Vehicle Make and Model Recognition Using Fine-Tuned AlexNet

Victor Onwosi, Jager Cooper, and Esther Success Ukpe

Abstract—Vehicle Make and Model Recognition (VMMR) is a challenging fine-grained image classification problem with real-world applications in traffic surveillance, intelligent transportation systems, vehicle analytics, and smart city infrastructure. The task involves distinguishing between hundreds of visually similar car categories, often differing only in subtle details such as trim, year, or body shape. In this project, we explore the effectiveness of transfer learning using AlexNet, a historically significant convolutional neural network architecture, as a backbone for VMMR on the Stanford Cars dataset. The dataset contains 16,185 images across 196 fine-grained vehicle classes. We build our training pipeline using the FastAI library in Google Colab, incorporating modern training techniques such as ImageNet-based normalization, dynamic data augmentation, automatic learning rate discovery, and model checkpointing with early stopping. We fine-tune a pretrained AlexNet model and achieve a validation accuracy of 68.7% — a relatively strong result given the model's simplicity and dataset limitation. Beyond accuracy, we conduct extensive evaluations including class-wise accuracy, confusion matrix analysis, top-loss visualizations, and classification reports. These insights highlight the model's strength in recognizing visually distinctive vehicles, while revealing its challenges with overlapping or underrepresented classes. Our results demonstrate that, when equipped with a well-structured pipeline and modern tooling, lightweight CNNs like AlexNet remain viable for fine-grained classification problems in constrained environments.

Index Terms—Vehicle Recognition, Transfer Learning, AlexNet, Image Classification, Fine-grained Recognition, Deep Learning, FastAI, Computer Vision.

1 Introduction

Vehicle Make and Model Recognition (VMMR) is a fine-grained image classification task that plays a critical role in intelligent transportation systems, vehicle surveillance, fleet management, and urban mobility analytics [5], [6]. The objective is to identify a vehicle's make and model from an image, a task made challenging by high visual similarity between classes, varying lighting, occlusion, and viewpoint changes [2], [7].

Early VMMR methods relied on handcrafted features such as HOG, SIFT, or LBP descriptors, but these approaches often failed to generalize under real-world conditions. With the rise of deep learning, convolutional neural networks (CNNs) have largely replaced traditional feature engineering by learning robust hierarchical representations directly from pixel-level data [3], [4].

AlexNet [11], though relatively shallow by today's standards, was a landmark CNN architecture that demonstrated deep learning's power in large-scale image classification. Despite the emergence of deeper models like ResNet, AlexNet remains attractive for research due to its simplicity, speed, and effectiveness when fine-tuned with transfer learning [2], [6].

In this project, we evaluate the effectiveness of fine-tuning AlexNet for VMMR using the Stanford Cars dataset [8], which includes 16,185 images spanning 196 car classes. We used a GitHub-hosted version of the dataset with pre-separated training and validation splits to streamline the process [9].

Our training pipeline is built using the FastAI v2 library [10] in Google Colab. It incorporates image normalization with ImageNet statistics, aggressive data augmentation, learning rate discovery, early stopping, and model checkpointing. We fine-tuned the model over 50 epochs and achieved a validation accuracy of 68.7%, a strong result given the task's fine-grained nature and AlexNet's architectural limitations.

To gain deeper insight into model performance, we visualize confusion matrices (raw and normalized), generate per-class accuracy reports, and analyze top misclassified samples. These diagnostics reveal that while the model performs well on visually distinct car types (e.g., SUVs, luxury coupes), it struggles with subtle intra-brand variants, offering directions for future improvements.

2 Related Work

Vehicle Make and Model Recognition (VMMR) has evolved significantly from early handcrafted approaches to modern deep learning pipelines. Traditional techniques relied on local descriptors such as SIFT, HOG, and LBP combined with classifiers like SVMs, but struggled to generalize under challenging real-world conditions like occlusion, motion blur, and varying viewpoints [5].

The introduction of convolutional neural networks (CNNs) revolutionized this task by learning hierarchical features directly from pixel data. Several benchmark studies have shown the effectiveness of deep CNNs in improving VMMR performance. Tafazzoli and Safabakhsh [7] introduced a large and diverse dataset for robust VMMR, emphasizing the importance of fine-grained annotation and dataset balance in training high-performance models.

