Universidad de Alcalá

PhD. Program in Information and Communications Technologies

Vehicle Keypoint Detection and Fine-Grained Classification using Deep Learning

Author

Héctor Corrales Sánchez

Advisors

Dr. D. David Fernández Llorca

Dr. D. Ignacio Parra Alonso

Alcalá de Henares, 5^{th} of October, 2021

"Como no estás experimentado en las cosas del mundo, todas las cosas que tienen algo de dificultad te parecen imposibles" Don Quijote de la Mancha

Agradecimientos

Parece que no haya pasado prácticamente nada desde que empecé esta aventura casi casi de rebote. Era septiembre del año 2017. Han pasado ya la friolera de 4 años. David y Nacho mandaron un correo a los que estábamos en el máster, a ver si liaban a alguien para comenzar en el laboratorio con vistas a seguir con la tésis. Yo mordí el anzuelo.

En primer lugar, gracias a mis tutores, los doctores D. David Fernández Llorca y D. Ignacio Parra Alonso. Me brindasteis la oportunidad de emprender esta aventura y me habéis acompañado y guiado en ella. Sin vosotros esto no habría sido posible.

En segundo lugar, no me podía olvidar de los tutores de trinchera: LSD, JLo, MA y AQP. Vuestro apoyo diario ha sido fundamental para llegar a buen puerto pese a los infinitos anexos. Entre todos nos hemos hecho expertos en finanzas, inversiones, criptomonedas, mercado inmobiliario, automatización y solución de problemas que nadie tiene. Esto se nos ha atragantado un poco ¿Verdad, JLo y MA?. AQP, empieza ya a escribir la tesis que con la procrastinación no acabas en la vida, ¡Hulio!

No me olvido de ti Noe. ¿Qué habrían sido las postcomidas sin tus postres y sin hacer el monguer bailoteando? Los anexos de WiFi son culpa tuya, tú tampoco te libras. Gracias por tus consejos y aguantar mis historias. Aunque huiste rápido del laboratorio (normal) y dejaste de consumir agua de fregar, perdón, café, ya me ocupé yo de subir a darte la brasa y tu de dejarme la puerta abierta.

Gracias Augusto, me has echado más de un cable, debatiendo propuestas imposibles, andando por el monte, comiendo y desayunando, si no me falla la memoria te debo uno.

A los nuevos, Antoine, Sandra, Laura y Sergio. ¡Mucha suerte y ánimo! Espero que atormentéis adecuadamente al pollito mayor (AQP).

Gracias Nieves. Hemos compartido muchísimas experiencias juntos. Hemos viajado. Nos hemos quedado atrapados en la otra punta del mundo, pero siempre juntos. Has sido prácticamente una más en el laboratorio y has aguantado mis altibajos. Gracias. Parece que este año va a ser un buen año, tésis mediante por mi parte, tú porfín con tu ansiada plaza de residente. Por muchos años más, creciendo y viviendo aventuras. Te quiero.

Mamá, Papá, gracias. Me habéis apoyado, enseñado y aconsejado. Sin vosotros, animandome en mis buenas ideas y deteniendome en las malas, no habría llegado hasta aquí. Esta tesis tambien es vuestra.

Resumen

Los sistemas de detección de puntos clave en vehículos y de clasificación por marca y modelo han visto como sus capacidades evolucionaban a un ritmo nunca antes visto, pasando de rendimientos pobres a resultados increíbles en cuestión de unos años. La irrupción de las redes neuronales convolucionales y la disponibilidad de datos y sistemas de procesamiento cada vez más potentes han permitido que, mediante el uso de modelos cada vez más complejos, estos y muchos otros problemas sean afrontados y resueltos con enfoques muy diversos.

Esta tesis se centra en el problema de detección de puntos clave y clasificación a nivel de marca y modelo de vehículos con un enfoque basado en aprendizaje profundo. Tras el análisis de los conjuntos datos existentes para afrontar ambas tareas se ha optado por crear tres bases de datos específicas. La primera, orientada a la detección de puntos clave en vehículos, es una mejora y extensión del famoso conjunto de datos PASCAL3D+, reetiquetando parte del mismo y añadiendo nuevos keypoints e imágenes para aportar mayor variabilidad. La segunda, se trata de un conjunto de prueba de clasificación de vehículos por marca y modelo basado en *The PREVENTION dataset*, una base de datos de predicción de trayectoria de vehículos en entornos de circulación real. Por último, un conjunto de datos cruzados (*Cross-dataset*) compuesto por las marcas y modelos comunes de tres de las principales bases de datos de clasificación de vehículos, CompCars, VMMR-db y Frontal-103.

El sistema de detección de puntos clave se basa en un método de detección de pose en humanos que mediante el uso de redes neuronales convolucionales y capas de-convolucionales genera, a partir de una imagen de entrada, un mapa de calor por cada punto clave. La red ha sido modificada para ajustarse al problema de detección de puntos clave en vehículos obteniendo resultados que mejoran el estado del arte sin hacer uso de complejas arquitecturas o metodologías. Adicionalmente se ha analizado la idoneidad de los puntos clave de PASCAL3D+, validando la propuesta de nuevos puntos clave como una mejor alternativa.

El sistema de clasificación de vehículos por marca y modelo se basa en el uso de redes preentrenadas en el famoso conjunto de datos ImageNet y adaptadas al problema de clasificación de vehículos. Uno de los problemas detectados en el estado del arte es la saturación de los resultados en las bases de datos existentes que, por otra parte, se encuentran sesgadas, limitando la capacidad de generalización de los modelos entrenados con ellas. Se han usado múltiples técnicas de aprendizaje y ponderación de los datos para tratar de aliviar el impacto del sesgo de los conjuntos de datos. Para poder evaluar la capacidad de generalización en situaciones reales de los modelos entrenados, se ha hecho uso del conjunto de pruebas derivado del *PREVENTION dataset*. Adicionalmente, se ha hecho uso del *Cross-dataset* para evaluar la complejidad de las bases de datos existentes y las capacidades de generalización de los modelos entrenados con ellas. Se demuestra que, sin hacer uso de complejas arquitecturas, se pueden obtener resultados competitivos y la necesidad de un conjunto de datos que refleje de manera adecuada el mundo real para poder afrontar adecuadamente el problema de clasificación de vehículos.

Palabras clave: Detección de Puntos Clave en Vehículos, Clasificación de Marca y Modelo de Vehículos, Aprendizaje Profundo, Conjunto de Datos Cruzados.

Abstract

Vehicle keypoint detection and fine-grained classification systems have seen their capabilities evolve at an unprecedented rate, from poor performance to incredible results in a matter of a few years. The advent of convolutional neural networks and the availability of large amounts of data and progress in computational capabilities have allowed these and many other problems to be tackled and solved with very different approaches using increasingly complex models.

This thesis focuses on the problems of keypoint detection and fine-grained classification of vehicles with a deep learning approach. After the analysis of the existing datasets to tackle both tasks, three new datasets have been built. The first one, oriented to the detection of keypoints in vehicles, is an improvement and extension of the famous PASCAL3D+ dataset, re-labelling part of it and adding new keypoints and images to provide more variability. The second is a vehicle make and model classification test set based on the PREVENTION dataset, a real-world driving scenario vehicle trajectory prediction dataset. Finally, a cross-dataset composed of common makes and models from three major vehicle classification databases, CompCars, VMMR-db and Frontal-103.

The keypoint detection system is based on a human pose detection method that by using convolutional neural networks and deconvolutional layers generates, from an input image, a heat map for each keypoint. The network has been modified to fit the problem of keypoint detection in vehicles obtaining results that improve the state of the art without using complex architectures or methodologies. Additionally, the suitability of the PASCAL3D+ keypoints has been analysed, validating the proposal of new keypoints as a better alternative.

The vehicle make and model classification system is based on the use of ImageNet pre-trained networks and fine-tuned for the vehicle classification problem. One of the problems detected in the state of the art is the saturation of the results in the existing datasets, which, moreover, are biased, limiting the generalisation capacity of the models trained with them. Multiple data learning and weighting techniques have been used to try to alleviate the impact of dataset bias. In order to assess the generalisation capabilities of the trained models in real situations, the PREVENTION test set has been used. Additionally, the cross-dataset has been used to evaluate the complexity of the existing datasets and the generalisation capabilities of the models trained with them. It is shown that competitive results can be achieved without the use of complex architectures and that a high quality dataset that adequately reflects the real world is needed in order to properly address the vehicle classification problem.

KeyWords: Vehicle Keypoint Detection, Fine-Grained Vehicle Classification, Deep Learning, Cross-dataset.

Table of Contents

Resumen **I**

Abstract **III**

Table of Contents **V**

List of Figures **VII**

List of Tables **XI**

List of Acronyms **XIII**

1 Introduction **1**
1.1 Motivation . . . 2
1.2 Problem Description . . . 4
1.2.1 Vehicle Keypoint Detection . . . 4
1.2.2 Fine-grained Vehicle Classification . . . 5
1.3 Document Outline . . . 6

2 State of the Art **9**
2.1 Existing Datasets . . . 9
2.1.1 Vehicle Keypoints Datasets . . . 9
2.1.2 Fine-grained Vehicle Classification Datasets . . . 13
2.2 Vehicle Keypoint Detection and Fine-grained Vehicle Classification . . . 16
2.2.1 Vehicle Keypoint Detection . . . 16
2.2.2 Fine-grained Vehicle Classification . . . 20
2.2.2.1 Pre-Convolutional Neural Networks (CNNs) methods . . . 21
2.2.2.2 CNNs era methods . . . 21
2.3 Conclusions . . . 25
2.4 Main Contributions . . . 25

3 Vehicle Keypoint Detection **27**
3.1 Proposed Method . . . 27
3.1.1 The Architecture . . . 28
3.1.2 Metrics, Data Augmentation and Dataset . . . 28
3.1.2.1 Metrics . . . 29
3.1.2.2 Data Augmentation . . . 29
3.1.2.3 Dataset . . . 29
3.2 Custom Keypoints Dataset . . . 30
3.3 Experiments . . . 31

4 Fine-grained Vehicle Classification **33**
4.1 Proposed Method . . . 33
4.1.1 The Architectures . . . 33
4.1.2 Metrics, Data Augmentation and Datasets . . . 34
4.1.2.1 Metrics . . . 34
4.1.2.2 Data Augmentation . . . 35
4.1.2.3 Datasets . . . 36
4.1.3 Learning Techniques . . . 37
4.1.3.1 Curriculum Learning . . . 37
4.1.3.2 Weighted Losses . . . 37
4.2 The PREVENTION test set and Cross-Dataset . . . 38
4.2.1 The PREVENTION test set . . . 39
4.2.2 Cross-Dataset . . . 40
4.3 Experiments . . . 40

5 Results **43**
5.1 Vehicle Keypoint Detection . . . 43
5.1.1 Data Augmentation . . . 43
5.1.2 Backbone and Input Size . . . 44
5.1.3 PASCAL3D+ . . . 45
5.1.4 Instance Size . . . 47
5.1.5 Keypoint Distribution Study . . . 48
5.1.6 Custom Keypoints . . . 50
5.2 Fine-grained Vehicle Classification . . . 54
5.2.1 Curriculum Learning . . . 54
5.2.1.1 Incremental Learning . . . 54
5.2.1.2 Progressive Learning . . . 56
5.2.2 Fine-grained Models . . . 59
5.2.3 Weighted Losses . . . 60
5.2.3.1 Why raw precision is not enough? . . . 60
5.2.3.2 Weighted losses for maker classification . . . 62
5.2.3.3 Weighted losses for model classification . . . 64
5.2.4 Complexity and Generalisation Capabilities . . . 65
5.3 Conclusions . . . 69
5.3.1 Vehicle Keypoint Detection . . . 69
5.3.2 Fine-grained Vehicle Classification . . . 70

6 Conclusions and Future Work **73**
6.1 Conclusions . . . 73
6.2 Contributions . . . 74
6.3 Future work . . . 75

Appendices **77**

A Further works derived from this thesis **79**
A.1 License Plate Localisation with Keypoints . . . 79
A.2 Keypoint enhanced Fine-grained Vehicle Classification . . . 81

B Publications Derived from this PhD Dissertation **85**
B.1 Journal Publications . . . 85
B.2 Conference Publications . . . 85

Bibliography **87**

List of Figures

1.1 Prediction of distribution of off-street and on-street parking spaces in European Parking Association (EPA) municipalities with more than 20,000 inhabitants. . . 3
1.2 Visual example of license plate swapping in a car park. Top shows a classic License Plate Recognition (LPR) system that does not detect the license plate swap. Bottom shows an enhanced LPR capable of detecting the license plate swap preventing the attempted theft. 3
1.3 Keypoint variability examples from VeRi776+ and ApolloCar3D datasets with 20 and 66 keypoints respectively. Images extracted from [1] and [2]. 5
1.4 Visual examples of the three main fine-grained vehicle classification problems. Top-left the *multiplicity* problem, with two Audi A4 and two KIA Sportage from different generations, it can be seen that even though they are the same model there are visual differences. Top-right the *ambiguity* problem, with BMW X3 and X5 and Volkswagen CC and Passat, it can be seen that while they are different models, they share visual appearance. Bottom the *bias* problem, with an example of a high quality image and a render, both far from real world, and two examples of vehicles only available in a specific region, in this case China. 6

2.1 Example of some of the images from PASCAL3D+ dataset. In green the bounding boxes and in red the keypoints. Top row images are from PASCAL subset and bottom ones from ImageNet subset. 10
2.2 ObjectNet3D example. On the left 3D pose annotations and on the right 3D shape annotation. 10
2.3 Example of some of the images from VeRi-776 dataset with its keypoints in red. 11
2.4 Example of some of the images from Car Renders dataset. 11
2.5 Example of some of the images from CarFusion dataset. In green the bounding boxes, in red the visible keypoints and in yellow the non visible keypoints. 12
2.6 Example of some of the images from ApolloCar3D dataset with the keypoints in red. 12
2.7 Example of some of the images from Cars-196 dataset. 13
2.8 Example of some of the images from CompCars dataset. 14
2.9 Example of some of the images from VMMRdb dataset. 15
2.10 Example of some of the images from Frontal-103 dataset. 15
2.11 Visual comparison of top-down and bottom-up keypoints detection methods. The first one is a top-down method. First all instances are detected and then the keypoints for each one are predicted individually. The second one is a bottom-up method. First all keypoints in the image are detected an then matched to reconstruct the instances. 17
2.12 Visual example of a stacked hourglass network (top) and an hourglass residual module (bottom). Extracted from [3]. 18
2.13 Overview of the two-stage 3D pose vehicle estimation framework presented by Ding et al. Extracted from [4]. 19

2.14 Visual example of the 3D bounding box used by Sochor et al. [5] to *unpack* vehicle shapes and orientation. 23

3.1 Aspect ratio enforcement examples. Extended bounding box on the left and black banded on the right. 28
3.2 Illustration of the full simple baseline network architecture pipeline proposed for vehicle keypoint detection. 28
3.3 Visual examples of the different data augmentation operations. On top, from left to right, no data augmentation, mild data augmentation (with horizontal flip, rotation and scaling), salt-and-pepper noise and poisson noise. On the bottom, from left to right, speckle noise, blurring, colour casting, and colour jittering. The hard data augmentation specific operations have been performed over the original image without the mild data augmentation to ease visualisation. 30
3.4 Example of some of the images from PASCAL3D+. In green the bounding boxes and in red the keypoints. Top row images are from PASCAL and bottom ones from ImageNet. 31
3.5 Visual comparison of PASCAL3D+ keypoints and our custom keypoints. Top row/green PASCAL3D+, bottom row/yellow custom keypoints. 32

4.1 ImageNet-1k Top-1 accuracy vs. computational cost for a single forward pass. The size of each ball corresponds to the model complexity. Figure extracted from [6]. 34
4.2 Example of precision and recall calculation. In green correct predictions and in red incorrect predictions. The first row (squared in green) is used to compute car precision. The first column (squared in yellow) is used to compute car recall. . . 35
4.3 Visual examples of the different data augmentation operations. On top, from left to right, no data augmentation, horizontal flip, salt-and-pepper noise and poisson noise. On bottom, from left to right, speckle noise, blurring, colour casting, and colour jittering. 36
4.4 On the left, standard and logarithmic weights for a given class according to its percentage of representation in the dataset. On the right, Focal Loss effect on loss depending on the γ value used. 39
4.5 Example of some of the images from the PREVENTION test set. 39

5.1 Visual comparison of the input size and output heatmaps for ResNet50 256x192 and ResNet152 384x288 models. On top the input images, in the middle the ResNet50 256x192 output heatmaps and on the bottom the ResNet152 384x288 output heatmaps. 45
5.2 Instances area distribution in square pixels for each subset of PASCAL3D+ without the outliers. 46
5.3 Output samples of Simple Baseline for Vehicle Pose Estimation (SBVPE)-PASCAL3D+/PASCAL model. 47
5.4 Amount of each keypoint on PASCAL3D+ dataset and its subsets. From left to right: front left wheel, rear left wheel, front right wheel, rear right wheel, top left windshield, top right windshield, top left rear window, top right rear window, front left light, front right light, left trunk and right trunk. 49
5.5 Per-keypoint Percentage of Correct Keypoints (PCK) (left) and Average Precision of Keypoints (APK) (right) with $\alpha = 0.1$ for PASCAL and PASCAL3D+. Keypoints are, from left to right: front left wheel, rear left wheel, front right wheel, rear right wheel, top left windshield, top right windshield, top left rear window, top right rear window, front left light, front right light, left trunk and right trunk. 50

5.6 Output heatmaps for SBVPE-PASCAL3D+/PASCAL. Examples of the keypoint confusion phenomenon for the wheels (first and second row) and the windshield-/rear window (third and fourth rows). For the first and second rows the second and third columns show the correct wheels prediction while the fourth and fifth columns show incorrectly predicted ones. For the third row the second and third columns show correctly predicted windshield corners while the fourth and fifth columns show incorrectly predicted rear window corners. For the fourth row second and third columns show incorrectly predicted windshield corners while the fourth and fifth columns show correctly predicted rear windows ones. 51
5.7 Amount of each keypoint on PASCAL3D+ dataset and our custom dataset. . . . 52
5.8 Per-keypoint PCK (left) and APK (right) with $\alpha = 0.1$ for PASCAL3D+ and the custom keypoints dataset. Keypoints are, from left to right: front wheel, rear wheel, top left windshield, top right windshield, bottom left windshield, bottom right windshield, top left rear window, top right rear window, bottom left rear window, bottom right rear window, left fog light, right fog light, left mirror, right mirror, top left plate, top right plate, bottom left plate, bottom right plate and logo. 53
5.9 Examples of predictions from our custom dataset. Top and bottom row are good and bad predictions respectively. 54
5.10 Per-class performance differences for the ResNet50 standard and *incremental learning* models. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *incremental learning* method. Differences above 0.025 (red squares) mean better performance for the standard model. Values in between (yellow triangles) mean similar performance in both models. 55
5.11 Per-class performance differences for the ResNet50 standard and *incremental learning* models depending on the number of training samples. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *incremental learning* method. Differences above 0.025 (red squares) mean better performance for the standard model. Values in between (yellow triangles) mean similar performance in both models. 56
5.12 Left images are the per-class performance differences of 0.001 and 0.01 learning rate standard ResNet50 and *progressive-10* ResNet50 respectively. Right images are the per-class performance differences for the 0.001 and 0.01 learning rate trains depending on the number of training samples. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *progressive-10* method. Differences above 0.025 (red squares) mean better performance fort the standard model. Values in between (yellow triangles) mean similar performance in both models. 58
5.13 Number of classes with a given precision and recall (VALIDATION) for InceptionV3 model trained with VMMR-db Makers on the left and VMMR-db Models on the right. 61
5.14 Per-class relation of number of samples, precision and recall (VALIDATION) for the InceptionV3 models trained with VMMR-db makers and models. On the left the 43 Makers, on the right the 472 Models. 62
5.15 Comparison of per-class performance (VALIDATION) in VMMR-db Makers for the different weighted models. On the left precision results, on the right recall. . 63
5.16 Comparison of per-class performance (TEST) in the PREVENTION Makers test set for the different weighted models trained with VMMR-db Makers. On the left precision results, on the right recall. 63

5.17 Comparison of per-class performance (VALIDATION) in VMMR-db Models for the different weighted models. On the left precision results, on the right recall. . 65
5.18 Comparison of per-class performance (TEST) in the PREVENTION Models test set for the different weighted models trained with VMMR-db Models. On the left precision results, on the right recall. 65
5.19 Top3 predicted classes of the Fusion-Makers trained model for sample images from the PREVENTION Makers test set. The first row shows correctly classified front view images. The middle row shows correctly classified rear view images. The bottom row shows misclassified images from both views. 67
5.20 Top3 predicted classes of the Fusion-Models trained model for sample images from the PREVENTION Models test set. The first row shows correctly classified front view images. The middle row shows correctly classified rear view images. The bottom row shows misclassified images from both views. 69

A.1 Examples of the car park entrance images used for train and validation. 80
A.2 Examples of the test images. 80
A.3 Example of the different heatmap resolutions. 81
A.4 Examples of license plate corners detection on the validation set (top row) and on the test set (bottom row). Green circles are the ground truth, red circles are the predictions. License plate area has been zoomed to ease visualisation. 81
A.5 Examples of the four masks generated from the keypoints for three different images. 82
A.6 Illustration of the feature extraction module. The first stage extracts a global feature vector and computes four local feature vectors with the masks. The second stage refines these vectors. 83
A.7 Illustration of the orientation-invariant feature aggregation module. First, the local feature vectors are concatenated and four weights computed to ponder them. After this, the weighted local feature vectors and the global one are concatenated and fed to a fully connected layer to predict the make and model. 83

List of Tables

2.1 Summary of the most relevant vehicle keypoints datasets. 13
2.2 Summary of the most relevant fine-grained vehicle classification datasets. 16
2.3 Keypoint detection state of the art summary. Methods with its dataset marked as Human* are exclusively trained and validated for human pose estimation. . . 20
2.4 CNNs era fine-grained vehicle classification state of the art summary. 24

4.1 Fusion sets number of classes, images and distribution of the images between the three source datasets. 40

5.1 PCK and APK with $\alpha = 0.1$ for different data augmentation strategies using the images from the PASCAL subset of PASCAL3D+. 44
5.2 PCK and APK with $\alpha = 0.1$ for different backbones and input sizes using the images from the PASCAL subset of PASCAL3D+ 44
5.3 PCK and APK with $\alpha = 0.1$ for the different subsets of PASCAL3D+. All runs with ResNet152 backbone and input size of 384x288. 45
5.4 PCK and APK with $\alpha = 0.1$ of different methods. SBVPE trained and validated with PASCAL, PASCAL3D+/PASCAL and PASCAL3D+ respectively. All 3 methods are the ResNet152 backbone. 47
5.5 Median and mean instance area in square pixels for PASCAL3D+ dataset and its subsets. 48
5.6 PCK with $\alpha = 0.1$ of different methods. Full results correspond to the complete validation set, occluded to the objects marked as truncated or occluded, small are the smaller third of the data and big the bigger third. SBVPE trained and validated with PASCAL3D+/PASCAL and PASCAL3D+/PASCAL3D+ respectively. Both methods are the ResNet152 backbone with 384x288 input size. . . . 48
5.7 PCK and APK with $\alpha = 0.1$. Comparison of performance between PASCAL3D+ keypoints and our keypoints. All 3 methods are the ResNet152 backbone with 384x288 input size. 52
5.8 CompCars validation accuracy comparison of the standard models and the *incremental learning* ones. Times for the *incremental learning* method models (marked with *) are roughly double their standard counterparts because the time required for training the maker and model networks are taken into account. 55
5.9 Comparison of the accuracy of ResNet50 models trained with standard training and *progressive learning* technique. 57
5.10 Comparison of the accuracy of ResNet50 models trained with the *progressive-10* learning technique and alternative class order (random and inverse) with standard training. 58
5.11 Information of number of classes and images of each of the subsets. 59
5.12 Accuracy comparison of the different datasets and subsets with its baseline results. 60
5.13 Extended metrics for VMMR-db Makers and Models. 61

5.14 Comparison of the accuracy, precision and recall of InceptionV3 models trained with VMMR-db Makers with and without weights and test on the PREVENTION Makers dataset. Standard 43 and Log 43 are the normalised version of the weights. The results of focal loss are the ones obtained using $\alpha = 1, \gamma = 2$. 62
5.15 Comparison of the accuracy, precision and recall of InceptionV3 models trained with VMMR-db Models with and without weights and test on the PREVENTION Models dataset. Standard 472 and Log 472 are the normalised version of the weights. The results of focal loss are the ones obtained using $\alpha = 1, \gamma = 2$. 64
5.16 Fusion sets number of classes, images and distribution of the images between the three source datasets. 66
5.17 Cross-Test performance comparison of Fusion-Makers dataset and its source makers datasets. 66
5.18 PREVENTION Makers test accuracy, precision and recall comparison of Fusion-Makers and its source datasets trained models. From the 27 classes there are 23 common with the PREVENTION Makers test set. 67
5.19 Cross-Test performance comparison of Fusion-Models dataset and its source models datasets. 68
5.20 PREVENTION Models test accuracy, precision and recall comparison of Fusion-Models and its source datasets trained models. From the 75 classes there are 34 common with the PREVENTION Models test set. 68

A.1 Effect of dilated convolutions on the output heatmap resolution for an input size of 640×480. 80
A.2 Average distance error obtained with 0, 1, 2 and 3 dilated convolutions in validation images. 80

List of Acronyms

6DoF	Six Degrees of Freedom.
AI	Artificial Intelligence.
ALPR	Automated License Plate Recognition.
AP-CNN	Attention Pyramid Convolutional Neural Network.
APK	Average Precision of Keypoints.
CAD	Computer-Aided Design.
CAF	Composite Association Field.
CIF	Composite Intensity Field.
CNN	Convolutional Neural Network.
CPM	Convolutional Pose Machine.
CPO	Cross Projection Optimization.
CRF	Conditional Random Field.
DNN	Deep Neural Network.
DPM	Deformable Part Models.
DSNT	Differentiable Spatial to Numeric Transform.
DVAN	Diversified Visual Attention Network.
EPA	European Parking Association.
EU	European Union.
eurostat	European Statistical Office.
FBI	Federal Bureau of Investigation.
FPN	Feature Pyramid Network.
GAN	Generative Adversarial Network.
GPU	Graphics Processing Unit.
HSV	Hue, Saturation, Value.
ITS	Intelligent Transportation Systems.
LPR	License Plate Recognition.
LSTM	Long Short-Term Memory.
MSE	Mean Squared Error.
PAF	Part Affinity Field.

PCA	Principal Component Analysis.
PCK	Percentage of Correct Keypoints.
PIF	Part Intensity Field.
R-CNN	Region Based Convolutional Neural Network.
RGB	Red, Green and Blue.
ROI	Region of Interest.
SBVPE	Simple Baseline for Vehicle Pose Estimation.
SGD	Stochastic Gradient Descent.
SVM	Support Vector Machines.
SWP	Spatially Weighted Pooling.
TFLOPS	Tera Floating Point Operations per Second.
USA	United States of America.
YOLO	You Only Look Once.

Chapter 1

Introduction

Intelligent Transportation Systems (ITS) are a group of advanced technologies that aim to make transport safer, more efficient and more sustainable by applying various information and communication technologies. In recent years, these technologies have experienced an explosion of development with an exponential growth, mainly driven by the use of *Convolutional Neural Networks (CNNs)* boosted by the development of *Graphics Processing Units (GPUs)* and the availability of large amounts of data. One clear indicator of this growth is the evolution in the GPUs market. In 2019, the global GPU market was valued at $19.75 billion, and is projected to reach $200.85 billion by 2027. Focusing on the last 10 years, the amount of transistors and computational power from a high-end GPU has gone from the 3 billion transistors and 1.58 *Tera Floating Point Operations per Second (TFLOPS)* of the GTX 580 to 28.3 billion transistors and 35.58 TFLOPS of the RTX 3090. This is 9 times more transistors and 22 times more computational power. Along with this incredible evolution in computational power, specific systems have appeared. Like the NVIDIA Drive platform, whose aim is the development of autonomous driving. NVIDIA DGX, an *Artificial Intelligence (AI)* data center aimed to large scale deep learning trainings. And NVIDIA Jetson, a series of low-power embedded computing boards designed for the development of portable machine learning applications. All these developments have contributed to the emergence and improvement of multiple applications. From semantic segmentation to image classification, through pose estimation, trajectory and intention prediction, sensor fusion, scene understanding or mapping.

Among all these applications, this thesis focuses on vehicle keypoint detection and fine-grained vehicle classification. On the one hand, vehicle keypoint detection consists in the detection and location of some previously defined key parts known as keypoints from which it is possible to extract information like vehicle orientation, distance or perspective. Vehicle keypoint detection and pose estimation is an important task, with multiple applications in a variety of domains like traffic surveillance or autonomous vehicles. Pose estimation requires several considerations and varies from one object or entity to another. Of all the pose estimation applications, the most widely researched is that of humans. Human pose estimation is one of the most challenging and complex pose estimation task, due to the large number of poses that the human body can adopt given its high degree of flexibility. Driven by advances in human pose, vehicle keypoint detection has greatly benefited from these developments. Traditionally, human pose estimation is defined as the localisation of human keypoints or joints. However, in the vehicles case there is no consensus. In this thesis, when we use vehicle pose, we refer to the 2D pose characterised by its keypoints, the same as in humans. Keypoint use has been widely explored both in humans and vehicles. For vehicles, its importance has been growing due to the increasing number of potential applications. These applications range from giving structural support to improve fine-grained classification, this is, classifying vehicles by make and model, to help enhancing instance segmentation like in [7]. Multiple ITS applications can benefit from the use of keypoints, like improving traffic surveillance [8], vehicle re-identification [1], vision-based vehicle

speed detection [9] or *Automated License Plate Recognition (ALPR)* using the four corners as keypoints [10].

On the other hand, fine-grained vehicle classification consists in the classification of vehicles by make and model, even going so far as to differentiate between different generations of a particular model, which is called ultra fine-grained classification. Fine-grained vehicle classification is a very useful ITS application, especially in the field of traffic surveillance which can be combined with other applications such as license plate recognition systems or keypoint detection methods. In this way, we are able, for example, to detect if a vehicle is driving with a fake number plate by matching the license plate registered vehicle with the predicted one, or detect an attempted theft in a car park by swapping license plates. Fine-grained vehicle classification is also useful for vehicle re-identification, task that consists in detecting an recognising the same vehicle in different places, for example, in two consecutive traffic surveillance cameras. Knowing the make and model of a vehicle, eases this re-identification process. One might think that by using license plate recognition systems, fine-grained classification is not necessary, but, as previously stated, this approach on its own is vulnerable to license swap, license plate detection errors, and, additionally, license plate information is not always available. Fine-grained vehicle classification and vehicle keypoint detection tasks complement each other perfectly and their joint use provides a large amount of information. For example, once the vehicle keypoints, make and model are known, a specific *Computer-Aided Design (CAD)* model can be projected using camera parameters and the keypoints as anchors obtaining precise information of distance, orientation, size or perspective in 2D images as in [2, 4, 11, 12]. For all these reasons, a robust system, capable of efficiently classify vehicle make and model is extremely useful.

To address these challenges, a state of the art investigation of both tasks is carried out and existing architectures and techniques are investigated to achieve models with better learning capabilities and enhanced performance. In addition, existing datasets will be analysed to tackle both tasks and two new improved datasets will be proposed.

1.1 Motivation

The future of mobility is connected, distributed and autonomous. The use of vehicle pose estimation and fine-grained vehicle classification systems, either in a joint way or separate will be of great use in this upcoming scenario. There is a growing need for robust systems that are capable of identifying vehicles regardless of the domain (traffic surveillance, speed cameras, access control, infrastructure use monitoring, autonomous vehicles, police vehicles, etc.).

Properly identifying a vehicle is of great value, since in this way, we get as much detail as a person might give, like for example maker, model, year, color and license plate. This information can be used to detect fake or swapped license plates, vehicle re-identification, improved access control to restricted areas (environmental certificate verification or weight/size limitations), as a visual aid to improve speed control systems or in the predictive sensing systems for autonomous vehicles to predict actions and trajectories, obtain other vehicles orientation or relative position information or enhancing segmentation tasks.

Classic identification systems based on *License Plate Recognition (LPR)* and database matching have multiple disadvantages. LPR systems may fail, and, if there is criminal intention, they are easily avoidable by, for example, changing the license plate. According to *European Statistical Office (eurostat)*, between 2008 and 2018, 9 million vehicles were stolen in *European Union (EU)*-28 [13], with a clear downward trend, going from more than 1 million to 650,000 thefts per year. In the same period, the *Federal Bureau of Investigation (FBI)* recorded a total of 7.24 million thefts in the *United States of America (USA)*, going from 810,000 to 662,000 thefts per year [14]. Many of these thefts occur when vehicles are parked unattended on the street, however, a not negligible part of the thefts occur in car parks with access control or surveillance. In numbers of the *European Parking Association (EPA)*, in 2013 there were an estimated

33.8 million regulated parking spaces, with 21.8 million off-street and 12 million on-street, in EPA municipalities with more than 20,000 inhabitants [15]. Figure 1.1 shows the distribution of off-street and on-street parking spaces.

