Traffic Sign Recognition Using CNNs

SEP740 DEEP LEARNING

Kenil Sachapara (400551600)

Om Kakadiya (400551508)

Yash Patel (400551627)

Jaimis Miyani (400551743)

Github link: TSR

Under Guidance of

Anwar Mirza

Introduction

- **Overview:**
  - The project aims to develop an accurate system for identifying traffic signs using CNNs.
  - Importance of traffic sign recognition for autonomous driving and ADAS.
- **Objectives:**
  - Prepare and augment data for diversity and quality.
  - Build a CNN architecture with batch normalization, dropout, pooling, and convolution layers.
  - Train and evaluate the model using metrics such as accuracy, confusion matrices, and categorical cross-entropy loss.
  - Learn different method of implementation to solve this scenario in real life.

Importance of Traffic Sign Recognition

- **Autonomous Driving:**
  - Ensures Safe Navigation: Traffic signs provide critical information that helps autonomous vehicles navigate roads safely, such as speed limits, stop signs, and yield signs.
  - Adherence to Traffic Laws: Recognizing and obeying traffic signs ensures that autonomous vehicles follow legal road requirements, reducing the risk of violations and accidents.
- **Advanced Driver Assistance Systems (ADAS):**
  - Enhances Driver Safety: ADAS uses traffic sign recognition to alert drivers about important road information, helping them stay aware and respond appropriately to traffic conditions.
  - Automating Driving Functions: Systems like adaptive cruise control and lane-keeping assist rely on traffic sign recognition to adjust vehicle speed and maintain safe distances based on current road signs.
  - Driver Fatigue Reduction: By automating responses to traffic signs, ADAS reduces the cognitive load on drivers, allowing them to stay focused and reduce fatigue during long drives.

Project Structure

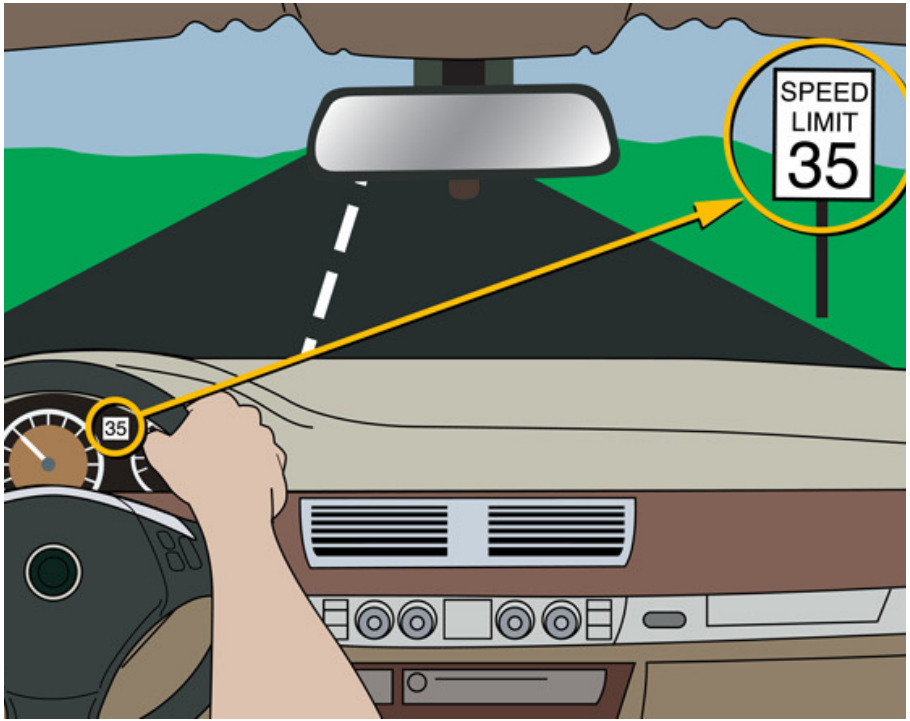
- **Data Preparation:**
  - Augmentation: Rotation, shifts, shear, zoom, flip, brightness adjustments.
  - Purpose: Increase data diversity for better model performance.
- **CNN Architecture:**
  - Layers:
    - Convolutional (features), ReLU (activation), Pooling (reduce dimensions), Dropout (prevent overfitting), Fully Connected (classification), Softmax (probabilities).
- **Evaluation:**
  - Metrics:
    - Accuracy, Confusion Matrix, Loss, Precision, Recall, Visualizations (track progress).

