INTELLIGENT TRAFFIC SIGN CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS FOR AUTONOMOUS AND ASSISTIVE DRIVING

Audrey Wang
Student# 1010173955
audreyxiwei.wang@mail.utoronto.ca

Judie Du
Student# 1009981407
judie.du@mail.utoronto.ca

Yiling Fan
Student# 1010161352
yiling.fan@mail.utoronto.ca

Yinyin(Eva) Zhang
Student# 1010477761
yinyin.zhang@mail.utoronto.ca

Colab Link:

https://colab.research.google.com/drive/1YtmGzW_bqlVXa0NJK2dj7S9FL4_WSwkQ?usp=sharing

ABSTRACT

Traffic signs are vital to maintaining road safety, and their accurate recognition is essential for both human drivers and autonomous systems. This project proposes a deep learning-based traffic sign classification system using Convolutional Neural Networks (CNNs), trained on the German Traffic Sign Recognition Benchmark (GTSRB). Our goal is to develop a high-performance model capable of identifying 43 traffic sign classes under diverse real-world conditions. To address challenges such as class imbalance and image variability, we apply preprocessing techniques including normalization, augmentation, and relabeling. A simple CNN is implemented as the baseline model to benchmark performance. Additionally, this report highlights the division of responsibilities across team members, outlines our project timeline, and discusses potential risks and ethical considerations related to model reliability and data fairness. —-Total Pages: 7

1 INTRODUCTION

Whether in everyday driving or autonomous navigation, traffic signs play a crucial role in ensuring safety and order on the roads. They convey essential information — from speed limits to warnings and regulations — that drivers must interpret quickly and accurately, as any failure can lead to serious consequences. With the increasing adoption of autonomous vehicles and advanced driver assistance systems (ADAS), the ability to recognize and respond to these signs without human input has become more important than ever. This need has motivated us to develop an intelligent traffic sign recognition system capable of reliably identifying and classifying signs from images.

The goal of our project is to build a model that can effectively analyze images and determine the specific type of traffic sign depicted (i.e., stop sign, speed limit indicator, or warning symbol). Such a system could support not only autonomous driving but also assistive technologies for human drivers, such as real-time alerts for overlooked signs.

Deep learning offers a compelling solution to this task due to its proven strengths in image classification. Convolutional Neural Networks (CNNs), in particular, are well-suited for capturing the spatial patterns, shapes, and color features that distinguish different traffic signs. By learning hierarchical representations directly from raw pixel data, CNNs can adapt to variations in lighting, angle, background noise, and more, all of which are common in real-world driving environments. This adaptability and generalization capability make deep learning especially suitable for traffic sign recognition, where consistency and high accuracy are essential for safety.

2 ILLUSTRATION

Figure 1 presents our proposed traffic sign recognition pipeline, visually summarizing the key processing stages from input to classification. The input consists of RGB traffic sign images from the German Traffic Sign Recognition Benchmark (GTSRB)(Stallkamp et al., 2011), which undergo preprocessing (normalization and augmentation) before being fed into our CNN architecture. The visualization highlights key stages: feature extraction through convolutional layers, spatial reduction via max-pooling, and final classification through fully connected layers.

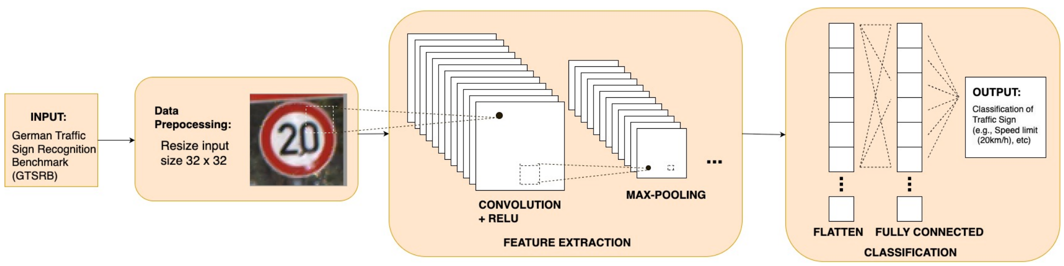

Figure 1: Traffic sign recognition model. Image: GTSRB(Stallkamp et al., 2011)

3 BACKGROUND AND RELATED WORK

The development of traffic sign recognition and classification systems can be traced back several decades, with one of the earliest efforts conducted by Akatsuka & Imai (1987). Their system used basic shape and color filtering techniques to classify Japanese road signs, relying heavily on handcrafted features such as geometric shapes and predefined color thresholds. Given the limited computational resources at the time, the system was relatively simple, but it marked an important step toward automating road sign interpretation and provided an early foundation for the evolution of more advanced classification systems.