AlexNet, despite its relatively shallow depth, has been repeatedly used as a lightweight yet effective backbone in this domain. Khan et al. [2] fine-tuned AlexNet on a custom vehicle dataset and reported substantial improvements over traditional methods. Similarly, Lediur [3] demonstrated

TABLE 1
Summary of Prior VMMR Approaches

Study	Architecture	Reported Accuracy
Khan et al. (2024) [2]	Fine-tuned AlexNet	84.4% (on custom dataset)
Espinosa et al. (2017) [4]	AlexNet vs Faster R-CNN	74% (detection accuracy)
Lediur (2015) [3]	AlexNet	66.9% (on CS231n VMMR split)
Aziz et al. (2021) [6]	ResNet + SVM Ensemble	89.6% (on FGVC-VMMR)
Najeeb et al. (2021) [5]	Custom CNN + augmentation	82.5% (urban traffic)
This Project	Fine-tuned AlexNet (FastAI)	**68.7%** (Stanford Cars)

that even without significant architectural modifications, AlexNet was capable of distinguishing visually similar vehicle models with competitive accuracy when trained on a clean dataset. Espinosa et al. [4] compared AlexNet and Faster R-CNN for general vehicle detection and highlighted AlexNet's efficiency on constrained hardware.

Recent works have explored various enhancements to improve CNN-based VMMR. These include combining deep transfer learning with ensemble classifiers such as SVMs [6], or optimizing data augmentation and training routines to reduce overfitting and improve class-wise balance. Most of these methods report diminishing returns when moving to deeper architectures, especially when computational constraints or dataset noise are involved.

Publicly available datasets such as Stanford Cars [1], [8] are commonly used for benchmarking. However, the standard dataset splits require preprocessing. For this project, we leveraged a community-organized version of the dataset with predefined training and validation folders [9], enabling direct compatibility with FastAI's image loading utilities [10].

Our work builds upon these findings by revisiting AlexNet as a competitive baseline, using a structured transfer learning approach. We integrate recent tooling—such as FastAI's LR finder, data block API, and augmented transforms—while keeping the model architecture fixed. Unlike past experiments that focused on dataset-specific models or heavy ensembling, we aim to demonstrate how far a cleanly-tuned AlexNet can go under modern training conditions.

3 METHODS

3.1 Overview of Initial Approach

Our initial implementation was built using raw PyTorch and torchvision on a local development machine. We fine-tuned AlexNet by freezing early layers and training only the classifier and final convolutional blocks. Despite employing basic augmentation and label smoothing, this approach achieved a maximum validation accuracy of only 36.1% before plateauing. We attribute the limited performance to insufficient learning rate scheduling, lack of dynamic early stopping, and less robust data augmentation strategies.

3.2 Final Approach: FastAI-Based Training Pipeline

To improve training stability and model generalization, we reimplemented the pipeline using the FastAI v2 library [10], which offers high-level abstractions and optimized defaults for deep learning tasks. The final method was executed in Google Colab and leveraged transfer learning with a pre-trained AlexNet [11], fine-tuned on a community-organized version of the Stanford Cars dataset [9].

3.3 Dataset

We used the Stanford Cars dataset [8], containing 16,185 images across 196 vehicle classes. Each image is labeled with its make, model, and manufacturing year. The dataset was pre-split into 8,144 training and 8,041 validation images. Images were resized to 224x224 and normalized using ImageNet statistics.

3.4 Data Augmentation

To improve model robustness, we applied FastAI's `aug_transforms()` function with the following settings: random horizontal and vertical flips, rotation (up to 5°), zoom (up to 10%), lighting noise, and warping. These were applied probabilistically during training along with normalization.

3.5 Model Architecture

The model architecture is based on AlexNet [11], a convolutional neural network with five convolutional layers followed by three fully connected layers. The final classification layer was replaced with a linear layer of size 196 to match the number of vehicle classes. We used a pretrained version of AlexNet from the torchvision model zoo as the base.

3.6 Training Strategy

We employed transfer learning via FastAI's `vision_learner()`, which automatically handles model construction, parameter splitting, and optimizer setup. Training proceeded in two phases:

- **Learning Rate Finder:** We used FastAI's built-in `lr_find()` utility to identify the optimal learning rate. The valley point of the loss curve was selected as the base learning rate.
- **Fine-tuning:** We trained the model for 50 epochs using the 1-cycle learning rate policy with a base learning rate of 4.37×10^{-3}. We added `SaveModelCallback` to store the best model based on validation accuracy and `EarlyStoppingCallback` with a patience of 8 epochs.

