GDAŃSK UNIVERSITY OF TECHNOLOGY

Faculty of Electronics, Telecommunications, and Informatics

The author of the PhD dissertation: **Sebastian Cygert**
Scientific discipline: **Information and communication technology**

DOCTORAL DISSERTATION

Title of PhD dissertation: **Robust and Efficient Machine Learning Algorithms for Visual Recognition**

Title of PhD dissertation (in Polish): **Skuteczne i wydajne obliczeniowo algorytmy uczenia maszynowego dla potrzeb rozpoznawania obrazu**

Supervisor

signature

prof. dr hab. inż. Andrzej Czyżewski

Second supervisor

Auxiliary supervisor

Cosupervisor

Gdańsk, 2021

Summary

In visual recognition, the task is to identify and localize all objects of interest in the input image. With the ubiquitous presence of visual data in modern days, the role of object recognition algorithms is becoming more significant than ever and ranges from autonomous driving to computer-aided diagnosis in medicine. Current models for visual recognition are dominated by models based on Convolutional Neural Networks (CNNs), which achieve impressive performance on many benchmarks. However, when deployed to the real world, the performance of these CNN models can drop drastically, lacking the desired robustness property. This is because of the so-called distributional shift, where the test-time data differ from data observed during training, and it poses one of the most important challenges in modern machine learning. At the same time, modern CNN-based models may be too expensive or too slow for general deployment.

As such, the goal of this thesis is to develop robust and efficient models for visual object recognition. In the experimental section, the focus is on autonomous driving because of the datasets' availability, and also because the aforementioned problems are essential for autonomous driving.

Evaluating robustness is challenging since collecting out-of-distribution data (for example, rare events, atypical weather conditions in the case of autonomous driving) is often not feasible. As such, this thesis starts with analyzing methods for evaluating models' robustness. This includes cross-dataset evaluation, adding synthetic distortions during testing, and for day-to-night transition (when models trained on daytime images are evaluated on night-time images). Throughout the experiments, it was shown that using different measures, such as model accuracy (that is, accuracy on "clean" datasets, but also using the proposed methods) and uncertainty estimation, is crucial for fully understanding the impacts of some methods (e.g., data augmentation) on model performance.

Equipped with the tools for evaluating model robustness, the next goal was to understand the impact of model compression methods is on model accuracy. Model compression works by removing some neurons or filters during training, which improves the inference time, without hurting overall accuracy. This is possible since current CNN-based models are over-parametrized. Such methods are very popular in visual recognition, however, the side effects of such techniques are unknown. Indeed, it was found that compression methods disproportionately increase the model

MOST WIEDZY Downloaded from mostwiedzy.pl

vulnerability to different corruption types. Some of the corruptions are heavily impacted by the compression methods (i.e., Gaussian noise), while others (blur effect) are only slightly affected. Also, it was found that compression techniques have a disproportionate impact on different object classes, even at moderate compression rates. It was hypothesized that one of the reasons for this is data imbalance, a compressed model (with smaller capacity) will firstly remove neurons responsible for recognition of less-common data. The experimental section found that using data balancing methods helped to improve the accuracy of some classes. Careful evaluation and analysis of the results are an important part of this thesis - fully understanding the impact of some interventions (using model compression in this case) is essential for models deployed to the real world.

Further, the robustness of model ensembles is investigated, which are known because of their remarkably high accuracy, which comes, however, at a significantly increased computational cost. Semantic segmentation models are studied in the domain adaptation setting under a varying level of distributional shift (when doing cross-dataset adaptation and when adapting from data from simulation). Indeed, it was found that using a model ensemble with diverse models (by means of different backbones and data augmentation schemes) resulted in very high accuracy, even when the distributional shift was large. Further, it was observed that uncertainty calibration improved in the distributional shift scenario, which was an important finding. Given those observations, the next goal was to transfer the ensemble's knowledge into a single model. For that case, a self-learning approach was utilized to efficiently distill the knowledge from the model ensemble and significantly improve the single model accuracy in the target domain.

The final research question investigates whether two previously studied methods, namely model compression, and ensembling, can be efficiently combined. The thesis proposed and implemented a design for multi-input multi-output (MIMO) architecture in the object detection task. The MIMO framework works by predicting multiple images simultaneously, using the same capacity as with the standard model (single image prediction). During inference, all images are the same, and as a result, several potentially different predictions are obtained for a single image. This is possible because of the over-parameterization of current CNN-based models. In practice, it was found that it is hard to obtain more than two parallel predictions for object detection. Overall, the results (including out of distribution accuracy and uncertainty calibration) were similar to model ensembling, with a much smaller computational cost. The detailed analysis showed that one of the reasons for the improved performance is that training the model in the MIMO framework works as a strong regularization.

MOST WIEDZY Downloaded from mostwiedzy.pl

Streszczenie pracy

Celem rozpoznawania obrazu jest identyfikacja oraz lokalizacja wszystkich obiektów zainteresowania w obrazie wejściowym. W związku z ogromną liczbą aktualnie dostępnych danych obrazowych, rola algorytmów rozpoznawania obrazu jest większa niż kiedykolwiek i obejmuje takie m. in. takie dziedziny jak pojazdy autonomiczne, inteligentny monitoring wizyjny czy wspomaganie decyzji personelu medycznego. Współczesne algorytmy przetwarzania obrazu wykorzystują splotowe sieci neuronowe, które w ostatnich czasach pozwoliły na znaczące zwiększenie dokładności rozpoznawania obrazu na wielu zbiorach danych. Niestety, w przypadku zastosowania w świecie rzeczywistym, dokładność tych modeli często znacząco maleje. Jest to najczęściej wynik warunków panujących w czasie testowania, które mogą się znacząco różnić od reprezentowanych w zbiorze treningowym. Zwiększenie zdolności generalizacyjnych modeli do nowych warunków jest jednym z największych wyzwań uczenia maszynowego. Ponadto, wiele modeli opartych o sieci splotowe ma zbyt duże wymagania obliczeniowe lub są zbyt kosztowne.

Celem pracy jest stworzenie modeli rozpoznawania obrazu o możliwie wysokich zdolnościach generalizacyjnych przy jednocześnie niewielkich wymaganiach obliczeniowych. W eksperymentach wykorzystano głównie zbiory danych dotyczące zastosowań analizy obrazu dla potrzeb pojazdów autonomicznych, ze względu na dużą ich dostępność oraz ponieważ opisane wyżej problemy są kluczowe w tym zastosowaniu. Opracowane algorytmy zostały przetestowane dla zadań detekcji obiektów i semantycznej segmentacji.

Ewaluacja zdolności adaptacji modeli do nowych warunków jest wymagającym zadaniem ponieważ zebranie tego typu danych (rzadkie wydarzenia, np. nietypowe warunki atmosferyczne, wypadki drogowe w przypadku zastosowania w pojazdach autonomicznych) często nie jest możliwe. W związku z tym w rozdziale 3 przeanalizowano metody oszacowania zdolności generalizacyjnych modeli na przykładzie zadania detekcji pieszych. Opisane metody wykorzystują testowanie na nowych zbiorach danych, dodawanie syntetycznego szumu w czasie testowania oraz badanie adaptacji modeli do nowych warunków (np. testowanie modeli na zdjęciach nocnych, podczas gdy zostały wytrenowane na zdjęciach uzyskanych w dobrych warunkach oświetleniowych). W toku eksperymentów sprawdzone zostało jak specjalistyczne metody wzmacniania danych wpływają na dokładność modeli oraz oszacowanie ich niepewności. Eksperymenty pokazały, że korzystanie z powyższych metod jest kluc-

MOST WIEDZY Downloaded from mostwiedzy.pl

zowe aby uzyskać możliwie pełen obraz pozwalający na ocenę działania modeli.

W kolejnym rozdziale przeanalizowano dokładniej metody kompresji modeli. Metoda ta działa poprzez usuwanie neuronów lub filtrów sieci neuronowej, pozwalając poprawić szybkość działania sieci, a jednocześnie w niewielkim stopniu wpływając na dokładność modelu. Takie metody są popularne w zastosowaniach przetwarzania obrazu, ale niezbadane zostało do tej pory, jak wpływają one na dokładności modeli w obliczu warunków, znacząco różniących się od zbioru treningowego. Eksperymenty z udziałem modeli detekcji obiektów pokazały, że metody kompresji szczególnie mocno wpływają na zakłócenia związane z szumem (np. szum Gaussowski, który może być efektem zakłóceń czujnika wizyjnego), a w znikomym stopniu wpływają na wrażliwość modeli na efekty związane z rozmyciem obrazu wejściowego. Ponadto, zauważono, że kompresja modeli istotnie obniżyła dokładność rozpoznawania niektórych klas. Założono, że jedną z przyczyn może być fakt, że niektóre klasy są bardziej liczne od innych i podczas kompresji w pierwszej kolejność usuwane są filtry związane z rozpoznawaniem mniej popularnych klas. W dalszym kroku, pokazano, że użycie standardowych technik bilansowania danych może być szczególnie pomocne w takim przypadku. Eksperymenty pokazują, że dokładna ewaluacja i analiza dokładności modeli w różnych warunkach jest szczególnie istotna dla modeli, które muszą działać w świecie rzeczywistym.

W kolejnym rozdziale wykorzystano metody grupowania modeli (ang. *model ensembling*), ponieważ są one znane z bardzo wysokiej dokładności. W metodzie tej uzyskiwane są predykcje dla wielu różnych modeli, a następnie są agregowane, co wiąże się z wysokim kosztem obliczeniowym. W eksperymentach skupiono się na zadaniu semantycznej segmentacji w przypadku adaptacji do nowej domeny (czyli gdy np. mamy do czynienia z innym zbiorem danych w czasie testowania). Aby utrudnić zadanie dla modelu (i tym samym sprawdzić jego zdolności generalizacyjne) w jednym z eksperymentów trenowany jest on na danych uzyskanych z symulatora, a testowany na danych pochodzących ze świata rzeczywistego. Zaproponowana metoda grupowania, korzystająca z 5 modeli (różniących się architekturą oraz metodą augmentacji danych) uzyskała wysoką dokładność segmentacji, również gdy różnica między dziedzinami była bardzo duża. Dodatkowo zauważono, że uzyskane modele charakteryzują się dokładnym oszacowaniem niepewności. Biorąc pod uwagę te wnioski, kolejne zadanie polegało na transferze wiedzy z grupy modeli do pojedynczego modelu, tak aby zmniejszyć koszt obliczeniowy. W tym celu została wykorzystana metoda samo-uczenia (ang. *self-learning*), która pozwoliła na znaczące zwiększenie możliwości adaptacji modelu do nowej dziedziny.

W ostatniej części pracy podjęto próbę połączenia metod opisanych w poprzednich rozdziałach: kompresji oraz grupowania modeli. W pracy zaproponowano wykorzystanie architektury wielokrotnego-wejścia wielokrotnego-wyjścia (MIMO) do zadania detekcji obiektów. Taka architektura działa poprzez uzyskanie wielu równoległych predykcji dla pojedynczego obrazu (a więc podobnie jak przy metodzie grupowania modeli), wykorzystując tyle samo parametrów co standardowy model (dokonujący pojedynczej predykcji). Jest to możliwe, ponieważ współczesne modele

MOST WIEDZY Downloaded from mostwiedzy.pl

oparte o sieci splotowe posiadają ogromnę liczbę parametrów, która często może być efektywnie zredukowana (poprzez kompresję modeli). W praktyce okazało się, że zapro- ponowany model jest najbardziej skuteczny gdy wykonywane są dwie predykcje jed- nocześnie. Ewaluacja pokazała, że zaproponowana architektura jest podobnie skuteczna jak metoda grupowania modeli, przy znacznie mniejszym koszcie obliczeniowym. Wnikliwa analiza działania pokazała, że architektura MIMO pozwala również na efektywną regularyzację trenowanego modelu.

MOST WIEDZY Downloaded from mostwiedzy.pl

Acknowledgements

I would like to thank my supervisor, prof. Andrzej Czyżewski, for all his supervision, advice I have received and the freedom he gave me during my studies to pursue my interests.

I want to thank all my colleagues at the Multimedia Systems Department at the Gdańsk University of Technology, for all support, the time we have been working together and nice atmosphere.

I would like to thank to all my colleagues at ETI faculty. In particular, prof. Jacek Rumiński for his efforts in growing an AI community in Gdańsk, including acquiring GPU servers, and Paweł Rościszewski for my making my life much easier when using them.

I am also very grateful to have an opportunity to work in the Amazon Scout project in Tübingen, which has greatly influenced this thesis and my research interests.

I would like to thank my amazing wife for her endless support and trust in me. And also for letting me spend 6 months of the internship abroad, in the middle of which we had our wedding!

Finally, I want to thank all of my family for everything. Especially I am very grateful to my mom, to whom I dedicate this thesis.

MOST WIEDZY Downloaded from mostwiedzy.pl

List of abbreviations

The following abbreviations has been used throughout this thesis.

acc	accuracy
BDD	Berkeley Deep Drive dataset
CJ	color jittering data augmentation
CNN	convolutional neural network
COCO	Common Objects in Context dataset
CSP	Center Scale Prediction architecture
CT	computed tomography
DE	deep ensembles
DNN	deep neural network
DPM	Deformable Part Model
ECE	Expected Calibration Error
ECP	EuroCity Persons dataset
ENS	effective number of samples method
FCN	fully convolutional network
FN	false negative
FP	false positive
fppi	false positives per image
GPU	graphical processing unit
GTA	Grand Theft Auto
HOG	histogram of oriented gradients
i.i.d	independent and identically distributed
INV	inverse square root re-weighting method
IoU	Intersection over Union
LAMR	log-average miss rate
mAP	mean average precision
MIMO	Multi-input Multi-Output framework
mIoU	mean Intersection over Union
MLP	multi-layer perceptron
mr	miss-rate
MRI	magnetic resonance imaging
MSE	mean squared error
NMS	non-maximum suppression
o.o.d	out of distribution

MOST WIEDZY Downloaded from mostwiedzy.pl

PCA	principal component analysis
pix. acc	pixel accuracy
R-CNN	Region-Based Convolutional Neural Network
ReLU	Rectified Linear Unit
RFS	repeat factor sampling method
ROI	region of interest
RPN	Region Proposal Network
SGD	stochastic gradient descent
SVM	support vector machine
TP	true positive
WBF	Weighted Boxes Fusion

MOST WIEDZY Downloaded from mostwiedzy.pl

List of Figures

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl

List of Tables

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl

Contents

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 1

Introduction

1.1 Foreword and Motivation

Recent years have witnessed noteworthy progress in the application of machine learning models to the real world. DeepMind company has developed an algorithm to play GO (Chinese chess-style game) that defeated the world champion [1], a goal that many considered a milestone in the development of artificial intelligence. Other prominent examples are significant improvements in machine translation [2], applications of visual recognition to medicine [3], and face recognition [4].

The current machine learning revolution started in 2012 when a large convolutional neural network (CNN) significantly improved image recognition performance over previous work [5]. However neural networks existed long before 2012; some of the foundations were established in the 1950s. For example, the perceptron network was the first example of a neural network built for image recognition [6]. Still, many architectural details such as the number and size of layers, appropriate non-linear functions, and weights initialization were required to make them work. Because modern neural networks usually contain many hidden layers, they are often referred to as deep learning.

Recent success would not be possible without the increasing computing power (thanks to Graphic Processing Units – GPUs) and the availability of large-scale datasets. A very prominent example is the ImageNet dataset [7]. It contains 14 million images grouped into 22,000 visual categories. Such datasets play a vital role in the field. First, they provide a lot of data for training current deep learning models known as data-hungry. Second, they are used for benchmarking and measuring progress, encouraging scientists to test their algorithms.

With the increasing potential of machine learning, there comes a need for building trustworthy algorithms, which can be used, for example, in autonomous driving or medical applications. One of the main challenges when applying machine learning algorithms to the real world, is the fact that the real world is very complicated, and no matter how big the dataset is, it will contain only an infinitely small

MOST WIEDZY Downloaded from mostwiedzy.pl

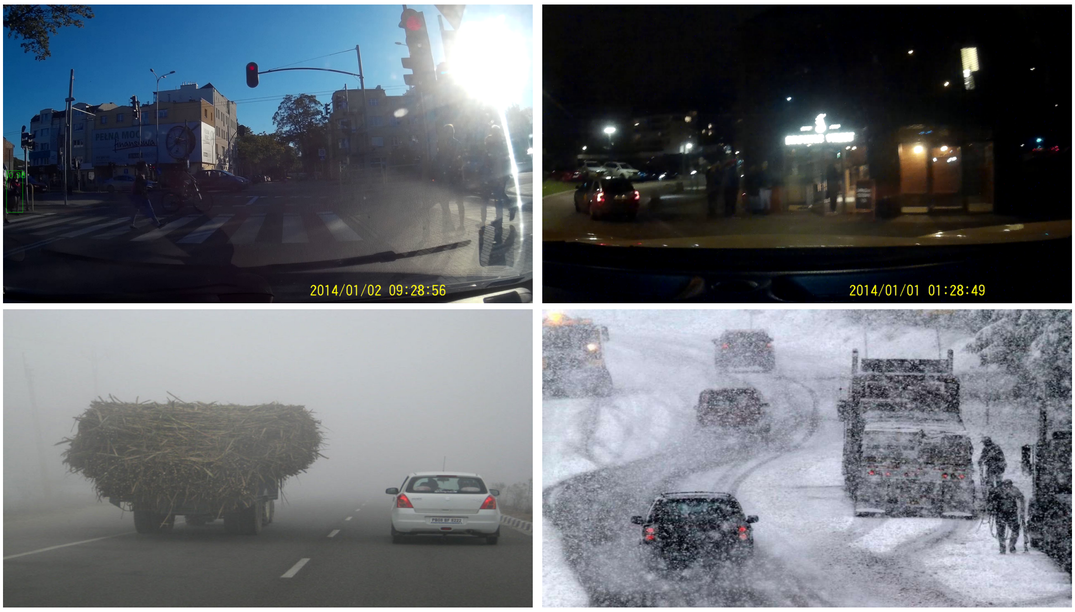


Figure 1.1: Examples of some challenges in visual recognition: overexposed images (upper left), camera movement and low light (upper right), adverse weather conditions (bottom row) and atypical object instances (bottom right). The upper row shows images recorded from the author's camera, the bottom row figures are from the DAWN dataset.

part of all of the possible situations that may occur. For example, in the case of autonomous driving, it is impossible to collect a dataset that will contain all of the different weather conditions, types of vehicles, or road situations.

Visual recognition plays a vital role in machine learning. With the ubiquitous presence of visual data across various domains (e.g., medicine, autonomous driving, surveillance, social media, computer games), the role of computer vision algorithms is more significant than ever. At the same time, many of these algorithms are deployed to new environments which may be different from those found in the training dataset, which may drastically reduce recognition accuracy. As the problem is very challenging and of great importance, the author is interested in developing algorithms that can be effectively used for visual recognition for models deployed to the real world.

Reliability is crucial for systems operating in the real world since current deep learning models are very vulnerable to changes in the data distribution [8, 9], do not generalize well across datasets [10, 11, 12], and tend to base their prediction on superficial features [13, 14, 15]. Such models perform very well only when the test-time distribution matches or is close to the training-time distribution, otherwise current models can make unexpected predictions in the presence of a distributional shift (Fig. 1.1). This is not an issue for many applications, e.g., when operating in a closed environment (e.g., inside a factory) or when it is satisfactory for the model to work well only in the 'typical' case (for example, in entertainment applications). How-

MOST WIEDZY Downloaded from mostwiedzy.pl

ever, for safety-critical applications operating in the real world (medicine, robotics, autonomous vehicles), such behavior is unacceptable. Therefore, **the first goal of this thesis is to design an algorithm that can be safely deployed to an unfamiliar environment, meaning it is resistant (robust) to a shift of data** (for example, due to the sensor noise or different weather conditions than those in the training data).

Regarding robustness, the conventional wisdom in the machine learning community is that “bigger models are always better” [9, 2, 16]. However, such an approach is not feasible for real-world deployment when the computational cost is important. Hence, **the second goal of this work is to develop algorithms that are computationally efficient**. When reducing computational cost, one needs to analyze the trade-off between hardware cost and model efficiency carefully.

It is also important for models operating in the real world to “know when the model does not know”. In such a case, the algorithm might inform the medical doctor that the prediction may be unreliable or ask the driver to take control of the steering wheel in the case of autonomous driving. Because of that, it is also crucial that developed algorithms provide reliable uncertainty estimates in order to safely act in the real world. As such, in this thesis, the methods for improving uncertainty calibration under distributional shift are studied.

As it was mentioned before, large-scale datasets are an essential aspect of the progress in machine learning. Because many such datasets exist for autonomous driving and because robustness to distributional shift is important for that task, most experiments in this thesis are conducted using autonomous driving datasets. However, the methods developed in this thesis are general, and their usage is by no means limited to autonomous driving.

1.2 Scope and Contributions

The goal of this thesis is to develop robust and efficient models for visual object recognition. High-level visual recognition tasks such as object detection and semantic segmentation are essential in computer vision applications such as surveillance systems, autonomous driving, and medicine. Semantic segmentation can be viewed as a pixel-wise classification problem where the goal is to assign to each pixel a predicted category $c \in \{1, ..., C\}$. Each label c represents a different class (e.g., pedestrian, car, bicycle). For object detection, the goal is to find the coordinates of important objects in the observed image using bounding box annotations. In the case of semantic segmentation, a list of classes is usually extended by adding a background class. A simple illustration is presented in Fig. 1.2. Both topics are explained in more detail in sections 2.2 and 2.3.

While visual recognition is a problem that, thanks to the recent advances in computer vision, can currently be solved in typical cases (e.g., clear weather and good lighting conditions, typical traffic scene in the case of autonomous driving),

MOST WIEDZY Downloaded from mostwiedzy.pl

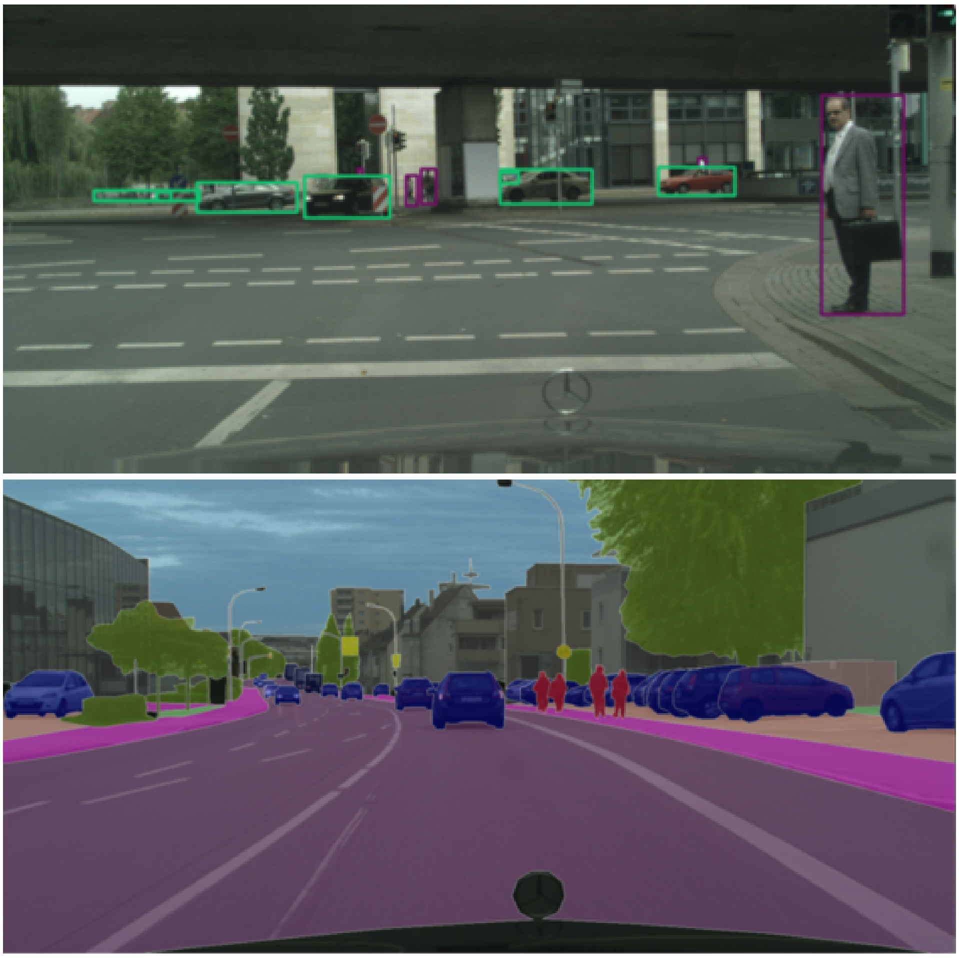


Figure 1.2: Example ground truth annotations for the object detection (upper row) and semantic segmentation (bottom row) task from the Cityscapes dataset.

it is challenging to create a system that will work with near 100% accuracy in all situations, which is crucial for developing systems operating in the real-world. Typical challenges in object detection and semantic segmentation include:

- changes in the illumination significantly alter objects' appearance. Recognizing objects in insufficient illumination (i.e., at night) is a significant challenge for the community,
- adverse weather conditions (heavy rain, snow),
- object occlusion,
- large variability in the appearance of different objects. In the case of pedestrians, those might be caused by diverse types and styles of clothing. Also, different accessories such as sunglasses, hats, backpacks further significantly change pedestrian appearance,
- presence of rare events (e.g., very rare medical conditions, vehicle accidents),
- domain adaptation. When the models are deployed to the real world, the conditions might differ from those in the training dataset.