Since the introduction of early systems, progress in traffic sign classification has been strongly driven by the availability of high-quality benchmark datasets. In particular, the GTSRB (Stallkamp et al., 2011) has become one of the most widely used datasets in this domain. Released in 2011, the dataset contains over 50,000 labeled images across 43 traffic sign classes, offering a standardized resource for training and evaluating classification models.

Other datasets have also contributed to advancing the field, such as the Swedish Traffic Sign Dataset (STSD) (Larsson & Felsberg, 2011), which captures signs under Northern European lighting and weather conditions, and the Belgium Traffic Sign Dataset (BelgiumTS) (Mathias et al., 2013), which includes both cropped and full-scene images with occlusions and varied perspectives. Together, these datasets have helped researchers build more accurate and reliable classification models by enabling consistent evaluation and encouraging better generalization to real-world environments.

The increasing availability of benchmark datasets also coincided with progress in image classification methods. A major turning point was the introduction of AlexNet (Krizhevsky et al., 2012), a deep convolutional neural network (CNN) that used deeper layers, ReLU activations, and regularization techniques to a new standard for image recognition performance. The success of AlexNet helped establish CNNs as a powerful tool for visual classification tasks. Building on this, Chen et al. (2017) proposed a two-stage architecture with separate CNNs for superclass and subclass classification. The first network identified broad categories of traffic signs, while the second refined the prediction to a specific sign type. Final classification was determined by combining the outputs of both stages. Their system achieved an accuracy of 95.6%. However, its relatively high computational cost limited its suitability for real-time applications, highlighting a trade-off between accuracy and speed in CNN-based methods.

More recent efforts have focused on designing lightweight CNN models that maintain strong classification performance while reducing computational demands. Youssouf (2022) introduced a compact CNN for classifying 43 traffic sign classes from the GTSRB dataset, achieving 99.2% accuracy with only 0.8 million parameters. Similarly, Zaibi et al. (2021) proposed an enhanced version of the classic LeNet-5 architecture, aiming to strike a balance between accuracy and efficiency. Their model achieved 99.84% accuracy on GTSRB (Stallkamp et al., 2011) and 98.37% on the BTSD (Mathias et al., 2013) dataset, while keeping the parameter count low at just 0.38 million. This lightweight design made it well-suited for real-time applications, particularly in embedded systems with limited resources.

By looking at these ongoing developments, we gain valuable insight into how traffic sign classification systems have evolved and where current research continues to explore new possibilities in performance, efficiency, and real-world applicability.

4 Data processing

The dataset used in this project was sourced from the official German Traffic Sign Recognition Benchmark (GTSRB)(Johannes Stallkamp, 2019), a publicly available archive created by Christian Igel at the University of Copenhagen. We extracted the pre-organized training and testing subsets from the source archive and constructed a validation set by allocating 20% of the training data.

To prepare the data for model training, we applied a series of preprocessing transformations. Each image was resized to 32×32 pixels to ensure uniform input dimensions for our CNN. The images were then converted into tensors and normalized across RGB channels to improve training stability. We used PyTorch's ImageFolder utility to organize the dataset based on folder names, which automatically assigned integer labels from 0 to 42 corresponding to each traffic sign class.

To improve interpretability, we created a dictionary mapping each class index to a human-readable traffic sign label. We also visualized the distribution of training samples across all 43 classes (Figure 2) and observed significant class imbalance, with certain signs such as "Speed limit (30km/h)" and "Keep right" appearing far more frequently than others. This imbalance can bias the model toward majority classes and reduce performance on rare categories. To address this, we plan to apply targeted data augmentation during training to increase the effective representation of rare classes. Our approach to label mapping and distribution analysis was inspired by a similar project shared on Kaggle (Sharma, 2021).

