Final Project for Vehicle Classification

Ori Daniel (318799145), Gal Levi (207685678)

March 20, 2025

TRANSFER LEARNING

1 Introduction

This report presents a deep learning-based approach for vehicle classification using Transfer Learning. The project utilizes a pre-trained ResNet50 model, fine-tuned for classification of 196 vehicle classes. We compare three configurations:

- Base Model: Pre-trained ResNet50 with minor modifications.
- Augmented Model: Data augmentation applied for improved robustness.
- Fine-Tuned Model: Additional layers trained with unfreezing of the final layers.

The dataset was split into 50% training and 50% testing to assess model generalization. ResNet50 was chosen due to its strong feature extraction capabilities, leveraging a deep convolutional network trained on ImageNet.

2 Experiment Setup

2.1 Description of the Experiment

Our experiment focused on utilizing transfer learning to classify images of vehicles efficiently. We leveraged the pre-trained ResNet50 model and evaluated different configurations to determine the optimal approach.

2.2 Hyperparameters

Base Model:

- Optimizer: Adam
- Learning Rate: 0.001
- Batch Size: 32
- Loss Function: Categorical Crossentropy
- Early Stopping: Enabled (patience: 10)

Augmented Model:

- Same hyperparameters as the base model
- Data augmentation applied (rotation, flipping, brightness adjustments, shear and zoom)

Fine-Tuned Model:

- Optimizer: Adam
- Learning Rate: 0.0005 (lower for stability)

- Batch Size: 16
- Loss Function: Categorical Crossentropy
- Early Stopping: Enabled (patience: 10)
- Additional layer unfreezing (last 20 layers unfrozen for fine-tuning)

2.3 Data Augmentation

For the augmented and fine-tuned models, we applied:

- Random rotations (15-20 degrees)
- Horizontal flipping
- Brightness adjustments
- Contrast variations
- Shear and zoom transformations

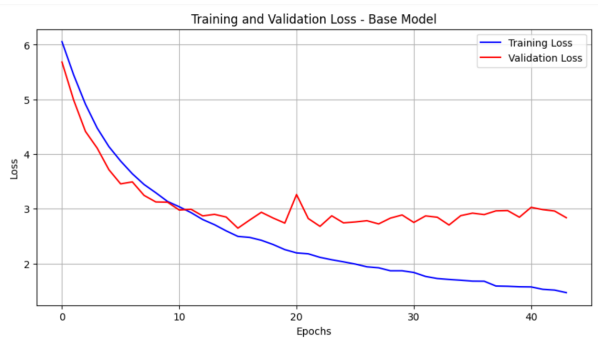
2.4 Architecture

Base Model: Standard ResNet50 with a fully connected head:

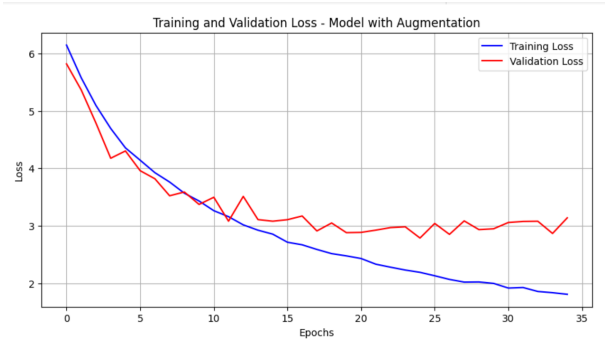
- Global Average Pooling Layer
- Fully Connected Layer (512 neurons, ReLU activation)
- Dropout (0.5)
- Fully Connected Layer (256 neurons, ReLU activation)
- Dropout (0.3)
- Output Layer (196 neurons, softmax activation)

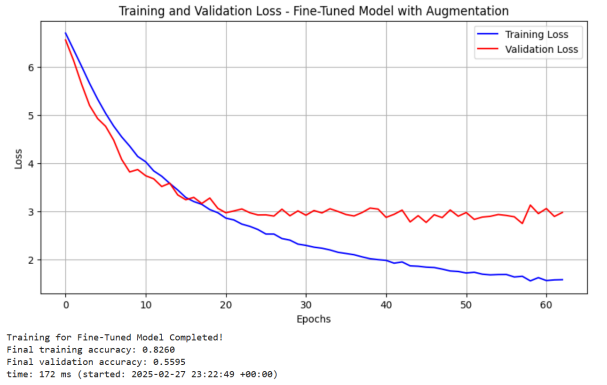
Augmented Model: Same architecture as the base model.
Fine-Tuned Model: More layers unfrozen and additional depth:

