FINAL REPORT: 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

This project develops an intelligent traffic sign classification system to support autonomous and assistive driving. Using the German Traffic Sign Recognition Benchmark (GTSRB), we implemented and evaluated three approaches: a classical Histogram of Oriented Gradients (HOG) with a linear Support Vector Machine (SVM) baseline, a custom Convolutional Neural Network (CNN) trained from scratch, and a transfer-learning model based on ImageNet-pretrained AlexNet. Data processing included resizing, a stratified 70/15/15 train–validation–test split, and on-the-fly augmentation to mitigate class imbalance. Evaluations showed clear gains from deep learning over the baseline: the HOG+SVM reached 89.1% test accuracy, the AlexNet-features model 98.44%, and our scratch CNN achieved the best performance at 98.85% on GTSRB. We also evaluated generalization on previously unseen traffic-sign images, where the CNN achieved 85.34% accuracy. This report details architecture choices, training setup, quantitative and qualitative analyses, and discusses trade-offs in accuracy, efficiency, and complexity, along with limitations related to dataset bias and deployment conditions.
—-Total Pages: 9

1 INTRODUCTION

Whether in everyday driving or autonomous navigation, traffic signs play a crucial role in ensuring safety and order on the roads. With the rise of autonomous vehicles, recognizing and responding to these signs without human intervention is increasingly important. This motivated us to develop an intelligent traffic sign classification system capable of identifying signs from images. The goal is to build a model that analyzes an image and determines the specific traffic-sign category (e.g., stop, speed limit, warning). Such a system supports both autonomous driving and assistive technologies for human drivers, including real-time alerts for overlooked signs. Deep learning is well-suited for this task, with Convolutional Neural Networks (CNNs) able to capture spatial patterns, shapes, and colors distinguishing signs. By learning hierarchical features from raw pixels, CNNs adapt to lighting, angle, and background changes, ensuring accuracy and consistency.

2 Illustration

Our traffic sign classification model (Fig. 1) is a custom CNN designed to balance accuracy and efficiency. It takes RGB images from the German Traffic Sign Recognition Benchmark (GTRSB) (Stallkamp et al., 2011), resized to 32×32 pixels, and processes them through convolutional layers to extract spatial and semantic features. We used batch normalization to stabilize training, dropout to mitigate overfitting, and data augmentation to address class imbalance. The final feature maps are passed through fully connected layers to classify each image into one of 43 traffic sign categories.

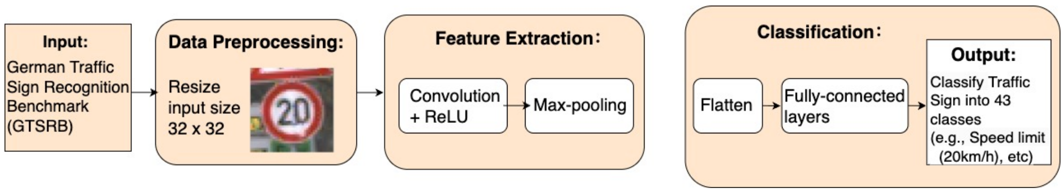


Figure 1: Traffic sign recognition model. Image: GTSRB(Stallkamp et al., 2011)

3 Background & Related Work

Traffic sign classification and recognition have been studied for decades, with one of the earliest efforts by Akatsuka & Imai (1987). Their system used basic shape and color filtering to classify Japanese road signs, relying on handcrafted features such as geometric contours. Although simple, it marked an important step toward automated road sign interpretation and laid the groundwork for future systems. The development of benchmark datasets has driven major progress. The GTSRB (Stallkamp et al., 2011) remains one of the most widely used, containing over 50,000 labeled images across 43 classes. Other datasets, such as the Swedish Traffic Sign Dataset (STSD) (Larsson & Felsberg, 2011) and BelgiumTS (Mathias et al., 2013), introduced variations in lighting, occlusion, and viewpoint, improving model generalization in real-world conditions. A breakthrough in visual recognition came with AlexNet (Krizhevsky et al., 2012), which demonstrated the power of deep CNNs for image classification. Building on this, Chen et al. (2017) proposed a two-stage CNN that classified signs through superclass and subclass hierarchies, achieving 95.6Recent work has focused on lightweight models. Youssouf (2022) introduced a compact CNN achieving 99.2% accuracy on GTSRB with 0.8M parameters. Similarly, Zaibi et al. (2021) presented an efficient LeNet-5 variant, reaching 99.84% on GTSRB (Stallkamp et al., 2011) and 98.37% on BelgiumTS (Mathias et al., 2013) with only 0.38M parameters. These studies illustrate the evolution of traffic sign classification and ongoing efforts to balance accuracy, efficiency, and real-world applicability.

