Classification of Car Models with ResNet-50

Omer Cem Tabar†, Jacopo Pegoraro†

***Abstract*—Future of the car producer companies started to look for new solutions for marketing strategies in order to identify customer preferences through social media by taking the advantage of processing the images that are posted on social accounts. Identification of the preferences that are based on car brand and models resulted in development of complex processing techniques which includes CNNs and deeper neural network architectures. A key challenge is the choice of state-of-art where the collected data contains huge amounts and reflects various characteristic for the learning process. In this paper for leveraging the identification process for identifying the trend for marketing strategies, different deep neural network models including various ResNet with fine-tuned versions are investigated. As a design proposal, ResNet-50 model was identified due to high number of classes where the class imbalance was overcame with different methods. The goal is to increase the accuracy of the model for classifying car models, propose a solution that reduces the computational complexity and generalizes the CompCars dataset in a proper way. The results reveal that ResNet-50 pipeline performs well for identifying distinct car make/model classes and outperforms the shallower networks in terms of time, resource and computational requirements.**

***Index Terms*—Deep Learning, CompCars Dataset, Convolutional Neural Networks (CNN), Residual Networks (ResNet), Car Make and Model Classification**

I. Introduction

In recent years, publication of posts that include the cars started to become more common in every social media platforms. Sharing instincts of the social media users for their car brand/model preferences from all over the world evolved into a crucial information for the car producer companies. For this reason, analysis of the brand and model ownership of the users for brand marketing strategies paved the way for different identification and analysis methods for locating the best strategy for the significant cities. As an identification strategy, usage of automized image processing techniques where the car make and model informations can be extracted and verified in a well-trained identification models made the identification of marketing strategies in a more solid and easier way.

Due to complexity of the model/brand classification task, extracting the intended information requires complex architectures than regular image processing techniques. By the usage of one of the latest Deep Neural Network (DNN) architecture designs called ResNet50, it is intended to train a model that increases the accuracy of recognition of car models. Additionally, new design aims to decrease the complexity of other considered models in terms of computations.

†Department of Information Engineering, University of Padova, email: {oemercem.tabar}@unipd.it

Special thanks to Dr. Daniele Mari and Dr. Riccardo Mazzieri

In the paper, the organization defined as follows. In the first section related works are reviewed. Secondly, proposed architectural design for the problem is explained in processing pipeline. Thirdly, characteristics of the dataset that has been used is explained in Dataset Preparation section. Next, high-level architecture of the considered design was shown in processing pipeline section where the details of the algorithm and learning process is explained in Learning Framework section. And lastly, obtained results are discussed in Results section where the outputs and drawbacks are commented in conclusion section.

II. Related Work

Many different methods including the original paper for fine-grained classification of car make and car models were considered. Apart from the methods that includes the identification of the cars from its parts, methods that considers entirety of the cars were also used for extracting the texture and silhouette features for the identification of the car models in many studies. Consideration of the complexity of the task and vast amount of data that is considered, resulted in the implementation of deeper neural network architectures.

In Yang et al. [1], for the fine-grained car model classification viewpoint specific CNN architectures were implemented where the each model performance is analyzed on different view points in order to identify which views are more suitable for the task. Even though the approach in the original paper is not considering the entirety of the whole dataset, as a starting point it reflects what should be done. But due to the lack of entirety of the dataset and classes, more experiments should be done to remove the bias on viewpoints for the car model classification.

As the same intended work, Wille et al. [2] proposed several fine-tuned MobileNet architecture to compare with the GoogleNet model that is suggested by the original paper. It is indicated that MobileNet-1.00-224 model has 2.5% higher top-1 accuracy and proposes the balance between computational complexity and accuracy of the model. However, since the model only reflects a small part of the used CompCars dataset it is not certain that whether the performance of the model will be high in the whole dataset.

As an inherited project and improvement on these works, suggested deeper neural network architecture was implemented as ResNet50 where the considered dataset includes all the viewpoints for the car model classification task. With inclusion of more classes, more complex architectures has

been experimented and fine-tuned to be used in car model classification tasks.

III. Processing Pipeline

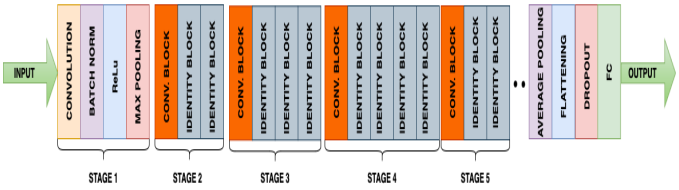

Fig. 1: Processing Pipeline Design

As it is known for many researchers, traditional neural networks and plain neural network suffers from vanishing gradient problem where the gradients are becoming really small each time that traverse the layers. Which results in catastrophic training and testing errors when the deeper architectures for neural networks are considered. Additionally, it may result in slow learning or non-learning circumstances of the models. As a solution, ResNet architecture introduces skip connections which denoted as residual connections that allows the gradients to traverse the layers more faster and easier.

