Car-make Classification Using Deep Learning Approaches

Giuseppe Labate†, Sana Amiri Manesh†

***Abstract*—This paper investigates a deep learning approach for image classification, focusing on the detection and recognition of car makes using the ResNet50 architecture. The study systematically examines the role of data augmentation in improving classification performance and addressing class imbalance in the CompCars dataset, which contains a diverse collection of car images. Various augmentation techniques, including geometric transformations, color adjustments, and random cropping, were applied to enhance data diversity and reduce overfitting. Additionally, the performance of two loss functions, focal loss and categorical cross-entropy loss, was analyzed to determine their effectiveness in handling class imbalance. Experimental results show that data augmentation significantly improved model generalization. While focal loss was particularly effective in handling class imbalance before data augmentation, cross-entropy loss outperformed focal loss after class imbalance was addressed through augmentation. This research highlights the importance of leveraging data augmentation and appropriate loss functions for optimal performance in car make classification tasks.**

***Index Terms*—Data Augmentation, Neural Networks, Residual Neural Networks, Optimization, Deep Learning.**

I. Introduction

Every day, a very large number of vehicles passes through the streets of our cities and towns. In fact, even though the car sales market volume is far from its 2007 peak, statistics from 2016 state that in Italy alone there are 625 passenger cars for every 1000 inhabitants, without considering trucks or motorbikes. It should not come as a surprise, then, that the idea of detecting and classifying these vehicles can be useful for a wide range of applications, from security to commercial use [1]. In the rapidly advancing field of machine learning and computer vision, classification tasks, such as car make recognition, have garnered significant attention due to their practical applications in automotive industries and beyond. The availability of large-scale datasets, like the CompCars dataset, has enabled the development of sophisticated models for car make classification. However, challenges arise from the dataset's large size, class imbalances, and the inherent difficulty in distinguishing between similar car makes and models. This paper presents a comprehensive approach to car make classification by leveraging deep learning techniques, particularly using the ResNet-50 architecture, and addressing the aforementioned challenges with innovative solutions such as data augmentation and loss function optimization.

†Department of Information Engineering, University of Padova, email: {name.surname}@studenti.unipd.it

Special thanks / Professor Jacopo Pegoraro

We begin by describing the high-level processing pipeline employed for this task, which includes data pre-processing, model design, and evaluation strategies. The dataset, CompCars, consists of a diverse range of images captured from various viewpoints, both in surveillance and web-based settings, providing a rich set of features for classification. Data pre-processing, including cropping using bounding boxes, plays a crucial role in managing the large dataset and enhancing model performance by focusing on the car's key attributes.Additionally, the paper examines the role of data augmentation in addressing class imbalance and achieving better results. It also compares the performance of cross-entropy and focal loss before and after applying data augmentation. The results show that focal loss performs better when class imbalance is present; however, after implementing data augmentation, its advantage diminished, indicating that data augmentation effectively mitigated the imbalance and improved overall model performance.

Through this approach, we aim to present a robust solution for multi-class car make classification, demonstrating the effectiveness of ResNet-50 for such tasks while also contributing valuable insights into how various techniques can improve model performance and efficiency.

II. Related Work

Buzzelli et al. [2] make several notable contributions to car image classification. It highlights biases in existing dataset splits, proposes a more realistic training/testing configuration, and extends dataset annotations with hierarchical labels and car-tight bounding boxes. However, the findings primarily focus on dataset improvements rather than innovative classification methodologies. Skolik et al. [3] explore the optimization of hyperparameters for hybrid quantum-classical neural networks applied to car classification. It introduces a tensor train optimization (TTO) method, demonstrating improved efficiency and reduced computational cost compared to traditional grid search approaches. The hybrid quantum-classical model achieves higher classification accuracy on the Stanford Cars dataset than its classical counterparts. The study emphasizes the potential of quantum-inspired techniques in addressing complex machine learning tasks while suggesting further exploration of sample complexity and generalization bounds for such models. Dwivedi et al. [4] address vehicle damage classification and detection, focusing on automating insurance claim processing using deep convolutional neural networks. Pre-trained CNN models on ImageNet, combined with advanced techniques, achieved a classification accuracy