4 Data Processing

4.1 Image Resizing & Dataset Split

The dataset for this project comes from the official German Traffic Sign Recognition Benchmark (Johannes Stallkamp, 2019), created by Christian Igel at the University of Copenhagen. We used the pre-organized training subset, containing 43 class folders with about 1000–2000 images each (over 50,000 total). Because the signs contain only simple shapes and short legends, resizing to 32×32 pixels preserves the necessary detail while enabling efficient CNN training. Using a fixed random seed (25) for reproducibility, we shuffled each class and split into 70% train / 15% validation / 15% test. This split provides sufficient data (70%) for the model to learn meaningful features, while reserving a dedicated validation set (15%) for hyperparameter tuning and a separate test set (15%) for unbiased final evaluation. Stratifying each class into these proportions ensures balanced representation, maintains reproducibility, and keeps the training, validation, and testing pipelines fully isolated. The images are then organized into three directories: `train/ val/ test/`, each containing 43 subfolders (one per class) for straightforward loading via `torchvision.datasets.ImageFolder`.

4.2 ADDRESSING CLASS IMBALANCE VIA AUGMENTATION

Initial analysis revealed a wide class distribution (some classes ∼2000 samples, others ∼200) as seen in Figure 2. Figure 3 below illustrates the data processing pipeline.

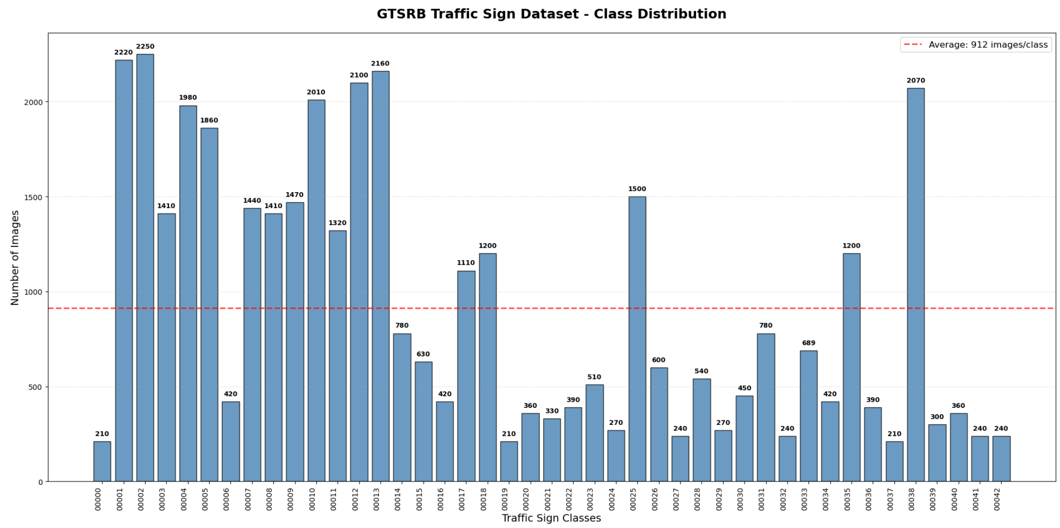


Figure 2: Class distribution of the GTSRB training dataset

To achieve a near-uniform class distribution (target ≈1,575 images/class/epoch), we oversample only the `train/` split via augmentation. Each image is resized to 32×32, then (with default probabilities) undergoes: random horizontal flip (p=0.5), rotation in $[-15°, +15°]$, and affine translation up to 10% of width/height in x and y. Flips simulate mirror invariance, rotations handle tilted angles, translations mimic framing shifts, and color jitter (brightness/contrast ± 0.2; factors $[0.8, 1.2]$) compensates for lighting changes. Images are then converted to tensors and normalized channel-wise using $[\mu{=}(0.485, 0.456, 0.406),\ \sigma{=}(0.229, 0.224, 0.225)]$ to stabilize optimization. No augmentation is applied to `val/` or `test/`; they receive only resize, tensor conversion, and the same normalization.

