

Road Sign Recognition with CNN and RCNN

Aksel Tahir 6548051

11.05.2021

Contents

1	Introduction	2
2	Related Work	2
3	Architectures	2
3.1	CNN	2
3.2	R-CNN	2
3.3	Faster R-CNN	2
4	Dataset	2
4.1	GTSRB - German Traffic Sign Recognition Benchmark	2
4.1.1	Loading the dataset	2
4.1.2	Preprocessing data	2
5	Implementation	3
5.1	CNN	3
5.1.1	Structure	3
5.1.2	Performance	3
5.2	R-CNN	3
5.3	Faster R-CNN	3
6	Analysis and Evaluation	3
7	Conclusions	3

1 Introduction

Road signs are a present system in virtually all road infrastructure. They are of critical importance to interpreting correct road usage, road regulations and route recommendations. Their presence is integral to the safe and functional road use.

Contemporary road signs follow strict design rules to optimise their clarity of intention. These rules allow them to be as easy as possible for human interpretation. However, humans are prone to distraction, misinterpretation and other general mistakes, which is why road sign recognition (RSR) algorithms are a fast-advancing point of development in autonomous driving research.

Standard computer vision methods are not versatile enough to deal with the plethora of different physical road conditions. This is why applying a deep learning approach to the problem is necessary - A well crafted AI can exceed even human vision in RSR.

In this project we propose an RSR solution using several different neural network models and evaluating their performance. Since standard computer vision methods are not versatile enough to deal with the plethora of different physical road conditions, it is necessary to apply a deep learning approach to the problem - A well crafted AI can exceed even human vision in RSR.

2 Related Work

With the advent of AI computing autonomous and assisted driving has been an area of extensive research. Road sign recognition (RSR) systems are integral to the field. The functional implementation of RSR systems depends on two related issues - Road sign detection (RSD) and road sign classification (RSC). RSD pertains to localising the relevant information from the data, and RSC to identifying the data with its correct labels. There have already been a number of outstanding studies in the detection and classification in [1], [2], [3], [4], [5].

In many RSR systems, the RSD part is done via conventional machine learning methods. This approach is used by authors like Amal Bouti et al in [6]. The advantages of deep learning methods have made themselves clear by now, but the author has chosen to discuss if SVMs do not still have a place in RSR. In their findings, "the representation of the HOG features and SVM greatly improves the results obtained and shows good results in terms of accuracy. The linear SVM not only achieves high accuracy but also costs least compared with another kernel function" [6, 6722 A. Bouti et al]

[Add info about R-CNN too. That'd be pretty useful]

3 Architectures

This project is based entirely on image recognition, the models we have decided to implement are all CNN-based. This section will describe them all in detail.

3.1 CNN

Recent decades have seen many advancements in representational learning from raw data, with one of the most pertinent approaches to the method being the varieties of CNN modelling. CNN dominates systems related to object recognition and detection. This ubiquity in the field is one of the reasons that made us choose it for our implementation.

Another major advantage of CNNs that led to our choice is the relatively sparse pre-processing required to make a functioning model, compared to more traditional image classifiers. The independent kernel optimisation that CNNs learn, without much prior knowledge is hugely beneficial to implementation complexity.

3.2 R-CNN

At the time of writing the R-CNN model has not been implemented.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

3.3 Faster R-CNN

At the time of writing the Faster-R-CNN model has not been implemented.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

4 Dataset

This project intends to use two datasets to conduct its study to ensure flexibility and better insight. The data needed for detection differs from that needed for classification, and many datasets do not meet both requirements. This section will discuss our choices and their features.

4.1 GTSRB - German Traffic Sign Recognition Benchmark

The German Traffic Sign Recognition Benchmark is our first choice. It contains 43 data classes of road signs in Germany with each having enough examples to be used in training.

- As seen from the examples in Figure 1 dataset is only useful for classification purposes and not detection, since its items only feature photos of isolated road signs with no irrelevant data in the image.



Figure 1: Some examples of raw data from the dataset

- The dataset also contains unequal numbers of items in each class, as seen in Figure 2, which introduces a challenge solved in Preprocessing.