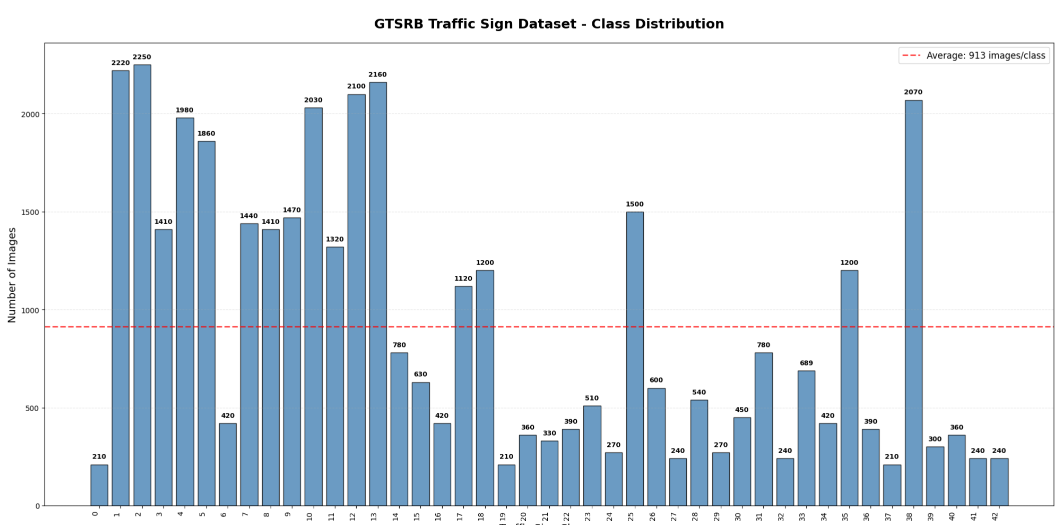

Figure 2: Class distribution of the GTSRB training dataset

5 Architecture

For this project, we will use Convolutional Neural Network(CNN) as our primary model architecture. CNNs are well-suited for image classification tasks because of their ability to automatically learn hierarchical feature representations through localized filter operations, capturing both low-level visual patterns (edges, colours) and high-level semantic features (shapes, symbols) operations(Zeiler & Fergus, 2013).

Our model takes 32×32 RGB images as input. The CNN will consist of multiple convolutional layers that extract features from simple edges to complex shapes, followed by max-pooling layers that reduce image size while preserving important features. The output feature maps are flattened and passed through fully connected layers, with dropout applied to reduce overfitting. The final output layer will use a softmax activation function to classify the input image into one of the 43 predefined traffic sign categories.

The hyperparameters of the model—such as the number of filters, kernel sizes, pooling window dimensions, dropout rate, and learning rate—will be carefully selected and fine-tuned through experimentation and validation.

6 Baseline model

Our classical baseline uses a Histogram-of-Oriented-Gradients (HOG) feature extractor paired with a linear Support Vector Machine (SVM). We begin by resizing each RGB image to 32×32 pixels and normalizing pixel values to $[0, 1]$. HOG descriptors are then computed with 9 orientation bins, 8×8-pixel cells, and 2×2-cell blocks using L2-Hys normalization—following the configuration established by Dalal and Triggs in the Pattern Recognition study (Dalal & Triggs, 2005). For a 32×32 image, this configuration yields a 324-dimensional feature vector ($3{\times}3$ blocks per image, each block contributing 4 cells$\times$9 bins).

These 324-D resulting vectors, which encode local shape information, will be fed into a linear SVM for classification (ℓ_2 regularization, $C = 1.0$, `max_iter`=10,000) using scikit-learn's LinearSVC. Prior research has shown that this HOG+SVM pipeline achieves classification accuracies in the range of 75–91% on GTSRB, and we expect our implementation to match that range (Stallkamp et al., 2012) (Vieira, 2024). Because this pipeline relies on classical CPU-based feature extraction and a convex SVM optimization problem, it does not require GPU acceleration and typically trains in under a few minutes on standard hardware.

This classical baseline is not only interpretable—each descriptor corresponds to an explicit edge-orientation histogram—but also reproducible in just a few lines of Python. While this approach serves as a strong and transparent performance floor, we may replace it in the future with a shallow convolutional neural network (CNN) to better align with course material and enable a more consistent architectural comparison with our primary deep learning model.