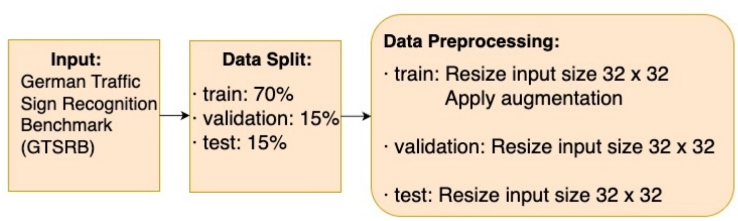


Figure 3: Data Processing Flowchart

4.3 SAMPLE VISUALIZATION & PROCESSING CHALLENGES

To verify dataset integrity, we plotted one random sample per class in a 7×7 grid using a snippet that selects the first index for each label from `train_data.samples`. Fig. 4 confirmed representation of all 43 classes and that augmentations preserved key visual features.

We encountered several challenges during data processing. Copying tens of thousands of small image files into split directories created a significant disk I/O bottleneck, which we reduced by batching filesystem operations and skipping redundant writes. In addition, tuning augmentation parameters required caution: rotation and translation ranges had to be limited to avoid cropping out critical sign elements, balancing diversity with legibility.

Figure 4: One Sample Image From Each Class

On-the-fly augmentation further increased GPU memory and loading demands, so we enabled `pin_memory = True` in our DataLoaders and used a moderate batch size of 68 to maintain smooth throughput. Finally, Colab's strict GPU limits and lack of persistent storage prompted migration to local runtime environments, with each team member storing the full 50,000+ image dataset locally and updating code paths for experiments.

To assess generalization on truly new data, we could collect a separate set of previously unseen traffic-sign images, either on our own or from GTSRB's official website. These images would include variations in design, lighting, and background, providing a realistic measure of performance under conditions different from the training data.

5 Architecture

We implemented two complementary CNN-based classifiers for GTSRB. First, our custom CNN (GTSRBCNN) ingests 32×32×3 RGB inputs and applies three sequential Conv–ReLU–MaxPool blocks: each block uses a 3×3 convolution (padding=1, stride=1) to expand channels (3→32→64→128), a ReLU activation, and a 2×2 max-pool (stride=2). After the third block, the feature maps have dimensions 128×4×4 (2048 features), which are flattened and passed through a 2048→256 fully-connected layer with ReLU and 50% dropout, then a final 256→43 linear layer produces the class logits. This network (about 630K parameters) is trained for 20 epochs with Adam (lr = 1e-3), batch size = 68, and data augmentations (random flips, rotations, affine shifts, color jitter) to prevent overfitting.

Our second model applied transfer learning on ImageNet-pretrained AlexNet. We resize inputs to 256×256, center-crop to 224×224, and normalize to AlexNet's statistics. Convolutional features are extracted via alexnet.features(), adaptive-avg-pooled and flattened into a 9216-dim vector. A lightweight three-layer MLP head (ANFClassifier) then maps 9216→512→256→43, with ReLU and 50% dropout after the first two layers. Only this head (about 4.9M parameters) is trained (AlexNet's conv layers remain frozen) for 20 epochs under the same optimizer settings, achieving strong accuracy with minimal tuning.

These fully reproducible designs together show the trade-off between a higher-capacity, feature-reusing transfer-learning head and a parameter-efficient, scratch-trained CNN.

