Road Surface Condition Classification using CNN

December 14, 2025

C417 - Computer Vision

Dr. Hend Dawood

Dr. Mahmoud Esmat

Authors:

Youssef Ahmed Nour El-Dien
2127137

Mina Lotfy Fekry
2127474

Menna Allah Hassan Youssef
2227273

Mohammed Ragab Mohammed
2127185

Amr Elbaroudy
2127039

Introduction

The problem at hand is the classification of road surface conditions based on images. This is a crucial task for various applications, including autonomous driving, road maintenance, and driver safety systems. An automated system that can accurately identify the condition of the road surface (e.g., good, poor, very poor) can provide valuable information for making real-time decisions. For instance, an autonomous vehicle could adjust its speed and driving parameters based on the road condition to ensure safety and ride comfort. Similarly, road maintenance agencies can use this information to prioritize repairs and allocate resources more efficiently.

Dataset

The dataset used for this project is a collection of images representing different road surface conditions. The dataset is organized into three classes: “good”, “poor”, and “very_poor”.

Preprocessing

The raw images undergo a series of preprocessing steps to prepare them for training. These steps are as follows:

- **Image Resizing**: All images are resized to a uniform resolution of 224x224 pixels.
- **Data Splitting**: The dataset is split into training (80%), validation (10%), and testing (10%) sets.
- **Data Augmentation**: To increase the diversity of the training data and prevent overfitting, the following data augmentation techniques are applied to the training set:
  - Brightness and contrast variation
  - Random rotation
  - Horizontal flipping

Methodology

Model Architecture

The model used is a Convolutional Neural Network (CNN) built with TensorFlow/Keras. The architecture is as follows:

- A series of convolutional layers with increasing filter sizes (32, 64, 128, 256), each followed by Batch Normalization, ReLU activation, and MaxPooling.
- Dropout layers are used after each convolutional block to prevent overfitting.
- The output of the convolutional layers is flattened and passed through a series of Dense layers with ReLU activation.
- The final output layer is a Dense layer with a softmax activation function to produce a probability distribution over the three classes.

Training

The model is trained using the Adam optimizer and the categorical cross-entropy loss function. The learning rate is adjusted during training using a `ReduceLROnPlateau` callback, which reduces the learning rate when the validation loss stops improving. Early stopping is also used to prevent overfitting and save the best model based on validation accuracy.

Hyperparameters

- **Input Shape**: (224, 224, 3)
- **Number of Classes**: 3
- **Optimizer**: Adam
- **Loss Function**: Categorical Crossentropy
- **Epochs**: 40 (with early stopping)
- **Batch Size**: 32

Results

The performance of the trained model is evaluated on the test set with accuracy of **98%**. The following visualizations from the `results/` directory are included to summarize the results.