3.7 Evaluation Metrics and Outputs

To evaluate model performance, we recorded metrics across multiple dimensions:

- **Overall Accuracy:** Computed on the validation set using FastAI's `accuracy` metric.

- **Class-wise Accuracy:** Calculated manually per class using NumPy operations on predicted and ground-truth labels.
- **Confusion Matrices:** Both raw and normalized confusion matrices were generated using FastAI's `ClassificationInterpretation`.
- **Classification Report:** We used `sklearn.metrics.classification_report()` to compute precision, recall, and F1-scores.
- **Visualizations:** We saved plots for sample predictions, top misclassifications, training/validation loss curves, learning rate curves, and class-wise accuracy distributions.

4 Experiments

To assess the effectiveness of transfer learning using AlexNet for vehicle make and model recognition, we conducted two sets of experiments: an initial baseline using raw PyTorch on a local machine and a final, more robust experiment using FastAI in Google Colab. Both experiments used the same dataset and general preprocessing pipeline to ensure comparability.

4.1 Dataset

We used the Stanford Cars dataset [1], which consists of 16,185 images across 196 vehicle classes. For both experiments, we used a pre-organized version of the dataset from [9], containing 'train/' and 'valid/' folders. All images were resized to 224×224 pixels and normalized using ImageNet statistics.

4.2 Initial Experiment (PyTorch)

In our initial experiment, we fine-tuned a pretrained AlexNet model using PyTorch. We implemented custom logic to freeze and unfreeze layers, applied label smoothing, and used a learning rate scheduler based on validation performance. The model was trained for up to 40 epochs with early stopping after 20 epochs. Key settings included:

- **Batch size:** 64
- **Optimizer:** SGD with momentum = 0.9, weight decay = 5×10^{-4}
- **Loss:** CrossEntropyLoss with label smoothing = 0.1
- **Learning rate:** 0.001

Despite early promise (training accuracy reached 90.1%), the model suffered from severe overfitting, peaking at just **37.2% validation accuracy**. This result is shown in Figure **??**, and training stopped due to no validation improvement for 5 epochs.

4.3 Final Experiment (FastAI)

To overcome the limitations of our baseline, we transitioned to FastAI v2 [10], leveraging its modular callbacks, augmentation, and dynamic learning rate finder. We used Google Colab with the A100GPU and GPU acceleration, enabling larger batch sizes and longer training.

Key improvements included:

- Automatic learning rate finder (valley at 4.37×10^{-3})
- Data augmentation via `aug_transforms` with affine and lighting noise
- `EarlyStopping` and `SaveModel` callbacks
- Fine-tuning for 50 epochs

Our final model achieved **68.7% validation accuracy** (Figure 1) with consistent improvements over time, as shown in the train/val loss curves (Figure 2). We observed a smooth and steady convergence without overfitting, indicating strong generalization. The top-performing model was exported and evaluated further.

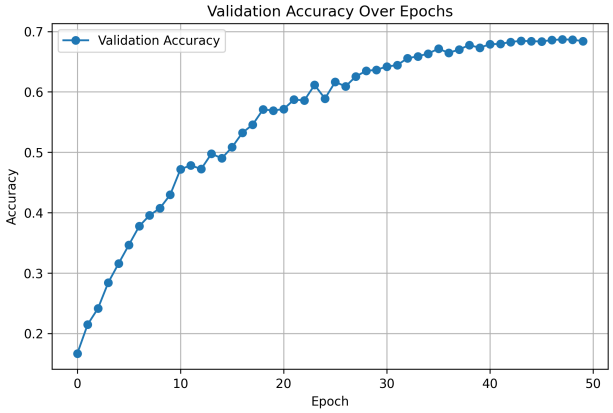

Fig. 1. Validation accuracy over 50 epochs in FastAI experiment.

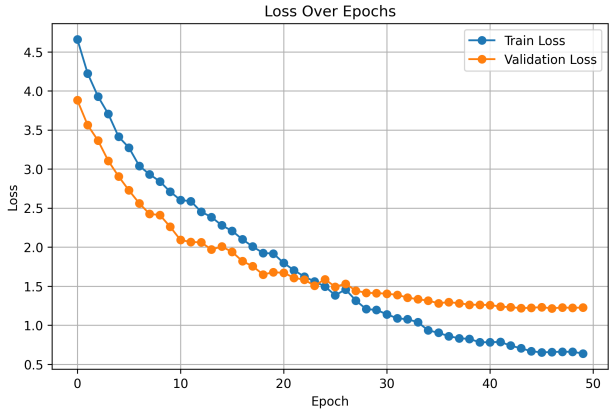

Fig. 2. Training vs Validation Loss Curves.

