Computer Vision System for Real-Time Traffic Sign Analysis

Ashraf Ali Khaja Basheer - 50539674 - ashrafal@buffalo.edu
Supraja Ashokbabu - 50496371 - suprajaa@buffalo.edu

Overview of the Project

Application

This project focuses on developing a Real-Time Traffic Sign Recognition and Classification (RTTSRC) system. The system aims to accurately identify and classify traffic signs in real-time, leveraging advanced computer vision and deep learning models. The main inputs to the system are images captured by on-road cameras, and the output is the classification of traffic signs present in the images. This application is critical for improving road safety, reducing accidents, and supporting autonomous vehicle navigation.

The RTTSRC system benefits multiple stakeholders: it aids autonomous vehicle manufacturers by improving navigation systems, supports transport authorities in traffic management and compliance monitoring, enhances driver assistance systems (ADAS) with real-time alerts, and promotes road safety by minimizing accidents caused by missed or misinterpreted signs. This technology serves as a critical tool for advancing modern transportation systems.

Inputs and Outputs:

Inputs:

- Images or video frames of traffic signs (640x640 pixels)

Outputs:

- Bounding boxes around detected traffic signs
- Classification labels for each detected sign
- Confidence scores for detections and classifications

Intermediate outputs:

- Feature maps from object detection
- Extracted features from the modified CNN model - likely custom ResNet model
- Build an hybrid model, that involves a fusion of ResNet and a Vision Transformer

State of the Art

The state-of-the-art in traffic sign recognition integrates real-time object detection with classification models. YOLO (You Only Look Once) is widely used for its ability to detect multiple objects in a single pass with high speed and accuracy. For classification, hybrid fusion models combining Convolutional Neural Networks (CNNs) like ResNet and Transformers such as Vision Transformers (ViT) have proven to be the most effective.

Our approach represents the best practice in this field. We developed and evaluated four models — VGG-13, ResNet34, Vision Transformer (ViT), and a hybrid ResNet-ViT fusion model. The two best-performing models, ResNet34 and ViT, were fused into a unified classification model. By leveraging ResNet's hierarchical feature extraction and ViT's global context understanding, our hybrid model achieves superior classification accuracy and robustness.

YOLO V11 is used for real-time detection of traffic signs before classification. Detected traffic signs are cropped and passed to the hybrid classification model, ensuring real-time processing of 55 distinct classes with high precision. This combined approach outperforms traditional methods in terms of accuracy, robustness, and real-time applicability.

Novelty:

The fusion model offers a unique approach to traffic sign recognition by combining ResNet34 and Vision Transformer (ViT) architectures. ResNet34 excels in extracting local hierarchical features with its deep residual blocks, while ViT captures global context through self-attention mechanisms. By fusing these strengths, the model achieves a richer representation of traffic sign images, leading to improved recognition accuracy and robustness. Along with the object Detection capabilities of the YOLO model. This approach goes beyond standard classification techniques, offering a novel solution that advances the state of the art in traffic sign analysis.

Pipeline : Dataset - > Train YOLO -> Train Hybrid Model -> Real Time Images -> YOLO Object Detection -> Cropped Dataset -> Hybrid Model Classification on it

Improves the accuracy. All the findings are attached in the Python Notebook and in the report.

Contributions

We initially tested several models, including ResNet34, VGG13, and ViT, to determine which would perform best for traffic sign detection. After evaluating their strengths, we decided to combine ResNet34 and ViT into a hybrid fusion model, which significantly improved both accuracy and robustness. For real-time object detection, we used YOLO, allowing us to efficiently detect and crop traffic signs from images with 55 different classes. This approach provided fast processing while maintaining high accuracy. The system also performed reliably under various conditions, such as changes in lighting, weather, and occlusions, ensuring dependable detection in real-world scenarios. The cropped images from YOLO is sent to the Hybrid model loaded with previously trained weights and then classified.

Approach:
Models Used:

VGG-13

1. **Description & Relevance**: We also experimented with VGG-13, a more traditional convolutional neural network. VGG-13 is known for its simple yet effective design, making it a good baseline model for image classification tasks. We decided to include it to compare its performance with more advanced models like ResNet-34.

