Multi-Class Single Image Classification

German and Persian Traffic Signs

By Sudharsanam Shyamsundar and Ziad Attia

Introduction:

The fundamental task in computer vision is to do the image classification apart from Object detection and the segmentation. In this project, we have built the neural network model using CNN and VGG16 model to classify the German and Persian Traffic signs (separately and combined).

Datasets:

For the task, we chose two datasets from Kaggle.

GTSRB - German Traffic Sign Recognition
https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign

Persian Traffic Sign Dataset

https://www.kaggle.com/datasets/saraparsaseresht/persian-traffic-sign-dataset-ptsd

The German dataset has train set of 14405 images and a test set of 2421 images across 43 classes. The Persian dataset has test set of 39209 images and a test set of 12631 images across 43 classes.

When we trained the models, we chose 20% of the training set to be used as a validation set.

Previous Solutions:

When it comes to the German traffic sign dataset, Many solutions used CNN models, and the vgg16 and vgg19. Beside using the vgg19 model, our code also uses Resnet 50 model.

Furthermore, the Persian Dataset is a relatively new dataset uploaded in the Kaggle, so it has very few solutions. One of the solutions trains the model with German traffic sign dataset and used this as base model to do transfer learning with the Persian dataset.

https://github.com/FarhadNasri/Persian-Traffic-Signs-Recognition

We, on the other hand, combined the two datasets across 21 classes and trained our models on the combined dataset which wasn't done before.

Proposed Methods:

First, we downloaded the Persian and the German datasets using a python script. We then combined them to create a third dataset with 21 classes which we will refer to as Combined dataset.

Then, we created our two models, CNN and VGG16. For VGG16, we used models available on Keras and added additional dense layers to the output for training with these datasets. However, we created the CNN from scratch.

Finally, we ran each of the three training and validation sets on our models (a total of 6 runs) and tested our trained model on the test datasets.

Evaluation Methods:

The main criterion for evaluating our models is accuracy. We were aiming to maximize the validation accuracy and loss during model training, and the test accuracy of the trained model. We also explored the precision, recall, f1-score, and support of our model. We also used a confusion matrix to visualize our results.

Results and Discussion:

CNN:

For the German dataset, the validation accuracy and loss were 0.9964 and 0.0103, respectively. The test accuracy was 0.97.

For the Persian dataset, the validation accuracy and loss were 0.9889 and 0.0408, respectively. The test accuracy was 0.98.

For the Combined dataset, the validation accuracy and loss were 0.9895 and 0.0419, respectively. The test accuracy was 0.98.

VGG16:

For the German dataset, the validation accuracy and loss were 0.8301 and 0.5456, respectively. The test accuracy was 0.55.

For the Persian dataset, the validation accuracy and loss were 0.8903 and 0.3799, respectively. The test accuracy was 0.71.

For the Combined dataset, the validation accuracy and loss were 0.8634 and 0.4222, respectively. The test accuracy was 0.61.

We can clearly see that the CNN that was created was much accurate than the VGG16 that we imported from Keras for the three datasets. The German dataset had the worst results using the VGG16. All the predictions from the CNN are extremely accurate across the three datasets.