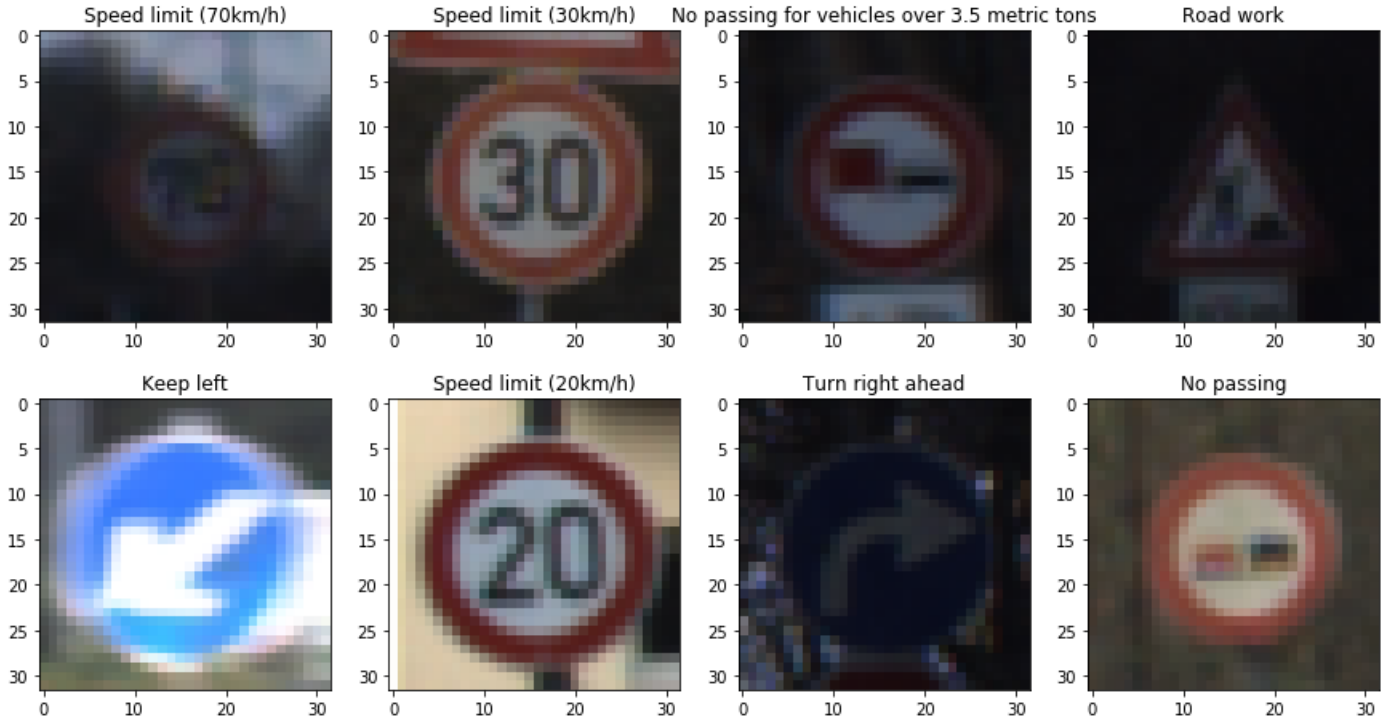
NM6101072 張育嘉

German Traffic Sign Classifier Project

- 使用工具
  - Platform：Jupyter Notebook
  - Language：Python 3.8
  - Packages：
    - NumPy
    - OpenCV
    - Matplotlib
    - Sklearn
    - Tensorflow
- Dataset Summary & Visualization
- Data Preprocessing
- Model Architecture
- Model Training and Evaluation
- Testing the Model
- Plotting activation features
- 心得