2. **Implementation Details**: VGG-13 uses a sequence of convolutional layers followed by max-pooling and fully connected layers. We implemented it to extract features from the traffic sign images, with the final output layer classifying the extracted features into one of the 55 traffic sign categories.

3. **Pros and Cons**:
    - **Pros**: VGG-13 was easy to implement and demonstrated good flexibility in terms of input image sizes. It provided solid results in terms of feature extraction.
    - **Cons**: The main downside we encountered with VGG-13 was its computational cost. Its deep architecture required a lot of memory and processing power, which slowed down

training times. Additionally, we found that its performance wasn't as strong on more complex tasks compared to newer models like ResNet-34.

ResNet-34

1. **Description & Relevance**: ResNet-34, which uses residual learning, was another model we tested. This architecture is specifically designed to avoid the vanishing gradient problem in deep networks, making it ideal for tasks requiring deep feature learning, such as traffic sign classification.

2. **Implementation Details**: ResNet-34 uses residual blocks with 3x3 convolutional layers, followed by adaptive average pooling and a fully connected layer for classification. We found that its deep architecture helped us extract more complex features from traffic sign images, improving classification accuracy.

3. **Pros and Cons**:
    - **Pros**: The deep architecture of ResNet-34 allowed it to learn more complex features from the images, which resulted in higher accuracy for traffic sign classification.
    - **Cons**: However, this depth came at the cost of high computational requirements. During training, ResNet-34 was slower compared to simpler models like VGG-13, and it also required more memory, which made it less efficient on machines with limited resources.

Vision Transformer (ViT)

1. **Description & Relevance**: We also experimented with the Vision Transformer (ViT), which applies transformer based architecture to image classification tasks. ViT is known for capturing global dependencies in images, which made it an interesting option for traffic sign classification, where relationships across the entire image can be important.

2. **Implementation Details**: ViT splits the input image into patches, which are then embedded and processed by a transformer encoder. The self-attention mechanisms used by ViT allow it to capture long-range dependencies, which is especially useful for tasks like traffic sign classification.

3. **Pros and Cons**:
    - **Pros**: The main advantage of ViT is its ability to capture global relationships within the image, which can improve classification accuracy for complex images.
    - **Cons**: One major drawback we encountered was that ViT requires large datasets and significant computational power to perform well. Training ViT from scratch on a smaller dataset took much longer than the CNN models, and its performance wasn't as strong without a massive amount of data to train on.

ViT-ResNet34 Fusion Model

1. **Description & Relevance**: After evaluating individual models, we decided to combine ResNet-34 and ViT in a hybrid fusion model. The idea was to leverage the strengths of both models ResNet-34's ability to capture local features and ViT's strength in capturing global dependencies to improve our traffic sign classification performance.

2. **Implementation Details**: We fused the output of ResNet-34, which excelled in local feature extraction, with ViT's embeddings, which captured long-range dependencies. This hybrid model gave us a comprehensive understanding of the images, enhancing classification accuracy and robustness.

3. **Pros and Cons**:
    - **Pros**: By combining both models, we gained the best of both worlds. The fusion model was able to extract detailed local features and understand global relationships within the images, resulting in higher accuracy and robustness, especially in complex scenarios.
    - **Cons**: However, the fusion model introduced additional complexity, making it more computationally expensive. Training the hybrid model took longer, and it required more memory and processing power, which could be a limitation on lower end hardware.

YOLO-V11 (Object Detection)

1. **Description & Relevance**: For real-time traffic sign detection, we used YOLO-V11, a powerful object detection algorithm. It was a perfect fit for our project because it allows us to detect and classify traffic signs across 55 categories quickly, even in images with varying object sizes. YOLO-V11's architecture, which builds on previous versions, is designed to be fast and accurate, making it ideal for our real-time detection needs.

2. **Implementation Details**: We utilized the pretrained weights of YOLO-V11, which significantly sped up our development process. By using these weights, we were able to adapt the model to our traffic sign detection task with minimal fine tuning, ensuring we could efficiently detect and classify the traffic signs in the images. Then we crop the detections, and we send it to the Vision Transformer x ResNet Hybrid Model.