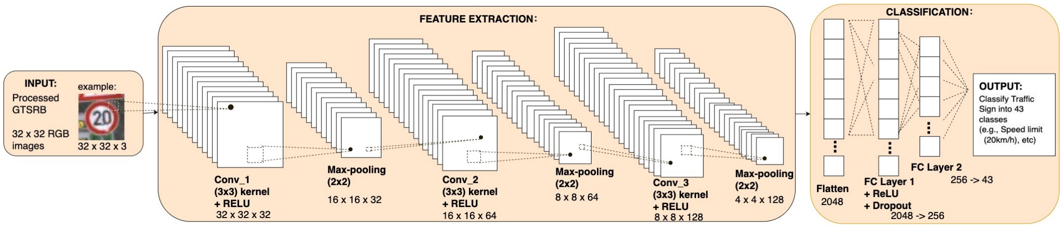


Figure 5: Primary Model

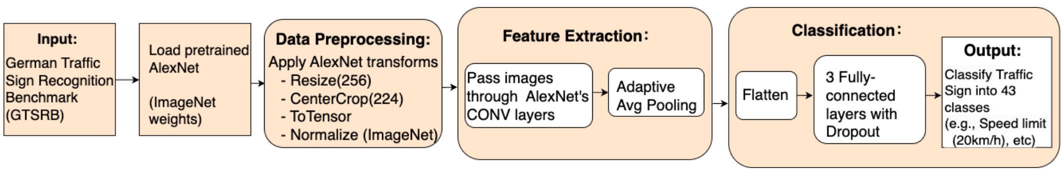


Figure 6: Transfer Learning Model

6 Baseline Model

Our classical baseline uses a Histogram-of-Oriented-Gradients (HOG) feature extractor paired with a linear Support Vector Machine (SVM). Each input image is first denormalized and converted from tensor to NumPy format, restoring the RGB pixel values to the [0, 255] range. HOG features are extracted independently for each of the three RGB channels, then concatenated to form a single flattened feature vector. These descriptors are passed into a linear SVM classifier trained using scikit-learn's LinearSVC, requiring minimal tuning and no GPU acceleration.

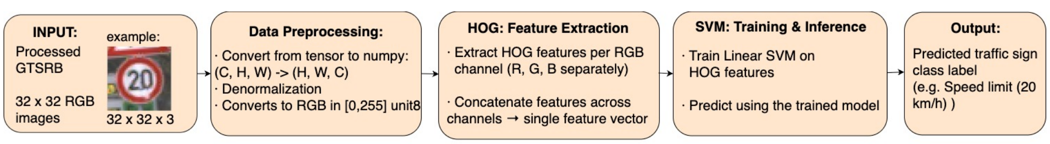
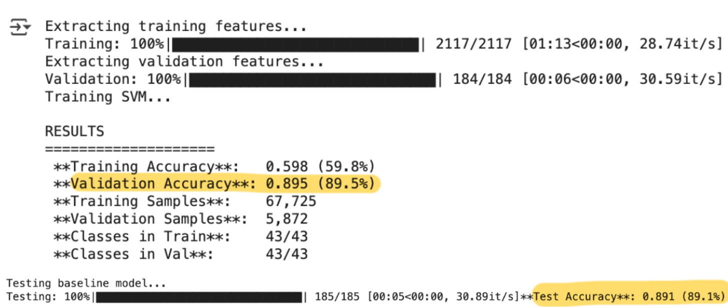


Figure 7: Baseline Model

Quantitatively, the HOG+SVM model achieves an overall classification accuracy of 89.1% on the GTSRB test set (Fig. 8). The macro-averaged F1-score is 0.87, with particularly strong performance on clearly distinguishable signs as shown in the Classification Report (Fig. 9). On the validation set, the model reaches a similar accuracy of 89.5%, suggesting good generalization. However, the model underperforms on less frequent or visually ambiguous classes, such as Class0 (Speed limit (20km/h)) and Class5 (Speed limit (80km/h)), as seen in the confusion matrix (Fig. 10).


```
Extracting training features...
Training: 100%|██████████| 2117/2117 [01:13<00:00, 28.74it/s]
Extracting validation features...
Validation: 100%|██████████| 184/184 [00:06<00:00, 30.59it/s]
Training SVM...

RESULTS
====================
 **Training Accuracy**:   0.598 (59.8%)
 **Validation Accuracy**: 0.895 (89.5%)
 **Training Samples**:    67,725
 **Validation Samples**:  5,872
 **Classes in Train**:    43/43
 **Classes in Val**:      43/43

Testing baseline model...
Testing: 100%|██████████| 185/185 [00:05<00:00, 30.89it/s]**Test Accuracy**: 0.891 (89.1%)
```

