Identify and localize traffic lights within images by using CNN and RestNet

Hongyuan Hua **Qiaosong Deng**
Department of Electrical and Computer Engineering
University of Toronto University of Toronto
hongyuan.hua@mail.utoronto.ca *qiaosong.deng@mail.utoronto.ca*

Wei Tao
Department of Mechanical and Industrial Engineering
University of Toronto
wei.tao@mail.utoronto.ca

Abstract

This project is conducted to identify and localize traffic lights within different images by utilizing Convolutional Neural Networks (CNN) and ResNet architectures. Nowadays, with the incremental demand and rapid development of self-driving vehicles all around the world, the accurate detection of traffic lights is vitally essential for ensuring passengers' safety. To improve the detection accuracy and explore the extent of differences between CNN and ResNet on the traffic lights classification, we leverage a comprehensive collection of annotated images, which consisting of various traffic scenarios captured under different lighting conditions, and eventually our trained ResNet model achieved an average accuracy of 98.95% while the comparative baseline CNN achieved an average accuracy of 97.59%. These results illustrate the effectiveness of the skip connection technique in dealing with complex computer vision tasks and demonstrate our experimental contribution to the development of self-driving.

Assentation of Teamwork

Wei Tao: Searched dataset, evaluated model, researched related information, wrote report

Hongyuan Hua: Trained and evaluated model, researched architecture, implemented model, tuned parameters and hyperparameters

Qiaosong Deng: Searched and cleaned dataset, preprocessed data and built data loader, researched architecture, implemented models

1 Introduction

With the incremental demand and rapid development of self-driving, the market of self-driving vehicles has grown a lot in the past three years and is expected to keep dramatically growing in the following decades. [1] This phenomenon provides insight that it is crucial to ensure the safety of passengers in a self-driving vehicle. One of the main components in ensuring safety is accurately detecting the traffic lights. This task is deemed significantly challenging due to reasons such as lighting conditions, weather conditions, traffic scenarios, etc. Some proposed solutions are able to detect correctly during daytime but have poor performance in the nighttime.[2] Hence, it is necessary to develop robust detectors to precisely identify and localize traffic lights under various conditions.

The motivation behind our project is to enhance the safety and reliability of self-driving vehicles, which are the fundamentals of vehicle transportation. Through the explorations of the datasets that contain images from various angles, distances, lighting conditions, and weather conditions, we are supposed to achieve high accuracy in traffic light detection. Potentially, we could contribute to the advancement and development of self-driving technology. In addition, we are interested in exploring and comparing the performance of CNN and ResNet. CNN is known for solving challenging image pattern recognition.[3] At the same time, ResNet is a particular case of CNN with skip connections.[4] We are particularly curious about how the skip connections solve the problem of vanishing gradient during the training of the deep neural network.

Our solution is to leverage a comprehensive collection of annotated images, named the LISA Traffic Light Dataset, which consists of various traffic scenarios captured under different environmental conditions, weather conditions, etc., along with the labels indicating the position and status of each traffic light within the images. Then, train the ResNet-based CNN model to incorporate skip connections to enhance the learning of deep features that are critical for traffic light detection and classification. We would also introduce a traditional CNN model without skip connections as a comparative baseline. Both the ResNet-based CNN model and the traditional CNN model would be configured to process visual data from the LISA Traffic Light Dataset and concentrate on the precise detection and identification of traffic lights.

2 Problem Specification

We aim to develop a system capable of identifying and localizing traffic lights within diverse images using Convolutional Neural Networks (CNN) and ResNet architectures. The input comprises three-channel RGB images capturing various scenes that contain traffic lights. The images are derived from a dataset called LISA Traffic Light Dataset with annotated traffic lights, splitting it into 80% for training and 20% for testing to ensure model generalization. Employing the

---

[1] Self-driving Cars and Trucks Market Size, Share & Trends Analyze Report. https://www.grandviewresearch.com/industry-analysis/driverless-cars-market

[2] P. Kanchanamala, A. Dharani, J. S. Chandana, D. Jyothsna, I. S. Kumar and G. Kushila, "Traffic Light Detection System in Self-Driving Cars," *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, Salem, India, 2023, pp. 885-890

[3] K. O'Shea, R., Nah. "An Introduction to Convolutional Neural Networks". https://arxiv.org/abs/1511.08458

[4] K. He, X. Zhang, S. Ren, & J. Sun (2015). "Deep Residual Learning for Image Recognition". Microsoft Research.

cross-entropy loss function and softmax activation, we would train the model with Adam optimizer, incorporating batch-normalization layers to avoid overfitting. Hyperparameter tuning, including learning rates and number of epochs, will optimize model performance. Evaluation metrics would include test accuracy, train loss, precision, and F1 score. Additionally, we would provide a demo showcasing the model's functionality by passing sample images through the network, displaying classification results and highlighting the model's practical utility.