3. **Pros and Cons**:
    - **Pros**: YOLO-V11's real-time detection capabilities were crucial for the project, as it processed images swiftly without compromising accuracy. It worked well across different traffic sign sizes, making it versatile for our dataset.
    - **Cons**: While the speed was impressive, we found that YOLO-V11 struggled with detecting very small or occluded objects. The model's computational demands were also quite high, requiring powerful hardware, especially during real-time processing.

Self-Coded Aspects and its contribution to the Project:

In our project, we took a hands-on approach by coding several key components from scratch. First, we designed and implemented the **fusion model architecture**, which combined ResNet-34 and ViT to create a powerful hybrid model for traffic sign classification. We had to integrate these two models carefully, ensuring that their strengths—ResNet-34's local feature extraction and ViT's global dependency capturing—complemented each other. This fusion model was essential for improving the overall accuracy and robustness of our system.

Next, we created **custom preprocessing scripts** to handle the output from YOLO-V11. YOLO-V11 was used for detecting traffic signs, but we had to preprocess its output to prepare it for classification by our models. These preprocessing steps ensured that the data flowed seamlessly through our system, making it more efficient and allowing for accurate classification.

We also **modified the architectures of VGG-13, ResNet-34, and ViT** to suit our traffic sign classification task. The changes we made to these models were necessary to improve their performance on our dataset, as the original versions of these models weren't fully optimized for this specific use case.

Aspects from Online Resources:

- YOLO-V11 pretrained weights and architecture were utilized, with modifications made for compatibility with the dataset.

- The VGG-13 architecture and ResNet-34 model were adapted from publicly available implementations. The ViT model was used based on the original Vision Transformer research paper.
- Appropriate citations for YOLO-V11's documentation, the ViT research paper, and ResNet-34 and VGG-13 repositories were included.

Experimental Protocol

Datasets:

We used the " V2 - Traffic Signs [https://www.kaggle.com/datasets/raduoprea/traffic-signs] ", which contains 55 traffic sign classes with 640x640 pixel images. The dataset has an unbalanced class distribution, which made the classification task more challenging. To prepare the data, we cropped images based on bounding box annotations. During training, we applied transformations such as Gaussian blur, normalization of images, and resizing to improve consistency and generalization. The dataset was split into training and validation sets with proper labeling for each class. There was an oversampling issue in some classes so we resampled the dataset and are using the dataset "Updated-Dataset.zip" for our YOLO model.

Evaluation Metrics:

To evaluate the success of our models, we used a combination of both quantitative and qualitative metrics. Quantitatively, we relied on accuracy, precision, recall, and F1-score to assess the models' performance in classifying traffic signs across 55 distinct categories. These metrics helped us measure how accurately the models identified the traffic signs and their ability to handle class imbalances. On the qualitative side, we used a confusion matrix to visually inspect how well the models performed on individual traffic sign classes. This gave us valuable insights into areas where the models excelled and where they struggled, allowing us to fine tune them for better performance.

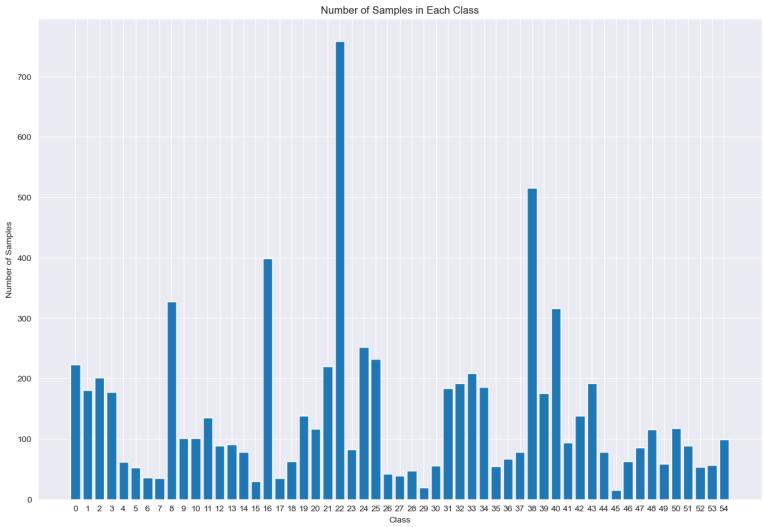
Compute Resources:

For the training and evaluation of our models, we utilized an NVIDIA RTX 3080 GPU with CUDA support to take advantage of parallel processing and speed up computations. Training multiple deep learning models, including YOLO-V11, VGG-13, ResNet-34, Vision Transformer, and their fusion model, was computationally intensive. The GPU and PyTorch framework were crucial for handling the large datasets and training times, ensuring we could experiment with different models and hyperparameters efficiently. One challenge we encountered was the high memory consumption during model training, especially when working with larger architectures like ViT, which made it difficult to run on machines with limited resources.