MOST WIEDZY Downloaded from mostwiedzy.pl

Recent neural-network-based computer vision algorithms have achieved impressive performance on many benchmarks, but they lack robustness to novel conditions (e.g., weather or lighting conditions that were not a part of the training data) and may produce erroneous predictions in such cases. Additionally, measuring progress in the models' robustness is very hard, especially for systems operating in the real world since it is usually unfeasible to collect a testing dataset that would contain all possible difficult situations for the visual perception system (e.g., adverse weather and lighting conditions, sensor noise, rare events). A meaningful way to measure model robustness is using out-of-distribution (o.o.d.) data [17], which tests a model on data from different distributions than the training data. However, it was only recently that out-of-distribution settings started to gain more attention in the research community. O.o.d. testing includes cross-dataset evaluation, evaluating models on conditions not seen during training (e.g., weather type not seen during training), or using synthetically generated distortions (e.g., simulated sensor noise). Throughout this work, several ways of measuring model robustness are evaluated.

At the same time, it was shown that current machine learning models are heavily over-parametrized, which allows them to fit random labels [18]. Optimizing the size of visual recognition models is essential in many applications because of the energy consumption and hardware cost. Consequently, many methods were developed to reduce the computational cost, including model pruning (removing model parameters) [19], quantization (using lower precision computation) [20], and designing specialized architectures [21]. On the other hand, it was shown that bigger models are more robust [9]. From the safety perspective, it is essential to study the effect of reduced model capacity (reduced network parameters). It was only recently shown that pruning significantly affects robustness in the image classification task and might disproportionately impact different object classes [22]. When reducing model capacity, it is hypothesized that information about rare classes might be lost first [23]. We hypothesize that using standard methods for data balancing could be effective in such setting, which formulates the first hypothesis of this work:

- **pruning neural networks impacts model accuracy disproportionately on different object classes for object detection, and data balancing methods could be highly effective in such conditions**.

On the other hand, particularly good accuracy can be obtained by an ensemble of models [24], which involves aggregating results from many classifiers, which is a standard trick to improve final model accuracy. While model ensembling has a high computational cost, which forbids its use in real-time systems, we investigate how ensembling can be used for domain adaptation settings (when there is limited data in the target environment). In this thesis, the model ensembling was combined with a self-learning approach [25]. Self-learning is an approach where an initially trained model (using available datasets) is used to generate predictions on the target domain. Then, the generated pseudo-labels (after some filtering) are used to adapt the model to the target domain. This leads to the formulation of the second hypothesis of this

MOST WIEDZY Downloaded from mostwiedzy.pl

work:

- **model ensembling can be efficiently combined with a self-learning approach for the domain adaptation task in visual recognition**.

Finally, a direct connection between the work on model compression and model ensembling is drawn in the last paragraph. Since current CNN-based models are heavily overparameterized, recent work results have shown that they can fit (train) more than one subnetwork within the original model capacity [26, 27, 28] for the more straightforward image classification task. Thus, it obtains multiple predictions (for the same image) with a minimal increase in computational cost. As such, the last hypothesis of this work is formulated as follows:

- **it is possible to obtain parallel-predictions from single object detection model, to improve model robustness**.

Scope of the thesis. While object detection and semantic segmentation are also studied in the context of another type of sensor, i.e., LiDAR, this work focuses on RGB cameras only. Such sensors are much more widespread and cheaper, thus creating algorithms for visual recognition using only RGB cameras has excellent potential for applications, which is the strategy is followed by the Tesla company.

Also, in this work, the subject of interest is visual recognition from a **single image**. Visual recognition from video sequences is a whole field by itself, which utilizes additional techniques such as temporal smoothing and localization. However, single-image recognition models are often an important component of video-based visual recognition models, which means that video-based visual recognition is likely to benefit from improved recognition from a single image. Also, a large gap still exists between human and machine vision in out-of-distribution setting [29, 30, 31], which ideally should be closed. Thus, visual recognition from a single image is a vital topic in computer vision. This also makes the methods developed in this thesis applicable across various domains.

1.3 Thesis outline

Section 2 outlines the essential aspects for the task of object detection and semantic segmentation. It starts with a brief introduction to supervised learning and describes the learning process. Then the bias-variance trade-off is introduced, which is a particularly important problem throughout the work. Further, the Convolutional Neural Networks (CNNs) for the object detection and semantic segmentation tasks are described, together with the datasets and evaluation metrics used throughout the work. Finally, ways to measure models' robustness are explored, and metrics used throughout this thesis are presented.

Section 3 shows the importance of out-of-distribution testing (using pedestrian detection as an example) and explores different data augmentation techniques.

MOST WIEDZY Downloaded from mostwiedzy.pl

Various strategies for evaluating model robustness are investigated. It was shown that some data augmentation strategies improve model robustness but may reduce model accuracy on the clean (uncorrupted) dataset. Simultaneously, some data augmentation strategies improve model accuracy on the clean dataset without affecting model robustness. A simple data augmentation strategy was proposed using those observations, which achieved competitive results. The evaluation setting used in this section will be used throughout the rest of the thesis.

Section 4 shows how the robustness in object detection is affected by the model capacity controlled by means of compression techniques. It was shown that pruning affects models' sensitivity to different distortion types and their accuracy for specific classes with different impacts. Several methods for handling data imbalance were evaluated, and it was shown that data balancing methods might be helpful in compressed neural networks, which confirms the first thesis of this work.

In section 5, accuracy and uncertainty calibration are evaluated in the distributional shift setting for the high-level vision task of semantic segmentation. Model ensembling is used to aggregate predictions, which significantly improves the performance, including the challenging task of adaptation from simulation. Finally, ensemble of models is used as a teacher of a single model, which allows for efficient domain adaptation.

Section 6 investigates whether combining the advantages of the techniques described in the previous sections (model compression and ensembling) is possible. A multi-input multi-output object detection architecture is proposed, which significantly improves the robustness with minimal computational cost.

Finally, the last section discusses the contributions reported in this thesis, which provide discussion and directions for future work.

1.4 Publications

This section lists all peer-reviewed journal and conference publications that the author has contributed throughout the Ph.D. studies. The core publications used this thesis are listed below:

- Section 3 - S. Cygert, A. Czyżewski, "Toward robust pedestrian detection with data augmentation", IEEE Access vol. 8, 2020, pages 136674-136683.
- Section 4 - S. Cygert, A. Czyżewski, "Robustness in Compressed Neural Networks for Object Detection", International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 2021.
- Section 5 - S. Cygert, B. Wróblewski, R. Słowiński, K. Woźniak, A. Czyżewski, "Closer Look at the Uncertainty Estimation in Semantic Segmentation under Distributional Shift", International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 2021.

MOST WIEDZY Downloaded from mostwiedzy.pl

- Section 6: Robust Object Detection with Multi-input Multi-output Faster R-CNN. This paper was accepted to be presented at the 21st International Conference on Image Analysis and Processing (ICIAP 2021).

Additionally, the following works are related to the thesis:

- S. Cygert, F. Górski, P. Juszczyk, S. Lewalski, K. Pastuszak, A. Czyżewski, A. Supernat, ”Towards Cancer Patients Classification Using Liquid Biopsy”, PRIME workshop at MICCAI conference 2021. This paper evaluates impact of several regularization techniques on model performance, using medical data.
- S. Cygert, A. Czyżewski, ”Vehicle Detection with Self-Training for Adaptative Video Processing Embedded Platform”, Applied-Sciences vol. 10, 2020, article number - 5763. This paper explores a self-training approach to improve model adaptation to the target environment.
- S. Cygert, A. Czyżewski - ”Evaluating Calibration and Robustness of Pedestrian Detectors”, International Conference on Multimedia Communications, Services and Security (MCSS) 2020.
- S. Cygert, A. Czyżewski, ”Style Transfer for Detecting Vehicles with Thermal Camera”, Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), 2019. This work proposes to use style-transfer data augmentation for adapting models trained on RGB images to the images obtained from the thermal camera.
- S. Cygert, A. Czyżewski - ”Vehicle detector training with labels derived from background subtraction algorithms in video surveillance”, Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA). This paper explores fine-tuning the object detection model using weakly supervised techniques.
- D. Węsierski, S. Cygert - Shape-Based Pose Estimation of Robotic Surgical Instruments, MICCAI Workshop on Computer-Assisted and Robotic Endoscopy (CARE), Quebec City, Canada, Best Paper Award (2nd place). This paper explores the task of surgical tool detection using an ensemble of shape-based templates.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 2

Background and related work

Visual recognition is an important research area that spans a few decades, which lies at the intersection of machine learning and computer vision. This section introduces essential concepts for the rest of the thesis. It starts with a brief introduction to machine learning and supervised learning (section 2.1). Further, modern models for object detection (section 2.2) and semantic segmentation (section 2.3) are described. Datasets used in this thesis are presented in section 2.4, while section 2.5 describes related works on the robustness of evaluated models. Finally, evaluation metrics such as accuracy and uncertainty calibration are presented (section 2.6). Since the subject related to visual recognition is extensive, this section only shows what is indispensable in this thesis to keep it concise.

2.1 Supervised learning

Machine learning algorithms build mathematical models based on the training data, which are then used during test time to make predictions on new data. Each input is typically represented as a vector $x \in R^d$, where d represents values for different features, hence the vector is often called a feature vector. Thus, the whole dataset, consisting of n examples, can be defined as the matrix $X \in R^{n*d}$. Machine learning is traditionally divided into three general categories[32]:

- supervised learning. The model is given a training set consisting of input vectors x_i, the desired output y_i, and the goal is to learn a mapping from the input to the output,
- unsupervised learning. In this scenario, no labeled data is available. Given only the unlabeled data, the goal is to find structure in the input, for example, dividing the input into similar clusters. Unsupervised learning is also commonly used as the feature extraction step,
- reinforcement learning. An agent can interact with the environment, in which it learns to satisfy a particular goal. Feedback (reward function) is used to find the model parameters.

MOST WIEDZY Downloaded from mostwiedzy.pl

This is only a high-level partition of the learning types, and some approaches cannot be easily categorized. Semi-supervised learning is an approach where only some of the data have a label, therefore learning in such a model usually consists of supervised and unsupervised loss functions. Self-supervised learning contains no labels, and the model is used to solve a so-called pretext task in a supervised manner. Regarding the tasks of object detection and semantic segmentation (and visual recognition in general), supervised learning is the most relevant type and is presented with more details in the following subsections.

2.1.1 Loss function

Supervised learning can be formulated as follows: given a labeled dataset $D = \{(x_i, y_i), i \in n\}$, where x_i is the i-th example, and y_i is its label (e.g., class), find model parameters w which parametrizes a function $f_w : X- > Y$, where X is the input space and Y is the output space. A function f_w is a chosen model (e.g., logistic regression). Given a model, the goal of supervised learning is to find its parameters w such that the function f_w best fits the training data. How well a function fits the training data is defined by the loss function $L_D(w)$. To reduce the chance of overfitting to the training data often a regularisation term $L_W(w)$ is added. The whole loss can be defined as:

$$L(w) = L_D(w) + \lambda L_W(w) \tag{2.1}$$

where λ is the regularization scaling factor. Hence, finding w is an optimization problem:

$$\overset{*}{w} = \operatorname*{argmin}_{w} L(w) \tag{2.2}$$

which is also called empirical risk minimization. The choice of the loss function depends on typy of the problem: classification or regression. In classification, a target label is a category (e.g., classifying patients into healthy and unhealthy). In regression, the output variable is a real value (e.g., depth estimation in the input image).

One of the simplest supervised learning algorithms is linear regression. **Linear regression** takes as an input the vector $x \in R^d$ and outputs the scalar value $y \in R$. The output is a linear function of the input, given the input x, the predicted value $\hat{y}$ is computed as:

$$\hat{y} = w^T x + b \tag{2.3}$$

where $w \in R^d$ is a vector of model parameters, often referred to as **weights**, and $b \in R$ is the bias term. Note, that the equation corresponds to a linear function plus

MOST WIEDZY Downloaded from mostwiedzy.pl

a constant. Each input feature x_i is associated with its feature weight w_i. If the feature x_i receives the positive weight w_i, an increase in the value of that feature also increases the prediction; similarly, the negative weight w_i implies that an increase in the value of the feature decreases the value of the prediction. At the same time, a large magnitude of the weight w_i, means that the feature x_i has a large effect on the prediction, while zero weight means that the feature x_i is ignored in the model.

The mean squared error is a natural choice for the loss function (also known as the cost function or objective function). Given the model prediction for the i-th training sample: $\hat{y} = f_w(x_i)$, the MSE loss is computed as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{2.4}$$

In simple linear problems (with a small dataset size), weights for the linear regression can be computed using a closed-form solution. However, many modern machine learning problems defined over large datasets are solved using the gradient descent methods described in section 2.1.3.

Linear regression can also be applied to the classification problem, where the label y_i belongs to a predefined number of classes. In binary classification, there are two classes: class 0 and class 1, the output from the linear regression can be then 'squashed' into the [0, 1] interval using the sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.5}$$

which results in the model prediction for an input belonging to one of the classes. Such a model is also known as **logistic regression**. In the case of multi-class classification, the problem model returns a logit vector $z \in R^C$, where C is the number of classes. A softmax function is then applied to obtain a categorical distribution.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{2.6}$$

Popular choice of loss function for a classification problem, is a cross-entropy function. Given a ground truth distribution $p(y|x)$ and model predictions $\hat{p}(y|x)$ the cross-entropy is computed as:

$$H(p, \hat{p}, x) = -\sum_{i=1}^{C} p(y_i|x) log\hat{p}(y_i|x) \tag{2.7}$$

This allows to formulate a cost function:

$$J(w) = \frac{1}{n}\sum_{i=1}^{n} H(p, \hat{p}, x_i) \tag{2.8}$$

MOST WIEDZY Downloaded from mostwiedzy.pl

Linear regression and its classification equivalent, logistic regression, are simple supervised learning algorithms that serve as a basis for other approaches. However, another supervised learning algorithm, namely neural networks, is fundamental in high-dimensional inputs such as images.

2.1.2 Neural networks

Neural networks are a common choice for dealing with the input of very high dimensionality (e.g., images, natural language processing). Building a connection with the previous section, logistic regression can be viewed as the simplest form of a neural network consisting of input and output layers. In the case of multi-layered perceptron (MLP), at least one hidden layer is added between those, and it can be written for the classification problem as:

$$\begin{aligned} h &= \sigma_1(W_1 x + b_1) \\ y &= softmax(W_2 h + b_2) \end{aligned} \tag{2.9}$$

where σ is the activation function, W is the weight matrix, and b is the bias vector for the corresponding layer. Cells in each layer are often referred to as neurons. There can be any number of hidden layers. While MLP with one hidden layer is a universal approximator function [33], modern deep neural networks contain many hidden layers (usually dozens), since it makes current architectures more effective at solving certain problems [5].

Computing the output of the neural network is also known as a *forward pass*. The predicted value is compared against the ground truth labels during training, and an error is calculated (using a loss function). The error is then backpropagated through the network, and the weights are updated in relation to the amount that they contributed to the error. This is achieved by computing the gradient of the cost function concerning the individual weights of the neural network. The chain rule makes it possible to compute the gradient, one layer at a time, iterating backward from the last layer, hence the name of the procedure: *backpropagation*. Those operations are computed over the whole training dataset. The next section provides more details on gradient descent optimization, which can train different machine learning algorithms over large training datasets.

An essential aspect of a neural network is the activation function, which accepts the output from the previous cell, and converts it into a form that serves as an input to the cell in the next layer. Usually, activation functions are non-linear, which allows the neural network to learn complex patterns from the data. As an example, it is impossible to model a simple XOR gate using only linear activation functions. The rectified linear unit (ReLU) is a vital activation function used in modern neural networks. In contrast to the previously used non-linear functions such as *softmax* and *hyperbolic tangent (tanh)*, it is less prone to the vanishing gradient problem. When neural networks consist of many layers, the chain rule will make the gradient

MOST WIEDZY Downloaded from mostwiedzy.pl

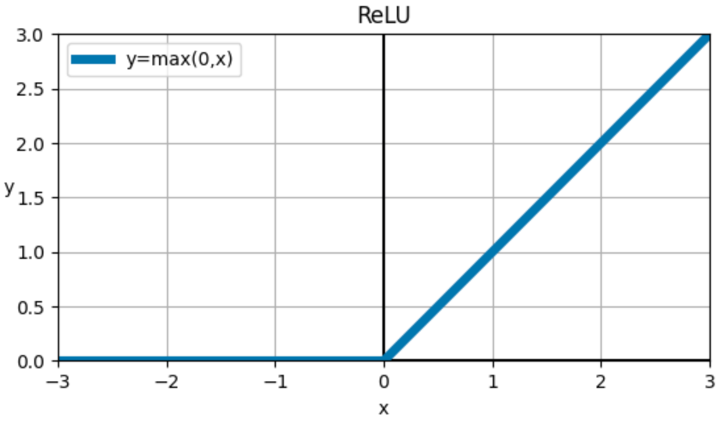


Figure 2.1: ReLU activation function

for the initial layers shift toward 0, which was a large problem in properly training deep neural networks before the introduction of the ReLU activation function. Interestingly, ReLU activation was loosely inspired by the functioning of the cortex [34]. An additional relevant property of the ReLU activation function is that it provides some level of sparsity, the output is non-0 only when the input is larger than 0.

MLP is an example of a fully connected neural network since each cell in one layer is connected to every cell in the following layer. As a result, even simple MLP architectures working on real-world high dimensional data can result in a model containing millions of parameters. The main building blocks of modern neural networks have already been known for a very long time. The first MLP architecture was described by Rosenblatt in 1961 [6], similar to backpropagation which was first presented in 1960 [35]. However, large datasets, computing power, and design tricks that stabilize training (such as the ReLU activation function) were still required to make neural networks work for real-world data. Another prominent development in the history of neural networks was the introduction of convolutional neural networks (CNNs), which drastically reduced the number of parameters for high dimensional input data such as images.

Convolutional Neural Networks. CNNs were already known in the 1990s. One of the first successful implementations was the LeNet5 architecture by Yann LeCun [36], which was used for recognizing characters in documents. CNNs were inspired by how the visual cortex works [37, 38], where individual neurons respond to stimuli only in a certain region of the visual field (receptive field). Similarly, in CNNs each neuron has a restricted receptive field and computes its response on small input image patches. Each neuron is defined by a small filter (usually of size 3x3 or 5x5) and computes its response to the input signal using the convolution operation:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{u=-k}^{k} H[u,v]F[i-u,j-v] \tag{2.10}$$

where H is the input image, F is the filter of size $2*k+1$. CNNs make extensive

MOST WIEDZY Downloaded from mostwiedzy.pl

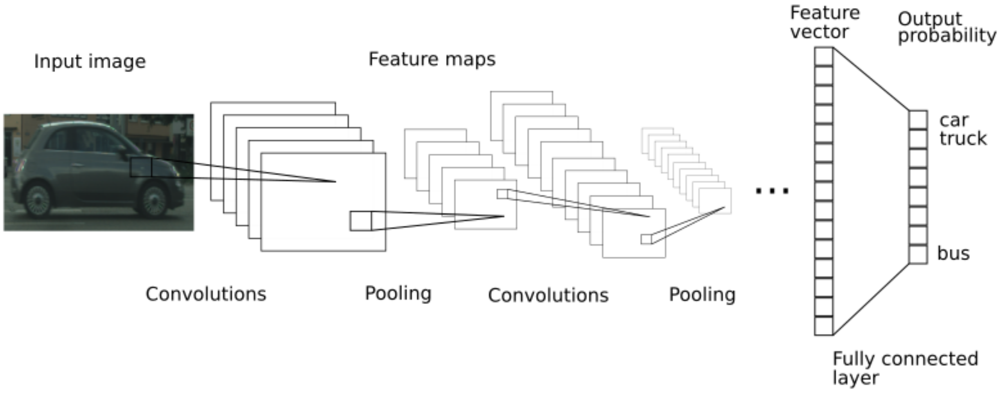


Figure 2.2: Schematic overview of a CNN architecture for image classification.

use of parameter sharing – the same feature is computed at various locations in the input, which allows CNNs to be translation invariant (to some extent), making CNNs computationally efficient. Between subsequent layers, a pooling operator is applied, which subsamples the feature map using the max or average operator, effectively reducing the size of the feature map. Typically, a subsample by a factor of two is used.

There are many reasons for CNNs being particularly useful in image processing:

- CNNs are equivariant to translation: if an object is translated in an image, then the convolution outcome is equally translated [39],
- Pooling operation makes the network invariant (to some degree) to small image translations and distortions,
- CNNs explore local spatial correlation in natural images; each neuron is connected to only a small region of the input volume,
- They work as feature extractors by providing a feature vector after the last convolutional layer in the network. Again, this is especially useful when a large-scale dataset is available. It was shown that features learned on large-scale datasets transfer well to a new domain, where only limited data are available. This is also known as transfer learning,
- Weight sharing, using pooling operations, make them very computationally efficient, especially for parallel processing with GPUs.

For most practical applications, the original input variables will be of very high dimensionality. Consider a camera placed inside a car with a modest (by current standards) resolution – 800x600 pixels. The dimensionality of the input size would be 800 * 600 * 3 (RGB channels) = 1,4440,000. When working with such high dimensional data, reducing its dimensionality by projecting the data to a lower-dimensional subspace is important [32]. Looking at Fig. 2.2, one can also view the whole architecture consisting of feature extraction and classification. CNNs are responsible for extracting feature vectors from the input image. In the last

MOST WIEDZY Downloaded from mostwiedzy.pl

fully connected layer, an image is classified as belonging to one of the classes, as in logistic regression. Computer vision has a long history of useful computing features for image processing, and nowadays, CNNs are commonly used in many computer vision pipelines.

As a side note, it is worthwhile drawing a connection to the classic algorithms in computer vision (before the application of neural networks). Different filters (e.g., Gaussian, and gradient filters) are commonly used in various computer vision pipelines, such as the Canny algorithm's edge detection [40]. In modern CNNs, on the other hand, the filters are learned by the machine learning algorithm itself. Interestingly, there is some evidence that filters in the early layers of the CNNs work as edge detectors, while filters in the deeper layers respond to high-level concepts (such as objects parts) [41, 42].

2.1.3 Training

Where no closed-form solution is available, machine learning algorithms are typically trained using the *stochastic gradient descent (SGD)* algorithm to minimize the objective function $J(w)$. It is a stochastic approximation of the *gradient descent* method as the gradient is computed over some small subset of the training examples (called a mini-batch) instead of the gradient computed over the entire dataset. The first step includes computing the gradient loss concerning the model parameters. The gradient can be computed using the chain rule because all operations in the neural networks are differentiable. In the gradient descent method, the gradient is computed on the whole dataset, which is very computationally expensive:

$$\nabla_w J(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla_w L(x_i, y_i, \hat{y_i}, w) \tag{2.11}$$

On the opposite side is *stochastic gradient descent*, which computes the gradient using only a single data point:

$$\nabla_w J(w) \approx \nabla_w L(x_i, y_i, \hat{y_i}, w) \tag{2.12}$$

While this can be efficiently computed, it results in much more noisy gradient estimation. Mini-batch gradient descent combines those two approaches and computes the gradient over a mini-batch of m examples:

$$\nabla_w J(w) \approx \frac{1}{m} \sum_{i=1}^{m} \nabla_w L(x_i, y_i, \hat{y_i}, w) \tag{2.13}$$

which results in fast and robust optimization of the objective function. In modern machine learning, the mini-batch size m usually ranges from 2 to a few hundred (in computer vision) and is the default setting known as simply *stochastic gradient*

MOST WIEDZY Downloaded from mostwiedzy.pl

descent.