Problem Statement

- **Need:**
  1. Safety: Crucial for the safe operation of autonomous vehicles to prevent accidents.
  2. Compliance: Ensures adherence to traffic laws and regulations.
- **Challenges:**
  1. Distorted or Obscured Signs: Recognizing traffic signs that are partially hidden, damaged, or at unusual angles.
  2. Environmental Conditions: Handling variations in lighting (day/night, shadows) and weather (rain, fog, snow) that affect visibility and recognition accuracy.

Project Goals

- **High-Precision Identification:**
  - CNN Utilization: Employ CNNs to accurately classify traffic signs within the dataset, providing a foundation for future real-world implementation.
- **Comprehensive Dataset:**
  - GTSRB Dataset: Utilize the German Traffic Sign Recognition Benchmark (GTSRB) dataset for robust training and evaluation, ensuring the model learns from diverse and realistic traffic sign scenarios.



Literature Review

- **Traditional Methods:**
  - SVM and k-NN:
    - SVM (Support Vector Machines): Effective for binary classification but struggles with multi-class problems and large datasets.
    - k-NN (k-Nearest Neighbors): Simple and intuitive but becomes computationally expensive with large datasets and lacks feature learning capability.
- **Advantages of CNNs:**
  - Enhanced Accuracy: CNNs automatically learn hierarchical features from images, leading to superior accuracy in classification tasks.
  - Robustness: Able to handle variations in lighting, angles, and partial occlusions better than traditional methods.
- **Why CNNs:**
  - Feature Extraction: CNNs excel at automatically detecting and learning relevant features from raw image data, eliminating the need for manual feature engineering.
  - Scalability: Scalable to large and complex datasets, making them suitable for real-world applications.
  - End-to-End Learning: Capable of learning directly from the input data to the output labels, simplifying the modeling process and improving performance.