Figure 8: Baseline Model Training & Validation Report

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.38	0.53	32
1	0.77	0.79	0.78	333
2	0.81	0.84	0.83	338
3	0.85	0.83	0.84	213
4	0.94	0.97	0.95	297
5	0.78	0.71	0.74	279
6	0.97	0.97	0.97	63
7	0.81	0.80	0.81	217
8	0.81	0.79	0.80	213
9	0.92	0.88	0.90	221
10	0.96	0.98	0.97	302
11	0.82	0.92	0.87	199
12	0.99	0.99	0.99	315
13	1.00	1.00	1.00	324
14	1.00	0.97	0.98	117
15	0.81	0.97	0.88	95
16	0.90	0.95	0.92	63
17	0.97	1.00	0.98	167
18	0.88	0.97	0.92	180
19	0.88	0.66	0.75	32
20	0.77	0.75	0.76	55
21	0.90	0.69	0.78	51
22	0.85	0.78	0.81	59
23	0.91	0.87	0.89	77
24	0.81	0.71	0.75	41
25	0.85	0.93	0.89	225
26	0.86	0.84	0.85	90
27	0.89	0.69	0.78	36
28	0.90	0.69	0.78	81
29	0.91	0.76	0.83	41
30	0.82	0.75	0.78	68
31	0.86	0.98	0.92	117
32	0.95	0.97	0.96	36
33	0.94	0.97	0.95	104
34	0.95	0.89	0.92	63
35	0.97	0.99	0.98	180
36	0.93	0.95	0.94	59
37	0.96	0.84	0.90	32
38	0.97	0.97	0.97	311
39	0.92	0.98	0.95	45
40	0.94	0.82	0.87	55
41	0.89	0.92	0.90	36
42	0.94	0.92	0.93	36
accuracy			0.89	5898
macro avg	0.89	0.86	0.87	5898
weighted avg	0.89	0.89	0.89	5898

Figure 9: Baseline: Per-Class Precision, Recall, and F1 Scores

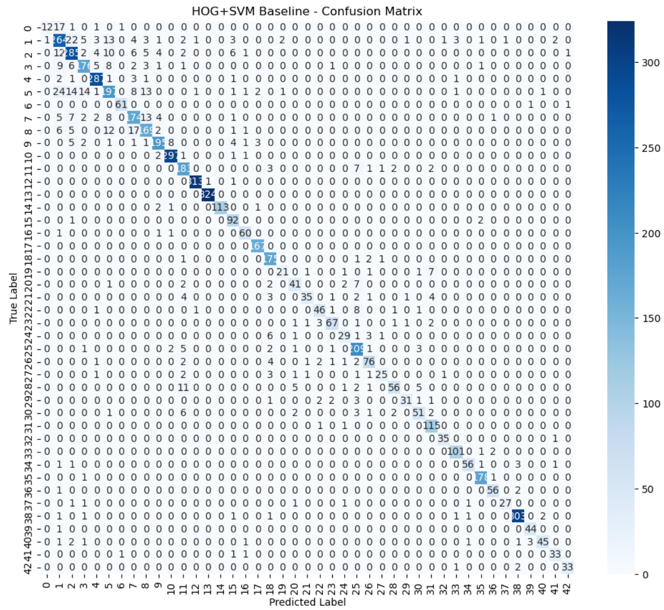


Figure 10: Baseline: Confusion Matrix

Visual inspection of the confusion matrix highlights consistent misclassification between semantically similar signs, particularly among speed limit signs (e.g., 30 km/h vs. 50 km/h) and warning triangles with similar shapes. These signs share color schemes and general structure, but differ in small text or icons, which HOG struggles to differentiate. This suggests a limitation in HOG's reliance on local gradient patterns and lack of contextual understanding.

Despite its simplicity, the baseline serves as a useful benchmark to measure improvements from learned representations. Its primary strength is classifying images with strong edge definitions and clear geometric features. However, the model lacks robustness to variations in lighting, occlusion, and intra-class similarity. These challenges motivate the use of deep convolutional architectures that are better suited for learning hierarchical, task-specific features directly from data.

