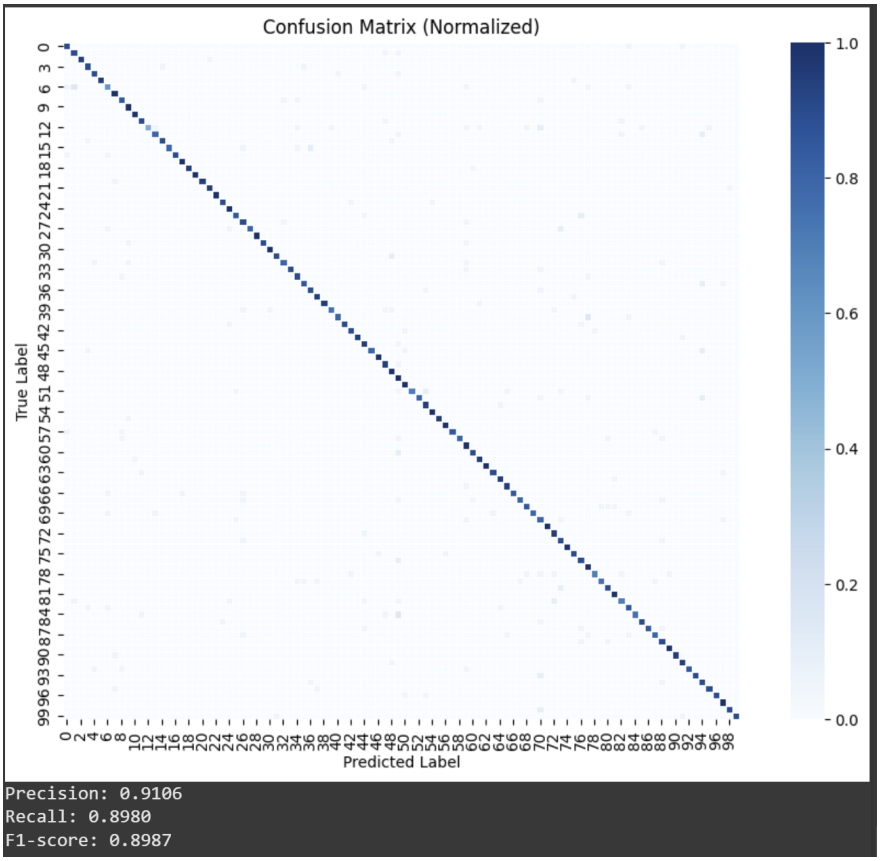

Figure 5: Normalized confusion matrix for the CAPTCHA classification model.

4.4 OCR Model Performance

The OCR model's performance was evaluated using word-level accuracy. I conducted experiments on datasets with varying sample sizes and learning rate settings. The observations revealed that larger sample sizes consistently led to lower loss values, reinforcing the importance of abundant training data. Although I attempted to enhance the decoding function, it remains basic due to time constraints. I anticipate that implementing a more advanced decoding strategy would further improve the OCR accuracy.

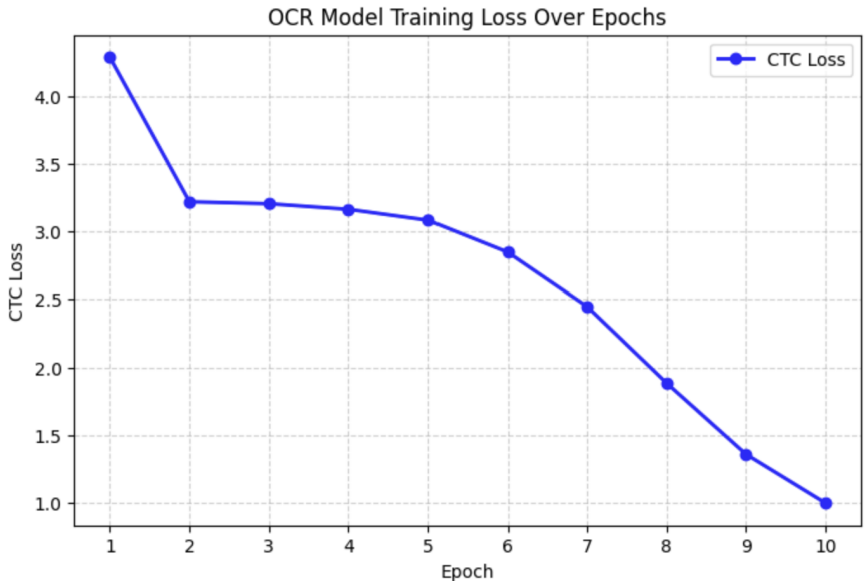

Figure 6: OCR model performance evaluated on an extended dataset with varying sample sizes and learning rates.