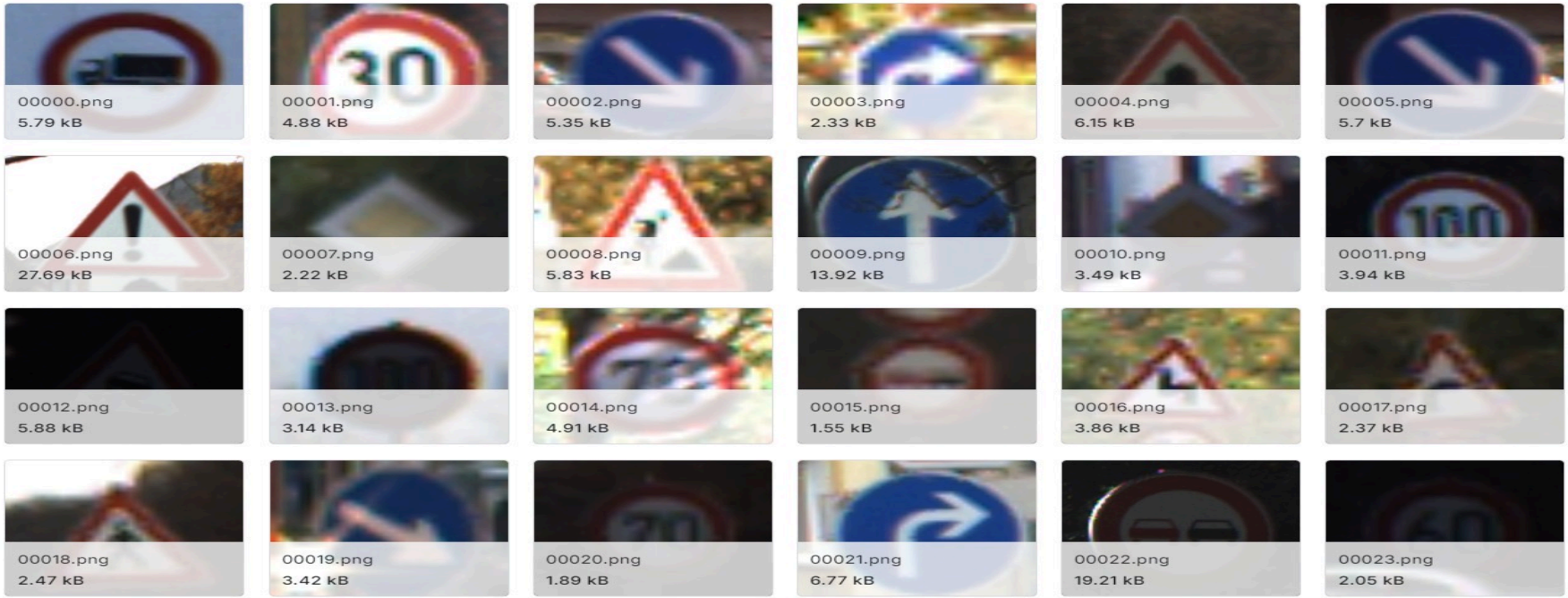
Previous Research

- **Studies on Traffic Sign Recognition:**
  - Successful Implementations:
    - GTSRB Challenge: Various studies using the GTSRB dataset have demonstrated high accuracy in traffic sign recognition, with some models achieving over 98% accuracy.
    - Real-World Applications: Research by companies like Tesla and Waymo has shown the practical implementation of CNNs in autonomous driving systems, successfully identifying and responding to traffic signs in diverse environments.
- **Challenges Addressed:**
  - Class Imbalance:
    - Techniques: Addressed through data augmentation and synthetic data generation to ensure all traffic sign classes are adequately represented.
  - Varied Conditions:
    - Solutions: Tackled by augmenting training data to simulate different lighting, weather, and occlusion scenarios, helping models perform reliably under diverse real-world conditions.

