Report: Image Classification for Self Driving Cars

Name: Muhammad Rayyan Khan
Intern ID: AIMLINT-240324-XT0017+AI|ML Development+ ImageClassification

---

Abstract

The project aimed to develop a multi-label image classification model to enhance the capabilities of autonomous vehicles. By leveraging the Berkeley DeepDrive (BDD100K) dataset, a convolutional neural network (CNN) was built to predict multiple labels per image, focusing on various objects such as cars, traffic lights, and pedestrians. The goal was to improve the accuracy and reliability of object detection in autonomous driving systems, which can potentially reduce car accidents caused by human error.

Data Collection

- **Dataset:** Berkeley DeepDrive (BDD100K)
- **Images Used:** 70,000 training images and 10,000 validation images
- **Labels:** 12 unique labels including 'lane', 'drivable area', 'truck', 'motor', 'person', 'rider', 'bike', 'train', 'car', 'traffic light', 'bus', 'traffic sign'
- **Diversity:** Geographic, environmental, and weather diversity in the dataset to ensure robustness in various conditions

---

Data Preprocessing

1. **Label Extraction:**
    - Extracted 'category' labels from the dataset, focusing on the 12 unique labels relevant to the project.
2. **Image Processing:**
    - Images were rescaled and resized using the `ImageDataGenerator` function to standardize inputs for the CNN.
    - Images were set to a size of 86 x 86 pixels, balancing computational efficiency and model performance.

Model Architecture

1. **Basic CNN Model:**
   - **Layers:** Input layer, hidden layers with Conv2D-Pooling pattern, and an output layer.
   - **Activation Functions:** 'ReLU' for hidden layers and 'sigmoid' for the output layer.
   - **Optimizer:** 'Adam' optimizer used for training the model.
   - **Loss Function:** Binary cross-entropy used for multi-label classification.
2. **Tuned CNN Model:**
   - **Pattern:** Conv2D-Conv2D-Pooling pattern.
   - **Hyperparameters:** Padding='valid', dropout=0.5, optimizer='Adam'.
   - **Early Stopping:** Implemented to prevent overfitting.
3. **Pretrained Model (InceptionResNetV2):**
   - **Transfer Learning:** Used InceptionResNetV2 as the base model with input and output layers adjusted for the specific task.
   - **Training Time:** Longer due to the complexity of the model.

Training Process

- **Epochs:** The final tuned CNN model trained up to 9 epochs to avoid overfitting.
- **Metrics:**
  - **Hamming Loss:** Primary evaluation metric. Lower values indicate better performance. The final model achieved a Hamming loss of 11% (89% accuracy).
  - **Validation Loss and Accuracy:** Monitored to assess model performance and detect overfitting.
  - **Test Accuracy:** Varied significantly across different runs, highlighting the importance of consistent evaluation.

Evaluation Metrics

- **Hamming Loss:** Measures incorrect predictions divided by the total number of predictions. The tuned CNN model achieved the best score of approximately 11%.
- **Accuracy:** Though not the primary metric, it provided a supplementary measure of performance, with the final CNN model showing around 10% on the test validation set, while the pretrained model showed around 93%.

Challenges Faced

1. **Overfitting:**
   - Initial models showed signs of overfitting, addressed by using early stopping and dropout layers.
2. **Label Frequency Imbalance:**
   - Labels like 'train', 'motor', and 'bus' appeared less frequently, making accurate prediction challenging.
3. **Computational Limitations:**
   - Image size was limited to 86 x 86 pixels to prevent system crashes, potentially affecting model performance.

Implications and Future Directions

Implications:

- **Driver Assistance:** The model can be used as a driver assistance system to alert drivers of surrounding objects, potentially reducing accident rates.
- **Foundation for Autonomous Vehicles:** Provides a baseline model for companies developing self-driving cars.

Future Directions:

1. **Explore Different Model Architectures:**
   - Test other advanced architectures like EfficientNet or custom-designed neural networks.
2. **Incorporate Real-Time Constraints:**
   - Optimize models for real-time inference to enhance practical usability in autonomous driving systems.
3. **Address Label Imbalance:**
   - Use techniques like oversampling, undersampling, or synthetic data generation to balance the label frequencies.
4. **Increase Image Resolution:**
   - Experiment with higher resolution images if computational resources allow, to potentially improve model accuracy.

Conclusion

This project demonstrated the feasibility of creating a multi-label image classification model for autonomous vehicles using CNNs. The findings underscore the importance of robust model design and diverse datasets in developing reliable autonomous driving systems. Future improvements could further enhance model performance, paving the way for safer and more efficient autonomous vehicles.