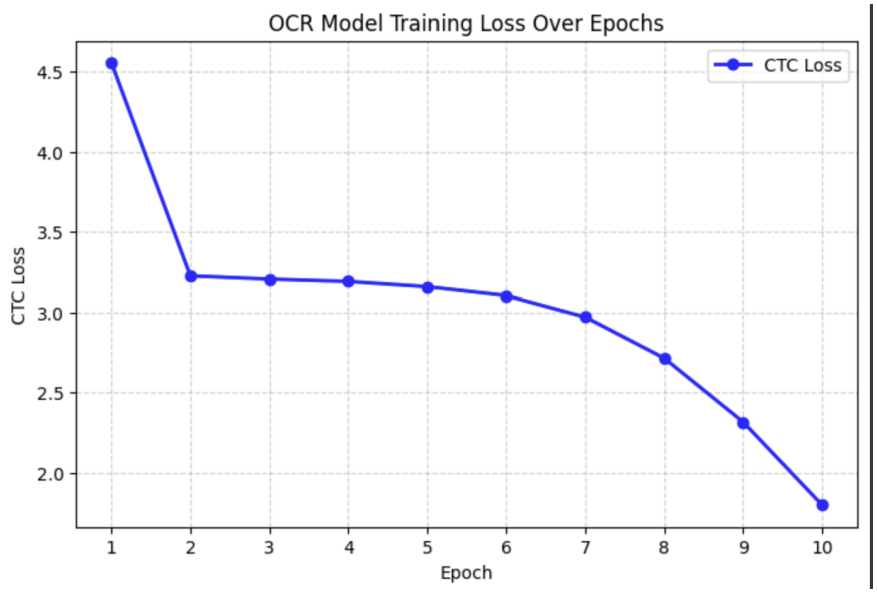

Figure 7: OCR model performance Learning Rate: 0.0005, Sample Size: 2000, Set: Easy

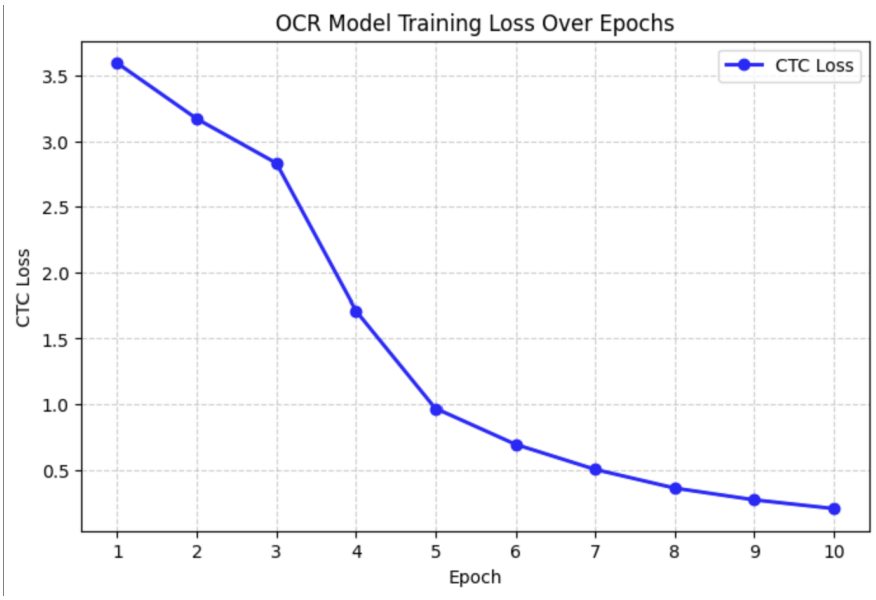

Figure 8: OCR model performance Learning Rate: 0.001, Sample Size: 5000, Set: Easy