3 Design Details

We would first prepare the dataset, the annotations for the dataset are stored in CSV files, where each file contains information about the bounding boxes and labels of traffic lights in corresponding images. The images are stored in directories corresponding to different clips. We would implement a class responsible for loading the dataset, reading annotations and preparing data for training. This class would basically use Pandas to handle CSV files and PIL (Python Imaging Library) to process images.

Then the loaded images would be transformed by using 'transforms.Compose' from the 'torchvision.transforms' module. Transformations include resizing the image shape, converting to tensors, and normalization. The transformed images are split into 80% of the training set and 20% of the testing set, and wrapped into DataLoader with a certain batch size.

Our primary model architecture for ResNet would be based on ResNet-34, which is a variant of the ResNet architecture developed by Microsoft Research. The basic architecture of ResNet-34 consists of 16 blocks of convolutional layers, with each block containing two convolutional layers followed by batch normalization and ReLU activation functions. The input to a block is added element-wise to the output of the block to realize the skip connections. Figure 1 below is the basic architecture diagram of ResNet-34.[5] The model would consist of convolutional layers, batch normalization, ReLU activation functions, max-pooling, average-pooling, and fully connected layers.

[5]K. He, X. Zhang, S. Ren, & J. Sun (2015). "Deep Residual Learning for Image Recognition". Microsoft Research.

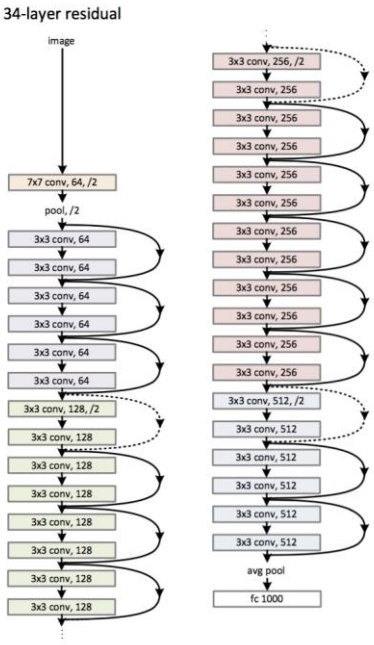


Figure 1:

As a comparative baseline, our model architecture for traditional CNN would be similar to the architecture of ResNet; it would consist of convolutional layers, batch normalization, ReLU activation functions, max-pooling, average-pooling, and fully connected layers; the only exception is the traditional CNN has no skip connections from the input of a block to the output of a block.

The training process would be conducted using stochastic gradient descent (SGD) and the Adam optimizer (torch.optim.Adam). Cross-entropy loss (torch.nn.CrossEntropy) is used as the loss function. The training loop iterates over the dataset for a certain number of epochs and updates trainable parameters based on computed gradients. During each epoch, the performance of ResNet is evaluated on the test dataset to monitor metrics such as train loss, test accuracy, precision, recall, and F1 score by calling 'accuray_score' and 'precision_recall_fscore_support' from sklearn.metrics.

4 Numerical Experiments

There are a total of 43,007 images and 113,888 annotated traffic lights.

- Label 0 indicates 'stop'
- Label 1 indicates 'stopLeft'
- Label 2 indicates 'go'
- Label 3 indicates 'goLeft'
- Label 4 indicates 'warningLeft'
- Label 5 indicates 'warning'

We implemented the TrafficLightDataset class to load the dataset, read annotations, and prepare data for training. This class basically uses Pandas to handle CSV files and PIL (Python Imaging Library) for image processing.

Then, the loaded images are transformed by using 'transforms.Compose' from the 'torchvision.transforms' module. Transformations include resizing the image shape into 224x224, converting to tensors, and normalization with the mean values of [0.485, 0.456, 0.406] and the standard deviation values of [0.229, 0.224, 0.225]. The transformed images are split into 80% of the training set and 20% of the testing set, and wrapped into DataLoader with a batch size of 100.