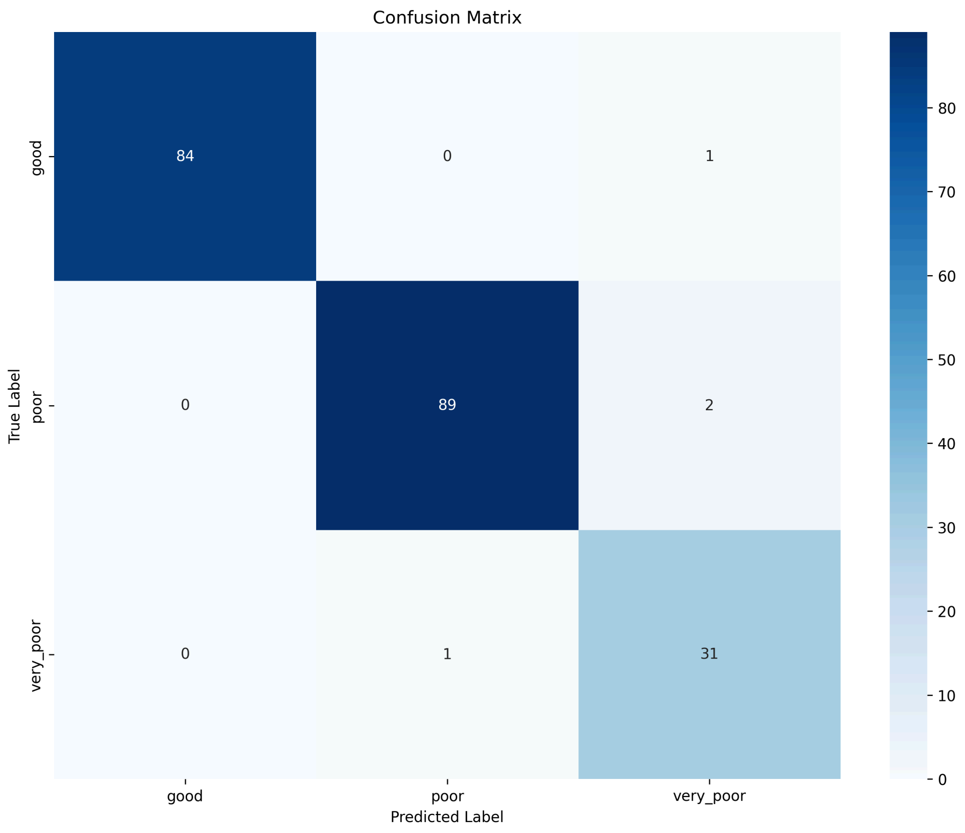

Figure 1: Confusion Matrix

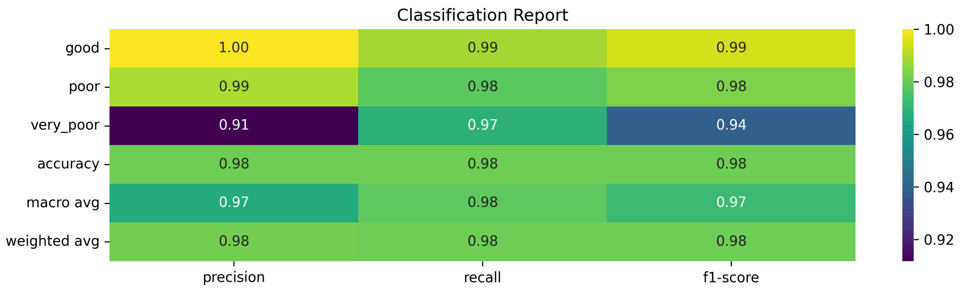

Figure 2: Classification Report

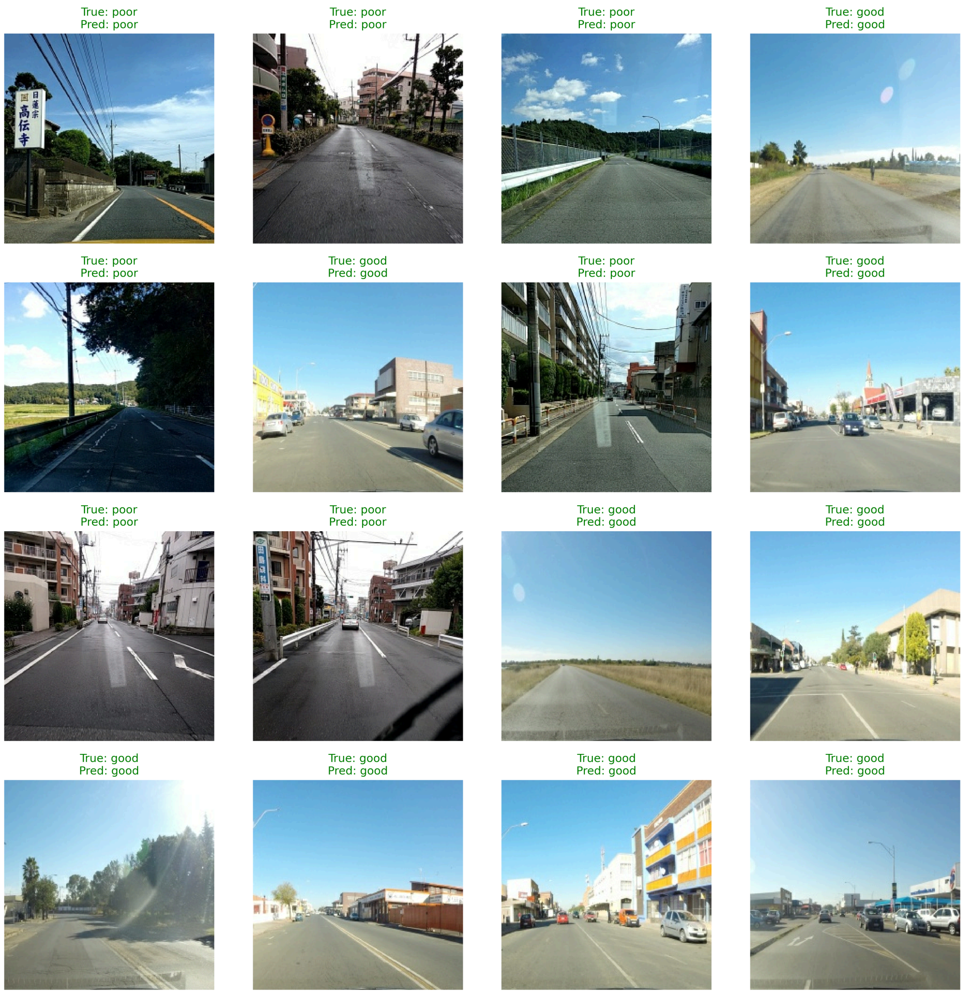

Figure 3: Sample Predictions

Training Curves

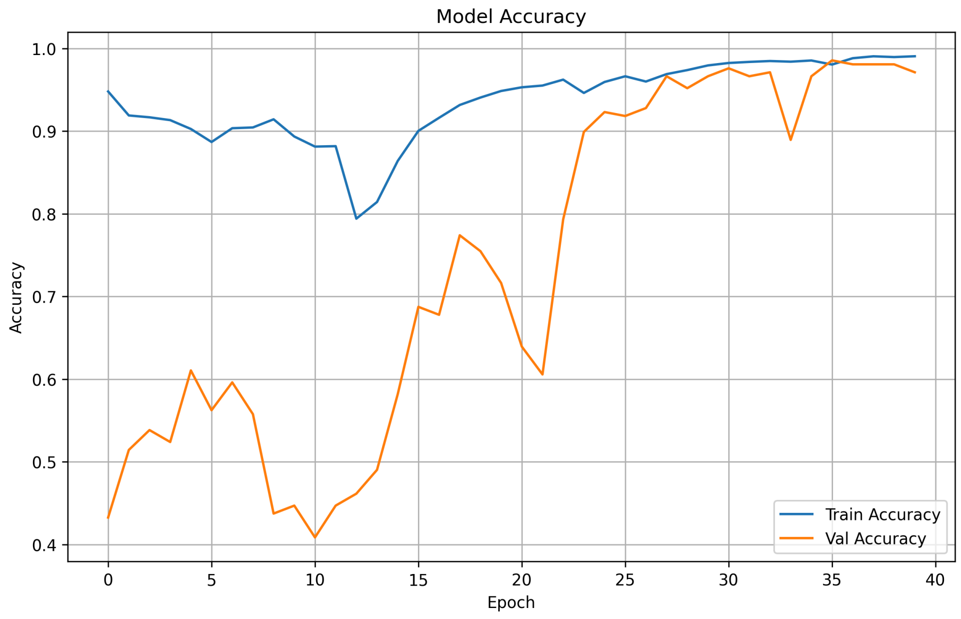

Figure 4: Accuracy Curve

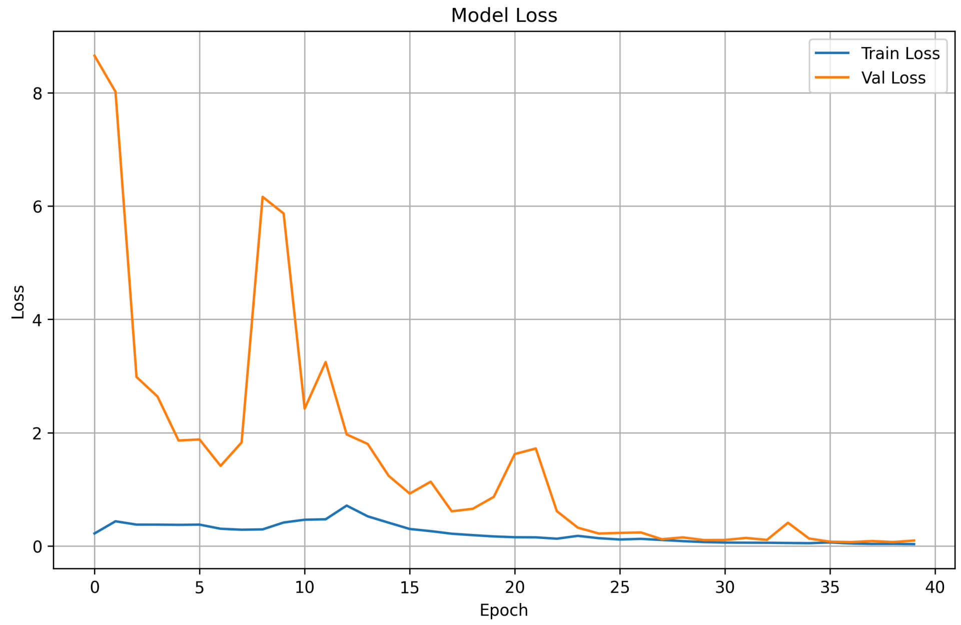

Figure 5: Loss Curve

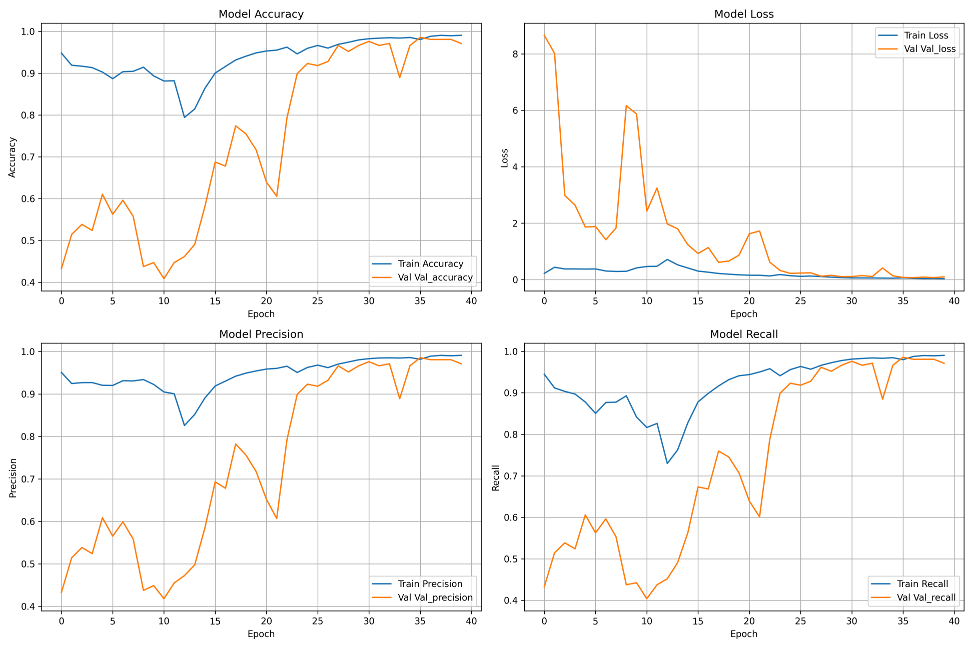

Figure 6: Training History

Discussion

One of the main challenges encountered during this project was related to the initial dataset. The original dataset included a “satisfactory” class that was visually very similar to the “poor” class. This led to inconsistent labeling, where images could arguably belong to either category, which in turn negatively impacted the model’s accuracy. To address this, a data preprocessing step was implemented to merge the “satisfactory” class into the “poor” class. This