By this reason, as a processing pipeline different considerations of the Residual Network designs were considered such as ResNet-18, ResNet-34 and ResNet-50. Since the amount of the images that we loaded in the data loaders are large and since there are 927 car make/model classes to be classified choice of design was identified as ResNet-50. As shown in the Fig. 1, the design architecture consists of 5 stages and a last block for ending layers in total which can be listed as follows:

- **Stage 1:** Consist of Convolution Layer, Batch Normalization Layer, ReLu Activation Layer and Max Pooling Layer.
  - **Convolution Layer:** Large convolution operation for processing the raw image to extract low-level features with various convolutional filters. Each filter identifies the low-level features locally where the filters slide over the image.
  - **Batch Normalization and ReLu:** For normalizing the activations and creating non-linearity.
  - **Max Pooling Layer:** To reduce the dimensions and keeping the important informations about the futures detected.
- **Stage 2:** One Convolution Block and two Identity Blocks.
  - **Convolution Block:** Reducing the dimension, extracting features with more filters
  - **Identity Blocks:** Learning Residuals. Input and output dimensions are maintained. Includes 3 Convolution layers that followed by batch normalization and ReLu activation layers.
- **Stage 3:** One Convolution Block and three Identity Blocks.
  - **Convolution Block:** Reducing the dimension, extracting features with more filters
  - **Identity Blocks:** Learning More Residuals. Input and output dimensions are maintained. Includes 3 Convolution layers that followed by batch normalization and ReLu activation layers.
- **Stage 4:**One Convolution Block and four Identity Blocks.
  - **Convolution Block:** Reducing the dimension, extracting features with more filters
  - **Identity Blocks:** Learning More Residuals.Input and output dimensions are maintained. Includes 3 Convolution layers that followed by batch normalization and ReLu activation layers.
- **Stage 5:** One Convolution Block and two Identity Blocks.
  - **Convolution Block:** Reducing the dimension, extracting features with more filters.
  - **Identity Blocks:** Learning More Residuals. Input and output dimensions are maintained. Includes 3 Convolution layers that followed by batch normalization and ReLu activation layers.
- **End Layers:** Average Pooling Layer, Flattening, Dropout and Fully Connected Layer.
  - **Average Pooling Layer:** Reducing the dimension of features in one dimension. Averaging each feature map and creating a feature vector.
  - **Flattening:** Conversion of the pooled features in one dimensional vector.
  - **Dropout:** Regularization, prevention from overfitting
  - **Fully Connected Layer:** Final class production, gives scores for each classification.

Design of the architecture depends on the intended task and the dataset that will be used. For the car model classification task using CompCars dataset, ResNet-50 implementation that follow this above-mentioned design performs better than other considered models.

IV. Dataset Preparation

In this paper, raw dataset was obtained from the CompCars dataset where the one of the usage was for fine-grained classification. CompCars dataset includes two distinct subsets where surveillance-nature set is obtained from surveillance cameras and web-nature set is obtained from public websites and search engines. As the main used subset, web-nature subset contains 163 car makes with 1716 car models which in total includes 136726 distinct images. Characteristic of the images can be defined as it includes the cars entirely where the pictures taken from different viewpoints and with different backgrounds.

For the car model classification task, make and models are combined where each directory collects the corresponding car make and car model without separation of the year that they produced. Dataset is also processed selectively by the CustomCarDataset script which includes only the selected models that has 79 images that is derived from the raw dataset as the mean of class distributions. As a result, used subset includes 927 distinct make-model classes which has 111078 images in total. Subset is divided into 70% (77754 images) for training, 15% for validating (16662 images) and 15% for testing (16662 images).

Since the the raw label files only include the viewpoint, bounding-box count and pixel coordinates of the bounding boxes, new labels were generated from the make/model folder names which is indexed for consistency to enable the model to make the classification accordingly. Label extraction and label mapping operation also obtained with the CustomCarDataset script.

Additionally, due to existence of class imbalance within the stratified sets, weighted sampling mechanism was applied to the training set during the dataloader phase. Weights of each class has been calculated and dataloader handled the class distribution among the batches accordingly.

