國立成功大學

自動駕駛實務 Project 2報告

German Traffic Signs Classifier

學生: 陳育政

學號: E24094198

指導教授: 傅志雄、莊智清

大綱

1. **前言**
2. **實作方法**
3. Load the Data
4. Dataset Summary & Exploration
5. Pre-process Dataset

* Convert to Grayscale
* Data Augmentation
* Normalize Dataset

1. Train the LeNet5 Model
2. Accuracy Curves, Confusion Matrix and Classification Report
3. Test LeNet5 model on New Images

**三、問題討論**

1. The Necessity of Data-Preprocessing

**四、結論**

**五、參考資料**

1. **前言**

Project 2的目的是試圖利用由Yann LeCun於1995年所提出的LeNet5模型架構去解決German Traffic Sign Recognition Benchmark(GTSRB)的問題，而GTSRB資料集共有43種不同情況的德國交通號誌RGB相片，例如:限速、右轉、禁止通行…等圖示。整體處理資料的pipeline包括資料前處理、建構訓練LeNet5模型以及測試模型的準確率，最後的目標就是訓練好的LeNet5模型可以針對不曾餵入訓練的德國交通號誌照片成功辨認種類，達成自動駕駛中不可或缺的辨識路標的功能。

1. **實作方法**
2. Load the Data

根據Fig. 1.的程式碼讀取p-code file，包括training、validation及test的dataset。

**A screen shot of a computer code

Description automatically generated**

Fig. 1. 讀取資料集

1. Dataset Summary & Exploration

印出有關dataset的基本資料，包括各資料集比數、圖片大小以及交通號誌種類數等數據，結果如Fig. 2.所示。

A screen shot of a computer

Description automatically generated

Fig. 2. 資料集的基本資料

除了印出dataset的基本資料，我還嘗試隨機挑選幾張資料集內的圖片顯示，讓我對於即將處理的資料有進一步的認知，結果如Fig. 3.所示，可以發現影像都是RGB，並且大小一致。

A collage of different road signs

Description automatically generated

Fig. 3. 隨機挑選8張traffic sign images觀察

接著，為了更好的了解資料集中的交通號誌種類分布，我將每個資料集的各個class的總數以直線圖呈現，training、validation、test dataset的直線圖依序由Fig. 4. (a)、(b)、(c)呈現。

A graph of blue and white bars

Description automatically generatedA graph of blue lines

Description automatically generatedA graph of blue lines

Description automatically generated

Fig. 4. (a)training dataset各class資料筆數 (b)validation dataset各class資料筆數 (c)test dataset各class資料筆數

1. Pre-process Dataset

* Convert to Grayscale

根據Fig. 5.的程式碼，我將原先資料集內的RGB影像皆轉成灰階圖，grayscale的效果如Fig. 6.呈現，而轉成灰階的原因不僅僅可以減少模型參數，讓訓練更快速，還可以讓模型專注在交通號誌圖示的輪廓差異，而非注意可能隨著光線而有顯著差異的RGB色階。

A computer screen shot of a program code

Description automatically generated

Fig. 5. 將RGB images轉成grayscale images的程式碼

A collage of different road signs

Description automatically generated

Fig. 6. Traffic sign images轉成灰階的效果

* Data Augmentation

藉由觀察Fig. 4.的資料集各class總數可以發現不同class之間的數量差異相當顯著，而這將會使訓練的效果因為資料的不平均性而不如預期，為了使資料筆數較少的class增加一些數據，我採用了常見的data augmentation，藉由平移、旋轉及縮放…等技術，在基於原先的資料下，產生更多筆訓練資料，達到各class的資料數平均的目標。

其中，我使用Affine Transformation、Perspective Transformation及Rotation共3種data augmentation的技術增加train dataset資料數，確保每個class都至少有3000筆資料。而3種data augmentation技術的code implementation可依序參考Fig. 7, Fig. 8及Fig. 9.。另外我也將data augmentation產生的圖片和原先的圖片顯示出來做比較，可參考Fig. 10.發現影像有輕微更動，但卻不失去本身原有的特徵。