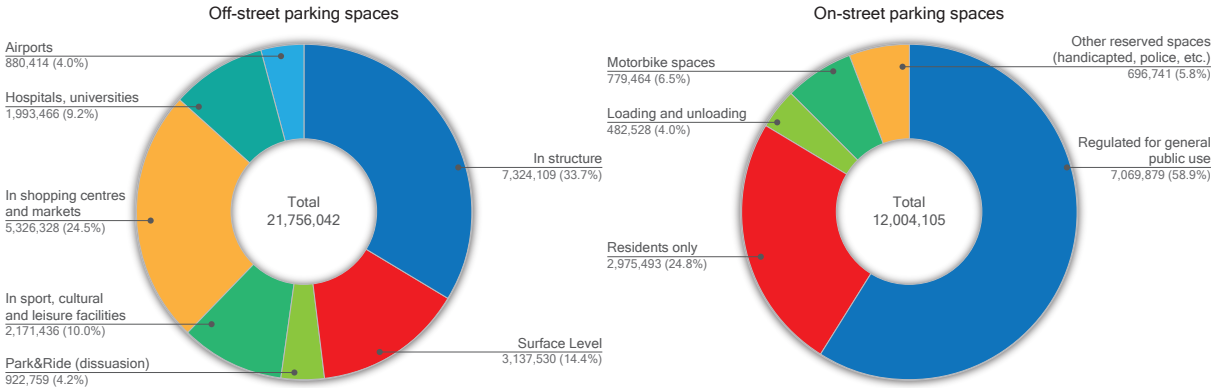

Figure 1.1: Prediction of distribution of off-street and on-street parking spaces in EPA municipalities with more than 20,000 inhabitants.

License plate swapping is a widespread practice in cases of vehicle theft. In the event that a stolen vehicle is detected by an automatic number plate scanner, if the number plate has been exchanged, this will prevent alarms from being triggered. Another particularly pernicious use is in the case of car parks with access control and ticket linked to the number plate. A widespread technique used by thieves is to enter with an old, low-end vehicle, also stolen, and once inside the car park they focus on high-end vehicles, to which they replace the number plate to be able to exit with the low-end vehicle ticket without problems. Another widespread practice is the use of a stolen license plate to commit an illegal act and then discarding it. All this can be prevented, at least to a large extent, if LPR systems are complemented with more visual information obtained with vehicle pose estimation and/or fine-grained vehicle classification systems, thus being able to detect inconsistencies between the vehicle registered for a given number plate and the carrier vehicle. Figure 1.2 shows a visual example of license plate swapping in a car park.

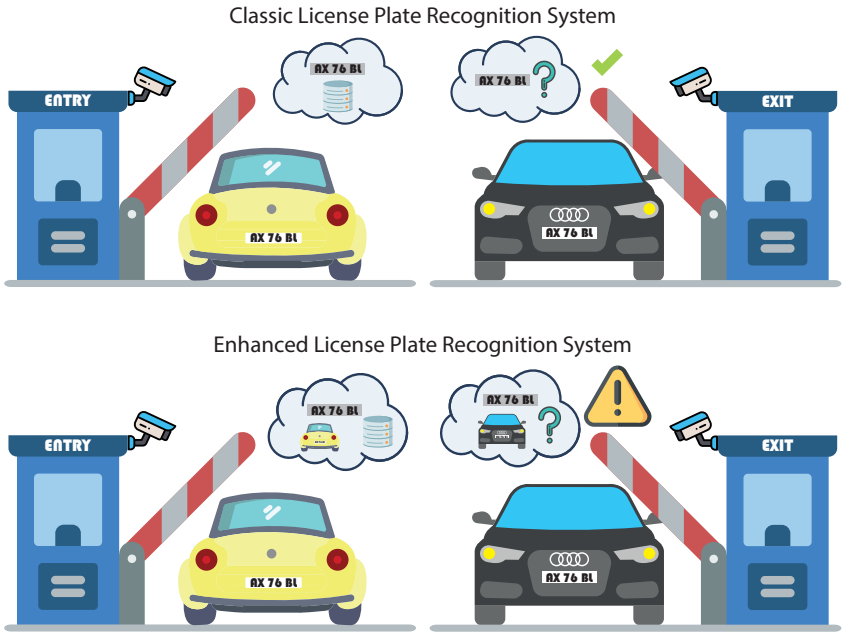

Figure 1.2: Visual example of license plate swapping in a car park. Top shows a classic LPR system that does not detect the license plate swap. Bottom shows an enhanced LPR capable of detecting the license plate swap preventing the attempted theft.

Going one step further in vehicle identification, it could be also useful to obtain the structural information of the vehicle making use of keypoints. This information can serve as a more efficient way of vehicle detection and identification, by detecting the license plate or the headlights instead the full vehicle. Keypoints by themselves are useful for motion detection and prediction systems, but, combined with an identification system, they can be useful in other applications like pose extraction, "digital twin" simulation environments generation or scene reconstruction.

As previously stated, vehicle pose estimation and fine-grained vehicle classification could complement each other, improving pose once the make and model are known, improving make and model prediction knowing the pose, and obtaining enhanced information when used jointly. For these reasons, it is necessary to study if these two methodologies can be combined to improve performance instead of using them individually. Additionally, it is clear that the development of efficient and robust vehicle pose estimation and fine-grained vehicle classification systems is of great importance, given their high impact and usefulness for ITS and in a wide range of applications. Finally, because of this wide range of applications, it is desirable that these systems are applicable to multiple domains (e.g., traffic cameras, on-board cameras, hand-held cameras) without the need to design and build them specifically for each domain, regardless of whether it is a traffic camera, an access control or an autonomous vehicle.

1.2 Problem Description

1.2.1 Vehicle Keypoint Detection

Many of the advances achieved in keypoint detection come from human pose estimation. Comparing vehicle keypoint estimation with human one, it should be easier, as the amount of possible poses is much lower given the rigid structure of vehicles, and, unlike humans, occlusions and overlapping are considerably less complex and cleaner. However, other specific difficulties arise that need to be considered. Camera perspective impact is stronger on cars than on people. While human anatomy is more or less homogeneous, vehicles have a much higher intra-class variability, due to the large number of different makers, models, sizes, and types. But, one of the most important problems, if not the most, is the definition of the keypoints. While the human body is practically telling, in an intuitive way, what the location of the keypoints should be, vehicles case is not the same. For example, a knee is always a knee, all knees are similar and placed in the same position. However, what happens with, for example, windshield corners or headlights? Each model has different proportions and shapes, which makes harder to define keypoints that are common in appearance and position for all keypoints.

Searching through the existing vehicle keypoints datasets [1, 2, 11, 16–18] a lack of consensus when labelling vehicle keypoints can be seen. This variability in the chosen keypoints makes very difficult to compare methods trained with different datasets. It is necessary to conduct a study to choose the most appropriate keypoints and, to our knowledge, there are no studies analysing the labelling consistency and each keypoint suitability. Figure 1.3 shows an example of keypoint variability from different datasets.

Another important issue is the wide variety of existing approaches to solve the problem of keypoint detection in vehicles, many of them adapted from human pose estimation. At the same time as their complexity has grown, models that are very different in conception obtain very similar results. This together with the variety of datasets makes it virtually impossible to compare different models or datasets.

For all these reasons, a simple baseline method for vehicle keypoint detection is proposed, adapted from a state of the art human pose estimation method together with an in-depth study of the suitability of the chosen keypoints from one of the main vehicle keypoints datasets and a custom keypoints dataset, with the aim of solving the shortcomings of the existing ones, paying special attention to the suitability of the chosen keypoints and their number.

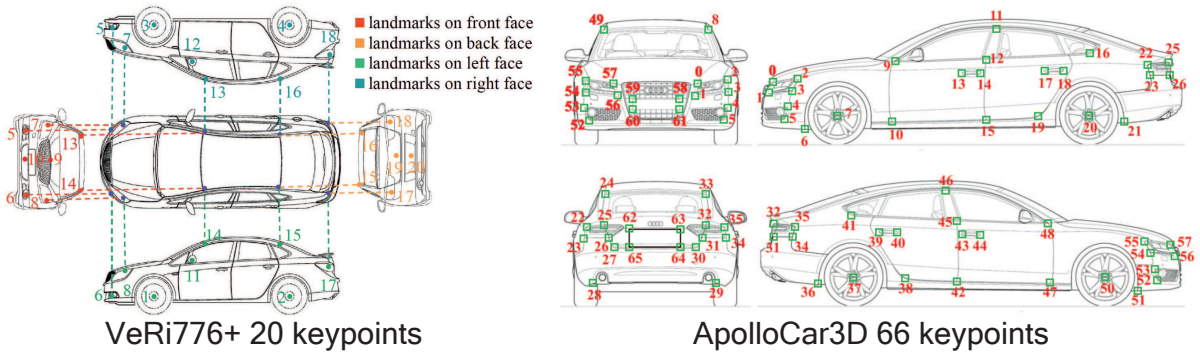

Figure 1.3: Keypoint variability examples from VeRi776+ and ApolloCar3D datasets with 20 and 66 keypoints respectively. Images extracted from [1] and [2].

1.2.2 Fine-grained Vehicle Classification

As previously stated, fine-grained vehicle classification is a task that consists in classifying vehicles according to make and model. When working to solve the fine-grained vehicle classification task we encounter three main problems. First, the *multiplicity* problem. This problem is the fact that for a particular vehicle model, its different generations have different shapes and/or appearance. This requires trained models to be able to differentiate between different vehicle models while being able to group different generations of the same model taking into account their similarities and differences. Second, the *ambiguity* problem. This problem is the fact that a manufacturer or group of manufacturers share platforms, designs and/or visual language. Thus, two different vehicle models can be visually very similar. Therefore, it is necessary that trained models learn to find the key features that allow them to differentiate between very similar models beyond their similarities. Third, the *bias* problem. This problem is the fact that available datasets are conditioned, with images far from real due to its high quality or for being renders, lack of diversity in viewpoints or illumination and the distribution of makes and models is not representative of the actual study population, with vehicles from a specific region. All these issues make the fine-grained vehicle classification problem a challenging task in which an unbiased and correctly constructed dataset is of vital importance. Figure 1.4 shows a visual example of these problems.

There are a large number of datasets aimed to solve the fine-grained vehicle classification task. Among all these datasets two different categories can be distinguished. First, specific datasets. These datasets have been created with the objective of solving a specific task or are limited to a given scenario, like surveillance [5, 19, 20]. Because of this, they are usually smaller, with little flexibility and offer limited capabilities of generalisation as they are biased. Second, general purpose datasets. These datasets have as objective to advance the fine-grained vehicle classification state of the art and are designed to be multipurpose, like [21–24]. Constructing a general dataset that accurately represents reality poses a number of challenges. The high difficulty of performing this task properly usually makes the datasets biased, with poor viewpoint variety, few scenarios or lighting conditions. All of this limits the use of these datasets in real world applications.

Most work focuses on tackling a specific problem, such as vehicle re-identification or traffic surveillance, or obtaining raw fine-grained classification performance, either on an existing dataset or on a purposely created dataset. There is a lack of analysis of per-class performance, which masks the obtained results. All this leads to the current situation where datasets such as CompCars [22] have their performance completely saturated with results around 98% accuracy. This may suggest that the fine-grained vehicle classification problem is solved, however, when the trained models are used in more challenging scenarios or per-class performance analysed, the results are not as expected, with significant drops in performance.

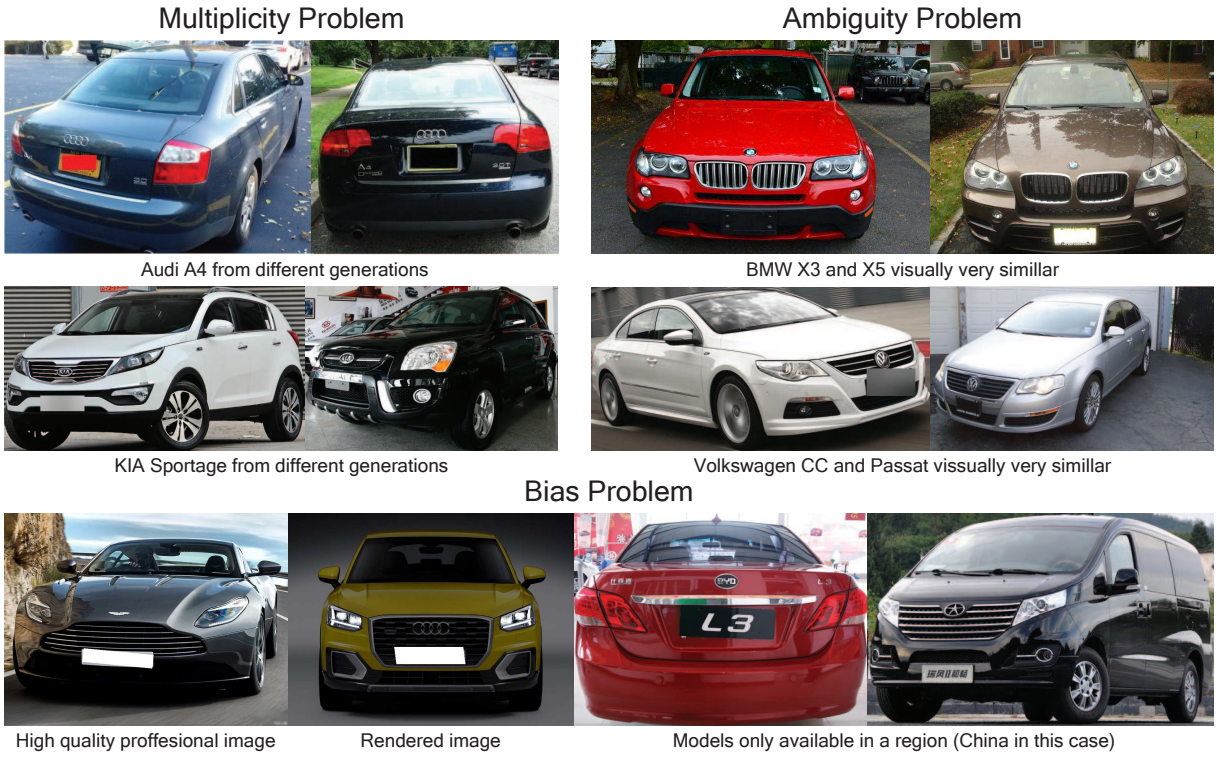

Figure 1.4: Visual examples of the three main fine-grained vehicle classification problems. Top-left the *multiplicity* problem, with two Audi A4 and two KIA Sportage from different generations, it can be seen that even though they are the same model there are visual differences. Top-right the *ambiguity* problem, with BMW X3 and X5 and Volkswagen CC and Passat, it can be seen that while they are different models, they share visual appearance. Bottom the *bias* problem, with an example of a high quality image and a render, both far from real world, and two examples of vehicles only available in a specific region, in this case China.

Along with the lack of analysis of per-class performance, dataset bias is often ignored and not taken into account. This dataset bias materialises among other problems, as class imbalance. In datasets composed of hundreds of classes, an average accuracy of 95% can be reported, despite having multiple classes with very low performance or poor generalisation ability. This is because, due to class imbalance, these classes have such a low number of samples that they hardly affect the overall results.

Special attention will be paid to generalisation capabilities, per-class performance and dataset bias. Our aim is not only to achieve a good average performance but also a good specific performance for all or as many classes as possible. Because of this, a new cross-dataset along with an external test set will be proposed to asses the complexity and generalisation capabilities of existing datasets and mitigate bias.

1.3 Document Outline

After the introduction in Chapter 1, Chapter 2 contains an in depth review of the state of the art of the main existing work for vehicle keypoint detection and fine-grained vehicle classification together with the datasets available to address these tasks.

Chapter 3 presents the approach, methodology and techniques used for vehicle keypoint detection along with the custom keypoints dataset and a relation of the experiments to be carried out.

Chapter 4 presents the approach, methodology and techniques used for fine-grained vehicle classification, the PREVENTION test set developed to evaluate the different models and datasets and the Fusion cross-dataset developed to evaluate complexity and generalisation capabilities of

existing datasets.

Results for experiments are presented and discussed in Chapter 5.

Chapter 6 contains the conclusions and main contributions to vehicle keypoint detection and fine-grained vehicle classification as well as future research lines.

Finally, Appendix A shows some of the open lines of research that were explored but did not meet the results or the applicability to be included in the main document. Appendix B summarises the main publications derived from this PhD dissertation.

Chapter 2

State of the Art

This chapter provides a detailed review of the state of the art. This includes an analysis of the existing work for vehicle keypoint detection and fine-grained vehicle classification as well as existing datasets to address these tasks.

The first section covers the main available datasets to address the tasks of vehicle keypoint detection and fine-grained classification. In the case of keypoints datasets we have focused on the source and number of images, the number of instances and the number of keypoints. For fine-grained datasets we have focused on the nature of the images, the amount of them, the number of classes and the variety of viewpoints.

The second section reviews existing vehicle keypoint detection and fine-grained classification work in depth. The keypoints part pays special attention to the two existing methodologies (top-down and bottom-up) and outlines their pros and cons. This is followed by an analysis of the early work on vehicle keypoint detection and how the advances made for human keypoints have been exploited by other authors for vehicles. The fine-grained classification part focuses on the different methods adopted in the literature, from the pre-CNNs era to the current methods mostly CNN-based. Of the latter, especial attention is put on how they tackle the problem, focusing on the location, appearance and parts of the vehicles, working in 3D space or focusing on the different modules of the networks and how to train them.

2.1 Existing Datasets

This section introduces and reviews the different datasets used to address the problems of vehicle keypoint detection and fine-grained vehicle classification.

2.1.1 Vehicle Keypoints Datasets

Unlike other keypoint detection tasks, like the human one, there is not a large number of publications concerning keypoint detection in vehicles. Because of this, and together with the high cost of acquiring and labelling the vehicle images, the amount of available vehicle keypoints datasets is limited.

- PASCAL3D+ [11]
  PASCAL3D+ dataset is one of the most famous 3D object detection datasets with keypoints labelled for 12 rigid categories of PASCAL VOC 2012 [25]. Focusing on vehicles, there are a total of 6,704 images, of which 1,229 are from PASCAL and 5,475 are from ImageNet [26]. These images contain a total of 7,791 instances, of which 2,161 are from PASCAL and 5,630 are from ImageNet. They provide 10 CAD models and a set of 12 keypoints: *the four wheels, windshield's and rear window's upper corners, headlights and*

left/right side of the trunk. They also provide a train/val split of approximately 50% with 621/608 images (1,091/1,070 instances) for the PASCAL subset and 2,763/2,712 images (2,850/2,780 instances) for the ImageNet subset. However, some problems have been detected that negatively affect the quality of the dataset. Many of the vehicles are old models that are almost impossible to see on the street and are very different in shape from today's vehicles. Many images have poor quality and the average resolution is low. All this can negatively affect the generalisation capabilities of a model trained with this dataset. Figure 2.1 shows some sample images from PASCAL3D+ dataset.

Figure 2.1: Example of some of the images from PASCAL3D+ dataset. In green the bounding boxes and in red the keypoints. Top row images are from PASCAL subset and bottom ones from ImageNet subset.

- ObjectNet3D [17, 27]
  ObjectNet3D is a large scale dataset for 3D object recognition. It has 100 categories, with a total of 90,127 images extracted from ImageNet [26], of which 7,345 contain at least one car, 201,888 instances, 12,886 of them cars and 44,147 3D shapes (6,601 cars) extracted from Trimble 3D Warehouse[1] and the ShapeNet repository [28]. Each 2D object has its 2D bounding box annotated and its aligned with its 3D shape. This provides accurate 3D pose annotation and 3D shape for each 2D object. The different viewpoints of the cars are homogeneously distributed and 12 keypoints have been labelled in the 3D models, making it possible to extract the 2D keypoints for each instance. The main problem of this approach is that having to project the 3D keypoints to 2D can lead to projection errors, especially if the 3D model does not fit the vehicle properly. Figure 2.2 shows some sample annotations from ObjectNet3D dataset.

Figure 2.2: ObjectNet3D example. On the left 3D pose annotations and on the right 3D shape annotation.

- VeRi-776+ [1]
  VeRi-776+ is an extended annotation of 20 keypoints and 8 different orientations to the whole VeRi-776 [29] dataset, which is a vehicle re-identification dataset that contains more than 50,000 images of 776 different vehicles captured by 20 traffic surveillance cameras

[1] https://3dwarehouse.sketchup.com/

covering 1km^2 in 24h. This dataset aims to improve vehicle re-identification making use of vehicle keypoints as a guide to learn discriminative information. Sadly, the images are fitted to the car instances, have low resolution, and, while there are 8 different orientations, the point of view is very conditioned by the fact that the images are captured from traffic surveillance cameras. Figure 2.3 shows some sample images from VeRi-776 dataset.

Figure 2.3: Example of some of the images from VeRi-776 dataset with its keypoints in red.

- Car Renders [16]
  Car Renders dataset is a vast synthetic dataset consisting of 600k fully visible and occluded car images. To build the dataset, 472 CAD models have been gathered from ShapeNet [28] and annotated with 36 3D keypoints. Then, each CAD model has been rendered using random parameters for camera viewpoint, light source, and surface reflection. To prevent over-fitting a set of real backgrounds have been used with these rendered instances. Unfortunately, the images have low resolution and, as it is a synthetic dataset, it is strongly biased. Figure 2.4 shows some sample images from Rendered Images (Car) dataset.

Figure 2.4: Example of some of the images from Car Renders dataset.

- KITTI-3D [16]
  KITTI-3D dataset is a set of 2,040 images from KITTI [30]. These images, together with 2D keypoints, are provided by Zia et al. [31]. Each instance is labelled with its occlusion type (no occlusion, truncation, multi-car occlusion and object occlusion) and a 3D groundtruth obtained by fitting a *Principal Component Analysis (PCA)* model by minimizing the 2D projection error for the known 2D keypoints and pose from the original KITTI dataset. The main purpose of this dataset is to test the performance of models trained with Rendered Images (Cars) dataset.

- CarFusion [18]
  CarFusion is a framework that improves detection, localisation and reconstruction of moving vehicles combining point tracking and part detection. Along with this framework, a dataset of 53k images, 100k instances and 14 manually annotated keypoints is provided. The images were captured from 18 moving cameras at multiple intersections in Pittsburgh, Pennsylvania. Unfortunately, the downloaded data only included 32,948 images and 63,203 instances with 13 keypoints (the 14th keypoint is set to 0 for all instances). Figure 2.5 shows some sample images from CarFusion dataset.

- ApolloCar3D [2]
  ApolloCar3D is a large-scale fully 3D shape labelled dataset comprising 4,283 images and a total of 55,616 car instances. To provide the 3D shape they used 34 industry-grade 3D

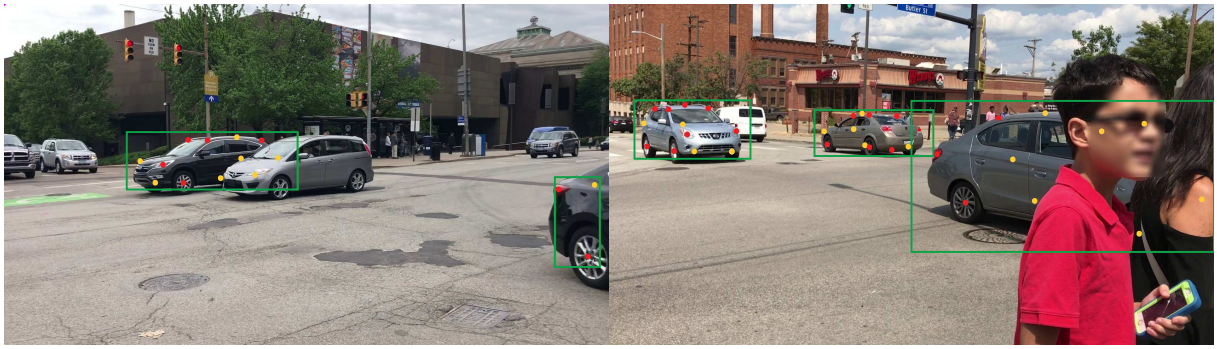

Figure 2.5: Example of some of the images from CarFusion dataset. In green the bounding boxes, in red the visible keypoints and in yellow the non visible keypoints.

CAD car models and 66 2D keypoints are labelled. Two main problems have been detected with this dataset. The first, using only 34 3D CAD models poses a problem, as the fitting will be approximate. The second, 66 keypoints are, by far, too many keypoints. Most of them do not represent structural key parts of the vehicle, introducing noise and increasing the complexity of the needed model to properly classify them all. Fortunately, the problem of the number of keypoints can be easily solved by discarding the undesired ones. Figure 2.6 shows some sample images from ApolloCar3D dataset.

Figure 2.6: Example of some of the images from ApolloCar3D dataset with the keypoints in red.

Table 2.1 shows a summary of the previously reviewed datasets, highlighting images source, the number of images and instances, amount of keypoints, 3D info and observed problems.

Dataset	Source	Images	Instances	Keypoints	3D	Problems
PASCAL3D+ [11]	Natural	6,704	7,791	12	10 CAD	Old cars, low resolution, poor quality
ObjectNet3D [17]	Natural	7,345	12,886	12	10+6,591 CAD	2D keypoints projected from 3D
Veri-776+ [1]	Surveillance	50,616	50,616	20	8 orientations	Low resolution, point of view
Car Renders [16]	Synthetic	600k	600k	36	472 CAD	Synthetic, low resolution
KITTI-3D [16]	Self-driving	2,040	2,040	36	38 CAD	Used as test
CarFusion [18]	Natural	53k	100k	14	No	Lower actual amount of data
ApolloCar3D [2]	Self-driving	4,283	55,616	66	34 CAD	CAD models fit approximately

Table 2.1: Summary of the most relevant vehicle keypoints datasets.

2.1.2 Fine-grained Vehicle Classification Datasets

Despite the existence of a significant amount of fine-grained vehicle classification research, only a few large-scale datasets are available. This has led researchers to work with their own datasets which, due to the high cost of acquiring and labelling thousands of images, are small or medium in size.

- Stanford Cars (Cars-196) [21]
  Cars-196 dataset is one of the first large-scale fine-grained vehicle classification datasets. It contains 16,185 images from multiple viewpoints of 196 classes of cars with labels for make, model and year. Each class has been roughly split in a 50-50 way, with 8,144 images for train and 8,041 for validation. The data was collected from Flickr, Google and Bing and refined on Amazon Mechanical Turk. As it is one of the first datasets with a relevant size, many works use it. However, the total amount of images is still low, many of the vehicles are from the same year (2012), affecting diversity, and the quality of images is too good, making them far from real world application. Figure 2.7 shows some sample images from Cars-196 dataset.

Figure 2.7: Example of some of the images from Cars-196 dataset.

- CompCars [22]
  The CompCars dataset is a large scale dataset that proposes three different tasks: fine-grained classification, attribute prediction and car verification. Its data has been obtained from two different scenarios, one of web-nature and the other of surveillance-nature. The web-nature set has a total of 136,727 full vehicle images and 27,618 vehicle parts from 163 makers and 1,716 models. These images have multiple viewpoints and were collected from various internet sources like forums, websites and search engines. The surveillance-nature set has a total of 44,481 front view images collected from road surveillance cameras. For the fine-grained classification task they propose the use of a web-nature subset composed of 52,083 images from 431 different car models. The quality of CompCars dataset is good but it is not free of some problems. Being a Chinese dataset, many of the vehicles are Chinese, which is a bias problem in case of using it in other regions as most of the models will not be present. The images have professional quality (high resolution, good lighting, etc.), affecting generalisation capabilities in a real-world scenario. Finally, the multiplicity problem (same model, different generations) has been ignored, with all generations of a model grouped into a single class. Figure 2.8 shows some sample images from CompCars dataset.

Figure 2.8: Example of some of the images from CompCars dataset.

- BoxCars [5]
  The BoxCars dataset is a vehicle dataset focused on surveillance applications. Its images were taken from surveillance cameras and each correctly detected vehicle has 3 different viewpoints images. This dataset contains 63,750 images from 21,250 different vehicles with a total of 27 makers and 126 models. Between the annotations they provide make&model, submodel and model year classes along with 3D bounding box information. BoxCars is a robust dataset with real-world quality and diversity of views and lightning. However, the images have low resolution and size.

- VMMR-db [23]
  The VMMR-db dataset is one of the biggest vehicle classification datasets available. It contains a total of 291,752 images of 9,170 different classes. The source of these images is considerably diverse, with images taken by different users and cameras which ensures variety of viewpoints, lighting and quality. To obtain these images the authors gathered them from online vehicle selling web pages and forums and used the information provided by the sellers to label the data. They also provide a CompCars overlapping subset of 51 classes and a 3,036 classes subset formed by all the classes with 20 or more images. Unlike CompCars, this dataset does not suffer the multiplicity problem, but, as the annotation process has been made in an automatic way, the same model generation has a different class for each year in which it was present. This, far from being an improvement, is a problem, since there are a non-negligible number of classes that are actually the same, introducing a significant amount of noise into the learning process. Figure 2.9 shows some

sample images from VMMR-db dataset.

Figure 2.9: Example of some of the images from VMMRdb dataset.

- Frontal-103 [24]
  Frontal-103 dataset is, so far, the most recent large-scale vehicle dataset. It contains a total of 65,433 frontal view web-nature images from 103 makers and 1,759 models. The images were collected from forums and online vehicle selling web pages. They tackle the multiplicity problem in a efficient way, assigning a class for each model generation. Unfortunately, it comes with some problems. Like CompCars, the amount of Chinese models is considerable, which is a problem in other regions, and the quality of the images, once again, is too high, with even some renders, making the data far from a real-world scenario. Figure 2.10 shows some sample images from Frontal-103 dataset.

Figure 2.10: Example of some of the images from Frontal-103 dataset.

Table 2.2 shows a summary of the previously reviewed datasets, highlighting viewpoint, image nature, number of samples, number of classes and observed problems.

Dataset	Viewpoint	Nature	# Samples	# Classes	Problems
Cars-196 [21]	Mixed	Web	16,185	196 models	Number of images, far from reality, poor diversity
CompCars [22]	Mixed	Web	52,083	431 models	Models source, far from reality, multiplicity problem ignored
BoxCars [5]	Mixed/3D	Surveillance	63,750	126 models	Size of images, quality, point of view
VMMR-db [23]	Mixed	Real	291,752	9,170 models	Many different classes for the same real class
Frontal-103 [24]	Front	Web	65,433	1,759 models	Models source, far from reality, labelling errors

Table 2.2: Summary of the most relevant fine-grained vehicle classification datasets.

2.2 Vehicle Keypoint Detection and Fine-grained Vehicle Classification

This section reviews the state of the art of vehicle keypoint detection and fine-grained vehicle classification in detail.

2.2.1 Vehicle Keypoint Detection

Keypoint detection is a widely explored task. While the most important advances have been made in the area of human keypoint detection, many other tasks such as vehicle keypoint detection have also received their share of attention.

Among all the work, two different approaches can be clearly distinguished, top-down and bottom-up methods. Figure 2.11 presents a visual comparison of top-down and bottom-up methods.

- **Top-down methods**. Top-down methods like [3,32–37] first detect all instances present in a given image by making use of an external object detector like Faster R-CNN [38], *Feature Pyramid Networks (FPNs)* [39] or *You Only Look Once (YOLO)* [40]. After all instances have been located, the keypoints of each one are predicted separately. This approach takes benefit of recent advances in instance detectors but has the problem of being strongly affected by the number of instances in the image, so that the processing time can increase considerably.

- **Bottom-up methods**. Bottom-up methods like [41–45] work in a different way than top-down methods. Instead of detecting each instance and predicting its keypoints, these methods first detect all keypoints present in the image. Once all keypoints have been detected each instance is reconstructed associating the different keypoints making use of different methods. The main advantage of these methods is their invariance to the number of instances in an image providing a stable and constant computing time. However, they tend to be slightly less accurate than top-down methods. This approach has been mainly studied for human pose obtaining very good results.

These two approaches are widely used and have their pros and cons. Top-down methods are easier to train and more reliable as they work with a single instance at a time. However, this could pose a problem. On the one hand, in crowded situations overlaps occur, so the system will detect keypoints from two different overlapping instances. On the other hand, as previously observed, in situations with a large number of instances, the computation time will increase.

