Traffic Lights and Sign Detection using ResNeXt-32

Jacob Brown **Kevin Shin**
Department of Electrical and Computer Engineering
University of California, San Diego
A16374531 A16666553
{jdb004,d3shin}@ucsd.edu

Abstract

Modern vehicles are equipped with Advanced Driver Assistance Systems (ADAS), enhancing safety and navigation through features such as traffic sign and signal recognition. However, older vehicles lack these advancements, posing increased risks on the road. To bridge this gap, we have developed a neural network based on the ResNeXt architecture, designed to detect traffic signs and signals in real-time using dash cam technology. This system aims to retrofit older vehicle models with safety features akin to those found in contemporary cars. To accommodate the computational and sensory constraints typical of older vehicles, our approach focuses on processing gray scale images so that it focuses on spatial relations. Furthermore, we leverage the grouped convolutional parallelization of ResNeXt to increase computational efficiency. In this paper, we will go through our process and highlight our findings

1 Introduction

ADAS plays an important role in reducing traffic accidents and improving survival rates. While modern vehicles are often equipped with next generation safety features, many older cars lack them. Integrating a neural network capable of detecting traffic lights and signs can significantly lower the risk of pedestrian incidents caused by drivers overlooking traffic lights and signs. Given that the majority of vehicles are not equipped with high-end sensors like LIDAR, we turn to dash cams as a viable alternative, with the possibility of even using smartphones for this purpose. Our goal is to develop a neural network that demands minimal computational resources, making ResNeXt an ideal choice for efficiently enhancing vehicle safety without the need for sophisticated hardware.

The model should thus satisfy the two below:

- **Inference Speed:** A minimum inference speed with frame processing at 100 ms with a goal of 50 ms.
- **Accuracy:** Higher than 60% accuracy in identifying and labeling traffic lights and signs, ideally reaching 80% or above.

2 Related Work

The evolution of traffic sign and light detection systems has been influenced by advancements in deep learning architectures. Traditional approaches have been replaced by neural networks, particularly by Convolutional Neural Networks (CNNs), due to their superior performance in image-based tasks [1].

One study utilized a modified version of the You Only Look Once (YOLO) architecture, specifically the Cross Stage Partial YOLOv4 (YOLOv4-CSP), for traffic light and sign detection. This system, optimized for real-time processing, demonstrated a mean Average Precision (mAP) of 79.77% at an

Intersection over Union (IoU) threshold of 0.5, achieving a frame rate of 29 FPS on an NVIDIA Tesla T4 GPU [1]. While YOLOv4-CSP exhibits high efficiency in real-time object detection, it primarily focused on speed, potentially compromising on the sensitivity to smaller or less prominent objects.

In contrast, our work integrates the ResNeXt architecture, known for its approach to increasing network capacity through cardinality expansion, which involves adding more transformation paths within each block. ResNeXt stands out for its method of arranging convolutions to enhance model learning capacity while also maintaining or even reducing computational efficiency relative to other CNN-architectures.

Compared to the modified YOLOv4-CSP approach, ResNeXt offers a structured way to increase model complexity without a substantial rise in resource consumption, making it ideal for usage in environments with computational constraints, such as older vehicle models. The aggregated residual transformations of ResNeXt allows it to capture nuanced features of traffic signs and lights, which can be crucial for accurate detection under varied and challenging conditions [3].

3 Dataset

For our ResNeXt-32 implementation, we use the LISA Traffic Sign Dataset, Tiny LISA Traffic Sign Dataset, and the LISA Traffic Light Dataset. As one of our focuses is efficiency under limited computational resources, we compare the LISA dataset with the Tiny LISA dataset in order to analyze the effects on the amount of data. We try to see if the performance of the Tiny Lisa dataset can be analogous to the performance of the LISA Dataset, which has 7 times as more data. We also pair both of these traffic sign datasets with the LISA Traffic Light Dataset to examine how one unified model performs.