4.4 Evaluation Tools

We conducted an in-depth evaluation using the following tools:

- **Confusion Matrices:** Raw and normalized versions revealed that the model performed well on visually distinctive classes (e.g., Hummer SUV 2000, BMW X5 SUV), but struggled on similar sedans.
- **Top Losses:** Images with the highest prediction loss (Figure **??**) highlighted confusion between nearly identical vehicle types.
- **Class-wise Accuracy:** CSV export showed per-class performance ranging from 25% to 95%, reflecting dataset imbalance and class difficulty.

- **Classification Report:** Precision, recall, and F1-scores (macro avg: 0.69) provided a well-rounded performance profile.

4.5 Ablation and Training Insights

Throughout model development, we conducted several iterations to optimize training performance, incorporating lessons from our initial PyTorch baseline and experimental trials in FastAI. This section highlights the impact of specific components and decisions made during training.

4.5.1 Effect of Training Framework

Our initial implementation in raw PyTorch resulted in slower experimentation and lower validation accuracy (max 37.2%) due to limited automation and callbacks. Transitioning to FastAI provided high-level abstractions such as `vision_learner`, streamlined data augmentation, and plug-and-play callbacks for early stopping and learning rate tuning. This contributed significantly to the improved final accuracy of 68.7%.

4.5.2 Impact of Learning Rate Tuning

The learning rate finder in FastAI suggested a value of 4.37×10^{-3}, corresponding to the loss curve's steepest descent region (Figure 6). Using this value under a 1-cycle learning rate policy ensured faster convergence and prevented oscillations observed in earlier fixed-LR runs.

4.5.3 Augmentation vs. Overfitting

The introduction of controlled augmentations using `aug_transforms` — including flips, zoom, rotation, and lighting noise — improved generalization and reduced overfitting. In contrast, our PyTorch pipeline had simpler augmentation logic and suffered from early stagnation in validation accuracy despite continued training accuracy improvements.

4.5.4 Callbacks and Early Stopping

Using the `SaveModelCallback` and `EarlyStoppingCallback` helped preserve the best weights and halted training when validation accuracy plateaued. These mechanisms prevented overfitting and reduced training time by avoiding unnecessary epochs.

4.5.5 GPU Acceleration and Batch Size

Running on Google Colab's A100 GPU allowed us to increase batch size (from 32 in PyTorch to 64 in FastAI) and train for 50 epochs without memory issues. This improved training stability and helped generalize the model better across classes.

4.5.6 Frozen vs. Unfrozen Layers

Early experiments explored different parameter groups. Freezing the initial convolutional layers for the first few epochs and gradually unfreezing all layers proved effective. This prevented catastrophic forgetting of pretrained ImageNet weights and allowed lower layers to retain generalizable features.

4.5.7 Label Smoothing and Loss Functions

While our initial PyTorch model used label smoothing (0.1), the FastAI training relied on default cross-entropy without smoothing. Despite this, performance improved due to better regularization via augmentation, callbacks, and learning rate scheduling — showing that optimization strategy had a greater effect than modifying the loss function alone.

4.6 Performance Insights

4.6.1 Per-Class Accuracy Insights

To better understand the model's strengths and weaknesses, we analyzed per-class accuracy and highlighted both the top-performing and most challenging vehicle categories. Table 2 and Table 3 show the five classes with the highest and lowest validation accuracy, respectively.

TABLE 2
Top 10 Performing Classes by Validation Accuracy

Car (Class) Name	Accuracy
MINI Cooper Roadster Convertible 2012	1.00
1.00	1.00
Chevrolet HHR SS 2010	1.00
1.00	1.00
FIAT 500 Abarth 2012	0.95
1.00	0.97
Ford F-150 Regular Cab 2012	0.92
1.00	0.96
Ford F-150 Regular Cab 2007	0.88
1.00	0.94
BMW 7 Series Sedan 2012	1.00
0.86	0.92
Plymouth Neon Coupe 1999	1.00
0.83	0.91
Jeep Compass SUV 2012	1.00
0.83	0.91
Dodge Dakota Club Cab 2007	0.88
0.83	0.85
Chevrolet Avalanche Crew Cab 2012	0.86
0.83	0.84