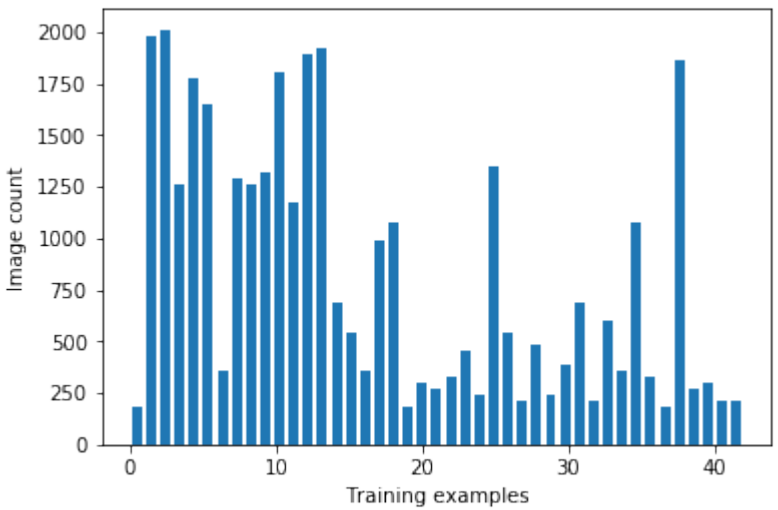
Dataset Summary & Visualization

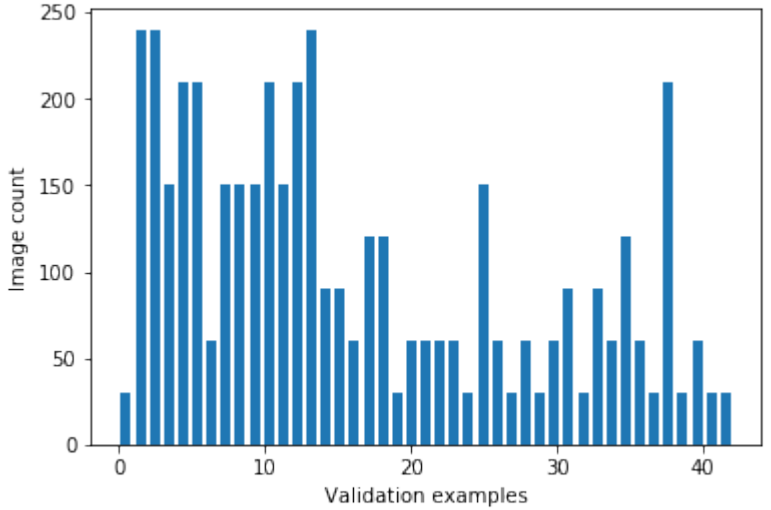
- 使用numpy讀取training、testing、validation照片數量及大小跟類別數量
  - Number of training examples: 34799
  - Number of testing examples: 12630
  - Number of validation examples: 4410
  - Image data shape = (32, 32, 3)
  - Number of classes = 43
- 使用matplotlib顯示8張training set裡面的圖