of 96.39%, while the YOLO detector obtained a maximum mAP score of 77.78% for damage region detection. A proposed pipeline integrates classification and detection for enhanced robustness. The study highlights the need for a more diverse dataset to advance automated vehicle damage identification systems. Yang et al. [1] used CNNs to classify fine-grained car models from images, achieving significant improvements over traditional methods. CompCars dataset has been widely used for car model and part classification due to its extensive variety of car models and part annotations. Ni el al. [5] focuse on vehicle attribute recognition, ranging from coarse-grained classifications like vehicle type to fine-grained distinctions such as make and model. It surveys existing algorithms and compares two methods: direct classification and a flexible metric learning approach. A simulated real-world scenario is designed to evaluate these methods, providing insights into their effectiveness. This study advances the understanding of vehicle attribute recognition by bridging gaps in literature and testing practical applications.

Class imbalance is a common issue in real-world datasets, particularly in tasks like car part classification, where certain parts or models may be overrepresented compared to others. Traditional Loss functions like categorical cross-entropy often perform poorly in such scenarios, leading to biased model prediction.Lin et al. [6] introduce focal loss in the context of dense object detection. Focal loss introduced as a modification of cross-entropy loss, mitigates this issue by focusing on hard-to-classify examples. It down-weights the loss contribution from easy examples allowing the model to focus more on underrepresented or challenging cases. Zhu et al. [7] focus on addressing the class imbalance issue in driving scene recognition, which results in different distribution patterns between majority and minority classes. They propose a novel class focal loss method that integrates category quantity distribution to better balance easy and difficult samples during training, improving recognition accuracy for underrepresented classes.

III. Processing Pipeline

This section outlines the high-level approach and methodology employed for the deep learning-based classification of car makes using the CompCar dataset. The pipeline encompasses several stages, each designed to ensure data integrity, model robustness, and accurate predictions. The key stages in the pipeline are as follows:

Data Collection and Preprocessing : The project begins with the CompCar dataset, a comprehensive dataset containing diverse images of cars with associated metadata. The data preprocessing stage ensures the input data is prepared for the deep learning models by performing the following steps: Data Normalization, Data Augmentation

Data Splitting: The dataset is partitioned into three subsets: Training Set, Validation Set, Test Set

Model Design and Training: The project employs a residual neural network (ResNet) architecture tailored for image multi-label classification tasks. Key aspects of model design and training include:

- Model Architecture: Implementing a custom ResNet50 model and fine-tuning its layers for the specific classification task.
- Loss Function: Use of a categorical cross-entropy loss function and focal loss suitable for multi-class classification problems.
- Optimization: Employing optimization techniques such as Adam with a learning rate scheduler to enhance training efficiency.
- Regularization: Incorporation of dropout and early stopping to mitigate overfitting.

Evaluation Metrics: During and after training, the model is evaluated using the top-1 and top-5 accuracy metrics.

Model Validation and Early Stopping: The training process utilized:

- Checkpointing: Automatically saving the model with the best performance on the validation set.
- Early Stopping: Halting the training process when the validation performance ceased to improve, preventing overfitting and saving computational resources.

Final Model Evaluation: For the final evaluation, the model was tested on the test set, with a function that would randomly selects an image from the test set and then classifies it using the network.

IV. Dataset

"CompCars" data set is a large-scale dataset that covers not only different car views, but also their different internal and external parts, and rich attributes. The dataset provides both a *train_test_split* file, which defines the division between training and test sets, and *bounding box* annotations that delimit the cars within the images.

Data preparation: A *preprocessing step* was implemented using the *bounding boxes* provided in the dataset's label folder to crop the images, focusing exclusively on the cars while removing unnecessary background details. With this approach by focusing solely on the car, the input data fed into the model became cleaner, with less noise from background elements such as roads, trees, or other vehicles. This likely improved the *model's performance* by emphasizing the features most relevant to the *classification task*, such as car shape, design, and unique attributes. Additionally, this step only needs to be performed once, making the training process faster.