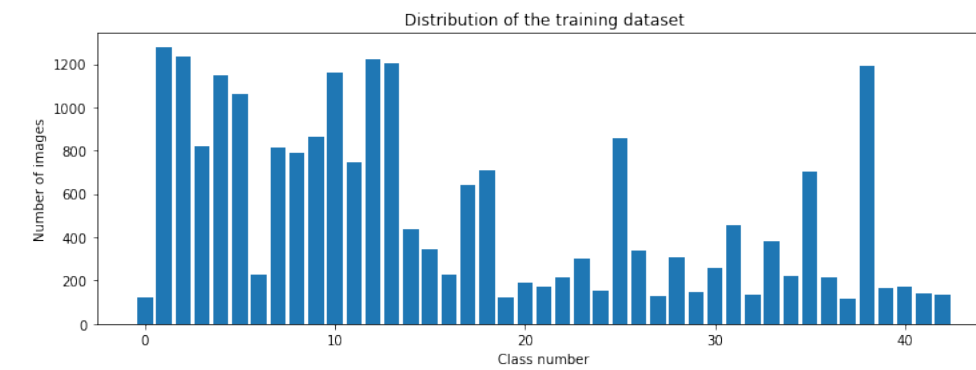


Figure 2: Number of items in each class of GTSRB

4.1.1 Loading the dataset

The 50K images loaded from the dataset were split into two groups for training and testing. The split was arbitrarily chosen to be 0.8/0.2 respectively. Out of the 80% chosen for training, 20% were chosen for validation.

```
1 testRatio = 0.2
2 # set aside 20% of images for testing, 80% for training
3 validationRatio = 0.2
4 # of all training images, set aside 20% for validation
5
6 X_train, X_test, y_train, y_test =
7 train_test_split(images, classNo, test_size=testRatio)
8 X_train, X_validation, y_train, y_validation =
9 train_test_split(X_train, y_train, test_size=validationRatio)
10
11 # X_train = ARRAY OF IMAGES TO TRAIN
12 # y_train = CORRESPONDING CLASS ID
```

4.1.2 Preprocessing data

There are quite a few tasks to be done with preprocessing the data from this dataset.

1. The uneven distribution from Figure 2 needs to be processed to make every class equal. At the time of writing this has not been implemented.
2. The images need to be visually augmented. They are converted to a monochromatic greyscale and then undergo histogram equalisation. Finally their values are normalised from 0-255 to 0-1.

5 Implementation

5.1 CNN

Modelling our CNN is very straightforward with Keras. The model we've implemented closely resembles LeNet (Figure 3), proposed by Yann LeCun et al in 1998. The dataset we've trained the model upon is the GTSRB.

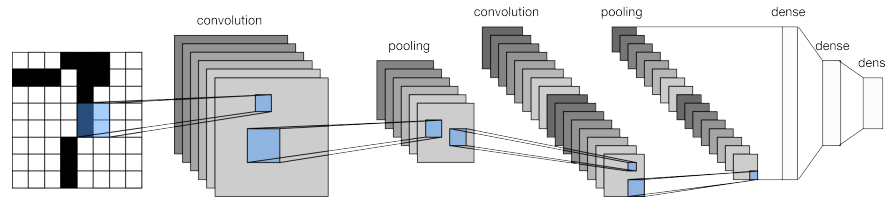


Figure 3: Generic LeNet structure

5.1.1 Structure

Model: "sequential_1"		
Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 28, 28, 60)	1560
conv2d_5 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d_2 (MaxPooling2D)	(None, 12, 12, 60)	0
conv2d_6 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_7 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_3 (MaxPooling2D)	(None, 4, 4, 30)	0
dropout_2 (Dropout)	(None, 4, 4, 30)	0
flatten_1 (Flatten)	(None, 480)	0
dense_2 (Dense)	(None, 500)	240500
dropout_3 (Dropout)	(None, 500)	0
dense_3 (Dense)	(None, 43)	21543
Total params: 378,023		
Trainable params: 378,023		
Non-trainable params: 0		

Figure 4: Structure of our CNN model

The first two layers are convolutions. Layer 1 has a depth of 60, a filter size of (5, 5), with relu activation. The second layer does the same. Following the second convolution layer is a max polling layer. Following the LeNet model, we use a (2,2) kernel size. It effectively downsizes the data by only selecting the max value pixel for adjacent pixels.

Next we implement two other convolution layers with new parameters. Half the number of filters and a filter size of (3,3). Their activation is also relu. They're followed by a max pooling layer and a dropout layer to prevent overfitting the model.

Afterwards, the data is flattened and a dense layer with relu activation is used. A second dropout layer prepares the model for the a last fully connected dense layer with class number as size (ie. 43). Softmax is used to return the probabilities of each class, and the model is compiled with the Adam optimiser.