Once the gradient loss is computed, the model parameters w can be updated:

$$w = w - \eta * \nabla_w J(w; x^{i:i+m}; y^{i:i+m}) \tag{2.14}$$

where η is the training step (learning rate), which is often the most important hyperparameter to be chosen for the training. A common approach is to reduce the learning rate in the later stages of the training or use learning rate annealing to improve model convergence [43]. Model weights are updated in iterations, where training samples are sampled from the training dataset (usually without repetition). An epoch is a machine learning term that indicates passing over the entire training dataset. Modern machine learning models (especially high-capacity neural networks) often contain millions of parameters that can easily overfit the training data [18]. Hence, a standard technique is to split the dataset into training and validation parts. While the gradient descent optimizes the loss in the training part, the loss (or accuracy) in the validation set is monitored. Model for testing is then chosen using the best performance on the validation part. The model performance is then usually reported on some other test dataset that was not used during training. The performance on the test datasets is then used as a proxy for the model's ability to generalize to new data. Cross-validation can also be viewed as a regularization technique that helps to improve model generalization properties, which is a central problem in machine learning, so it is explored in the next subsection.

2.1.4 Bias-variance trade-off

Machine learning is about creating a model that works well on the training set and performs well on the unseen test data, which is called generalization, which is a fundamental problem. In machine learning, the training and test data are often assumed to be independently and identically distributed (i.i.d. assumption). Samples in the dataset are produced by a data generating process, which itself is a probability distribution. It means that the samples in each dataset (training and test) should be drawn from the same distribution [44].

While the training error is optimized during training, the goal is to find a model with minimal test error. It turns out that the relation between these two is nuanced. In particular, one can obtain a model that achieves zero training errors but performs poorly on the test set. For example, consider that data points obtained from a noisy signal correspond to a sine curve. Trying to reconstruct that line, one could use polynomial regression (which is a special case of linear regression), that is, to approximate the signal with the equation:

$$f(w) = w_0 + w_1 * x + w_2 * x^2 + w_3 * x^3 ... \tag{2.15}$$

Before solving the equation, one needs to decide on the degree of the polyno-

MOST WIEDZY Downloaded from mostwiedzy.pl

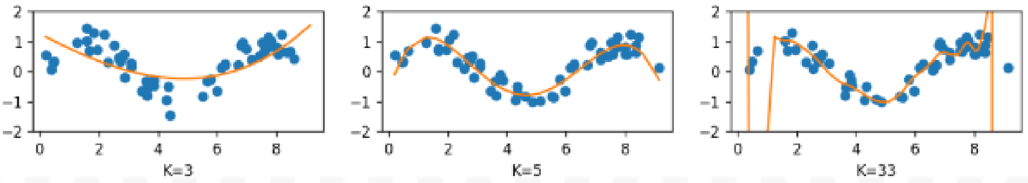


Figure 2.3: Modelling sine curve with polynomial regression of different degrees (K). The left column shows an example of underfitting and the right column of overfitting. Only the center model has appropriate capacity.

mial, which corresponds to controlling the model capacity. If the degree is too small, the fitted curve will not be well approximated (Fig. 2.3 – first column), which is called underfitting. If the degree is too big, the curve will be well fitted, including all of the underlying noise in the signal (overfitting, column on the right in Fig. 2.3). Finding an optimal model is also known as a bias-variance trade-off.

While underfitting might not be a problem, since many modern neural networks have many parameters, overfitting is an important issue. It was shown that neural networks have a large capacity, which allows them to fit even a random noise [18], with zero training error. Model capacity is only one of the many crucial factors that impact the final model generalization properties. To optimize the trade-off, several regularization techniques are used:

- regularization of neural network weights. Additional loss is added to the norm of the matrix weights, which penalizes large weights (eq. 2.1),
- Dropout regularization [45]. During training, some weights of the neural network are removed (their weights are set to zero), which forces the model to focus on many input features, instead of overfitting to just a few ones,
- data augmentation prevents models from memorizing data points. In the case of image processing using random affine transformations (translation, scale, rotation), image flipping, and changing colors during training is standard practice,
- cross-validation,
- model ensembling. Aggregating the outputs of many models has been shown to be a highly effective approach [24].

Currently, measuring the performance of machine learning algorithms on an i.i.d. test set is standard practice; however, this can be very misleading, especially for the models deployed to the real world. This is because it is often impossible to gather data that represents all situations encountered after deployment in the real world. In the case of autonomous driving systems, after deployment, they will often be faced with situations that were not available in the training dataset, e.g., adverse weather conditions (fog, heavy rain, or a combination of both), different types of road crossings, and a new type of vehicles. Furthermore, machine learning

MOST WIEDZY Downloaded from mostwiedzy.pl

algorithms are very vulnerable to even slight changes in the distribution during test-time [46, 47], which will be discussed in more detail in section 2.5. While i.i.d. test set validation is an important evaluation practice, it can often produce a false sense of security [17]. Recently, testing algorithms on various datasets (in cross-dataset setting) and using out-of-distribution data (o.o.d) has been gaining more attention and is followed throughout this work.

2.2 Object Detection

Object detection is a highly active field of computer vision for both academia and real-world applications such as autonomous driving, security monitoring, robot vision, and many more. It is also an intermediate step in other computer vision tasks, such as object tracking, image captioning, and instance segmentation. Object detection aims to discover visual instances of certain classes (humans, vehicles, etc.) in digital images. While previous sections describe solving classification or regression tasks, object detection is an example of a multi-task problem: the goal is to find whether some classes are visible in the image (classification) and localize those objects (regression problem). The fundamentals of modern object detectors are outlined in the following sections, and Faster-RCNN [48], a popular object detection algorithm, is presented in more detail.

2.2.1 Modern Object Detectors

The development of object detection methods is often divided into two groups: traditional methods and deep learning-based methods (starting from 2014). A typical object detector consists of the following components: feature extraction, region of interest proposals generation, and regions of interest classification. In the early days of computer vision, feature extraction was a critical part of the systems, which gained a lot of attention from the research community, as there was no standard feature extraction method.

The first successful object detector described by Paul Viola and Michael Jones [49] in 2001 was based on Haar-like features for real-time face detection. The algorithm used the simplest method for region proposals generation, namely sliding window. For each image, an exhaustive list of bounding boxes at predefined positions are generated and further classified as containing a human face or not. Proposals are also generated at different scales and aspect ratios to incorporate variations of objects in the image region. It results in a substantial number of region proposals, which is the biggest drawback of the sliding window approach. Intuitively, most region proposals will not contain any object, and it should be possible to discard such proposals efficiently. This is the main idea behind the Viola-Jones detector, which uses a cascade of gradually more complex classifiers. Early classifiers in the cascade consist of simple classifiers, which can efficiently discard simple negative windows.

MOST WIEDZY Downloaded from mostwiedzy.pl

More complex classifiers in deeper layers of the cascade are only used to classify more complex image proposals. It allows the classifier to run in real-time efficiently, and the idea of cascaded predictions is also used in modern machine learning pipelines [50].

Another important work was the Histogram of Oriented Gradients (HOG) feature descriptor successfully applied in 2005 for human pose estimation [51]. The descriptor counts occurrences of gradient orientation over a dense grid of uniformly spaced cells. Compared to previous approaches, such a descriptor increased the scale, translation, and illumination invariance and was successfully applied to the pedestrian recognition task. While such an approach can work well for "easy" cases, occlusions or objects seen in rare poses are still particularly challenging. This approach was further extended using a deformable-part model (DPM), removing some of the limitations [52]. DPMs work by decomposing an object into a collection of smaller components (object parts). However, this comes at the cost of tuning several parameters by the user (e.g., number of components, and number of component types), complicated optimization procedure, and increased computational budget.

Over the years, many different approaches have been described for the task of object detection. Considerable progress in visual recognition took place after 2012 when a Convolutional Neural Network (CNN), AlexNet, won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) by a large margin [5]. Shortly after that, CNNs were also used in object detection by R. Girshick et al. using the Region-Based Convolutional Neural Networks (RCNN) model [53]. The idea behind the RCNN is quite simple. Firstly, a large set of candidate object boxes are extracted from the image using specialized algorithms (e.g., selective search algorithm [54]). Then each candidate box is rescaled to fixed image size, for which features are extracted using a CNN (backbone) and classified using a linear SVM into one of the object classes (or background). This approach improved the mean average precision metric (mAP) over previous DPM models from 33.7% to 58.5%. Note that the DPM model also used SVM for optimization, which shows how effective are the CNNs features. The whole pipeline was further improved by using Region Proposal Network to generate region proposals (instead of a selective search algorithm) and a classification layer to classify regions of interest in one large neural network [48].

Such a two-stage architecture (region proposal computation and classification) was very influential in the object detection community. Some researchers further optimized the overall architecture to reduce the inference time. Region proposals can be sampled at the regular grid (similar to the sliding window approach) and the model is also responsible for adjusting initial proposals to match the object localization. This is the main idea behind so-called one-stage object detectors (e.g., YOLO [55], SSD [56]), which allowed them to significantly increase the algorithm speed, with a small accuracy decrease.

Those two types of architectures dominate the object detection community paradigms, which are used as the starting point for developing new algorithms.

MOST WIEDZY Downloaded from mostwiedzy.pl

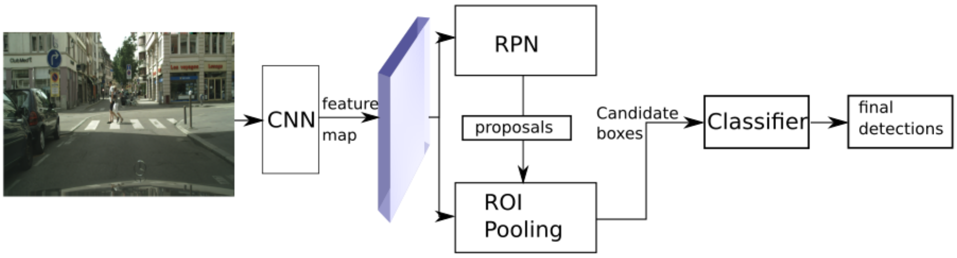


Figure 2.4: Architecture of a two-stage object detector (i.e., Faster R-CNN) consists of CNN backbone, regional proposal network (RPN), ROI pooling layer and a classifier.

Faster R-CNN is a very influential object detection neural network, commonly used by the research community and throughout this thesis. In the next subsection, the details of the Faster R-CNN are given.

2.2.2 Faster R-CNN

Faster R-CNN consists of two main modules: firstly, regions of interests are generated using Region Proposal Network (RPN). Then, in the second stage, the proposals are classified and refined. The main novelty was introducing the Regional Proposal Network (RPN), which efficiently computes a set of candidate boxes, for which the feature vector computation is shared with the classification network. The overall architecture is presented in the Fig. 2.4.

Let us define a true class label $u \in \{0, 1, ..., C\}$ where C is the number of classes being classified, and by convention, the zero label corresponds to the background class. The RPN predicts a set of candidate boxes (anchors), for which it predicts the probability that an anchor is an object and its coordinates. Anchor is defined by bounding box coordinates x_a, y_a, w_a, h_a, which denote pixel coordinates of the bounding box centre (x_a and y_a) and it width w_a and height h_a. The following parameterization is used for the coordinates of the bounding boxes [53]:

$$t_x = \frac{x - x_a}{w_a}, t_y = \frac{y - y_a}{h_a}, t_w = log(\frac{w}{w_a}), t_h = log(\frac{h}{h_a}) \tag{2.16}$$

where x, y, w, h are the ground truth coordinates. Given a set of anchor proposals the RPN is trained using the following loss [48]:

$$L(\{\hat{p_i}\}, \{\hat{t_i}\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(\hat{p_i}, p_i) + \frac{\lambda}{N_{box}} \sum_i p_i \; smooth_{L1}(\hat{t_i} - t_i) \tag{2.17}$$

where $\hat{t_i}$ are predicted parametrized bounding boxes, t_i are corresponding ground truth coordinates, $\hat{p_i}$ are the predicted probabilities of the anchor being an object, and p_i is a ground truth label with a value of 1 if the anchor corresponds to an

MOST WIEDZY Downloaded from mostwiedzy.pl

object and 0 otherwise. Index i iterates over region proposals. N_{cls} normalizes the equation – the mini-batch size, and N_{box} – the number of anchors (e.g., 2400). L_{cls} is simply the log loss over two classes (object vs. not object). The $smooth_{L1}$ loss is a differentiable version of the $L1$ loss, which the authors of the cited paper also claim works better for the task of detection (compared to the $L1$ loss):

$$smooth_{L1}(x) = \begin{cases} 0.5, & if\ |x| < 1 \\ |x| - 0.5, & otherwise \end{cases}$$

A result of this stage of computation is a list of region proposals, which are further classified and refined by the final layers in the network. Before this can be done, a feature vector for each region proposal must be computed. This is not trivial since the region proposals are of varied sizes. Region of Interest Pooling is a computational block that allows the computation of fixed-size feature vectors per region [57]. ROI Pooling splits the input feature map into k similar-sized regions and then applies Max-Pooling to every region, resulting in a feature vector of length k. Each region proposal is processed by classification and regression layers in the final stage. The first outputs a probability distribution $p = (p_0, p_1, ..., p_C)$ over all classes (per ROI). The second outputs bounding-box regression offsets for each of the C object classes, initial class-agnostic candidate boxes, are refined to match ground truth boxes more accurately. Each ROI corresponds to ground truth class u and regression offset v. The multi-task loss function combines the losses of classification and bounding box regression [57]:

$$L(p, u, \hat{v_u}, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(\hat{v_u}, v) \tag{2.18}$$

$\hat{v}_u$ are the predicted regression offsets for ground truth class u and λ is a balance term between the classification and localization loss. The classification loss is the log loss for ground truth class u:

$$L_{cls}(p, u) = -logp_u \tag{2.19}$$

The Iverson bracket indicator function $[u \geq 1]$ evaluates to 1 when $u \geq 1$ and 0 otherwise, and hence the localization loss is only computed for the non-background objects, using the $smooth_{L1}$ distance between the predicted and target regression offsets:

$$L_{loc}(\hat{v_u}, v) = \sum_{i \in x,y,w,h} smooth_{L1}(\hat{v_{ui}}, v_i) \tag{2.20}$$

The generated bounding boxes can highly overlap each other. Non-maximum suppression (NMS) algorithms are applied, as it is common to remove redundant boxes in the object detection task. If the intersection over union (IOU) between two boxes is higher than some threshold (for example, 0.7), the one with a lower confidence score is discarded. The same applies to the regions generated by the

MOST WIEDZY Downloaded from mostwiedzy.pl

RPN.

In Faster R-CNN, the whole pipeline is trained end-to-end within one large neural architecture. Currently, Faster R-CNN is commonly used. Thanks to their improvement in object detection, they have become widely used in the industry. One of the areas where object detection is important is in the automotive industry. The next subsection describes algorithms models for the task of semantic segmentation.

2.3 Semantic Segmentation

Semantic segmentation is a pixel-wise classification problem, intending to assign a predicted category $c \in \{1, ..., C\}$ to each pixel. The list is usually extended by adding a background class. Popular use cases of semantic segmentation include:

- medical applications. Finding abnormal regions in CT, MRI, etc. scans (including 3D scans).
- Robotics. Segmenting images in the camera helps the robot to navigate the environment.
- separating the foreground from the background (e.g. blurring the background when using an online communicator).

An important aspect of semantic segmentation is that it does not differentiate between different object instances (e.g., all vehicles in the image are given the same label). This is a task for instance-aware semantic segmentation, which also labels each pixel with the object identifier.

Semantic segmentation has a long history in computer vision, and similar to other tasks in computer vision, CNN-based methods are currently dominating the field. Before the introduction of deep nets, traditional methods included threshold segmentation [58], clustering [59], and graph theory [60], with some of those methods being unsupervised. However, accuracy is key for many applications (medicine, robotics), and so supervised methods are the dominant approaches. In the case of semantic segmentation, the labeling cost is usually higher when compared to the object detection task. To reduce the labeling costs, initially trained models (using available data) are usually used to provide initial segmentation on new data, and then human labelers refine the segmentation masks. The cost of labeling is extremely high in the case of 3D medical scans.

For conciseness, in this section, the focus is on a CNN-based semantic segmentation, which will be further used in section 5. The first successful application of neural networks to the task of semantic segmentation was the Fully Convolutional Network (FCN) [61]. It introduced many important ideas, which were the basis for further generations of semantic segmentation models. The main idea here was to create a fully convolutional model by replacing fully connected layers with convolutions of appropriate size. A learned upsampling operator is used to obtain a dense

MOST WIEDZY Downloaded from mostwiedzy.pl

classification layer, which makes it possible for the output to match the input image resolution. Those changes allow the input to be of arbitrary size.

The FCN architecture consists of the encoder part (squeezing the input information into a condensed feature map) and the decoder part (computing a full-resolution semantic segmentation map from the feature map). For the encoder, a standard image classification architecture can be used, which allows the semantic segmentation models to be pretrained using the image classification task (for which a lot of annotated data exists). For the decoder, the FCN paper uses a *transposed convolution*, which is also known as an upsampled convolution and allows to produce an output that is larger than the input.

However, because the spatial resolution of the bottleneck is downsampled by a factor of 32, the decoder struggles to produce accurate segmentation. To address this issue, a skip connection from earlier layers (before downsampling) was added as additional input to the decoder, which improved the segmentation quality significantly. To train the model a pixel-wise cross-entropy loss was used [62]:

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}(y_i = c)log(p(\hat{y}_i = c)) \tag{2.21}$$

where $(y_i = c) \in \{0, 1\}$ indicates whether class c is the correct class for pixel i and $p(\hat{y}_i = c)$ is a predicted probability for class c at pixel i, and N is the number of pixels. For each pixel model returns a logit vector $z_i \in R^C$. Further a $softmax$ function is applied $p_i = softmax(z_i)$, which returns a list of predicted class probabilities for a given pixel. The class with the highest probability is used as the predicted class with an associated probability score.

As the FCN was very successful, its architecture was a basis for many other semantic segmentation models, with many of the improvements focusing on the decoder architecture. In the U-Net architecture [63], the decoder contains multiple steps of upsampling operations, which are symmetrical to the encoder. Such a u-shaped architecture was very successful, especially in medical imaging applications. Over the years, many different architectures have been developed, and currently, the DeepLab models family [64, 65] is dominating and will be used in section 5. The DeepLab models introduced several specific improvements such as:

- Spatial pyramid pooling was used to deal with different input image sizes,
- Atrous convolution incorporated a larger context without increasing the number of parameters,
- Depth-wise separable convolution was used to increase computational efficiency.

While there was a large progress in the accuracy of object detection and semantic segmentation models, the progress would not have been possible without the

MOST WIEDZY Downloaded from mostwiedzy.pl

availability of large-scale datasets, and in the next subsection, the datasets used throughout this thesis are presented.

2.4 Datasets

Creating large-scale datasets with annotated images is time-consuming and costly [66]. At the same time, large-scale datasets are essential for training modern visual recognition models. In this thesis, the datasets related to autonomous driving are mostly used because of data availability and because robustness is essential for autonomous driving.

While early datasets usually focused on some narrow tasks (e.g., pedestrian detection), modern datasets have a larger focus (objects detection, semantic segmentation) and focus on gathering varied data (i.e., different weather conditions). Furthermore, while early datasets were created mostly by universities, as the datasets became larger (and thus more expensive), nowadays, more and more datasets are owned by companies that sell licenses to them.

An early example of large scale-dataset was the Caltech pedestrian dataset, which consists of approximately 10 hours of video recorded in a vehicle driving in an urban environment [67], mostly in good weather conditions. There are 350,000 pedestrian boxes annotated, which gives a ratio of 1.4 pedestrians per frame. While the Caltech dataset focused on pedestrian detection, other datasets such as KITTI [68] and Cityscapes [69] focused on general object detection. Those datasets contain bounding box annotations for each object belonging to 8 different classes: person, rider, car, truck, bus, train, motorcycle, and bicycle. Both datasets also contain dense pixel annotations, which can be used for semantic segmentation tasks. The KITTI dataset also includes data from LiDAR sensors and is more often used for multi-sensor object detection. Hence, the Cityscapes dataset is used in this thesis, which is very popular for visual-based object detection.

The Cityscapes dataset was recorded in 27 cities from 3 countries, with dense pedestrian scenes (more than six pedestrians per frame). Cityscapes is an important and challenging benchmark, with heavy occlusion being the challenge and the fact that the test dataset includes cities that were not part of the training set. However, the whole dataset was again recorded in mostly good illumination and weather conditions, limiting generalization to the real world. In section 3, a CityPersons [70] dataset was used, providing refined bounding box annotations for pedestrians and cyclists on Cityscapes images.

As such, further benchmarks were developed and EuroCity Persons [71] improved the diversity of pedestrian detection datasets. It was recorded in 31 cities in 12 European countries. Data were collected during all seasons in changing weather conditions (rain, fog). In total, there are around 238,200 person instances annotated in over 47,300 images. A subset of 7,000 images recorded during the night-time was a novelty at the time of the release of the dataset.

MOST WIEDZY Downloaded from mostwiedzy.pl

Dataset	Number of frames	Classes	Pedestrians per image	Conditions
Caltech [67]	250,000	pedestrians	1.4	dry, day-time
Cityscapes [69]	5,000	8 classes	7.0	dry, day-time
ECP [71]	47,337	2 classes (pedestrians, riders)	4.6	dry, wet, day and night-time
Nightowls [72]	279,000	3 classes (pedestrians, cyclists, motorcyclists)	0.2	dry, wet, night-time
BDD [73]	100,000	10 classes	1.2	Dry, wet (including snow), fog, day and night-time

Table 2.1: Comparison of various object detection datasets for autonomous driving

On the other hand, the NightOwls dataset focused on night-time pedestrian detection, providing 40 annotated video sequences [72]. In comparison to day-time images, it is a much more challenging task due to the illumination variation, light reflections, blur artifacts, and changes in contrast. In total, there are 279,000 annotated frames from 3 countries. Night-time pedestrian detection is very important for robust vision applications. However, the author showed that pedestrian detectors do not perform well at night, even when trained on night-time data.

One of the biggest datasets is Berkeley Deep Drive (BDD) [73] which contains 100,000 short driving videos. Within each video, one frame has object annotations. Altogether there are over 1 million cars and 129,000 pedestrians annotated. It also includes images recorded with larger weather diversity than previous benchmarks.

Table 2.1 shows a summary of selected autonomous driving datasets. It is by no means a comprehensive list of datasets for autonomous driving applications. Other recent examples include the Waymo Open Dataset [74] and nuScenes [75]. Both datasets were developed by automotive companies and focus on large-scale data, diversity, and multi-task learning (such as lane detection).

Also, some of the datasets from Table 2.1 contain semantic segmentation labels (Cityscapes, BDD), which will be used in chapter 5. As semantic segmentation labels are even more costly to obtain, using synthetic labels was also considered in this thesis. Data collected from the simulation, i.e., a modern computer game (Grand Theft Auto), was used for that purpose. Using data from a simulation and increasing its realism is another important research direction to reduce annotations costs.

While this thesis focuses mostly on autonomous driving datasets, the methods developed here can be used for general visual recognition. Some experiments

MOST WIEDZY Downloaded from mostwiedzy.pl

are also performed on the COCO dataset [76], which is a large-scale (328K images) dataset for general purpose object recognition (91 different classes, including animals, household items, food, etc.).

It is important to realize that no matter how big the datasets are, they will never cover all combinations of situations in the real world (for example, snow at night). As such, testing on the above datasets may not be sufficient for predicting the real-world performance of visual recognition systems. In the next section, techniques for approximating the model's generalization properties are reviewed.

2.5 Model Robustness

The generalization to new environments and situations is an important trait of the human visual system. Humans can easily recognize objects when viewed in different contexts or when the input signal is noisy (to some extent). However, creating computer vision algorithms with similar capabilities is a great challenge. Some studies compared the generalization properties of humans and algorithms, i.e., sensitivity to the input noise, concluding that the current algorithms are far behind human perception [30, 31].