3.1 LISA Traffic Sign Detection Dataset

The LISA dataset contains traffic sign annotations from U.S. roads, featuring 47 distinct sign types across 7,855 annotations in 6,610 frames. Sign dimensions vary from 6x6 to 167x168 pixels. The dataset includes images captured with different cameras, resulting in varied image resolutions from 640x480 to 1024x522 pixels, with some images in color and others in grayscale. Additionally, the full dataset offers videos for all annotated signs. Annotations detail the sign type, position, size, occlusion status, and whether the sign is on a side road. All annotations are stored in plain text .csv files. The dataset also comes with Python tools for managing the annotations and extracting specific signs easily.

3.2 Tiny LISA Traffic Sign Detection Dataset

The Tiny LISA dataset is a reduced version of the larger LISA Traffic Sign Dataset. It comprises of 900 annotated images spanning 9 distinct labels. Each annotation in the dataset is detailed across six columns: the first column specifies the image name, the subsequent four columns define the bounding box coordinates (x1, y1, x2, y2), and the final column denotes the label of the traffic sign.

3.3 LISA Traffic Light Dataset

The LISA Traffic Light Dataset contains 13 daytime and 5 nighttime clips containing a total of 43,007 frames and 113,888 traffic light annotations. The dataset contains each frame in a JPEG format and annotations are in a CSV format with bounding boxes (Upper Left X, Upper Left Y)(Lower Right X, Lower Right Y) and 6 classes (stop, warning, go, stop left, warning left, and go left).

4 Methods

We trained two separate instances, with one dedicated to training a neural network for general traffic light and sign detection using both the LISA Traffic Light and Traffic Sign Dataset. The other instance analyzed the accuracy difference for traffic sign detection between being trained on Tiny LISA or the combined dataset of Tiny LISA and LISA. The general architecture can be referenced in Figure 2.