TABLE 3
Bottom 10 Performing Classes by Validation Accuracy

Car (Class) Name	Accuracy
Audi S5 Coupe 2012	0.22
0.25	0.23
Spyker C8 Coupe 2009	0.28
0.22	0.25
Audi S4 Sedan 2012	0.36
0.30	0.33
Acura ZDX Hatchback 2012	0.38
0.33	0.35
Rolls-Royce Phantom Drophead Coupe 2012	0.40
0.33	0.36
Audi 100 Wagon 1994	0.45
0.38	0.41
Hyundai Veracruz SUV 2012	0.50
0.36	0.42
Hyundai Sonata Hybrid Sedan 2012	0.54
0.36	0.43
Chevrolet Malibu Hybrid Sedan 2010	0.60
0.40	0.48
Mercedes-Benz 300-Class Convertible 1993	0.67
0.40	0.50

These results reflect common challenges in fine-grained vehicle classification. The top-performing classes, such as

the Chevrolet HHR SS 2010 and the FIAT 500 Abarth 2012, have highly distinctive features (e.g., compact size, unique body shape, or color patterns) that make them easy to distinguish from other models. Their consistent visual signatures across the dataset may have also contributed to higher accuracy.

In contrast, the lowest-performing classes often share visual similarities with other models in the same brand or series. For example, the Audi S5 Coupe 2012 and Audi S4 Sedan 2012 have subtle differences from other Audi sedans, which can be difficult for a relatively shallow architecture like AlexNet to separate. Low class frequency and poor intra-class variation may also have contributed to misclassification. Vehicles like the Spyker C8 Coupe 2009 are rare and may appear similar to other exotic or sports cars in the dataset, further complicating recognition.

4.6.2 Confusion Matrix Analysis

A confusion matrix provides a holistic view of misclassifications across classes. The normalized version (Figure 3) helps identify relative error rates regardless of class frequency.

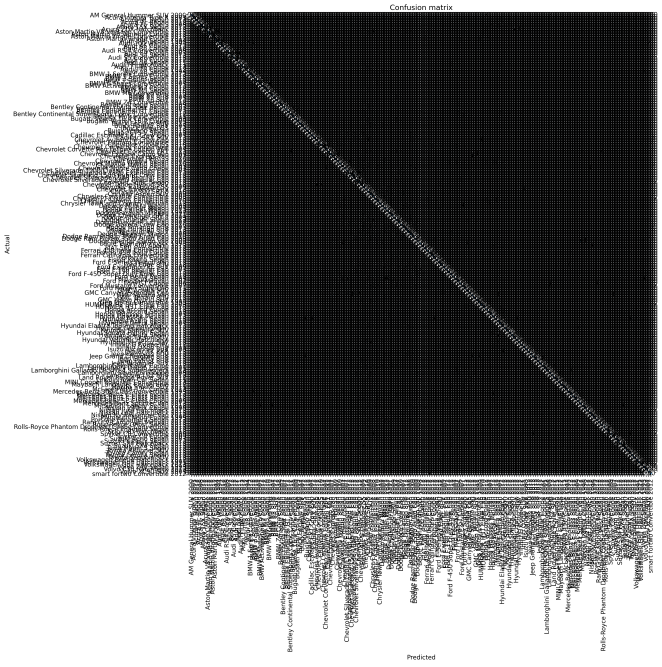

Fig. 3. Normalized Confusion Matrix revealing common confusions between Audi and Mercedes models

Diagonal dominance in the matrices confirms that most predictions fall on the correct class, but notable horizontal or vertical bands indicate confusion between visually similar car models, especially across brands like Audi, BMW, and Mercedes.

4.6.3 Classification Report Summary

The full classification report provides precision, recall, and F1-score per class. Figure **??** summarizes macro and weighted averages.

The accuracy metric represents the percentage of correct predictions across all 8041 validation images. The macro average provides an unweighted mean of F1-scores across all classes, treating each class equally. In contrast, the weighted average considers class support (number of instances per class), offering a more realistic reflection of overall performance when class imbalance is present. These values reflect consistent generalization across many classes, though some minority or visually overlapping classes suffer in precision.

TABLE 4
Global Classification Metrics

Metric	Value
Accuracy	0.6871
Macro Avg F1-score	0.69
Weighted Avg F1-score	0.69

Performance Interpretation:

- The macro and weighted F1-scores are nearly identical, suggesting a relatively balanced model performance across classes, even if some are underrepresented.
- Precision and recall values for many individual classes (not shown here due to space) reflect challenges in distinguishing visually similar vehicles—particularly luxury sedans, certain sports cars, and models from the same brand with minimal body variation.
- Notable class-level performance disparities were observed: vehicles with distinctive silhouettes (e.g., the AM General Hummer SUV 2000) achieved high precision and recall, while several Audi sedans and convertibles suffered from confusion, possibly due to limited distinguishing features and frequent class overlap.