Our primary model architecture for ResNet is based on ResNet-34, as introduced in the last part. The classes of 'BasicBlock' and 'ResNet' are implemented to define the building blocks and the overall architecture of ResNet-34. The following is our detailed architecture of ResNet in sequence:

- The input convolutional layer has 64 filters with a size of 7x7, stride of 2 and padding of 3.
- Three blocks group 6 convolutional layers together, each with 64 filters of size 3x3, stride of 1 and padding of 3.
- A max pooling layer with a size of 3x3, stride of 2 and padding of 1.
- Four blocks group 8 convolutional layers together, each with 128 filters of size 3x3, stride of 2 and padding of 3.
- Six blocks group 12 convolutional layers together, each with 256 filters of size 3x3, stride of 2 and padding of 3.
- Three blocks group 6 convolutional layers together, each with 512 filters of size 3x3, stride of 2 and padding of 3.
- An average pooling layer downsamples the input tensor to a spatial size of (1, 1).
- A fully connected layer takes input size 512 and outputs 6 classes.

During the forward propagation in each block, the input to a block is added element-wise to the output of the block to realize the skip connections. We implement this functionality in the class of 'BasicBlock':

```
def forward(self,x):
    identity = x

    out = self.conv1(x)
    out = self.bn1(out)
    out = self.relu(out)

    out = self.conv2(out)
    out = self.bn2(out)

    if self.downsample is not None:
      identity = self.downsample(x)

    out += identity
    out = self.relu(out)

    return out
```

self.downsample consists of a convolutional layer and batch normalization:

```
    downsample = nn.Sequential(
```

```
        nn.Conv2d(self.in_channels, out_channels *
block.expansion, kernel_size=1, stride=stride,
bias=False),
        nn.BatchNorm2d(out_channels * block.expansion),
    )
```

As a comparative baseline, our model architecture for traditional CNN is implemented in the classes of 'PlainBlock' and 'PlainCNN'. The architecture is similar to the architecture of ResNet above, the only exception is that traditional CNN has no skip connections from the input of a block to the output of a block.

The training process is conducted using stochastic gradient descent (SGD) and the Adam optimizer (torch.optim.Adam) with a learning rate of 0.0001. Cross-entropy loss (torch.nn.CrossEntropy) is used as the loss function. The training loop iterates over the dataset for 10 epochs and updates trainable parameters based on computed gradients. During each epoch, the performance of ResNet is evaluated on the test dataset to monitor metrics such as train loss, test accuracy, precision, recall, and F1 score by calling 'accuray_score' and 'precision_recall_fscore_support' from sklearn.metrics. Figure 2 below shows the training results against the number of epochs for the ResNet and Figure 3 shows the training results against the number of epochs for the traditional CNN.