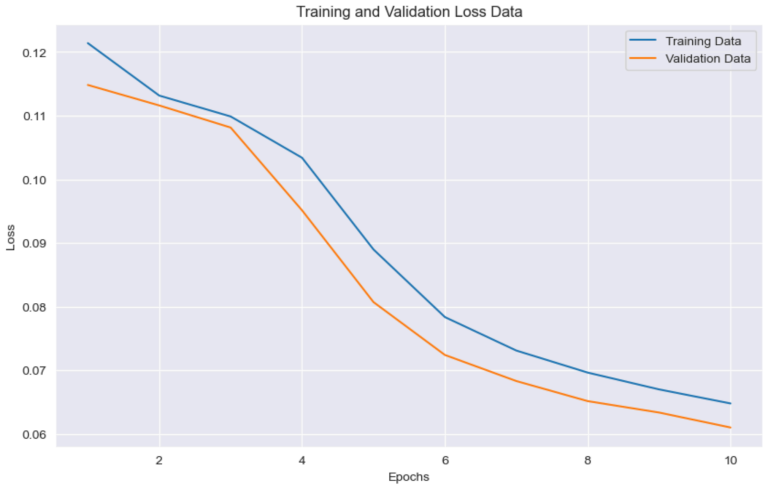
Results:

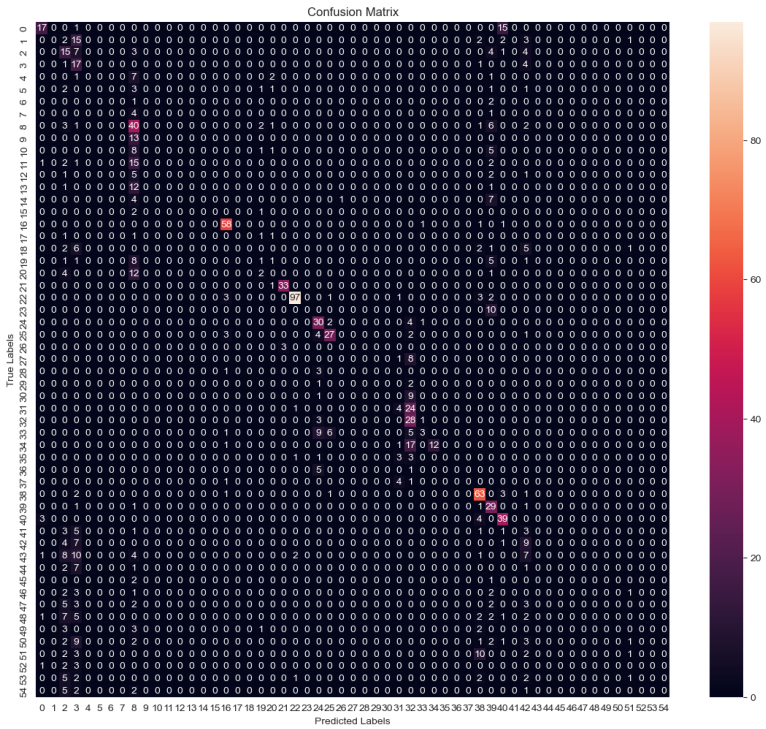
Qualitative Results

Cropped traffic sign from the dataset, with the predicted class labels



VGG 13 Model