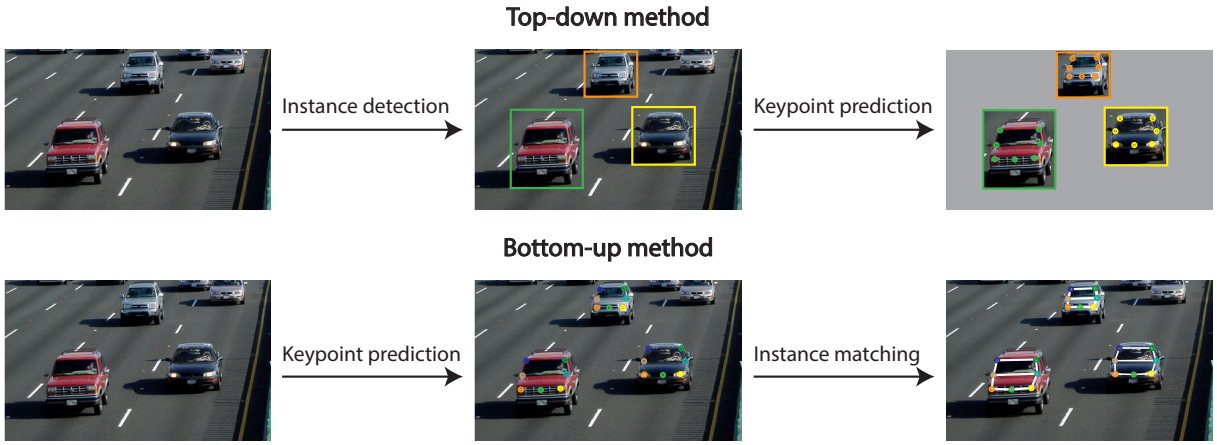

Figure 2.11: Visual comparison of top-down and bottom-up keypoints detection methods. The first one is a top-down method. First all instances are detected and then the keypoints for each one are predicted individually. The second one is a bottom-up method. First all keypoints in the image are detected an then matched to reconstruct the instances.

Bottom-up methods, although more difficult to train, have a constant throughput and work especially well in crowded scenarios, with far better performance for occluded or overlapped instances.

As previously stated, most of the advances in keypoint detection have been made in the area of human keypoint detection. Despite this, generic object pose estimation and vehicle pose estimation have also received some attention. Long et al. [46] presented one of the first approaches to use convolutional layers. They proposed the use of five convolutional layers intended to extract features from the image and then feed these features into a *Support Vector Machines (SVM)* for each keypoint. Following in their footsteps, Tulsiani et al. [32] used a fully convolutional regressor. Their approach makes use of a viewpoint prior combined with response maps from two different scales to calculate a likelihood map for each keypoint. In [12], Murthy et al. also used a fully convolutional regressor. With this regressor they predict the keypoints and subsequently refine them with a set of finetuning networks. In [16,33], Li et al. proposed an interesting approach. They used intermediate shape concepts like viewpoint, keypoint visibility, and keypoints to supervise the training process. To have enough training data they built a full synthetic dataset (Rendered Images (Car)) and showed state of the art performances when testing with KITTI-3D and PASCAL VOC.

One of the most widely used and exploited architectures are the stacked hourglass networks [3]. The structure of these networks is composed of residual convolutional modules packed in blocks with symmetric bottom-up/top-down capacity (from high to low resolutions and from low to high resolutions again) that seek to capture information at every scale. For this purpose, they make use of a single pipeline with skip layers that connect each branch with their symmetrical at the other end of the module. These basic modules shaped like an hourglass are then stacked to form the stacked hourglass network. One of the strengths of this architecture is that any number of modules can be stacked, making these networks very versatile and endowing them with a wide learning capacity. Because of this, and to prevent vanishing gradient, an intermediate supervision loss is calculated at the end of each module. Figure 2.12 shows a visual example of a stacked hourglass network.

The stacked hourglass networks have been used by multiple authors, mainly for human pose estimation. As a result, many of the developments related to these networks have taken place in this area. In [47], Wang et al. proposed to replace the residual modules used to construct the stacked hourglass by densely connected convolutional ones reducing the number of parameters and complexity of the network while maintaining the performance on MPII [48], a human pose dataset. A very interesting approach is the one adopted by Radwan et al. in [49]. They proposed

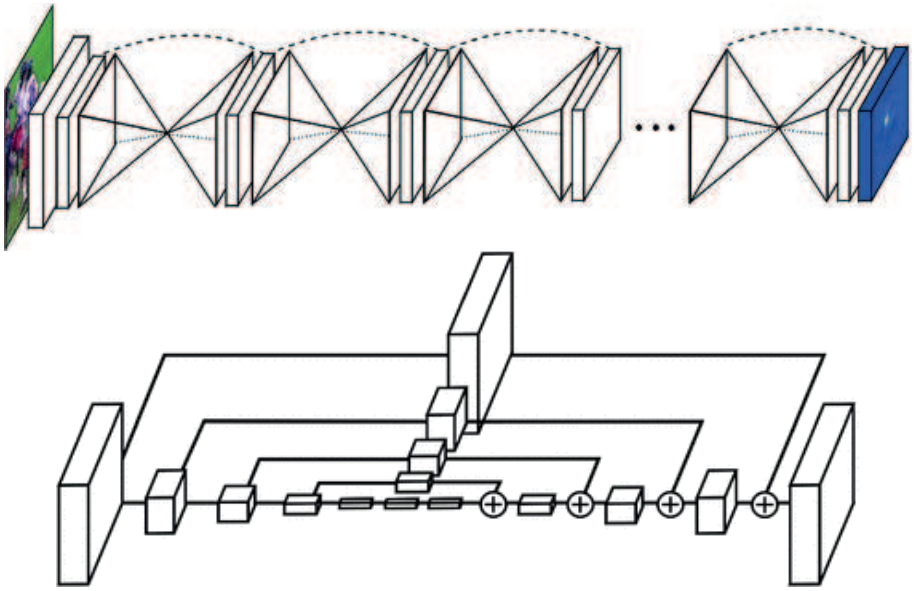

Figure 2.12: Visual example of a stacked hourglass network (top) and an hourglass residual module (bottom). Extracted from [3].

the use of a *Generative Adversarial Network (GAN)* [50] scheme and a keypoint hierarchy. The main contribution of their approach is the use of hourglass modules both in the generator and discriminator of the GAN obtaining comparable performance with other state of the art methods. Following this approach, Wang et al. [51] combined a self-attention mechanism with an hourglass based GAN outperforming previous methods on MPII.

The versatility and performance of the stacked hourglass networks has led many researchers to make use of them in other areas such as vehicle keypoint detection. Several works can be highlighted like [52], [53], [1], [4], [18] and [34]. One of the first authors, if not the first, to propose the adaptation of a stacked hourglass network for vehicle keypoint detection was Pavlakos et al. In [52] they presented a novel approach to estimate the *Six Degrees of Freedom (6DoF)* pose from a single *Red, Green and Blue (RGB)* image. To do so they used a two-tiered hourglass network with intermediate supervision for keypoint localisation trained on PASCAL3D+ and a deformable shape model achieving satisfactory results. In [53], Murthy et al. used a *Conditional Random Field (CRF)*-Style loss function at the end of each hourglass module to enhance keypoint localisation and enforce inter-keypoint distance constraints. In [1], Wang et al. presented a vehicle re-identification framework which makes use of 20 keypoints that they have added to the VeRi-776 [29] dataset (VeRi-776+) to obtain an orientation-based region proposal and extract features that are aggregated and used as identifier of each vehicle. To obtain the keypoints that will be used by their framework they use a modified hourglass-like fully convolutional network. Following Pavlakos' et al. approach, Ding et al. [4] presented a two-stage 3D pose vehicle estimation framework from off-board multi-view images. First, they use a four-tiered stacked hourglass network along with intermediate supervision for vehicle keypoint detection trained on PASCAL3D+ along with a custom dataset. After having obtained keypoints location they perform a *Cross Projection Optimization (CPO)* to estimate the 3D pose reporting state of the art results and outperforming by far Pavlakos' results.

In [18], Reddy et al. presented a framework that improves detection, localisation and reconstruction of moving vehicles combining point tracking and parts detection. To perform the keypoints detection they use a standard stacked hourglass network. Later, continuing their work, they focused on the problem of occlusions in keypoints prediction. To tackle this they presented Occlusion-Net [34], a framework to predict 2D and 3D occluded vehicle keypoints. First, they use a stacked hourglass network to detect visible keypoints. After this, they use

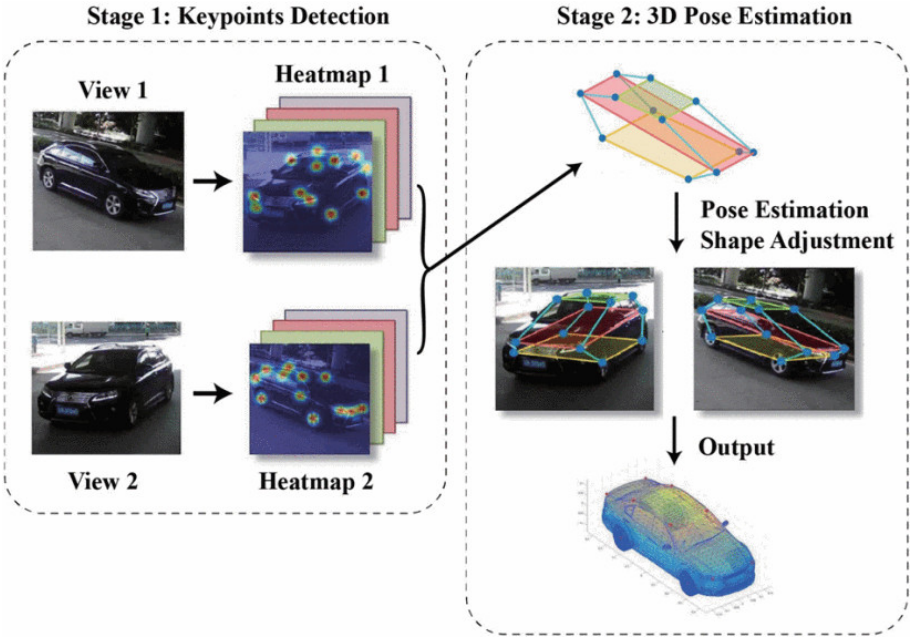

Figure 2.13: Overview of the two-stage 3D pose vehicle estimation framework presented by Ding et al. Extracted from [4].

an encoder-decoder scheme to predict the occluded keypoints exploiting multiple views of the object.

Stacked hourglass networks are not the only human pose estimation method that has been used for vehicle keypoint localisation. In [2], Song et al. used *Convolutional Pose Machines (CPMs)* [54] as its vehicle keypoint detector. In [41], Cao et al. presented the evolution of [55], the substitute of CPM [54]. This new method makes use of a multi-stage CNN with *Part Affinity Fields (PAFs)*, achieving better performance and showing that their approach is not only limited to human pose but it can also be applied to any keypoint detection task. To prove it, they trained the same model for vehicle keypoint detection and achieved competent results in CarFusion [18] dataset. Following the PAFs idea from [41], Kreiss et al. presented PifPaf [45], a composite field network that makes use of *Part Intensity Fields (PIFs)* and PAFs to detect and associate the keypoints. Later, in an evolution of their network [56], they replaced PIFs and PAFs with *Composite Intensity Fields (CIFs)* and *Composite Association Fields (CAFs)* outperforming existing methods for human pose estimation and achieving better results than Song et al. [2] in ApolloCar3D. In [10], Llorca et al. used the same approach presented by Nibali et al. in [57] to detect license plate corners instead of human keypoints. This consist in using a *Differentiable Spatial to Numeric Transform (DSNT)* along with dilated convolutions [58] to convert a fully convolutional network to coordinate regression.

Finally, table 2.3 shows a summary of all the keypoint detection reviewed methods.

Authors	Year	Dataset	Approach	Model
Long et al. [46]	2014	PASCAL VOC	top-down	conv. layers+SVM
Tulsiani et al. [32]	2015	PASCAL VOC	top-down	fully conv. regressor + viewpoint prior and response maps
Newell et al. [3]	2016	Human*	top-down	Stacked Hourglass
Murthy et al. [12]	2017	PASCAL3D+	top-down	fully conv. regressor + finetuning refinement networks
Pavlakos et al. [52]	2017	PASCAL3D+	top-down	Stacked Hourglass + deformable shape model
Murthy et al. [53]	2017	PASCAL3D+	top-down	Stacked Hourglass + CRF loss
Wang et al. [1]	2017	VeRi-776+	top-down	Stacked Hourglass like
Li et al. [33]	2018	Rendered Images (Car) KITTI-3D PASCAL VOC	top-down	fully conv. regressor + intermediate shape concepts
Wang et al. [47]	2018	Human*	top-down	densely connected Stacked Hourglass
Ding et al. [4]	2018	PASCAL3D+	top-down	Stacked Hourglass
Reddy et al. [18]	2018	CarFusion	top-down	Stacked Hourglass
Radwan et al. [49]	2019	Human*	top-down	Stacked Hourglass GAN
Wang et al. [51]	2019	Human*	top-down	self-attention Stacked Hourglass GAN
Reddy et al. [34]	2019	CarFusion	top-down	Stacked Hourglass + decoder-encoder
Song et al. [2]	2019	ApolloCar3D	bottom-up	CPMs
Cao et al. [41]	2019	CarFusion	bottom-up	CPMs
Kreiss et al. [45]	2019	Human*	bottom-up	PIFs+PAFs network
Llorca et al. [10]	2020	Custom	top-down	fully conv. regressor + DSNT
Kreiss et al. [56]	2021	ApolloCar3D	bottom-up	CIFs+CAFs network

Table 2.3: Keypoint detection state of the art summary. Methods with its dataset marked as Human* are exclusively trained and validated for human pose estimation.

2.2.2 Fine-grained Vehicle Classification

The task of vehicle classification, understood in many different ways, has been widely discussed in the literature. From the most basic prediction, such as the type of vehicle (sedan, SUV, familiar, etc), to the most complex, such as make, model and year. Of the multiple interpretations of the term vehicle classification, we have focused on the most specific fine-grained vehicle classification,

also known as make and model classification.

2.2.2.1 Pre-CNNs methods

Before the irruption of CNNs as the basic tool to solve any vision problem, classification tasks laid on hand-crafted features. One of the most widely used approaches to the task of vehicle classification in the pre-CNN era was to model the inherent characteristics of vehicles through their geometry and appearance. This approach is the one used by some authors like [59–61].

Santos et al. [61] proposed an automatic car recognition system composed of two different recognition methods, both relying on the external features of the car. The first one relies on the rear view shape of the car, focusing on dimensions and edges. The second one focuses on computed features from the car back lights like orientation, position, eccentricity, shape or angle to the license plate. Both methods perform well, and, although the first one is less reliable, when used together they produce a robust system with enhance performance. The authors highlight the fact that by making use of immutable, or at least difficult to modify features, their method is more reliable than previous approaches.

Following the rear view approach, Llorca et al. [60] proposed a novel approach to face the model recognition problem. First, they applied a previously developed license plate recognition module and a logo based make recognition system [62] to predict the car make. After this, they take advantage of the fact that most vehicles have emblems with model and/or version information and model the appearance and geometry of these emblems to predict the model.

Although these models perform well, they are limited to rear view images, which is a problem in a more general use case. A recurring problem is the diversity of viewpoints and the difficulty to properly tackle the pose variation of vehicles. To address this problem Gu et al. [59] proposed a mirror morphing scheme that, by exploiting the symmetry of cars, normalises any orientation image into a typical view, getting rid of the problem of severe pose variation. With this technique they are able to achieve high recognition rates despite using images with multiple viewpoints. However, their approach is not perfect. Some types of vehicles, such as 4x4, often lack symmetry at the rear (due, for example, to the spare wheel), so in these cases the system is not able to function properly.

2.2.2.2 CNNs era methods

One of the main problems of the pre-CNN era methods was the use of handcrafted features. The fact that CNNs have the ability to learn on their own, coupled with their power and versatility, has enabled an unimaginable number of approaches. Looking deeper into the existing works that make use of CNNs for vehicle fine-grained classification, many different approaches can be find, like focusing on location, appearance and/or parts, working in 3D space or using different networks, modules or training techniques, among others.

Location, appearance and parts. The first group is made up of methods that focus on location, appearance and/or parts [63–71].

In [63], Lin et al. proposed a novel end-to-end trained CNN architecture for fine-grained visual recognition called Bilinear CNNs. The idea is to have two networks that extract location and appearance related features and then, combine them as a pooled outer product, obtaining localised feature interactions invariant to translations. They also proved that these bilinear features are highly redundant and that can be reduced an order of magnitude keeping performance practically unaltered. They report results on Cars-196, and although they do not surpass the state of the art, they are close to it.

A recurrent approach to solving the vehicle classification problem was the use of vehicle parts that required annotations. With CNNs this problem disappeared. An example of this is the method proposed by Krause et al. [64]. By using bounding boxes, instead of part annotations

like in their previous work [21], they generate parts using co-segmentation and alignment in combination with *Region Based Convolutional Neural Networks (R-CNN)*. They show that this approach achieves state of the art results in their dataset (Cars-196) outperforming methods that use part annotations during training.

Continuing with the use of parts, there is the approach taken by Fang et al. [65]. They propose a coarse-to-fine method that makes use of CNNs to extract feature maps and locate discriminative regions where the differences between vehicles are more evident. These feature maps are used to detect refined regions and extract more features until there are no regions left. Finally, all the global and local features are jointly used on a one-versus-all SVM classifier obtaining state of the art results in the CompCars surveillance subset.

Following the discriminative region approach, Fu et al. [66] propose the use of a recurrent attention CNN to recursively learn discriminative region attention and region-based feature representation at multiple scales. With this framework they obtain similar results to [64] in Cars-196, but without needing to use human defined bounding boxes.

In [67], Zhao et al. proposed a *Diversified Visual Attention Network (DVAN)* that is able to gather discriminative information using multiple attention canvases from which it extracts convolutional features. A *Long Short-Term Memory (LSTM)* recurrent unit is then used to learn the attentiveness and discrimination of these canvases.

Following a similar approach to that of Fang et al. [65], Tian et al. [68] proposed to use an iterative discrimination CNN based on selective multi-convolutional region feature extraction to obtain global and local features. These features are then used to iteratively localise deep pivotal features and feed them to a fully-connected fusion layer. They report a performance similar to the state of the art in Cars-196 and in CompCars.

Another way to discover parts is the one proposed by Elkerdawy et al. in [69]. They use a co-occurrence layer to discover parts in an unsupervised way, thus avoiding the need of annotated parts or 3D bounding boxes. They report state of the art results in BoxCars and competent results in CompCars.

A different and interesting method is the one proposed by Du et al. [70]. They propose a framework that adds new layers in each training step and exploits the information from previous steps to enhance network input jointly with a jigsaw puzzle generator that forms images containing information from different granularity levels. They performed several tests in multiple fine-grained classification benchmarks obtaining state of the art results in Cars-196.

Recently, these results were outperformed by Ding et al. [71] by using enhanced feature representations and discriminative regions. To do so they presented the *Attention Pyramid Convolutional Neural Network (AP-CNN)*. This new network has two pathways, one for features and other for attention, with which to learn high-level semantic features and low-level detailed features. After this, they use a *Region of Interest (ROI)* guided strategy to refine the features and eliminate background noise.

3D Space. Among those that work in 3D space there are [5, 21, 72, 73].

The main limitation of 2D classification models is that their ability to generalise across different viewpoints is limited. To prove this limitation and overcome it Krause et al. [21] upgraded two 2D methods to 3D, obtaining better results with the 3D version. To do so they first estimate the 3D geometry of the object and then represent the appearance of local features and their locations in 3D space. However, as stated in the previous group of methods, this approach required parts annotations and was outperformed by their new method [64].

A common 3D approach is the one taken by Ramnath et al. [72]. In their work, they propose a method that recognises make and model from an arbitrary view by creating a 3D hull from the image. Once they have this 3D hull, a set of 3D space curves are projected and refined using three-view curve matching. Finally, these 3D curves are associated with 2D image curves through an alignment technique.

Instead of working separately with 3D model fitting and fine-grained classification, Lin et al. [73] proposed to optimise both tasks jointly. First of all, they extract initial part locations with *Deformable Part Models (DPM)*. Following this, they estimate landmarks locations with regression techniques. After this, they use the predicted 2D landmarks to fit 3D landmarks from a deformable model and extract part-based features that will be feed to a SVM classifier to predict the vehicle model. Finally, they use the prediction to further refine the landmark fitting.

Another interesting way to use 3D information is the one proposed by Sochor et al. in [5]. Rather than using plain 2D images, they use a 3D bounding box to *unpack* the vehicle shape and orientation boosting performance both for classification and recognition. Together with their method they present the BoxCars dataset with which they report performance results. Figure 2.14 shows an example of the 3D bounding box used.

Figure 2.14: Visual example of the 3D bounding box used by Sochor et al. [5] to *unpack* vehicle shapes and orientation.

At first it seemed that 3D methods were a clear bet over conventional methods, which make use exclusively of 2D information. However, their increased complexity and the need for datasets that are more difficult to build pose a barrier to their development. This, together with advances in CNNs, has meant that eventually 2D methods have outperformed them.

Other approaches. Finally, there is a diverse body of existing work that makes use of different network architectures, modules or training techniques like [74–78].

One of the most widespread techniques is fine tuning. Fine tuning consists in using the weights learned for a task such as ImageNet classification and retrain the full network to the desired task. In [74], Anderson et al. proposed to use a modular approach and, instead fine tune an already pre-trained network, combine a pre-trained network with a new untrained one. In this way, they lock the pre-trained network and only train the new one making it learn complementary features to those of the pre-trained one. They prove their method with Cars-196 showing a significant improvement in validation after 50 epochs when compared to the fine tuned model. However, it should be noted that they are making use of two networks, so their model has almost twice as many parameters.

Instead of a new network or training technique, Hu et al. [75] proposed the use of a *Spatially Weighted Pooling (SWP)* layer to improve the robustness and effectiveness of CNNs feature representations. This novel pooling layer contains a predefined number of spatially weighted masks that are learnt to pool the extracted features in a discriminative way. They obtain state of the art results in both Cars-196 and CompCars.

Another different approach is the one of Li et al. [76]. Instead of focusing on the CNN structure they focus on the loss function proposing a new regularisation term to cross-entropy loss. This Dual Cross-Entropy Loss alleviates the vanishing gradient problem and shows good performance with small datasets. They prove their approach with Cars-196 obtaining state of the art results.

In [77], Corrales et al. presented an end-to-end training methodology for CNN-based fine-grained vehicle classification. Their method consists of the joint use of various data augmentation

techniques, learning rate policies and fine tuning strategies applied on different backbones. They report state of the art results in CompCars.

Recently, Buzzelli et al. [78] *revisited* CompCars. They defined a new train/test split considerably more challenging and realistic preventing that very similar images where placed both in train and test splits. They also propagated the existing type-level annotations (sedan, familiar, hatchback, etc.) to the whole dataset. Together with this new split they also performed multiple experiments and implemented three different methods: one that predicts make-model-year, a two-step approach that predicts vehicle type and then uses that information to predict make-model-year and, finally, a multilabel approach that jointly predicts type and make-model-year. With the new split the new baseline performance considerably drops, going down from ∼90% to 61% accuracy confirming that the previous split was inadequate and the new one much more challenging. They achieve 70% accuracy in the new split with the two-step method, which is the best performing one.

Finally, table 2.4 shows a summary of the CNNs era reviewed methods.

Authors	**Year**	**Dataset**	**Approach**	**Model**
Krause et al. [21]	2013	Cars-196	3D space	SVM
Ramnath et al. [72]	2014	Custom	3D space	Three-view curve matching
Lin et al. [73]	2014	FG3DCar	3D space	DPM+SVM
Lin et al. [63]	2015	Cars-196	Location, appearance, parts	Bilinear CNNs
Krause et al. [64]	2015	Cars-196	Location, appearance, parts	R-CNN co-segmentation and alignment
Anderson et al. [74]	2016	Cars-196	Other	CNN complementary features
Sochor et al. in [5]	2016	BoxCars	3D space	CNN
Fang et al. [65]	2017	CompCars	Location, appearance, parts	CNN+SVM
Fu et al. [66]	2017	Cars-196	Location, appearance, parts	Recurrent attention CNN
Hu et al. [75]	2017	Cars-196 CompCars	Other	CNN SWP layer
Zhao et al. [67]	2017	Cars-196	Location, appearance, parts	DVAN+LSTM
Tian et al. [68]	2018	Cars196 CompCars	Location, appearance, parts	CNN fully-connected fusion layer
Elkerdawy et al. [69]	2018	BoxCars CompCars	Location, appearance, parts	CNN co-ocurrence layer
Li et al. [76]	2019	CompCars	Other	CNN dual cross-entropy loss
Corrales et al. [77]	2020	CompCars	Other	CNN fine tuning strategies
Du et al. [70]	2020	Cars-196	Location, appearance, parts	CNN incremental # of layers
Ding et al. [71]	2021	Cars-196	Location, appearance, parts	AP-CNN
Buzzelli et al. [78]	2021	CompCars	Other	CNN hierarchical approaches

Table 2.4: CNNs era fine-grained vehicle classification state of the art summary.

2.3 Conclusions

The previous sections have analysed the different existing datasets for dealing with vehicle keypoint detection and fine-grained vehicle classification as well as the different methods used to solve these two tasks. The conclusions drawn from this analysis are:

- While there are a multitude of vehicle keypoints datasets, their use is heterogeneous. This makes it difficult to explore and develop new methods or approaches, as the used datasets are different with varying keypoints, number of them, or labelling criteria. Additionally, there is a lack of analysis of the appropriateness of the chosen keypoints as well as the number of them, so there is no information regarding the usefulness of the keypoints being used, its difficulty or the impact that they have in the models.
- The use of human keypoint detection techniques with vehicles is a widespread practice that has reported good results. However, very different models, with differing underlying ideas and varying degrees of complexity achieve similar results. This makes it very difficult to compare them and assess their suitability, which raises the question: *How good could a simple model be?*
- Although there is a large number of existing fine-grained vehicle classification datasets, their use is not homogeneous and it is very common to find works that present their own dataset and make use of it. Taking an in depth look, it can be seen that the main datasets are saturated, with results above 95% validation accuracy.
- There are multiple approaches to address the fine-grained vehicle classification task. However, there is a clear tendency to increase the complexity of the models focusing on improving the overall results while neglecting other aspects like class imbalance, generalisation capabilities or dataset bias.

2.4 Main Contributions

After reviewing the state of the art, and taking into account the discussion previously presented, the main contributions of this thesis are as follows:

1. The use of human keypoint detection techniques seems to be the way to go. However, the variety of approaches and complexity of existing models that obtain similar results makes it very difficult to choose one over the others. Following on the ideas proposed by [37], we wanted to answer the question: *How good could a simple model be?*. Therefore, the applicability of a simple baseline approach to deal with vehicle keypoint detection has been studied and evaluated, comparing its performance with the state of the art. Extensive experimental validation is carried out using PASCAL3D+ improving state of the art results.
2. The most important shortcomings of the current vehicle keypoints datasets are discussed and verified and a new custom vehicle keypoints dataset is proposed and evaluated focusing on per-keypoint metrics to gain insight about the suitability of each of the proposed keypoints, its difficulty and their impact in the models. In this way, the lack of analysis of existing datasets and their keypoints is addressed.
3. As the state of the art methods exclusively focus on improving the overall performance by increasing the complexity of the models, we wanted to explore other techniques such as curriculum learning or weighted losses with the aim of enhance learning capabilities. Because of this, the applicability of these techniques to fine-grained vehicle classification is studied and its impact on overall performance, per-class performance and generalisation capabilities evaluated.

4. Being the main datasets saturated with validation results above 95% accuracy one may think that fine-grained vehicle classification is solved. However, we differ. To prove that there is still room for improvement a test set built from the PREVENTION dataset [79] with the objective of externally evaluate performance and generalisation capabilities in a realistic scenario is proposed. Additionally, a cross-dataset to assess the complexity and generalisation capabilities of existing datasets is presented.

Chapter 3

Vehicle Keypoint Detection

This chapter describes the approach, methodology and techniques used for vehicle keypoint detection. The custom keypoints dataset developed to address the suitability of the chosen keypoints and the shortcomings of the existing datasets are presented and the experiments that are going to be carried out described.

This chapter is organised as follows: the proposed method is described in section 3.1. Here, the general approach is presented followed by an in-depth description of the architecture, the metrics used to evaluate the performance of the trained model, the data augmentation techniques that are going to be applied to preprocess the data and a brief discussion of the main available datasets and which one has been chosen. In section 3.2, the developed custom keypoints dataset is presented and compared with the chosen one. Finally, a relation of the different experiments that are going to be performed and their aim is described in section 3.3.

3.1 Proposed Method

As described in the vehicle keypoint detection state of the art in section 2.2.1, there are two different approaches to address the keypoint detection task, top-down and bottom-up methods. These methods have both pros and cons. In the case of top-down methods, as they work with a single instance, they are easier to train and more reliable. However, this advantage is also a disadvantage in crowded scenarios, where overlaps occur, the system is not capable of separate the overlapped instances detecting the keypoints together. Additionally, in situations with a large number of instances, the throughput is affected, as the model has to detect the keypoints for each instance one by one instead of all together. This is the main advantage of bottom-up methods, while they are harder to train, as they first detect all the keypoints in the image, its throughput is more or less constant and work especially well in crowded scenarios, obtaining far better performance in overlapped instances.

When comparing human body with vehicle structures it can be seen that as vehicles are rigid structures, the amount of possible poses is much lower, and overlapping is, by far, less complex and cleaner, as there is no possibility of having two vehicles intertwined (except in the event of a serious collision). For this reasons, and taking into account the fact that they are easier to train, top-down methods seem the way to go.

As previously seen, many of the developments in vehicle pose estimation have their origins in human pose estimation. Of all existing methods, the one proposed by Xiao et al. in [37] is the one that has attracted our attention the most. In their work, Xiao et al. analyses the current status of human pose estimation methods and datasets. They show that, in a matter of a couple of years, the results in the existing datasets have improved considerably and some datasets can be considered as saturated. At the same time, the architectures used have become increasingly complex and at the same time very different from each other. All this makes comparing the different methods extremely difficult. Therefore, they raised a question from the

opposite direction: *how good could a simple method be?* and provided a simple baseline method for human pose estimation that achieved state of the art results. Following this philosophy, we asked ourselves, *can this simple baseline method be adapted for vehicle keypoint detection?*

3.1.1 The Architecture

The proposed architecture uses an ImageNet pre-trained ResNet [80] as backbone (all three main ResNets will be tested: 50, 101 and 152) from which the final fully connected layer has been replaced with three deconvolutional layers and a 1x1 convolutional layer to generate predicted heatmaps for each keypoint. The deconvolutional layers have 256 filters and 4x4 kernel size with stride 2. The input ground truth vehicle bounding boxes have a fixed aspect ratio of *width:height = 4:3*, modified from the original *width:height = 3:4* used for humans. Note that in the case of humans, the normal view of a person is considerably taller than wide in most poses, in the case of vehicles this is the reverse, with most views having a greater proportion of width than height. Because of this, the input aspect ratio has been changed to fit the vehicle reality. This aspect ratio is enforced by extending the existing bounding box if possible, and if not, by adding black bands where necessary. Figure 3.1 shows a visual example of the bounding box extension/black banding to enforce aspect ratio.

Figure 3.1: Aspect ratio enforcement examples. Extended bounding box on the left and black banded on the right.

Two different input sizes, 256x192 and 384x288, will be tested with the three main ResNets (50, 101 and 152) and the generated output heatmaps will have a size of 64x48 for the 256x192 input and 96x72 for the 384x288 input. Adam is used as training optimiser with *Mean Squared Error (MSE)* as loss between the predicted and target heatmaps. The target heatmaps are generated from the keypoint coordinates using them as the centre of a 2D gaussian. The networks are trained for 140 epochs with a momentum of 0.9 and an initial learning rate of 0.001 that is reduced by a tenth at epochs 90 and 120. As the model proposed by Xiao et al. [37] is called *Simple Baselines for Human Pose Estimation* we will refer to our adaptation as *Simple Baseline for Vehicle Pose Estimation (SBVPE)*. Figure 3.2 shows a illustration of the full network structure pipeline.

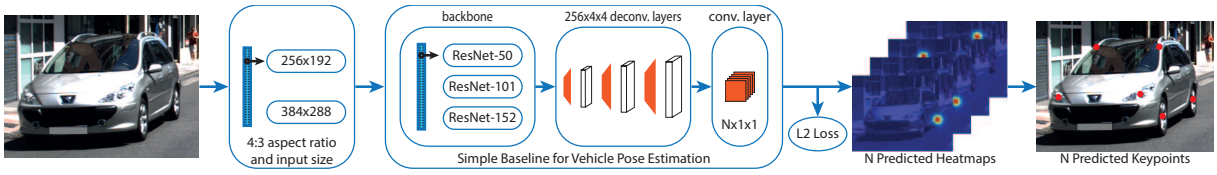

Figure 3.2: Illustration of the full simple baseline network architecture pipeline proposed for vehicle keypoint detection.