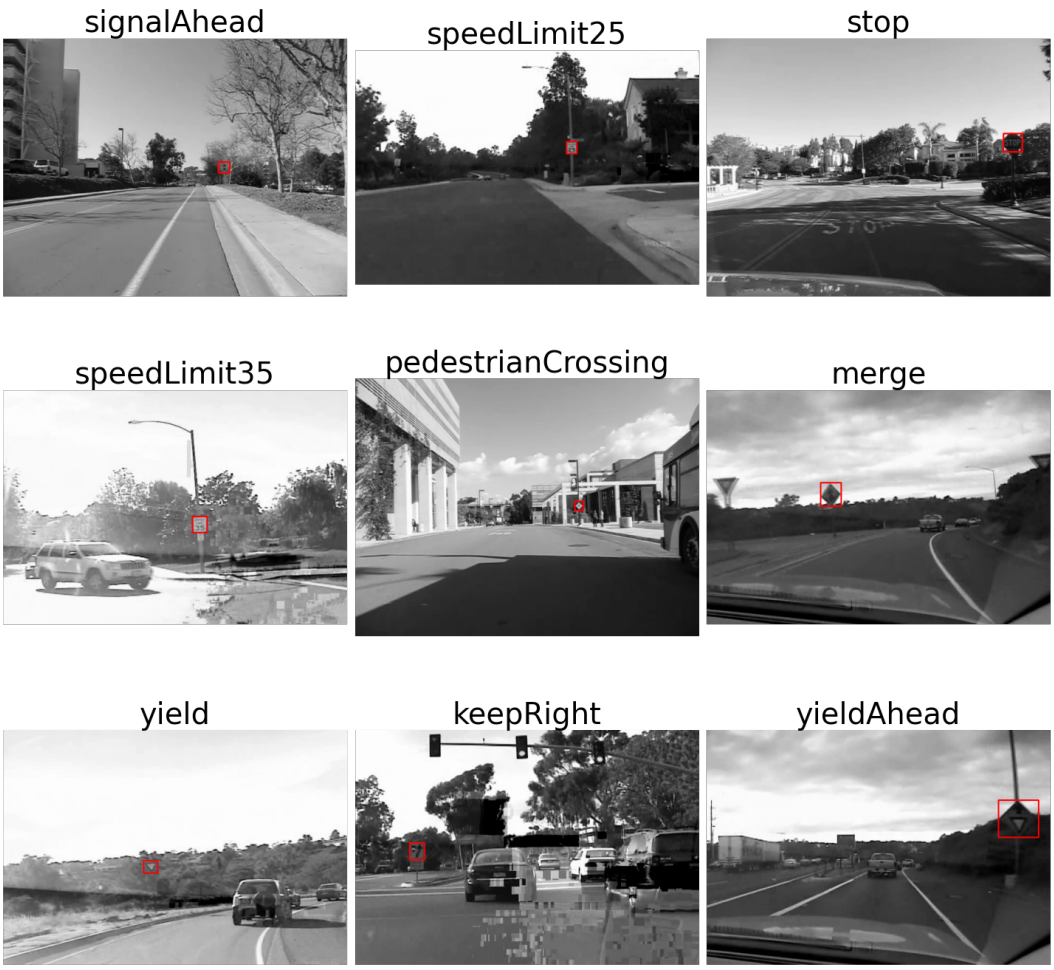

Figure 1: Labels of Tiny LISA Dataset

We perform data augmentation to process our data in order for them to be consistent while also meeting our specifications. We keep the channel count to be 32 but remain relatively shallow as we already allow the model to learn a large number of features at each layer. By adding more layers, we only increase the number of hyper-parameters which we do not want. The total number of parameters T in the network is given by the formula:

$$T = \sum_{\text{conv}} [(\text{kw} \times \text{kh} \times \text{ic} \times \text{oc}) + \text{oc}] + \sum_{\text{bn}} (2 \times \text{f}) + \sum_{\text{fc}} [(\text{if} \times \text{of}) + \text{of}] \tag{1}$$

where:

- T is the total number of parameters.
- kw and kh are the kernel width and height, respectively.
- ic and oc are the number of input and output channels for the convolutional layers.
- f is the number of features in the batch normalization layers.
- if and of are the number of input and output features in the fully connected layers.

4.1 Data Augmentation

In order to reduce the number of hyper-parameters as it must work under our computational limitations, we downsized all images to be 256 x 256. We then normalize the data so that we have a mean of

stage	output	ResNet-50	**ResNeXt-50 (32×4d)**
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
		$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times3$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128, C{=}32 \\ 1\times1, 256 \end{bmatrix} \times3$
conv3	28×28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times4$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256, C{=}32 \\ 1\times1, 512 \end{bmatrix} \times4$
conv4	14×14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times6$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512, C{=}32 \\ 1\times1, 1024 \end{bmatrix} \times6$
conv5	7×7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times3$	$\begin{bmatrix} 1\times1, 1024 \\ 3\times3, 1024, C{=}32 \\ 1\times1, 2048 \end{bmatrix} \times3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		$\mathbf{25.5}\times10^6$	$\mathbf{25.0}\times10^6$
FLOPs		$\mathbf{4.1}\times10^9$	$\mathbf{4.2}\times10^9$

Figure 2: ResNet vs ResNeXt [3]

0 and a standard deviation of 1 which reduces noise. For a small dataset of 900, this is critical in stabilizing our loss function.

4.2 ResNet

ResNet, or Residual Networks, leverage "shortcut connections" to tackle the vanishing gradients issue, enabling the training of much deeper neural networks. These connections allow layers to learn residual functions with reference to the input, making it easier to train deeper networks and improve performance on various tasks, as demonstrated by their success in competitions such as ILSVRC & COCO 2015 [4].

4.3 ResNeXt

The ResNeXt architecture, presented by Xie et al.[3], introduces a highly modularized network design for image classification that improves upon the ResNet model by introducing a new dimension of network design called "cardinality," in addition to depth and width. The architecture builds on the idea of repeating a building block that aggregates a set of transformations with the same topology, leading to a multi-branch architecture with few hyper-parameters.