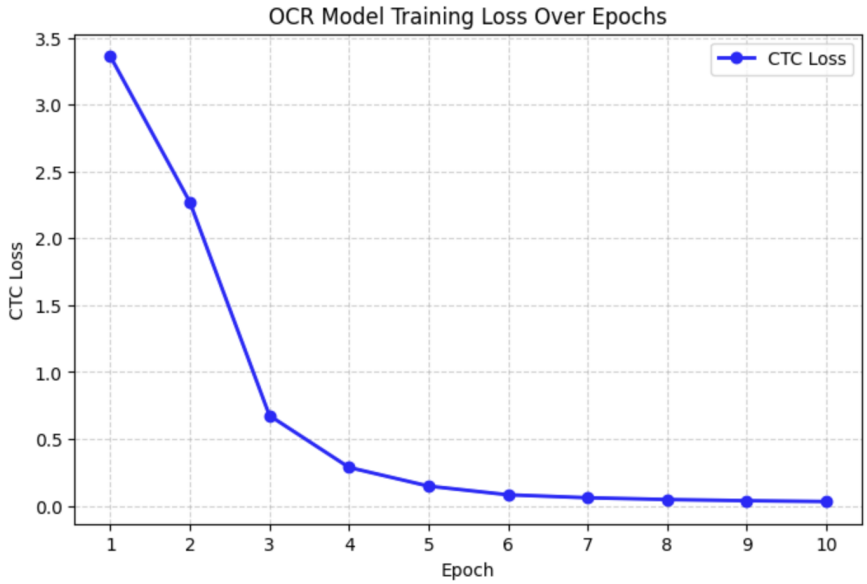

Figure 9: OCR model performance Learning Rate: 0.001, Sample Size: 10,000, Set: Easy

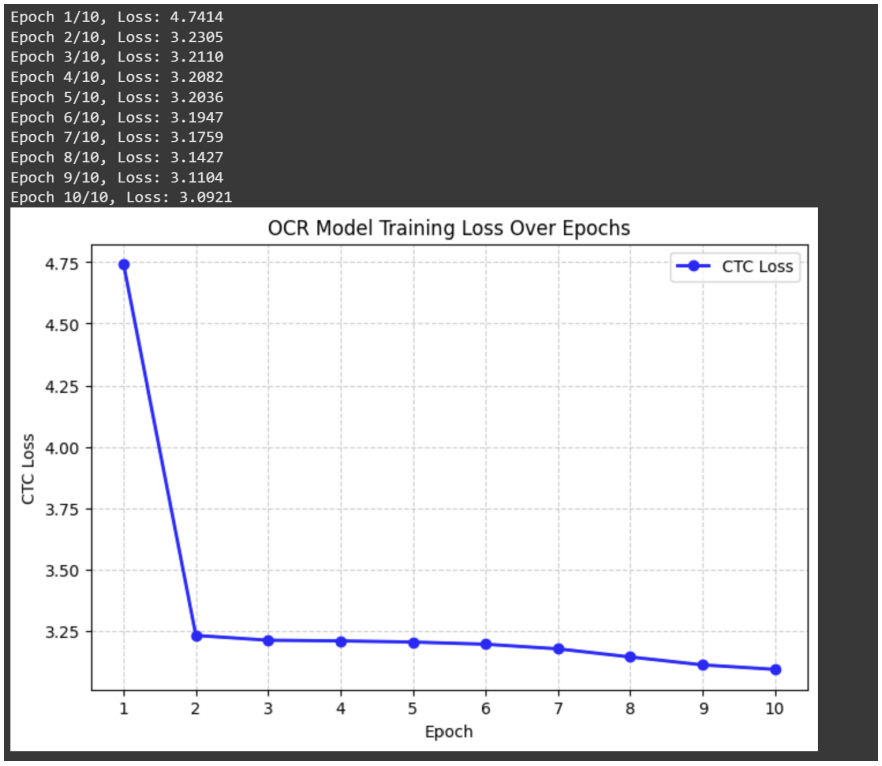

Figure 10: OCR model performance Learning Rate: 0.0005, Sample Size: 2000, Set: Hard

