



Emergency Vs Non-Emergency Vehicle Classification

Abhidharini R

22-PDS-004

Abstract

The project "Emergency vs Non-Emergency Vehicle Classification using Deep Learning Models" addresses a critical aspect of traffic management and emergency response systems by leveraging deep learning techniques for vehicle classification. The objective is to develop an automated system that accurately distinguishes between emergency and non-emergency vehicles based on image data. Deep learning models, including Convolutional Neural Network, DenseNet, ResNet, Xception, and VGG16, are employed to achieve this classification task. The project focuses on creating an efficient and accurate model that can be integrated into traffic monitoring systems to aid in real-time decision-making. Key aspects of the project include data collection and preprocessing to curate a robust dataset, model selection based on performance evaluation, training, validation, and fine-tuning for optimal results. The project aims to provide a scalable solution that can be seamlessly integrated into existing traffic management systems, enhancing decision-making and ultimately contributing to safer and more organized urban landscapes.

Keywords: Deep Learning, Convolutional Neural Networks, Emergency Vehicles, Non-Emergency Vehicles, Image Classification, Transfer Learning, Model Evaluation, Traffic Management.

Introduction

The rapid growth of urban areas has necessitated efficient traffic management and emergency response systems. One crucial component of these systems is the ability to differentiate between emergency vehicles (e.g., ambulances, police cars, fire trucks) and regular vehicles to optimize

traffic flow and prioritize emergency responses. In this project, we propose a deep learning-based approach to classify vehicles into two categories: emergency and non-emergency.

Methodology

The project "Emergency vs Non-Emergency Vehicle Classification using Deep Learning Models" utilizes various methods to achieve accurate vehicle classification.

1. Data Collection and Preparation

- Acquired a dataset consisting of images of emergency and non-emergency vehicles.
- Preprocessed the images, including resizing, normalization, and data augmentation.

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    shear_range=0.1,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    vertical_flip = True,  
    validation_split = 0.2)
```

2. Model Selection


- Choose deep learning models suitable for image classification: DenseNet, ResNet, Xception, VGG16 and CNN.

1. DenseNet

The pre-trained DenseNet121 model is loaded without its top layers (fully connected layers). The input shape for the model is set to (128, 128, 3).

All layers in the DenseNet121 base model are set to non-trainable to preserve the pre-trained weights and avoid overfitting. A global average pooling layer is added to reduce the spatial dimensions of the previous layer. A dense layer with 128 units and ReLU activation is added. A dropout layer with a dropout rate of 50% is added to reduce overfitting. The final dense layer with 1 unit and a sigmoid activation for binary classification. A new model ('model') is created using the DenseNet121 base model's input and the custom classification layers as the output. The model is compiled, using binary cross-entropy loss (suitable for binary classification) and accuracy as the evaluation metric. The model summary is displayed.

2. Xception



The pre-trained Xception model is loaded without its top layers (fully connected layers). The input shape for the model is set to (128, 128, 3). All layers in the Xception base model are set to non-trainable to preserve the pre-trained weights and avoid overfitting. Global Average Pooling Layer: A global average pooling layer is added to reduce the spatial dimensions of the previous layer. A dense layer with 128 units and ReLU activation is added. The final dense layer with 1 unit and a sigmoid activation for binary classification. A new model (`model`) is created using the Xception base model's input and the custom classification layers as the output. The model is compiled, using binary cross-entropy loss (suitable for binary classification) and accuracy as the evaluation metric. The model summary is displayed.

3. ResNet

Pre-trained ResNet models were fine-tuned, and training was performed on a dataset comprising emergency and non-emergency vehicle images. The model demonstrated efficient learning and achieved high classification accuracy, making it a valuable component in the automatic classification of emergency and non-emergency vehicles, contributing to optimized traffic management and emergency response systems.

4. VGG16

The pre-trained VGG16 model is loaded without its top layers (fully connected layers), retaining the convolutional base. The input shape for the model is set to (128, 128, 3). All layers in the base VGG16 model are set to non-trainable to preserve the pre-trained weights and avoid overfitting. A global average pooling layer is added to reduce the spatial dimensions of the previous layer. Dropout with a rate of 0.5 is applied to prevent overfitting. A dense layer with 32 units and ReLU activation is added. Another dropout layer with a rate of 0.5 is added for regularization. The final dense layer with 1 unit and a sigmoid activation for binary classification. A new model (`model_vgg`) is created using the VGG16 base model's input and the custom classification layers as the output. The model is compiled using the Adam optimizer, binary cross-entropy loss (suitable for binary classification), and accuracy as the evaluation metric.

5. CNN

The model starts with a convolutional layer (Conv2D) with 32 filters and a 3x3 kernel, creating a feature map from the input image. After each convolutional layer, there's a max-pooling layer (MaxPooling2D) to reduce the spatial dimensions and retain important features. Another convolutional layer follows, now with 64 filters and a 3x3 kernel to extract higher-level

features. The output from the last convolutional layer is flattened into a 1D array to prepare for the fully connected layers. The flattened array is connected to a dense layer with 128 units, using ReLU activation to introduce non-linearity and learn complex patterns. The final dense layer with a single unit and sigmoid activation is for binary classification, predicting the probability of the input belonging to the positive class (emergency) or negative class (non-emergency). The model has a total of 7,392,449. During training, the model learns to optimize these parameters using backpropagation and an appropriate optimizer (e.g., Adam) to minimize the chosen loss function (e.g., binary cross-entropy). The model's performance is evaluated on a separate test dataset, measuring metrics like accuracy, precision, recall, and F1-score to assess its effectiveness.

→ Explored pre-trained models and fine-tuning to leverage existing knowledge.

3. Model Training

- Divided the dataset into training and validation sets.
- Trained the selected models on the training set and fine-tuned them for optimal performance.

5. Model Evaluation

- Evaluated the models on a separate test dataset to measure their accuracy, precision, recall, and F1-score.

Results and Discussion

In the context of the "Emergency vs. Non-Emergency Vehicle Image Classification" project, the following results were obtained from training and evaluating various deep learning models:

Model	Accuracy
Xception	87%
DenseNet	89%
ResNet	86%
CNN	79%
VGG16	79%

The performance of the models varies, with DenseNet exhibiting the highest accuracy at 89%. DenseNet is known for its densely connected layers, allowing for efficient feature reuse and better performance in deep networks. On the other hand, the CNN and VGG16 models achieved an accuracy of 79%, suggesting that the models might not be complex enough to capture intricate features from the images compared to the more sophisticated architectures. Xception, a deep convolutional neural network architecture, performed well with an accuracy of 87%. It uses depth-wise separable convolutions, making it computationally efficient and effective in learning complex features. ResNet achieved an accuracy of 86%, demonstrating its ability to combat the vanishing gradient problem using residual connections, which helps in training very deep networks. It's important to note that model performance is affected by various factors, including the complexity of the architecture, the amount and quality of training data, data preprocessing, and hyperparameters. Further fine-tuning and experimentation with hyperparameters may enhance the performance of the models.

Conclusion

DenseNet emerged as the top-performing model for this specific image classification task, achieving the highest accuracy among the models evaluated. However, continued experimentation and possibly exploring more advanced architectures may further improve the classification accuracy.