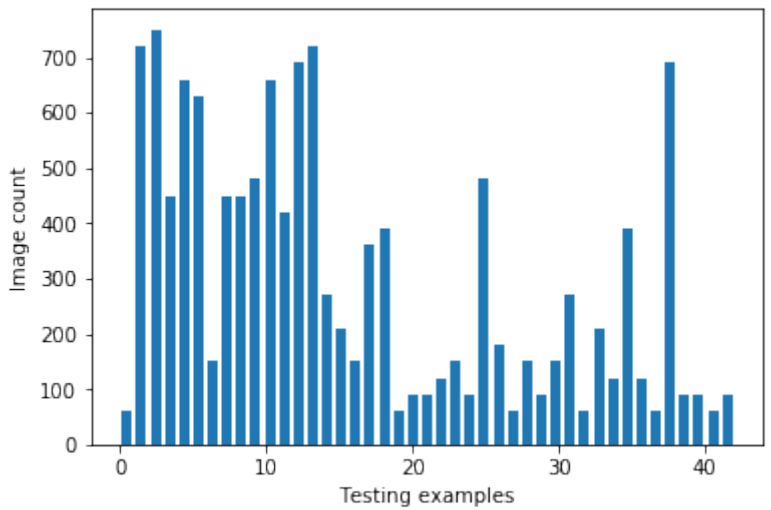


Dataset Summary & Visualization

- 使用numpy讀取training、testing、validation各個類別的數量
- 使用matplotlib顯示

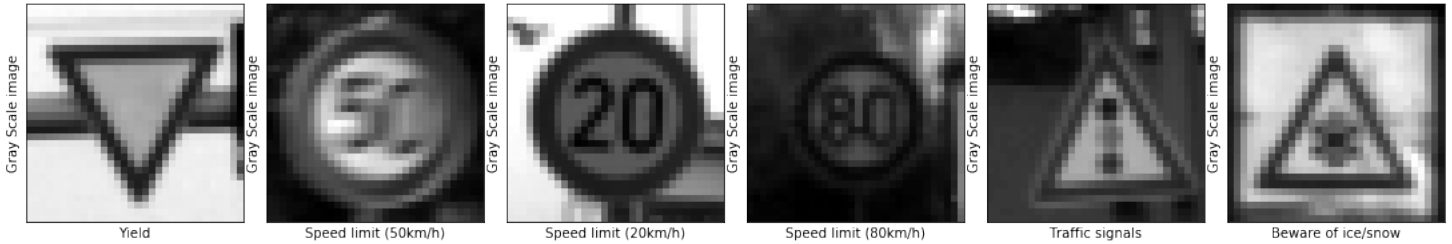




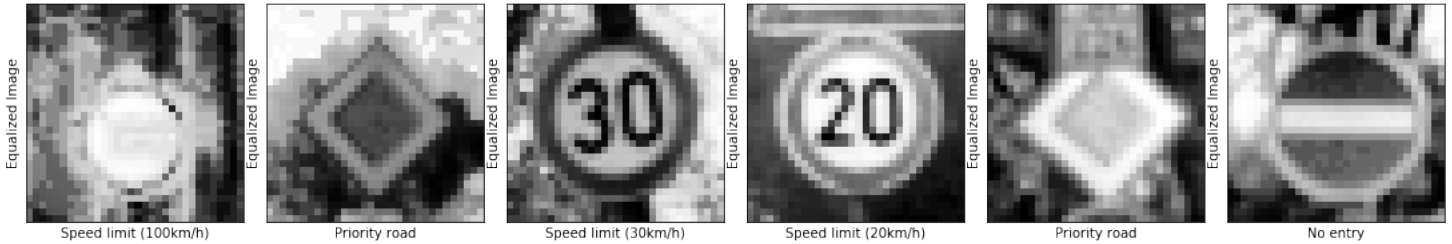


Data Preprocessing

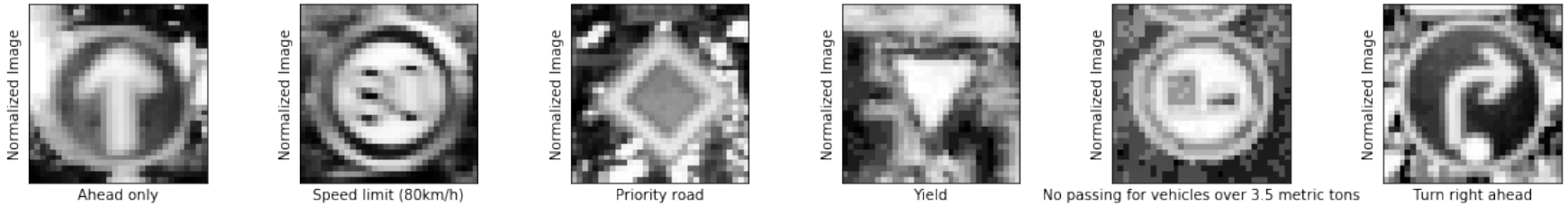
- Shuffling：使用sklearn增加training data的隨機性。
- Grayscaling：使用OpenCV將資料轉成灰階，增加ConvNet準確度。