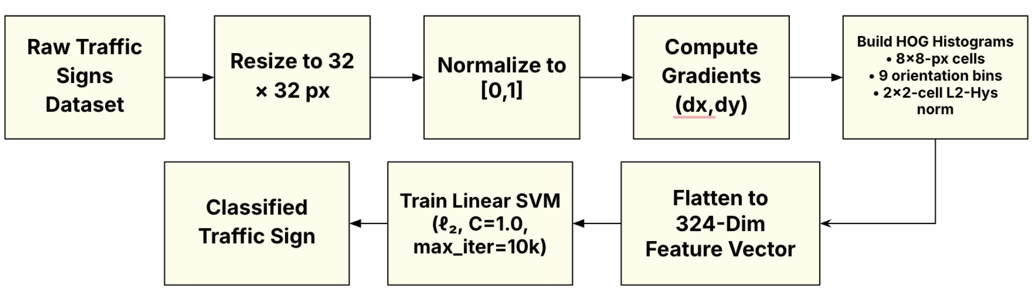

Figure 3: Baseline HOG+SVM Model Architecture

7 Ethical consideration

All data used in this project are sourced from a publicly available research benchmark, free of personally identifiable information. Therefore, our use of the dataset aligns with ethical standards for data privacy and responsible research.

However, deploying traffic sign recognition systems in real-world settings presents ethical challenges. A key concern is maintaining consistent and safe performance in diverse and unpredictable conditions. Our model is trained on the GTSRB dataset, which contains clear and well-annotated images. However, this controlled dataset does not fully capture real-world variability, such as damaged, obscured, or poorly lit signs. As a result, the system's ability to generalize and perform reliably in safety-critical contexts, like autonomous driving, may be limited.

Additionally, the dataset contains significant class imbalance, with certain sign categories occurring far more frequently than others. This imbalance risks skewing the model's predictions toward majority classes and underrepresenting rare but important signs.

8 Project Plan

To keep our traffic sign classification project on track, the team will meet every Tuesday at 11 a.m. EDT via our WeChat/Discord group channel for weekly status updates, code reviews, and task planning. These meetings ensure alignment on deadlines and allow members to raise issues or clarify goals in real time. Outside of our weekly syncs, team members will communicate through WeChat or Discord for quick questions, and version control will be managed via GitHub or Colab. Everyone is expected to respond to messages within 24 hours to maintain steady project momentum.

To avoid conflicts during collaborative coding, we will create separate code sections in our shared Google Colab notebook for each member. This structure allows everyone to contribute simultaneously without overwriting each other's work. Team members are also expected to document their code with meaningful comments and descriptions, especially when making assumptions, importing external libraries, or modifying model architecture. Any major updates or merges will be preceded by group discussion and informal peer review to ensure compatibility.

Before dividing tasks, we will hold a brief planning session to review the evaluation criteria and ensure all members fully understand the deliverables. The project tasks will be divided based on skillset and availability to ensure fair workload distribution. Table 1 outlines the timeline, responsible team members, deadlines, and the type of each milestone (internal or external).

Table 1: Project Plan Timeline for the Traffic Sign Classification Project

Task	**Responsible Team Member**	**Deadline**	**Type**
Brainstorm & Idea Generation and Selection	All members	June 6th	Internal online meeting
Background Research	Judie	June 9th	Internal deadline
Flowchart Generation	Eva	June 9th	Internal deadline
Data Collection & Research	All members	June 9th	Internal deadline
Data Processing in Google Colab/Github	Yiling, Audrey	June 12th	Internal deadline
Baseline Model	Audrey	June 12th	Internal online meeting
Project Proposal	All members	June 13th	External deadline
Primary Model Research	Eva, Judie	July 5th	Internal deadline
Primary Model Implementation	Judie, Audrey	July 9th	Internal deadline
Progress Report	All members	July 11th	External deadline
Test & Evaluation	Eva, Yiling	August 10th	Internal deadline
Final Presentation Slide	All members	August 10th	Internal deadline
Final Presentation Video	All members	August 13th	Internal deadline
Project Presentation & Final Deliverable	All members	August 15th	External project deadline

9 Risk register

In this course's project, proactively identifying and mitigating risks is key to maintaining progress and ensuring reliable results. For our traffic sign classification project, we considered both technical and teamwork-related risks that could impact accuracy, timelines, or coordination. The table below outlines the five most significant risks we anticipate, the conditions under which they might arise, and the concrete strategies we will use to prevent or address them.