7 RESULTS

7.1 QUANTITATIVE RESULTS

We evaluate with three complementary measures: (i) overall test accuracy (how often the model is correct), (ii) class–wise precision/recall/F1 from scikit–learn's `classification_report` (how balanced performance is across all 43 classes), and (iii) training time (practicality of training and iteration). Together, these metrics capture both correctness and efficiency.

On the 15% held–out GTSRB test split, the baseline reaches **89.1%** accuracy with **weighted precision/recall/F1 = 0.89** (Fig. 9). The confusion matrix in Figure 10 shows most errors come from visually similar speed–limit signs (e.g., 80 vs. 100 km/h) and occasional swaps between triangular warnings and small circular limits. Training the SVM on HOG descriptors takes $<$35 minutes on CPU, and inference is essentially instant, but the sub–90% accuracy leaves room to improve.

Our 7-layer primary CNN trained achieves **98.85%** test accuracy on the 5898 images GTSRB test set (Fig. 12). The `classification_report` yields **weighted precision, recall, and F1** all equal to 0.99 (macro averages are also 0.99). In Figure 11 we can see that most classes are near-perfect, the remaining errors concentrate in a few categories: *Speed limit (20 km/h)* has recall=0.94, *Speed limit (80 km/h)* recall=0.96, *Speed limit (100 km/h)* precision=0.93 with recall=0.99, *Traffic signals* recall=0.93, and *Bicycles crossing* recall=0.95. A normalized confusion matrix confirms these mistakes mostly occur between visually similar or low-contrast categories (especially adjacent speed-limit digits and small symbol signs). Training for 20 epochs on our laptop GPU takes approximately 50 minutes. Relative to the HOG+SVM baseline (89.1% accuracy), this reduces the error rate from 10.9% to 1.15%—$\approx 9.5\times$ fewer errors (about an 89% relative error reduction).

Primary CNN Classification Report:

	precision	recall	f1-score	support
Speed limit (20km/h)	1.00	0.94	0.97	32
Speed limit (30km/h)	0.98	0.98	0.98	333
Speed limit (50km/h)	0.99	0.99	0.99	338
Speed limit (60km/h)	1.00	0.98	0.99	213
Speed limit (70km/h)	0.99	1.00	0.99	297
Speed limit (80km/h)	0.98	0.96	0.97	279
End of speed limit (80km/h)	0.98	1.00	0.99	63
Speed limit (100km/h)	0.93	0.99	0.96	217
Speed limit (120km/h)	0.98	0.96	0.97	213
No passing	1.00	1.00	1.00	221
No passing veh over 3.5 tons	1.00	1.00	1.00	302
Right-of-way at intersection	0.99	1.00	1.00	199
Priority road	0.99	1.00	1.00	315
Yield	1.00	0.99	1.00	324
Stop	1.00	0.99	1.00	117
No vehicles	0.99	0.99	0.99	95
Veh > 3.5 tons prohibited	1.00	0.98	0.99	63
No entry	0.99	1.00	1.00	167
General caution	0.97	1.00	0.99	180
Dangerous curve left	1.00	1.00	1.00	32
Dangerous curve right	1.00	0.95	0.97	55
Double curve	1.00	1.00	1.00	51
Bumpy road	0.95	1.00	0.98	59
Slippery road	0.99	0.99	0.99	77
Road narrows on the right	1.00	1.00	1.00	41
Road work	0.97	1.00	0.99	225
Traffic signals	0.99	0.93	0.96	90
Pedestrians	1.00	0.97	0.99	36
Children crossing	1.00	1.00	1.00	81
Bicycles crossing	1.00	0.95	0.97	41
Beware of ice/snow	0.97	0.99	0.98	68
Wild animals crossing	1.00	0.98	0.99	117
End speed + passing limits	1.00	1.00	1.00	36
Turn right ahead	1.00	1.00	1.00	104
Turn left ahead	1.00	1.00	1.00	63
Ahead only	1.00	0.99	1.00	180
Go straight or right	0.97	0.98	0.97	59
Go straight or left	1.00	0.97	0.98	32
Keep right	1.00	0.99	1.00	311
Keep left	0.98	1.00	0.99	45
Roundabout mandatory	1.00	1.00	1.00	55
End of no passing	1.00	1.00	1.00	36
End no passing veh > 3.5 tons	1.00	0.97	0.99	36
accuracy			0.99	5898
macro avg	0.99	0.99	0.99	5898
weighted avg	0.99	0.99	0.99	5898