This classification report reinforces findings from the confusion matrix and top-loss analysis, confirming that our fine-tuned AlexNet model performs well in many realistic cases but can still misclassify when faced with visually ambiguous or underrepresented examples.

4.6.4 Top Loss Visualizations

We use top-loss plots to visually inspect the most incorrect predictions by the model. Figure 4 presents a subset of those predictions with associated ground truth and predicted labels.

These examples help reveal confusion patterns, such as sports cars being mistaken for similar body-type coupes or convertibles, especially under challenging lighting or angles.

4.6.5 Learning Progression and Training Dynamics

Figure5 illustrates the model's learning over 50 epochs. The performance steadily increased, confirming effective transfer learning with minimal overfitting.

Early peaks in validation accuracy confirmed the role of learning rate scheduling and early stopping, which were introduced after initial experimentation.

4.6.6 Learning Rate Tuning

We used FastAI's built-in learning rate finder to automatically suggest an optimal value. Figure 6 shows the LR curve with the valley marked.

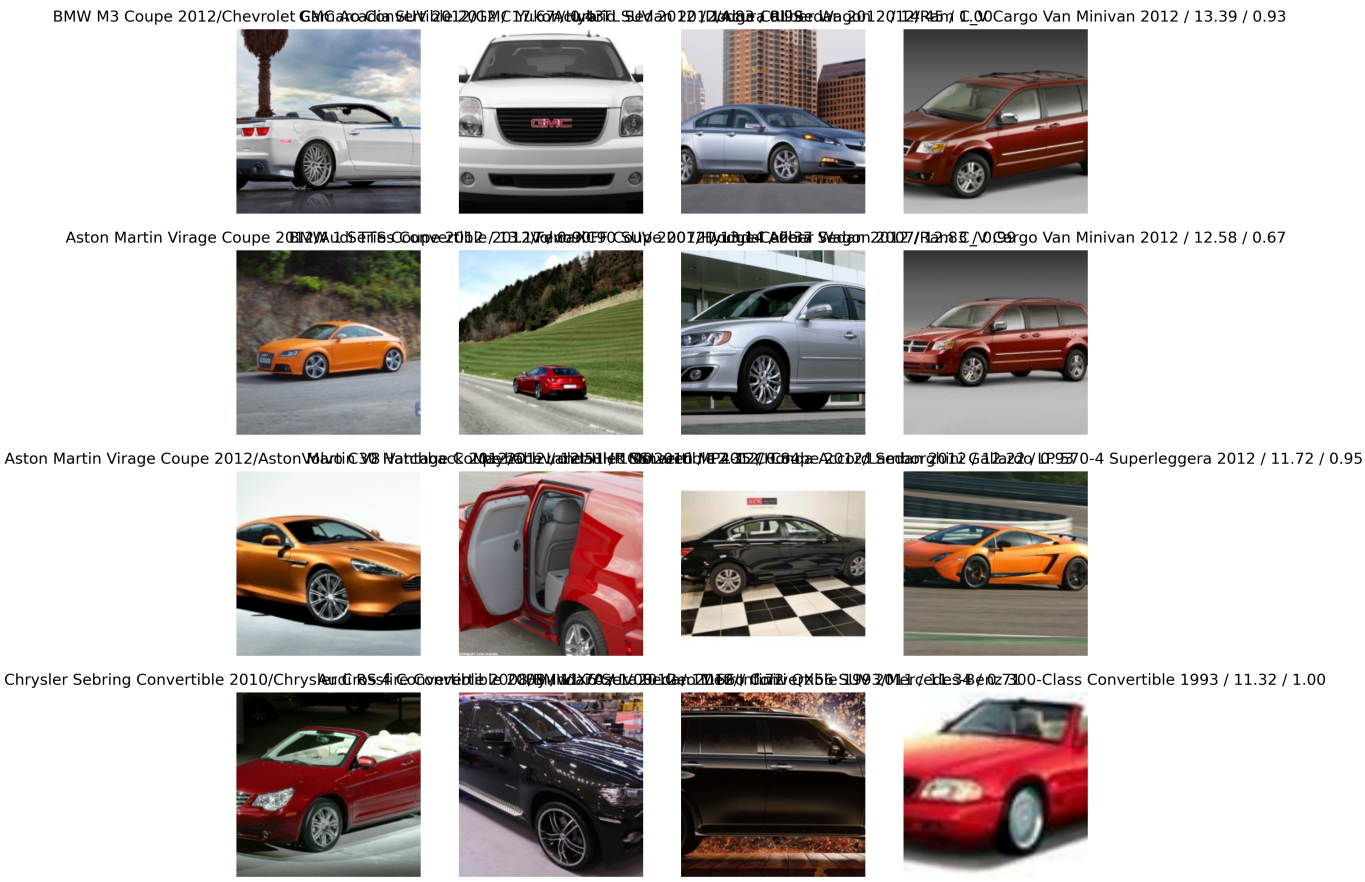

Fig. 4. Top Loss Examples (Across Classes)

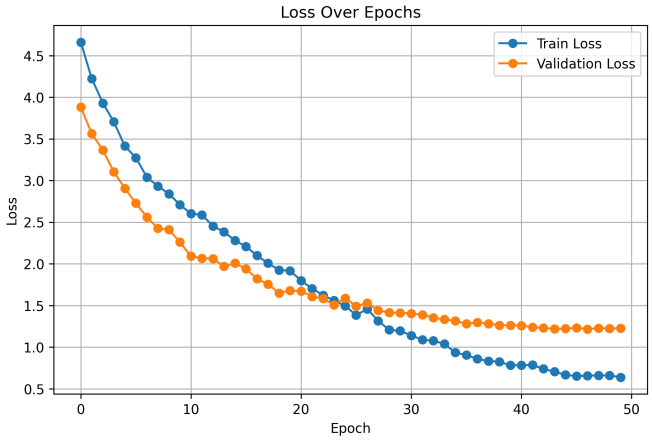

Fig. 5. Training vs Validation Loss

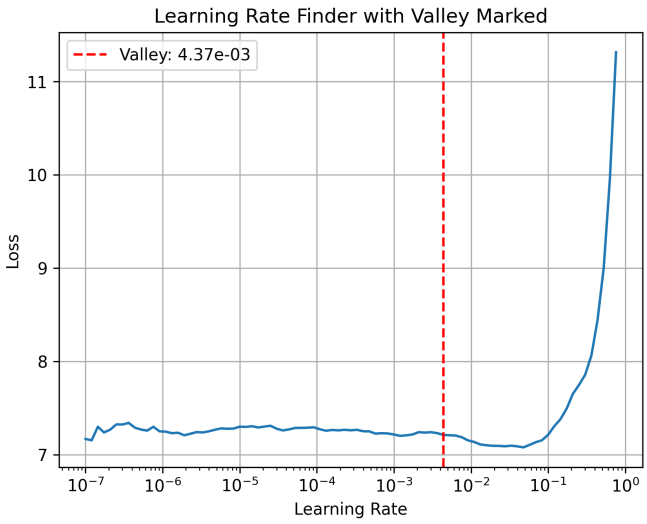

Fig. 6. Learning Rate Finder (Valley at 4.37×10^{-3})

This automated technique provided a strong starting point and helped avoid issues seen in earlier manual tuning.

4.6.7 *Qualitative Prediction Samples*

To further assess the model's behavior, we visualize a selection of randomly drawn predictions from the validation set. Each image is annotated with the ground truth (top, green) and predicted label (bottom, red if incorrect, green if correct). Figure 7 illustrates the model's performance across a range of vehicle types and brands.

This visualization helps identify consistent behavior pat-

Fig. 7. Sample Predictions from the Validation Set. Correct predictions are labeled in green, while incorrect predictions are labeled in red.

terns and potential weaknesses. We observe that:

- Vehicles with distinct rear or front shapes like the Ford F-450 and Toyota Camry are often predicted correctly.
- Misclassifications tend to occur between visually similar sedan models or convertibles that share body styles across brands (e.g., Chrysler 300 vs. Buick Rainier).
- Certain fine-grained distinctions, such as between two model years of the Audi TT Hatchback, are handled well, while others (e.g., convertibles from the early 1990s) remain more error-prone.

By combining this qualitative sample with earlier class-wise accuracy, F1-scores, and confusion matrix insights, we confirm that the model excels when structural features are strong indicators, and struggles more with subtle styling nuances or overlapping class identities.