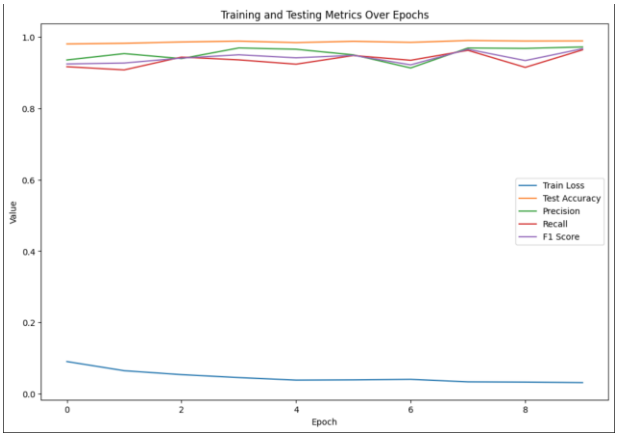


Figure 2: training results against number of epochs for ResNet

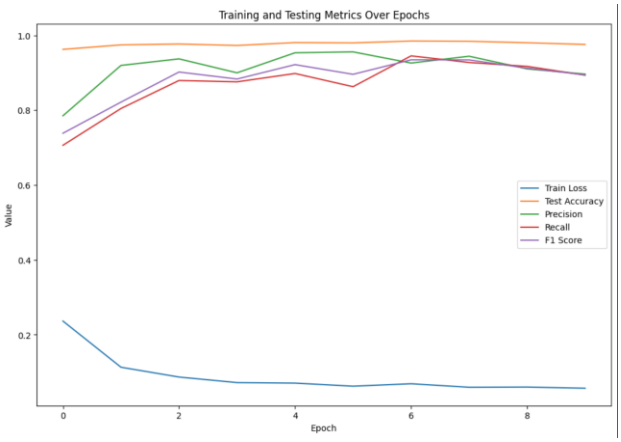


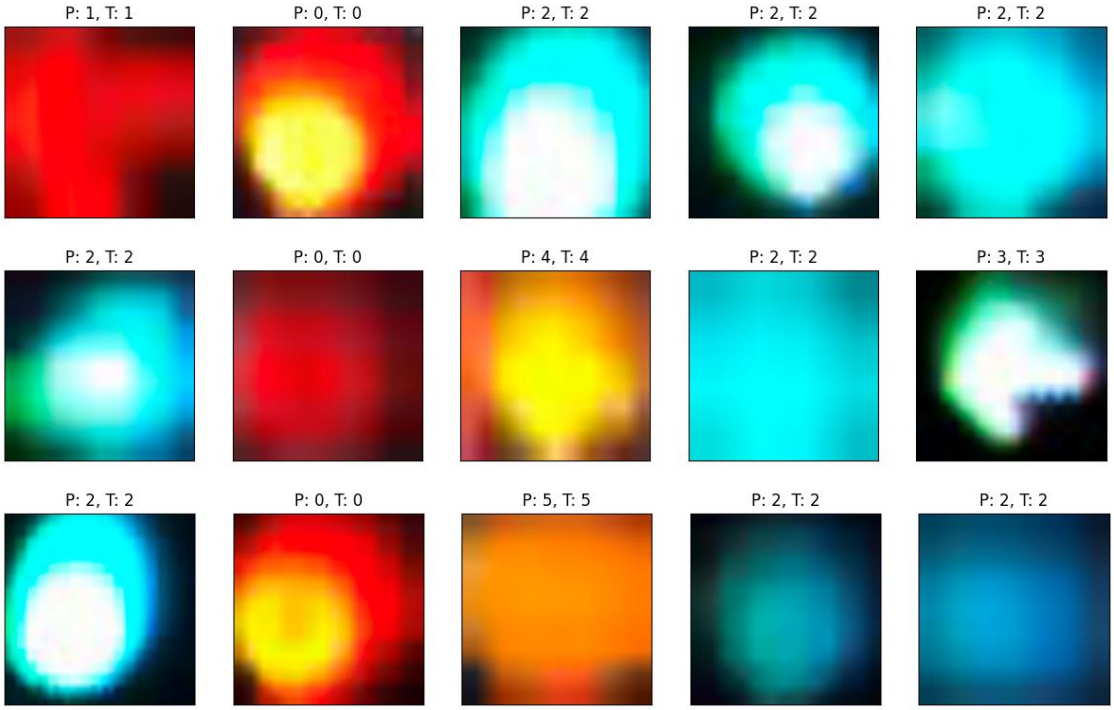
Figure 3: training results against number of epochs for traditional CNN

Table 1: Results

Architecture	Train Loss	Test Accuracy	Precision	Recall	F1 Score
CNN	0.0569	97.59%	89.67%	89.36%	89.46%
ResNet	0.0312	98.95%	97.24%	96.44%	96.76%

The demo of our model performance is shown below with selected images and corresponding predictions, ‘P’ indicates prediction, ‘T’ indicates true label:

Figure 4:



5 Conclusions

Overall, the best performance results in a 98.95% accuracy for identifying and localizing traffic lights through ResNet, which is ideal for our objective. Meanwhile, CNN also has a high accuracy of 97.59%, which is a bit lower than the performance of ResNet. However, in terms of precision, recall, and F1 Score, CNN has significant differences from the performance of ResNet, as seen in Table 1. Thus, in the aspect of identifying and localizing traffic lights, ResNet performs more robustly than traditional CNN.

References

[1] Self-driving Cars and Trucks Market Size, Share & Trends Analyze Report. https://www.grandviewresearch.com/industry-analysis/driverless-cars-market

[2] P. Kanchanamala, A. Dharani, J. S. Chandana, D. Jyothsna, I. S. Kumar and G. Kushila, "Traffic Light Detection System in Self-Driving Cars," *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, Salem, India, 2023, pp. 885-890.

[3] K. He, X. Zhang, S. Ren, & J. Sun (2015). "Deep Residual Learning for Image Recognition". Microsoft Research.

[4] K. O'Shea, R., Nah. "An Introduction to Convolutional Neural Networks". https://arxiv.org/abs/1511.08458

[5] M. B. Jensen, M. P. Philipsen, Andreas Møgelmose, T. Moeslund, M. Trivedi (2015, September). Traffic light detection: A learning algorithm and evaluations on challenging dataset. In intelligent transportation systems (ITSC), 2015 IEEE 18th international conference on (pp. 2341-2345). IEEE.