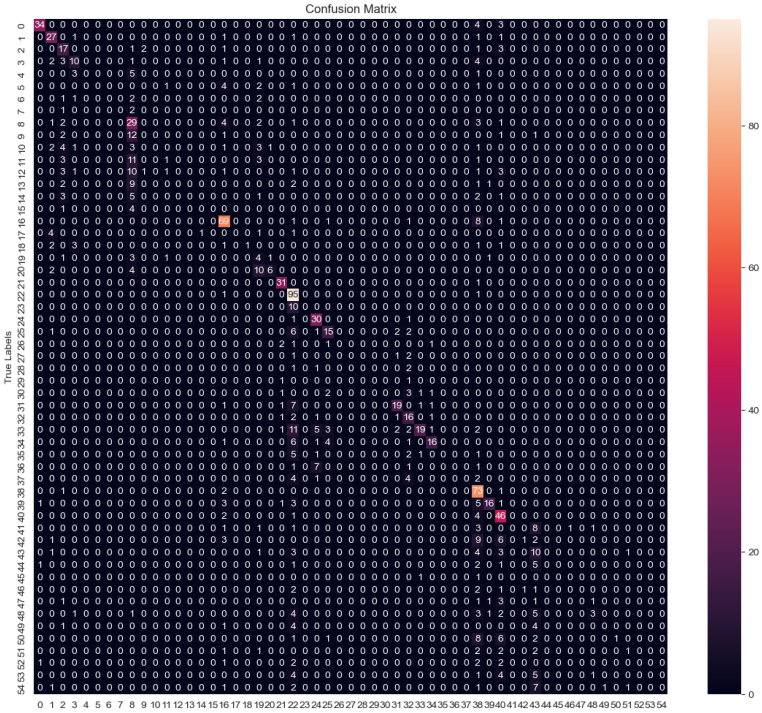




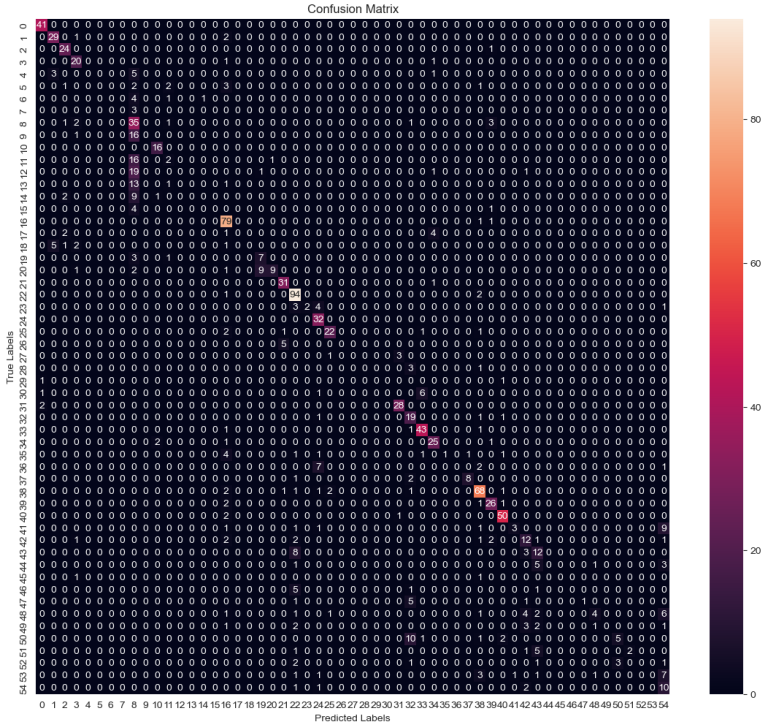
ResNet34 Model


Training and Validation Accuracy Data
Training Data
Validation Data
Accuracy
Epochs