Machine learning models perform well when the test-time distribution of data matches or is close to the training-time distribution, otherwise the model can make unexpected predictions under the distributional shift. A well-known example of "failing" visual systems are so-called adversarial examples [77]. By having access to the machine learning model, it is possible to compute the smallest distortion in the input image that will cause a change in the outcome of the model. However, those changes are usually invisible to the human eye, and they question whether machine learning models can be trusted. Adversarial examples artificially change the input distribution of the images, resulting in undesired outcomes of the classifier.

What is more, some adversarial attacks seem to transfer to different architectures [78]. Defending against adversarial attacks is an important research area in the community that could help build trust in machine learning systems. However, creating adversarial examples requires having access to the model. Hence, an interesting question arises: can one easily fool a machine learning algorithm without accessing the model?

An important work in this area was done by Hendrycks et. al in [79], where a Common Corruptions benchmark was created. It contains 15 types of distortions, which are meant to be used only during testing to measure model robustness. They found that current machine learning models are very vulnerable to even tiny noises, for example, adding salt & pepper noise can easily fool the detector. Fig. 2.5 presents the selected type of noises applied to an image from Cityscapes dataset, applied at different severity intensities. As one can see, even though the image's semantics have not changed, such minor distortions can easily fool modern visual perception systems.

MOST WIEDZY Downloaded from mostwiedzy.pl

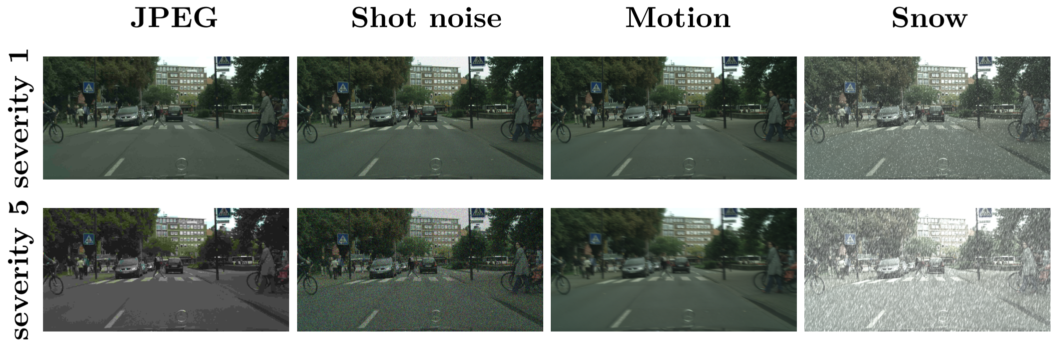


Figure 2.5: Examples of different corruption types from Common Corruptions benchmark with different severity. Note that at the lowest severity distortions are barely visible whereas at the highest severity they are clearly visible, however semantics of the images are not changed.

Furthermore, some works show that models are also vulnerable to small translations and rotations of the input image [47, 80], small changes in distribution [12] and cannot recognize known objects from an unusual perspective [81]. These are just a few examples of neural networks' vulnerability to cases of slight changes in the input image. One of the hypotheses is that current neural networks tend to learn superficial features that work well within the dataset but fail to generalize outside of the training distribution [13, 17]. As a result, decisions made by the classifier are sometimes based on just a few pixels. Furthermore, they tend to heavily exploit the background context [82], which explains why objects in a new context are often erroneously detected.

It was also shown that the decisions made by the classifiers are mainly based on the object texture (low-level feature), while the object shape (high-level feature) is mostly ignored [14]. Therefore, the authors applied neural style transfer data augmentation to focus on high-level features, generalizing them better. Neural style transfer is a technique that applies the style of one image to the content of another image, and as a result, an image is obtained with the same content but in a different style (texture) [83]. As a result of such training on stylized images, the neural network was more texture invariant and more robust to distinct types of noises.

Data augmentation traditionally plays a key role in improving the robustness of the machine learning models. A quite simple method, CutOut, works by randomly erasing patches of the input image [84]. Such augmentation forces the model to look at the whole image instead of focusing on some small patches, which might be highly predictive within the dataset. Adding diverse types of noises, e.g., Gaussian noise and blur, also proved to be an efficient data augmentation method [85]. However, adding noise to the whole image might decrease the clean data performance (while improving the robustness). It was also proposed to apply Gaussian noise only to the input image patches [86]. As mentioned earlier, style-transfer data augmentation is also very efficient at increasing models' robustness [14, 9].

One of the methods that also consequently works well is large-scale pretrain-

MOST WIEDZY Downloaded from mostwiedzy.pl

ing. For example, The Facebook company pre-trained their classification model on 3 billion Instagram images where the goal of the network was to predict social media hashtags [87]. Such (very costly) training showed state-of-the-art performance for transfer learning. Similarly, pretraining a model on many datasets containing people annotations in pedestrian detection yielded satisfactory performance [88]. Large-scale self-supervised pretraining also proved to work well [89]. In general, presenting a model with a lot of data (also during the pretraining stage) provides reliable results. It was also shown that using bigger models brings improvement in the model's robustness [79]. This is an essential context because, throughout this thesis, the goal is to find robust models which are computationally efficient.

An important challenge is evaluating the robustness of machine learning models. The general idea is to use various out-of-distribution (o.o.d.) tests [17, 90], which have recently gained much attention. The simplest scenario is testing models in a cross-dataset setting. Unfortunately, this is often not sufficient, as all collected datasets contain some biases, which the learning algorithm may utilize. For example, the datasets mentioned in the previous section were collected in developed countries and urban areas. As such, cross-dataset evaluation is an important metric but might not be good enough for real-world deployment.

Another type of evaluation includes testing the model's sensitivity to distortions in the input image. It includes creating synthetic distortions, which are then used during testing to evaluate model robustness. For example, the aforementioned Common Corruptions benchmark consists of 15 varied corruption types, grouped into 4 categories: blur (defocus blur, frosted glass blur, motion blur, zoom blur), noise (Gaussian noise, shot noise, impulse noise, salt-and-pepper noise), digital (elastic transformations, pixelation, JPEG lossy compression) and weather distortions (snow, fog, brightness, contrast). Adversarial examples are another example of synthetic distortions. An attacker intentionally designed those images to fool the machine learning model with the smallest changes in the input data [77] and can be seen as an example of worst-case noise. However, sensitivity to corruptions might be more important for autonomous driving and is used throughout this thesis [9].

While synthetic distortions are a particularly useful tool, it is unclear whether robustness to synthetic distortions transfers well to the distributional shifts in real data [91]. Another way of measuring robustness is testing algorithms on natural shifts: such as day-to-night transition. For example, the model is trained using "clean" data (day-time images) and evaluated under natural distributional shift (for example, night-time or foggy images) [9]. However, the problem with this approach is that it requires large and diverse datasets with diverse types of natural distortions, which are not always feasible.

Robustness is also related to the problem of algorithmic fairness. Distributional shifts often degraded performance on minority subpopulations of the data. For example, it was shown recently that computer vision algorithms work significantly better at recognizing household items from developed countries because the

MOST WIEDZY Downloaded from mostwiedzy.pl

training data were mostly collected in those countries [92]. There are specialized algorithms that address improving accuracy on minority subgroups, but this is beyond the scope of this thesis [93, 94]. However, in section 4, standard data balancing methods are used to improve the accuracy in models with reduced capacity.

In summary, distributional shifts are a great challenge for machine learning algorithms deployed to the real world. Ideally, large-scale test data with rare events such as heavy snow and car accidents should be collected to evaluate model robustness to conditions unseen during training. However, as this is often unfeasible, it is possible to approximate model robustness using synthetic and natural distortions or cross-dataset evaluation. None of these methods is sufficient on its own; however, combining them is the best thing that can currently be achieved, and they will be used throughout this thesis.

2.6 Evaluation Metrics

In this section, the evaluation metrics used in object detection and semantic segmentation are presented. Section 2.6.2 focuses on quantifying model uncertainty.

2.6.1 Model Accuracy

Object detection. The first step in the object detection pipeline is filtering out low-confidence predictions. The threshold value has to be determined manually; typically, a threshold value of 0.5 is used. To decide whether the given predictions are correct (true positive - **TP**) or not (false positive - **FP**) an intersection-over-union (IoU) between the predicted bounding box p and the corresponding ground truth g is computed:

$$IoU(u,p) = \frac{Area(g) \cap Area(p)}{Area(g) \cup Area(p)} \tag{2.22}$$

If the IoU is bigger than some fixed threshold (for example, 0.5 as in the popular Pascal VOC challenge [95]), the prediction is correct. Several detections may be matched with ground truth. In that case, only the most confident detection is matched, and the other detections are considered false positives (**FP**). Now, it is possible to compute precision and recall metrics[96]:

$$Recall = \frac{TP}{TP + FN} \tag{2.23}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.24}$$

However, such point estimates do not give a full picture of detector reliability. Creating precision-recall curves, where the precision (y-axis) and the recall (x-axis)

MOST WIEDZY Downloaded from mostwiedzy.pl

are plotted for different confidence thresholds, is more informative. The area under the precision-recall curve is called the *average precision* metric (AP). Similarly, the IoU threshold can also be varied, for example, $\mathrm{AP}_{0.75}$ accepts only predictions with an IoU higher than 0.75 and is more restrictive than the 0.5 value used in the Pascal VOC challenge. *Mean average precision* (mAP) is simply average precision for all of the classes and is the most commonly used metric in the object detection community

An IoU threshold of 0.5 is widely used in pedestrian detection because of big deformations and articulations in pedestrians' bodies. Furthermore, in pedestrian detection, special attention is put into reducing the number of false negatives to increase pedestrian safety. As a result, the *miss-rate* (mr) and *false positives per image* (fppi) are popular metrics:

$$mr(c) = \frac{fn(c)}{tp(c) + fp(c)} \tag{2.25}$$

$$fppi(c) = \frac{fp(c)}{N_{img}} \tag{2.26}$$

Threshold c is used for adjusting the ratio between false positives, false negatives, and true positives, and $\mathrm{N_{img}}$ is the number of images. Both of the above metrics can be combined in the popular *log-average miss rate (LAMR)* metric, which is computed by averaging at nine fppi rates spaced equally in the log-space in the range 10^{-2} to 10^{0}, as it is done in the community [67]:

$$LAMR(c) = exp(\frac{1}{9}\sum_{f} log(mr(\underset{fppi(c)<=f}{\arg\max}\ fppi(c))) \tag{2.27}$$

Semantic Segmentation. Problem formulation makes the metrics used in semantic segmentation very easy to define. The most basic metric is *pixel accuracy*, which simply reports the number of correctly classified pixels:

$$pix_{acc} = \frac{TP}{N_{pixels}} \tag{2.28}$$

The main problem with this metric is that it is biased toward common classes; it is possible to obtain very high accuracy but to still not detect rare classes at all. To compute accuracy for a given class a IoU metric (Jaccard index) or F1-score (Dice score) can be computed as follows:

$$IoU = \frac{TP}{TP + FP + FN} \tag{2.29}$$

$$Dice = \frac{2TP}{2TP + FP + FN} \tag{2.30}$$

The final result is then an average of all classes, for example, mean intersection over union ($mIoU$), which computes per-class IoU values and then reports their average.

MOST WIEDZY Downloaded from mostwiedzy.pl

mIoU is commonly used metric in semantic segmentation. Compared to the Dice score, the IoU metric more heavily penalizes instances of bad classification. Also, some specialized metrics exist for semantic segmentation, such as Boundary IoU, which focuses on the quality of the boundary [97].

2.6.2 Quantifying uncertainty

Providing reliable uncertainty estimates is an essential element of safe autonomous systems [8] and could also be useful in other high-stakes applications, such as machine learning-aided medical diagnosis. However, current machine learning models are often overconfident in their predictions [98], and the effect is more apparent in the out-of-distribution setting [99].

Expected Calibration Error (ECE) is one of the metrics used to compute the uncertainty calibration of the model. The intuition behind model calibration is that when a well-calibrated model predicts a bounding box (or pixel class) with 95% confidence, it should be accurate in 95% of the cases. ECE can be easily computed by partitioning predictions into M bins based on their confidences:

$$ECE(c) = \sum_{m=1}^{M} \frac{|B_m|}{n} |acc(B_m) - conf(B_m)| \tag{2.31}$$

where B_m is the set of prediction indices for which the confidence falls into the mth bin. A lower ECE score means better model calibration.

To sum up, the background knowledge in machine learning and computer vision was introduced. Further, modern CNN-based models for object detection and semantic segmentation were presented. It was shown that the generalization of machine learning algorithms to the real world is an open problem. In the next chapter, the model's robustness will be evaluated for pedestrian detection, and the data augmentation methods will be used to improve model performance in terms of accuracy and quantifying uncertainty.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 3

Towards Robust Pedestrian Detection with Data Augmentation

The literature review in the previous chapter shows that modern neural networks for visual recognition achieve impressive performance, but only when the test-time conditions are like those encountered during training. It is, however, a too strong assumption for the models deployed to the real world, for example, autonomous driving systems. In this chapter, the safety of the most vulnerable road user - pedestrians, is considered. Furthermore, this chapter explores different ways of evaluating models' robustness. Further, the robustness and uncertainty calibration of several data augmentation methods is evaluated, and based on the insights from experiments, a new data augmentation scheme is proposed. Work described in this chapter is based on the article [100].

3.1 Introduction

Even though CNN-based models surpassed human performance on some of the benchmarks [101], the application of deep learning methods in safety-critical applications like medicine or autonomous vehicles has been limited [8]. This is because CNNs often fail to generalize outside of the training data distribution. Vulnerability to tiny changes in the input might be explained by the fact that neural networks tend to exploit non-robust, high-frequency patterns in the training dataset, which causes them to fail under the distributional shift [77, 46, 47, 80, 102, 12].

A popular approach to improve model robustness is employing data augmentation techniques, i.e., using style-transfer data augmentation [14, 103]. Meanwhile, the use of only stylized representation may hurt performance on clean (original) data; hence a popular strategy is to use both clean and stylized samples during training [9, 104]. Removing part of the image during training has also improved

MOST WIEDZY Downloaded from mostwiedzy.pl

model accuracy [84]. Instead of occluding a portion of an image, an approach called CutMix replaces a portion with a patch from a different image [105]. AugMix approach uses different image augmentations and interpolates between them to obtain training samples [106]. It is also possible to learn augmentation policy; however, such a process tends to be very costly [107]. Other approaches include using self-supervision [89], adversarial training [108], or using large-scale pretraining [109].

Pedestrian detection is an essential topic in the context of autonomous driving. It is an important component of advanced driver-assistance systems (ADAS) and is crucial in reaching autonomous vehicles. This topic traditionally borrows a lot from standard object detection models; current state-of-the-art models like Faster-R-CNN [48] and Mask R-CNN [110] are used for that purpose. In addition, there exist specialized models which modify the loss function to handle occlusions [111], run multi-step prediction for improved localization [112], simultaneously predict the full and visible boxes of pedestrians [113], or utilize low variance in an aspect ratio of visible pedestrians [114]. However, as it was shown, a general-purpose Faster R-CNN provides very competitive results [70, 88]. Another line of research is dedicated to using thermal images for pedestrian detection at night [115, 116, 117].

Providing reliable uncertainty estimates is essential for safe autonomous systems [8]. The previous finding of the poor model calibration was confirmed in the context of object detection from LiDAR data [118]. In this chapter, uncertainty calibration is evaluated the context of pedestrian detection under the distributional shift. In this chapter, the following experiments are described:

- the impact of the distributional shift on the accuracy of pedestrian detection models is analyzed, i.e., detection models are evaluated in cross-dataset setting, by adding different types of image distortions, and also while testing on night-time images,
- popular data augmentation methods are evaluated in terms of model robustness, and a new relatively simple scheme for data augmentation is proposed,
- uncertainty calibration of existing models is evaluated experimentally.

3.2 Pedestrian detection model

A pedestrian (object) detection task is to associate a list of bounding box coordinates with the predicted class and its score with an image. Faster R-CNN is a standard algorithm in generic object detection, which is commonly used in pedestrian detection [88, 70]. Therefore, it is a strong baseline utilized in this chapter. Faster R-CNN was described in section 2.2.2. For pedestrian detection also specialized architectures exist, and a Center-Scale-Prediction (CSP) [114] is a recent model that achieves state-of-the-art results. It simplifies the object detection pipeline by simply predicting the center of the objects and their scale. Such an anchor-free framework does not require defining anchors hyperparameters, i.e., the sizes of anchors or the

MOST WIEDZY Downloaded from mostwiedzy.pl

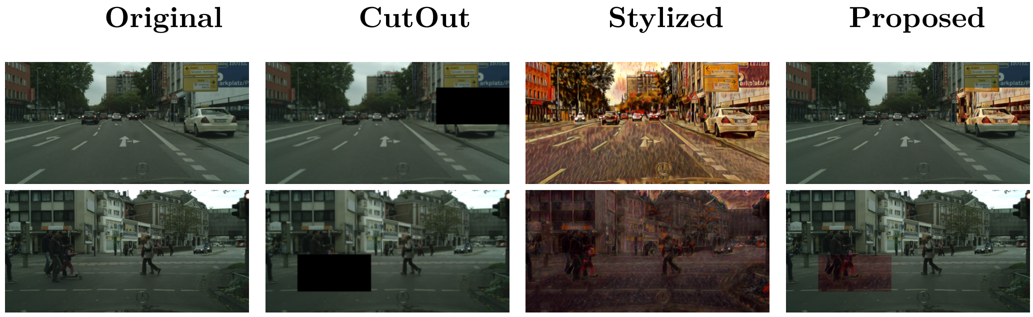


Figure 3.1: Different augmentation strategies. The second column shows random region removal with CutOut, and the third column shows a stylized version of the original image. The last column shows the proposed augmentation that combines CutOut with style augmentation.

number of scales (as in Faster R-CNN), and works very well for pedestrian detection. Both architectures are used in this chapter.

Style-transfer is a popular technique that allows transferring style (texture) from one image into another image. While such image synthesis is not perfect, many studies have shown that using style-transfer data augmentation can improve the robustness of the model [14, 104]. A problem with style-transfer data augmentation is that whereas it increases the robustness of models, it can decrease the accuracy of clean data since the stylized image differ quite significantly from real images. As such, a popular approach is to train a model using a 1:1 ratio of stylized and original images [9]. Here, a popular approach in the literature is followed, and as a source of style, images from Kaggle's Painter by Numbers dataset [119] are used.

Proposed augmentation. There is a growing literature of research that, during training, augments only patches of the image. CutOut, for example, removes random patches from the image that shows positive for model accuracy [84]. In CutMix, on the other hand, random patches are cut and pasted among training images [105]. Inspired by data augmentation which works on regions of the image [86, 84] here, we propose to apply style data augmentation only to random patches of the input image. Our method works by adding a patch of the stylized image to the original image to the same location. The center of the patch is sampled to be within the image and the method allows for varying the patch size. Details are presented in the Alg. 1 diagram. Fig. 3.1 shows proposed data augmentation in comparison to other methods.

Gaussian augmentation. Using Gaussian augmentation also proved to be successful in increasing model robustness; therefore it is also used for our experiments. Two variants (parametrized by σ_{max}) are evaluated:

- Gaussian augmentation. Firstly for each pixel sample σ from uniform distribution - $\sigma \sim U(0, \sigma_{max}]$. Add noise to each pixel sampled from $N[0, \sigma]$.
- Patch Gaussian augmentation [86]. Same as above, but the Gaussian noise is added only to the random patch of the image.

MOST WIEDZY Downloaded from mostwiedzy.pl

Algorithm 1 Proposed data augmentation

```
Input: Input image $I$, Stylized image $S$,
    Image width $I_W$, Image height $I_H$,
    Patch width $P_W$, Patch height $P_H$
Output: Augmented Image $I_{out}$
    %Compute patch localization
    $x1 \leftarrow random.normal(0, I_W - P_W - 1)$
    $y1 \leftarrow random.normal(0, I_H - P_H - 1)$
    %Compute masks
    $mask[H][W] \leftarrow \{0\}$
    $mask[y1 : y1 + P_H][x1 : x1 + P_W] \leftarrow \{1\}$
    $mask\_inverse[H][W] \leftarrow \{1\}$
    $mask\_inverse \leftarrow mask\_inverse - mask$
    %Compute final image
    $I_{out} \leftarrow mask\_inverse * I + mask * S$
    return $I_{out}$
```

3.3 Experiments

Implementation details. MMdetection library [120] is used for the Faster R-CNN model, with ResNet-50 backbone. All models are pre-trained on ImageNet [7]. Stochastic Gradient Descent with an initial learning rate of 0.002 and a momentum of 0.9 is used. The training lasts for 40 epochs, and the learning rate drops by a factor of 10 after 25 epochs. All models are trained on the Cityscapes dataset. Data from 3 cities (Darmstadt, Mönchengladbach, and Ulm) from the training set are moved into trainval set similar as in the literature [121]. The model with the best accuracy on the trainval set is used for testing. Since no ground truth data is publicly available for all the datasets, results are reported on their validation sets. All models used the same training settings. Standard random vertical flipping and resizing of the image are used for data augmentation. For the CSP architecture, the code published by the paper authors is used [114]. A training protocol from the original paper is followed, i.e. model is trained for 37.5K iterations and then used for the validation. The only difference is that a single GPU was employed for training (instead of two). During the evaluation, each training is repeated 5 times, and the mean accuracy is the final score. For evaluation, the so-called reasonable setup [70] is used. It means that only pedestrians whose height is bigger than 50 pixels and the occlusion level are smaller than 0.35 are used for training and evaluation.

In the experiments, firstly, the effect of patch size on the model accuracy with the proposed data augmentation scheme was analyzed. Then, robustness and uncertainty calibration of several data augmentation methods was evaluated using different robustness tests.

MOST WIEDZY Downloaded from mostwiedzy.pl

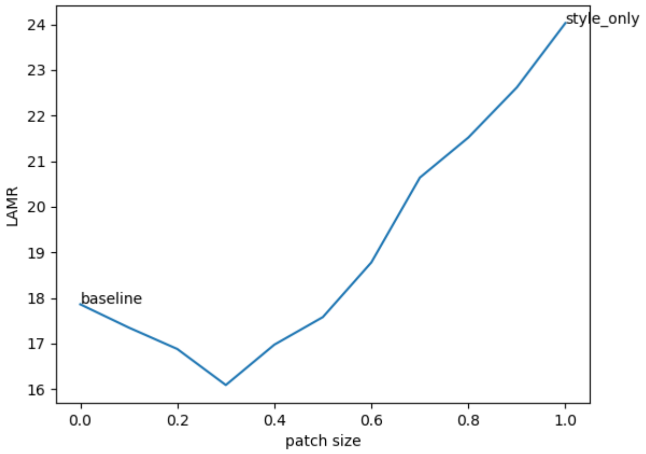


Figure 3.2: Log-average miss rate for pedestrian detection accuracy (**lower is better**) on CityPersons dataset using proposed data augmentation as a function of the patch size. Note that the most left data point corresponds to the baseline trained on original data, while the most right data point is a model trained using only stylized images.

3.3.1 Patch-size selection

First, hyperparameter search for the optimal size of the stylized patch is run. A too small patch size might reduce the positive effects of using style-transfer for model robustness, whereas a too big size of the patch might reduce the clean accuracy. In the experiment the stylized patch is of size kW x kH pixels, where $k\epsilon[0, 1]$. Note that when $k = 0$, it means that style transfer augmentation is not used at all, whereas when $k = 1$, only stylized-images are used.

Figure 3.2 plots the model accuracy as a function of patch size on the CityPersons dataset. It can be noticed that the model accuracy firstly increases with the size of the patch. However, after the size of the patch is bigger than 0.3 of the image size, then accuracy decreases. This is expected as the model is biased more towards stylized images, and as a result, accuracy on the clean data decreases. All of the remaining experiments are conducted with the selected patch size ($k = 0.3$).

3.3.2 Evaluation under distributional shift

In this section, different data augmentation methods are evaluated with regard to the model robustness. In particular, the following models are evaluated:

- *Baseline* model corresponds to the model trained with standard data augmentation,
- *Sin* [14] is a model trained on both clean data and the stylized version using 1:1 ratio,
- *StyleOnly* model is trained using only stylized data,

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 3.1: Accuracy comparison of Faster R-CNN models trained with different augmentation strategies on clean data (first column) and related to specific corruption types from the Common Corruptions benchmark (the remaining columns). LAMR is reported (lower is better). For models that used Gaussian augmentation, values in the noise column are marked in grey color because the tested corruption type was a part of the training.