To handle unexpected issues such as delays, technical errors, or absences, we will communicate promptly through our shared WeChat group and email. Responsibilities may be redistributed if needed to meet internal deadlines, and all code will be organized in separate sections within the shared Colab notebook to avoid conflicts. Table 2 provides a summary of these risks and our mitigation plan.

Table 2: Risk Register for the Traffic Sign Classification Project

Risk	**Situation Description**	**Likelihood**	**Impact**	**Mitigation Strategy**
Dataset Size and Memory Limitations	GTSRB dataset is large, and loading all images (43 classes) may exceed memory limits in Colab.	High	Medium	Use a subset of 5–10 classes; resize images to 32×32 or 28×28; clear unused variables; batch loading.
Model Underperformance (Baseline or CNN)	HOG+SVM may perform poorly on more complex signs; CNN may overfit due to limited data.	Medium	High	Justify HOG+SVM as a classical baseline; tune CNN with dropout and validation monitoring; compare relative performance.
Class Imbalance and Labeling Errors	GTSRB has uneven class distribution and potential errors in relabeling images.	Medium	Medium	Use stratified sampling; visually check sample labels; apply class weights if needed.
Time and Compute Constraints	Limited Colab session time and no GPU guarantees may slow training.	Medium	Medium	Reduce epochs and batch size; test on small datasets first; save checkpoints regularly.
Team Coordination Issues	Team members may overwrite each other's code in Colab, delay pushing updates, or miss internal deadlines due to unclear task ownership or time zone differences.	Low	Medium	Assign separate sections in the Colab notebook to each member; set internal checkpoints; use a shared Google Doc or Instagram group for daily updates; hold short sync meetings twice a week.

References

Hidehiko Akatsuka and Shinichiro Imai. Road signposts recognition system. *SAE Technical Paper*, 870239:1–12, 1987. URL `https://doi.org/10.4271/870239`.

Lingying Chen, Guanghui Zhao, Junwei Zhou, and Li Kuang. Real-time traffic sign classification using combined convolutional neural networks. In *Proceedings of the 2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 399–404, 2017. URL `https://ieeexplore.ieee.org/document/8575857`.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893, San Diego, CA, USA, 2005.

Jan Salmen Shivank Patel Johannes Stallkamp, Marc Schlipsing. German traffic sign recognition benchmark gtsrb, 05 2019. URL `https://sid.erda.dk/public/archives/daaeac0d7ce1152aea9b61d9f1e19370/published-archive.html`.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012. doi: 10.1145/3065386.

Fredrik Larsson and Michael Felsberg. Using fourier descriptors and spatial models for traffic sign recognition. In *Image Analysis*, pp. 238–249. Springer Berlin Heidelberg, 2011. URL `https://link.springer.com/chapter/10.1007/978-3-642-21227-7_23`.

Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition — how far are we from the solution? In *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2013. URL `https://ieeexplore.ieee.org/document/6707049`.

Shivank Sharma. Gtsrb cnn - 98% test accuracy, 2021. URL `https://www.kaggle.com/code/shivank856/gtsrb-cnn-98-test-accuracy`.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IEEE International Joint Conference on Neural Networks*, pp. 1453–1460, 2011. URL `https://ieeexplore.ieee.org/document/6033395?arnumber=6033395`.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012.

Luis Vieira. Comparing performance of preprocessing techniques for traffic sign recognition using a hog-svm. *arXiv*, abs/2504.09424, Apr 2024. doi: 10.48550/arXiv.2504.09424. URL `https://arxiv.org/abs/2504.09424`. preprint.

Njayou Youssouf. Traffic sign classification using cnn and detection using faster-rcnn and yolov4. *Heliyon*, 8(12), 2022. doi: https://doi.org/10.1016/j.heliyon.2022.e11792. URL `https://www.sciencedirect.com/science/article/pii/S2405844022030808`.

Ameur Zaibi, Anis Ladgham, and Anis Sakly. A lightweight model for traffic sign classification based on enhanced lenet-5 network. *Journal of Sensors*, 2021:1–12, 2021. doi: 10.1155/2021/8870529. URL `https://doi.org/10.1155/2021/8870529`.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks, 2013. URL `https://arxiv.org/abs/1311.2901`.

[1]

[1] The reference list is organized using iclr2022_conference format.