3.1.2 Metrics, Data Augmentation and Dataset

To achieve the best results possible, as important as a good architecture is the data used to train it, how this data is preprocessed and the tools used to evaluate performance.

3.1.2.1 Metrics

There are a multitude of metrics to properly evaluate the performance of the keypoint detection system. Among all of them, the ones proposed by [81] will be used:

- ***Percentage of Correct Keypoints (PCK)***. PCK measures the number of labelled keypoints that are correctly predicted. In order to consider a predicted keypoint as correct its distance to the labelled ground-truth keypoint has to be equal or less than $\alpha * L$, with $L = max(h, w)$ (L is the bigger side of the labelled bounding box) and $0 < \alpha < 1$. We use $\alpha = 0.1$.
- ***Average Precision of Keypoints (APK)***. As in PCK, a predicted keypoint is correct if its distance to the given ground-truth keypoint is equal to or less than $\alpha * L$. Each predicted keypoint has associated a confidence and a threshold is used to calculate the area under the precision-recall curve. This evaluation penalises missed-detections and false positives. The way to compute APK can be seen in equation (3.1), being P_n and R_n the precision and recall at the n^{th} confidence threshold.

$$APK = \sum_{n} (R_n - R_{n-1}) * P_n \tag{3.1}$$

3.1.2.2 Data Augmentation

Data preprocessing when training a CNN is of vital importance. The proper use of data augmentation techniques is essential to achieve better results, greater generalisation capabilities and prevent overfitting. For the vehicle pose estimation task three different data augmentation approaches will be tested:

- **No data augmentation at all.** Used as baseline to measure the impact in performance of data augmentation.
- **Mild data augmentation.** 50% chance of horizontal flip, random rotation of up to $\pm 30^{\text{o}}/40^{\text{o}}/50^{\text{o}}$, and random scaling of up to $\pm 30\%$.
- **Hard data augmentation.** Mild data augmentation and a randomly selected operation between salt-and-pepper noise, poisson noise, speckle noise, blurring, colour casting, and colour jittering.

Figure 3.3 shows visual examples of the different data augmentation operations described above.

Regardless of the approach taken, as the backbones used are pre-trained with ImageNet, all images are ImageNet normalised.

3.1.2.3 Dataset

One of the most important parts of training is the data used. The main available vehicle keypoints datasets have been presented in section 2.1.1. Of all the datasets, the most famous one, in part because it is one of the oldest, is PASCAL3D+. Along with PASCAL3D+, the most interesting ones are CarFusion and ApolloCar3D. During the dataset selection process, all three were thoroughly analysed. In the end we opted for PASCAL3D+. There have been several reasons for this decision, firstly, the other two datasets are more recent and do not have available results with which to compare the performance of our system. Secondly, while CarFusion have a higher amount of images and instances, the actual amount of data available from is less than stated. Lastly, ApolloCar3D is a great dataset, with a huge amount of images, instances and keypoints. The arbitrary amount of keypoints (66) could pose a problem, but this can be easily

Figure 3.3: Visual examples of the different data augmentation operations. On top, from left to right, no data augmentation, mild data augmentation (with horizontal flip, rotation and scaling), salt-and-pepper noise and poisson noise. On the bottom, from left to right, speckle noise, blurring, colour casting, and colour jittering. The hard data augmentation specific operations have been performed over the original image without the mild data augmentation to ease visualisation.

solved by discarding the undesired ones. However, ApolloCar3D is designed to train bottom-up systems and because of this the annotations does not have the adequate format to train our method. The possibility of modifying them was considered but finally discarded as the chaotic file structure and the amount of data made the task require an enormous amount of resources. Taking all of this into account, and even though PASCAL3D+ is not the best of the three, if we weigh up the possibility to easily compare our system with others and the effort needed to adapt and restructure ApolloCar3D, PASCAL3D+ is the dataset that best suits our needs.

PASCAL3D+ augments 12 rigid categories of PASCAL VOC adding 3D annotations. Focusing on vehicles, the original PASCAL images are extended with ImageNet images. There are a total of 6,704 images, 1,229 from PASCAL and 5,475 from ImageNet. A total of 7,791 instances are present in the images, with 2,161 from PASCAL and 5,630 from ImageNet. This means that there are roughly 1.76 instances per image in the PASCAL data and 1,03 instances per image in the ImageNet data. This means that while most ImageNet images have a single vehicle, PASCAL images usually have more than one. 12 keypoints are provided: *the four wheels, windshield's and rear window's upper corners, headlights and left/right side of the trunk.* In the labelling process two different reference frames have been taken, one from the front and one from the back, so the left/right side is cross-referenced with respect to the front and back keypoints. This adds an additional degree of complexity when working with the keypoints and can be misleading. We would also like to point out that during the use of the dataset multiple inconsistencies have been found, with keypoints not labelled or labelled in "arbitrary" places. This is especially true for trunk and headlights keypoints. Figure 3.4 shows some images from PASCAL3D+. Those at the top are from PASCAL and multiple instances can be seen, with some being truncated. Those at the bottom are from ImageNet.

3.2 Custom Keypoints Dataset

One of the most important parts of vehicle keypoint detection that we want to address is the suitability of the chosen keypoints and how they affect the results along with the shortcomings of the existing datasets. For this purpose, a custom dataset has been created.

Our custom dataset is an upgrade of PASCAL3D+ dataset. We decided to focus on the ImageNet subset images, as they have higher quality, which facilitates the labelling process. In order to enhance the variety, images from two different sources, CompCars and the PREVEN-

Figure 3.4: Example of some of the images from PASCAL3D+. In green the bounding boxes and in red the keypoints. Top row images are from PASCAL and bottom ones from ImageNet.

TION dataset, have been added. The total amount of images is 4,042 with 4,080 instances. The instances are 2,801 from the PASCAL3D+ ImageNet subset, 898 from the fine-grained vehicle classification dataset CompCars, and 381 from the vehicle intention prediction dataset PREVENTION.

From the original 12 PASCAL3D+ keypoints, we have gone to 19 in the custom dataset. The main reason why these keypoints have been chosen is that we believe that the structural information that they provide is better and they are less susceptible to crossed confusion. When compared to the 12 original keypoints, the side of the wheels have been eliminated, going from the four wheels to front/rear wheel. From the upper corners of windshield and the rear window to the four corners, as they serve to better characterise the upper vehicle structure. Replacing the front lights, that are complex and of varied shapes, with the fog lights, that are much more homogeneous and are usually placed in the same area. rear view mirrors have been added, as they are easily distinguishable elements and provide information about the limits of the vehicle. And finally, the four license plate corners and the logo, as these elements can be very helpful for make and model classification or surveillance. Unlike in PASCAL3D+, only one reference frame has been taken, the front view. Because of this al keypoints share left/right reference from this view, making the process of interpreting the keypoints easier. Figure 3.5 shows a visual comparison of PASCAL3D+ keypoints and our custom keypoints.

3.3 Experiments

These are the experiments that are going to be performed to address the different vehicle keypoint detection issues:

- **Data Augmentation.** The different data augmentation approaches will be tested in order to evaluate its impact and choose the best solution.
- **Backbone and Input Size.** As previously stated, all three main ResNets will be tested in order to assess the impact of deeper backbones together with the increase of the input resolution from 256x192 to 384x288.

Figure 3.5: Visual comparison of PASCAL3D+ keypoints and our custom keypoints. Top row/green PASCAL3D+, bottom row/yellow custom keypoints.

- **PASCAL3D+.** As previously described, PASCAL3D+ consist of two subsets, PASCAL and ImageNet. The impact in performance of these subsets will be tested.
- **Instance Size.** A comparison of the two PASCAL3D+ subsets, PASCAL and ImageNet, shows important differences in complexity. An analysis of the impact of instance size on the model will be performed.
- **Keypoint Distribution Study.** Raw performance by itself tells us little about the suitability of the chosen keypoints. Because of this, an analysis of the keypoint distribution and their individual performance will be carried out.
- **Custom Keypoins.** To address the shortcomings of existing datasets a custom keypoints dataset has been proposed. The impact of this new dataset and the suitability of the chosen keypoints will be evaluated.

Chapter 4

Fine-grained Vehicle Classification

This chapter describes the approach, methodology and techniques used for fine-grained vehicle classification. The PREVENTION test set developed to externally evaluate performance and generalisation capabilities in realistic scenarios and the cross-dataset developed to mitigate biases and asses the complexity and generalisation capabilities of existing datasets is presented and the experiments that are going to be carried out described.

This chapter is organised as follows: the proposed method is described in section 4.1. Here, the general approach is presented followed by a description of the chosen architectures, metrics, data augmentation techniques that are going to be applied to preprocess the data and a brief discussion of the main available datasets and which ones have been chosen. Additionally, the different learning techniques that are going to be used are presented and described. In section 4.2, the PREVENTION test set and the cross-dataset are presented. Finally, a relation of the different experiments that are going to be performed and their aim is described in section 4.3.

4.1 Proposed Method

As presented in section 2.2.2, many different approaches have been adopted to tackle the problem of fine-grained vehicle classification. However, there has been a clear trend towards increasing the complexity of these models in order to improve the overall accuracy without taking into account other factors. Since our aim is not to obtain the best performing model, falling into the same errors as previous work, we decided that the best approach was to opt for the simplest possible solution while still obtaining overall results that were up to standard. All this without forgetting details such as class imbalance, generalisation capabilities or dataset bias.

Of all the existing techniques, one of the most widely used, yet simple, is fine-tuning. Fine-tuning consists of taking a network pre-trained with another dataset, usually ImageNet, and after replacing the final fully connected layers of the network, retraining it completely or partially with the desired dataset. As this approach makes use of architectures designed for image classification the only complexity source is the chosen architecture. Since our aim is to try to find a trade-off between complexity and raw performance that allows us to focus on the impact of class imbalance or dataset bias, the architecture selection process is not trivial.

4.1.1 The Architectures

In [6], Bianco et al. performed an in depth analysis of the main *Deep Neural Networks (DNNs)* used for image classification. Multiple metrics were used taking into account aspects such as accuracy, complexity or computational cost. Figure 4.1 shows a comparison of the main image classification architectures. Considering the data regarding accuracy, computational cost and complexity, along with the availability and popularity of the models, the chosen architectures to address the fine-grained vehicle classification problem are ResNet50 and InceptionV3. The

main reasons are two. First, these two architectures are, among the most popular, the ones that have the best performance/complexity ratio, with an extremely efficient use of their parameters. And second, they are perfectly capable of tackling the fine-grained vehicle classification problem not only obtaining a good overall performance, but also leaving space to study the impact of different learning techniques on per-class performance and generalisation, thus allowing a more comprehensive assessment of the proposed solution.

In short, the architectures that are going to be used are the ImageNet pre-trained ResNet50 and InceptionV3. The input size of these networks are 224x224 and 299x299 respectively. *Stochastic Gradient Descent (SGD)* is used as training optimiser with Cross Entropy Loss. Unless otherwise stated, the networks are trained for 50 epochs with a momentum of 0.9, an initial learning rate of 0.01 or 0.001, and two learning rate policies, keep the learning rate constant through the training, and reducing it every n epochs an order of magnitude in a stepped pattern.

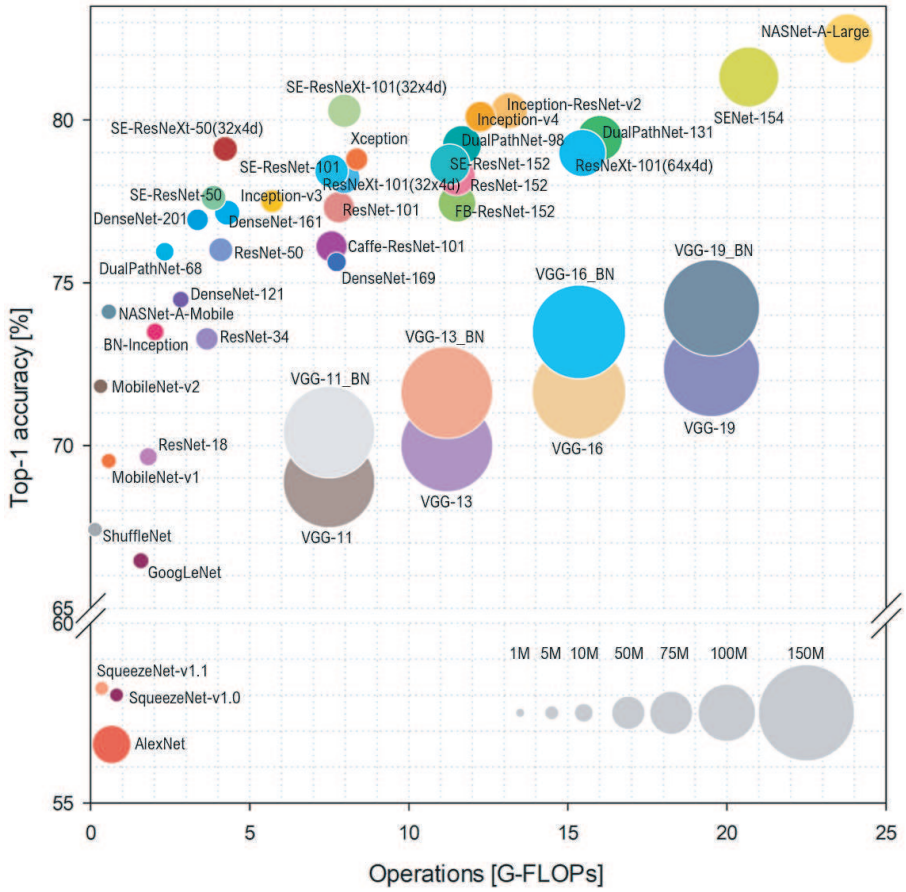

Figure 4.1: ImageNet-1k Top-1 accuracy vs. computational cost for a single forward pass. The size of each ball corresponds to the model complexity. Figure extracted from [6].

4.1.2 Metrics, Data Augmentation and Datasets

As important as the choice of architecture is the choice of the metrics to evaluate the models, the data that is going to be used and how the images will be preprocessed.

4.1.2.1 Metrics

There are multiple metrics to evaluate the performance of classification problems. Of all of them these are the ones to be used:

- **Precision**. For a given class, precision measures the amount of correct predictions of all samples predicted as the given class.
- **Recall**. For a given class, recall measures the amount of correct predictions for all the samples of the given class.
- **Accuracy**. Accuracy measures how many of the total predictions are correct.
- **Average Precision/Recall**. The average precision or recall of all classes without taking into account the weight of each class.
- **Weighted Precision/Recall**. The average precision or recall of all classes weighted to take into account the amount of samples of each sample.

Figure 4.2 shows an example of precision and recall calculation with an easy 3-class problem.

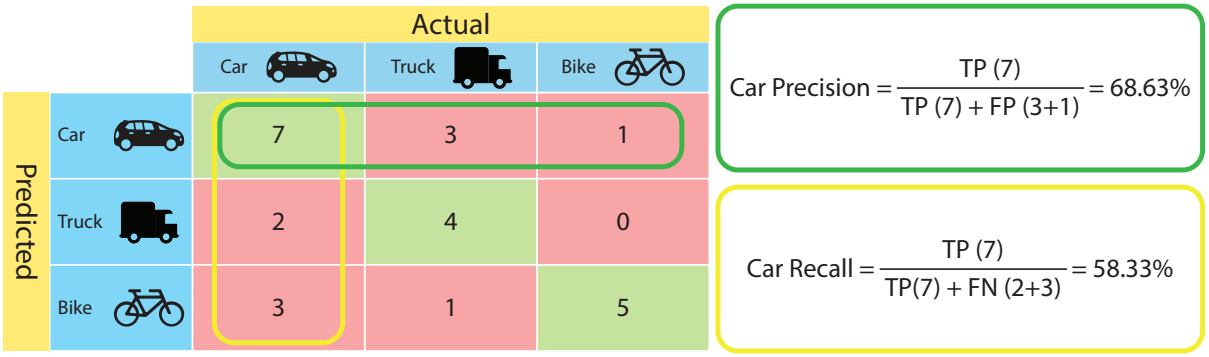

Figure 4.2: Example of precision and recall calculation. In green correct predictions and in red incorrect predictions. The first row (squared in green) is used to compute car precision. The first column (squared in yellow) is used to compute car recall.

4.1.2.2 Data Augmentation

Data augmentation is a vital part in the training process of a CNN. It is widely accepted that data pre-processing is essential to improve model performance, generalisation capabilities and prevent overfitting. For the fine-grained vehicle classification task various techniques will be used:

- **Horizontal Flip.** The image is rotated over the y axis with a 50% probability.
- **Salt and Pepper noise.** With a probability of 2% each pixel of the image is set to 0 or 255 (black/white).
- **Poisson noise.** Noise created with a poisson process is added to the image to simulate the shot noise originated from the discrete nature of electric charge.
- **Speckle noise.** Noise is added to the image to simulate the effect of environmental conditions on the imaging sensor.
- **Blurring.** The image is blurred using a gaussian kernel with random size between 3 and 11 and standard deviation of 6.
- **Colour Casting.** Each channel of the image is modified displacing its values with a probability of 33%.
- **Colour Jittering.** The image is converted to *Hue, Saturation, Value (HSV)* colour space and saturation and value are randomly modified.

Figure 4.3: Visual examples of the different data augmentation operations. On top, from left to right, no data augmentation, horizontal flip, salt-and-pepper noise and poisson noise. On bottom, from left to right, speckle noise, blurring, colour casting, and colour jittering.

The data augmentation operations are going to be applied to the images as follows. First, the image is randomly flipped. Second, one of the remaining six operations is randomly chosen and performed over the image. Finally, as the architectures that are going to be used are pre-trained with ImageNet, ImageNet normalisation is applied.

4.1.2.3 Datasets

Data is of vital importance for training a competent model. The main available fine-grained vehicle classification datasets have been presented in section 2.1.2. Of all these datasets, the most famous ones are Cars-196 and CompCars. Of these two, the most interesting one is CompCars, as it has a considerably larger number of images and classes. Of the others, the most notable is VMMR-db, since, to our knowledge, is the biggest fine-grained vehicle classification dataset available. The quantity and variety of images, as well as of situations, qualities and viewpoints make it probably one of the least biased and most interesting datasets to tackle the vehicle classification problem. Another dataset worth mentioning is Frontal-103. This dataset is, so far, the most recent large-scale vehicle dataset. Although, unfortunately, it only contains frontal images, it deals with the problem of multiplicity in an effective way.

For al these reasons the datasets that are going to be used are:

- **CompCars.** The CompCars fine-grained classification subset has a total of 52,083 images from 431 different car models. These images have multiple viewpoints and were collected from multiple internet sources like forums, websites and search engines. This is a solid dataset, but it has two main problems. The quality of the images makes it far from real-world conditions, and many of its classes are region-specific (China), which makes it biased when used in other regions.

- **VMMR-db.** With a total of 291,752 images and 9,170 different classes, this is the biggest fine-grained vehicle classification dataset to our knowledge. The source of the images is very varied, with multiple users and cameras, which leads to a great diversity of viewpoints, lighting and qualities. This is probably the least biased dataset in existence. However, since the labelling process has been done semi-automatically, many of the classes are actually the same, introducing noise in the training process.

- **Frontal-103.** In the same way as CompCars, the images are of web-nature and have been collected from forums and online car sales websites. It contains a total of 65,433 front view images from 103 makers and 1,759 models. Although it is a competent dataset, it suffers

from the same region-specific bias problem as CompCars and, as all images are front view, is limited to this kind of scenario.

4.1.3 Learning Techniques

As previously stated, there is a lack of analysis of per-class performance and the impact of dataset bias. The main source of these problems is the data, so one way to solve them is to make use of correctly constructed datasets with as little bias as possible. Unfortunately, this is not always an option given the high cost of creating a dataset. If there is no choice but to make use of a biased dataset made up of classes with diverse difficulties the use of different learning techniques can help alleviating the impact of the bias and compensate the varying difficulty of the classes. Among these learning techniques, we will focus on Curriculum Learning and Weighted Losses.

4.1.3.1 Curriculum Learning

Thinking about learning processes, one can intuitively come to the conclusion that they are much more efficient if the information is received in an organised way, expanding the concepts and their difficulty progressively. While this is the way humans learn, this approach has not been widely applied to deep learning. This idea was first proposed in 1993 by Elman et al. [82] and subsequently explored in 2009 by Bengio et al. [83], showing solid improvements in performance for multiple tasks.

Focusing on fine-grained vehicle classification, the hierarchical structure of the data (makes $\rightarrow$ models) eases the implementation of a progressive training process. Two different curriculum learning techniques will be used and its utility and impact on raw and per-class performance will be evaluated. These techniques are:

- **Incremental Learning.** This technique consists in first train an easier more general problem. After that, once the general knowledge has been acquired, retrain for the desired, more specific, task. In the case of fine-grained vehicle classification, it seems reasonable to first train the model for vehicle make classification (the general task), and then, refine the network for model classification (the desired more specific task).

- **Progressive Learning.** This technique consist in starting with a simple, easier problem, and then, progressively increase the difficulty at each epoch. For example, in a multi-class classification problem one starts with the easier classes and gradually adds more difficult classes in each epoch until all the classes have been added. To use this technique the fully connected layer is initialised for all the classes and progressively more and more data is shown to the network. The first batch of classes are the ones that had the best performance in a standard training and 5 or 10 new classes are added every epoch. After all classes have been added the training will continue for a few more epochs to ensure the last added classes are correctly learnt.

4.1.3.2 Weighted Losses

A recurrent problem with datasets is class imbalance. This problem occurs when one or more classes present in the dataset have a number of samples several orders of magnitude below the rest of the classes. This often leads to the fact that in the training process these classes are irrelevant during back-propagation. When this happens, it may seem that the model performs apparently well, however, these particular classes perform well below average. This means that the dataset has a bias problem that will negatively affect performance if these classes appear in real world conditions. However, this effect can be mitigated by using weighted losses to favour the under-represented classes or penalise over-represented ones.

Three different weighting approaches will be used:

- **Standard Weights.** Equation (4.1) defines the standard weights. This is, for a given class i, its weight is the percentage of representation it has in the total dataset.

$$W_i^1 = 1 - \left(num\_samples_i / \sum (num\_samples) \right) \tag{4.1}$$

- **Logarithmic Weights.** Equation 4.2 defines the logarithmic weights. This is, for a given class i, its weight is the percentage of representation in the dataset modified with the non-linear function $-log(x)$.

$$\begin{aligned} w_i^2 &= num\_samples_i / \sum (num\_samples) \\ W_i^2 &= -log(w_i^2) \end{aligned} \tag{4.2}$$

- **Focal Loss [84].** Focal Loss is a modified version of the cross entropy loss that reduces the relative loss of well-classified data while focusing on the harder misclassified samples. Equations (4.3), (4.4) and (4.5) define cross entropy loss and focal loss. In them, $y = 1$ means that the class has been correctly classified, and p is the predicted class probability.

$$CE(p_t) = -log(p_t) \tag{4.3}$$

$$FL(p_t) = -(1 - p_t)^{\gamma} log(p_t) \tag{4.4}$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases} \tag{4.5}$$

Figure 4.4 shows, on the left, the standard and logarithmic weights for a given class according to its percentage of representation in the dataset (from 0 to 1). On the right, the effect of Focal Loss depending on the γ value used. For the weights, it can be seen that while the standard values are linear, with the logarithmic weights the percentage of representation has much more effect, giving more importance to under-represented classes and less to over-represented ones. For Focal Loss, it can be seen that the loss of well-classified examples (higher probability of ground truth class), is reduced depending on the γ value. It should be noted that, for $\gamma = 0$ focal loss is equivalent to cross entropy loss.

A detail to take into account is the fact that when no weights are used (all equal to 1) the sum of the weights is equal to the number of classes. Along with the different weights, normalised versions with the number of classes will also be tested in order to maintain the ratio, thus having the weights centred at 1, with classes under-represented having weights greater than 1 and over-represented under 1.

4.2 The PREVENTION test set and Cross-Dataset

In order to be able to properly evaluate the results obtained in the different datasets and compare them two datasets have been developed. The first one is the PREVENTION test set, developed to externally evaluate performance and generalisation capabilities in realistic scenarios. The second one is a cross-dataset developed to alleviate biases and asses the complexity and generalisation capabilities of existing datasets.

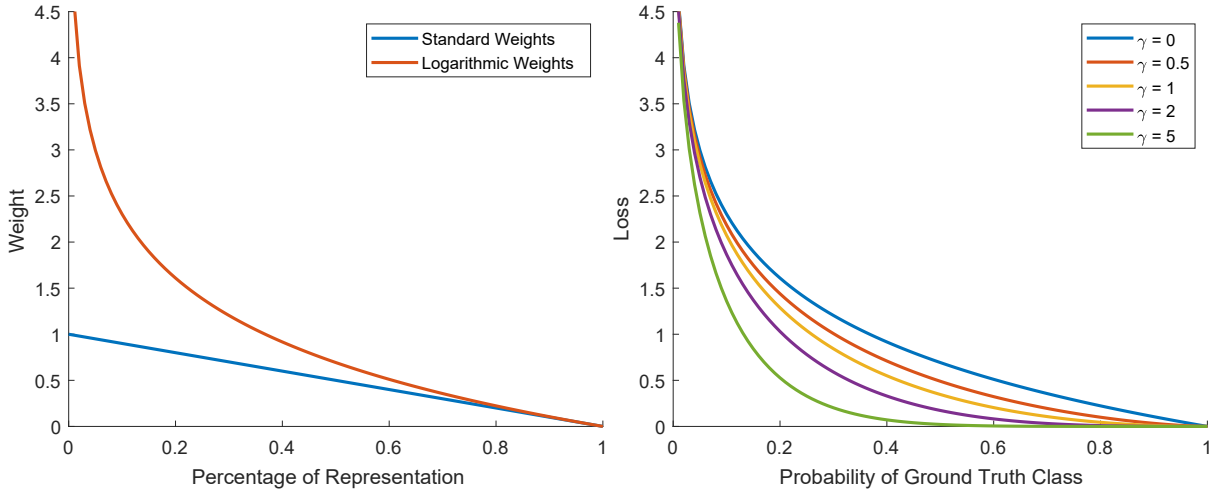

Figure 4.4: On the left, standard and logarithmic weights for a given class according to its percentage of representation in the dataset. On the right, Focal Loss effect on loss depending on the γ value used.

4.2.1 The PREVENTION test set

The PREVENTION dataset [79] is a vehicle intention prediction dataset that contains images from real-world driving scenarios. It has a total of 356 minutes of records for a distance of 540km. The images were obtained from two front and rear view facing cameras. Being a real-world driving dataset, with the cameras mounted on a vehicle, the viewpoint and nature of the images is very different when compared with those found in existing fine-grained vehicle classification datasets. Because of this, a test set based on the PREVENTION dataset has been built to externally evaluate the generalisation capabilities and performance of different datasets, architectures and techniques.

A total of 2,685 vehicles have been manually selected, of which, 1,452 are front view images and 1,233 rear view. There are 33 different makers labelled and, from these 2,685 vehicles, 1,113 have also been labelled with model (618 are front view and 515 rear view), making a total of 87 different models. The reasons why not all images have been labelled with model is that for some it has been impossible to reliably obtain the corresponding model and for others the lack of consensus among annotators. Figure 4.5 shows some sample images from the PREVENTION test set.

Figure 4.5: Example of some of the images from the PREVENTION test set.

4.2.2 Cross-Dataset

A quality dataset must capture the real world in a way as reliable as possible with the minimum amount of deviations. The higher the quality of a dataset, the better the generalisation capabilities of the models trained on it. Unfortunately, the process of building a dataset is tedious and complex. And this, without even taking into account factors such as bias. Most datasets are either designed to solve a specific problem, with the bias that implies, or intended for general use. A general-purpose dataset must reflect the world reliably, but most of them tend to be conditioned.

In order to properly assess the quality of the chosen datasets (CompCars, VMMR-db and Frontal-103), a cross-dataset composed of their common classes has been built. In this way, complexity and generalisation capabilities of the models trained with each dataset will be evaluated by performing cross-tests.

Two sets have been built, one of makers and other of models:

- **Fusion-Makers.** The Fusion-Makers set features 27 different manufacturers with a total of 265,833 images, 28,960 from CompCars, 198,644 from VMMR-db and 38,229 from Frontal-103.
- **Fusion-Models.** The Fusion-Models set features 75 different vehicle models with a total of 101,335 images, 13,211 from CompCars, 72,142 from VMMR-db and 15,982 from Frontal-103.

The fact that the set of models has a smaller number of images than the set of makers may appear to be a mistake. The reason is that the makers set is much less strict, thus allowing different models from the same manufacturer to be grouped, even though they are not present in all three source datasets. On the contrary, the models case is much stricter, as for a given model to be included in the set it has to be present in all three source datasets. Additionally, for some of the source datasets, the model classes have been grouped into a single class as they are different equipment levels from the same model, e.g. BMW 320 vs BMW 325 as BMW 3 Series. Table 4.1 shows a summary of the cross-dataset Fusion sets.

Dataset	# Classes	# Images	# CompCars (%)	# VMMR-db (%)	# Frontal-103 (%)
Fusion-Makers	27	265,833	28,960 (10.89%)	198,644 (74.73%)	38,229 (14.38%)
Fusion-Models	75	101,335	13,211 (13.04%)	72,142 (71.19%)	15,982 (15.77%)

Table 4.1: Fusion sets number of classes, images and distribution of the images between the three source datasets.

4.3 Experiments

These are the experiments that are going to be performed to address the different fine-grained vehicle classification issues:

- **Curriculum Learning.** The applicability and impact of the curriculum learning techniques on performance and per-class performance will be tested.
- **Fine-grained Models.** Multiple fine-grained models will be trained making use of the three datasets and compared with its baselines performances.
- **Weighted Losses.** The effect of the different weighted losses on per-class performance will be tested and the external PREVENTION test set used to asses generalisation capabilities.

- **Complexity and Generalisation Capabilities.** The complexity and generalisation capabilities of the models trained with each of the datasets will be assessed making use of the cross-dataset and tested with the PREVENTION test set.

Chapter 5

Results

This chapter presents the proposed experiments to address vehicle keypoint detection and fine-grained vehicle classification issues in chapters 3 and 4 as well as the results obtained. Vehicle keypoint detection results are presented and discussed in section 5.1. Section 5.2 covers the fine-grained vehicle classification experiments and analyses the results achieved. Finally, in section 5.3 the conclusions derived from these experiments are exposed.

5.1 Vehicle Keypoint Detection

This section presents the results obtained for vehicle keypoint detection with the different experiments described in section 3.3. First, data augmentation techniques will be evaluated. Once the best data augmentation solution has been identified the impact of different backbones and input resolutions will be assessed. Choosing the best combination of data augmentation, backbone and input resolution, the impact of PASCAL3D+ and an analysis of instance size effects will be performed. Finally, a keypoint distribution study will be conducted and the per-keypoint performances analysed to finish evaluating the proposed custom keypoints dataset.

5.1.1 Data Augmentation

As described in 3.1.2.2, three different data augmentation strategies have been tried. To carry out this first set of tests and evaluate the impact of data preprocessing ResNet50 backbone with 256x192 input size is used. The images used to train the models are the 1,229 from the PASCAL subset.

Table 5.1 shows the results obtained with the different data augmentation strategies. For each model, PCK and APK with $\alpha = 0.1$ metrics have been obtained. On the first place, it can be clearly seen the effect of using data augmentation compared to not using it. In the worst case, the use of data augmentation implies an improvement of 6.33% and 6.89% on PCK and APK respectively. Comparing the mild approaches, there is a significant improvement (+5%/+7% PCK/APK) when using rotations higher than $\pm 30^{o}$, with practically the same results for the $\pm 40^{o}$ and $\pm 50^{o}$ options. Having seen that the $\pm 30^{o}$ option performs worse and that the $\pm 40^{o}$ and $\pm 50^{o}$ options obtain practically the same results, the hard approach was only tried with the $\pm 40^{o}$ option. Once again, the results are pretty much the same. This is rather curious, as the hard approach applies several additional modifications to the images compared to the mild one. Given that the keypoint detection problem is mostly a problem of part detection, it is likely that the only data augmentation operations that have any real effect are those that modify the geometry of the image. Precisely, all the additional operations of the hard approach exclusively modify the visual appearance of the image, leaving the geometry unchanged.