1) **Loading Image Paths and Labels:** Image paths and their associated labels are loaded directly from the dataset directory structure. Each class (car make or model) is represented by its corresponding image paths, which are mapped to unique integer labels derived from the folder hierarchy. The CroppedCarDataset class dynamically iterates through the dataset directories, extracting image paths and associated labels based on their file paths. These integer labels are then mapped to car make and car model names using the *make_model_name.mat* file in the dataset.

2) **Data Splitting and Loaders:** The dataset is split into training, validation, and test sets:
   - **Training Set:** Includes augmented samples and original data, shuffled for robust training.
   - **Validation Set:** Comprises 30% of the training data and is used for hyperparameter tuning and early stopping.
   - **Test Set:** Contains unseen data for evaluating the model's performance.

   Data loaders are created using PyTorch's DataLoader class to enable efficient data handling with batch processing and parallelism. Batch size is set to 32, and transformations are applied dynamically during data loading.
3) **Data Normalization:** To normalize the data across the training, validation, and test sets, we first compute the mean and standard deviation of the training set. The values obtained are as follows:
   Mean: tensor([0.4262, 0.4063, 0.4067])
   Standard Deviation: tensor([0.2833, 0.2797, 0.2832])
   These values are then used to normalize the data in all three sets to ensure consistent scaling.

4) **Balancing Class Distributions with Data Augmentation:** To address class imbalance in the dataset, augmentation techniques are applied to generate additional samples for underrepresented classes.Additionally, data augmentation helps prevent overfitting by artificially increasing the diversity of the training data, allowing the model to generalize better to unseen data.
   Underrepresented classes with fewer than 250 samples were identified using a label counting function. For each of these classes, augmented samples were generated to increase their total count to 400. This process involved constructing a function to detect classes below the threshold, randomly selecting images to maintain the original data distribution, and applying augmentation techniques. The AugmentedDataset class then combined the original and augmented samples into a unified dataset for training. Originally, the training dataset consisted of 11,211 images; after applying data augmentation, this number increased to 30,099.
   Augmentation Pipeline: The augmentation process included the following transformations::
   - **Geometric Transformations:** Random affine transformations, rotations, and horizontal flips.
   - **Color Transformations:** Adjustments to brightness, contrast, saturation, and hue.
   - **Resizing and Cropping:** Random resized cropping and center cropping to ensure consistent input dimensions.

   These transformations are implemented using Pytorch's transforms. Compose functionality and applied dynamically during training. Figure 2 illustrates the class distribution before and after data augmentation, highlighting the significant imbalance in the original dataset and its resolution through augmentation.

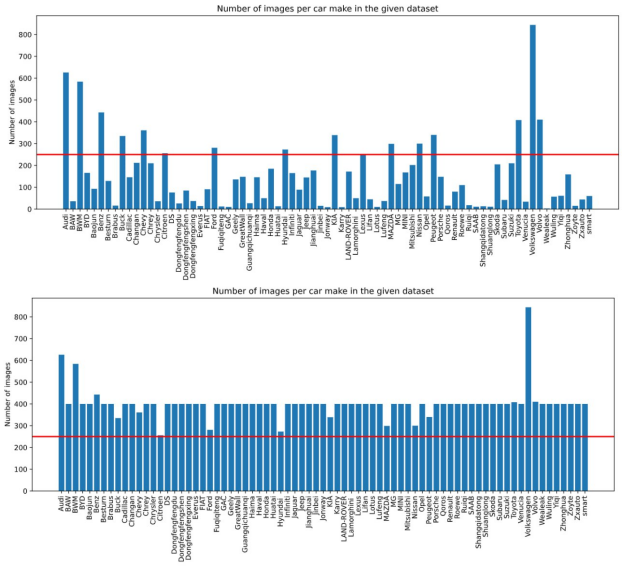

Fig. 1: The class distribution before data augmentation (top) and The class distribution after data augmentation (bottom)

Training vs. Validation/Test Transforms: Different transformations are applied to the training set compared to the validation and test sets to serve distinct purposes:
- **Training Set:** Augmentations such as random affine transformations, color jitter, and random cropping introduce variability in the training data. This helps the model generalize better by simulating real-world conditions and reducing overfitting.Transformations like random flips and rotations ensure that the model is not biased toward specific orientations or positions.
- **Validation and Test Sets:** Only basic transformations (e.g., resizing and normalization) are applied to maintain consistency and evaluate the model's performance on clean, unaltered data. Augmentations are avoided to ensure that the validation and test results reflect the model's true generalization ability.