- Local Histogram Equalization：使用skimage增強影像對比。



- Normalization：資料歸一化，提升模型準確度。



Model Architecture

- VGGNet精確度比LeNet好
- 使用12層VGGNet
  - Layer 1 (Convolutional): The output shape 32x32x32.
  - Layer 2 (Convolutional): The output shape 32x32x32.
  - Layer 3 (Pooling) The output shape 16x16x32.
  - Layer 4 (Convolutional): The output shape 16x16x64.
  - Layer 5 (Convolutional): The output shape 16x16x64.
  - Layer 6 (Pooling) The output shape 8x8x64.
  - Layer 7 (Convolutional): The output shape 8x8x128.
  - Layer 8 (Convolutional): The output shape 8x8x128.
  - Layer 9 (Pooling) The output shape 4x4x128.
  - Flattening: Flatten the output shape of the final pooling layer.
  - Layer 10 (Fully Connected): This should have 128 outputs.
  - Layer 11 (Fully Connected): This should have 128 outputs.
  - Layer 12 (Fully Connected): This should have 43 outputs.

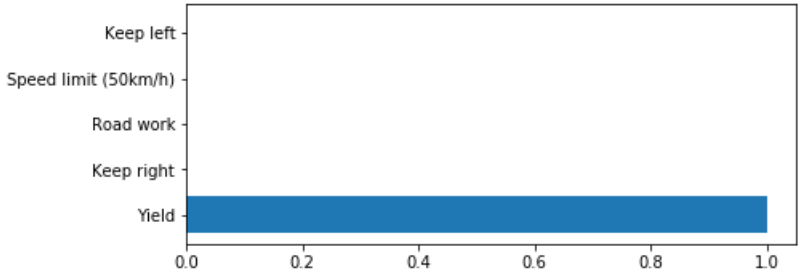
Model Training and Evaluation

- 參數設定
  - 平均值 mu 設為0 ，標準差 sigma設為0.1，學習率設為0.001，dropout：0.5(fully-connected layer)、0.7(convolutional layer)
- Training準確度
  - EPOCH 1 : Validation Accuracy = 31.66%
  - EPOCH 2 : Validation Accuracy = 59.60%
  - EPOCH 3 : Validation Accuracy = 78.64%
  - EPOCH 4 : Validation Accuracy = 88.62%
  - EPOCH 5 : Validation Accuracy = 92.81%
  - EPOCH 6 : Validation Accuracy = 95.60%
  - EPOCH 7 : Validation Accuracy = 96.67%
  - EPOCH 8 : Validation Accuracy = 97.53%
  - EPOCH 9 : Validation Accuracy = 98.39%
  - EPOCH 10 : Validation Accuracy = 98.32%
  - EPOCH 11 : Validation Accuracy = 98.78%
  - EPOCH 12 : Validation Accuracy = 98.73%
  - EPOCH 13 : Validation Accuracy = 98.62%
  - EPOCH 14 : Validation Accuracy = 98.57%
  - EPOCH 15 : Validation Accuracy = 99.03%
- 在15個epoch之後，準確度已到達99%。

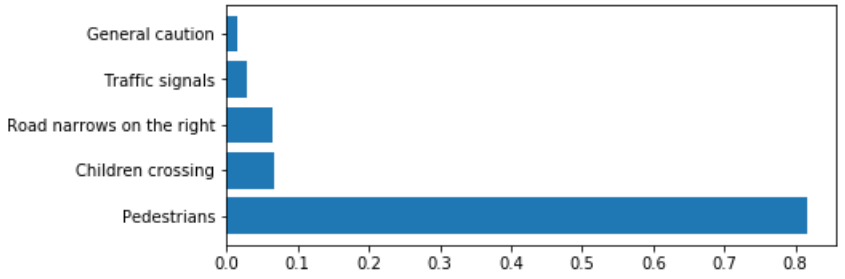
Testing the Model

- 網路上找5張German traffic signs來測辨識效果
  - 在行人號誌中，模型有80%可信度，猜測是因為三角形裡面的雜訊比較多。
  - 在限速號誌中，模型準確辨識該圖是限速號誌，有90%以上可信度。
  - 其餘交通號誌可信度都達到99%以上。