For these reasons, the hard approach has been discarded, as it has a higher computational cost with the same performance. From now on, all experiments will take place using the $\pm 40^{o}$

Data Augmentation Strategy	PCK (%) $\alpha = 0.1$	APK (%) $\alpha = 0.1$
No data augmentation	64.09	26.77
Mild: Rot $\pm 30^{o}$	70.42	33.66
Mild: Rot $\pm 40^{o}$	**75.52**	**40.53**
Mild: Rot $\pm 50^{o}$	75.5	40.43
Hard: Rot $\pm 40^{o}$	75.41	40.2

Table 5.1: PCK and APK with $\alpha = 0.1$ for different data augmentation strategies using the images from the PASCAL subset of PASCAL3D+.

mild approach, as it performs considerably better than the $\pm 30^{o}$ option and there is no need to go further in the rotations with the $\pm 50^{o}$ option. With these experiments, the usefulness and necessity of data augmentation has been demonstrated, achieving an improvement of 11.43% and 13.76% for PCK and APK respectively.

5.1.2 Backbone and Input Size

After having evaluated the best data augmentation approach the impact of the different backbones and input resolutions have been studied. Once again, the images used to train the models are the 1,229 from the PASCAL subset. All three ResNet backbones (50, 101 and 152) are used with both input sizes (256x192 and 384x288).

Table 5.2 shows the results obtained with each backbone and input size configuration. Here, two different effects have to be analysed. On the one hand, the impact of increasing the input size, on the other hand, the impact of using a deeper backbone. Going first with the impact of using a deeper backbone, a small performance improvement can be observed when going from ResNet50 to 101 and 152, with 75.52%, 75.76% and 76.54% PCK respectively. It is when changing the input resolution that a greater effect is appreciated. With the enhanced resolution, the ResNet50 backbone is capable of outperform the ResNet152 one with the standard resolution while matching the computational cost. Once again, the best performance is achieved by the deeper backbone, with a PCK of 82.18%, 82.75% and 83.17% for the ResNet50, 101 and 152 respectively. As in the standard input size case, the performance difference between models remains.

Backbone	Input Size	PCK (%)	APK (%)	Training Time
ResNet50	256x192	75.52	40.53	35m
ResNet50	384x288	82.18	48.09	1h 3m
ResNet101	256x192	75.76	41.76	46m
ResNet101	384x288	82.75	49.8	1h 29m
ResNet152	256x192	76.54	45.38	1h
ResNet152	**384x288**	**83.17**	**50.69**	**1h 56m**

Table 5.2: PCK and APK with $\alpha = 0.1$ for different backbones and input sizes using the images from the PASCAL subset of PASCAL3D+

Focusing on the computational cost of training each model, the same behaviour as in PCK and APK can be seen, with a greater impact when increasing the input resolution. While changing the backbone has a computational cost, in terms of training time, of 30% more for ResNet101 than ResNet50, and another 30% more for ResNet152 than ResNet101, the change in resolution costs 80% more for ResNet50 backbones and 93% more for ResNet101 and ResNet152 backbones.

With all this information various conclusions can be extracted. First, in a resource-constrained environment, is much better to increase the input resolution than using a deeper backbone. Second, as expected, a deeper, more capable backbone achieves better results. Lastly, if there are no resource constraints, the best option is to combine the deeper backbone with the enhanced resolution input. By combining these two options, PCK and APK have gone from 75.52% to 83.17% and from 40.53% to 50.69%, an increase of 7.65% and 10.16% respectively. That said, with a 331% increase in training time. Figure 5.1 shows a comparison of the input size and the output heatmaps of the ResNet50 256x192 and the ResNet152 384x288. It can be seen that for the correct keypoints the responses are stronger for the ResNet152 384x288 model and slightly weaker for the incorrect ones.

Figure 5.1: Visual comparison of the input size and output heatmaps for ResNet50 256x192 and ResNet152 384x288 models. On top the input images, in the middle the ResNet50 256x192 output heatmaps and on the bottom the ResNet152 384x288 output heatmaps.

For all these reasons, subsequent experiments will be performed using the ResNet152 backbone with 384x288 input resolution.

5.1.3 PASCAL3D+

So far, all models have been trained exclusively using the 1,229 images from the PASCAL subset of PASCAL3D+. Here, the impact of using the 5,475 images from ImageNet and the full PASCAL3D+ dataset is evaluated. To do so, three different models have been trained. Table 5.3 shows the results obtained with these models. A considerable leap in performance can be seen, going from 83.17% to 98.98% PCK when using ImageNet subset or to 97.12% PCK when using both subsets in combination.

Train Data	Validation Data	PCK (%)	APK (%)	Training Time
PASCAL	PASCAL	83.17	50.69	1h 56m
ImageNet	ImageNet	98.98	74.51	4h 44m
PASCAL+ImageNet	PASCAL+ImageNet	97.12	72.38	6h 27m
ImageNet	PASCAL	76.26	45.28	4h 44m
PASCAL+ImageNet	PASCAL	88.49	55.09	6h 27m

Table 5.3: PCK and APK with $\alpha = 0.1$ for the different subsets of PASCAL3D+. All runs with ResNet152 backbone and input size of 384x288.

This large difference in numbers may seem worrying, but looking at the characteristics of each subset clears any doubts. While PASCAL subset has 2,161 instances in 1,229 images, the

ImageNet subset has 5,630 instances in 5,475 images. With this data, is evident that occlusions and overlaps are much more common in the PASCAL subset, making it more complex. But, this is not the only difference between PASCAL and ImageNet subsets. The size of the instances is also relevant. Focusing on this, it can be seen that ImageNet instances have a considerably larger area than the ones from PASCAL, which accentuates the difference in complexity. Figure 5.2 shows a comparison of the instances area for PASCAL, ImageNet and full PASCAL3D+ dataset.

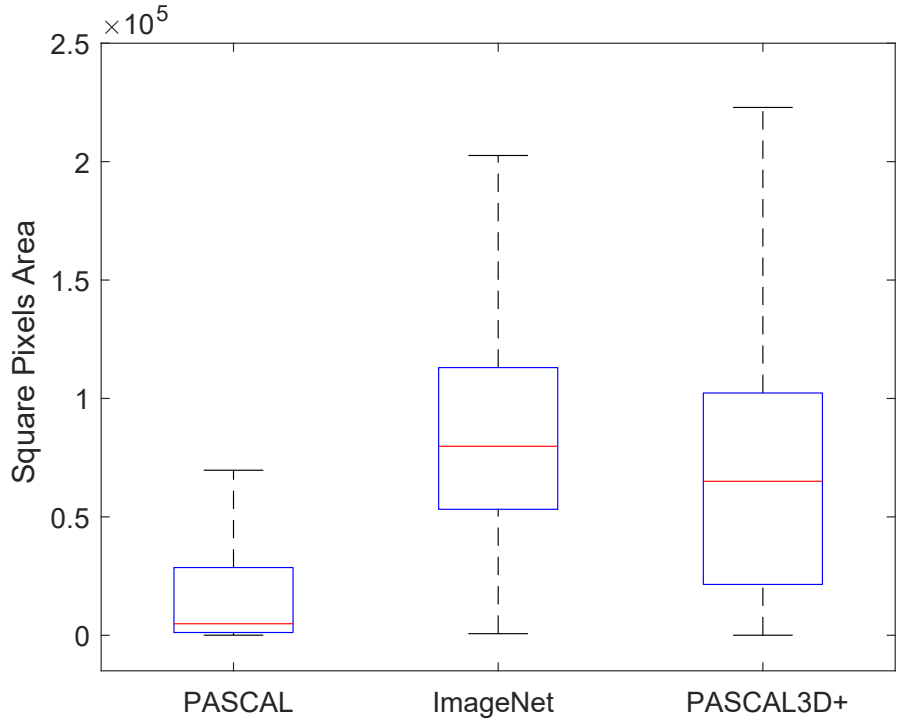

Figure 5.2: Instances area distribution in square pixels for each subset of PASCAL3D+ without the outliers.

Having found that, at least in part, the big difference in performance can be justified by the difference in complexity, the other remaining option is overlearning. To discard this, ImageNet and PASCAL3D+ models have been validated with PASCAL data alone. First, a drop in performance from 83.17% to 76.26% PCK can be seen in the ImageNet model, which is almost certainly caused by the difference in complexity. However, in the case of the PASCAL3D+ trained model, a considerable improvement over training with PASCAL data alone can be seen. PCK has raised from 83.17% to 88.49% and APK from 50.69% to 55.09%. With these results, not only the overlearning option can be discarded, but it is also clear that the joint use of ImageNet an PASCAL has a positive effect, improving generalisation capabilities of the model. With regard to training times, the difference is justified by the number of images used, with each epoch taking longer to complete the more images there are.

Once the best data augmentation strategy, the optimal combination of backbone and input resolution, and the impact of the dataset have been determined, it is time to compare the results with previous approaches. Table 5.4 shows a comparison of previous approaches like the ones from [32, 33, 46, 53] with our method, SBVPE. [32, 33, 46] methods use PASCAL VOC data while [53] use PASCAL3D+. Our results are reported with the PASCAL model and the PASCAL3D+ model validated with PASCAL and PASCAL3D+.

Comparing SBVPE with the previous methods, it can be seen that it outperforms them, regardless the subset used, with the exception of the method presented by Murphy et al. [53], which makes use of PASCAL3D+ dataset. In any case, our worst performing model performs better than the others, with the previous exception, having solid and consistent results, not

Method	Input Size	PCK (%)	APK (%)
Long et al. [46]	227x227	45.7	-
Tulsiani et al. [32]	384x384 + 192x192	81.3	40.7
Li et al. [33]	64x64	81.8	45.4
Murthy et al. [53]	64x64	93.4	-
SBVPE-PASCAL/PASCAL	384x288	83.17	50.69
SBVPE-PASCAL3D+/PASCAL	**384x288**	**88.49**	**55.09**
SBVPE-PASCAL3D+/PASCAL3D+	384x288	97.12	72.38

Table 5.4: PCK and APK with $\alpha = 0.1$ of different methods. SBVPE trained and validated with PASCAL, PASCAL3D+/PASCAL and PASCAL3D+ respectively. All 3 methods are the ResNet152 backbone.

to mention APK, which is almost 5% better. In the case of [53], our PASCAL3D+ model outperforms it. Focusing on PASCAL results, our PASCAL3D+ model reports a PCK of 88.49% and an APK of 55.09%, achieving an improvement of 6.69% and 9.69% for PCK/APK with respect to Li's model.

With these results it can be said that SBVPE is a robust approach, with solid and consistent results, achieving good PCK and APK, and, as it has been trained with a higher input resolution, is able to make better use of the additional information available by using higher resolution images. Figure 5.3 shows some output examples of the SBVPE-PASCAL3D+/PASCAL model.

Figure 5.3: Output samples of SBVPE-PASCAL3D+/PASCAL model.

5.1.4 Instance Size

As has been shown in the previous experiment, PASCAL and ImageNet subsets are considerably different in terms of complexity. While PASCAL images have a mean of 1.76 instances per image, ImageNet ones have only 1.03 instances per image. But this is not the only difference, there is also a significant variation in the instance size, as Figure 5.2 shows. Table 5.5 shows in numbers these differences. It is only by carefully analysing this information that the complexity variations between both subsets are clear, having ImageNet a median area 15 times greater and a mean area 4 times greater than PASCAL.

It is evident that the difficulty of detecting the keypoints depends on the area of the instance. Therefore, and taking into account the important variations in instance size of the PASCAL and ImageNet subsets, it is particularly interesting to analyse the effect that this variations have on performance, as well as that of overlaps and occlusions. To do so, validation data has been

Subset	# Instances	# Outliers	Median Area	Mean Area
PASCAL	2,161	288	5,265	26,695
ImageNet	5,630	348	79,810	107,630
PASCAL3D+	7,791	283	66,515	85,181

Table 5.5: Median and mean instance area in square pixels for PASCAL3D+ dataset and its subsets.

divided following the setup described by Tulsiani et al. in [32]. Table 5.6 shows a comparison of the performance of different methods on this division. Full results correspond to the complete validation set, occluded are the instances marked as truncated or occluded (739 instances in PASCAL, this information is not available for ImageNet), small are the smaller third of the data (357 instances in PASCAL and 932 in ImageNet) and big the bigger third (357 instances in PASCAL and 932 in Imagenet). SBVPE outperforms previous approaches in all categories.

PCK (%)	Full	Occluded	Small	Big
Tulsiani et al. [32]	81.3	62.8	67.4	90.0
Li et al. [33]	81.8	59.0	74.3	87.7
SBVPE-PASCAL3D+/PASCAL	**88.49**	**84.07**	**83.85**	**92.4**
SBVPE-PASCAL3D+/PASCAL3D+	97.12	84.07	92.31	98.6

Table 5.6: PCK with $\alpha = 0.1$ of different methods. Full results correspond to the complete validation set, occluded to the objects marked as truncated or occluded, small are the smaller third of the data and big the bigger third. SBVPE trained and validated with PASCAL3D+/PASCAL and PASCAL3D+/PASCAL3D+ respectively. Both methods are the ResNet152 backbone with 384x288 input size.

Analysing the results category by category, our approach represents a huge leap in performance, with an increase of more than 21% for occluded data, 9.55% for low resolution objects, and 2.4% for high resolution ones, reaching 84.07%, 83.85% and 92.4% PCK respectively. It is curious that the model has a slightly better performance for occluded instances than for low resolution ones. Having a good performance for these two types of instances (occluded and low resolution) is critical in real scenarios, as occlusions happen constantly and distant objects are small. Focusing on the results obtained with the full PASCAL3D+ dataset, the performance is better than when exclusively validating with PASCAL data. It is important to keep in mind that ImageNet subset has, by far, more images than PASCAL subset. This, together with the higher resolution contributes to dilute the metrics. Because of this, the small data in reality is not so small, and the big one is really big, making easier the keypoint detection (occluded results are the same as there are no ImageNet images labelled with this information).

These experiments have allowed us to verify that the proposed model not only has a good performance, but that it is also independent of the type of instance, obtaining competent results both for occluded and low resolution instances, which makes it a perfect candidate for real world use.

5.1.5 Keypoint Distribution Study

One of the most important parts of keypoint detection is the chosen keypoints. As previously mentioned, there is no consensus on the labelling process. This poses a problem when comparing methods trained on different datasets. In addition, there is a lack of analysis of the suitability of the chosen keypoints. So far, the results obtained tell us about the overall performance of the models, the raw performance. It is of great importance and relevance to perform a detailed analysis of the performance at keypoint level.

Remembering the 12 PASCAL3D+ keypoints, these are: *the four wheels, windshield's and*

rear window's upper corners, headlights and left/right side of the trunk. The first thing is to know the distribution of the keypoints in the dataset. Figure 5.4 shows the amount of each keypoint in PASCAL and ImageNet subsets and in PASCAL3D+ full dataset.

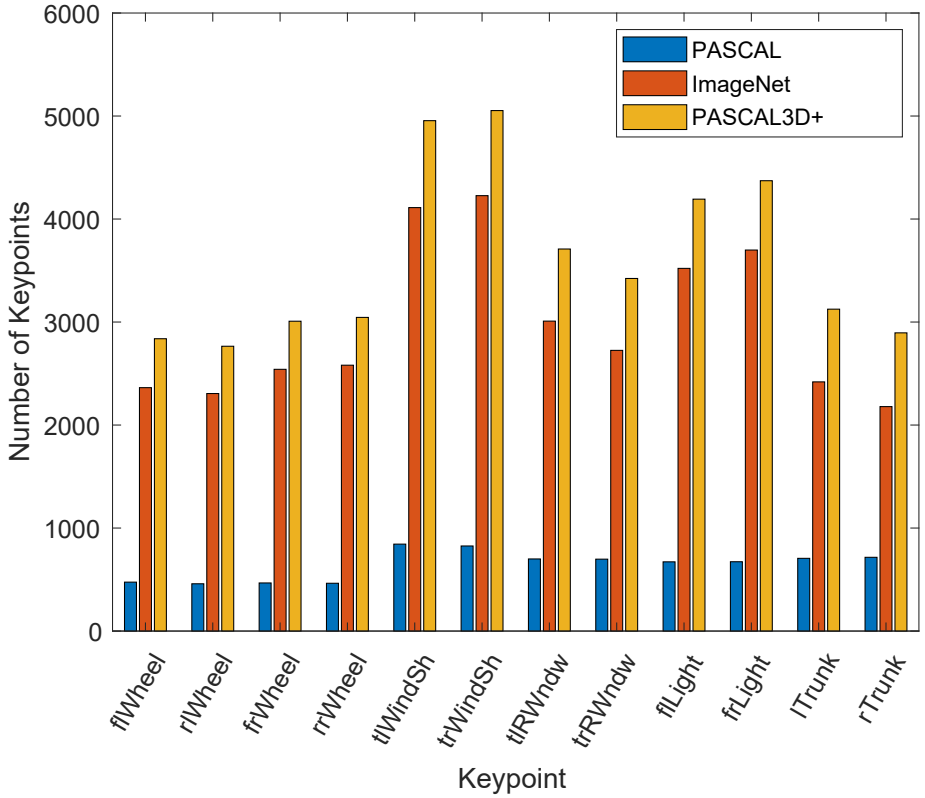

Figure 5.4: Amount of each keypoint on PASCAL3D+ dataset and its subsets. From left to right: front left wheel, rear left wheel, front right wheel, rear right wheel, top left windshield, top right windshield, top left rear window, top right rear window, front left light, front right light, left trunk and right trunk.

Relevant information can be obtained by looking at the keypoint distribution. The amount of keypoints is quite homogeneous among the subsets maintaining the proportions. The distribution of left/right view instances is practically balanced (similar amount of left and right keypoints), with a slight predominance of the right side, and front view instances are predominant over rear view ones (more windshield and headlights keypoints than rear window and trunk).

Once the keypoint distribution is know, is time to evaluate the performance for each keypoint individually. Figure 5.5 shows the per-keypoint PCK and APK with $\alpha = 0.1$ for PASCAL and PASCAL3D+. Focusing on PCK, for both models, the best performing keypoints are the wheels, while the remaining keypoints have a clear relation between the number of samples and the performance. With PCK results, one might think that the system has an excellent performance. It is only when looking at APK results that the truth is unveiled. While a pattern similar to that of PCK can be observed for all keypoints, the wheels, that were the best performing keypoints in PCK terms, have the worst APK results. Thinking for a moment about PCK and APK metrics, PCK measures the ability of the model to accurately detect the labelled keypoints, and APK measures the ability to accurately detect the present keypoints. For example, in the case of wheels, only one side of the vehicle can be visible, but the model may be detecting the four wheels. For PCK, if the two visible wheels, which are the labelled ones, are correctly detected that is all, but for APK, detecting the two extra wheels penalises performance. With this in mind, the most probable cause for the difference in PCK and APK results is that the model is having issues detecting non-present keypoints.

To verify this theory, several tests were performed and results showed that, indeed, the model is having trouble with the wheels and, to a lesser extent, with the windshield/rear window, which

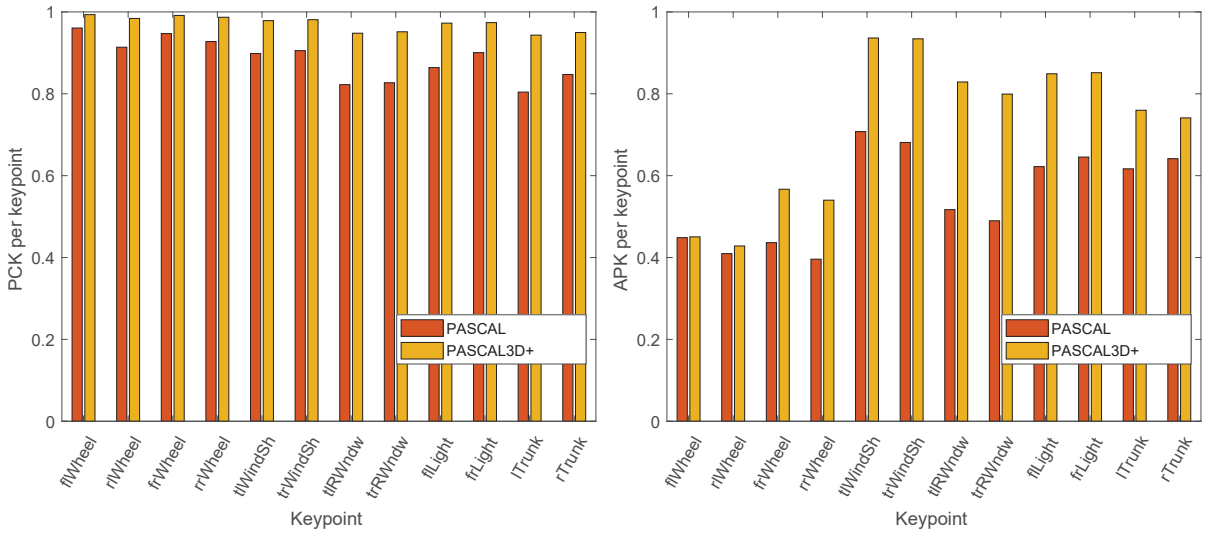

Figure 5.5: Per-keypoint PCK (left) and APK (right) with $\alpha = 0.1$ for PASCAL and PASCAL3D+. Keypoints are, from left to right: front left wheel, rear left wheel, front right wheel, rear right wheel, top left windshield, top right windshield, top left rear window, top right rear window, front left light, front right light, left trunk and right trunk.

explains the drop in APK performance. Some examples of this phenomenon can be seen in Figure 5.6. First and second rows show the wheels confusion phenomenon with the model detecting the four wheels on the two visible ones. While the second and third columns show the detection for front and back right wheels (correct), the fourth and fifth columns show the detection of the left ones (incorrect). Third and fourth rows show the windshield/rear window confusion problem, with the model detecting the four corners in the two visible ones. In the third row the model correctly detects the windshield corners (second and third columns) while incorrectly detecting the rear window ones (fourth and fifth columns). In the fourth row the model correctly detects the rear window corners (fourth and fifth columns) while incorrectly detecting the windshield ones (second and third columns).

These experiments have shown two things. First, the importance of conducting an in depth analysis when choosing the keypoints, otherwise, situations such as these can arise. Second, the importance of using appropriate metrics to evaluate the performance, as PCK by its own does not provide enough information.

5.1.6 Custom Keypoints

As previous experiments have revealed, the proposed method has solid performance both in general terms and per-keypoint PCK. However, APK results showed some problems, especially with certain specific keypoints. This has highlighted the importance of choosing keypoints and analysing its suitability. So, the following question arises, *why these 12 keypoints?* Taking a closer look at PASCAL3D+ labels, a number of problems and questionable decisions can be seen. For example, the headlights and trunk keypoints. In old vehicles with a single optic it may seem a good option to choose its centre as keypoint but, in modern vehicles, lighting systems have evolved and have multiple and complex optics. Where should the keypoint be placed? For the trunk keypoints, given the wide variety of vehicle types this does not seem like a good option at all, and proof of this is the fact that there are inconsistencies in the labels provided, with high variability in the position of these keypoints. Additionally, as previously mentioned, the model will never be able to see all four wheels of the vehicle at the same time. Therefore, it might be a good option to use exclusively two wheel keypoints, one for the front wheel and another for the rear wheel and obtain the lateral information by the context with the rest of the keypoints.

Taking all of the above into account, and with the aim of solving the shortcomings of PAS-

Figure 5.6: Output heatmaps for SBVPE-PASCAL3D+/PASCAL. Examples of the keypoint confusion phenomenon for the wheels (first and second row) and the windshield/rear window (third and fourth rows). For the first and second rows the second and third columns show the correct wheels prediction while the fourth and fifth columns show incorrectly predicted ones. For the third row the second and third columns show correctly predicted windshield corners while the fourth and fifth columns show incorrectly predicted rear window corners. For the fourth row second and third columns show incorrectly predicted windshield corners while the fourth and fifth columns show correctly predicted rear windows ones.

CAL3D+ dataset, an upgrade to PASCAL3D+ has been performed by re-labelling part of it. As described in section 3.2, we have focused on the ImageNet images, as they have higher quality, easing the labelling process. Along with the ImageNet images, the diversity has been enhanced by the addition of images from CompCars and the PREVENTION dataset. A total of 4,042 images with 4,080 instances have been labelled with 19 keypoints. The instances are 2,801 from the PASCAL3D+ ImageNet subset, 898 from CompCars and 381 from the PREVENTION dataset.

The chosen keypoints are: *front and rear wheels, the four windshield corners, the four rear window corners, left and right fog lights, left and right rear mirrors, the four license plate corners, and the logo.* As previously explained, the side of the wheels has been removed. Windshield and rear window keypoints have been extended to include the four corners, as they serve to better characterise the upper vehicle structure. Headlights have been replaced by fog lights, which are simpler. Rear view mirrors have been added, as they provide information about the lateral limits of the vehicle and are easily distinguishable. Finally, the four license plate corners and the logo, as they are elements that can be useful for other problems like surveillance or fine-grained classification.

Once the custom keypoints dataset has been introduced, is time to evaluate its performance. Table 5.7 shows a comparison of the best performing PASCAL3D+ model validated with PASCAL and with PASCAL3D+ and the model trained with the new custom keypoints dataset. As can be seen, the custom trained model outperforms the PASCAL3D+ trained ones, with an improvement of 10.34% and 1.71% respect the PASCAL and PASCAL3D+ validations, reaching 98.83% PCK. For APK the improvement its even greater, reaching 81.82%, 26.83% and 9.54% more than PASCAL and PASCAL3D+ respectively. It is true that the comparison with the

PASCAL validated model is not fair, as it is a more complex and difficult set. In any case, our proposal has comparable complexity with the full PASCAL3D+ dataset, which allows us to state that, in the absence of the per-keypoint analysis, our 19 keypoints are a significant improvement over the 12 from PASCAL3D+.

Method	**Train/Val**	**PCK (%)**	**APK (%)**
SBVPE	PASCAL3D+/PASCAL	88.49	55.09
SBVPE	PASCAL3D+/PASCAL3D+	97.12	72.38
SBVPE	**Custom**	**98.83**	**81.92**

Table 5.7: PCK and APK with $\alpha = 0.1$. Comparison of performance between PASCAL3D+ keypoints and our keypoints. All 3 methods are the ResNet152 backbone with 384x288 input size.

As has been done with PASCAL3D+, it is very interesting to study the distribution of the new keypoints, compare them with the old ones, and evaluate their individual performance. Figure 5.7 shows a comparison of the amount of keypoints on PASCAL3D+ and our custom keypoints dataset. The matching keypoints are the wheels, which have been grouped in the case of PASCAL3D+ for the comparison, the top corners of the windshield and rear window and the fog lights, which are compared with the headlights from PASCAL3D+. As expected, having PASCAL3D+ more instances, its amount of keypoints is higher. Regarding keypoints distribution, the same proportions are maintained in the custom dataset.

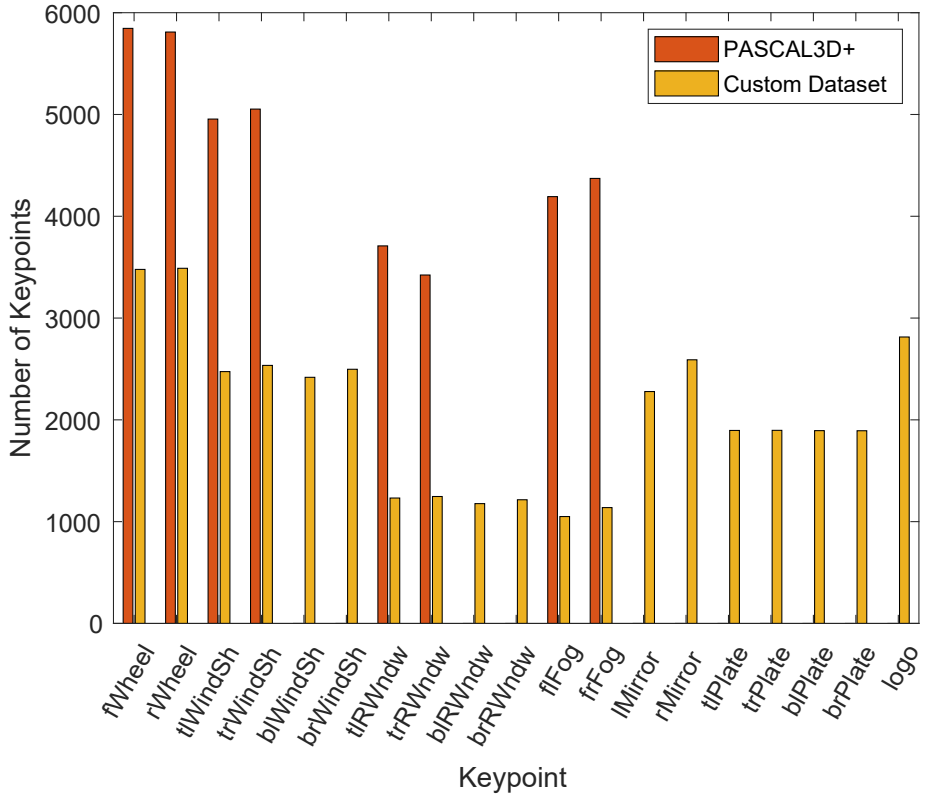

Figure 5.7: Amount of each keypoint on PASCAL3D+ dataset and our custom dataset.

Once again, knowing the keypoint distribution, is time to evaluate the performance for each keypoint individually. Figure 5.8 shows the per-keypoint PCK and APK with $\alpha = 0.1$ for PASCAL3D+ and the custom dataset. PCK shows consistent, almost perfect, results. Regarding APK, the same behaviour as in PASCAL3D+ is seen, with a clear correlation between the amount of keypoints and the corresponding APK.

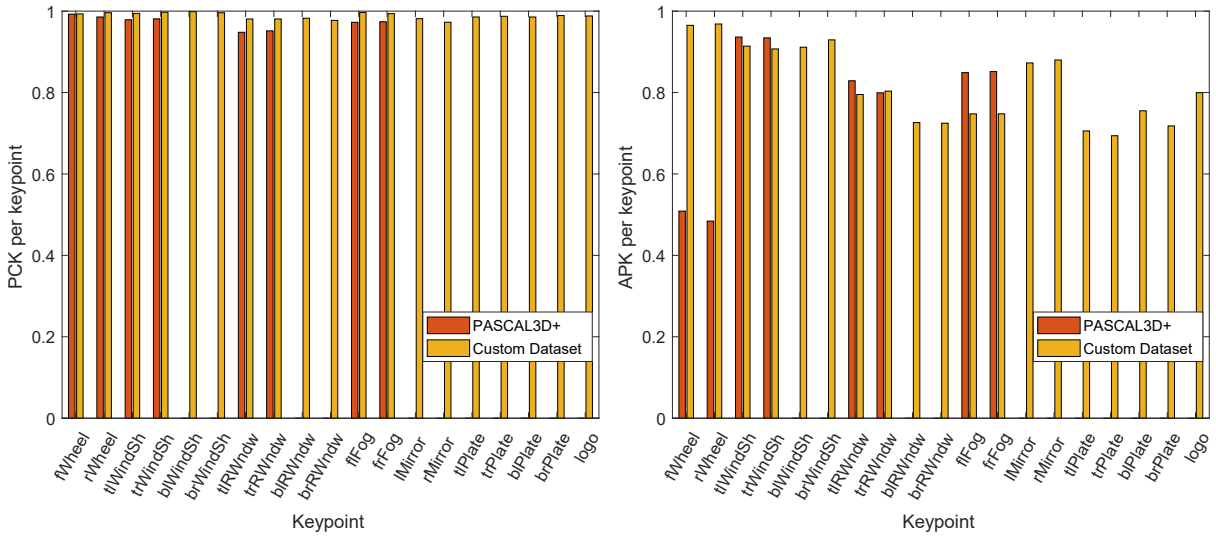

Figure 5.8: Per-keypoint PCK (left) and APK (right) with $\alpha = 0.1$ for PASCAL3D+ and the custom keypoints dataset. Keypoints are, from left to right: front wheel, rear wheel, top left windshield, top right windshield, bottom left windshield, bottom right windshield, top left rear window, top right rear window, bottom left rear window, bottom right rear window, left fog light, right fog light, left mirror, right mirror, top left plate, top right plate, bottom left plate, bottom right plate and logo.

Focusing on the problems detected in PASCAL3D+ keypoints, we proposed to move from the four wheels keypoints to a two keypoint approach, differentiating only between front and rear. Results speak for themselves, showing that our proposal is solid, with an APK more than 40% greater. Continuing with the windshield and rear window corners, consistent performance is achieved, with results very similar to those of PASCAL3D+ for the top corners and equivalent for the bottom ones. There is a noticeable difference in APK between the windshield and the rear window, with the latter having a worse result. The most likely reason is the difference in the number of keypoints. Regarding the lights, our proposal achieves a slightly worse result, but it is necessary to take into account the huge difference in number of keypoints, having the custom dataset 4 times less fog lights than PASCAL3D+ headlights. Finally, the rear view mirrors, the license plate and the logo achieve good APK results, with the mirrors being the best. We do not know the reasons for the "low" APK obtained by the license plate or the disparity of results between the 4 corners having the same number of samples. With regard to the logo, although it has an APK of 80%, we believe that this results could be better, and the fact that practically all the makers put the logos on the wheels is influencing.

