# GTSRB http://benchmark.ini.rub.de/?section=gtsrb&subsection=news

## Der letzte Führerscheinneuling ist schon geboren!

### Setup

**Dataset: German Traffic Sign Recognition Benchmark**
**Training Images: 39'254 / Testing Images: 12'630**

**Environment:** CO + Drive

**Multi-Class, Single Image Classification with CNNs using KERAS**

### Challenges

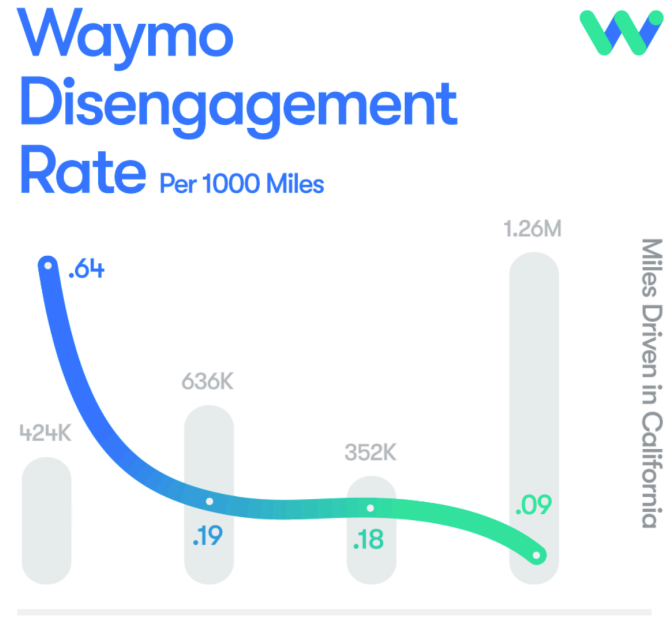
- Amount of data to process (processing time)
- Image alignment
- Compatibility of open source packages

### Lessons learned

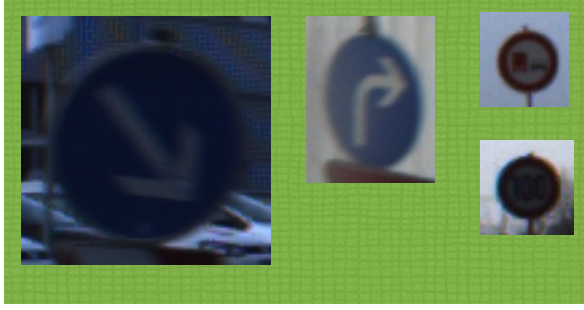
- ✓ Make Use of H5 Files to save data and trained models
- ✓ Decide what to do local and what in the cloud (load h5 files with data => local / train model => cloud)
- ✓ Amount of training data is critical to success
- ✓ Proper infrastructure is a must (CPU/GPU etc.)
- ✓ Dealing with open source packages can be a pain (version compatibility)

# Waymo launches its first commercial self-driving car service

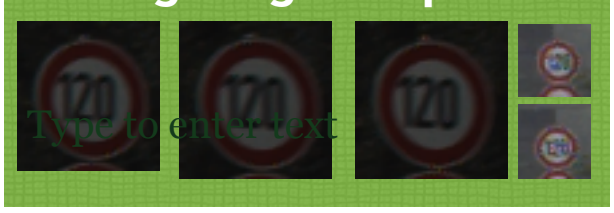
Waymo One's on-demand autonomous rides come with human backup for now.




### Test Image Samples



### Training Image Samples



### Zeit pro Epoche

**TPU** - 34s 877us/sample

**GPU** - 10s 261us/sample

**CPU** ETA: 24:27 - ETA: 5:51:05

### Prepare Training Images

- Random Load Image/Label from G Drive
- PreProcess Images
  - Histogram Normalization / Rescale etc.
- Store in H5 file for later use

### Improve Model (IM) & Train again

- Using Keras built-in data augmentation features (ImageDataGenerator) to increase amount of training data
- Slightly manipulate original images on the fly
- Explodes Training Data and training =>

```
Test accuracy = 0.9771971496437054 T
PredictLabel [16  1 38 ... 41  7 10]
TestLabel [16  1 38 ...  6  7 10]
```

### Create Model (BM)

- Sequential
- 6 convolutional layer with ReLU activation
- Use of MaxPooling & DropOut in between
- 1 fully connected hidden layer
- Optimizer: Stochastic gradient descent (SGD) + Nesterov enabled
- Loss Function: categorical_crossentropy
- Goal: Accuracy

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_1 (Conv2D) | (None, 32, 48, 48) | 896 |
| conv2d_2 (Conv2D) | (None, 32, 46, 46) | 9248 |
| max_pooling2d_1 (MaxPooling2 | (None, 32, 23, 23) | 0 |
| dropout_1 (Dropout) | (None, 32, 23, 23) | 0 |
| conv2d_3 (Conv2D) | (None, 64, 23, 23) | 18496 |
| conv2d_4 (Conv2D) | (None, 64, 21, 21) | 36928 |
| max_pooling2d_2 (MaxPooling2 | (None, 64, 10, 10) | 0 |
| dropout_2 (Dropout) | (None, 64, 10, 10) | 0 |
| conv2d_5 (Conv2D) | (None, 128, 10, 10) | 73856 |
| conv2d_6 (Conv2D) | (None, 128, 8, 8) | 147584 |
| max_pooling2d_3 (MaxPooling2 | (None, 128, 4, 4) | 0 |
| dropout_3 (Dropout) | (None, 128, 4, 4) | 0 |
| flatten_1 (Flatten) | (None, 2048) | 0 |
| dense_1 (Dense) | (None, 512) | 1049088 |
| dropout_4 (Dropout) | (None, 512) | 0 |
| dense_2 (Dense) | (None, 43) | 22059 |

Total params: 1,358,155
Trainable params: 1,358,155
Non-trainable params: 0

### Train Model (BM)

- Load H5 file with training images
- Train with batch size 32 over 30 epochs
- Use ModelCheckPoint to save best as H5 file

### Score Model

- Load trained Model from H5 file
- Load test Images from H5 file
- Score Model

```
Test accuracy = 0.9737318840579711
```