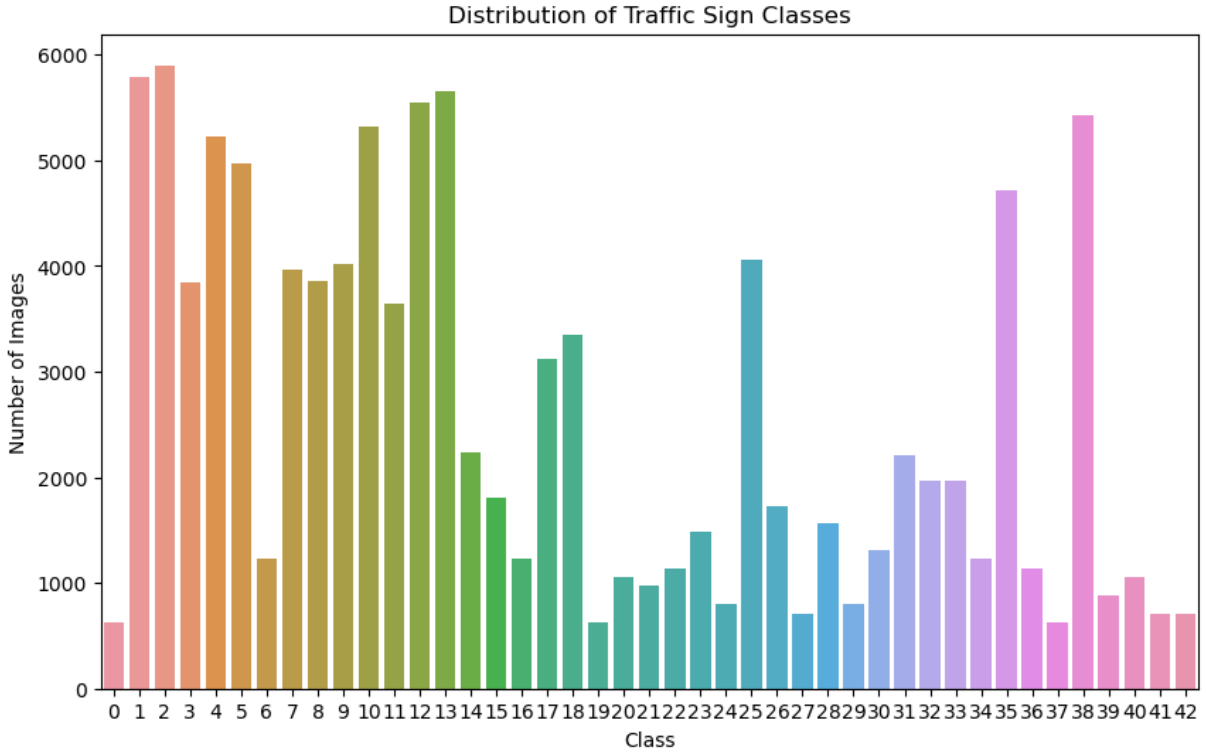
Dataset Overview

- **GTSRB Dataset:**
  - Description: The German Traffic Sign Recognition Benchmark (GTSRB) dataset contains over 50,000 images of traffic signs in more than 43 categories. This dataset includes a variety of traffic signs captured in different conditions, providing a comprehensive foundation for training and evaluating traffic sign recognition models.
- **Class Examples:**
  - Speed Limit Signs (e.g., 30km/h, 50km/h, 70km/h)
  - Prohibitory Signs (e.g., No passing, No entry)
  - Warning Signs (e.g., Dangerous curve, Slippery road)
  - Mandatory Signs (e.g., Turn right, Go straight)
  - Priority Signs (e.g., Stop, Yield)
- **Examples:**
  - Here is included images of speed limits, warning signs, and stop signs from the dataset to show diversity and complexity.

Dataset Overview



Dataset


Distribution of Traffic Sign Classes
Number of Images
Class
0
1000
2000
3000
4000
5000
6000
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42

Theory of CNNs

- **Multi-layer Models:**
  - Designed for structured grid data: CNNs are specifically designed to process structured grid data, such as images, where the spatial arrangement of pixels is crucial.
- **Components:**
  - Convolutional Layers: Extract local features by applying filters that slide over the input image to detect patterns like edges, textures, and shapes.
  - Activation Functions: Introduce non-linearity into the model, allowing it to learn complex patterns. Commonly used functions include ReLU (Rectified Linear Unit).
  - Pooling Layers: Reduce the spatial dimensions of the feature maps, which helps in decreasing the computational load and preventing overfitting while retaining essential information. Examples include max pooling and average pooling.
  - Fully Connected Layers: Combine the features extracted by the convolutional layers for final classification, where each neuron is connected to every neuron in the previous layer, enabling the model to learn global patterns.

Regularization

- **Regularization Techniques:**
  - Batch Normalization:
    - Normalizes the input of each layer, ensuring stable and faster training.
    - Helps in reducing the internal covariate shift, making the model more robust.
  - Dropout:
    - Randomly drops neurons during training to prevent the model from becoming overly reliant on specific neurons.
    - Helps in reducing overfitting by forcing the model to learn redundant representations.