V. Learning Framework

1) **Model Construction:** The backbone of the learning framework is the ResNet-50 architecture, constructed for multi-class classification with 163 car makes.
   Residual Blocks: The architecture includes custom implementations of IdentityBlock and ConvolutionalBlock. The IdentityBlock preserves input dimensions, while the ConvolutionalBlock performs down-sampling with a stride of 2.
   Custom Layers:
   - **Dropout Layers:** Applied after residual blocks with a rate of 0.2 to mitigate overfitting.

- **Batch Normalization:** Normalizes activations after convolutional layers to improve convergence and training stability.
- **Final Linear Layer:** A fully connected layer maps the extracted features to the 163 output classes, representing car makes. The input size to the linear layer is 8192, computed using the formula:

$$O = \left(\frac{d - F + 2p}{S}\right) + 1 \tag{1}$$

Here, d is the input dimension, F is the filter size, P is the padding, and S is the stride. This formula is used to compute the size of the output of convolutional layers, ensuring the correct input size for the final layer.

Output Function: The model uses the softmax function on the output layer to calculate class probabilities.

Weight Initialization: Xavier initialization is applied to both convolutional and fully connected layers to improve convergence when training from scratch.

2) **Training Process:** The models were trained for 20 epochs with robust optimization and monitoring strategies to ensure effective learning. The training process evaluates the performance of two loss functions.

Cross-Entropy Loss: This loss function was primarily used for standard multi-class classification tasks. It calculates the difference between predicted class probabilities and true class labels.

Focal Loss: A custom implementation of focal loss was also utilized to address class imbalance. This function emphasizes hard-to-classify samples using parameters α (alpha = 1) and γ (gamma = 2).

Optimization: The Adam optimizer with a learning rate of 1e-3 was employed for parameter updates. A StepLR scheduler reduced the learning rate by a factor of 0.1 every 7 epochs.

Early Stopping: Training incorporated early stopping with a patience of 5 epochs and a delta of 0.05, halting further training if the validation loss stopped improving. The validation loss had to decrease by at least 0.05 compared to the previous best loss to be considered an improvement.

Checkpointing: Checkpoints were saved whenever the validation loss improved, ensuring that the best-performing model weights were preserved.

3) **Evaluating and Testing:** The trained model's performance is evaluated on unseen test data using detailed metrics.

Test Loop: The model is set to evaluation mode, and test data is processed in batches of size 16. Predictions and ground truth labels are stored for comparison.

Metrics:

- **Top-1 Accuracy:** Measures the percentage of correct predictions.
- **Top-5 Accuracy:** Checks if the correct class is among the top 5 predictions, which is particularly useful for multi-class tasks.

Inference: A function is provided to predict the class of a single image by pre-processing it, passing it through the model, and decoding the output.

VI. Results

The car make classification task was performed using the ResNet50 architecture, trained both with and without data augmentation, and using two loss functions: cross-entropy and focal loss. All models were trained with early stopping (patience of 5) and evaluated on the validation and test sets. Data augmentation significantly improved the model's performance by increasing the diversity of training samples and reducing overfitting. While focal loss addressed class imbalance and performed better than cross-entropy loss before applying data augmentation, once data augmentation was introduced to mitigate the imbalance, focal loss did not lead to further improvements in classification performance.