- Global Average Pooling Layer
- Fully Connected Layer (1024 neurons, ReLU activation)
- Dropout (0.5)
- Fully Connected Layer (512 neurons, ReLU activation)
- Dropout (0.4)
- Fully Connected Layer (256 neurons, ReLU activation)
- Dropout (0.2)
- Output Layer (196 neurons, softmax activation)




```
Training with Data Augmentation complete!
Final training accuracy: 0.8141
Final validation accuracy: 0.5519
time: 162 ms (started: 2025-02-27 20:28:46 +00:00)
```





3 Performance Metrics

To evaluate model performance, we used the following metrics:

- Top-1 Accuracy
- Top-5 Accuracy

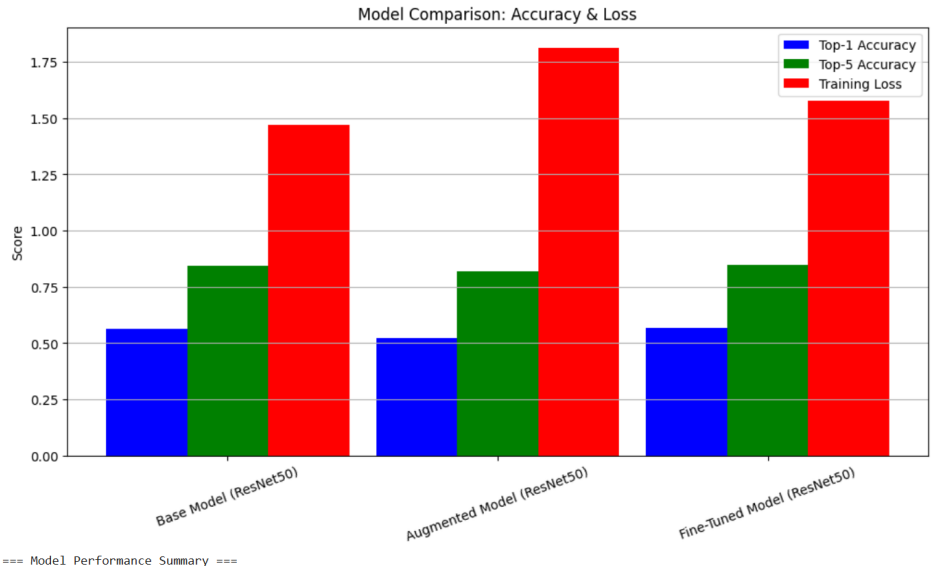
4 Results

4.1 Training and Validation Loss

Training and validation loss graphs for each model:

4.2 Performance Comparison

Model	Top-1 Accuracy	Top-5 Accuracy
Base Model	56.36%	84.33%
Augmented Model	52.12%	82.07%
Fine-Tuned Model	56.85%	84.87%




```
=== Model Performance Summary ===
```

5 Additional points about our understanding

Deep learning models consist of multiple layers, each serving a specific role in feature extraction and classification. Below is an overview of key components used in our project: Convolutional layers are responsible for detecting patterns and spatial hierarchies in images. In ResNet50, these layers help extract edges, textures, and object parts, making them crucial for feature extraction in transfer learning.

Fully connected (FC) layers at the end of the network map extracted features to class probabilities. In our case, we added additional FC layers to adapt ResNet50's feature maps to the vehicle classification task. Batch normalization (BatchNorm) stabilizes training by normalizing activations across mini-batches. It helps reduce internal covariate shift and speeds up convergence. Pretrained weights in transfer learning retain prior knowledge from training on large datasets (e.g., ImageNet). By reusing these weights, our model leverages previously learned representations, allowing faster convergence and better feature extraction. Our problem involves classifying 196 different vehicle types, making it a multi-class classification challenge. We used categorical crossentropy loss, a standard choice for such tasks, and evaluated performance using Top-1 and Top-5 accuracy. The learning rate affects how quickly the model updates weights. A high learning rate may lead to instability, while a low rate may slow convergence. We used a lower learning rate (0.0005) for fine-tuning to ensure stable adaptation of deeper layers.