A computer code with text

Description automatically generated with medium confidence

Fig. 7. Affine Transformation實作程式碼

A computer code on a black background

Description automatically generated

Fig. 8. Perspective Transformation實作程式碼

A computer code on a black background

Description automatically generated

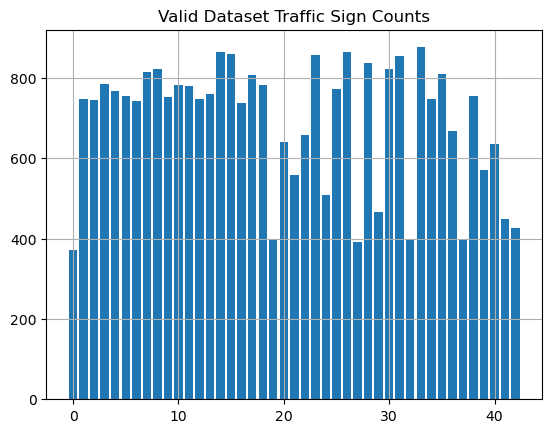
Fig. 9. Rotation實作程式碼

A close-up of a clock

Description automatically generated

Fig. 10. 右圖為原本資料集的圖片，而左側則為data augmentation產出的圖片

針對原有的資料做完data augmentation之後，我再次把dataset的各class資料數以直線圖呈現，更新過的training、validation dataset的直線圖依序為Fig. 11. (a)、(b)。我們可以發現training及validation的資料數相較於Fig. 4.因為使用data augmentation技術，各class的資料筆數都顯著提升，且class之間的數量平均。

A graph of blue lines

Description automatically generated

Fig. 11. (a)data augmentation後的training dataset各class資料筆數

(b) data augmentation後的validation dataset各class資料筆數

* Normalize Dataset

對原先的資料集images做完grayscale轉換和data augmentation之後，我又把資料做Normalization正規化的處理，將所有pixel value按比例轉換至[-1, 1]區間，詳細程式碼如Fig. 12.所示。

A computer screen with text and numbers

Description automatically generated

Fig. 12. 對資料集圖片做Normalization正規化

1. Train the LeNet5 Model

我先依據Yann LeCun所提出的LeNet5架構建構出一個模型，詳細的LeNet5參數及架構如Fig. 13.所示，並開始訓練這個LeNet5 model。我把batch size設為128，epochs設為200次，為了防止不必要的多餘訓練造成over-fitting的反效果，也有另外設定Early-Stopping的限制，詳細的訓練model的程式碼(包括Early-Stopping及optimizers的設置)可參考Fig. 14.。至於不同的optimizer一定也會對model的訓練效果有不同的影響，為了找出較適合的optimizer，我將嘗試RMSProp、Adam、Adagrad及SGD共4種optimizer，最後的訓練結果將呈現於Table 1.。

A screenshot of a computer program

Description automatically generated

Fig. 13. LeNet5詳細架構

A computer code on a black background

Description automatically generated

Fig. 14. Model training程式碼(包括Early-Stopping及optimizers的參數設定)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Different Optimizers’ Effect on GTSRB Dataset** | | | | |
| Optimizer Name | ***Adam*** | ***RMSProp*** | ***SGD*** | ***Adagrad*** |
| Training Epochs | 17 | 12 | 24 | 112 |
| Training Accuracy | 0.97 | 0.96 | 0.97 | 0.95 |
| Validation Accuracy | 0.96 | 0.96 | 0.97 | 0.95 |
| Prediction Accuracy | 0.93 | 0.93 | 0.93 | 0.91 |
| **Note: Adagrad的訓練所需epochs最多且accuracy最低，因此之後將排除該optimizer** | | | | |

Table 1. 考慮不同optimizers對於GTSRB辨識的表現差異

觀察Table 1.的結果可以發現Adagrad的training epochs遠比其他三個optimizers高，代表訓練所需花的時間相對多，accuracy也不如其他三者高，因此可以將其從最終的optimizer選擇名單中汰除。至於Adam、RMSProp及SGD的accuracy大致上沒有太大的差別，因此我接下來將再想其他方法嘗試是否能提高準確率，並由這三個optimizers中試圖找出最適合此次project的LeNet5 model。

在參考網路上的文章及詢問ChatGPT的意見後得到了可以嘗試learning rate scheduler的概念，learning rate會依據epochs或是accuracy的變化動態更新learning rate的值，於是我試著增加名為ReduceLROnPlateau的callback，但效果並沒有像預期的一樣變好，training和validation的accuracy的確有增加，但是testing accuracy依舊卡在0.93，無法再提高，因此這個方法只能作罷。

1. Accuracy Curves, Confusion Matrix and Classification Report
2. Test LeNet5 model on New Images
3. **問題討論**
   1. The Necessity of Data-Preprocessing

**四、結論**

**五、參考資料**

1. [https://github.com/Goddard/udacity-traffic-sign-classifier/](https://github.com/Goddard/udacity-traffic-sign-classifier/tree/master)
2. [data-augmentation-explanation](https://medium.com/mlait/affine-transformation-image-processing-in-tensorflow-part-1-df96256928a)
3. [perspective-transformation-explanation](https://rumn.medium.com/yolo-data-augmentation-explained-turbocharge-your-object-detection-model-94c33278303a)
4. [learning-rate-scheduler-idea](https://ithelp.ithome.com.tw/articles/10298468)