Figure 5.9 shows some examples of correct and incorrect predicted keypoints from our custom dataset. In the case of incorrect predictions, it can be seen how for the first car, the model is detecting the keypoints in other car. For the second car the model is having issues with the license plate and the bottom corners of the rear window, which, it must be said, is rather strange. The third vehicle has an extreme point of view, which can justify the behaviour of the system. Finally, for the last vehicle the problem is in the lights. While this particular vehicle does not have fog lights, the model is detecting one keypoint in the area where they could be and the other in the headlight.

These experiments have shown the importance of evaluating the suitability of the chosen keypoints. Results support our labelling proposal, achieving performance levels equivalent or even better than the ones from PASCAL3D+.

Figure 5.9: Examples of predictions from our custom dataset. Top and bottom row are good and bad predictions respectively.

5.2 Fine-grained Vehicle Classification

This section presents the results obtained for fine-grained vehicle classification with the different experiments described in section 4.3. First, curriculum learning techniques applicability and effects on overall and per-class performance will be tested. After that, the three main datasets will be used to train multiple fine-grained models and compare the achieved results with its baselines performances. Then, the effect on per-class performance of using different weighted losses will be tested and evaluated with the external PREVENTION test set, with which the generalisation capabilities will be assessed. Finally, the complexity and generalisation capabilities of the models trained with each of the datasets will be evaluated making use of the cross-dataset and tested with the PREVENTION test set.

5.2.1 Curriculum Learning

As described in section 4.1.3.1, one way to cope with dataset bias and the impact of the varying difficulty of the classes is to use various learning techniques. Among these learning techniques, the ones that are going to be used are *Incremental Learning* and *Progressive Learning*. These experiments have taken place using the 431 classes and 52,083 images of CompCars.

5.2.1.1 Incremental Learning

As previously described, *incremental learning* consists in first train an easier more general problem, and after that, retrain for the more specific task. In this case, this is to first train the model for make classification and after that, retrain for model classification. All the models trained for this experiment have been trained for 50 epochs using a learning rate of 0.001 with constant learning rate policy.

Table 5.8 shows a comparison of a standard trained ResNet50 and InceptionV3 with its counterparts trained making use of *incremental learning*. Solid performance can be seen with a slight improvement for *incremental learning* models. This indicates that the *incremental learning* method is working, and, although the training time doubles, as two networks have to be trained to achieve these results, this techniques could be useful to enhance generalisation. Given that the results for ResNet50 and InceptionV3 are practically the same, having ResNet50 an slightly better performance and being less complex, the following analysis will be made using exclusively ResNet50.

As has been said, one of the main problems of fine-grained classification methods is that only general results are reported. With these results alone, one may think that the *incremental learning* models are better than the standard ones, just because they achieve a better accuracy, but this does not gives us any information about the real effect of this technique. A per-class performance analysis has to be made. Figure 5.10 shows the per-class performance differences between the standard and the *incremental learning* trained ResNet50.

Model	Method	top1/5 acc (%)	Training Time
ResNet50	standard	95.28 / 99.26	3h5m
ResNet50	incremental learning	95.55 / 99.42	6h3m*
InceptionV3	standard	95.02 / 99.10	4h25m
InceptionV3	incremental learning	95.39 / 99.29	8h53m*

Table 5.8: CompCars validation accuracy comparison of the standard models and the *incremental learning* ones. Times for the *incremental learning* method models (marked with *) are roughly double their standard counterparts because the time required for training the maker and model networks are taken into account.

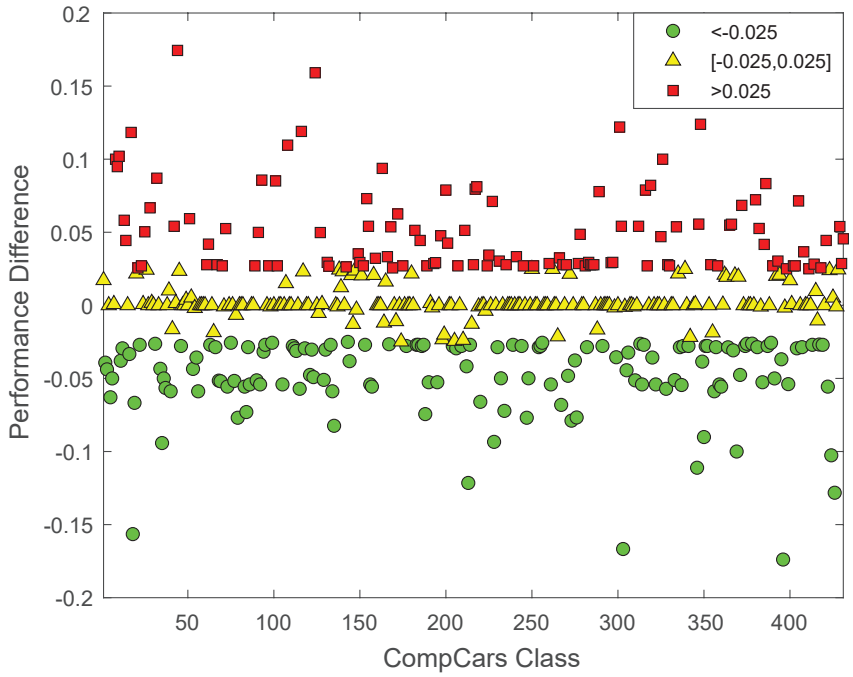

Figure 5.10: Per-class performance differences for the ResNet50 standard and *incremental learning* models. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *incremental learning* method. Differences above 0.025 (red squares) mean better performance for the standard model. Values in between (yellow triangles) mean similar performance in both models.

In order to be able to analyse the results more clearly, per-class performances have been subtracted and a ±2.5% performance difference threshold has been applied. In this way, the results are centred on zero, with classes above 2.5% (red squares) meaning best performance for the standard model, classes below -2.5% (green circles) meaning best performance for the *incremental learning* model and classes between -2.5% and 2.5% (yellow triangles) meaning equivalent performance. It can be seen that most classes obtain similar results or have minimum differences and that there are some outlayers with differences of more than 10%. From the 431 CompCars classes, 176 have similar performances, 116 perform better with the standard model and 139 perform better with the *incremental learning* approach. Of the latter two, classes that perform better with the standard model have, on average, a 5.13% better performance, while the ones performing better with the *incremental learning* model have, on average, a 4.79% better performance. With this information, it can be stated that *incremental learning* improves the performance of more classes than it worsens. However, the classes that are negatively affected worsen on average more than the other classes improve. In any case, the enormous variations of more than 10% for some classes made us wonder if there is a relation between these variations and the number of samples.

The relationship of per-class performance differences with the number of samples can be

observed in Figure 5.11. This gives valuable information. It can be clearly seen a pyramid pattern. This is telling that the more samples a class has, the less performance difference between the standard method and the *incremental learning* one. Meanwhile, the greatest differences are concentrated in some of the classes with the least number of samples. These results support the theory that the number of samples is related with the variations in performance between one method and the other. However, we believe that the main effect is not related with the methods themselves, but to the fact that these classes with fewer samples are more exposed to the random variations of each training.

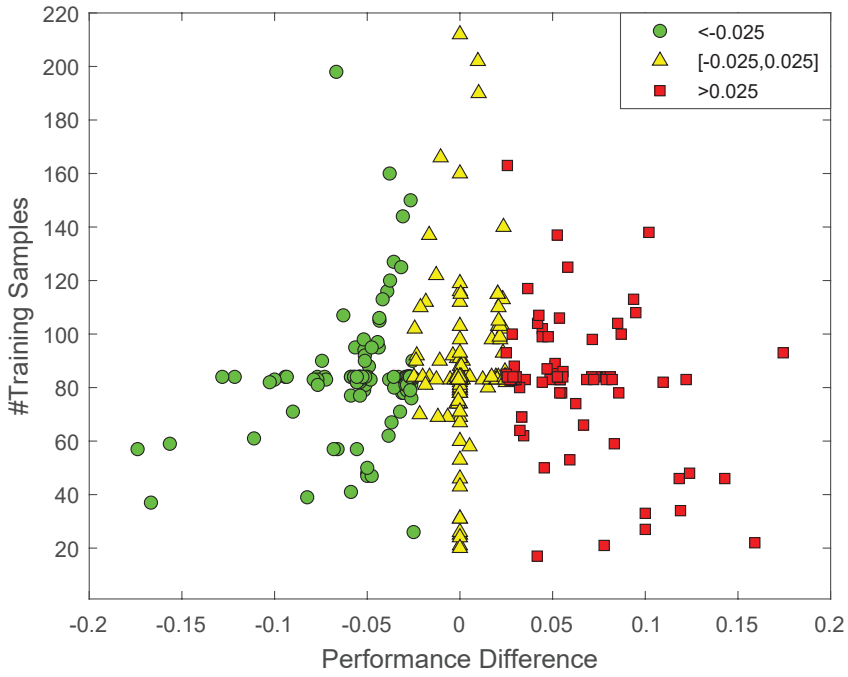

Figure 5.11: Per-class performance differences for the ResNet50 standard and *incremental learning* models depending on the number of training samples. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *incremental learning* method. Differences above 0.025 (red squares) mean better performance for the standard model. Values in between (yellow triangles) mean similar performance in both models.

5.2.1.2 Progressive Learning

Continuing with the curriculum learning experiments, *progressive learning* consist in starting with a simple, easier problem, and then, progressively increase the difficulty at each epoch. In this case, this is to start with the easier, best performing classes, and then adding 5 or 10 new classes every epoch (*progressive-5* or *progressive-10*). To do so, two standard trained ResNet50 models, one with constant, 0.001 learning rate and the other with 0.01 initial learning rate and step-10 policy, are compared with a set of ResNet50 models trained using both *progressive* variants. Table 5.9 shows the performance of these models.

Results show that, for the standard training, the best option is the one with 0.01 initial learning rate and step-10 policy. For the *progressive learning* approach the best performing option is the *progressive-10*, outperforming the *progressive-5* in both learning rates. Comparing the standard with the *progressive* runs, it can be seen that the 0.001 learning rate *progressive-10* model achieves an slightly better performance than its standard counterpart (95.43% vs 95.28%) while the 0.01 learning rate *progressive* model practically matches standard model performance (97.02% vs 97%). Regarding the training times, *progressive-10* models obtain equivalent performance to that of the standard models in less time (2h44m vs 3h5m).

These results look promising, with the *progressive* technique achieving equivalent accuracy

Model	Method	lr	top1/5 acc (%)	Train Time
ResNet50	standard	0.001	95.28 / 99.26	3h5m
ResNet50	standard	step-10 0.01	97.00 / 99.62	3h5m
ResNet50	**prog10**	**0.001**	**95.43 / 99.34**	**2h44m**
ResNet50	prog5	0.001	95.27 / 99.33	4h6m
ResNet50	**prog10**	**0.01**	**97.02 / 99.53**	**2h44m**
ResNet50	prog5	0.01	96.61 / 99.51	4h6m

Table 5.9: Comparison of the accuracy of ResNet50 models trained with standard training and *progressive learning* technique.

in less time. However, as has been said, raw performance is not everything, and attention needs to be paid to the impact at the level of individual classes. Figure 5.12 shows the comparison between the standard models and its *progressive* counterparts. On the left, per-class performance differences. On the right, per-class performance differences depending on the number of training samples. As with the *incremental* method, most classes obtain similar results or have minimum differences with the presence of some outlayers that reach differences of 20% or even 30%. From the 431 CompCars classes, the 0.001 learning rate models have 194 classes with similar performance, 118 that perform better with the standard model and 119 that perform better with the *progressive-10* model. The 0.01 learning rate models have 250 classes with similar performance, 87 with better performance with the standard model and 94 with better performance with the *progressive-10* model.

Regarding the effect of the number of samples the same pyramidal pattern can be seen, with the classes with greater performance differences being the ones with less samples. For both groups of training, the number of classes that improve and worsen is balanced, with the best performing models (the 0.01 learning rate ones) having the largest number of classes that are not affected by the approach taken. The average effect over the improving and worsening classes is also balanced, with the 0.001 learning rate models having a 4.84% and 4.71% of average difference for the classes that perform better with the standard model and classes that perform better with the *progressive* one. In the case of the 0.01 learning rate models, the average difference in performance are 4.5% and 4.62% for worsening and improving classes respectively. Once again, it seems that the main effect of the differences in performance is due to the greater impact that random training variations have over the less represented classes.

With these results, the positive effects of using the *progressive* method get diluted, as the the global accuracy of the *progressive* model is equal to the standard one, the amount of classes improving matches the worsening ones, and the average improving and worsening effects are practically the same. With all this in mind, it is worth considering whether order really matters, and if progressively increasing difficulty is working or not. In order to test this, two additional ResNet50 models have been trained with 0.01 learning rate, *progressive-10* method, and one with random class order and the order with decreasing difficulty class order.

Table 5.10 shows the comparison of the previous standard and *progressive-10* models with the new *progressive-10* models trained with random and inverse class order.

As can be seen, all models have virtually the same accuracy, with the random and inverse order ones proving that the class order has no effect. These results are unexpected, as the structure of the fine-grained classification problem suggested that these technique could improve performance. However, although no improvement has been achieved either in overall performance or in per-class performance, it has been shown that the models can be trained in a *progressive* way, achieving equivalent results in less time and allowing new classes to be incorporated to already trained models.

These experiments have allowed us to evaluate the applicability and impact of curriculum

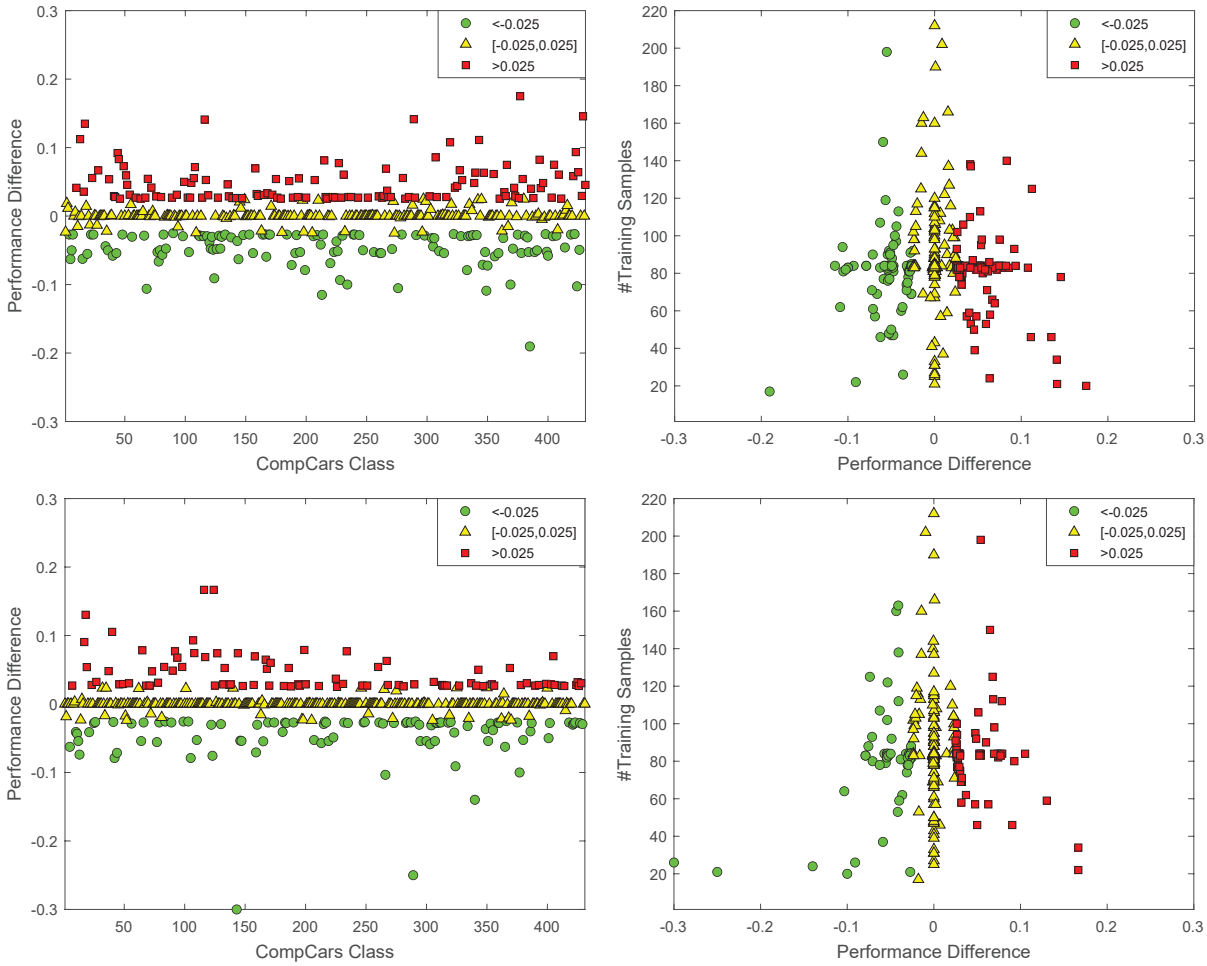

Figure 5.12: Left images are the per-class performance differences of 0.001 and 0.01 learning rate standard ResNet50 and *progressive-10* ResNet50 respectively. Right images are the per-class performance differences for the 0.001 and 0.01 learning rate trains depending on the number of training samples. Difference threshold of 0.025 (2.5%). Differences below -0.025 (green circles) mean better performance for the *progressive-10* method. Differences above 0.025 (red squares) mean better performance fort the standard model. Values in between (yellow triangles) mean similar performance in both models.

Model	**Method**	**lr**	**top1/5 acc (%)**	**Training Time**
ResNet50	standard	step-10 0.01	97.00 / 99.62	3h5m
ResNet50	prog10	0.01	97.02 / 99.53	2h44m
ResNet50	prog10-random	0.01	97.03 / 99.51	2h44m
ResNet50	prog10-inv	0.01	97.01 / 99.50	2h44m

Table 5.10: Comparison of the accuracy of ResNet50 models trained with the *progressive-10* learning technique and alternative class order (random and inverse) with standard training.

learning techniques on performance and per-class performance. *Incremental-learn* has shown an slight improvement in overall accuracy and a balanced effect in per-class performance, with similar gains and losses and a clear relation between per-class performance and the number of samples. *Progressive-learn* has practically the same behaviour, with virtually same performance and per-class performance differences regardless class order, which turned out to be irrelevant. These results make difficult to justify curriculum learning techniques as a way to improve the learning process, however, *progressive-learn* has proven to be a useful tool to reduce training time and resources needed as well as an option to add new classes to already trained models.

5.2.2 Fine-grained Models

After having evaluated the applicability and impact of curriculum learning techniques, is time to analyse the performance of fine-grained classification models trained with the different datasets and compare them with the baseline results reported by their creators. It could also be interesting to compare these results with previous works, but our aim is to prove, firstly, that it is not necessary to make use of complex models and techniques to obtain a good overall performance and, secondly, that overall performance is not everything and that there is a lack of per-class performance analysis and generalisation capabilities assessment. The datasets that are going to be used are CompCars, VMMR-db and Frontal-103.

For CompCars, two subsets have been evaluated. One of makers and other of models, with 73 and 431 different classes respectively. From VMMR-db three different subsets have been evaluated. First, the one called 3,040 is built with all the classes with more than 20 images (the authors indicate that there are 3,036 classes that meet this condition). After the 3,040 subset has been generated, one subset of makers and other of models are built grouping the classes from the 3,040 subset in the same way as is donde in CompCars. The number of classes is 43, 472 and 3,040 for the makers, models and 3,040 subsets respectively. Finally, for Frontal-103 three subsets have also been evaluated. One of makers, one of models and one of ultra fine-grained-models. The number of classes is 103, 1,050 and 1,759 respectively. Table 5.11 shows a summary of the different subsets, number of classes and number of images.

Subset	**# Classes**	**# Images**
CompCars Makers	73	52,083
CompCars Models	431	52,083
VMMR-db Makers	43	246,290
VMMR-db Models	472	246,290
VMMR-db 3,040	3,040	246,290
Frontal-103 Makers	103	65,433
Frontal-103 Models	1,050	65,433
Frontal-103 Ultra	1,759	65,433

Table 5.11: Information of number of classes and images of each of the subsets.

With each of these subsets, an InceptionV3 model has been trained for 50 epochs using a 70/30 train/val split and 0.01 initial learning rate with step-10 policy. Table 5.12 shows the results of each of these models and compares them with the baseline results reported by the authors of each dataset.

A clear tendency can be seen, with the easiest task, maker classification, being the best performing for all the datasets, followed by fine-grained classification and finally ultra fine-grained classification. For makers subsets, the results show that the best performing option is the one trained with Frontal-103, as it is the easiest of the datasets having only front view images, followed by CompCars and VMMR-db, which is the most complicated and extensive. For models subsets the best performing option is the one trained with CompCars, as its the one with the fewest classes, followed by the one trained with Frontal-103, which, although it has more classes than VMMR-db, is, as previously said, easier having a single view-point. Finally, for ultra fine-grained subsets, there is a huge difference in accuracy between the VMMR-db trained model and the Frontal-103 one. While Frontal-103 model is still capable of achieve good performance, with 95.62% top1 accuracy, the model trained with VMMR-db drops to 42.16%. This result reveals the main problem of VMMR-db dataset, its labelling process. As the process of labelling is done in a semiautomatic way, authors decided to create a class for each year for

Model	Subset	top1/5 acc(%)
InceptionV3	CompCars Makers	98.84 / 99.68
InceptionV3	CompCars Models	97.29 / 99.57
CompCars [22]	CompCars Models	91.20 / 98.10
InceptionV3	VMMR-db Makers	97.34 / 99.64
InceptionV3	VMMR-db Models	94.46 / 99.15
InceptionV3	VMMR-db 3,040	42.16 / 91.58
VMMR-db [23]	VMMR-db 3,036	51.76 / 92.90
InceptionV3	Frontal-103 Makers	99.30 / 99.87
InceptionV3	Frontal-103 Models	96.88 / 99.65
InceptionV3	Frontal-103 Ultra	95.62 / 99.48
Frontal-103 [24]	Frontal-103 Ultra	91.28 / -

Table 5.12: Accuracy comparison of the different datasets and subsets with its baseline results.

which they have images of a particular model. The problem is that these different classes most of the time are the same class. This can be seen in the top5 accuracy, which is 91.58%. In [23], the authors justify this drop in performance by the increased difficulty of going deeper in the hierarchy. However, this statement does not sufficiently hold. In the case of Frontal-103, there is indeed a drop in accuracy due to the increased difficulty, but from 96.88% top1 accuracy of models to 95.62% for ultra fine-grained models. This shows that the year-based labelling of VMMR-db is not appropriate as does not respect the real class hierarchy.

These experiments have shown that there is no need of using complex models or techniques to achieve good overall performance. All the trained models perform better than the baseline proposed by the authors of the datasets with the exception of the VMMR-db 3,040. In any case, the importance of data and a proper labelling has been seen once again, becoming clear that the labelling of one class for each year of a model of VMMR-db is not appropriate.

5.2.3 Weighted Losses

As has been said, the vast majority of work focuses exclusively on general accuracy and improving it, completely neglecting per-class performance. There is no point in having a model that achieves excellent levels of performance while it ignores a part of the classes. Therefore, per-class performance is going to be analysed and some techniques such as weighted loss used to improve performance and generalisation capabilities. VMMR-db is the chosen dataset, as it is the most complex and diverse of the previous ones. Maker and model classification will be assessed and the performance and generalisation capabilities in realistic scenarios evaluated with the PREVENTION test set.

5.2.3.1 Why raw precision is not enough?

Previous experiments have achieved an accuracy of 97.34% for VMMR-db Makers and 94.46% for VMMR-db Models. With this results one might think that the trained models work perfectly, but there is a lack of information regarding individual classes. The main problem of using accuracy for a multi-class imbalanced problem is that accuracy is a good metric to summarise the performance, but this mask the per-class results. Table 5.13 shows the results of VMMR-db Makers and Models taking into account not only accuracy, but also precision, recall and its weighted versions.

Comparing the different metrics, it can be seen that while the accuracy or the weighted

Model	Precision(%)	W. Precision(%)	Recall (%)	W. Recall(%)	Accuracy(%)
VMMR-db Makers	94.14	97.33	92.46	97.34	97.34
VMMR-db Models	89.03	94.35	85.60	94.46	94.46

Table 5.13: Extended metrics for VMMR-db Makers and Models.

precision/recall are practically identical, the average precision or recall are considerable lower. This is due to the fact that while the weighted versions of precision and recall are masking the results in the same way as accuracy, average precision and recall does not take into account the number of samples of each class, thus providing interesting information. Figure 5.13 shows an histogram of the per-class precision and recall of VMMR-db Makers and Models. As can be seen, even though both models have excellent performance in terms of accuracy, with 97.34% and 94.46% for Makers and Models respectively, there are several classes with performances in terms of precision and recall much lower. Regarding the distribution of performance both precision and recall are quite balanced.

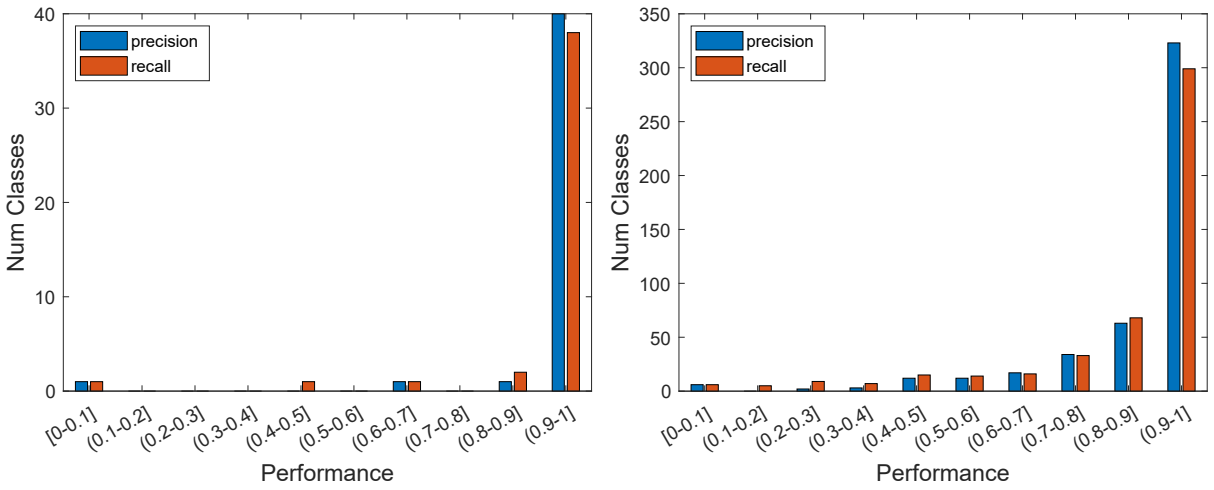

Figure 5.13: Number of classes with a given precision and recall (VALIDATION) for InceptionV3 model trained with VMMR-db Makers on the left and VMMR-db Models on the right.

Figure 5.14 shows the relation between the number of samples of each class, precision and recall. It can be seen a clear relation between the number of samples and a lower precision/recall performance. In the case of Makers, as it is an easier problem and has a lower number of classes the effect is less evident. In the case of Models this effect can be clearly appreciated. The majority of classes have a balanced performance of precision/recall, but there is a group with an excellent precision and low recall. This means that, for these classes, when the model predicts them this prediction is almost always correct, however, the low recall means that there are a lot of samples from these classes that are being incorrectly predicted, lowering the precision of other under-represented classes, or getting diluted in the performance of over-represented ones.

After having verified that there is a problem with under-represented classes, various weighted losses techniques and focal loss are going to be used to try to mitigate the effects of unbalanced data. In addition to the previous metrics, the PREVENTION test set is going to be used to evaluate generalisation capabilities of the different solutions. The PREVENTION Makers test set has a total of 33 classes, of which 25 are present in VMMR-db Makers with a total of 1,523 images. The PREVENTION Models test set has a total of 87 classes, of which 50 are present in VMMR-db Models with a total of 780 images.

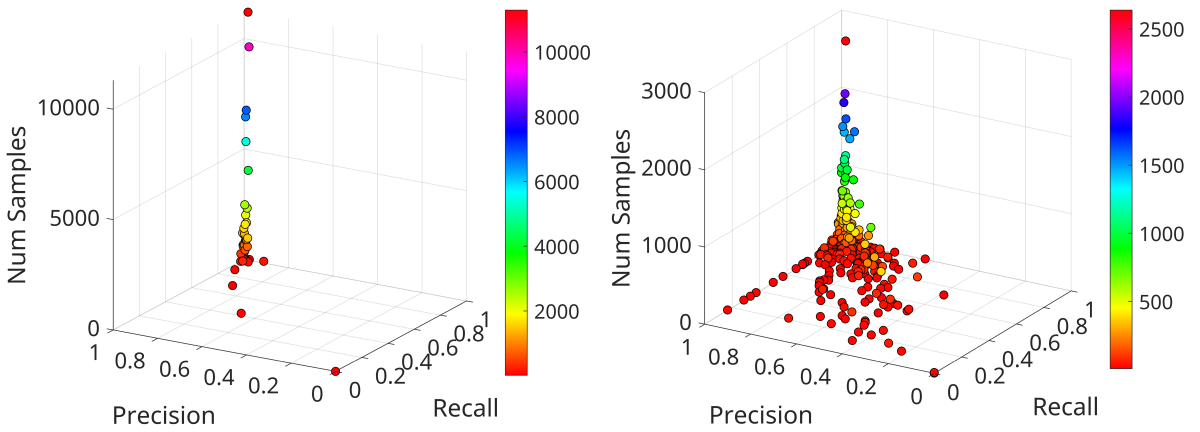

Figure 5.14: Per-class relation of number of samples, precision and recall (VALIDATION) for the InceptionV3 models trained with VMMR-db makers and models. On the left the 43 Makers, on the right the 472 Models.

5.2.3.2 Weighted losses for maker classification

Table 5.14 shows a comparison of the accuracy, precision and recall of the different weighted models and the weightless one trained with VMMR-db Makers and tested with the PREVENTION Makers test set. Note that the precision and recall results correspond to the mean unweighted value (the number of samples of each class has not been taken into account), thus providing more accurate information by considering all classes equally important. Analysing the results, in terms of validation accuracy the best model is the weightless one, but closely followed by the weighted models with the normalised standard approach (*Standard 43*) virtually matching its performance (97.34% vs 97.31%). Focusing on precision and recall, it can be seen that all the weighted approaches outperform the weightless one, which suggest that the use of weights is helping with the less represented classes. Regarding test results, *Standard 43* approach outperforms the rest of the models in accuracy, precision and recall. It can be seen how the front samples from the test set are easier, as the results are better than those of rear samples for all models. All weighted models but the *Standard* one achieve better test accuracy.

Weights	**top1/5 acc(%)**	**prec/recall(%)**	**test top1 acc(%) front / rear / all**	**test all prec/recall(%)**
None	**97.34** / 99.64	94.14 / 92.46	81.45 / 69.84 / 76.17	69.91 / 66.45
Standard	97.22 / 99.64	96.21 / 92.69	79.16 / 70.85 / 75.38	69.68 / 69.20
Standard 43	97.31 / 99.64	**96.34** / 92.97	**83.37** / **73.02** / **78.66**	**76.62** / **69.38**
Log 43	97.23 / 99.66	95.96 / **93.50**	81.81 / 69.55 / 76.23	72.16 / 66.63
Focal Loss	96.94 / 99.66	95.65 / 92.56	82.65 / 70.71 / 77.22	68.13 / 66.79

Table 5.14: Comparison of the accuracy, precision and recall of InceptionV3 models trained with VMMR-db Makers with and without weights and test on the PREVENTION Makers dataset. Standard 43 and Log 43 are the normalised version of the weights. The results of focal loss are the ones obtained using $\alpha = 1, \gamma = 2$.

With this information it seems that the weighted approach is working, with the *Standard 43* approach achieving practically identical validation accuracy, while outperforming the weightless model in terms of precision, recall and test accuracy both in validation and test, in which also improves accuracy, going from 76.17% to 78.66% for all samples. Once again, this results lack a per-class perspective. Figure 5.15 shows a comparison of per-class performance in terms of precision and recall for each of the previous models trained with VMMR-db Makers.