V. Learning Framework

Before going in the details about the learning framework, training setup must be explained beforehand that is considered for increasing the performances. If we list the considered setup before defining the model:

- Focal Loss definition as the main loss function where the alpha calculated according to the class distributions.
- Adam optimizer has been used for optimizing the model parameters considering various learning rates and weight decay rates. It is crucial for avoiding the model from over-fitting during training and also for generalizing the train set where for performing better on validation and test sets.
- Event scheduler has been used during training as ReduceLROnPlateau which enables the model to start with higher learning rates that makes the model learn faster in early epochs where it changes the learning rate if the validation loss and accuracy is not changing anymore. In this case, it decreases the learning rate to avoid the model overfit on the training set where enables the generalization.
- Early stopping has been applied during training considering the model performance on the validation set. If the model results in increase in validation loss 3 times in a row training stops since it may result in overfit in the model.

As one of the setup considerations,the choice of loss function was handling the class imbalance inside the training set which may resulted in bias of the model towards the over-sampled classes. Even though oversampling methods to the under-sampled classes and under-sampling method to the over-sampled classes were considered, instead of changing the size of the dataset and augmenting artificial data more complex loss functions for handling the class imbalance such as cross-entropy and focal loss were considered. Since focal loss implementation is also based on cross-entropy loss, focal loss was chosen as the loss function. Focal loss has been calculated from this formula:

$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t) \tag{1}$$

where:

- p_t probability estimation for each class obtained from the model.
- α_t factor the that adjusting weighting to the classes.
- γ parameter that adjust the rate of down-weighting

Most common value of the alpha is 1 where it indicates giving the same importance to all classes. Since the class distributions are different and alpha has been calculated according to the class counts where it penalizes the higher counted classes and rewards the lower counted classes.

- **Stage 1:** As shown in the first table, first convolutional layer takes the raw image with dimensions (224x224x3) and output a feature map with dimensions (112x112x64). Layer uses 64 filters with a 7x7 kernel size and a stride of 2. After, batch normalization layer is normalized the activation where the ReLu activation function creates non-linearity. As a last step, max pooling layer with a pool 3x3 and stride 2 applied for down-sampling the feature map in both dimensions. As a result, output is 56x56x64 feature map which is an input for the following convolutional blocks.
- **Stage 2:** As shown in the second and third table, this stage includes one convolutional block followed by the 2 identity blocks where it introduces convolution, batch normalization and ReLu activations subsequently to apply more filters in various amounts and sizes, and extract more features for the input. Convolution block reduces the dimension by half and increases the channel size to 256. Where the identity blocks do not change the dimension and channel count, it only creates a residual connection to the main path.
- **Stage 3:** As shown in the fourth and fifth table, this stage includes one convolutional block followed by the 3 identity blocks where it introduces convolution, batch normalization and ReLu activations subsequently to apply more filters in various amounts and sizes, and extract more features for the input. Convolution block reduces the dimension by half and increases the channel size to 512. Where the identity blocks do not change the dimension and channel count, it only creates a residual connection to the main path.
- **Stage 4:** This stage includes one convolutional block followed by the 4 identity blocks where it introduces convolution, batch normalization and ReLu activations subsequently to apply more filters in various amounts and sizes, and extract more features for the input. Convolution

block reduces the dimension by half and increases the channel size to 1024. Where the identity blocks do not change the dimension and channel count, it only creates a residual connection to the main path.

- **Stage 5:** This stage includes one convolutional block followed by the 2 identity blocks where it introduces convolution, batch normalization and ReLu activations subsequently to apply more filters in various amounts and sizes, and extract more features for the input. Convolution block reduces the dimension by half and increases the channel size to 2048. Where the identity blocks do not change the dimension and channel count, it only creates a residual connection to the main path.
- **Ending Layers:** This stage includes Average Pooling Layer, Flatten Layer, Dropout and Fully Collected Linear Layer. Average Pooling Layer averages the values in all dimension and produces a tensor 2048x1x1. Flatten layer converts the tensor into 1D tensor 64x2048. Dropout is for regularization that randomly assigns 0 to some values during training. And finally, Linear Fully Connected Layer to map the vector of 2048 dimensions into 927 classes.