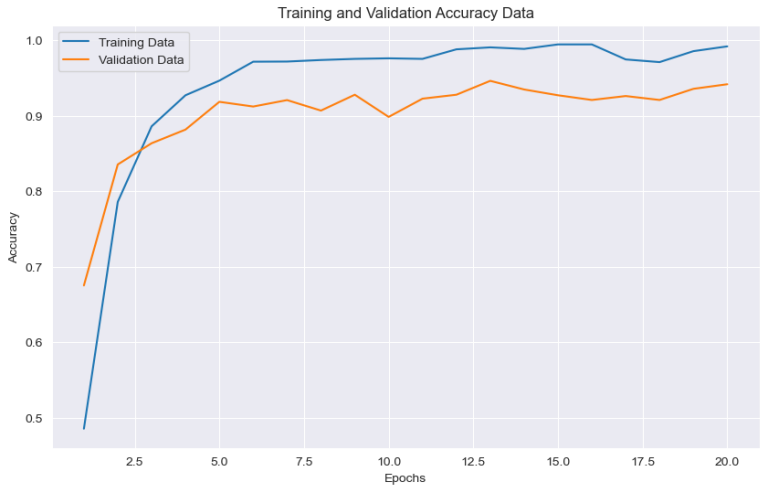

Training and Validation Loss Data
Training Data
Validation Data
Loss
Epochs

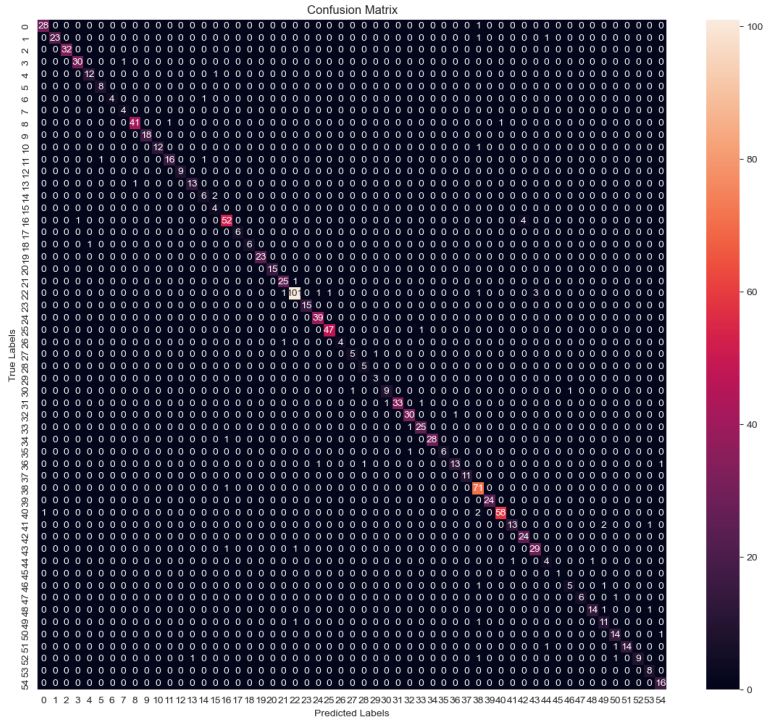

Confusion Matrix
True Labels
Predicted Labels

Vision Transformer Model



VIT ResNet Hybrid Model





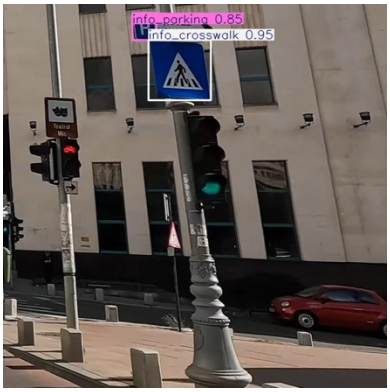
YOLO V11

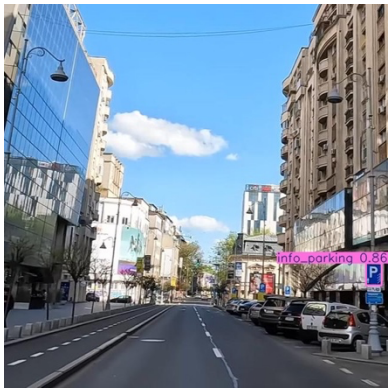
Predictions from the runs based on YOLO source (Auto generated):