Name	Clean	Noise	Blur	Weather	Digital
Baseline	17.86	94.72	69.61	54.05	51.78
CutOut	16.91	91.04	66.72	49.97	50.61
StyleOnly	24.03	78.02	63.88	49.3	44.16
Sin	17.83	**76.14**	**59.63**	44.43	42.36
Ours	**16.09**	**76.17**	61.46	**42.74**	43.12
Gaussian_0.1	17.58	45.73	63.97	48.85	39.56
Gaussian_0.5	21.19	39.33	60.17	49.52	**38.46**
PatchGaussian_0.1	16.25	52.9	66.71	50.28	46.00
PatchGaussian_0.5	16.41	45.71	63.72	48.91	41.65
Combined augmentations					
Ours + PatchGaussian_0.5	**16.28**	47.55	61.55	**42.73**	40.5
Sin + PatchGaussian_0.5	18.11	43.27	**59.64**	44.86	**37.27**

- *Ours* corresponds to the proposed augmentation model,
- *CutOut* [84] model with the same patch size as *Ours* model. It serves as another baseline to the proposed data augmentation and as a sanity check to make sure that similar gains cannot be obtained by simply removing patches of the image,
- *Gaussian* data augmentation with $\sigma_{max}\epsilon\{0.1, 0.5\}$,
- PatchGaussian data augmentation with $\sigma_{max}\epsilon\{0.1, 0.5\}$. The same patch size is used for the *Ours* and *CutOut* model.

Table 3.1 shows the accuracy of the trained models on the original CityPersons dataset and as well on the Common Corruptions benchmark grouped by distortion category. First, the results emphasized the importance of robustness testing. While the *Baseline* provides reasonable accuracy on the clean data, it constantly has worse accuracy on all corruption types by a large margin. Furthermore, different data augmentation provides the best accuracy on different corruption types. *Ours* data augmentation best performs on clean data, noise corruption (together with *Sin* model), and weather-related corruption types. The *Sin* model performs the best also on blur corruptions, whereas Gaussian augmentation helped the most with digital noise. The findings of other authors [86] are confirmed, and it was shown that while Gaussian augmentation improves robustness, it may hurt performance on clean data (21.19% LAMR when $\sigma = 0.5$), whereas using only patches of Gaussian noise provides a balance between clean accuracy and robustness. Finally, it can be observed that combining Style-Transfer using *Ours* or *Sin* models with *PatchGaussian* provides the best accuracy across all corruption types. However, many of the corruptions still drastically degrade the performance, so there is a large room for

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 3.2: LAMR for each corruption type of Faster R-CNN models.

	Noise			Blur				Weather				Digital			
Model	shot	impulse	gauss	defocus	glass	motion	zoom	snow	fog	frost	bright	contrast	elastic	pixel	jpeg
Baseline	92.69	97.06	94.42	55.25	67.32	60.68	95.22	89.3	30.24	66.32	30.32	47.64	**14.47**	72.75	72.26
Sin	**72.25**	**79.2**	76.99	**43.88**	**54.27**	**48.57**	**91.83**	76.1	28.35	49.87	23.39	43.77	20.93	**46.66**	**56.39**
Ours	72.4	79.39	**76.73**	44.85	56.89	52.31	92.54	**75.52**	**25.97**	**49.39**	**20.06**	**39.12**	20.84	54.17	58.37

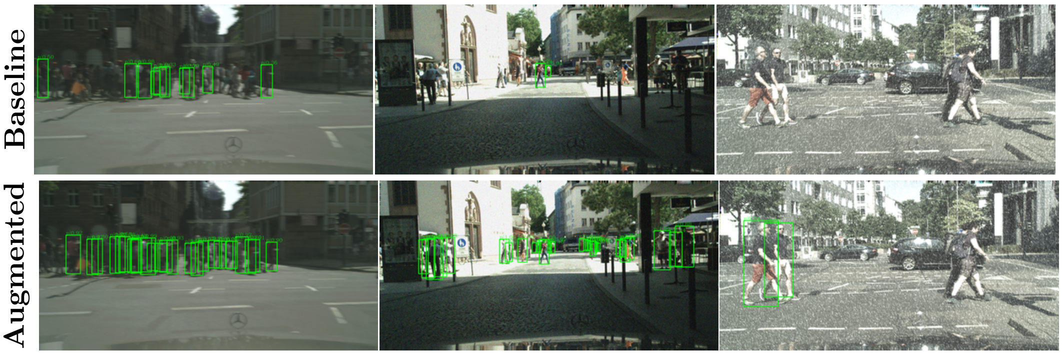


Figure 3.3: Detection samples for baseline and augmented (*Ours + PatchGaussian_0.5*) models for different corruption types. The first column - motion blur (severity intensity of 4), the second column - Gaussian noise (severity intensity of 2), third column - artificial snow with a severity intensity of 2. Note that the distortion for Gaussian noise is almost imperceptible, yet it greatly reduces accuracy of the model. Augmented model is more robust, however in the last column pedestrians on the right are missed by both models.

improvement in terms of increasing model robustness.

Also, it is interesting to directly compare *Ours* and *Sin* data augmentations as they are competitive approaches. For that purpose, those two approaches are directly compared for each corruption type. Table 3.2 shows that the proposed model provides the most significant gains for fog, brightness, and contrast deformations. This is very interesting in light of findings by Yin et al. [46] where they perform Fourier spectral analysis of different distortion types and find that the aforementioned distortions are concentrated in low-frequencies components of images. This means that the proposed data augmentation might be particularly useful for low-frequency distortions types.

Fig. 3.3 shows example detections for the *Baseline* model and a model trained with proposed augmentation and Patch Gaussian. In general, data augmentations significantly improve model accuracy, however, there are still many situations when the model lacks robustness. Some types of distortions (especially noise), drastically change the output of the detection model even at the low distortion severity, when the input image is only slightly changed. Even though our best-trained models are more robust than the *Baseline* model, the problem is still far from being solved. In previous experiments, synthetically generated distortion types were used for the evaluation. However, it is very important to test a model on a different dataset because even if the datasets look similar, there still will be a lot of differences regarding data collection protocol (e.g., camera sensor and its placement inside a vehicle, ge-

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 3.3: Accuracy comparison of Faster R-CNN models trained with different augmentation strategies on day-time and night-time images from the ECP dataset as well on NightOwls dataset (night-time). LAMR values are reported.

Name	ECP-day	ECP-night	NightOwls
Baseline	25.05	45.5	76.23
CutOut	21.93	41.25	70.23
StyleOnly	26.83	36.2	58.22
Sin	21.22	30.69	56.51
Ours	**21.04**	**29.89**	56.1
Gaussian_0.1	23.17	32.25	56.21
Gaussian_0.5	26.11	32.22	**47.55**
PatchGaussian_0.1	23.6	34.24	64.4
PatchGaussian_0.5	22.31	32.61	58.87
Combined augmentations			
Ours + PatchGaussian_0.5	**20.35**	**26.13**	52.34
Sin + PatchGaussian_0.5	20.62	28.80	**49.08**

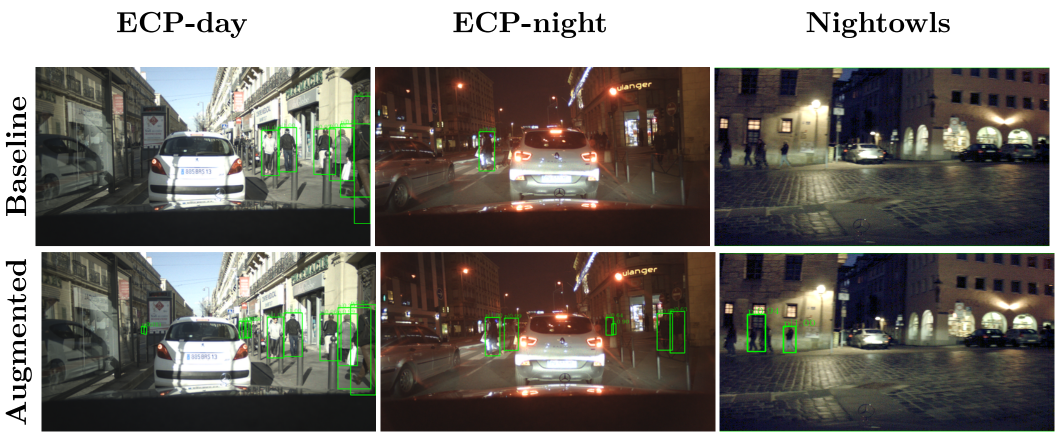


Figure 3.4: Detection samples for baseline and augmented model. The augmented model is more accurate on the night-time images, however some pedestrians are still not detected.

ographical location) which might affect final accuracy. Furthermore, synthetically generated distortions are only approximations of real-world adverse conditions that is why the models are also tested on night-time images.

Table 3.3 shows the accuracy of the models for different datasets. First, a significant drop in accuracy for all the models can be noticed, i.e., *Baseline* model LAMR increases from 17.86% to 25.05% when tested on ECP-day compared to CityPersons. It can be explained by the fact that the ECP dataset is more challenging for both but also because of the dataset shift. Accuracy further drops when models were tested on night-time images - for the *Baseline* model, the average miss-rate is almost doubled. For data augmentation based on stylization or Gaussian noise, the decrease is not that severe, and for the best performing combination (*Ours + PatchGaussian_0.5*) the average miss-rate increases only from 20.35% to 26.13%. The drop in accuracy for NightOwls dataset is bigger than for the night-time im-

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 3.4: Comparison of CSP models accuracy trained with different data augmentations, on various datasets, and on specific corruption types from the Common Corruptions benchmark applied to Cityscapes dataset (columns 2-5). For models that used Gaussian augmentation, values in the noise column are marked in grey, because the tested corruption type was part of the training. LAMR values are reported.

Name	City	Noise	Blur	Weather	Digital	ECP-day	ECP-night	NightOwls
Baseline	12.37	95.54	63.67	52.89	49.77	22.06	60.83	65.48
CutOut	**12.17**	95.06	63.39	51.52	49.83	23.31	60.52	67.13
Sin	13.85	**83.24**	**51.45**	41.23	37.63	19.48	31.25	**51.16**
Ours	12.44	84.21	54.72	**40.91**	41.56	**18.24**	**31.17**	52.71
Gaussian_0.5	17.99	34.64	55.74	59.59	**36.22**	32.11	66.32	60.72
PatchGaussian_0.5	12.47	43.64	59.31	49.95	39.99	20.06	47.20	58.42
Combined augmentations								
Ours + PatchGaussian_0.5	**12.62**	44.21	53.13	**40.44**	37.41	**18.39**	**28.99**	52.21
Sin + PatchGaussian_0.5	14.12	44.27	**50.69**	40.72	**33.73**	19.15	30.08	**49.88**

ages from the ECP dataset. On that benchmark, simple Gaussian augmentation obtained the best result - this might be, because the NightOwls dataset contains a lot of noise due to the very low light intensity. It is also worth noting the effect of the *Cutout* data augmentation on the accuracy of the clean data, however, it only slightly affected the robustness, especially when compared to the *PatchGaussian* and style data augmentations. Finally, the proposed data augmentation provides the same or better accuracy compared to *Sin* across all benchmarks. Fig. 3.4 shows example detections. Again model accuracy is improved, but the augmented model still lacks robustness for some situations (last column).

3.3.3 Center Scale Prediction

Table 3.4 shows obtained results on all of the benchmarks for the CSP architecture. Some interesting observations can be made. Firstly, in most cases, the new architecture provides better accuracy than the Faster R-CNN, i.e., LAMR decreases for the *Baseline* model from 17.86% to 12.37% on the Cityscapes dataset and 25.05% to 22.06% on the ECP day-time images. Surprisingly the new model is worse than the Faster R-CNN when testing the *Baseline* model on ECP night-image images. When stylized data augmentation is used, the new model is better or on par with Faster R-CNN except for the performance under noise corruption (LAMR increases from 76.14% to 83.24% for the *Sin* data augmentation).

Interestingly, while testing on the Cityscapes dataset (no distributional shift) standard data augmentation already provides a strong baseline and only *CutOut* augmentation can slightly improve over that. The proposed data augmentation obtains the best result (by a significant margin) when testing on ECP day-time images. When testing on night images, both stylized augmentation provides the best accuracy. For different types of corruption, the best methods are the same as in the Faster R-CNN model. Experiments on the CSP architecture confirm that the

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 3.5: Comparison of ECE for selected Faster R-CNN models on different datasets (lower value means better calibration).

Name	CityPersons	ECP-day	ECP-night	NightOwls
Baseline	**0.1418**	**0.1468**	0.1985	0.3765
Sin	0.1434	0.166	**0.1429**	0.3393
Ours	0.1429	0.1569	0.1572	**0.331**

proposed data augmentation allows for competitive results across different benchmarks and offers a good balance between accuracy on the clean dataset and under distributional shift

3.3.4 Uncertainty estimation

Providing reliable uncertainty estimates is an essential aspect of safe autonomous systems. In this section, the ECE score is measured using cross-dataset evaluation. It was found that stylized and Gaussian augmentations help to improve prediction confidence with no clear leader between them, so for conciseness Table 3.5 shows ECE scores of Faster R-CNN models for the *Baseline*, *Sin*, and for *Ours* models.

Ideally, the ECE score would be a constant and small value across different datasets, which would mean that the model "knows what it does not know." However, for the *Baseline* the ECE score goes up when testing on night-time images (jump from 0.1468 to 0.1985 on ECP dataset). *Ours* model has almost constant calibration error when switching to night-time images on the ECP dataset, whereas *Sin* model has an even smaller calibration error. Note from the previous section that the model accuracy drops in that case, which means that the improved calibration cannot be only attributed to the improved model accuracy. All of the models are significantly worse calibrated for the NightOwls dataset, because it is more challenging in general and because the distributional shift is greater in this case.

Fig. 3.5 shows calibration plots. It can be observed that the *Baseline* model has worse uncertainty calibration when switching from day to night-time images, especially in the area of high confidence predictions, which means that the model is underconfident in its predictions. However, for the proposed model (and other stylized and gaussian augmentations), there is no apparent difference in the calibration plot between day and night-time for the ECP dataset.

CSP architecture shows similar findings - using data augmentation usually improves model calibration, especially for the large distributional shift (night-time images). Interestingly, however, it was found that CSP models have worse calibration, e.g., for the Cityscapes dataset Faster R-CNN Baseline model has an ECE score of 0.1418, whereas the CSP model has a score of 0.2446 (Fig. 3.6).

MOST WIEDZY Downloaded from mostwiedzy.pl

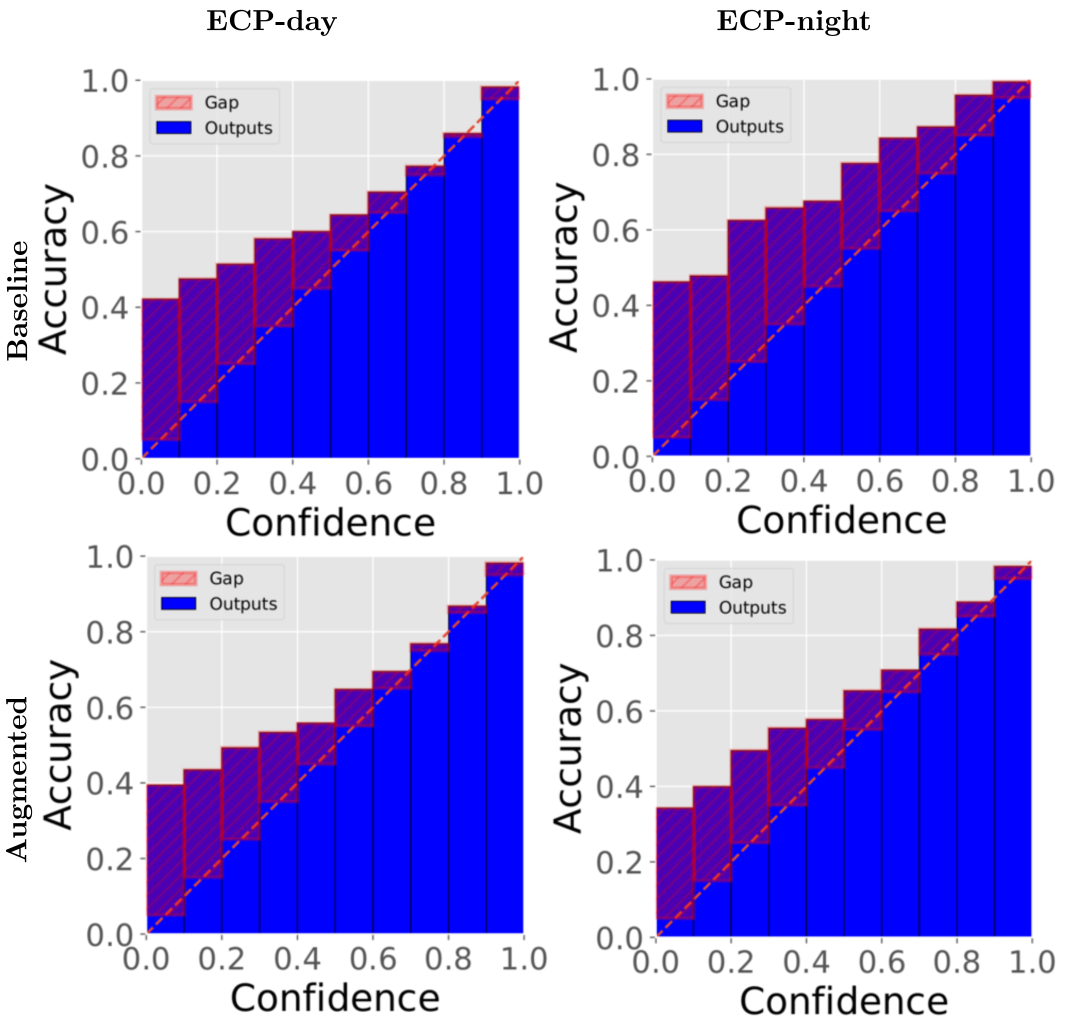


Figure 3.5: Calibration plots for selected Faster R-CNN models for day-time and night-time images on ECP dataset. Accuracy near diagonal means perfect calibration.

3.4 Conclusions

In this chapter, the examination was performed of pedestrian detection models in the real-world setting when test-time data come from a different distribution than in training: using cross-dataset evaluation, testing the model by switching illumination conditions (day to night), and through testing it on synthetic distortions. It was confirmed that such testing is crucial for a realistic evaluation of the model since the accuracy of the baseline model drops drastically. Further, it was shown that data augmentations in the form of stylized and Gaussian augmentations significantly improve the robustness of the model. Furthermore, a new data augmentation scheme was proposed that uses stylization but only on patches of the original image, and it was shown that such augmentation offers competitive accuracy. Finally, it was demonstrated that the use of data augmentations also improves the classification calibration of the pedestrian detection models.

It was shown that data augmentation is a simple and highly effective way

MOST WIEDZY Downloaded from mostwiedzy.pl

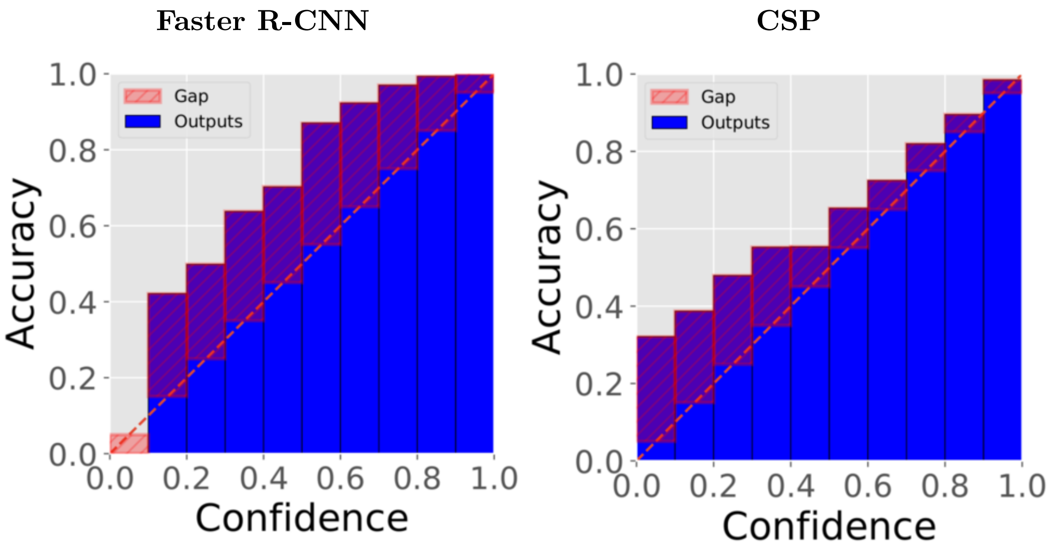


Figure 3.6: Calibration plots for Faster R-CNN and CSP architectures on Cityscapes dataset.

of improving model robustness to novel conditions encountered during the testing phase. At the same, the problem is challenging and remains far from being solved. Nevertheless, this chapter showed that using distinct types of tests is essential for realistically evaluating model performance, and using a limited number of tests may result in an over-optimistic estimation of model accuracy.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 4

Robustness in Compressed Neural Networks for Object Detection

In this chapter, the robustness of machine learning models again provides the main topic, but this time the focus is also on computational efficiency. Recent works, shown that it is possible to efficiently compress the neural networks (by removing neurons or whole filters) without accuracy loss [122]. It is an attractive method that would allow reducing computational cost during deployment significantly. However, most methods were only evaluated using the in-distribution testing, shown in the previous chapter to provide an over-optimistic estimation of model performance. Here a closer look at the accuracy of compression methods is taken in the out-of-distribution setting and when using per-class evaluation, and a method to mitigate some of the found issues is proposed. The work description contained in this chapter is based on the article [123].

4.1 Introduction

Optimization of the size of visual recognition models is of great importance, e.g., for autonomous driving, because of energy consumption and hardware cost. Typical methods to reduce the computation cost include deploying specialized architectures [21], model compression techniques such as reducing model precision (quantization) [20], and/or setting the number of weight or filters to zero (pruning) [19]. For example, Han et al. showed [122], that it is possible to reduce the size of the VGG network by a factor of 13 when benchmarking on the ImageNet dataset [7] with no loss in accuracy.

The most popular approach is magnitude pruning which removes several small magnitude weights resulting in only a small decrease in accuracy [122, 124]. However, to reduce the computational cost of such pruned models, specialized hardware is

MOST WIEDZY Downloaded from mostwiedzy.pl

required to optimize sparse operations. As a result, structured pruning was proposed where entire filters and/or layers are removed [125, 126, 127]. Standard approaches to model compression assume training the base model, pruning, and then fine-tuning [122] or gradually pruning the model during training [124, 127]. For the task of visual recognition, model compression has mostly been applied to image classification, with very few works on the task of object detection [128]. Since object detection is a more complex task than image classification and has significant application potential, it is important to evaluate model compression methods in the object detection task.

Model compression is also an interesting problem from a research perspective. It is well-known that current machine learning models are heavily over-parameterized, allowing them to fit random labels [129] easily, which is exploited but the compression techniques, to greatly reduce the model size, with only a small decrease in inaccuracy. But, investigating only the mean accuracy might not give the full picture of compression methods' impact on model predictions. Highly accurate models (in terms of average precision) can still fail in rare and atypical cases [93, 94]. Yet, most of the works in model compression focus on clean test-set accuracy, ignoring model robustness, such as out-of-distribution (o.o.d.) accuracy, which is crucial for systems operating in the real world.

It was only recently shown that pruning significantly affects robustness in the image classification task and might disproportionately impact different object classes [22, 130]. This chapter starts with those observations and applies them to object detection from RGB images. Further, the effect of naturalistic data augmentation on compressed models was analyzed. It was demonstrated that measuring out-of-distribution performance or per-class accuracy is crucial in safety-critical applications.

A parallel work to ours also investigates the effects of model compression in an out-of-distribution setting and confirms that such testing is critical in the context of safety-critical systems [131]. This chapter focuses on the object detection task (which was not investigated yet) and measures its impact on both synthetic and real distributional shifts and per-class accuracy.