1) **Impact of Data Augmentation:** In this section, we present the results of applying data augmentation to address class imbalance in the dataset.The impact of data augmentation on the model's performance was substantial. Without data augmentation, the model showed clear signs of overfitting. In this setup, the top-1 accuracy was 18%, and the top-5 accuracy was 45%. Conversely, when data augmentation was applied, the model's performance improved drastically, with top-1 accuracy rising to 40% and top-5 accuracy increasing to 68%. This improvement in accuracy, along with a stabilization in validation loss, highlighted the critical role of data augmentation in enhancing the model's generalization and robustness. The graphs illustrating these results are available in Figure 2.The top graph in Figure 2, which shows the results for before the data augmentation, incorrectly displays 60 epochs, while it should have been two separate graphs, each representing 30 epochs. Unfortunately, we couldn't correct this issue due to time constraints.

   To evaluate the performance of our model, in addition to validation accuracies, we developed a test function that randomly selects an image from the test set and has the network classify it. Before applying data augmentation, the model exhibited a tendency to predict labels corresponding to classes with more samples in the dataset. However, after applying data augmentation, the model's performance significantly improved, correctly predicting labels most of the time.

2) **Impact of Loss Functions:** Before applying data augmentation, the comparison between focal loss and cross-entropy loss revealed key differences in performance. Using focal loss, the model showed better convergence and stability, as evidenced by smoother loss curves. Validation accuracy improved steadily, demonstrating the model's ability to handle class imbalances. In contrast, after applying data augmentation, the model performed better with cross-entropy loss, as shown in Figure 3.

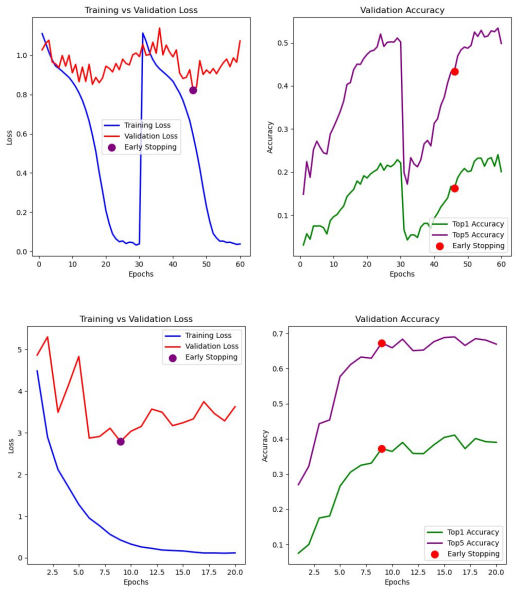

Fig. 2: The results before data augmentation using cross entropy loss (top) and The results after data augmentation using cross entropy loss(bottom)

We believe this improvement occurred because data augmentation already addressed the class imbalance, making the use of focal loss unnecessary.

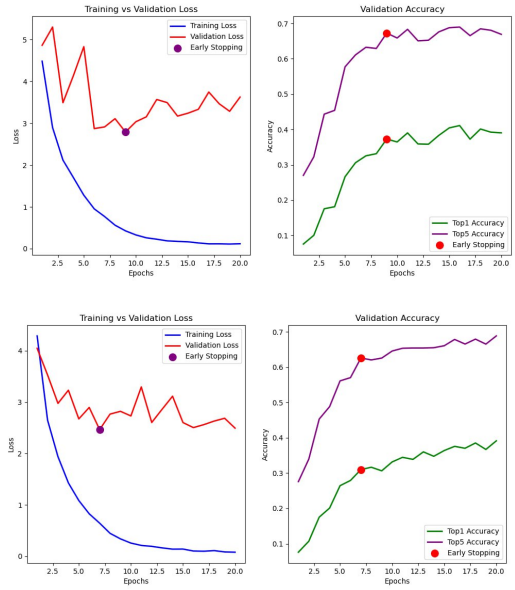

Fig. 3: The results using the cross entropy loss (top) and The results using the focal loss (bottom)

VII. Concluding Remarks

This project successfully demonstrated the application of deep learning techniques for car make classification using the CompCars dataset. The findings highlight the effectiveness of ResNet50 and data augmentation in addressing class imbalance and enhancing classification performance. However, we encountered several challenges throughout the process. Initially, our goal was to tackle both car make and car model classification, and we designed our code to accommodate both tasks. However, due to difficulties in achieving the desired accuracy for car make classification, we prioritized refining this model rather than expanding to a new classification task.