Figure 11: Primary CNN: Per-Class Precision, Recall, and F1 Scores

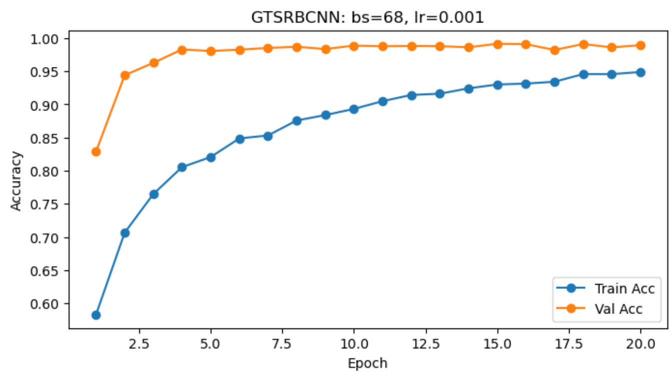


Figure 12: Primary Model's Training Curves

We also tested an AlexNet–features + MLP head and observed **98.44%** accuracy in our setup, but it was slower end–to–end due to 224×224 inputs and a larger head. Since it was both slightly less accurate and slower, we selected the scratch CNN as our final model (Fig. 13).

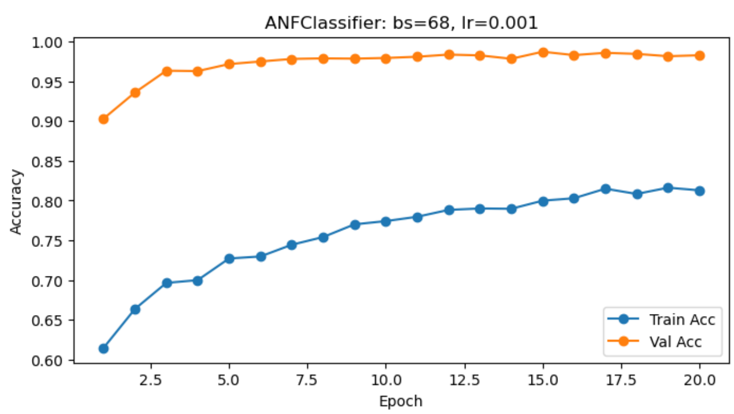


Figure 13: Training Curve with Transfer Learning

7.2 Qualitative Results

To complement the quantitative metrics, we randomly sampled four *distinct* classes from the held–out GTSRB test set and visualized one crop per class. For each image we display the **ground–truth label** and the model's **predicted label** with its **softmax confidence**. The examples (Fig. 14) include *Road narrows on the right*, *Speed limit (20 km/h)*, *Yield*, and *Dangerous curve left*. Our primary CNN predicted all four correctly with very high confidence (approximately 100%), indicating strong performance on prototypical, high–contrast instances—even under mild blur or low light visible in the warning–triangle frames.

Figure 14: Examples from four distinct classes in the test set. Each panel shows the ground truth (top line) and the model's prediction with confidence (second line)

These qualitative observations align with the class–wise analysis: while most categories are near–perfect, remaining errors concentrate in visually similar classes (especially adjacent speed–limit digits) and small, low–contrast symbol signs (e.g., *Traffic signals*, *Bicycles crossing*). In such cases, predicted probabilities are typically lower and occasional confusions occur, matching the reduced recall for these classes. The mosaic and confusion matrix together highlight the model's strengths (clear prototypical signs) and its limitations (occlusion/blur and look–alike classes).

Although results on GTSRB are strong, several limitations remain. (i) GTSRB contains pre–cropped, centered signs. Our model was not evaluated in a full detection–then–classification pipeline, so domain shift (background clutter, scale variation) may reduce real–world accuracy. (ii) Errors concentrate on visually similar or low–contrast categories (e.g., adjacent speed–limit digits, small symbol signs) and under blur, occlusion, or low illumination. (iii) Predicted confidences are often near 100%, suggesting potential over–confidence, we did not assess calibration (e.g., ECE/Brier score). (iv) Class support is imbalanced, so estimates for rare classes are less stable. (v) All experiments used 32×32 crops, which may eliminate detail needed for fine–grained distinctions.