4.7 Limitations

While our FastAI-based pipeline significantly outperformed the baseline implementation, several limitations remained that could affect generalization and scalability:

- **Architectural Constraints:** AlexNet's relatively shallow depth limits its ability to model fine-grained distinctions across highly similar vehicle classes. More modern architectures with residual or attention-based mechanisms may offer improved feature extraction capabilities.
- **Dataset Imbalance:** The Stanford Cars dataset exhibits class imbalance, with some vehicles appearing far more frequently than others. This likely contributed to reduced precision and recall for under-represented or rare models, such as the Spyker C8 Coupe.
- **No Test Set Evaluation:** Our evaluation was conducted solely on the validation split due to lack of access to official test set labels. As such, the reported performance may not reflect real-world generalization.
- **Limited Hyperparameter Search:** Due to time and computational constraints, we relied heavily on FastAI defaults and a single LR finder result. A more exhaustive hyperparameter sweep (e.g., optimizer types, layer freezing schedules, augmentation inten-

sity) might yield additional performance gains.
- **Misleading Visual Cues:** Some misclassifications appeared to result from background context or lighting cues, which may indicate that the model overfits to dataset-specific artifacts rather than learning generalizable vehicle features.
- **Hardware Dependency:** Our best results were obtained using a high-end A100 GPU in Google Colab Pro. Performance may degrade significantly when retraining in lower-resource environments or without GPU acceleration.

Understanding these limitations provides valuable direction for future experiments and highlights areas where further improvements or architectural changes could drive better results.

5 Conclusion

In this project, we explored the effectiveness of fine-tuning AlexNet for the task of Vehicle Make and Model Recognition (VMMR) on the Stanford Cars dataset. Despite its relatively simple architecture, AlexNet delivered competitive performance when paired with modern training techniques such as transfer learning, dynamic learning rate discovery, data augmentation, and early stopping.

Our initial baseline using raw PyTorch reached a validation accuracy of 36.1%, highlighting the challenges of manual pipeline construction and tuning. In contrast, our final FastAI-based pipeline achieved a significantly higher validation accuracy of 68.7%, showcasing the impact of using robust abstractions and tooling.

Through extensive analysis using confusion matrices, class-wise accuracy distributions, and classification reports, we identified consistent strengths in recognizing distinctive vehicle classes, while struggles remained for visually similar or underrepresented models. Our insights also suggest that legacy architectures like AlexNet, though often overlooked, can still provide solid results for fine-grained classification tasks when properly optimized.

Future work may involve experimenting with deeper architectures like ResNet or EfficientNet, incorporating attention mechanisms, or leveraging vision transformers. Additionally, techniques such as curriculum learning, few-shot learning, or domain adaptation could further improve generalization in class-imbalanced or real-world scenarios.

References

[1] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2013, pp. 554–561.

[2] M. A. R. Khan, R. Talukder, M. A. Hossen, and N. Jahan, "Car make and model recognition using convolutional neural network: fine-tune AlexNet architecture," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 33, no. 1, pp. 370–379, 2024.

[3] A. Lediur, "Monza: Image Classification of Vehicle Make and Model Using Deep Learning," *CS231n Course Project Report*, 2015.

[4] J. E. Espinosa, S. A. Velastin, and J. W. Branch, "Vehicle Detection Using AlexNet and Faster R-CNN Deep Learning Models: A Comparative Study," in *Advances in Visual Informatics*, 2017, pp. 3–15.

[5] S. A. Najeeb, R. H. Raza, A. Yusuf, and Z. Sultan, "Fine-Grained Vehicle Classification in Urban Traffic Scenes using Deep Learning," *arXiv preprint arXiv:2111.09403*, 2021.

[6] S. Aziz, S. Naseer, M. U. Khan, S. M. A. Shah, and K. Iqtidar, "Vehicle Make and Model Recognition using Deep Transfer Learning and Support Vector Machines," *ResearchGate*, 2021.

[7] R. Tafazzoli and R. Safabakhsh, "A Large and Diverse Dataset for Improved Vehicle Make and Model Recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017.

[8] J. Li, "Stanford Cars Dataset," *Kaggle*, [Online]. Available: https://www.kaggle.com/datasets/jessicali9530/stanford-cars-dataset

[9] Y. Cyizhuo, "Stanford_Cars_dataset," *GitHub Repository*, [Online]. Available: https://github.com/cyizhuo/Stanford_Cars_dataset

[10] FastAI, "fastai v2 Library," [Online]. Available: https://docs.fast.ai/

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.