4.2 Methodology

Model Compression. In experiments, standard magnitude pruning approaches were utilized. While more advanced approaches exist, magnitude pruning has been shown to consistently achieve excellent results across a number of datasets and tasks [132]. Another advantage is that magnitude pruning is a very general method that can be easily applied to a wide range of tasks and architectures. The automatic gradual pruning technique is used during training, which progressively increases the sparsity in the network throughout the training, up to the desired

MOST WIEDZY Downloaded from mostwiedzy.pl

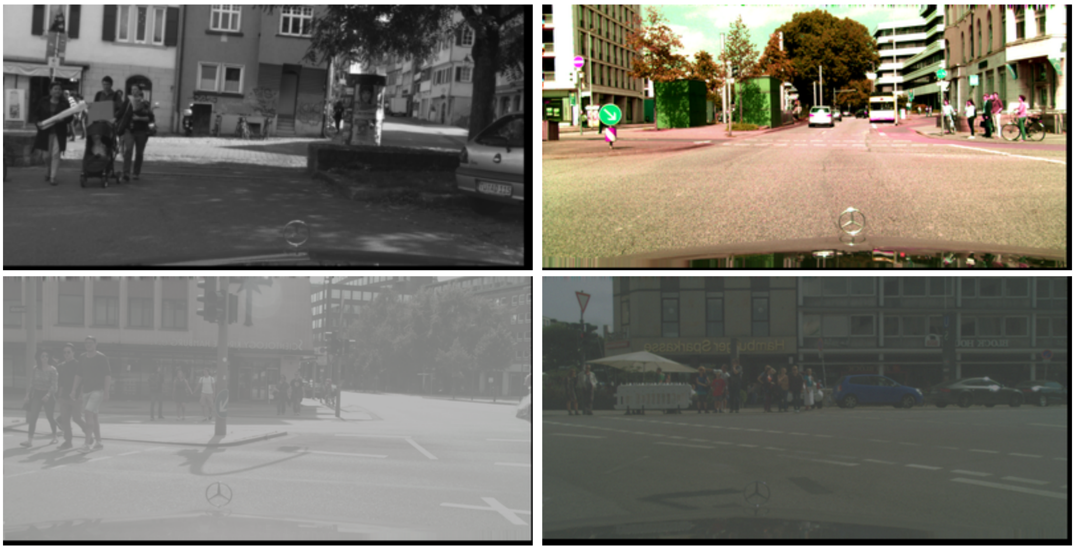


Figure 4.1: Examples of augmented images: color drop (top left image), color distortion (top right), overexposed image (bottom left), gaussian noise (bottom right).

compression rate. Specifically, the sparsity s_t at epoch t is computed as [124]:

$$s_t = s_f + (s_i - sf) * (1 - \frac{t - t_0}{n\Delta t})^3 \text{ for } t \in \{t_0, t_0 + \Delta t, ..., t_0 + n\Delta t\} \tag{4.1}$$

where n is the number of pruning steps, Δt is the pruning frequency, s_f is a final sparsity value, s_i an initial sparsity value (usually 0) and t_0 is an epoch at which pruning starts. At each iteration, L_1-norm is computed for each tensor and tensors with the lowest norm are zeroized, such that the desired level of layer sparsity s_t at a given epoch is achieved. Similarly, for structured pruning, the L_1-norm is computed at the filter level, which weights are set to 0.

Data augmentation. Several data augmentation techniques have been proposed to improve model robustness, in particular, style-transfer data augmentation is quite often used, as was also shown in the previous chapter. However, it might be possible that using simpler data augmentation, which distorts the texture information, might work as well, which was confirmed in recent work [133]. Namely, during training, the following augmentation is used in the pipeline (Fig. 4.1):

- Color distortion with a probability of 50%. This includes changes in the brightness, contrast, saturation, and hue of the image as specified in [39].
- Color drop (grayscale image) with a probability of 20%.
- Gaussian blur with a probability of 50%.
- Gaussian noise with a probability of 50%.

Imbalanced Data. Real-world datasets often follow a long-tail distribution: a few dominant classes are represented by a great number of examples, significantly

MOST WIEDZY Downloaded from mostwiedzy.pl

higher than of other less represented classes. As a result, models trained on such datasets provide poor accuracy on the underrepresented classes [134]. Significant research exists on dealing with such data imbalance, which can be categorized into *re-sampling* and *cost-sensitive learning*. For example, some of the training examples for the minority classes are repeated [135] or examples from dominating classes are undersampled). Cost-sensitive learning deals with the problem by assigning a relatively higher cost to the minority classes, e.g., computing the loss using the inverse of the class frequencies [136] or the inverse of the effective number of samples [137]. In this subsection, a few techniques for handling imbalanced data are described, the first technique is based on sampling, and the others are based on cost-sensitive learning.

The **repeat factor sampling (RFS)** strategy was recently shown to yield competitive results on class imbalance problems [138]. For each category c, let's define f_c as a portion of images that contain at least one instance of object category c. The category-level repeat factor is defined as:

$$r_c = max(1, \sqrt{(t/f_c)}) \tag{4.2}$$

where t is a hyperparameter. Intuitively, this means that categories which frequency f_c is below threshold t, will be over-sampled. Then the image-level repeat factor is computed as the maximum value over the categories in the image i:

$$r_i = \max_{c \in i} r_c \tag{4.3}$$

Cost sensitive learning, on the other hand, applies class-specific weights w_c to the cross-entropy loss in the classification task. For a given observation, the weighted cross-entropy can be computed as:

$$L_{wCE} = -\sum_{c=1}^{C} w_c * (y == c) log(\hat{y_c}) \tag{4.4}$$

where $(y == c) \in \{0, 1\}$ indicates whether class c is the correct class for given observation and $\hat{y_c}$ is a predicted probability for class c, and w_c is a weighting factor for every class. If $w_c = 1$ for all classes then the above formulation relates to the standard cross-entropy loss. Below, different approaches to computing w_c for data imbalance problems are briefly described.

Inverse square root of class frequency computes the w_c in exactly the same way as the repeat factor r_c was computed for the RFS algorithm. For our experiments, another variant was also tested where the weights were computed as $w_c = \sqrt{(t/f_c)}$ (so removing the *max* function), which allowed the weights of some frequent classes to be smaller than 1.

Computing class weights by means of a category-level repeat factor, as defined above, may yield suboptimal results, since some of the images may contain just

MOST WIEDZY Downloaded from mostwiedzy.pl

one instance of a given category, while others may contain dozens of them. As such, it was proposed in [137] to compute the weighting factors using the number of instances. Our implementation follows the details provided in [139]. First, the number of instances for each category N_c is computed. Then, the *effective number of samples* E_n for each category c can be computed as:

$$E_n = \frac{1 - \beta^{Nc}}{1 - \beta} \tag{4.5}$$

where β is the hyperparameter.

However, note that the above methods have mostly been tested on the image classification task, and object detection brings further challenges. First, object detection has a multi-task objective, and scaling classification loss may introduce side effects to the overall performance (for example, changing accuracy of the regional proposal network). Second, the above calculation does not take into account the background class (because it is hard to estimate the "frequency" of the background class, a class-weight of 1 is applied to the background class in all cases as in [139]). Third, since foreground/background separation is a very important part of object detection, one must be very careful when applying different class balancing methods. Therefore, in the experiments section, experiments are also conducted with linearly scaled variants of the above methods.

4.3 Experiments

Implementation details. The models were trained using the Faster R-CNN general-purpose object detector. The Distiller package [140] was used for pruning, using both structured and unstructured methods. Similar to [9], the Cityscapes model was trained for 64 epochs, with a learning rate step reduction by a factor of 10 at epoch 48. The initial learning rate was 0.01, and the batch size was 6 as this is the maximum that the GPU used can concurrently process. The pruning used the automated gradual pruning scheme [124] starting from the first epoch until epoch 56. The model was trained for 11 epochs for ECP and BDD datasets, with a learning rate step reduction by a factor of 10 at epoch 7. The pruning was gradually starting from the first epoch until epoch 8.

The models were pruned at 30%, 50%, and 70% compression rates for the structured pruning and at 50%, 80%, and 95% compression rates for the unstructured pruning. For each method, all of the compression rates can be considered to be a reasonable setup, with the first compression rates being more conservative and the last being more aggressive. Note that higher compression rates can be achieved for the unstructured pruning, which is why the compression rates were higher in that setting. The above models were trained 5 times, and the mean accuracy is reported.

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 4.1: Accuracy comparison for models trained with different pruning strategies tested on the Cityscapes dataset (first column) and different corruption types from the Common Corruptions benchmark (the remaining columns).

Model	Clean	Noise	Blur	Weather	Digital
Baseline	0.352	0.0	0.049	0.152	0.146
Unstructured pruning (compression rate)					
50%	0.351	0.0	0.047	0.151	0.14
80%	0.338	0.0	0.041	0.138	0.135
95%	0.323	0.0	0.029	0.115	0.118
Structured pruning (compression rate)					
30%	0.352	0.0	0.037	0.134	0.135
50%	0.337	0.0	0.027	0.105	0.131
70%	0.33	0.0	0.023	0.088	0.125

Table 4.2: Accuracy comparison for models trained using day-time images and tested on day-time images (first column) and night-time images (second column).

Model name	ECP-day	ECP-night
Baseline	0.468	0.392
Unstructured pruning (compr. rate)		
50%	0.462	0.396
80%	0.45	0.383
95%	0.414	0.331
Structured pruning (compr. rate)		
30%	0.457	0.382
50%	0.444	0.363
70%	0.431	0.34

4.3.1 Measuring impact of model compression on the robustness

Table 4.1 presents the results obtained for the models trained on the Cityscapes dataset using different compression strategies and evaluated on the clean Cityscapes dataset and its corrupted versions. For simplicity, in the evaluation, distortions are applied at the medium intensity level. The results from the second to the last column measure the robustness of the models (o.o.d. test). The first thing that can be noticed is that the models lack robustness and are very vulnerable to different distortions, as the mAP metric is very low for all distortion types. Further, for the structured pruning, it was possible to prune 30% of the filters and still achieve the same accuracy on the clean dataset (first column), however, models' sensitivity to different distortion types was already negatively affected.

While the previous experiment measured robustness to some synthetically generated distortions, using the ECP dataset, one can measure the robustness to natural distortion such as the transition from day to night (Table 4.2). Specifically, a model trained on day-time images is evaluated on day-time images (first column) and night-time images (second column, o.o.d. test). The decrease in mAP metric is comparable for both tests compared to the baseline model, for both pruning

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 4.3: Accuracy comparison for models trained using naturalistic data augmentation with different pruning strategies tested on the Cityscapes dataset and corruption types from the Common Corruptions benchmark. Values in brackets show accuracy change due to the added augmentation.

Name	Clean	Noise	Blur	Weather	Digital
Baseline	0.367 (+0.015)	0.194	0.126	0.258	0.271
Unstructured pruning (compr. rate)					
50%	0.364 (+0.013)	0.193	0.127	0.258	0.264
80%	0.359 (+0.021)	0.18	0.125	0.255	0.256
95%	0.326 (+0.003)	0.064	0.112	0.226	0.233
Structured pruning (compr. rate)					
30%	0.36 (+0.008)	0.178	0.122	0.252	0.252
50%	0.352 (+0.015)	0.154	0.122	0.243	0.251
70%	0.324 (-0.006)	0.103	0.113	0.221	0.232

Table 4.4: Accuracy comparison for models trained using naturalistic data augmentation on day-time images and tested on day-time images (first column) and night-time images (second column). Values in brackets show accuracy change due to the added augmentation.

Model name	ECP-day	ECP-night
Baseline	0.456 (-0.012)	0.419 (+0.027)
Unstructured pruning (compr. rate)		
50%	0.453 (-0.009)	0.417 (+0.023)
80%	0.444 (-0.006)	0.407 (+0.024)
95%	0.399 (-0.015)	0.363 (+0.032)
Structured pruning (compr. rate)		
30%	0.447 (-0.01)	0.407 (+0.025)
50%	0.433 (-0.011)	0.393 (+0.03)
70%	0.418 (-0.013)	0.381 (+0.041)

methods and evaluations (ECP-day and ECP-night).

4.3.2 Naturalistic data augmentation

A standard way to improve model robustness is using specialized data augmentation, it is however unclear what the effect of such augmentation will be on compressed models, especially at the highest compression rates. In this section, the models' robustness was again evaluated, but this time a naturalistic data augmentation' [133] was used during training. The results are presented in Table 4.3 and Table 4.4. Overall, one can see that, as expected, the out-of-distribution detection accuracy has greatly increased for both datasets for all models. For example, looking at the structurally pruned model at the 50% compression rate, one can see that the accuracy on the distortions unseen during training has greatly increased (0.243 and 0.251 mAP for weather and digital distortions compared to 0.105 and 0.131 mAP, respectively). Also, the accuracy on the clean dataset (first column) has significantly

MOST WIEDZY Downloaded from mostwiedzy.pl

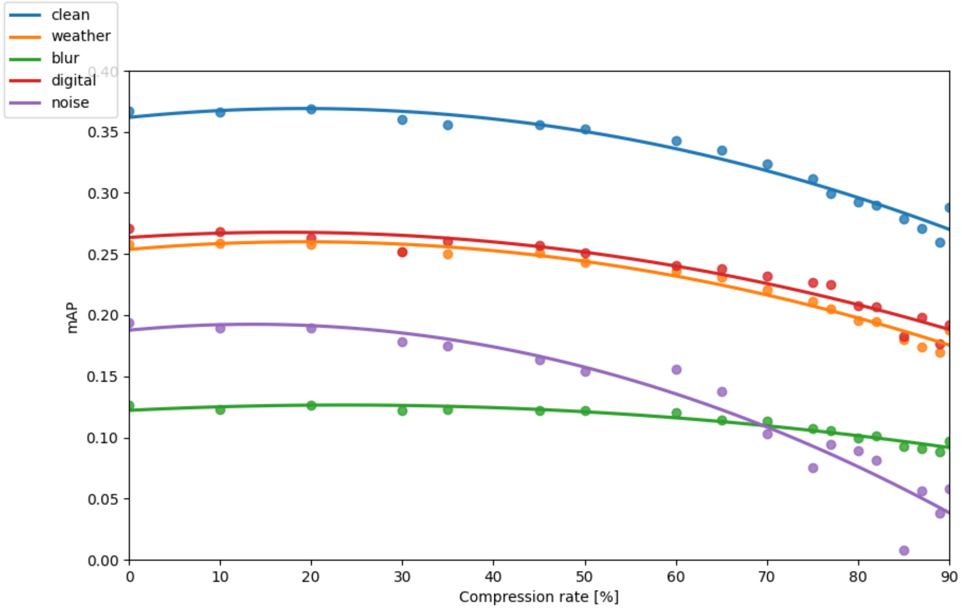


Figure 4.2: Effect of structured pruning with different compression rates (x axis) across different distortion types on a mAP metric, for a model using naturalistic data augmentation.

increased for all models, except the structurally pruned model at the highest compression rate, where the accuracy has slightly decreased. Interestingly, the effect of using naturalistic data augmentation is smaller at the highest compression rates.

On the ECP dataset, the loss in accuracy on day-time images was significant (0.15 and 0.13 at the highest compression rates for unstructured and structured pruning), however, this might be because a similar decrease can be noticed for the uncompressed model (decrease in mAP from 0.468 to 0.456). This shows that one has to be careful when setting the data augmentation parameters, probably using less aggressive augmentation on the ECP dataset would improve the results for day-time images. Nevertheless, the results for the night-time images greatly improved at all compression rates. For example, for the model structurally pruned at the 50% compression rate, after using naturalistic data augmentation, the mAP on the night-time images increased from 0.363 to 0.393. This shows that, in spite of limited capacity, the compressed models were still able to learn more texture-invariant representations of the objects.

It is also worth looking at how accuracies for specific corruptions is affected as the compression rate increases (Fig. 4.2). A few very interesting observations can be made. First, the accuracy for each corruption type was differently impacted by the pruning, and models' sensitivity to noise distortion was the most heavily impacted. While the initial accuracy was fair (0.194 mAP without any compression), the accuracy started to deteriorate very quickly when more than 30% of the filters were pruned. On the other hand, the accuracy for the blur distortions was almost flat, being only slightly reduced at the highest compression rates. Digital and weather

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 4.5: Per class accuracy of trained models. Aug stands for the naturalistic data augmentation and INV for the inverse class frequency re-weighting method. For the unstructured pruning, using data balancing methods bring similar gain across different compression rates, here only the accuracy at the highest compression rate is reported.

Name	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	mAP
Cityscapes									
Baseline + aug	0.39	0.404	0.577	0.265	0.495	0.222	0.256	0.329	0.367
Baseline + aug + INV	0.39	0.407	0.576	0.283	0.512	0.229	0.263	0.332	0.374
Unstructured (95%) + aug	0.359	0.38	0.552	0.205	0.454	0.16	0.228	0.312	0.331
Unstructured (95%) + aug + INV	0.354	0.378	0.546	0.221	0.466	0.186	0.233	0.31	0.337
Structured (30%) + aug	0.385	0.409	0.576	0.249	0.497	0.186	0.246	0.333	0.36
Structured (30%) + aug + INV	0.384	0.406	0.572	0.257	0.512	0.222	0.26	0.331	0.368
Structured (50%) + aug	0.378	0.402	0.57	0.251	0.473	0.176	0.236	0.332	0.352
Structured (50%) + aug + INV	0.379	0.399	0.567	0.256	0.502	0.214	0.245	0.331	0.362
Structured (70%) + aug	0.362	0.394	0.559	0.21	0.432	0.1	0.221	0.314	0.324
Structured (70%) + aug + INV	0.36	0.389	0.555	0.233	0.478	0.184	0.233	0.314	0.343
BDD									
Baseline + aug	0.318	0.256	0.408	0.392	0.417	0.0	0.218	0.221	0.279
Baseline + aug + INV	0.317	0.26	0.404	0.396	0.428	0.031	0.232	0.222	0.286
Structured (70%) + aug	0.266	0.193	0.388	0.322	0.339	0.0	0.150	0.163	0.228
Structured (70%) + aug + INV	0.276	0.222	0.389	0.35	0.369	0.0	0.179	0.185	0.246

distortions were similarly impacted by model compression, comparably to the performance of the original Cityscapes dataset. Relating the results to other work [8], it is worth noting that different distortions had different Fourier statistics. Some of them (i.e., shot and impulse noise) were concentrated in the high-frequency components of the image, while others (e.g., brightness, contrast) were concentrated in the low-frequency components. This might mean that pruning the visual recognition changes the models' sensitivity to the image's high- and low-frequency components.

4.3.3 Per class evaluation

In this section, instead of observing only the mean accuracy of the model, a per-class accuracy is also examined to see how the compression impacted different classes to understand the impact of model compression methods fully. This section conducted the experiments on Cityscapes and BDD datasets, as they provide ground truth for many classes. Different classes were disproportionately affected by the compression techniques (Table 4.5). Some classes were heavily impacted by the compression (e.g., truck, train, bus), while others were less affected (i.e., car). Many factors influence the final impact. One of those is the class imbalance (Fig. 4.3, i.e., car class is dominant in both datasets), but some classes were also inherently more complex than others (because they were similar to other classes, occurred with high occlusion rates, or were hard to distinguish from the background).

As some classes are more impacted than others, we have conducted experiments using methods for imbalanced datasets, namely:

- Repeat factor sampling (RFS),

MOST WIEDZY Downloaded from mostwiedzy.pl

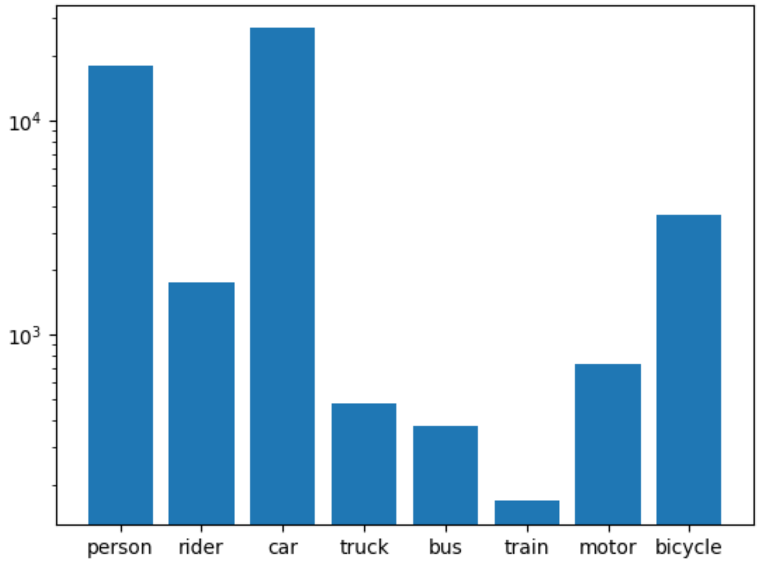


Figure 4.3: Cityscapes dataset class histogram (logarithmic scale).

- Inverse squared class frequency re-weighting with (INV_{cap}) and without (INV) setting the minimal weight to be 1.0 (as described in sec. 4.2),
- Effective number of samples (ENS) with $\beta = 0.99$.

For the re-weighting methods, we also experimented with the linear variants of the above methods using scaling factor $\lambda \in \{0.5, 1, 2\}$, and results are reported for the best performing scale.

Overall, very interesting results were obtained. The best-performing method utilized inverse class frequency re-weighting. Interestingly, while the effect of data imbalance was relatively small without any compression (mAP increased from 0.367 to 0.374 on Cityscapes, similarly on BDD, Table V), the effect was much more striking at the highest compression rates for the structurally pruned models. At the 70% compression rate level, the accuracy significantly increased from 0.324 to 0.343 on Cityscapes and 0.228 to 0.246 on the BDD dataset. As a sanity check, models have also tested at the 75% and 80% compression rates, confirming those results - the overall accuracy increased by around 0.02 mAP in both cases. For the unstructured pruning, the above finding was not observed. It might occur because structured pruning is a harder problem, and in the case of a model pruned with the unstructured method, it might be easier to accommodate different classes.

Fig. 4.4 compares different data balancing methods on the Cityscapes dataset. It can be noticed that for some classes (i.e., train, truck), the accuracy greatly increased after data balancing was applied, while on others, the accuracy remained almost the same. In general, all methods improved the compressed model (i.e, for the INV method, train accuracy increased from 0.1 to 0.184, and bus accuracy

MOST WIEDZY Downloaded from mostwiedzy.pl

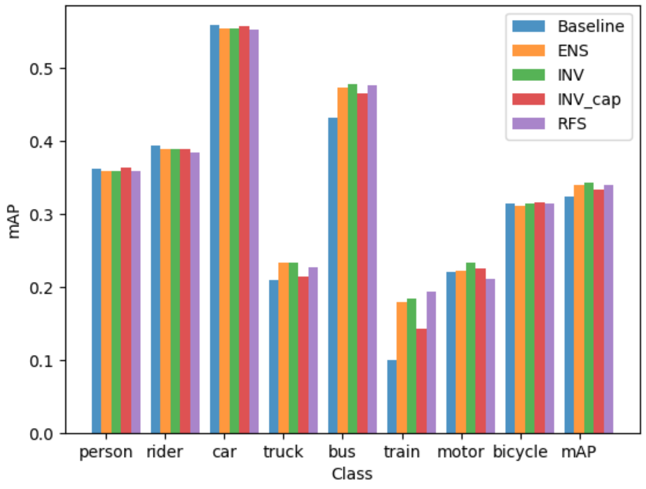


Figure 4.4: Per-class accuracy for models structurally pruned at the 70% compression rate using different class-balancing strategies.

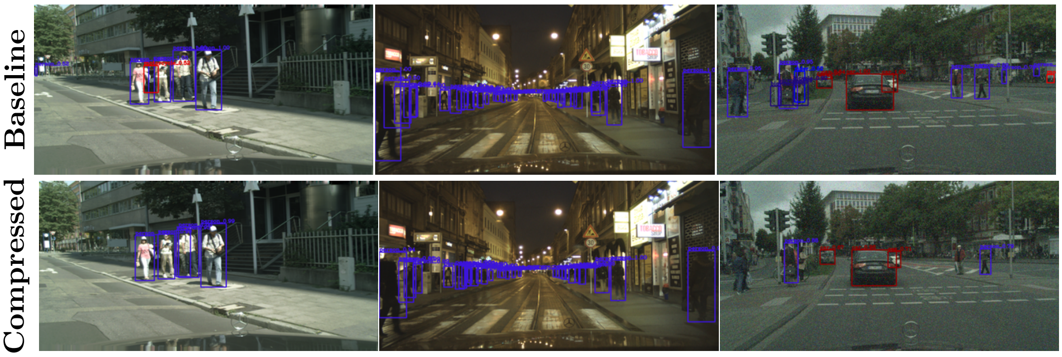


Figure 4.5: Detection samples for the baseline model and the model structurally pruned at 70% compression rate. Detections on the original Cityscapes dataset (first column), ECP dataset (second column) and noise distortion (third column).

increased from 0.432 to 0.478). Recent work studied models performance of the minority groups and show that the overparameterized models seem to learn patterns that generalize well on the majority groups but obtain inferior performance on the underrepresented classes [46]. On the other hand, this section studies per class accuracy on the real-world dataset in low-capacity models and shows that different data balancing methods could be very effective (for structurally pruned models).