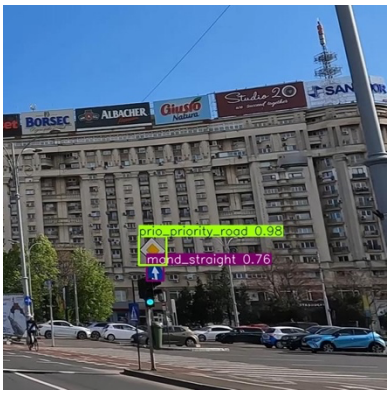

forb_right 0.93
prio_give_way 0.87

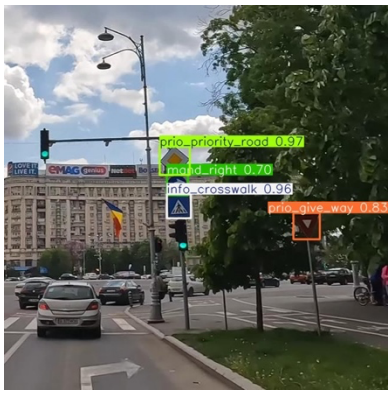

forb_ahead 0.99
info_parking 0.95


forb_stopping 0.86

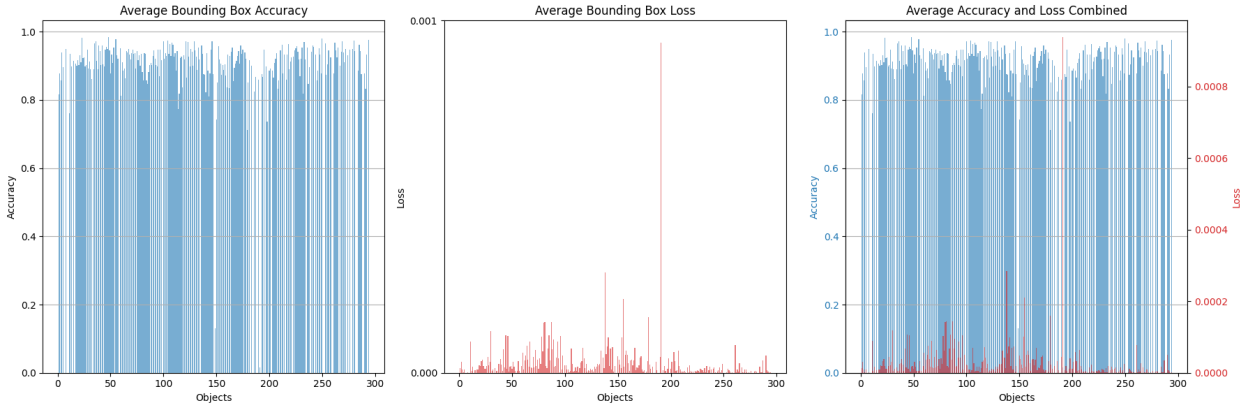

info_parking 0.85
info_crosswalk 0.95


info_parking 0.86


BORSEC
ALBACHER
Giusto
Studio 20
prio_priority_road 0.98
mand_straight 0.76

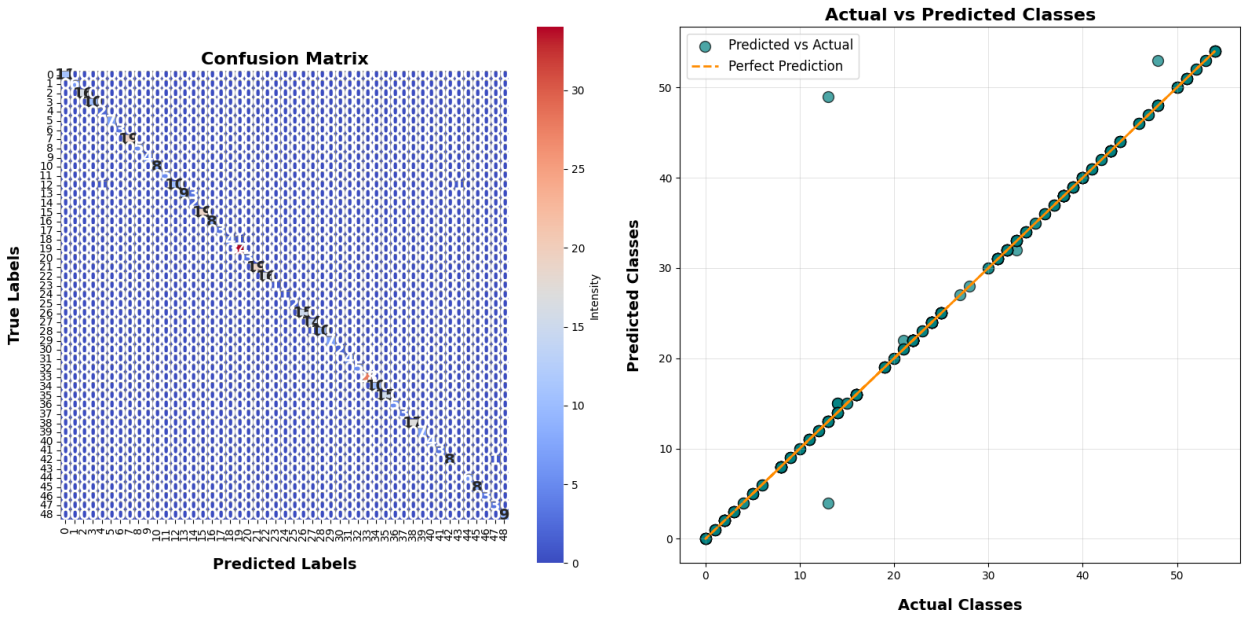

prio_priority_road 0.97
mand_right 0.70
info_crosswalk 0.96
prio_give_way 0.83


60
info_one_way_traffic 0.85
info_crosswalk 0.91



ViT x ResNet on the images from YOLO

Overall classification accuracy on test images: 98.1% which is slightly better than the pipeline without YOLO model.



Interpretation about the results:
VGG-13 showed gradual improvement but had modest accuracy, reaching 44.5% validation accuracy by epoch 10. ResNet-34 performed better, achieving 51.8% validation accuracy by epoch 10. ViT outpaced both, with validation accuracy reaching 64.8% by epoch 10.

The Fusion model, combining ResNet-34 and ViT, delivered the best results, starting strong with 67.5% validation accuracy in the first epoch and reaching 94.2% by epoch 10. It proved the most effective, highlighting the benefits of model fusion for improved performance.

The Fusion model performed significantly better than the individual models, achieving an accuracy of 94.25%, with high precision, recall, and F1 score across most classes. ViT also showed strong performance with an accuracy of 66.20%. ResNet-34 and VGG-13, while performing better than expected

in some areas, had relatively lower accuracies (51.39% and 45.56%, respectively), and their precision and recall varied greatly across different classes.

Conclusion:
Overall, model fusion proved to be the most effective approach for maximizing performance.

Analysis

Advantages

1. The hybrid ResNet + ViT model effectively combined local feature extraction with global attention mechanisms.
2. YOLOv11 provided robust real-time detection capabilities.

Limitations

1. Class imbalance in the dataset significantly impacted classification performance for underrepresented classes.