One of the primary challenges was dataset imbalance, particularly in the Volkswagen class, which contained nearly twice as many samples as other car makes. While data augmentation was applied, reducing the number of Volkswagen samples was not a viable option, as it would have resulted in the loss of valuable information given the wide range of models. However, retaining this imbalance inevitably introduced bias into the model, leading to an overrepresentation of Volkswagen in predictions.

Another challenge in data augmentation arose for car makes with limited samples. Despite applying diverse augmentation techniques to randomly selected images, classes with very few samples suffered from excessive repetition, limiting the effectiveness of augmentation.

Furthermore, an issue was identified in the final execution of the code. Previously, the structure of the splitting operations and transformations applied to the sets was different because we used precomputed mean and standard deviation as hard-coded values. Now, to unify everything, we had to modify the splitting and transformation structure, as normalization could not be applied before computing the mean and standard deviation. This change led to a decrease in top-1 accuracy to 31.7% and top-5 accuracy to 66%, as shown in Figure 4.

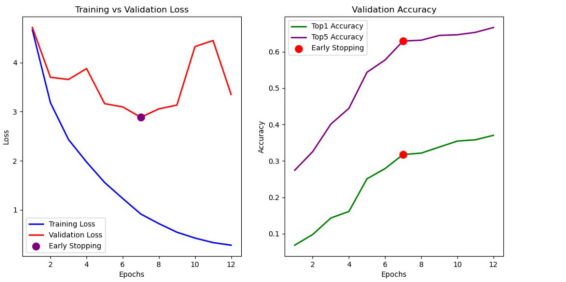

Fig. 4: The final results after calculating mean and standard deviation

Additionally, various optimization strategies were explored, including the removal of the dropout layer due to its limitations with structured data and the application of augmentation techniques such as Gaussian Blur and grayscale conversion. However, these approaches did not lead to performance improvements.

Our findings highlight the complexities of car make classification and emphasize the need for further research into alternative techniques to handle class imbalance, mitigate bias, and improve model robustness.

VIII. Exam rules

References

[1] C. C. L. Linjie Yang, Ping Luo and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3973–3981, 2015.

[2] M. Buzzelli and L. Segantin, "Revisiting the compcars dataset for hierarchical car classification: New annotations, experiments and results," *Sensors*, vol. 21, no. 2, p. 596, 2021.

[3] A. M. D. K. M. P. A. M. A. S. Asel Sagingalieva, Andrii Kurkin and D. V. Dollen, "Hyperparameter optimization of hybrid quantum neural networks for car classification," *arXiv*, vol. 2205, no. 04878, 2022.

[4] S. N. O. E. B. M. B. K. S. R. S. A. T. Mahavir Dwivedi, Hashmat Shadab Malik and A. Rathi, "Deep learning-based car damage classification and detection," in *Proceedings of the International Conference on Artificial Intelligence and Data Engineering*, vol. 1133 of *Advances in Intelligent Systems and Computing*, pp. 207–221, Springer, 2020. First Online: 14 August 2020.

[5] X. Ni and H. Huttunen, "Vehicle attribute recognition by appearance: Computer vision methods for vehicle type, make and model classification," *Pattern Recognition Letters*, vol. 93, pp. 357–368, 2021. Open access, Published: 26 June 2020.

[6] R. G. K. H. P. D. Tsung-Yi Lin, Priya Goyal, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, vol. 2017 of *IEEE*, pp. 2980–2988, IEEE, 2017. First Online: 2 October 2017.

[7] L. Y. K. L. Xianglei Zhu, Jianfeng Men, "Imbalanced driving scene recognition with class focal loss and data augmentation," *Neural Processing Letters*, vol. 13, pp. 2957–2975, 2022. Published: 07 June 2022.