5.1 Regularization Techniques

To prevent overfitting, we employed:

- **Dropout:** Randomly deactivates neurons to improve generalization .
- **L2 Regularization:** Penalizes large weight values to encourage simpler models.
- **Early Stopping:** Stops training when validation loss stops improving.

.

6 Conclusion

The fine-tuned model achieved the highest accuracy among all configurations. However, due to computational limitations, we were unable to further optimize its performance. If additional computing power were available, unfreezing more layers in ResNet50 could potentially yield even higher accuracy. One of the key challenges encountered was avoiding overfitting. This issue became apparent as the validation loss plateaued while the training loss continued to decrease, indicating that the model was memorizing the training data rather than generalizing effectively. To mitigate this, we implemented **early stopping**, which prevented excessive training once the validation loss ceased to improve.

Image Retrieval

We used the fine-tuned model from Notebook 1 to extract feature embeddings from images. Instead of training a new model, we leveraged the learned representations from the fine-tuned ResNet50 model to generate embeddings for vehicle images.

For each input image, we extracted its feature representation using the model's final convolutional layer, followed by a global average pooling layer. These feature vectors were then compared using a K-Nearest Neighbors (KNN) classifier to retrieve the most similar images based on different values of k.

6.1 KNN-Based Image Retrieval

- We extracted embeddings for images using the fine-tuned model.
- When a query image was given, we computed its embedding and found the closest images using KNN.
- The distance function used in KNN was Euclidean distance.
- We evaluated different values of k to find the optimal setting [1,5,10].

This approach enables efficient image retrieval without the need to retrain a model. Instead of performing classification, we simply find the closest matches in the embedding space.

6.2 Performance Comparison

To evaluate the effectiveness of our retrieval system, we compared Top-1 and Top-5 accuracy across different values of k. The results are summarized in the following table:

k	Top-1 Accuracy	Top-5 Accuracy
1	58 %	59 %
5	59 %	79 %
10	60 %	81 %

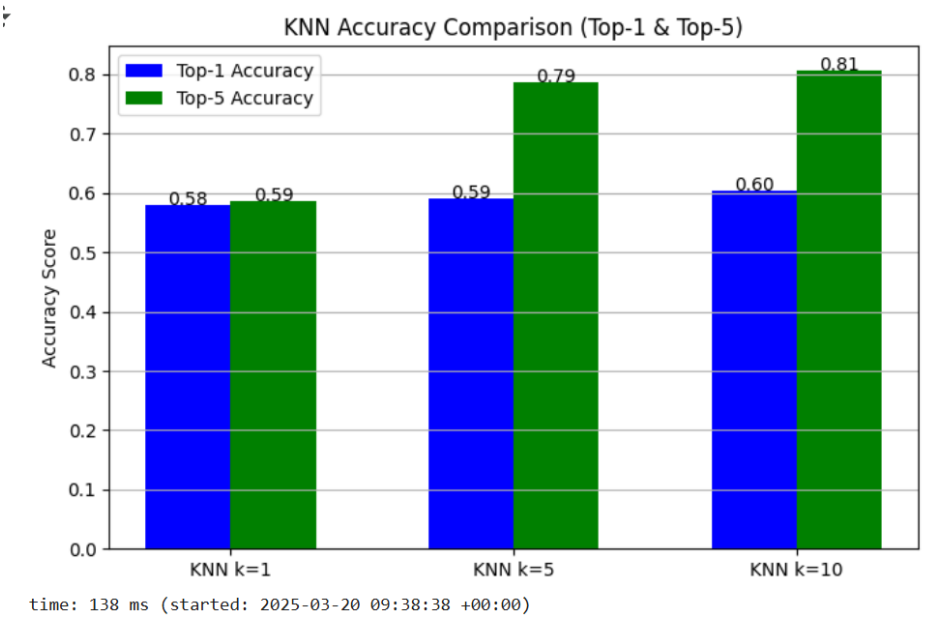


Figure 1: Comparison of Top-1 and Top-5 accuracy across different k values in KNN.

