Project Report: SVHN Digit Classification Using CNN

Prepared by: Zahid Hussain

Roll NO.: 229

Introduction

The Street View House Numbers (SVHN) dataset is a real-world image dataset used to recognize digits in natural scene images. This project focuses on building and training a Convolutional Neural Network (CNN) model to classify the digits from the SVHN dataset.

Objectives

- Understand and preprocess the SVHN dataset for digit classification.
- Design and implement a CNN model using TensorFlow/Keras.
- Apply data augmentation techniques to improve model performance.
- Evaluate model accuracy using performance metrics.
- Visualize model predictions and misclassifications.

Dataset Overview

The SVHN dataset contains 73,257 training images and 26,032 test images, each representing digits cropped from natural scene images. The images are 32x32 pixels in size, with three color channels (RGB). The labels correspond to digits (0-9), and a one-hot encoding technique was used to convert these labels for classification.

Methodology

1. Data Preprocessing

Key preprocessing steps include normalization (scaling pixel values to the range [0, 1]), one-hot encoding of labels, and handling any misclassified labels.

2. Data Augmentation

Techniques like rotation (up to 15 degrees), zoom (up to 10%), random flips, and horizontal shifts were applied to enhance the generalization of the model. These augmentations increase the diversity of training data.

3. Model Architecture

A CNN model with the following layers was used:

- Convolutional layers with ReLU activation functions
- Max-pooling layers for down-sampling
- Dropout layers to prevent overfitting
- Fully connected layers for the final classification

4. Model Training

The model was trained over 20 epochs using the Adam optimizer with a batch size of 64. The training time was approximately 45 minutes on a GPU-powered system. The learning rate was set to 0.001, and early stopping was applied to avoid overfitting.

5. Evaluation Metrics

The model's performance was evaluated using accuracy, precision, recall, F1-score, and confusion matrix. Additionally, training and validation accuracy, as well as loss curves, were analyzed to track the model's performance during the training process.

Results

1. Accuracy

The final model achieved a test accuracy of 91.23%.