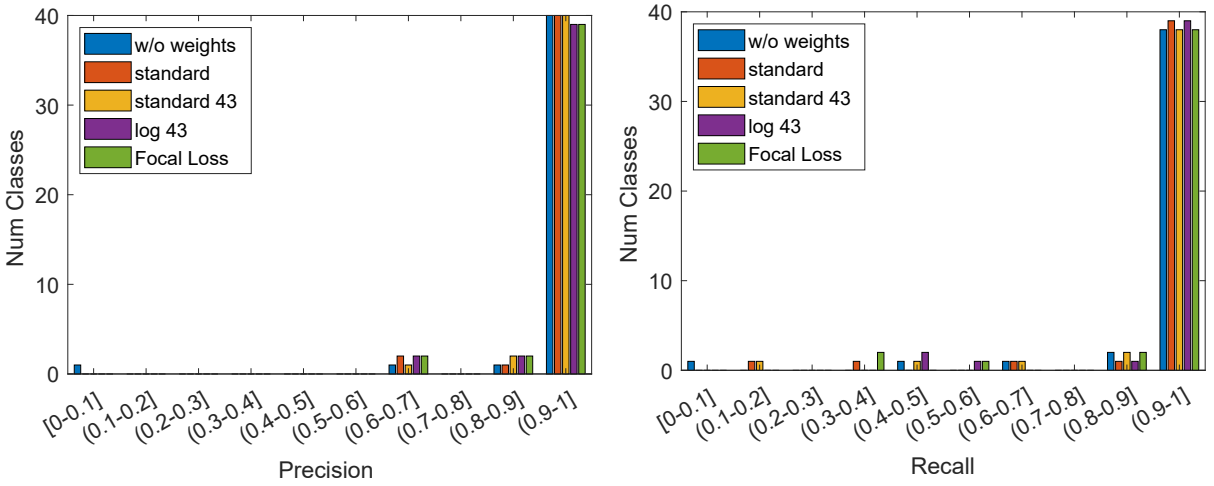

Figure 5.15: Comparison of per-class performance (VALIDATION) in VMMR-db Makers for the different weighted models. On the left precision results, on the right recall.

The effect of the weights can be appreciated, with all the weighted models improving the one class with less than 10% performance of the weightless model both for precision and recall. It is curious that while precision results are compact, recall results are more dispersed. While this may seem strange, the reason for this is that while most classes have a good precision (the predicted class is correct), the classes with less samples have a lower recall (some samples are predicted as other class). This behaviour should have an impact on the precision of the other classes, but since the incorrectly predicted class is one with a much larger number of samples the effect is diluted.

Continuing with the per-class analysis, Figure 5.16 shows a comparison of per-class precision and recall for the weighted models trained with VMMR-db Makers and tested with the PREVENTION Makers test set. As can be seen, test results are much more dispersed both for precision and recall than in validation. The best performing solution is the *Standard 43*, with a clear improvement in precision and a less evident improvement in recall on par with the *Standard* weights approach. The most probable reason for the dispersion in both precision and recall is the number of samples of the PREVENTION test set. While in validation the effect of incorrectly predicted classes gets diluted by the huge number of samples of some classes, in this case this difference in samples is not that big.

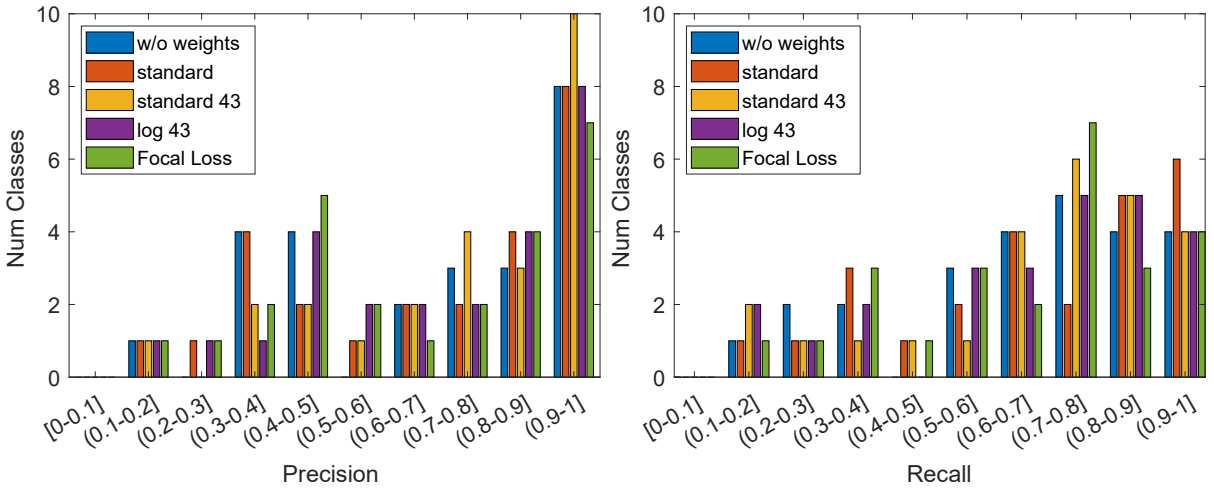

Figure 5.16: Comparison of per-class performance (TEST) in the PREVENTION Makers test set for the different weighted models trained with VMMR-db Makers. On the left precision results, on the right recall.

5.2.3.3 Weighted losses for model classification

Table 5.15 shows a comparison of the accuracy, precision and recall of the different weighted models and the weightless one trained with VMMR-db Models and tested with the PREVENTION Models test set. As with makers, the best validation performance in therms of accuracy is for the weightless model, closely followed by the weighted models. Focusing on precision and recall, this time the weightless model still achieves the best precision, with the *Log 472* approach achieving the second best precision (89.03% vs 88.85%) and the best recall (85.60% vs 86.64%). Regarding test accuracy, this time all the weighted models outperform the weightless one, with *Standard 472* and *Log 472* models tied and *Focal Loss* achieving the best overall accuracy (54.23% vs 56.03%). It is curious that the best performing model in test is also the worst performing one in validation. For test precision and recall the best performing models are *Standard 472* and *Focal Loss* respectively. It is worth noting the large drop in test performance when compared with makers. This is most likely due to the increased difficulty and the smaller number of samples of the PREVENTION Models test set, which makes it more biased.

Weights	**top1/5 acc(%)**	**prec/recall(%)**	**test top1 acc(%) front / rear / all**	**test all prec/recall(%)**
None	**94.46** / 99.15	**89.03** / 85.60	51.27 / 57.93 / 54.23	64.54 / 54.42
Standard	94.30 / 99.16	88.21 / 85.11	51.96 / 58.21 / 54.74	63.63 / 51.97
Standard 472	94.41 / 99.22	88.80 / 85.42	53.12 / 59.37 / 55.90	**70.89** / 54.52
Log 472	94.40 / 99.22	88.85 / **86.64**	**53.81** / 58.50 / 55.90	68.53 / 54.36
Focal Loss	93.92 / 99.25	88.64 / 84.61	53.12 / **59.65** / **56.03**	62.55 / **54.90**

Table 5.15: Comparison of the accuracy, precision and recall of InceptionV3 models trained with VMMR-db Models with and without weights and test on the PREVENTION Models dataset. Standard 472 and Log 472 are the normalised version of the weights. The results of focal loss are the ones obtained using $\alpha = 1, \gamma = 2$.

Figure 5.17 shows a comparison of per-class precision and recall for the weighted models trained with VMMMR-db Models. This time, the positive effects of using weights are not as evident as with makers, and the results are very similar, which makes sense as the performance is almost identical. The distribution of classes for both precision and recall is pretty balanced, with all methods compensating better performance in one section with worse in another. With these results, it may seem that the use of weights is not justified, as practically identical results are achieved and there is no benefit in terms of poorly performing classes improved. However, looking at test results a considerable improvement can be seen, going from 52.27% accuracy for front images to 53.81% with the *Log 472* weights. From 57.93% to 59.65% for rear images and from 54.23% to 56.03% for overall accuracy with the *Focal Loss* approach.

Figure 5.18 shows a comparison of per-class precision and recall for the weighted models trained with VMMR-db and tested with the PREVENTION Models test set. Once again, results are much more dispersed than in validation. This time, the best performing model is *Focal Loss* with 56.03% overall accuracy followed by *Standard 472* and *Log 472* with 55.90% accuracy. Focusing on precision results, in the range of 0.6 to 1 these three models have better or equivalent performance to the weightless model, with some cases below that are compensated. However, recall is much tighter, with equivalent results. Regarding the poor performing classes, the number of classes below 0.1 is worrying, although the number of classes is practically the same. It is important to remember that the number of samples of PREVENTION Models is the half of PREVENTION Makers, making it much more susceptible to variations.

Although these results are not spectacular and do not clearly show the benefits of the use of weights they are promising. A clear improvement in overall test performance is achieved, pointing to an improvement in generalisation capabilities.

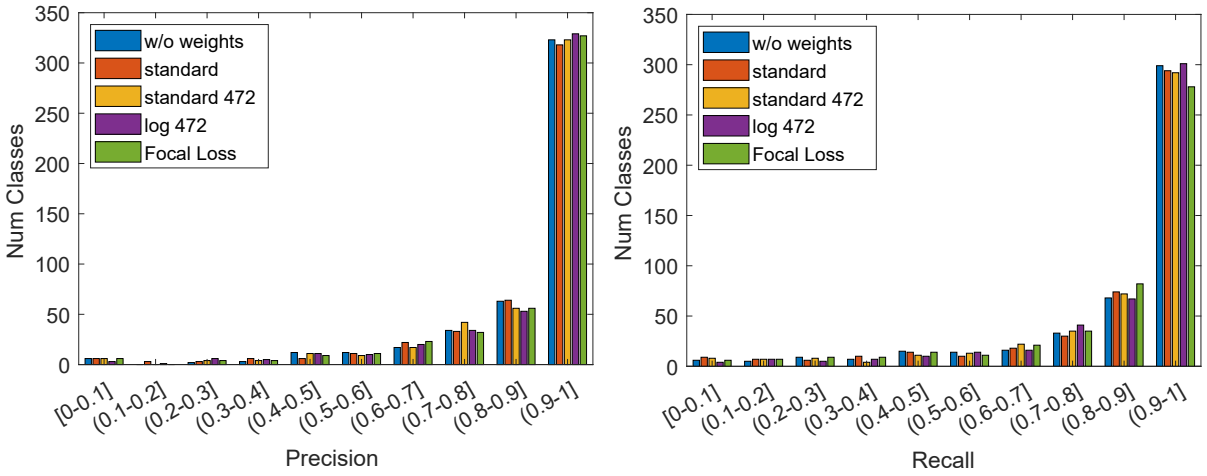

Figure 5.17: Comparison of per-class performance (VALIDATION) in VMMR-db Models for the different weighted models. On the left precision results, on the right recall.

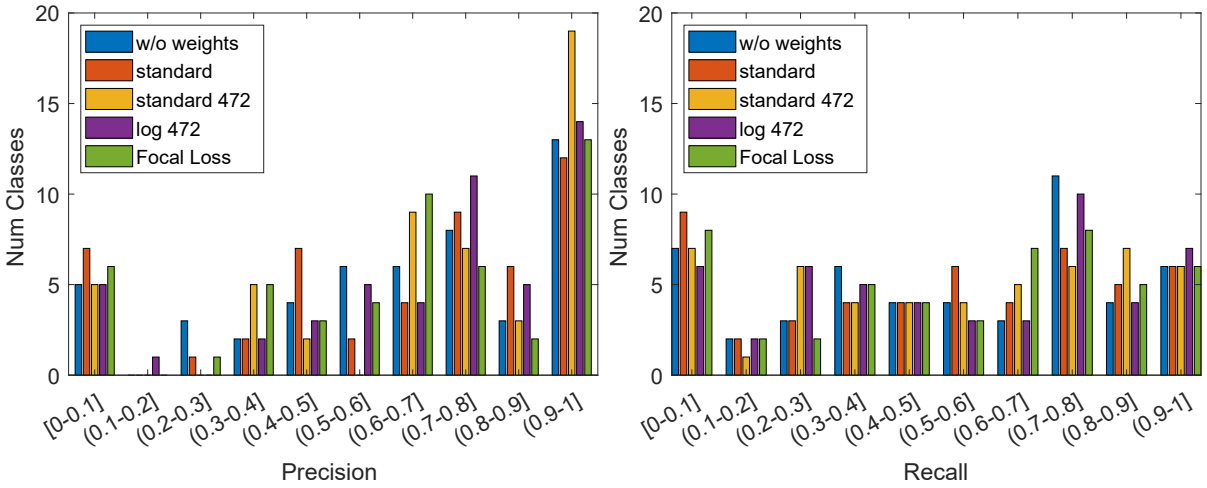

Figure 5.18: Comparison of per-class performance (TEST) in the PREVENTION Models test set for the different weighted models trained with VMMR-db Models. On the left precision results, on the right recall.

These experiments have allowed us to evaluate per-class performance and the effect of different weighting approaches. Weighted models achieve equivalent results in validation to the weightless model both for makers and models while improving overall test performance. The claim that weighted models could reduce the amount of poor performing classes is not validated, with a worrying amount of classes below 0.1 performance when testing with PREVENTION Models. However, results point to an improvement in generalisation capabilities, as test accuracy for makers achieves an improvement of 2.49%, going from 76.17% to 78.66% and a 1.8% for models, going from 54.23% to 56.03%. It is worth noting that the PREVENTION test set is very limited, with low amount of samples and classes. We strongly believe that because of this the results are not conclusive, especially for models, as the effect of the weights is more clearly appreciated in makers. It is necessary to conduct further experiments and build a more extensive and adequate test dataset that allow to extract more solid and robust conclusions.

5.2.4 Complexity and Generalisation Capabilities

So far, CompCars, VMMR-db and Frontal-103 have been used. To be considered of high quality, a dataset must capture the real world as reliably as possible, covering a wide range of situations and conditions with minimal deviations and bias. Therefore, the models trained with a good

dataset will have better performance and generalisation capabilities. As previously described, most datasets are designed to solve a specific problem, thus they are biased, or intended for generic use. A general-purpose dataset should be of high quality, providing a solid base. However, the reality is different, and most of them are conditioned and biased to some extent.

With the aim of assessing the complexity and generalisation capabilities of the trained models, a cross-dataset making use of the common classes of CompCars, VMMR-db and Frontal-103 has been constructed. Four different models are going to be trained, one for each of the source datasets and the fourth for the cross-dataset. These models will be cross-tested and externally evaluated with the PREVENTION dataset.

As described in section 4.2.2, the cross-dataset has two subsets, one of makers and one of models. The Fusion-Makers subset has 27 different manufacturers and a total of 265,833 images, 28,960 from CompCars, 198,644 from VMMR-db and 38,229 from Frontal-103. Fusion-Models subset has 75 different vehicle models and a total of 101,335 images, 13,211 from CompCars, 72,142 from VMMR-db and 15,982 from Frontal-103. The reason for the big difference in the number of samples between Fusion-Makers and Fusion-Models is that the makers subset is much less strict, allowing all images from the same manufacturer regardless the model. In contrast, makers allows only the models present in the three source datasets, thus reducing the number of samples. Table 5.16 shows a summary of the composition of Fusion-Makers and Fusion-Models. All these experiments have been performed with the InceptionV3 architecture.

Dataset	**# Classes**	**# Images**	**# CompCars**	**# VMMR-db**	**# Frontal-103**
Fusion-Makers	27	265,833	28,960 (10.89%)	198,644 (74.73%)	38,229 (14.38%)
Fusion-Models	75	101,335	13,211 (13.04%)	72,142 (71.19%)	15,982 (15.77%)

Table 5.16: Fusion sets number of classes, images and distribution of the images between the three source datasets.

Table 5.17 shows the results of the cross-tests performed with the different makers sets. As can be seen, for all subsets the best performing model is the one trained with the full Fusion-Makers dataset, followed by the model trained with the tested subset.

	Cross-Test top1/3 accuracy(%)			
Train	Fusion-Makers	CompCars-Makers	VMMR-db-Makers	Frontal-103-Makers
Fusion-Makers	**98.47 / 99.60**	**99.33 / 99.77**	**98.09 / 99.59**	**99.81 / 99.95**
CompCars-Makers	47.39 / 58.43	99.17 / 99.77	30.92 / 44.88	93.73 / 97.52
VMMR-db-Makers	94.96 / 97.75	79.70 / 88.92	98.04 / 99.47	90.49 / 95.54
Frontal-103-Makers	33.32 / 46.18	41.34 / 58.13	19.38 / 34.11	99.78 / 99.92

Table 5.17: Cross-Test performance comparison of Fusion-Makers dataset and its source makers datasets.

This proves that the joint use of the datasets enhances variety and mitigates bias impact, thus allowing the trained model to have better generalisation capabilities. Apart from the Fusion model, the one trained with VMMR-db is the only one capable of achieving competent results on the other subsets. It is important to note that VMMR-db accounts for almost 75% of the Fusion dataset, which could justify the good results when testing with Fusion, but not the performance achieved when testing with CompCars or Frontal-103. Both CompCars and Frontal-103 trained models obtain poor performances when tested with Fusion or VMMR-db. It is curious that CompCars model achieve better results than the VMMR-db one when tested with Frontal-103. This shows that CompCars and Frontal-103 are very similar, hence the better performance. Frontal-103, on the other hand, achieves really poor results, which is understandable, given the strong bias it has as it is exclusively made up of front view images.

Table 5.18 shows a performance comparison of the Fusion-Makers models on the PREVENTION Makers test set. As expected, the best performing model is the one trained with the full Fusion dataset, achieving an overall accuracy of 86.19% with a mean precision and recall of 77.34% and 80.40% respectively.

Train	test top1 accuracy(%) front / rear / all	test all prec/recall(%)
Fusion-Makers	**89.81 / 81.82 / 86.19**	**77.34 / 80.40**
CompCars-Makers	75.26 / 65.07 / 70.64	58.85 / 60.11
VMMR-db-Makers	82.80 / 73.52 / 78.60	76.16 / 70.02
Frontal-103-Makers	78.44 / 27.43 / 55.31	54.88 / 46.28

Table 5.18: PREVENTION Makers test accuracy, precision and recall comparison of Fusion-Makers and its source datasets trained models. From the 27 classes there are 23 common with the PREVENTION Makers test set.

Figure 5.19 shows some examples of the top3 predicted classes of the Fusion-Makers trained model in the PREVENTION Makers test set. The first row shows correctly predicted front view images, the middle row correctly predicted rear view images and the bottom row misclassified images from both views. As can be seen, the model has practically total confidence in correct predictions, with a mean confidence of 97.78%. On the contrary, the confidence in wrong predictions is much lower, with a mean confidence of 68.05%. This is a powerful feature, as a confidence threshold can be set to discard the predictions below it.

Figure 5.19: Top3 predicted classes of the Fusion-Makers trained model for sample images from the PREVENTION Makers test set. The first row shows correctly classified front view images. The middle row shows correctly classified rear view images. The bottom row shows misclassified images from both views.

Table 5.19 shows the results of the cross-tests performed with the different models sets. Once again, the best performing model is the one trained with the full Fusion-Models dataset. For the rest of the models the same pattern as for makers repeats, with the Fusion model followed

by the one trained with the tested subset. It is curious that while the Fusion-Models model achieves practically identical performances when compared with the Fusion-Makers, the rest of the models obtain lower performances. This is probably due to the increased difficulty, which, moreover, highlights the effects of using the datasets jointly, allowing the Fusion trained model to remain unaffected to this increase in difficulty.

Table 5.20 shows a performance comparison of the Fusion-Models models on the PREVENTION Models test set. Once again, the best performing model is the one trained with the full Fusion dataset, and, as expected, the performances are lower than with makers. The Fusion trained model achieves an overall accuracy of 78.37% with a mean precision and recall of 79.27% and 77.45% respectively. The huge gap in precision but above all in recall is remarkable, with a difference for the next best performing model of 14.3% and 19.11% for precision and recall respectively. This gap also appears in accuracy, with a difference 13.32% with the next model, which supports the importance of having a quality dataset that enhances generalisation capabilities.

	Cross-Test top1/3 accuracy(%)			
Train	Fusion-Models	CompCars-Models	VMMR-db-Models	Frontal-103-Models
Fusion-Models	**98.51 / 99.58**	**99.32 / 99.75**	**98.11 / 99.47**	**99.62 / 99.64**
CompCars-Models	43.85 / 56.20	98.41 / 99.57	25.17 / 39.98	83.25 / 93.69
VMMR-db-Models	86.28 / 93.19	63.30 / 80.15	97.90 / 99.32	52.67 / 76.24
Frontal-103-Models	25.76 / 31.69	34.93 / 44.04	7.90 / 14.43	99.26 / 99.81

Table 5.19: Cross-Test performance comparison of Fusion-Models dataset and its source models datasets.

Train	**test top1 accuracy(%) front / rear / all**	**test all prec/recall(%)**
Fusion-Models	**80.63 / 75.67 / 78.37**	**79.27 / 77.45**
CompCars-Models	53.35 / 49.43 / 51.56	62.88 / 53.52
VMMR-db-Models	62.22 / 68.44 / 65.05	64.97 / 58.34
Frontal-103-Models	53.97 / 4.94 / 31.66	54.49 / 33.69

Table 5.20: PREVENTION Models test accuracy, precision and recall comparison of Fusion-Models and its source datasets trained models. From the 75 classes there are 34 common with the PREVENTION Models test set.

Figure 5.20 shows some examples of the top3 predicted classes of the Fusion-Models trained model in the PREVENTION Models test set. The first row shows correctly predicted front view images, the middle row correctly predicted rear view images and the bottom row misclassified images from both views. As with makers, the model has practically total confidence in correct predictions, with a mean confidence of 95.68%. And again, a much lower confidence in wrong predictions, with a mean confidence of 69.91%. As expected, the confidence in correct predictions is 2.1% lower than for makers (95.68% vs 97.78%) and the confidence in wrong predictions is 1.86% higher (69.91% vs 68.05%). This is perfectly normal as the model classification problem is more complex than the makers one. In any case, these differences are minimal, largely due to the use of the Fusion cross-dataset.

These experiments have allowed us to evaluate the importance of datasets, the complexity of the main existing ones and the generalisation capabilities of the models trained with them. From these datasets, the one with higher quality and complexity is VMMR-db, followed by CompCars with a significant step down in cross-performance and finally Frontal-103, which turns out to be a poor quality dataset, given that it is strongly biased. It has been demonstrated that the

Figure 5.20: Top3 predicted classes of the Fusion-Models trained model for sample images from the PREVENTION Models test set. The first row shows correctly classified front view images. The middle row shows correctly classified rear view images. The bottom row shows misclassified images from both views.

combined use of datasets improves generalisation capabilities achieving the best results in cross-tests both for makers and models and obtaining good results in the external test performed with the PREVENTION test set.

The results obtained with the existing datasets suggest that fine-grained classification problem is solved. However, cross-testing reveals the shortcomings of existing datasets. Both Comp-Cars and Frontal-103 are highly biased, followed by VMMR-db, which although performs better is not free of bias. It is only when cross and externally testing that it is found that the vehicle classification problem is not solved, and not only for a complex task such as fine-grained classification, but in a simpler one like maker classification. For these reasons, it is necessary to create a sufficiently large and varied dataset, with images of multiple origins, qualities and viewpoints, to be able to tackle the classification problem satisfactorily.

5.3 Conclusions

In this chapter results of the proposed experiments to address vehicle keypoint detection and fine-grained vehicle classification have been presented. The conclusions that can be extracted from these results are the following.

5.3.1 Vehicle Keypoint Detection

Several models and training techniques have been studied to tackle the vehicle keypoint detection problem. First of all, different data augmentation techniques, backbones and input sizes have been evaluated, showing the usefulness and necessity of using data augmentation, with an improvement of 11.43% for PCK and 13.76% for APK, and achieving the best results with the deeper backbone (ResNet152) and the higher input resolution (384x288). An exhaustive analysis of PASCAL3D+ has been conducted, revealing the differences in quality and complexity

of PASCAL and ImageNet subsets and that the joint use of them improves performance. The results show that SBVPE is a robust approach that achieves state of the art results with a PCK of 88.49% and an APK of 55.09% for PASCAL validation, an improvement of 6.69% and 9.69% respectively. Continuing with the experiments, the impact of instance size and occlusions have been assessed, showing that our approach has a good performance that is independent of the instance type, with competent results both for occluded and low resolution instances, with an increase of more than 21% PCK for occluded data and 9.55% for low resolution objects. This makes our model a perfect candidate for real world use. Finally, a keypoint distribution study of PASCAL3D+ was performed, showing the importance of conducting an in depth analysis when choosing keypoints. While PCK results were almost perfect for all keypoints, APK showed some problems with the wheels and the windshield/rear window. To address these issues a custom dataset has been developed, enhancing the labelling of PASCAL3D+ and extending the images. Although results are not directly comparable due to differences in complexity, SBVPE achieves 98.83% PCK with the custom dataset vs 97.12% with PASCAL3D+ and 81.92% APK vs 72.38%. These results support our labelling proposal, and once again show the importance of evaluating the suitability of the chosen keypoints.

The results obtained with all these experiments support our approach of adapting a human pose estimation method for vehicle keypoint detection. SBVPE achieves state of the art results in PASCAL and PASCAL3D+ while being a simple, no frills architecture. The importance of preprocessing data has been shown, along with the need of analysing keypoint suitability and having an adequate dataset.

5.3.2 Fine-grained Vehicle Classification

The strengths and shortcomings of CompCars, VMMR-db and Frontal-103 have been analysed along with multiple training methods and approaches. In the first place, different curriculum learning techniques have been evaluated. *Incremental learning* approach has shown a slight improvement in overall performance with a clear relation between per-class performance and the number of samples. *Progressive learning* achieved virtually identical performance regardless of class order (increasing complexity, decreasing complexity and random). These results make difficult to justify the use of these techniques as a way to improve learning, however, *progressive learning* has proven to be an useful tool for reducing training time and adding classes to already trained models, with the resulting savings in time and resources. After this, the different datasets have been evaluated training various InceptionV3 networks and showing that there is no need of using complex models or techniques to achieve good overall performances. All the trained models with the exception of VMMR-db 3,040 outperform the baselines methods proposed by the datasets authors. These results showed the poor construction of VMMR-db 3,040 and confirmed that even though ultra fine-grained classification is a more challenging task, with the correct data it can be tackled. Continuing with the experiments, the effect of different weighting approaches has been evaluated. Results showed that per-class performance is directly related with the number of samples and that weighed models achieved similar validation performance while improving test results, with the normalised weights being the best option. On the other hand, the claim that weights could help with poor performing classes has not been validated. However, there is evidence of an improvement in generalisation capabilities, with an improvement of 2.49% for makers and 1.8% for models. These results are very limited and are not conclusive, as PREVENTION test set has a low amount of samples and classes. Finally, the complexity of existing datasets and the generalisation capabilities of the models trained with them have been assessed. Results showed the importance of datasets, being VMMR-db the one with higher complexity and quality followed by CompCars, with a significant step down in cross-performance and finally Frontal-103. The combined use of datasets in the Fusion cross-dataset has proven to improve generalisation capabilities achieving the best results in cross-tests both for makers and models, obtaining good results in the external test performed with the PREVENTION test

set. The Fusion-Models trained model achieves 98.51% validation accuracy and an overall test accuracy of 78.37% with solid mean precision and recall. The fact that the mean confidence in correct predictions is 95.68% while in wrong predictions is of 69.91% makes this model an excellent option for real use along with a threshold to discard predictions below it.

The results obtained with the existing datasets suggest that fine-grained classification problem is practically solved and that there is only room for minor improvements. However, cross-testing has showed the shortcomings of the existing datasets, which are biased. Only with the cross-tests performed along with the external test is it possible to see that vehicle classification is not solved, presenting problems not only with a complex task like fine-grained classification, but with a simpler one like maker classification. Thus, it is necessary to create a competent dataset that allows to face the vehicle classification task in a satisfactory manner.

Chapter 6

Conclusions and Future Work

This chapter presents the global conclusions and discusses the main contributions of this thesis as well as the future lines of research.

6.1 Conclusions

The goal of this thesis was the development of predictive systems to detect vehicle keypoints and achieve accurate unbiased fine-grained vehicle classification.

- The proposed keypoint detection model, based on a state of the art human pose estimation model, has proven to be a simple and efficient option, outperforming previous methods in PASCAL3D+ and showing solid performance regardless of the size of the instances or whether they are truncated or occluded. Our PASCAL validated method achieves 88.49% vs 81.8% PCK and 55.09% vs 45.4% APK while the PASCAL3D+ validated one achieves 97.12% vs 93.4% PCK and 72.38% APK.

- The importance of properly choosing the keypoints and evaluating them has been tested. Showing the problems of some of the PASCAL3D+ keypoints and proposing a new labelling that obtains a slightly higher PCK than PASCAL3D+ (97.12% vs 98.83%) with a considerably better APK (72.38% vs 81.92%).

- As the state of the art methods exclusively focus on improving the overall performance by increasing the complexity of the models, we wanted to explore other techniques such as curriculum learning or weighted losses with the aim of enhance learning capabilities. Curriculum learning techniques (*incremental learning* and *progressive learning*) achieved very similar performance to not using them, making difficult to justify their use. However, *progressive learning* has proven to be an useful tool for reducing training time and adding classes to already trained models, thus reducing time and resources needed. While the standard ResNet50 needs 3h5m to reach 97.00% accuracy, with the *progressive* technique and random class order the model reach 97.03% accuracy in 2h44m. Slightly better performance with 11.35% less training time.

- No conclusive results have been reached regarding the effect of weights in poor performing classes. However, weighted models achieve similar validation performance while improving PREVENTION test results. In the case of makers, *Standard 43* weights practically matches validation accuracy (97.34% vs 97.31%) outperforming the weightless model in test (76.17% vs 78.66%). For models, both normalised weighting approaches match validation accuracy, with 55.90% overall test accuracy vs 54.23% of the weightless model. However, *Focal Loss*, the worst performing model in terms of validation accuracy, achieves the best overall test accuracy with 56.03%.

- A cross-dataset (Fusion) has been proposed to assess the complexity and generalisation capabilities of existing datasets. A series of cross-tests have been performed with the different datasets, showing that Fusion trained models achieve the best results in all cross-tests with good results in the PREVENTION test set. The Fusion-Models trained model achieves a 98.51% validation accuracy and an overall test accuracy of 78.37%. The mean confidence in correct predictions of 95.68% and in wrong predictions of 69.91% makes this models a perfect candidate for real world applications.

6.2 Contributions

The main contributions of this thesis are as follows:

- State of the art has shown that the way to go for vehicle keypoint detection is to adapt a human pose estimation model. Following on the ideas proposed by [37], we wanted to answer the question: *How good could a simple model be?*. Therefore, a state of the art human pose estimation model has been proposed, adapted and validated for vehicle keypoint detection achieving state of the art results in PASCAL3D+ dataset.

- During the study of the different datasets and methods for vehicle keypoint detection, a lack of analysis of the datasets and the suitability of the chosen keypoints, its number, usefulness or impact in the models was detected. Because of this, an exhaustive analysis of PASCAL3D+ and its subsets has been carried out, finding the differences in complexity between PASCAL and ImageNet subsets and the issues with some of the chosen keypoints and the labelling process.

- Having identified some of the shortcomings of PASCAL3D+ structure and keypoints, the Custom Keypoints dataset has been developed and validated to address these issues. Part of PASCAL3D+ has been re-labelled and extended with images from CompCars and the PREVENTION dataset. The Custom Keypoints datasets has a total of 4,042 images with 4,080 instances and 19 different keypoints. An extensive analysis of the proposed keypoints has been carried out.

- Fine-grained vehicle classification state of the art methods exclusively focus on improving the overall performance by increasing the complexity of the models completely neglecting other key aspects as per-class performance, generalisation capabilities or data imbalance. Curriculum learning techniques and weighted losses have been evaluated as a way to enhance learning capabilities of the models. Curriculum learning techniques have shown similar results to not use them, but *progressive learning* reduces the training time needed and allows new classes to be added to already trained models while achieving equivalent results.

- No conclusive results have been obtained regarding the usefulness of weighted losses for the purpose of enhancing performance of underrepresented poor performing classes. However, weighted models achieve similar results to weightless ones but with an enhanced generalisation capability obtaining better results when externally tested with the PREVENTION dataset.

- As the main fine-grained vehicle classification datasets have its validation results saturated (accuracy above 95%), one may think that fine-grained vehicle classification is solved. Our theory is that these datasets are biased and that the fine-grained classification task is far from being solve. To prove it we developed the PREVENTION test set, a real world driving scenario dataset to externally evaluate the generalisation capabilities and performance of different datasets, models and techniques. The PREVENTION Makers test set has a

total of 33 classes with 2,685 images, 1,452 are front view and 1,233 rear view. The PREVENTION Models test set has a total of 87 classes with 1,113 images, 618 are front view and 515 rear view.