8 Evaluate model on new data

To assess generalization, we tested the primary model on a separate set of traffic sign images not included in the original GTSRB training or test splits. This dataset, also from the official German Traffic Sign Recognition Benchmark (Johannes Stallkamp, 2019), contains signs captured under varied conditions such as different lighting and viewing angles. After preprocessing to match the model's input format, the evaluation yielded a test accuracy of 85.34%, indicating reasonable performance on previously unseen data. For a qualitative check, we randomly selected 10 samples and compared predictions with ground truth labels (Figure 15). The model correctly predicted 8 out of 10 test samples. While simple signs were classified reliably, errors in similar speed-limit categories (e.g., 50,km/h vs. 80,km/h) suggest areas for improvement in handling fine textual differences.

Figure 15: 10 randomly selected samples from the new data. Each panel shows the ground truth (top line) and the model's prediction with confidence (second line)

9 Discussion

The model performs best on clear, high-contrast, and centered signs, where it often produces confidence scores close to 100%. This suggests the architecture and training procedure are well-suited for clean and standardized inputs. In these cases, the lack of background clutter, occlusion, and scale variation makes classification easier. However, this controlled testing setup may not reflect more challenging real-world environments such as dashcam footage or roadside cameras.

We also found that the model's predicted confidence scores are often higher than its actual accuracy. For example, while the test accuracy is 98.85%, many predictions still have confidence near 100%. This suggests mild overconfidence, and it can be a problem in safety-critical applications like traffic sign recognition because it can hide uncertainty. In future work, we could measure calibration more precisely using metrics like Expected Calibration Error (ECE) or Brier Score, and apply techniques such as temperature scaling or confidence penalization to better align predicted probabilities with actual correctness.

10 Project Difficulty

Our project presented several challenges beyond course labs. Unlike binary or low-class-count tasks, our model had to distinguish between 43 traffic sign classes, many of which have subtle visual differences. The dataset also exhibited class imbalance and real-world noise, requiring thoughtful data augmentation and tuning to ensure accuracy.

Both a classical HOG+SVM baseline and a deep learning model are implemented to provide comparative analysis. We also experimented with transfer learning by adapting and fine-tuning AlexNet's architecture for our 43-class classification task. Our evaluation included multiple metrics such as macro-averaged F1-scores, per-class precision and recall, and confusion matrix analysis to gain deeper insight into model performance.

These efforts reflect our initiative to engage with course concepts at a deeper technical level and to consider how such models could be applied in realistic scenarios. Overall, our team applied and extended core deep learning techniques in a thoughtful and practical way.

11 Ethical Considerations

The primary ethical concern is the risk of misclassification, particularly for signs that are visually similar. In real-world scenarios, incorrect predictions (e.g., misidentifying a stop sign) could pose safety risks, especially if users over-rely on the system without proper supervision or intervention. Another concern lies in the training data itself. The GTSRB (Stallkamp et al., 2011) dataset contains only German traffic signs captured in relatively clear, sunny conditions. As a result, the model may struggle to generalize to poor weather scenarios such as rain, fog, or snow. Since it has not been trained on international signage, its deployment should be restricted to regions with similar traffic sign standards unless the model is retrained accordingly.

12 Conclusion

Overall, our primary CNN achieved strong performance on GTSRB and maintained reasonable accuracy on previously unseen images. Future work could focus on expanding the dataset to include traffic signs from different countries and regions, enabling the model to recognize a wider range of designs beyond the German standard. Such improvements would make the system more applicable in diverse contexts, including use in Toronto to assist drivers, particularly new international students or recent immigrants who may be unfamiliar with local traffic signs. Additional targeted augmentation or training samples could also help address class imbalance and reduce errors in visually similar categories, such as adjacent speed-limit signs or small symbol signs. Focusing on these areas would help make our system more dependable in realistic autonomous and assistive driving scenarios.

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`.