If we go in the details the learning architectures it can be described as follows:

Layer Type / Stride	Filter Shape	Padding	Input Size	Output Size
Conv2d / Stride 2	7x7, 64 filters	3	224x224x3	112x112x64
BatchNorm2d	-	-	112x112x64	112x112x64
ReLU	-	-	112x112x64	112x112x64
MaxPool2d / Stride 2	3x3	1	112x112x64	56x56x64

TABLE 1: ResNet50 Initial Layer Details

Layer Type / Stride	Filter Shape	Padding	Input Size	Output Size
Conv2d / Stride 2	1x1, 64 filters	0	56x56x64	28x28x64
BatchNorm2d	-	-	28x28x64	28x28x64
ReLU	-	-	28x28x64	28x28x64
Conv2d / Stride 1	3x3, 64 filters	1	28x28x64	28x28x64
BatchNorm2d	-	-	28x28x64	28x28x64
ReLU	-	-	28x28x64	28x28x64
Conv2d / Stride 1	1x1, 256 filters	0	28x28x64	28x28x256
BatchNorm2d	-	-	28x28x256	28x28x256
Conv2d (shortcut) / Stride 2	1x1, 256 filters	0	56x56x64	28x28x256
ReLU	-	-	28x28x256	28x28x256

TABLE 2: ResNet50 Convolutional Block 1

Layer Type / Stride	Filter Shape	Padding	Input Size	Output Size
Conv2d / Stride 1	1x1, 64 filters	0	28x28x256	28x28x64
BatchNorm2d	-	-	28x28x64	28x28x64
ReLU	-	-	28x28x64	28x28x64
Conv2d / Stride 1	3x3, 64 filters	1	28x28x64	28x28x64
BatchNorm2d	-	-	28x28x64	28x28x64
ReLU	-	-	28x28x64	28x28x64
Conv2d / Stride 1	1x1, 256 filters	0	28x28x64	28x28x256
BatchNorm2d	-	-	28x28x256	28x28x256
ReLU	-	-	28x28x256	28x28x256

TABLE 3: ResNet50 Identity Block (x2)

Layer Type / Stride	Filter Shape	Padding	Input Size	Output Size
Conv2d / Stride 2	1x1, 128 filters	0	28x28x256	14x14x128
BatchNorm2d	-	-	14x14x128	14x14x128
ReLU	-	-	14x14x128	14x14x128
Conv2d / Stride 1	3x3, 128 filters	1	14x14x128	14x14x128
BatchNorm2d	-	-	14x14x128	14x14x128
ReLU	-	-	14x14x128	14x14x128
Conv2d / Stride 1	1x1, 512 filters	0	14x14x128	14x14x512
BatchNorm2d	-	-	14x14x512	14x14x512
Conv2d (shortcut) / Stride 2	1x1, 512 filters	0	28x28x256	14x14x512
ReLU	-	-	14x14x512	14x14x512

TABLE 4: ResNet50 Convolutional Block 2

Layer Type / Stride	Filter Shape	Padding	Input Size	Output Size
Conv2d / Stride 1	1x1, 128 filters	0	14x14x512	14x14x128
BatchNorm2d	-	-	14x14x128	14x14x128
ReLU	-	-	14x14x128	14x14x128
Conv2d / Stride 1	3x3, 128 filters	1	14x14x128	14x14x128
BatchNorm2d	-	-	14x14x128	14x14x128
ReLU	-	-	14x14x128	14x14x128
Conv2d / Stride 1	1x1, 512 filters	0	14x14x128	14x14x512
BatchNorm2d	-	-	14x14x512	14x14x512
ReLU	-	-	14x14x512	14x14x512

TABLE 5: ResNet50 Identity Block (x3)

VI. Results

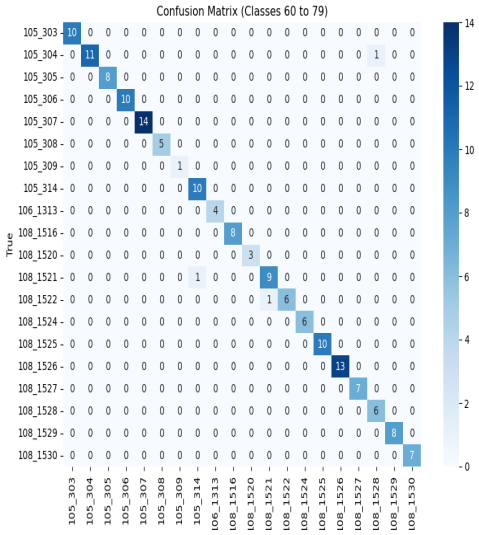

Fig. 2: Confusion Matrix Output on The Test Set

When the performance of the above mentioned ResNet-50 architecture is analysed, it is obtained that even though the expected accuracy on the validation set was identified low, prediction of the class labels on both validation and test sets seems promising. As shown in the Fig.2, there is a trend in the diagonal of the confusion matrix on test dataset even if there are some outliers existed. It means that model is not performing in a bad condition however it indicates that it needs improvements since there are misclassified outliers.