As such, we used the ResNeXt architecture to take advantage of our limited computational resources so that we can get better accuracy than ResNet with similar hyper-parameters. With the usage of grouping, we can effectively maximize our usage of the GPU.

5 Experiments and Results

For training both models, we employed the Adam optimizer, chosen for its proficiency in handling noisy data and its convergence capabilities due to momentum, setting the learning rate at 1×10^{-3}. We opted out of using weight decay, considering the uniformity of traffic signs throughout the USA, which diminishes the need for such regularization. Our dataset, whether containing 900 or 7600 images, was consistently divided in a 4:3:3 ratio for training, validation, and testing, ensuring a

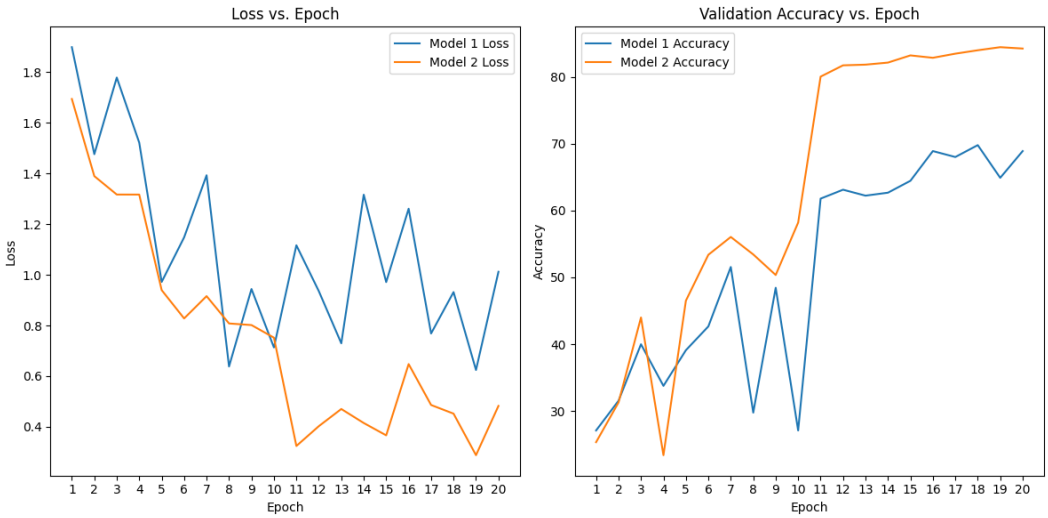

Figure 3: Tiny LISA vs LISA

standardized approach in model evaluation and performance comparison across different dataset sizes.

5.1 Traffic Signs

For this model, we used an NVIDIA RTX 3060i GPU and an Intel i9-10900k 1–core CPU. Training on the Tiny LISA Dataset took 2 minutes and 23 seconds, while the combined LISA and Tiny LISA datasets took 28 minutes and 38 seconds. The results, shown in Figure 3, indicate that trading off 20% in accuracy led to a ninefold decrease in training time.

We can see that the effects of data are in regards to time and accuracy. Our model trained on a smaller dataset requires less computational resources and achieves 60% accuracy, which meets our specifications.

5.2 Traffic Lights and Signs

For this model, we used an NVIDIA RTX 2060 GPU and an Intel i7-10750H 6-core CPU. Training on the combined LISA Trafic Sign dataset as well as the LISA Traffic Sign Dataset took on average 4 mins per epoch.

References

[1] Alvin Abraham, Djoko Purwanto, Hendra Kusuma. "Traffic Lights and Traffic Signs Detection System Using Modified You Only Look Once." *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[2] Oshada Jayasinghe, et al. "Towards Real-time Traffic Sign and Traffic Light Detection on Embedded Systems." *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[3] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He. "Aggregated Residual Transformations for Deep Neural Networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. "Deep Residual Learning for Image Recognition." *arXiv preprint arXiv:1512.03385*, 2015.