2. VGG13's limited capacity hindered its ability to capture complex patterns in traffic signs.
3. High computational requirements for YOLOv11 and the hybrid model posed challenges for deployment on resource-constrained devices.

Discussion and Lessons Learned

In this project, we integrated multiple models for traffic sign classification and detection. Here are the key lessons we learned:

1. **Model Fusion**: Combining ResNet and ViT significantly improved accuracy by leveraging their complementary strengths in feature extraction and attention mechanisms.
2. **Preprocessing is Crucial**: Techniques like cropping based on bounding boxes ensured consistency and smooth integration between models.
3. **Class Imbalance**: The dataset imbalance affected performance, especially for underrepresented classes. Addressing this is critical for better model generalization.
4. **Efficiency Challenges**: While the hybrid ResNet + ViT model performed well, its high computational demand posed challenges for deployment on resource-limited devices.

Future Work

For future improvements, we plan to:

- Expand the model to detect road markings and traffic lights, making it more versatile for real-world applications.
- Use advanced data augmentation and ensemble methods to improve model performance under varied conditions.
- Explore model compression techniques to reduce computational requirements, enabling real-time processing on resource-constrained devices.

Bibliography

A. Bochkovskiy, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020. [Online]. Available: https://github.com/AlexeyAB/darknet.

K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv preprint arXiv:1409.1556, 2014. [Online]. Available: https://arxiv.org/abs/1409.1556.

K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778. [Online]. Available: https://arxiv.org/abs/1512.03385.

A. Dosovitskiy and T. Brox, "Inverting Convolutional Architectures for Image Segmentation," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016. [Online]. Available: https://arxiv.org/abs/2010.11929.