Epoch	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy
1	341.0957	0.0000	1.1997	0.0028
2	280.5049	0.0172	1.1254	0.0142
3	278.2763	0.0345	1.0417	0.0362
4	231.1740	0.0690	0.9274	0.0669
5	170.1077	0.2241	0.8337	0.1152
6	151.2827	0.3276	0.7093	0.1934
7	124.2810	0.3448	0.5943	0.2715
8	79.4760	0.5517	0.5509	0.3142
9	54.1260	0.6379	0.5137	0.3561
10	29.8233	0.7586	0.4666	0.3936
11	20.5368	0.7931	0.4305	0.4325
12	18.3052	0.8621	0.4289	0.4280
13	24.8939	0.7931	0.4152	0.4475
14	12.9198	0.8793	0.4037	0.4643
15	4.0602	0.9310	0.3906	0.4851
16	21.3736	0.8448	0.3877	0.4888
17	7.7213	0.8966	0.3861	0.4966
18	9.3884	0.8966	0.3791	0.4998
19	7.8192	0.9310	0.3818	0.5044
20	8.2801	0.8966	0.4042	0.4869
21	10.1433	0.9310	0.3667	0.5176
22	6.3482	0.9138	0.3835	0.5085
23	7.0078	0.9138	0.3800	0.5184
24	13.1792	0.9310	0.3904	0.5061

TABLE 6: Training and Validation Metrics per Epoch

Even though there are some fluctuations during training in the training loss and validation losses, with the 50% validation accuracy model performed well on the test set where the outliers for classes were in small amount. However, it is understood that deeper residual networks should be preferred for increasing the performance in training. But, since the computational cost is really high for the implementation of the deeper ResNet architecture it was not considered beforehand. Also, as another consideration to handle the class imbalance it is suggested that using augmented and balanced dataset will perform better for this model or more precautions must be considered for handling the class-imbalance during training. Since the distributions of the image counts belonging to the classes in each set differs, handling the class-imbalance problem inside the batch distribution with weighted sampler or using loss function that considers class-imbalance as focal loss may be not performing that much well than it is expected.

Learning Rate	Dropout Rate	Weight Decay
1e-4	0.2	1e-5
1e-4	0.3	1e-5
1e-4	0.4	1e-5
1e-2	0.2	1e-5
1e-2	0.3	1e-5
1e-2	0.4	1e-5
1e-3	0.2	1e-5
1e-3	0.3	1e-5
1e-3	0.4	1e-5

TABLE 7: Training Metrics for Various Learning Rates and Dropout Rates

Since the model was really sensitive, different metrics were considered in separate setups where each run started with 100 epochs and ended according to the early stopping criteria. Best model for generalization and not over-fitting on the training data was the setup where the learning rate was 1e-3, dropout rate was 0.2 and weight decay was 1e-5, batch size was 64 and number of epochs was 100. Increasing the dropout rate showed that is beneficial for regularization of the model which reduces the overfit problem and starting from higher learning rates that will be lowered later on by the learning rate scheduler is also beneficial approach for enabling the model learn well in the first epochs and avoiding overfit according to the performance of the model on the validation set.

VII. Concluding Remarks

As a conclusion, for fine-grained car model classification task with using CompCars dataset, various ResNet setup were analyzed in accordance with their performances on identifying the car models. Due to severe occurrence of class imbalance, interchanging viewpoint characteristics of the images and lack of the required amount of images for some classes it is understood that dataset preparation phase must be implemented more precisely. Even if the weighted sampling method, data transformations and focal loss implementation solved the class imbalance at some point, for better accuracy on test and validation datasets there has to be more balanced dataset. Despite the mentioned need, trained ResNet-50 model generalizes the training dataset and performs well on the validation and test sets. The only future work can be

application of more deep Residual Networks and preparation of a balanced dataset for the classification task purposes.

As a paragraph that is dedicated for my own experience, I understood that training a well generalized model takes a lot of time, patience, dedication and resources since in order to obtain a better model you have to fine-tune the model over and over again . Understanding the architecture, time that takes for training the model and handling the class imbalance were the main difficulties for this project. Even if the output model is not resulted in perfect from now on I know where should I start.

VIII. References

[1] L. Yang, P. Luo, C. C. Loy and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 3973-3981, doi: 10.1109/CVPR.2015.7299023. keywords: Biological system modeling,

[2] Wille, Renan Carnieri, Ricardo Borba, Gustavo Pipa, Daniel. (2018). Classification of vehicle make and model with MobileNets. 10.14209/sbrt.2018.3.