Traffic Sign Detection and Classification Using Faster R-CNN

Abstract

In this project, we will design a simple and reliable system that will spot and name common road signs—like stop signs, speed limit signs, and no-entry signs—in photos or video from a car's camera. We plan to build on a well-known detection method called Faster R-CNN, which has been trained on a wide range of objects. By fine-tuning it with traffic sign examples and applying basic image tweaks (like adjusting brightness, adding slight blur, or covering parts of signs), our system will learn to recognize over 35 types of signs. We expect it to correctly find and label signs roughly three-quarters of the time (around 75–80%) and to process video at about 10 frames per second on a standard GPU. All of our code, sample data, and setup instructions will be shared publicly so others can try or improve our work.

Objectives

- **Gather Data:** We will collect traffic sign images from three public datasets (Germany, the U.S., and Belgium).
- **Standardize Labels:** We will convert different label formats into a single, common format.
- **Train the Model:** We will adapt and retrain a Faster R-CNN model to recognize our traffic sign categories.
- **Evaluate Performance:** We will measure how often the system finds signs (recall) and how often its labels are correct (precision), aiming for about 75–80%.
- **Demonstrate Real-Time Use:** We will set up a simple demo that shows live detection on dashcam video.
- **Share Our Work:** We will release all scripts, data samples, and clear instructions so others can reproduce or build on our project.

Methodologies

1. **Data Preparation:** Download and review images and labels; resize all images to the same size; normalize pixel values; apply simple augmentations like flips, rotations, brightness changes, and slight blur or occlusions to make the model more robust.
2. **Model Setup:** Use a popular library's Faster R-CNN implementation; swap in our traffic sign category list; choose training settings such as learning rate, batch size, and number of steps.
3. **Training Process:** Train the model on our prepared data, periodically checking its performance on a separate validation set to monitor learning progress.
4. **Performance Evaluation:** Calculate recall and precision on a test set, and generate easy-to-read charts or reports showing how well the model performs.
5. **Optimization and Export:** Convert the trained model to a format that runs faster (e.g., using TorchScript) and test it on a standard GPU to reach around 10 frames per second.

Key Findings

- **Detection Accuracy:** We expect the system to correctly identify traffic signs in about 75–80% of cases.
- **Classification Accuracy:** We expect that when a sign is found, it will be labeled correctly around 75–80% of the time.
- **Processing Speed:** After optimization, we anticipate real-time video processing at approximately 10 frames per second on a regular GPU.
- **Robustness:** We believe that basic image augmentations will help the system handle different lighting conditions, slight motion blur, and partially covered signs.

Implementation Steps

1. **Setup Environment:** Install Python, the deep learning library, and other required tools.
2. **Annotation Conversion:** Write a script to convert existing labels into one unified format (e.g., COCO JSON).
3. **Augmentation Scripts:** Create simple functions to apply flips, rotations, brightness tweaks, and blur to images.
4. **Training Pipeline:** Write code that loads data, trains the model, and logs performance metrics.
5. **Evaluation Tools:** Implement scripts to calculate precision and recall and to visualize results.
6. **Model Export and Demo:** Export the final model for fast inference and build a demo script that shows live detection on video.
7. **Documentation and Release:** Prepare a clear README with step-by-step instructions and publish everything on a public repository.

References

- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
- Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., & Igel, C. (2013). The German Traffic Sign Detection Benchmark (GTSDB).

Team

- Shubh Sonakiya – E22CSEU1290
- Ayush Sharma – E22CSEU1271
- Utkarsh Goel – E22CSEU1396