Fig. 4.5 shows some detection examples. In general, the compressed model detects well visible objects in the image, however, the occluded objects might not be detected (the first column, missed motorcycle detection). Additionally, pruned models are more sensitive to noise distortion and might include more false positive detections.

MOST WIEDZY Downloaded from mostwiedzy.pl

4.4 Conclusions

This chapter showed that, despite limited capacity, compressed models could make effective use of naturalistic data augmentation to learn more texture-invariant representations, which significantly increased model robustness to synthetic distortions and day to night transition. It was found that model compression affects models' sensitivity to different distortion types differently. Some of them, i.e., those concentrated in the high-frequency domain such as Gaussian noise, were heavily affected by pruning techniques, while for others (i.e., blur distortions) the impact was much smaller.

In particular, it was demonstrated that data balancing methods might be especially useful in structurally pruned neural networks. Inverse class frequency re-weighting increased the overall mAP by 0.007 (a 1.9% relative increase). On the Cityscapes dataset, at the 70% compression rate, in the case of structured pruning, the mAP increased by 0.019 (5.9% relative increase). Similar results were obtained for the BDD dataset. Both sampling-based methods (repeat factor sampling) and cost-sensitive methods (i.e., inverse squared class frequency re-weighting) turned out to be effective.

Overall, this work explores the relationship between model robustness and compression techniques and provides insights into improving deployed models' performance and computational cost. It was shown that testing compressed models in out-of-distribution settings or measuring per class accuracy for safety-critical systems is important to fully understand the effects of model pruning.

In general, this chapter again shows the importance of carefully evaluating visual models: employing out-of-distribution testing and also by measuring per class accuracy. Furthermore, as in the previous chapter, data augmentation strategies were shown to play a crucial role in enhancing models' robustness.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 5

Closer Look at the Uncertainty Estimation in Semantic Segmentation under Distributional Shift

In this chapter seeking robust methods for visual recognition is continued, this time focusing on semantic segmentation task. Model ensembling is another popular method for improving model accuracy. Consequently, it was used to understand how well does it work in the o.o.d. setting, with varied level of distributional shift. Further experiments focused on transferring the knowledge from model ensemble to single model in the domain adaptation setting.

The work description contained in this chapter is based on the article [62] and was done in collaboration with students with the following division of work. K. Woźniak was responsible for conducting style-transfer experiments, R. Słowiński for models' evaluation and B. Wróblewski has conducted domain adaptation experiments. The author of the thesis was responsible for obtaining baseline models and has designed all of the experiments. The supervisor of the thesis, A. Czyżewski, coordinated the joint effort.

5.1 Motivation

One of the problems with modern neural networks is that they are poorly calibrated and tend to be overconfident in their predictions [98]. Improved model calibration has many potential use cases as follows:

- It can be used to merge predictions from different modalities effectively,
- It can also be used offline, e.g., to annotate a new dataset automatically, and flag images that the model is not confident about.

MOST WIEDZY Downloaded from mostwiedzy.pl

In this chapter, methods for improving model calibration are of interest when no assumption is made about the data (i.e., it may contain conditions not prevailing in the training dataset).

Different techniques exist for improving estimates of predictive uncertainty. A classical approach is called temperature scaling, where the model confidences are scaled using a post-hoc procedure on the held-out validation set [141]. A popular approximate Bayesian approach is a dropout-based model, where the predictive uncertainty is computed based on the multiple outputs of the model on a given image (with dropout enabled) [142]. Another sampling-based approach uses agreement between an ensemble of models as a measure of model uncertainty. Interestingly, using ensembles has been shown to yield the best results on uncertainty estimation under the distributional shift [99, 143]. The ensembles' common setup is to use neural networks trained using different random initialization weights to induce diversity between the models [24]. This is because it has been shown that networks pre-trained on the same dataset stay in the same basin in the loss landscape and thus reduce variation in the models [144].

When the models are deployed to the real world, the encountered data usually comes from different (but similar) distributions. As such, many methods for domain adaptation have been proposed which use unlabeled data from the target domain to improve the accuracy of the model. Popular approaches include matching image statistics between domains [145], learning shape-based representation [146], self-learning [147], and using data from simulation [148, 149]. Using simulated data, in particular, is interesting since a simulation allows numerous and diverse training examples to be generated. Simultaneously, the difference in data distribution between the source and target domains is challenging for real-world problems, and sometimes using labeled data can negatively impact performance (when there is a large distributional shift) [150]. As such, it is important to evaluate the models' performance under varying levels of distributional shift.

The self-learning method was used in this work, which works in two stages. First, given a trained model, confident predictions are gathered for the target domain called pseudo-labels. Then, in the next stage, the pseudo-labels are used to fine-tune the model, which allows for domain adaptation. The potential problem with self-learning is that the gathered pseudo-labels might contain erroneous predictions. It was proposed to use an ensemble approach to gather the pseudo-labels, as ensembles are known to have good accuracy and uncertainty estimation, which are crucial for the efficient pseudo-labeling stage. Self-learning was also used by author of this thesis, to adapt the model to the target domain (in traffic surveillance setting) [151].

This chapter focuses on studying uncertainty estimation for semantic segmentation, an important task with significant application potential. The uncertainty calibration is studied in two different settings, namely:

1. when a model trained on the simulation is tested on real-world data (large

MOST WIEDZY Downloaded from mostwiedzy.pl

distributional shift),

2. when cross-dataset evaluation (mild distributional shift) is used.

This chapter is aimed at the reality-check for uncertainty estimation and domain adaptation methods. Studying the varying domain shift performance for the methods above is vital empirical work for real-world applications.

Similar to this chapter, [152] shows that an ensemble of models efficiently improves uncertainty estimation in medical image segmentation. Additionally, the effect of ensembles under distributional shift and their utility for the downstream domain adaptation task was shown. Ensemble predictions on unlabeled datasets were also used as soft targets for direct training supervision for the classification problem [153]. Here, an alternative approach with hard labels was used, where the least confident predictions are discarded during training for semantic segmentation.

A similar approach to ours [146] uses style-transfer data augmentation to train a base model, which is further adapted to the target domain using self-training. It was shown that simpler data augmentations could also be very efficient and that ensemble of models makes the fine-tuning stage more efficient, which, in turn, makes our work complementary.

5.2 Experiments

5.2.1 Methodology

Ensemble of models. The model ensemble was utilized to improve model calibration, which has been shown to provide the best results among other methods, especially under distributional shift [99]. In the case of the ensemble approach, it is common to train models using randomly initialized networks to induce diversity between models [24]. However, it was found that semantic segmentation models trained on the GTA or Cityscapes dataset without pretraining performs rather poorly. As a result, ImageNet pretraining was used. However, different backbones and/or augmentation methods were used to achieve diversity between the models. It was shown in the literature that using as few as 5 models can provide excellent results [99], and because of the computational budget, 5 models were used in experiments. Namely, given M independently trained models, a final semantic segmentation p_E for the image x can be computed as the average of all models predictions:

$$p_E(x) = \frac{1}{M} \sum_{m=1}^{M} p_m(x) \tag{5.1}$$

where p_m is the prediction of the *m-th* model in the ensemble.

To improve the models' adaptation to the distributional shift, style-transfer data augmentation, and color jitter transformations were utilized, similarly to pre-

MOST WIEDZY Downloaded from mostwiedzy.pl

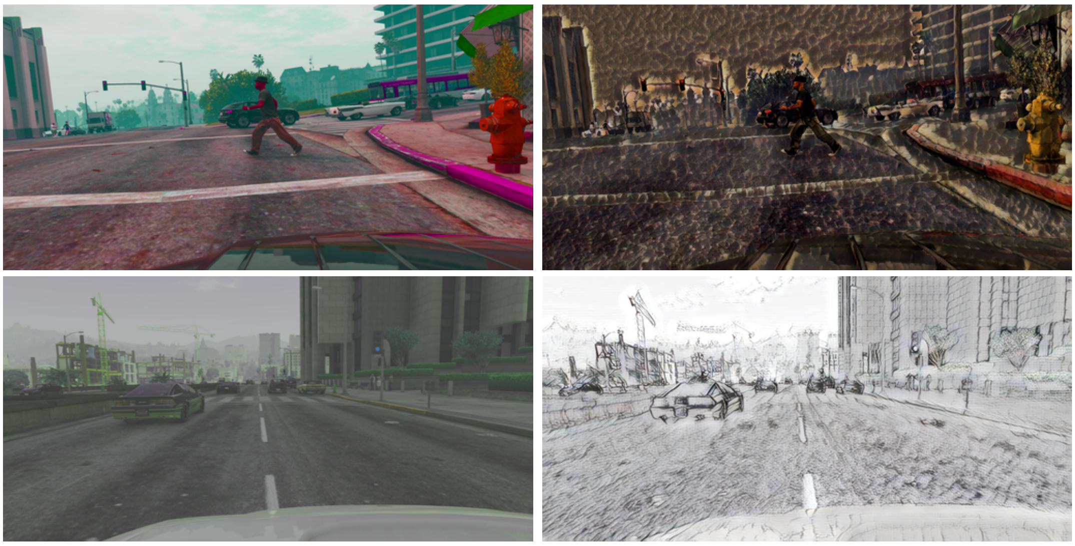


Figure 5.1: Different augmentation strategies applied to sample images from the GTA dataset. First column - color transformations, second column - style transfer.

vious chapters. However, using different augmentation strategies could also be beneficial in the ensemble, as the models trained with different augmentations might learn different representations. Fig. 5.1 shows examples of augmented images.

For the experiments, popular semantic segmentation datasets were used. They contain dense pixel-level semantic annotations for the same 20 classes (including the "ignore" class – usually representing the background). Experiments included domain adaptation from simulation to real data (GTA-to-Cityscapes) and cross-dataset evaluation (Cityscapes-to-BDD).

Implementation details. For all experiments, the DeepLabv3+ [65] network was used with different backbones (ResNet-101, Xception41, Xception65) pre-trained on ImageNet. Specifically, all models were trained on 2 GPUs for 100,000 steps with a batch size of 16. A polynomial decay learning rate was used, with an initial learning rate = 0.01 and exponent parameter set to 0.9.

The data augmentations are consistent with the official implementation[1], specifically random scaling (in the range 0.5 to 2.0), and left-right flipping was applied during the training procedure. All images were rescaled to the size of 512 x 1024 pixels. Color jittering was applied using TensorflowAPI with the following transformations: random brightness (adjustment factor in the range [0, 0.25)), random contrast (contrast factor in the range [0.5, 1.5)), random saturation (saturation factor in the range [1.0, 3.0)), and random hue (hue offset in the range [0, 0.25)). The chosen hyperparameters were experimentally validated to provide visually diverse images.

All models were evaluated using the validation sets (as the test sets' ground

[1] https://github.com/tensorflow/models/tree/master/research/deeplab

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 5.1: Performance of DeepLabv3 using ResNet-101 backbone under different evaluation settings. CJ models were trained using color jittering and SIN models used style-transfer augmentation.

Model name	mIoU	pix. acc	ECE	mIoU	pix. acc	ECE
	GTA-to-GTA			**GTA-to-Cityscapes**		
Baseline	**80,8**	**96,6**	**0.16**	25,4	60,1	23.36
CJ	80,6	96,4	0.21	**40,4**	83,7	6.5
SIN	77,2	95,9	0.21	40	**83,9**	**5.06**
	Cityscapes-to-Cityscapes			**Cityscapes-to-BDD**		
Baseline	74,1	**95,5**	1.49	42,8	83,9	9.55
CJ	**74,4**	95,4	1.36	49,1	89,4	5.07
SIN	71,4	95	**1.18**	**49,3**	**89,8**	**4.56**

truth data are not publicly available). During the fine-tuning stage, the models were trained for 25,000 steps as the training loss converged around 20,000 steps for all models. CJ stands for a model trained using color jittering transformations when reporting the results, while SIN stands for a model trained using style-transfer, as in [14].

5.2.2 Baseline models

First, the DeepLabV3+ model with the ResNet backbone was trained on both the GTA and Cityscapes datasets and further evaluated (Table 5.1). Several observations can be made. First, there was a very significant drop in accuracy when the models were evaluated under domain shift. The gap was larger for sim-to-real adaptation (GTA-to-Cityscapes) compared to the cross-dataset evaluation (Cityscapes-to-BDD). Further, it can be observed that evaluated data augmentations only slightly affected the performance on the source domain, but they showed impressive performance in the domain adaptation setting. For GTA-to-Cityscapes, the mIoU increased from 25.4 to 40.4, and similarly, for Cityscapes-to-BDD, the mIoU rose from 42.8 to 49.1. Nevertheless, the domain gap was still quite large; a model trained on the Cityscapes dataset achieved a mIoU of 74.1, compared to 40.4 achieved by a model trained on the GTA dataset.

Noticeably, applying simple color transformations worked as well as using a more complicated style transfer technique, consistent with recent findings [133]. Looking at the model calibration, one can notice that all of the models were almost perfectly calibrated when evaluated on the source domain, however when evaluated under domain shift, the ECE metric greatly increased, e.g., for a model trained on the Cityscapes dataset, the metrics increased from 1.49 to 9.55 when evaluated on the BDD dataset instead of Cityscapes. Consistent with section 3, it was shown that using texture-based data augmentation improved model calibration under domain shift [16], with the SIN model obtaining slightly better results than using color transformations. In general, using any of the data above augmentations was crucial

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 5.2: Xception models performance under cross-dataset setting.

Name	mIoU	pix. acc	ECE
GTA-to-Cityscapes			
Xception41 (CJ)	41.8	82.7	7.3
Xception41 (SIN)	**43.7**	**86.2**	**4.05**
Xception65 (CJ)	41.3	82.0	7.47
Cityscapes-to-BDD			
Xception41 (CJ)	**52.6**	90.3	4.5
Xception41 (SIN)	51.1	**90.9**	**3.74**
Xception65 (CJ)	52.4	90.4	5.09

Table 5.3: Ensemble of models performance. Also, mean performance of all models is reported.

Model name	mIoU	pix. acc	ECE	mIoU	pix. acc	ECE
	GTA-to-GTA			**GTA-to-Cityscapes**		
M=3	**81.9**	**96.8**	0.81	43.2	84.7	2.45
M=5	81.4	96.7	1.02	**44.5**	**86.3**	**1.1**
Models mean	79.0	96.3	**0.21**	41.4	83.7	6.08
	Cityscapes-to-Cityscapes			**Cityscapes-to-BDD**		
M=3	**77.2**	**96.0**	0.36	55.7	91.3	1.99
M=5	77.0	**96.0**	**0.29**	**56.2**	**91.7**	**1.09**
Models mean	73.8	95.4	1.16	49.3	89.8	4.56

in the domain adaptation setting.

5.2.3 Model calibration

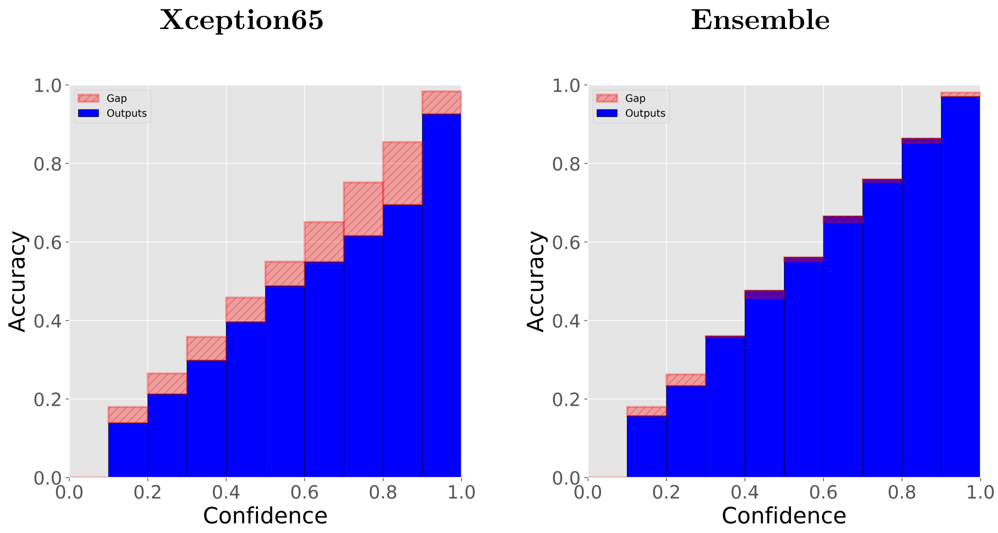

Xception65 **Ensemble**

Figure 5.2: Calibration plots for Xception65 model and model ensemble (M=5) evaluated on the GTA-to-Cityscapes adaptation. Note great calibration for the ensemble of models.

MOST WIEDZY Downloaded from mostwiedzy.pl

An ensemble of models method was used to improve model calibration, which utilized three different backbones (ResNet-101, Xception41, Xception65) and two different augmentation methods (color jitter and style-transfer). We also experimented with the PNAS architecture [154], which is known to achieve great accuracy, however, the performance was not satisfactory, as no pre-trained model is currently available for that model. Table 5.2 shows the performance in the cross-dataset setting for the Xception models. Comparing this to Table 5.1, one can see that Xception models performed slightly better than models using ResNet-101 as the backbone.

Table 5.3 shows ensemble performance. The constructed ensemble consists of $M = 3$ models, for which color jittering transformations were used during training using ResNet-101 and Xception backbones. Further, two additional models were trained using style transfer augmentation (ResNet-101 and Xception41 backbones) and results in another ensemble variant with $M = 5$ models. While the results with no domain shift were comparable, obtained results were better under the domain shift when using 5 models. The mIoU increased from 43.2 to 44.5, and 55.7 to 56.2 on the Cityscapes and BDD datasets, when compared to the ensemble using 3 models.

Similarly, the ECE was significantly reduced on both datasets. It is also very important to notice that the ensemble performance was better than its strongest member, i.e., for the Cityscapes-to-BDD transfer, the strongest single model obtained a mIoU of 52.6 (Xception41 - CJ), while the ensemble accuracy was 56.2. Similarly, the ECE significantly improved for the ensemble under domain shift: for the GTA-to-Cityscapes transfer, the ensemble ECE was 1.09, while the best result from a single model was 4.05 (Xception41 – SIN). Additionally, Fig. 5.2 shows the calibration plot, comparing the model calibration of the highest capacity model (Xception-65) with the calibration of the ensemble. Overall, it was confirmed that the ensemble improved both accuracy and uncertainty calibration, especially under domain shift.

One of the potential usages of well-calibrated uncertainty estimation is self-training. For this purpose, the precision/recall points for different confidence thresholds t were evaluated (Fig. 5.3). Namely, such a curve approximates how many pixels can be automatically annotated with a given precision. Overall, it can be noticed that much higher recall values were obtained for the ensemble. For example, with a precision of 95% for the Xception65 model, the recall was around 56.5%, while it increased to 71.2% for the ensemble. This shows that ensembles are a potent technique. A complimentary work to ours shows that ensembles can be used to label a new dataset efficiently [155]. The ensemble was there used to annotate new data with high accuracy coarsely, and then human annotators were employed to refine the initial predictions.

MOST WIEDZY Downloaded from mostwiedzy.pl

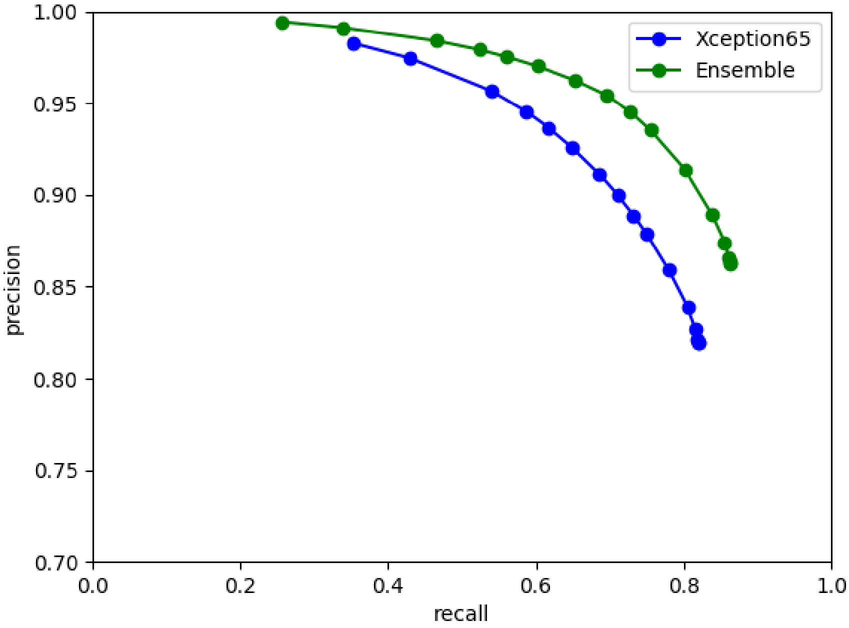


Figure 5.3: Precision / recall points evaluated at different confidence threshold starting from 0.1 (bottom-right points) to 0.995 (top-left points) on GTA-to-Cityscapes transfer. Note that Y-axis (precision) starts at 0.7 value to provide more detailed view.

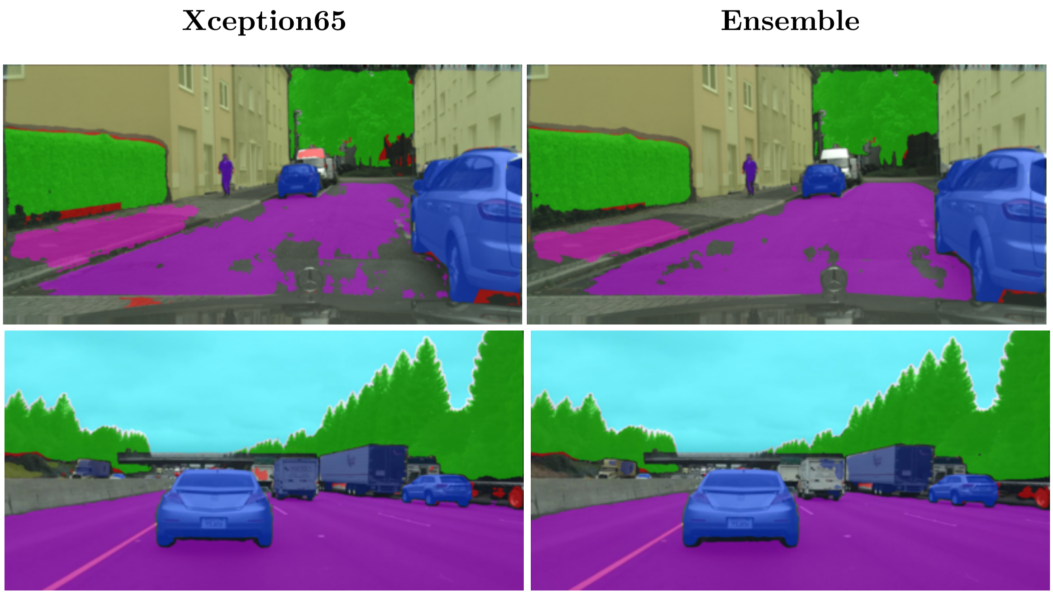
5.3 Domain adaptation

As was shown, the ensemble of models improved the model precision in the domain adaptation setting and greatly improved uncertainty estimation, which can be efficiently utilized in the self-training setting. First, a semantic segmentation model was used to obtain pseudo-labels on the target datasets, using some threshold t. The threshold of value 0.9 is commonly used [146], and the same value was used in the experiments. For the ensemble variant, such a threshold allowed 70.1% of the pixels to be annotated with 92.6% accuracy on GTA-to-Cityscapes transfer. Fig. 5.4 shows obtained pseudo-labels. In general, it can be noticed that the ensemble's labels were less noisy, and the object boundaries were more tightly aligned around the object of interest.

After the pseudo-labels were obtained for the target datasets, they were used for model fine-tuning. In this section results for different models are presented:

1. ResNet-101 using standard data augmentation.
2. ResNet-101 trained using additional color jittering data augmentation.
3. The previous model fine-tuned on target datasets using pseudo-labels obtained by that model (*CJ + fine* in the tables)
4. ResNet-101 fine-tuned on target datasets using pseudo-labels obtained by the

MOST WIEDZY Downloaded from mostwiedzy.pl



Xception65

Ensemble

Figure 5.4: Examples of pseudo-labels obtained on GTA-to-Cityscapes transfer (first row), and on Cityscapes-to-BDD transfer (second row).