7 Conclusion

The image retrieval approach using pre-trained embeddings and KNN classification proved effective. By leveraging the fine-tuned ResNet50 model, we avoided the need for additional training while still achieving meaningful retrieval results. Furthermore, we observed that as we increased the value of k, accuracy improved. This is because a higher k allows the model to consider more nearest neighbors, leading to a more robust decision-making process that accounts for a wider context of similar images.

End-to-End CNN

8 Introduction

This report presents a deep learning-based approach for vehicle classification, distinguishing between 196 car models using CNNs. The project explores three models:

- **Custom CNN:** A sequential model with convolutional layers, batch normalization, and dropout.
- **Augmented CNN:** A variant with extensive data augmentation for improved generalization.
- **ResNet-Inspired Model:** A deeper architecture leveraging identity and projection blocks.

Using the Stanford Cars dataset, we trained models on an 80%-20% train-validation split, optimizing categorical cross-entropy loss and accuracy. Performance was evaluated using Top-1 and Top-5 accuracy to determine the most effective classification approach.

9 Experiment Setup

9.1 Description of the Experiment

Our experiment aimed to classify 196 vehicle models using CNN-based architectures. We trained and evaluated three models to compare their accuracy and generalization capabilities.

9.2 Hyperparameters

Custom CNN:

- Optimizer: Adam, Learning Rate: 0.001
- Batch Size: 32, Loss Function: Categorical Crossentropy
- Early Stopping: Enabled (patience: 10)

Augmented CNN:

- Same as Custom CNN with extensive data augmentation (rotation, flipping, brightness, contrast)

ResNet-Inspired Model:

- Optimizer: Adam, Learning Rate: 0.0001
- Deeper architecture with skip connections

9.3 Data Augmentation

Applied in the Augmented CNN and ResNet models:

- Rotations (15°), flips, brightness, contrast adjustments

9.4 Architecture

Custom CNN:

- Four convolutional layers, batch normalization
- Fully connected layer (1024 neurons), dropout (0.5)
- Output layer (196 neurons, softmax)

Augmented CNN:

- Five convolutional layers, higher dropout (0.25–0.5)
- Fully connected layers (512, 256 neurons)

ResNet-Inspired Model:

- Identity and projection blocks for deeper feature extraction
- Global Average Pooling, fully connected layers

10 Performance Metrics

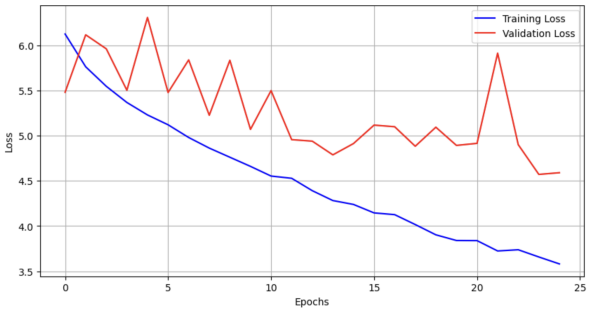
We evaluated models using:

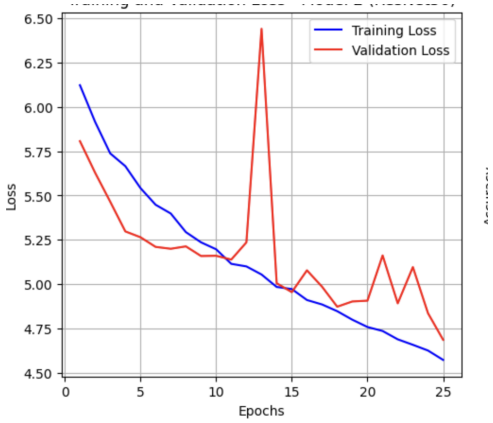
- Top-1 Accuracy
- Top-5 Accuracy

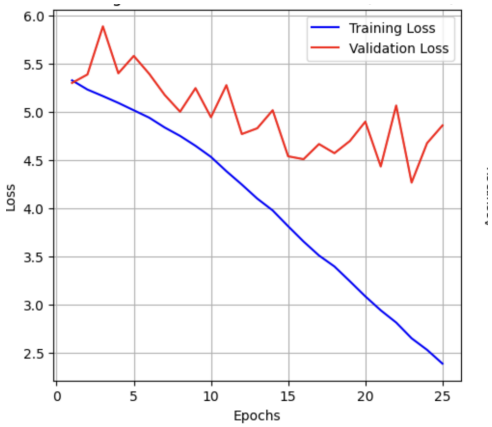
11 Results

11.1 Training and Validation Loss

Training and validation loss graphs for each model:

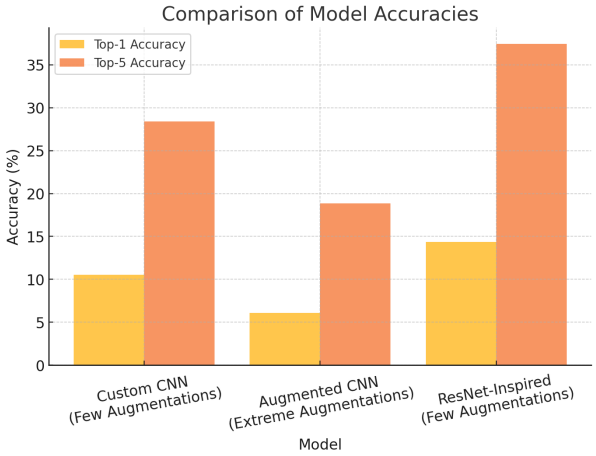






11.2 Performance Comparison

Model	Top-1 Accuracy	Top-5 Accuracy
Custom CNN(Few augmentations)	10.55%	28.39%
Augmented CNN(Extreme augmentations)	6.09%	18.85%
ResNet-Inspired Model(Few augmentations)	14.36%	37.45%



12 Additional Points About Our Understanding

Deep learning allows architectural flexibility, but performance gains are not always guaranteed. Our experiments showed that modifying CNNs and applying augmentations had limited impact, whereas leveraging a well-established model like ResNet yielded better results.

Convolutional layers are key to pattern recognition, but our custom CNNs struggled with fine-grained distinctions. ResNet's pre-trained filters provided richer feature extraction, demonstrating the power of transfer learning.

Fully connected layers map extracted features to class probabilities, but deeper networks alone did not always improve accuracy. Instead, structured fine-tuning with pre-trained models proved more effective.

Batch normalization improved convergence but could not compensate for feature extraction limitations in shallow architectures. The ResNet-inspired model outperformed others due to its residual connections and deeper representations, highlighting the advantage of using proven architectures.

12.1 The Impact of Transfer Learning

One key takeaway is that **adapting a proven architecture is often more effective than creating a new one from scratch**. While data augmentation can enhance generalization, extreme transformations

did not necessarily improve classification accuracy. Instead, transfer learning with proper fine-tuning yielded the best results, leveraging prior knowledge from large-scale datasets.

12.2 Regularization Techniques

To prevent overfitting, we applied:

- **Dropout:** Randomly deactivated neurons to improve generalization.
- **L2 Regularization:** Penalized large weight values to encourage simpler models.
- **Early Stopping:** Halted training when validation loss plateaued.

13 Conclusion

This study explored deep learning approaches for fine-grained vehicle classification. While data augmentation and custom CNNs offered some improvements, the ResNet-inspired model performed best, highlighting the effectiveness of transfer learning. Adapting proven architectures proved more impactful than excessive modifications. Future work could refine fine-tuning strategies or explore ensemble methods for further improvements.

14 References

- ResNet50 Model: `https://keras.io/api/applications/resnet/`
- Keras : `https://keras.io/`

15 Code

- First notebook - Transfer Learning : `https://colab.research.google.com/drive/1xP2nK4cThH9g_Abxk-gT8PCvqwMTil4R`
- Second Notebook - image retrival : `https://colab.research.google.com/drive/1k8lxHfSiVBWtuJJoJubd78X4gP usp=sharing`
- Third Notebook - End to End CNN : `https://colab.research.google.com/drive/1HbdoOmvOYWhh7T9MOpOfrc2c usp=sharing`