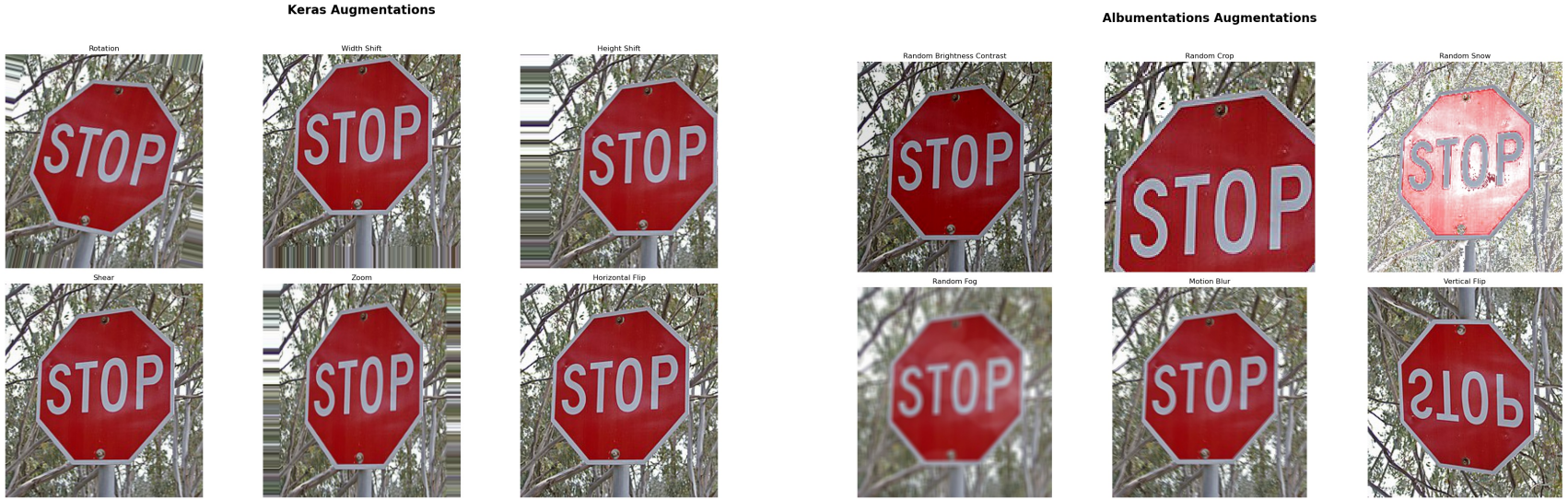
Evaluation Metrics

- **Accuracy:**
  - Definition: The ratio of correctly predicted instances to the total instances.
  - Purpose: Provides a straightforward measure of the model's overall performance, indicating how often the model correctly identifies traffic signs.
- **Confusion Matrices:**
  - Definition: A matrix that shows the true positive, true negative, false positive, and false negative predictions for each class.
  - Purpose: Offers a detailed analysis of the model’s performance across different traffic sign classes, helping to identify specific areas where the model may be making errors, such as confusing similar-looking signs.

Methodology - Data Preprocessing

1. **Data Collection and Preparation:**
   - Dataset: German Traffic Sign Recognition Benchmark (GTSRB) dataset with over 50,000 images in more than 40 categories.
   - Resizing and Normalization: Standardized image size to 32x32 pixels and normalized pixel values for stable training.
2. **Data Augmentation:**
   - Techniques: Applied rotation, width and height shifts, shear transformations, zoom, horizontal flips, and brightness adjustments to increase data diversity and improve model robustness.

Methodology - Data Preprocessing



Methodology - Model Architecture

CNN Model Design:

- Input Layer: Receives the 32x32 pixel images.
- Convolutional Layers: Extract features from the input images using filters.
- Activation Functions: Use ReLU (Rectified Linear Unit) to introduce non-linearity.
- Pooling Layers: Reduce spatial dimensions while retaining important information.
- Dropout Layers: Randomly drop neurons during training to prevent overfitting.
- Fully Connected Layers: Combine features extracted by convolutional layers for final classification.
- Softmax Output Layer: Outputs probability distribution over the 43 traffic sign classes.

Methodology - Training Process

Loss Function and Optimization:

- Loss Function: Categorical cross-entropy used for multi-class classification to measure the difference between predicted probabilities and actual labels.
- Optimizer: Adam optimizer used for efficient training, combining the advantages of AdaGrad and RMSProp.

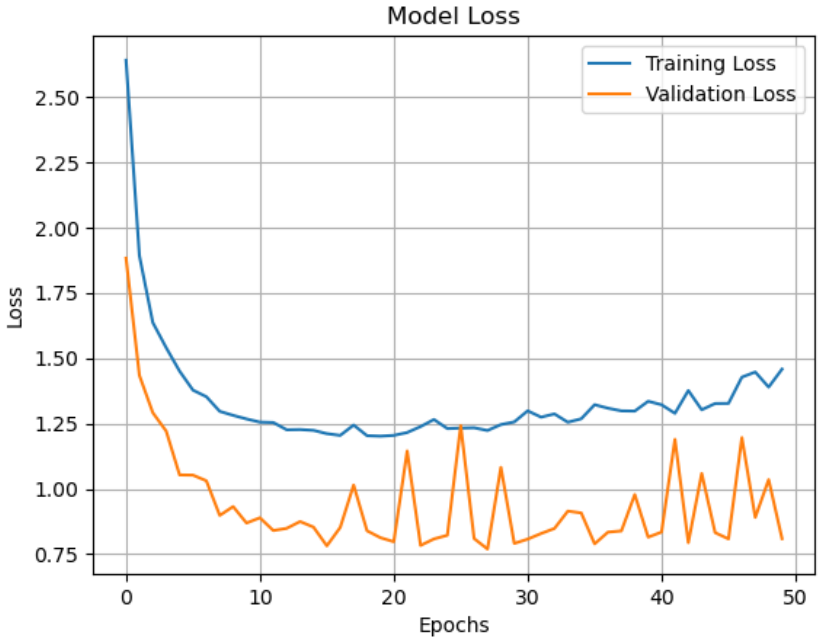
Hyperparameters:

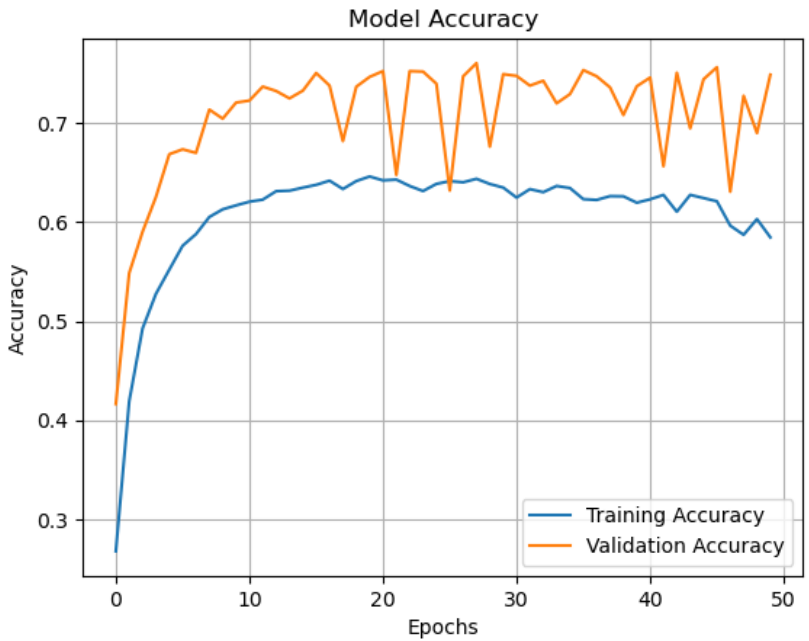
- Batch Size: Number of training examples utilized in one iteration.
- Number of Epochs: Total number of times the model iterates over the entire training dataset.
- Learning Rate: Controls how much to change the model in response to the estimated error each time the model weights are updated.

Model Training:

- Training Process: Trained the model using the prepared dataset, monitored the training and validation accuracy and loss to ensure the model is learning effectively and adjusted hyperparameters as needed.

Methodology - Training Process





Predictions Analysis

Visual Representation:

- Sankey Plot: Illustrates the flow from true labels (left) to predicted labels (right) for each traffic sign class.
- Purpose: Highlights model performance by showing where misclassifications occur.

Observations:

- Correct Predictions: Thicker lines going straight indicate correct predictions where true labels match predicted labels.
- Misclassifications: Diverging lines show incorrect predictions. Color and thickness help identify commonly misclassified classes.
- Class Distribution: Width of bars on the left and right indicates true and predicted class distribution, revealing class imbalances and misclassification trends.

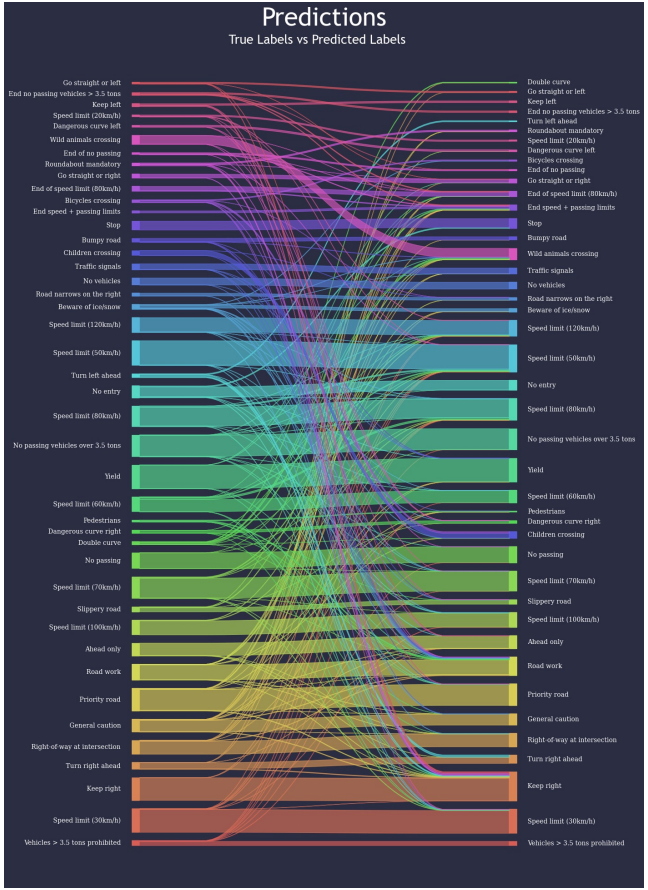
Predictions Analysis

Analysis:

- Common Confusions: Some classes, especially those with similar features, are frequently confused.
- Successful Recognition: Classes like 'Stop' and 'Yield' show clear, straight lines, indicating high accuracy.
- Challenges: Significant misclassification in speed limits and no-entry signs suggests the need for improved feature extraction or data augmentation.

Conclusion and Recommendations:

- Data Augmentation: Enhance training data variety to better differentiate between confusing classes.
- Model Enhancements: Consider adding layers or exploring different architectures for better feature extraction.
- Focused Training: Create targeted training subsets to emphasize classes with high misclassification rates, improving overall model accuracy.


Predictions
True Labels vs Predicted Labels
Go straight or left
End no passing vehicles > 3.5 tons
Keep left
Speed limit (20km/h)
Dangerous curve left
Wild animals crossing
End of no passing
Roundabout mandatory
Go straight or right
End of speed limit (80km/h)
Bicycles crossing
End speed + passing limits
Stop
Bumpy road
Children crossing
Traffic signals
No vehicles
Road narrows on the right
Beware of ice/snow
Speed limit (120km/h)
Speed limit (50km/h)
Turn left ahead
No entry
Speed limit (80km/h)
No passing vehicles over 3.5 tons
Yield
Speed limit (60km/h)
Pedestrians
Dangerous curve right
Double curve
No passing
Speed limit (70km/h)
Slippery road
Speed limit (100km/h)
Ahead only
Road work
Priority road
General caution
Right-of-way at intersection
Turn right ahead
Keep right
Speed limit (30km/h)
Vehicles > 3.5 tons prohibited
Double curve
Go straight or left
Keep left
End no passing vehicles > 3.5 tons
Turn left ahead
Roundabout mandatory
Speed limit (20km/h)
Dangerous curve left
Bicycles crossing
End of no passing
Go straight or right
End of speed limit (80km/h)
End speed + passing limits
Stop
Bumpy road
Wild animals crossing
Traffic signals
No vehicles
Road narrows on the right
Beware of ice/snow
Speed limit (120km/h)
Speed limit (50km/h)
No entry
Speed limit (80km/h)
No passing vehicles over 3.5 tons
Yield
Speed limit (60km/h)
Pedestrians
Dangerous curve right
Children crossing
No passing
Speed limit (70km/h)
Slippery road
Speed limit (100km/h)
Ahead only
Road work
Priority road
General caution
Right-of-way at intersection
Turn right ahead
Keep right
Speed limit (30km/h)
Vehicles > 3.5 tons prohibited

Challenges and Problem Solving

- **Challenges:**
  - Recognizing traffic signs under varying environmental conditions (lighting, weather).
  - Handling distorted or obscured traffic signs.
  - Class imbalance in the dataset.
  - Visually similar classes causing misclassification.
- **Problems Faced by Other Groups:**
  - Poor performance under varied lighting conditions.
  - Difficulty in recognizing partially obscured or distorted signs.
  - Ineffective handling of class imbalance leading to poor accuracy for underrepresented classes.
- **Solutions Implemented:**
  - Data augmentation to simulate diverse conditions and improve robustness.
  - Advanced regularization techniques like dropout and batch normalization to prevent overfitting.
  - Use of a comprehensive and diverse dataset (GTSRB) for training and evaluation.
  - Implementing ensemble techniques and transfer learning to enhance feature extraction and classification accuracy.

Recommendations for Future Work

Generative Models:

Use GANs for class imbalance and data augmentation.

Advanced Model Architectures:

Explore ensemble techniques and transfer learning.

Regularization Techniques and Hyperparameter Tuning:

Improve generalization with advanced techniques.

Model Optimization:

Apply quantization and pruning for smaller, faster models.

Cost-Sensitive Learning:

Implement strategies for underrepresented categories.

Dataset Expansion:

Use diverse images and crowdsource traffic signs from different countries.

Grad-CAM:

Utilize for better interpretability.


Thank you