simplification of the classification problem into three more distinct classes (“good”, “poor”, and “very_poor”) resulted in a more robust and accurate model.

The data augmentation techniques, including variations in brightness and contrast, random rotations, and horizontal flipping, proved to be effective in increasing the diversity of the training data and mitigating overfitting. The chosen CNN architecture, with its series of convolutional layers, batch normalization, and dropout, was well-suited for this image classification task. The use of callbacks like `ReduceLROnPlateau` and `EarlyStopping` was also beneficial for optimizing the training process and preventing unnecessary training epochs.

Limitations and Future Work

While the model performs well, there are several avenues for future improvement. The model’s performance could be further enhanced by training on a larger and more varied dataset, covering a wider range of road conditions and environmental factors (e.g., different weather conditions, lighting, and seasons).

Experimenting with more advanced CNN architectures, such as utilizing transfer learning with pre-trained models like ResNet, VGG, or EfficientNet, could also lead to significant performance gains. These models, having been trained on massive datasets like ImageNet, have learned rich feature representations that can be fine-tuned for this specific task.

Furthermore, the current system classifies static images. A more advanced application would involve real-time road condition classification from a video feed. This would require a more complex system capable of processing sequential data and could provide more dynamic and timely information for applications like autonomous driving.

References

Data Set: https://www.kaggle.com/datasets/prudhvignv/road-damage-classification-and-assessment/data