- To further show the shortcomings of existing dataset we developed the Fusion cross-dataset to evaluate complexity of existing datasets and generalisation capabilities of the trained models. The dataset is composed of the common classes of CompCars, VMMR-db and Frontal-103. Fusion-Makers set features 27 different manufacturers with a total of 265,833 images and Fusion-Models set features 75 different vehicle models with a total of 101,335 images.

6.3 Future work

Following the results obtained in the various experiments carried out and the conclusions reached, multiple lines of research are open to further improve the performance of both systems, either separately or jointly.

- The Custom Keypoints dataset has proven to be a more than adequate option, achieving incredible results despite the reduced number of samples. It is interesting to continue improving the dataset by adding new samples and keypoints, testing their viability and impact on performance.
- SBVPE approach has shown to be an excellent option achieving state of the art results. However, as a top-down approach it has some limitations. Other architectures need to be explored, specially those with a bottom-up approach.
- Dataset bias has shown to be an important problem when training unbiased models with good generalisation capabilities. It is necessary to construct a dataset with images of diverse nature, resolutions, qualities and viewpoints, with a wide variety of makes and models from different geographical regions and adequate class hierarchy. Only in this way will it be possible to obtain unbiased models capable of performing fine-grained vehicle classification in multiple realistic environments.
- The PREVENTION test set has proven to be an useful tool to externally evaluate performance. It is necessary to upgrade it with more samples, makers and models to achieve more reliable results.
- A hybrid model that joins keypoint detection and fine-grained vehicle classification in a single pipeline could improve both tasks or, at least, give a more compact and efficient solution.

Appendices

Appendix A

Further works derived from this thesis

In this appendix we present some of the open lines of research that were explored at some point and that finally did not meet the results or the applicability to be included in the main document. Still, we feel that they are interesting enough to deserve an appendix that could help future researchers to pick up from this point and continue a new line of research.

A.1 License Plate Localisation with Keypoints

During the development of this thesis, a first approach to the use of keypoints was the localisation of vehicle license plates with them. Instead modelling the appearance of the full license plate, we focus on modelling the corners as keypoints. Inspired by advances in human pose estimation, the DSNT layer [57] is used to build and train a CNN-based model and perform coordinate regression of the four license plate corners.

The proposed model use an ImageNet pretrained ResNet50 backbone followed by a DSNT layer. This DSNT layer allows a fully differentiable and spatially generalisable coordinate regression architecture with no trainable parameters.

The DSNT layer is composed of two $m \times n$ matrices X and Y for each keypoint (four in our case). This X and Y matrices are defined as:

$$X_{i,j} = (2j - (n+1))/n \tag{A.1}$$

$$Y_{i,j} = (2i - (m+1))/m \tag{A.2}$$

and the operations performed by this layer are as follows:

$$DSNT(\hat{Z}) = \left[\left\langle \hat{Z}, X \right\rangle_F, \left\langle \hat{Z}, Y \right\rangle_F \right] \tag{A.3}$$

being F the Frobenius inner product of real values (scalar dot product of vectorised matrices) and $\hat{Z}$ the normalised heatmaps obtained as output from the ResNet50 backbone using L^1 norm.

As the output produced by the DSNT layer are numerical coordinates, 2D Euclidean distance between the predicted locations and the ground truth ones can be used as loss function. Additionally, a specific regularisation term is introduced in the loss function using the Jensen-Shannon divergence. This is intended to force the heatmaps resemble a 2D Gaussian by minimising its divergence.

In order to train the model, a specific dataset was created using images from the entrance of a public car park. This custom dataset contains a total of 557 images manually labelled with the four license plate corners. The input images size is 640×480. Figure A.1 shows some examples of the car park images. Additionally to the car park images used for training, an independent qualitative test has been carried out using images of other environments with different viewpoints. Figure A.2 shows some examples of the test images.

Figure A.1: Examples of the car park entrance images used for train and validation.

Figure A.2: Examples of the test images.

For the given input size (640×480), the standard ResNet50 backbone outputs a heatmap of 20×15 pixels. We empirically found that higher heatmap resolutions increase localisation accuracy. In order to obtain higher output heatmap resolutions, the stride and dilation factors of the two last layers of the backbone can be modified. By varying the amount of dilated convolutions we obtain different heatmap sizes, increasing the size by a $\times 2$ factor with each one, as can be seen in Table A.1.

# dilated convs.	Heatmap size
0	20×15
1	40×30
2	80×60
3	160×120

Table A.1: Effect of dilated convolutions on the output heatmap resolution for an input size of 640×480.

The data was split in a 70/30 way, with 389 images for training and 168 images for validation. The initial learning rate was set to 10^{-4} and the model trained for 50 epochs. At epochs 20 and 40 the learning rate was reduced by a factor of 10. RMSProp was used as the optimiser. Table A.2 shows the results (average distance error) for the four different models (0, 1, 2 or 3 dilated convolutions). As can be seen, the best performance is obtained with the 3 dilated convolutions model, achieving an average distance error of 0.3244 pixels. It should be noted that the use of dilated convolutions results in higher resource and memory usage, requiring ~9 times more computational power than the model without dilated convolutions. Figure A.3 shows a visual example of the effect of the dilated convolutions on the output heatmaps.

# dilated convs.	Average validation distance error (pixels)
0	1.907
1	0.7694
2	0.5047
3	**0.3244**

Table A.2: Average distance error obtained with 0, 1, 2 and 3 dilated convolutions in validation images.

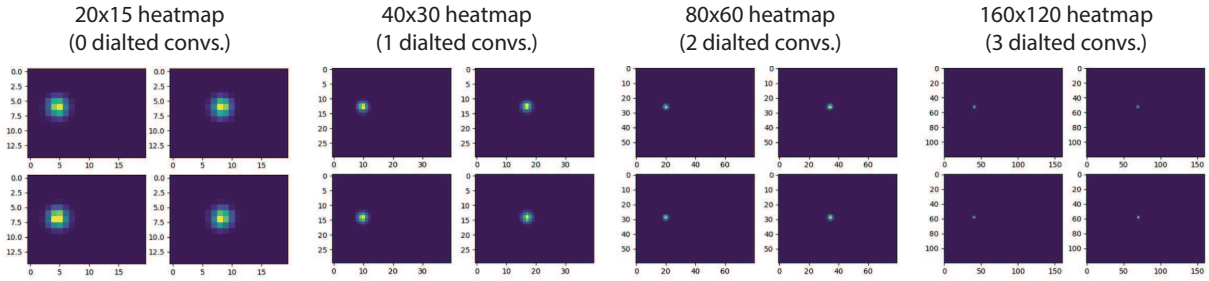

Figure A.3: Example of the different heatmap resolutions.

Results on validation and test data can be seen in Figure A.4. The top row shows some examples of the predicted coordinates for validation images (green ground truth and red predictions). As can be seen, the system works accurately, practically matching the ground truth. Given that the size of the dataset used for training is small, it could be thought that the system is overfitting. In order to rule out this event, a qualitative test has been carried out with images obtained with different cameras and in different scenarios. The bottom row shows some examples of these test images. As can be seen, the model has learned to generalise appropriately, accurately detecting license plate corners in images from different sources.

Figure A.4: Examples of license plate corners detection on the validation set (top row) and on the test set (bottom row). Green circles are the ground truth, red circles are the predictions. License plate area has been zoomed to ease visualisation.

This approach proved to be a simple and effective way to detect the corners of license plates. Further experiments were conducted using deeper backbones (such as ResNet101 or 152). Unfortunately, the computational requirements to cope with a larger number of keypoints and deeper architectures became a problem. This, together with other state of the art developments, led to the discarding of this method.

A.2 Keypoint enhanced Fine-grained Vehicle Classification

The joint use of vehicle keypoints and fine-grained vehicle classification could improve performance of both tasks. Exploring this idea, and following the approach adopted by Wang et al. [1], an attempt was made to create and train a model that, through the use of keypoints, would obtain extended information to improve fine-grained vehicle classification.

The approach proposed by Wang et al. consists in making use of vehicle keypoints to build four orientation masks (front, rear and sides) and use them as an attention-like mechanism to extract features from vehicle images and re-identify them in different scenes.

Following this idea, an adaptation of the proposed architecture was designed and implemented to perform fine-grained vehicle classification in CompCars. To do so, 19 different vehicle keypoints where extracted using our keypoint detection architecture derived from [37] and then, grouped to build the masks. These keypoints are: *front and rear wheels, the four windshield corners, the four rear window corners, left and right fog lights, left and right rear mirrors, the four license plate corners, and the logo*, and the groups to build the masks are:

- Front: *the four windshield corners, left and right fog lights, left and right rear mirrors, the four license plate corners, and the logo.*
- Rear: *the four rear window corners, left and right rear mirrors, the four license plate corners, and the logo.*
- Right Side: *front and rear wheels, right windshield corners, right rear window corners, right fog light and right rear mirror.*
- Left Side: *front and rear wheels, left windshield corners, left rear window corners, left fog light and left rear mirror.*

As some of the keypoints lack side information (wheels, license plate and logo) the orientation of the vehicle is obtained from context relations between the detected keypoints and, once the orientation is known, the masks are constructed according to the sides of the vehicle that are visible. Figure A.5 shows some examples of the computed masks.

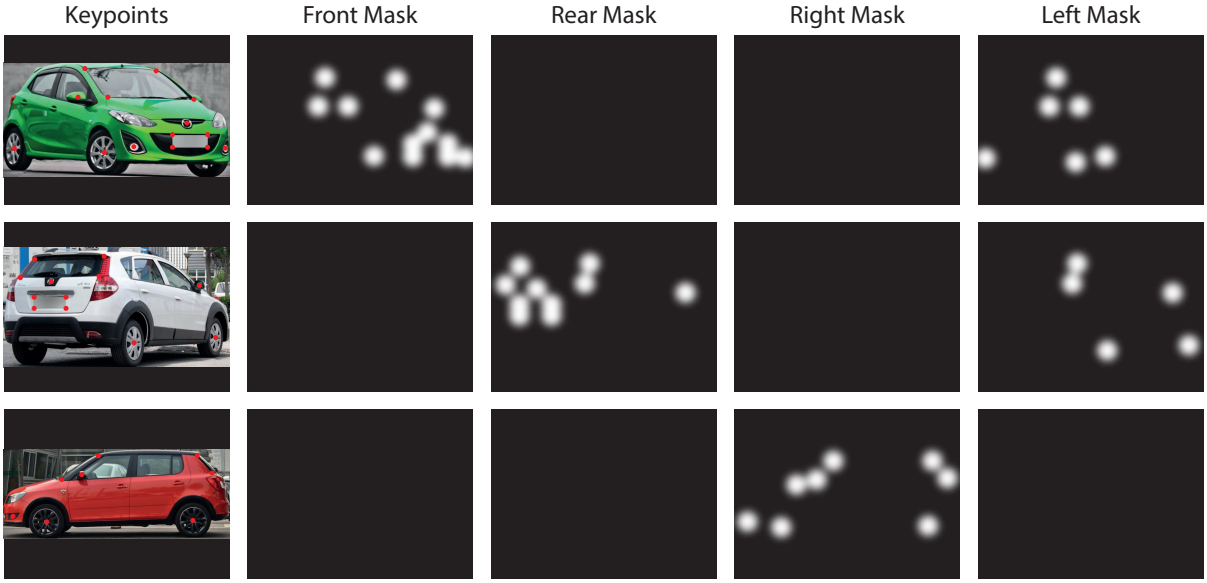

Figure A.5: Examples of the four masks generated from the keypoints for three different images.

The proposed architecture has two main parts or modules. The first module is in charge of feature extraction using the orientation masks obtained from the keypoints. In a first stage, a global feature vector is computed and four local feature vectors derived using the masks. In a second stage these vectors are refined. Once these vectors are obtained the second module performs an orientation-invariant feature aggregation. Weights for the four local feature vectors are computed and then the weighted local vectors are concatenated with the global one and a fully connected layer is used to predict the make and model of the vehicle. Figures A.6 and A.7 show an illustration of the feature extraction module and orientation-invariant feature aggregation module respectively.

In order to train the full architecture a modular and stepped approach was taken. First, the global feature vector stages 1 and 2 are trained. Once this branch has been trained, the learnt

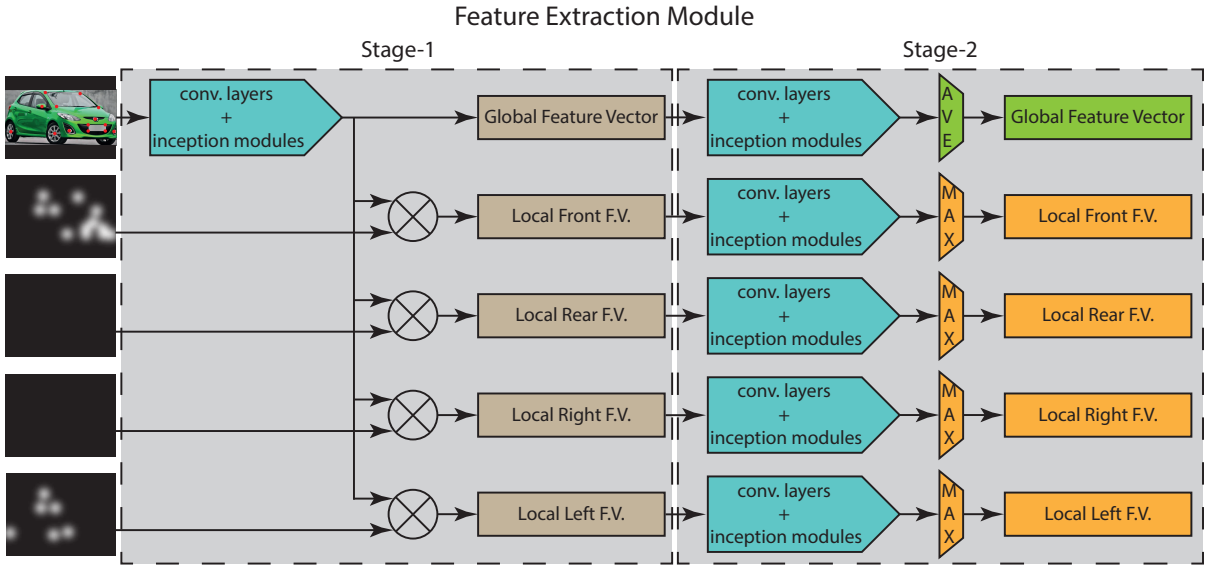

Figure A.6: Illustration of the feature extraction module. The first stage extracts a global feature vector and computes four local feature vectors with the masks. The second stage refines these vectors.

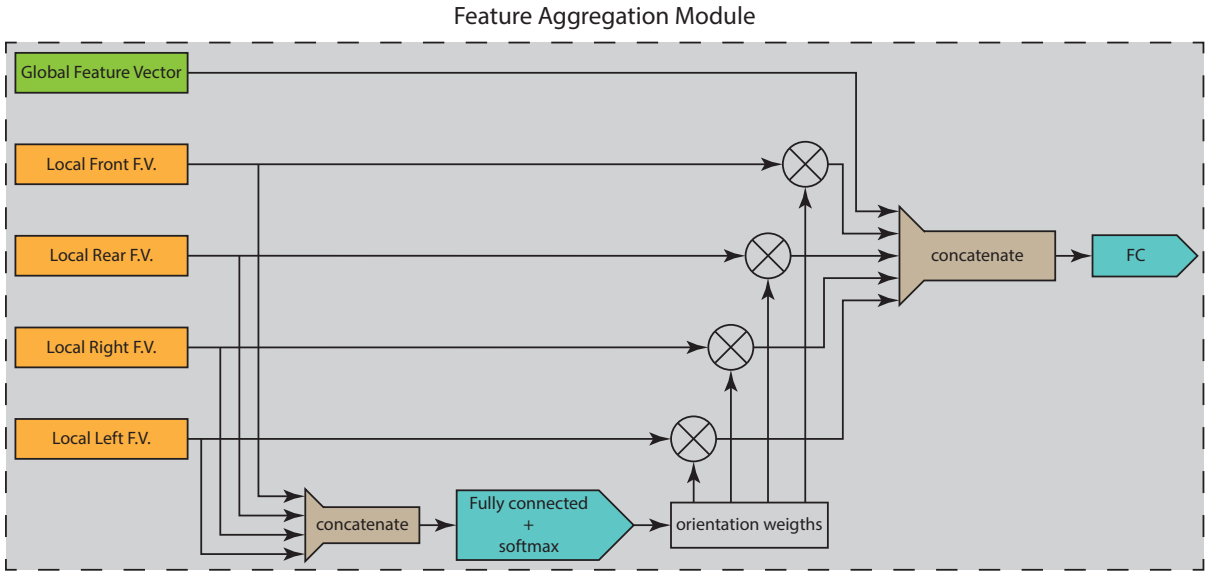

Figure A.7: Illustration of the orientation-invariant feature aggregation module. First, the local feature vectors are concatenated and four weights computed to ponder them. After this, the weighted local feature vectors and the global one are concatenated and fed to a fully connected layer to predict the make and model.

parameters are copied to the local branches and each of them is trained separately. With all the branches trained, both stages are frozen and the feature aggregation module is trained on its own. Finally, all the system is trained together. This training approach is very similar to the one taken by Wang et al., unfortunately, we were not able to get the system to converge and achieve satisfactory results.

Appendix B

Publications Derived from this PhD Dissertation

B.1 Journal Publications

2021 ***Are We Ready for Accurate and Unbiased Fine-Grained Vehicle Classification in Realistic Environments?**, *Corrales, Héctor and Hernández, Noelia and Parra, Ignacio and Nebot, Eduardo and Fernández-Llorca, David*, IEEE Access (ISSN: 2169-3536), Vol. 9, pages 116338-116355.

2021 **WiFiNet: WiFi-based indoor localisation using CNNs**, *Hernández, Noelia and Parra, Ignacio and Corrales, Héctor and Izquierdo, Rubén and Ballardini, Augusto Luis and Salinas, Carlota and García, Iván*, Expert Systems with Applications (ISSN: 0957-4174), Vol. 177, pages 114906-114915.

2020 ***Simple Baseline for Vehicle Pose Estimation: Experimental Validation**, *Corrales, Héctor and Hernández, Antonio and Izquierdo, Rubén and Hernández, Noelia and Parra, Ignacio and Fernández-Llorca, David*, IEEE Access (ISSN: 2169-3536), Vol. 8, pages 132539-132550.

B.2 Conference Publications

2020 **3D-DEEP: 3-Dimensional Deep-learning based on elevation patterns for road scene interpretation**, *Hernández, Álvaro and Woo, Seongyoun and Corrales, Héctor and Parra, Ignacio and Kim, Euntai and Fernández-Llorca, David and Sotelo, Miguel Ángel*, 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas (United States).

2020 ***CNNs for Fine-Grained Car Model Classification**, *Corrales, Héctor and Fernández-Llorca, David and Parra, Ignacio and Vigre, Susana and Quintanar, Álvaro and Lorenzo, Javier and Hernández, Noelia*, Lecture Notes in Computer Science (ISSN: 0302-9743), Vol. 12014, pages 104-112.

2020 ***License Plate Corners Localization Using CNN-Based Regression**, *Fernández-Llorca, David and Corrales, Héctor and Parra, Ignacio and Rentero, Mónica and Izquierdo, Rubén and Hernández, Álvaro and García, Iván*, Lecture Notes in Computer Science (ISSN: 0302-9743), Vol. 12014, pages 113-120.

2019 **WiFi-based urban localisation using CNNs**, *Hernández, Noelia and Corrales, Héctor and Parra, Ignacio and Rentero, Mónica and Fernández-Llorca, David and Sotelo, Miguel Ángel*, 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland (New Zealand).

2019 **Performance analysis of Vehicle-to-Vehicle communications for critical tasks in autonomous driving**, *Parra, Ignacio and Corrales, Héctor and Hernández, Noelia and Vigre, Susana and Fernández-Llorca, David and Sotelo, Miguel Ángel*, 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland (New Zealand).

2019 ***Deep Convolutional Neural Networks for Fine-Grained Car Model Classification**, *Corrales, Héctor and Fernández-Llorca, David and Parra, Ignacio and Sotelo, Miguel Ángel*, 17th International Conference on Computer Aided Systems Theory (EUROCAST 2019), Las Palmas de Gran Canaria (Spain).

2019 ***License Plate Localization using CNN-Based Numerical CoordinateRegression**, *Fernández-Llorca, David and Corrales, Héctor and Parra, Ignacio, and Sotelo, Miguel Ángel*, 17th International Conference on Computer Aided Systems Theory (EUROCAST 2019), Las Palmas de Gran Canaria (Spain).

* Publications directly related to this thesis.

Bibliography

[1] Z. Wang, L. Tang, X. Liu, Z. Yao, S. Yi, J. Shao, J. Yan, S. Wang, H. Li, and X. Wang, "Orientation Invariant Feature Embedding and Spatial Temporal Regularization for Vehicle Re-Identification," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 379–387.

[2] X. Song, P. Wang, D. Zhou, R. Zhu, C. Guan, Y. Dai, H. Su, H. Li, and R. Yang, "ApolloCar3D: A Large 3D Car Instance Understanding Benchmark for Autonomous Driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 5452–5462.

[3] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 483–499.

[4] W. Ding, S. Li, G. Zhang, X. Lei, and H. Qian, "Vehicle Pose and Shape Estimation through Multiple Monocular Vision," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 709–715.

[5] J. Sochor, A. Herout, and J. Havel, "BoxCars: 3D Boxes as CNN Input for Improved Fine-Grained Vehicle Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 3006–3015.

[6] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark Analysis of Representative Deep Neural Network Architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.

[7] D. Zhou and Q. He, "PoSeg: Pose-Aware Refinement Network for Human Instance Segmentation," *IEEE Access*, vol. 8, pp. 15 007–15 016, 2020.

[8] S. Zhang, C. Wang, Z. He, Q. Li, X. Lin, X. Li, J. Zhang, C. Yang, and J. Li, "Vehicle global 6-dof pose estimation under traffic surveillance camera," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 159, pp. 114–128, 2020.

[9] D. F. Llorca, C. Salinas, M. Jimenez, I. Parra, A. Morcillo, R. Izquierdo, J. Lorenzo, and M. Sotelo, "Two-camera based accurate vehicle speed measurement using average speed at a fixed point," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 2533–2538.

[10] D. F. Llorca, H. Corrales, I. Parra, M. Rentero, R. Izquierdo, Á. Hernández-Saz, and I. García-Daza, "License Plate Corners Localization Using CNN-Based Regression," *Computer Aided Systems Theory – EUROCAST 2019. Lecture Notes in Computer Science*, vol. 12014, pp. 113–120, 2020.

[11] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond PASCAL: A Benchmark for 3D Object Detection in the Wild," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2014, pp. 75–82.

[12] J. K. Murthy, G. S. Krishna, F. Chhaya, and K. M. Krishna, "Reconstructing Vehicles from a Single Image: Shape Priors for Road Scene Understanding," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 724–731.

[13] European Statistical Office, "Crime statistics," https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Crime_statistics&stable=0&redirect=no#car_thefts_in_the_EU-27_2018, 2020, accessed: 23/06/2021.

[14] Federal Bureau of Investigation, "Federal bureau of investigation - crime data explorer," https://crime-data-explorer.app.cloud.gov/pages/explorer/crime/property-crime, 2021, accessed: 23/06/2021.

[15] European Parking Association and IREA-UB, "Scope of Parking in Europe," 2013. [Online]. Available: https://www.europeanparking.eu/media/1180/epa_data_collection_rev.pdf

[16] C. Li, M. Zeeshan Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker, "Deep Supervision With Shape Concepts for Occlusion-Aware 3D Object Parsing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 5465–5474.

[17] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese, "ObjectNet3D: A Large Scale Database for 3D Object Recognition," in *Proceedings of the European Conference Computer Vision (ECCV)*. Springer, 2016, pp. 160–176.

[18] N. D. Reddy, M. Vo, and S. G. Narasimhan, "CarFusion: Combining Point Tracking and Part Detection for Dynamic 3D Reconstruction of Vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 1906–1915.

[19] H. Liu, Y. Tian, Y. Yang, L. Pang, and T. Huang, "Deep Relative Distance Learning: Tell the Difference Between Similar Vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 2167–2175.

[20] X. Liu, W. Liu, H. Ma, and H. Fu, "Large-Scale Vehicle re-Identification in Urban Surveillance Videos," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2016, pp. 1–6.

[21] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D Object Representations for Fine-Grained Categorization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*. IEEE, 2013, pp. 554–561.

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

[23] F. Tafazzoli, H. Frigui, and K. Nishiyama, "A Large and Diverse Dataset for Improved Vehicle Make and Model Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. IEEE, 2017, pp. 1–8.

[24] L. Lu, P. Wang, and H. Huang, "A Large-Scale Frontal Vehicle Image Dataset for Fine-Grained Vehicle Categorization," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012, accessed: 15/06/2021.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 248–255.

[27] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese, "Supplementary Material for "ObjectNet3D: A Large Scale Database for 3D Object Recognition"," Tech. Rep., 2016. [Online]. Available: https://cvgl.stanford.edu/papers/xiang_eccv16_tr.pdf

[28] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv preprint arXiv:1512.03012*, 2015.

[29] X. Liu, W. Liu, T. Mei, and H. Ma, "A Deep Learning-Based Approach to Progressive Vehicle Re-identification for Urban Surveillance," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 869–884.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 3354–3361.

[31] M. Z. Zia, M. Stark, and K. Schindler, "Towards Scene Understanding with Detailed 3D Object Representations," *International Journal of Computer Vision*, vol. 112, no. 2, pp. 188–203, 2015.

[32] S. Tulsiani and J. Malik, "Viewpoints and keypoints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 1510–1519.

[33] C. Li, M. Z. Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker, "Deep Supervision with Intermediate Concepts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1828–1843, 2018.

[34] N. D. Reddy, M. Vo, and S. G. Narasimhan, "Occlusion-Net: 2D/3D Occluded Keypoint Localization Using Graph Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 7326–7335.

[35] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Cascaded Pyramid Network for Multi-Person Pose Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 7103–7112.

[36] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 2961–2969.

[37] B. Xiao, H. Wu, and Y. Wei, "Simple Baselines for Human Pose Estimation and Tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2018, pp. 466–481.

[38] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[39] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 2117–2125.

[40] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[41] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[42] A. Newell, Z. Huang, and J. Deng, "Associative Embedding: End-to-End Learning for Joint Detection and Grouping," in *Proceedings of the Conference on Advances in Neural Information Processing Systems*, 2017, pp. 2277–2287.

[43] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele, "DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 4929–4937.

[44] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, "DeeperCut: A Deeper, Stronger, and Faster Multi-person Pose Estimation Model," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 34–50.

[45] S. Kreiss, L. Bertoni, and A. Alahi, "PifPaf: Composite Fields for Human Pose Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 11 977–11 986.

[46] J. L. Long, N. Zhang, and T. Darrell, "Do Convnets Learn Correspondence?" in *Proceedings of the Conference on Advances in Neural Information Processing Systems*, 2014, pp. 1601–1609.

[47] Z. Wang, G. Liu, and G. Tian, "A Parameter Efficient Human Pose Estimation Method Based on Densely Connected Convolutional Module," *IEEE Access*, vol. 6, pp. 58 056–58 063, 2018.

[48] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, "2D Human Pose Estimation: New Benchmark and State of the Art Analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014, pp. 3686–3693.

[49] I. Radwan, N. Moustafa, B. Keating, K.-K. R. Choo, and R. Goecke, "Hierarchical Adversarial Network for Human Pose Estimation," *IEEE Access*, vol. 7, pp. 103 619–103 628, 2019.

[50] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Proceedings of the Conference on Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[51] X. Wang, Z. Cao, R. Wang, Z. Liu, and X. Zhu, "Improving Human Pose Estimation With Self-Attention Generative Adversarial Networks," *IEEE Access*, vol. 7, pp. 119 668–119 680, 2019.

[52] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-DoF Object Pose from Semantic Keypoints," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2011–2018.

[53] J. K. Murthy, S. Sharma, and K. M. Krishna, "Shape Priors for Real-Time Monocular Object Localization in Dynamic Environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1768–1774.

[54] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional Pose Machines," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 4724–4732.

[55] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 7291–7299.

[56] S. Kreiss, L. Bertoni, and A. Alahi, "OpenPifPaf: Composite Fields for Semantic Keypoint Detection and Spatio-Temporal Association," *arXiv preprint arXiv:2103.02440*, 2021.

[57] A. Nibali, Z. He, S. Morgan, and L. Prendergast, "Numerical Coordinate Regression with Convolutional Neural Networks," *arXiv preprint arXiv:1801.07372*, 2018.

[58] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[59] H.-Z. Gu and S.-Y. Lee, "Car model recognition by utilizing symmetric property to overcome severe pose variation," *Machine Vision and Applications*, vol. 24, no. 2, pp. 255–274, 2013.

[60] D. F. Llorca, D. Colás, I. G. Daza, I. Parra, and M. Sotelo, "Vehicle model recognition using geometry and appearance of car emblems from rear view images," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 3094–3099.

[61] D. Santos and P. L. Correia, "Car recognition based on back lights and rear view features," in *Proceedings of the Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*. IEEE, 2009, pp. 137–140.

[62] D. F. Llorca, R. Arroyo, and M. A. Sotelo, "Vehicle logo recognition in traffic images using HOG features and SVM," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2013, pp. 2229–2234.

[63] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN Models for Fine-Grained Visual Recognition," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015, pp. 1449–1457.

[64] J. Krause, H. Jin, J. Yang, and L. Fei-Fei, "Fine-Grained Recognition Without Part Annotations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 5546–5555.

[65] J. Fang, Y. Zhou, Y. Yu, and S. Du, "Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1782–1792, 2017.

[66] J. Fu, H. Zheng, and T. Mei, "Look Closer to See Better: Recurrent Attention Convolutional Neural Network for Fine-Grained Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4438–4446.

[67] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified Visual Attention Networks for Fine-Grained Object Classification," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1245–1256, 2017.

[68] Y. Tian, W. Zhang, Q. Zhang, G. Lu, and X. Wu, "Selective Multi-Convolutional Region Feature Extraction based Iterative Discrimination CNN for Fine-Grained Vehicle Model Recognition," in *Proceedings of the International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 3279–3284.

[69] S. Elkerdawy, N. Ray, and H. Zhang, "Fine-grained vehicle classification with unsupervised parts co-occurrence learning," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. Springer, 2018, pp. 664–670.

[70] R. Du, D. Chang, A. K. Bhunia, J. Xie, Z. Ma, Y.-Z. Song, and J. Guo, "Fine-Grained Visual Classification via Progressive Multi-granularity Training of Jigsaw Patches," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 153–168.

[71] Y. Ding, Z. Ma, S. Wen, J. Xie, D. Chang, Z. Si, M. Wu, and H. Ling, "AP-CNN: Weakly Supervised Attention Pyramid Convolutional Neural Network for Fine-Grained Visual Classification," *IEEE Transactions on Image Processing*, vol. 30, pp. 2826–2836, 2021.

[72] K. Ramnath, S. N. Sinha, R. Szeliski, and E. Hsiao, "Car Make and Model Recognition using 3D Curve Alignment," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2014, pp. 285–292.

[73] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis, "Jointly Optimizing 3D Model Fitting and Fine-Grained Classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 466–480.

[74] A. Anderson, K. Shaffer, A. Yankov, C. D. Corley, and N. O. Hodas, "Beyond Fine Tuning: A Modular Approach to Learning on Small Data," *arXiv preprint arXiv:1611.01714*, 2016.

[75] Q. Hu, H. Wang, T. Li, and C. Shen, "Deep CNNs With Spatially Weighted Pooling for Fine-Grained Car Recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 11, pp. 3147–3156, 2017.

[76] X. Li, L. Yu, D. Chang, Z. Ma, and J. Cao, "Dual Cross-Entropy Loss for Small-Sample Fine-Grained Vehicle Classification," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4204–4212, 2019.

[77] H. Corrales, D. F. Llorca, I. Parra, S. Vigre, A. Quintanar, J. Lorenzo, and N. Hernández, "CNNs for Fine-Grained Car Model Classification," *Computer Aided Systems Theory – EUROCAST 2019. Lecture Notes in Computer Science*, vol. 12014, pp. 104–112, 2020.

[78] 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.

[79] R. Izquierdo, A. Quintanar, I. Parra, D. Fernández-Llorca, and M. A. Sotelo, "The PREVENTION dataset: a novel benchmark for PREdiction of VEhicles iNTentIONs," in *Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3114–3121.

[80] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 770–778.

[81] Y. Yang and D. Ramanan, "Articulated Human Detection with Flexible Mixtures of Parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2878–2890, 2012.

[82] J. L. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993.

[83] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the International Conference on Machine Learning*, 2009, pp. 41–48.

[84] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 2980–2988.