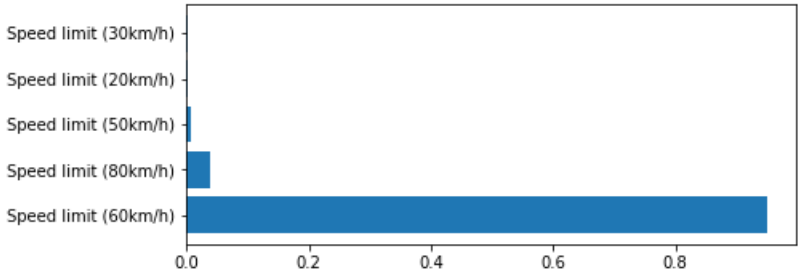
Yield



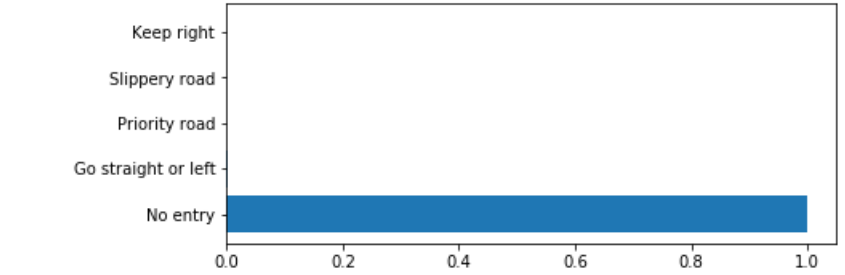
Pedestrians



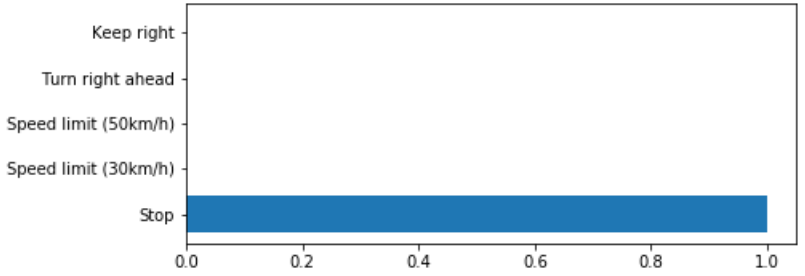
Speed limit (60km/h)



No entry

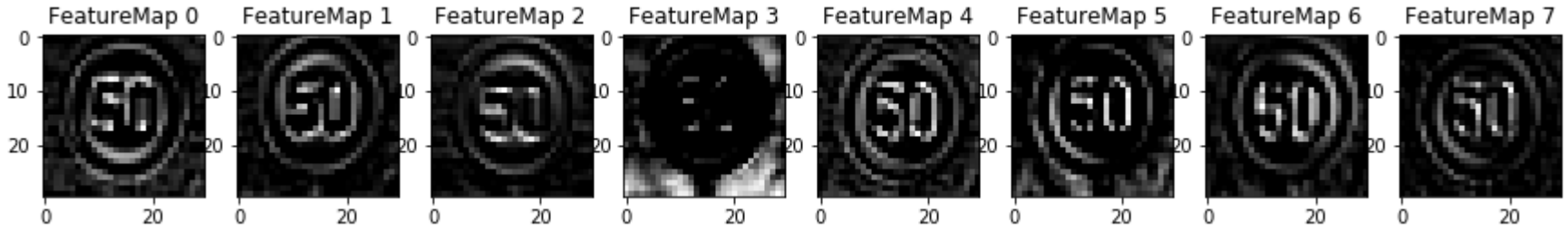


Stop



Plotting activation features

- 使用tensorflow畫出特徵圖，觀察神經網路的權重變化。
  - 透過可視化的特徵圖，我們可以理解神經網路在每一階段關注的特徵是哪些。
  - 可以看到有的階段對交通號誌輪廓感興趣，有的對數字感興趣。



心得

- 提高模型準確度
  - 使用 VGGNet可以提高模型的準確度，在9個epoch以後準確率就達到98%，可以透過減少epoch數來降低訓練時間。
  - 增加其他預處理技術也可以提高模型的準確度。
- 降低運算量
  - 置換模型架構，比如MobileNet。
- 其他應用
  - 增加分類數量，使模型不只可以分辨交通號誌。