Within 10 epochs, the accuracy of both training and validation is above 0.98, after which the model shows minimal improvements and the cost/gain balance becomes unreasonable.

5.1.2 Performance

The simple CNN model achieved formidable results with relatively light training. After roughly 30 minutes on Heron labs computers and 10 epochs the test accuracy averaged in the 98th percentile. Figure 5 shows the relevant values throughout training

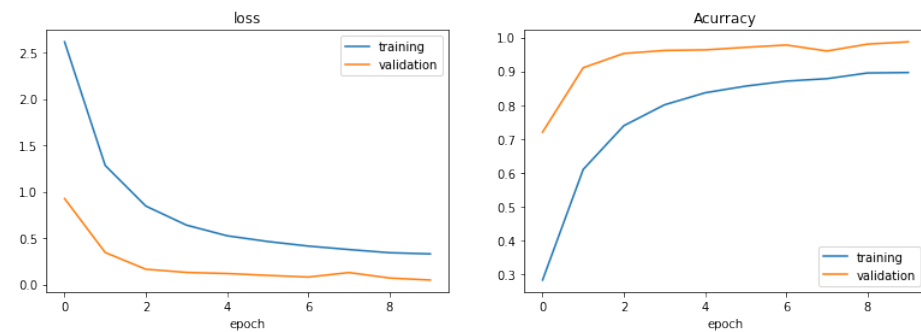


Figure 5: The Loss and Accuracy values throughout the epochs

5.2 R-CNN

5.3 Faster R-CNN

6 Analysis and Evaluation

7 Conclusions

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

References

- [1] Smit Mehta, Chirag Paunwala, and Bhaumik Vaidya. Cnn based traffic sign classification using adam optimizer. In *2019 International*

Conference on Intelligent Computing and Control Systems (ICCS), pages 1293–1298, 2019.

- [2] Jianming Zhang, Wei Wang, Chaoquan Lu, Jin Wang, and Arun K. Sangaiah. Lightweight deep network for traffic sign classification. *Annales des tlcommunications*, 75(7-8):369–379, Aug 2020.
- [3] S. Mehta, C. Paunwala, and B. Vaidya. Cnn based traffic sign classification using adam optimizer. In - *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1293–1298, 2019. ID: 1.
- [4] D. Ciresan, U. Meier, J. Masci, and J. Schmidhuber. A committee of neural networks for traffic sign classification. pages 1918–1921. IEEE, Jul 2011.
- [5] Mrinal Haloi. Traffic sign classification using deep inception based convolutional networks. Nov 10, 2015.
- [6] Amal Bouti, Med A. Mahraz, Jamal Riffi, and Hamid Tairi. A robust system for road sign detection and classification using lenet architecture based on convolutional neural network. *Soft computing (Berlin, Germany)*, 24(9):6721–6733, May 2020.
- [7] Reza F. Rachmadi, Keiichi Uchimura, and Yoshinori Komokata. Japan road sign classification using cascade convolutional neural network, Jan 01, 2016.
- [8] Reza F. Rachmadi, Reza F. Rachmadi, Keiichi Uchimura, and Gou Koutaki. Road sign classification using spatial pyramid convolutional neural network campus grid and render farm view project face analysis view project road sign classification using spatial pyramid convolutional neural network, 20-03 1725.
- [9] Amal Bouti, Med A. Mahraz, Jamal Riffi, and Hamid Tairi. A robust system for road sign detection and classification using lenet architecture based on convolutional neural network. *Soft computing (Berlin, Germany)*, 24(9):6721–6733, May 2020.
- [10] Danyah A. Alghmgham, Ghazanfar Latif, Jaafar Alghazo, and Loay Alzubaidi. Autonomous traffic sign (atsr) detection and recognition using deep cnn. *Procedia computer science*, 163:266–274, 2019.
- [11] Faming Shao, Xinqing Wang, Fanjie Meng, Jingwei Zhu, Dong Wang, and Juying Dai. Improved faster r-cnn traffic sign detection based on a second region of interest and highly possible regions proposal network. *Sensors (Basel, Switzerland)*, 19(10):2288, May 17, 2019.
- [12] Rongqiang Qian, Qianyu Liu, Yong Yue, Frans Coenen, and Bailing Zhang. Road surface traffic sign detection with hybrid region proposal and fast r-cnn. pages 555–559. IEEE, Aug 2016.
- [13] Alexander Shustanov and Pavel Yakimov. Cnn design for real-time traffic sign recognition. *Procedia Engineering*, 201:718–725, 2017. ID: 278653.