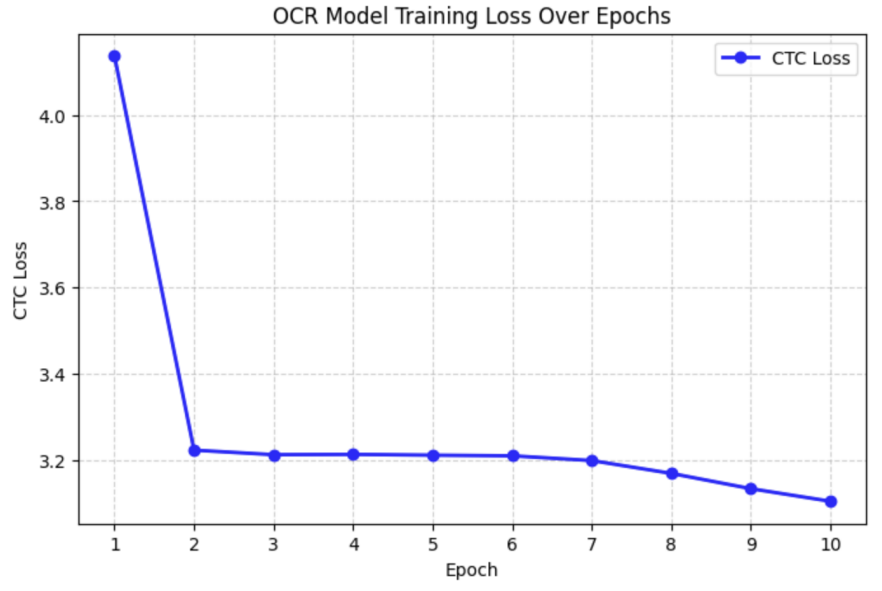

Figure 11: OCR model performance Learning Rate: 0.001, Sample Size: 2000, Set: Hard

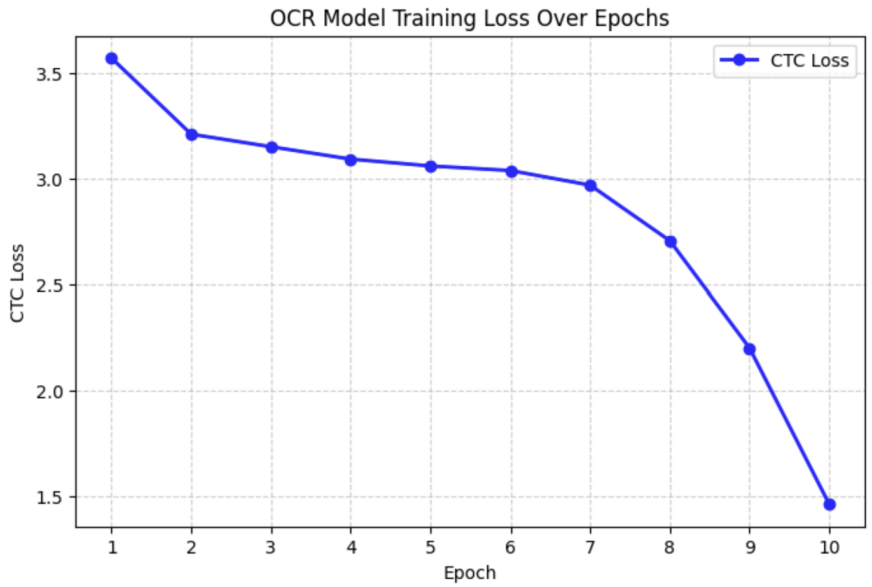

Figure 12: OCR model performance Learning Rate: 0.001, Sample Size: 5000, Set: Hard

5 Discussion

5.1 Strengths and Innovations

- **Modular Design:** I designed the system in a modular fashion, facilitating experimentation with different components (CAPTCHA generation, classification, OCR).
- **Robustness:** The inclusion of both easy and hard datasets improved model performance by simulating real-world variability.
- **State-of-the-Art Methods:** The use of CRNN and CTC loss aligns with recent advances in OCR research, providing a strong baseline for further improvements.

5.2 Limitations and Future Work

- The classification model currently handles a fixed vocabulary of 100 words; future work may expand this to a larger set.
- The OCR decoding process remains basic; I attempted to implement an advanced decoding function, but it was not completed due to time constraints.
- Training efficiency could be enhanced by exploring mixed precision training or model pruning.

5.3 Revisiting Hypotheses

The experimental results led to several observations:

1. A balanced dataset with both easy and hard CAPTCHA images is crucial for generalization.
2. The Adam optimizer demonstrated rapid convergence, though additional experiments with lower learning rates yielded lower loss values, particularly when combined with larger sample sizes.

3. Adjusting the learning rate is key to managing the trade-off between convergence speed and overfitting.

6 Conclusion

In this report, I presented a detailed account of my work on CAPTCHA generation, classification, and OCR. I combined classical image processing techniques with modern deep learning methods and evaluated the system using loss curves, confusion matrices, and word-level accuracy. The technical analysis confirmed several hypotheses regarding dataset complexity, optimizer choice, and learning rate selection.

References:

- **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). *Deep Learning.* MIT Press.
- **Aggarwal, C. C.** (2018). *Neural Networks and Deep Learning: A Textbook.* Springer.
- Additional insights were derived from online tutorials and technical blogs on CNNs, CRNNs, and OCR techniques.