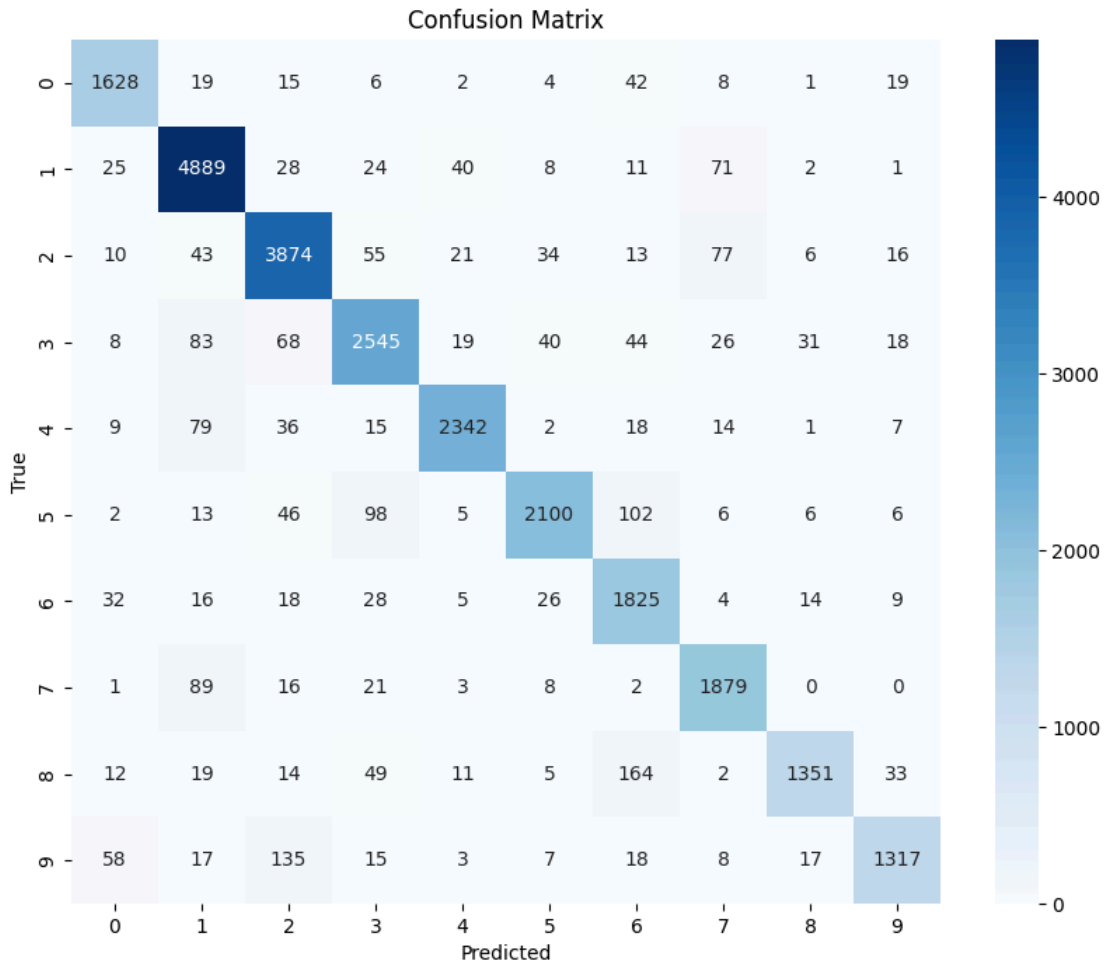
2. Classification Report

The precision, recall, and F1-scores for each digit class were calculated. The model achieved high performance across most classes, with F1-scores ranging between 0.87 and 0.94. The overall macro and weighted average F1-scores were both 0.91, indicating balanced performance for all digit classes.

- Precision for digit '0': 0.93
- Precision for digit '1': 0.91

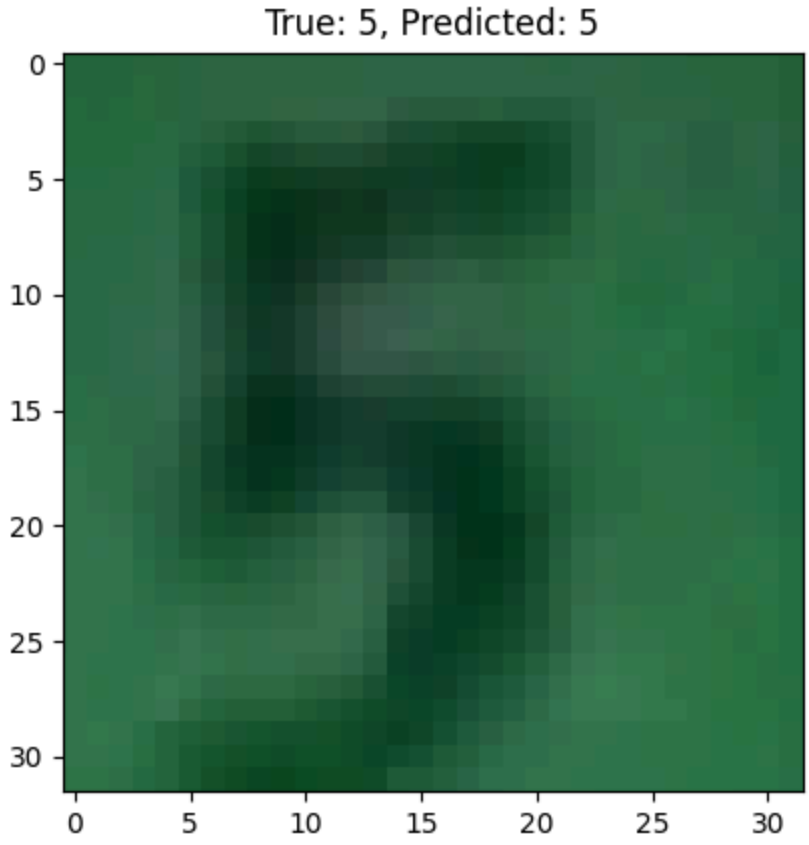
3. Confusion Matrix

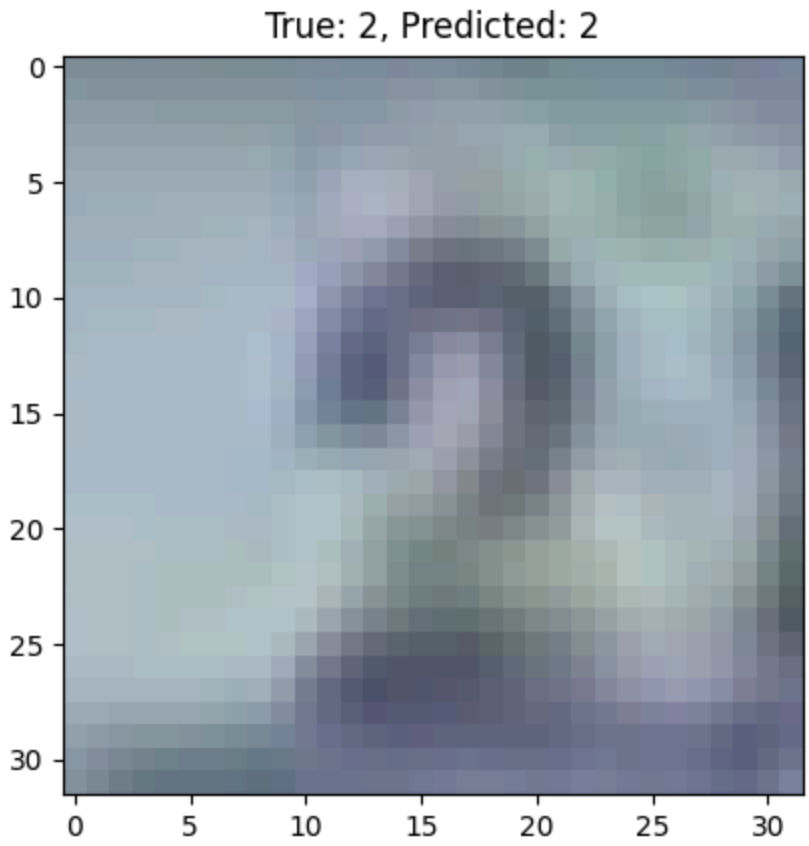
The confusion matrix shows the true vs. predicted labels for each digit class. Below is a snippet of the matrix:



4. Model Predictions

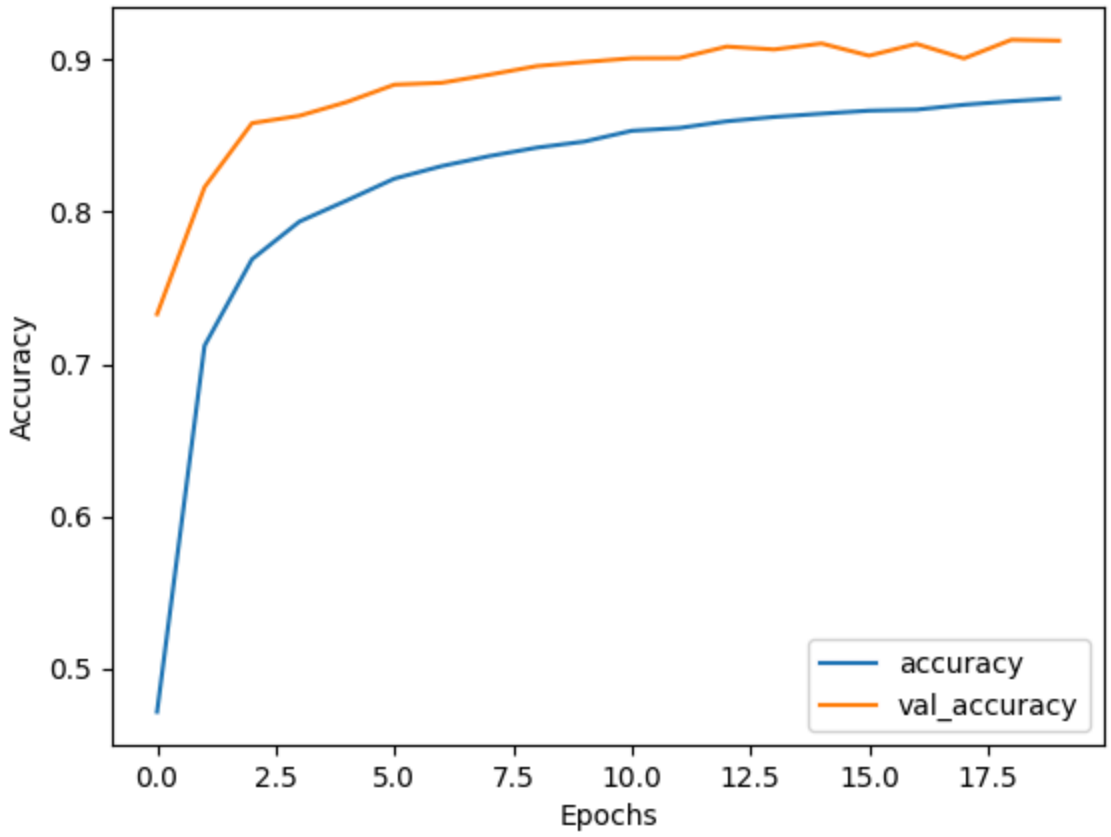
Sample test images are displayed alongside their true and predicted labels, providing insights into the model's accuracy and misclassifications. These samples illustrate where the model performs well and where it struggles.


True: 5, Predicted: 5
0
5
10
15
20
25
30
0
5
10
15
20
25
30


True: 2, Predicted: 2
0
5
10
15
20
25
30
0
5
10
15
20
25
30

5. Training and Validation Curves

Below is the plot showing the accuracy and loss over 20 epochs for both training and validation sets. The model's accuracy improved steadily, and the validation loss decreased without signs of overfitting.



Conclusion

The project successfully demonstrated how to classify digits from the SVHN dataset using CNNs. The use of data augmentation techniques and careful architecture design helped the model generalize well to unseen data.

Future Work

Future improvements include experimenting with deeper networks, hyperparameter tuning, and transfer learning. Adding ensemble methods or

trying more advanced architectures like ResNet could further enhance the model's performance.

References

TensorFlow Documentation: https://www.tensorflow.org

SVHN Dataset: http://ufldl.stanford.edu/housenumbers/

Keras API Reference: https://keras.io/api/