Table 5.4: Domain adaptation results for our models with per-class evaluation.

Name	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle	**mIoU**
Gta to Cityscapes																				
Baseline	29.9	17.4	62.8	13.2	14.7	15.5	26.8	10.7	79.0	8.4	47.4	53.5	10.3	48.2	25.7	3.04	0.	11.4	4.5	25.4
CJ	80.3	28.9	80.9	30.9	22.5	25.8	37.0	17.5	83.8	31.0	76.6	**58.4**	**19.6**	83.0	28.7	24.7	0.	**27.4**	**11.0**	40.4
CJ + fine	86.1	36.4	83.1	24.9	28.7	27.8	**39.6**	19.4	85.7	38.4	79.5	56.9	13.0	86.5	31.0	23.6	0.	22.6	0.	41.2
CJ + ens	**88.6**	**43.2**	**85.0**	**36.3**	**33.8**	**30.7**	37.4	**21.9**	**86.8**	**44.9**	**83.9**	57.5	14.5	**87.3**	**37.2**	**32.2**	0.	15.0	0.	**44.0**
Cityscapes to BDD																				
Baseline	88.9	52.4	65.2	18.5	18.7	35.2	35.7	31.9	78.2	36.1	75.8	47.3	22.3	78.5	23.4	32.7	0.	41.2	32.0	42.8
CJ	91.8	54.5	79.9	**19.8**	27.1	**41.9**	**43.3**	43.8	82.5	39.1	91.4	58.2	29.7	85.2	27.7	25.5	0.	**49.1**	42.6	49.1
CJ + fine	93.2	60.4	**81.4**	18.7	36.6	37.4	40.5	44.2	83.0	42.0	91.7	62.2	43.7	85.1	36.4	23.6	0.	47.6	48.7	51.4
CJ + ens	**94.4**	**62.5**	81.0	17.5	**37.7**	38.6	38.6	**45.5**	**85.0**	**43.2**	**92.2**	**63.2**	**46.8**	**87.1**	**42.6**	**54.7**	0.	44.9	**53.4**	**54.2**

model ensemble (*CJ + ens* in the tables)

Table 5.4 shows the final results, including per-class evaluation. Firstly, consistent with other works, the self-training approach improved the model accuracy (from 40.4 to 41.2 and from 49.1 to 51.4 on the Cityscapes and BDD datasets, respectively). When the pseudo-labels were collected using an ensemble approach, the model accuracy was further greatly increased (from 41.2 to 44.0 on the Cityscapes dataset and from 51.4 to 54.2 on the BDD dataset). Fig. 5.5 shows qualitative results. In general, it can be noticed that obtained segmentation maps are less noisy, especially in more challenging cases (second and third row).

The fine-tuning was also performed on the highest-capacity model (Xception65). In that case, the mIoU has increased from 41.3 to 45.3, and from 52.4 to 53.6 on Cityscapes and BDD datasets, respectively. This shows that an ensemble approach is effective, also when the fine-tuned model is a very strong member of the ensemble.

MOST WIEDZY Downloaded from mostwiedzy.pl

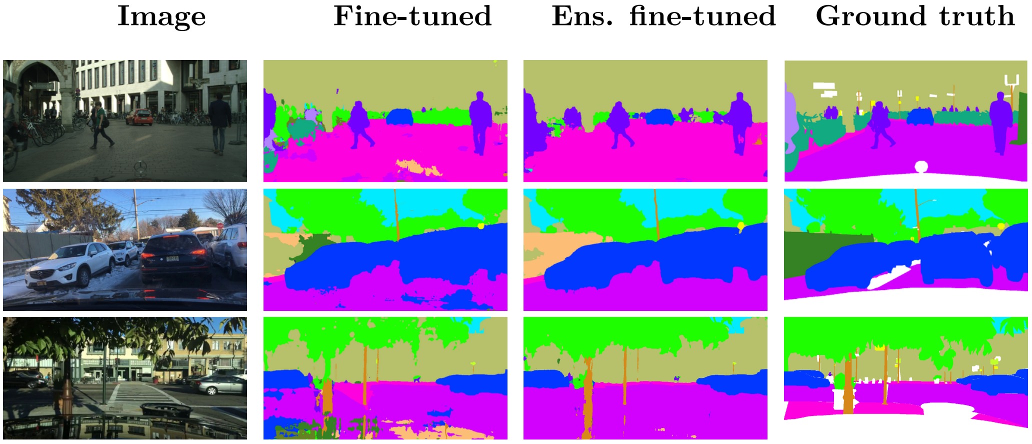


Figure 5.5: Qualitative results of trained models on GTA-to-Cityscapes transfer (first row) and Cityscapes-to-BDD transfer (consecutive rows). White color corresponds to the ignore label.

In general, the results are very promising. The ensemble approach turned out to be very effective in terms out model accuracy and uncertainty estimation, even for the large distributional shift (GTA to Cityscapes). On the other hand, ensembling did not improve for classes with the lowest precision (bicycle, motorcycle, rider). It might occur because the baseline model is a weak detector of such classes, so as a result, there are very few pseudo-labels collected for those classes. Improving accuracy for such classes remains an open challenge in self-training methods. Another important problem with an ensemble is that multiple models have to be trained and evaluated, which is very costly. However, the recently introduced BatchEnsemble method significantly reduced the computational and memory costs [156].

5.3.1 Conclusions

This chapter studied a calibration of predictive model uncertainty under different, realistic real-world application settings. It was shown that the ensemble of models significantly improved the uncertainty estimation and overall accuracy, especially under domain shift. Notably, the performance gains are consistent even when the domain gap is large (simulation-to-real transfer case).

Our ensemble consists of models using different backbones and/or data augmentations. Interestingly, it was also shown that simple color transformations can lead to a similar performance improvement as much more sophisticated style-transfer augmentation. Both types of data augmentation are crucial in the domain adaptation setting, which confirms and extends recent findings [39].

Further, the ensemble of models was utilized for domain adaptation using the self-training method. The improved uncertainty calibration and model accuracy allowed the fine-tuning stage to improve significantly since the mIoU increased from 41.2 to 44.0 and from 51.4 to 54.2 on the Cityscapes and BDD datasets. Furthermore, the proposed approach is complementary to other domain adaptation methods

MOST WIEDZY Downloaded from mostwiedzy.pl

based on self-training. Thus it could be easily combined with them, providing an interesting future work subject.

The main difficulty with applying model ensembles is its very high computational cost and the next chapter aims at reducing its cost while maintaining accuracy.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 6

Robust Object Detection using Multi-input Multi-output Framework

The previous two chapters explored two apparently contrary methods. In section 4, the model robustness was evaluated when compressing the models, which reduced computational cost. On the other hand, chapter 5 explored model ensembling, which improved model accuracy, but at the same time, it largely increased computational budget. The research question in this paragraph seeks to determine whether those two approaches can be efficiently combined. Work described in this chapter is based on the article [157].

As shown in the previous chapter, sampling based-methods were shown to obtain very good results in terms of accuracy, out-of-domain robustness, and improving model predictive uncertainty [152, 99]. Unfortunately, this comes with a high computational cost. As a result, several methods were developed to reduce the high computational cost, such as test-time dropout [142] or batch ensemble [156]. Another competitive approach is the m-heads model [158], which can be viewed as an ensemble with parameter sharing in the first layers of the network. Those methods, however, do not always match ensembling accuracy, as the success of sampling-based methods lies in the diversity of the predictions, which is a challenging problem [159, 160].

The literature on obtaining many predictions from one model using a single inference step has recently increased. These methods were inspired by the compression methods, which show that it is possible to remove even 90% of the weights without affecting the final model accuracy [122, 161], as was also shown in chapter 4. Therefore, instead of compressing the model, it should be possible to fit more than one subnetwork within the main network. For example, [28, 27] uses a single model in the multitask setting, and the latter approach retrieves a subnetwork (from the main model) to efficiently solve the target task. Another method uses the

MOST WIEDZY Downloaded from mostwiedzy.pl

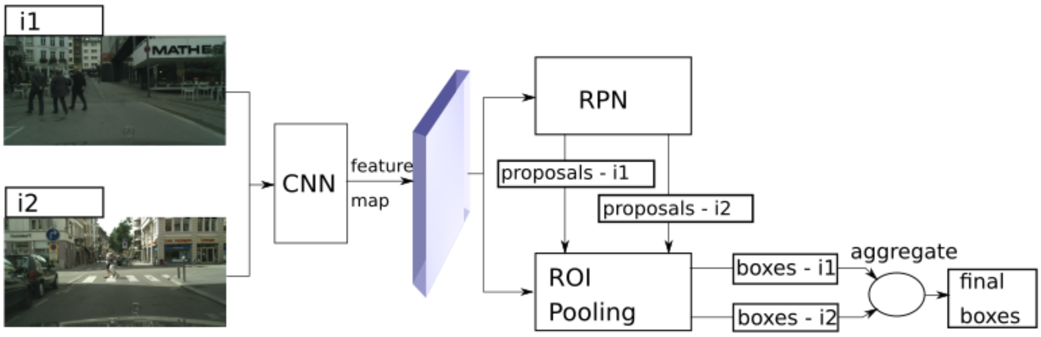


Figure 6.1: Architecture of the proposed MIMO Faster R-CNN. Both images are sampled independently during training, and each subchannel in the network is responsible for predicting boxes in the corresponding image. During testing, both inputs to the network are the same, and the final results are obtained by running aggregation on both channel results.

multi-input multi-output (MIMO) approach, where a single model makes multiple predictions simultaneously. MIMO was shown to increase the computational cost only slightly while matching the accuracy of model ensembling and was showcased on the image classification task. Yet, whether the MIMO approach would work in a multi-task setting such as object detection, particularly when regressing objects' localizations. The MIMO method is adapted for object detection tasks and further evaluated in this work. To summarize, the contributions of this chapter are as follows:

- The multi-input/multi-output model was adapted to the object detection task, and the architectural changes and implementation details are presented.
- The in-depth analysis of the MIMO approach was performed.
- The robustness of the MIMO approach is presented by comparing its results to different Deep Ensemble approaches, which it outperforms (unless a larger number of models are used for the Deep Ensemble) or matches in the accuracy.

6.1 Proposed architecture

Similar to previous chapters, standard object detection architecture was used, namely Faster R-CNN. Recall, from chapter 2.2.2, that the standard Faster R-CNN model consists of two main modules: first, region proposals are generated using a Region Proposal Network (RPN), and in the second stage, the region proposals are classified and refined using a Region of Interest Pooling network. Finally, computation of the feature map is shared between both networks and obtained using some standard CNN architecture.

In this chapter, the multi-input multi-output approach is applied to the Faster R-CNN model, the architecture overview is presented in Fig. 6.1. To adapt the Faster R-CNN model into the MIMO framework following changes were applied:

MOST WIEDZY Downloaded from mostwiedzy.pl

- Multiply the number of input channels by M (ensemble size).
- Region Proposal Network now outputs M sets of region proposals (each per input image).
- The ROI Pooling layer independently processes M set of proposals and the outputs need to be aggregated at test-time. This can be done using the standard non-maximum suppression (NMS) method or more advanced methods, such as Weighted Boxes Fusion [162].

Note that the feature map (output from the convolutional backbone) is of the same size as before, however, now it contains information about M images, without forcing any explicit structure on how to share the information from different images. Using that shared feature map RPN returns M independent sets of region proposals, which requires changing the RPN loss function to:

$$L(\{\hat{p_{im}}\},\{\hat{t_{im}}\}) = \frac{1}{M}\sum_{m=1}^{M}(\frac{1}{N_{cls}}\sum_{i} L_{cls}(\hat{p_{im}}, p_{im}) + \frac{\lambda}{N_{box}}\sum_{i} p_i \; smooth_{L1}(\hat{t_{im}} - t_{im})) \tag{6.1}$$

where i is the anchor index, and m is the index of input/output pair, $\hat{t_{im}}$ are predicted parametrized bounding boxes, $\hat{p_{im}}$ are predicted probabilities of the anchor being an object, and t_{im}, p_{im} are the ground truth counterparts. The equation is normalized by N_{cls} - mini-batch size, and N_{box} - number of anchors. Note that when $M = 1$, this refers to the standard RPN loss in the Faster R-CNN (eq. 2.17). Similarly, the loss for the ROI layer, now becomes a sum over M input/output pairs.

During training, each input is being sampled independently. However, during testing, the input is repeated M times so that M possibly different outputs are obtained for **the same input image**. It was empirically shown in the task of image classification that each of the M outputs provides good accuracy on its own and that the results are diverse enough, which allows them to be efficiently combined. In practice, $M = 2$ is often used. For more complex tasks, the network capacity does not allow processing a larger number of images in parallel. This agrees with the literature on model compression, which shows that usually modest compression rates (up to 50%) are achievable when using structured pruning (removing whole filters) [122, 125].

After M sets of results are obtained, they need to be efficiently combined. A standard approach in object detection to reduce redundant boxes is the NMS algorithm, which clusters together detections with high overlap, and keeps only detections with high confidence. However, such a procedure might be non-optimal when combining predictions from different models. Recently, the Weighted Boxes Fusion (WBF) [162] method was proposed. It efficiently combines different predictions by updating the final bounding box coordinates by using the confidence-weighted average of coordinates forming a cluster. Additionally, the final confidence score is also an average of all boxes forming a cluster. In this work, both aggregation methods

MOST WIEDZY Downloaded from mostwiedzy.pl

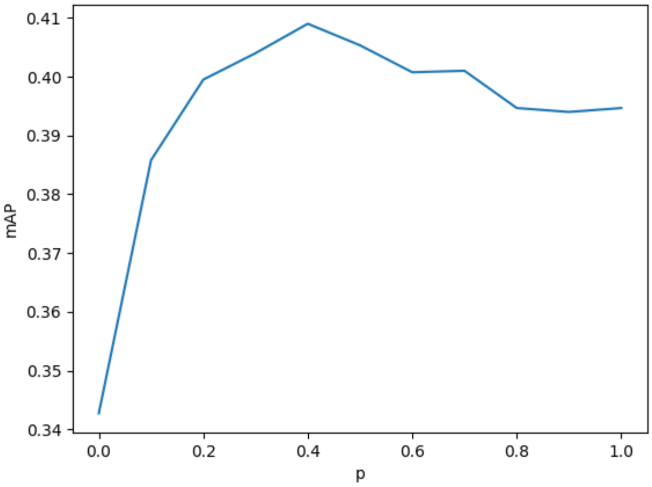


Figure 6.2: Mean average precision metric (mAP) on Cityscapes dataset as a function of probability p that the same images are sampled, when the model is trained with $M = 2$ input/output pairs.

(NMS and WBF) are evaluated.

6.2 Experiments

Implementation details. In the experiments, the MMDetection framework [120] was used. For the Cityscapes experiments, the models were trained for 64 epochs, using SGD optimizer, with an initial learning rate of 0.01 and a learning rate step reduction by a factor of 10 at epoch 48, similar as in section 4. All models were trained on a single GPU (Tesla V100) using a batch size of 6. During the training standard vision-based augmentations are applied: horizontal flipping and random resize. All of the models were pre-trained on ImageNet [7], as it is a standard in the community. For BDD and COCO datasets, the training lasted for 12 epochs, with learning rate reductions at epochs 8 and 11. Results are reported on the held-out validation sets.

The color jittering data augmentation was applied using the Albumentations library [163] with default parameters and the following transformations: random changes in brightness, contrast, saturation, and hue. In addition, style-transfer data augmentation was also used for some models to improve the diversity of the ensemble approach. For the style-transfer data augmentation, the same strategy was applied as in previous chapters.

Standard MIMO architecture struggles with fitting more subnetworks, especially on more challenging datasets. For example, in [26] the authors found that when training a ResNet-50 [164] classifier on the ImageNet dataset with $M = 2$, the model performed worse than a baseline. It was hypothesized that this happens

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 6.1: Accuracy and computational cost of different methods.

Model	mAP	num. params.	inf. time
Baseline	0.386	41.384M	0.088
MIMO (M=2)	0.409	41.397M	0.102
Deep Ensemble (M=2)	0.406	82.768M	0.176

when the main network does not have sufficient capacity to correctly classify two independent images at once. To improve that, the authors proposed relaxing independence between the inputs and added another hyperparameter p, which defines the probability that the networks use the same data during training. Namely, when $p = 0$, both images are sampled independently, and when $p = 1$, the training images are the same. As a result, in the first experiments on the Cityscapes dataset, a model with $M = 2$ input/output pairs was trained, and the p parameter was varied to see how it affected the final model performance (Fig. 6.2).

At $p = 0$ the inputs were fully independent, however, the final performance is limited by the network capacity. As the p grew, the subnetworks used the same image during training (with p probability), which allowed some of the features to be shared, which improved the performance. The performance peaked at $p = 0.4$ and then is slightly decreased. It is similar to that described in [26] when using ResNet-50 for the ImageNet classification task. As a result, further experiments were performed using $M = 2$, and $p = 0.4$.

Further, the results are compared with the standard Faster R-CNN model and Deep Ensemble approach (also consisting of $M = 2$ models) (Table 6.1). First, the MIMO Faster R-CNN outperformed a single model, improving the mAP score from 0.386 to 0.409. It also slightly outperformed the Deep Ensemble model. Importantly, the MIMO model brought only a slight increase in the parameters compared to the standard Faster R-CNN (from 41.38M to 41.4M). Inference time (as measured on a Tesla V-100 GPU) has increased by 15.9% (from 88ms to 102 ms per image). Note that the increase in inference time was very small when applying the MIMO framework to the image classification task (around 1%) [26]. For object detection, a larger increase in the processing time is attributed to M-times larger number of proposal regions being processed and the additional aggregation method. However, the processing time was still significantly shorter when compared to the Deep Ensemble method.

It is important to note that starting the training with ImageNet weights was crucial for the Cityscapes dataset (for all models). Additionally, when training the MIMO Faster R-CNN, one must also copy the ImageNet to the new filters (in the first channel). For the Deep Ensemble approach, the WBF aggregation method provided better results, yielding an improvement in the mAP score of 0.01, over the NMS approach. Overall, the WBF method performed the same or better than the NMS method, and all of the results were achieved using WBF aggregation. Experiments were also conducted on the m-heads architecture [158], in which the

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 6.2: Models' accuracy and calibration using different models and augmentation methods. Last two columns present results for corrupted Cityscapes. CJ stands for the color jittering augmentation and DE for the Deep Ensemble.

Model	mAP	ECE	c-mAP	c-ECE
Baseline	0.386	0.066	0.106	0.113
CJ	0.388	0.064	0.124	0.115
MIMO ($M = 2$)	**0.409**	0.045	**0.172**	0.075
MIMO ($M = 2$) + CJ	0.408	**0.04**	**0.172**	**0.071**
DE ($M = 2$, baseline)	0.406	0.068	0.116	0.124
DE ($M = 2$, CJ)	0.408	0.062	0.134	0.112
DE (MIMO, *M=2*)	**0.426**	0.05	0.184	0.087
DE (MIMO+CJ, *M=2*)	0.425	**0.046**	**0.186**	**0.068**
DE (baseline, *M=5*)	0.417	0.078	0.122	0.129
DE (CJ + style, *M=5*)	0.421	0.075	0.139	0.114

backbone was shared and the RPN and ROI nets were doubled. However, such an approach resulted in poor performance (mAP score of 0.378) and a larger increase in the number of parameters (55.9M). Such poor performance might be a result of the non-optimal structure of the proposed m-heads approach, and as such other variants could be explored.

6.2.1 Robustness and uncertainty

In this section, further experiments are described which focus on robustness and uncertainty estimation. First, it can be noted that the accuracy in the o.o.d. setting is severely impacted (Table 6.2), as was previously shown in the literature. For the baseline model, the accuracy on the corrupted version of the dataset was equal to 0.106. However, the accuracy was significantly improved when using the MIMO approach (0.172), outperforming Deep Ensemble by a large margin (0.116).

Further, the impact of adding color jittering data augmentation was measured. As expected, it improved the accuracy of the baseline model in the o.o.d. setting. On the other hand, the MIMO approach did not result in significant changes, except for slightly improving model calibration. Furthermore, deep Ensemble also benefited from the added data augmentation, but it lacks the robustness of the MIMO approach (e.g., 0.134 mAP score in the o.o.d. testing, compared to the 0.172 of the MIMO approach). Finally, when using color jittering, no significant changes were observed when measuring the impact of the p-value on the final accuracy (as in Fig. 6.2), and as a result, $p = 0.4$ was further used.

It can also be noticed that the MIMO approach provided the best classification calibration results above of the tested models. The ECE score on the clean dataset equaled 0.045 (compared to 0.066 of the baseline model) and was further reduced to 0.042 when color jitter data augmentation was used. The ECE in the o.o.d. setting was again the lowest out of the evaluated methods, significantly outperforming the

MOST WIEDZY Downloaded from mostwiedzy.pl

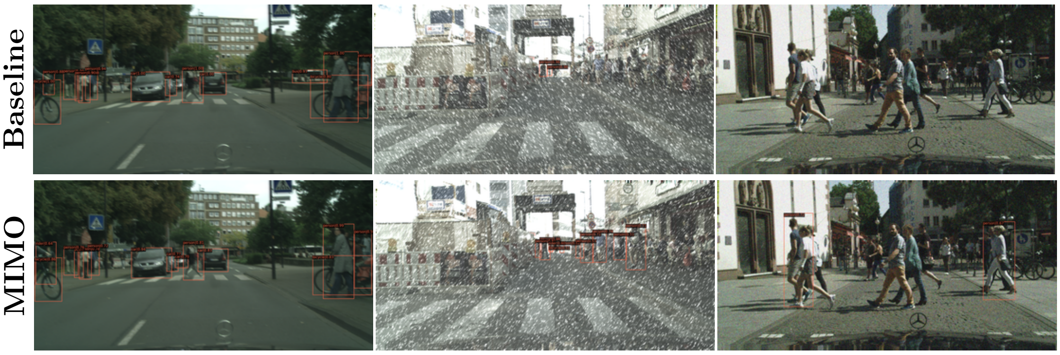


Figure 6.3: Detection results for the baseline and MIMO Faster R-CNN on different distortion types (motion blur, snow effect, Gaussian noise in the consecutive columns). Note, smaller confidence values for the MIMO model (i.e., 1st column). MIMO model performs on par or better than the standard model, however the corruptions vulnerability remains challenging (3rd column). Best viewed in digital format.

Deep Ensemble approach.

A potential critique of the evaluated Deep Ensemble approach is that a very small ensemble size was used and that the models' diversity is limited. As such, the ensemble method was also tested when 5 models were used, which was shown in the previous chapter to provide good results already. Additionally, one of the ensembles consisted of models of which some used color jittering data augmentation, and some used style-transfer data augmentation, to improve ensemble diversity. That setting allowed the Deep Ensemble approach to obtain very competitive results (Table 6.2, bottom part). The mAP increased to 0.421 and 0.139 on the clean and corrupted versions of the Cityscapes datasets, respectively (note that the accuracy in the o.o.d. is still worse than when using the MIMO approach). It was also checked whether using a MIMO Faster R-CNN models ensemble could further improve the results. When combining the outputs from two MIMO Faster R-CNN models, an impressive mAP of 0.426 was obtained on the clean dataset, and 0.184 on the corrupted version, which is an improvement over the Deep Ensemble approach consisting of 5 models. The usage of color jittering has improved the model calibration. Overall, these results further confirm the robustness of the MIMO Faster R-CNN model. Sample detection is presented in Fig. 6.3.

It is also interesting to look at the accuracy of the MIMO Faster R-CNN when using only one output. In such a scenario, model accuracy equals 0.405, which is a 0.004 drop compared to the full MIMO approach, but it is still a significant improvement over the baseline (0.386). This shows that the MIMO framework acts as a strong regularizer during training, which leads to strong feature representations.

The proposed method was also evaluated on the BDD dataset (Table 6.3). The model was trained using daytime images only and evaluated on daytime and nighttime images in this setting. Overall, compared to the standard training, the MIMO approach improved the accuracy on the clean daytime images from 0.293 to

MOST WIEDZY Downloaded from mostwiedzy.pl

Table 6.3: Accuracy on BDD Dataset when training on daytime images. $M = 2$ was used.

Model	BDD-day	BDD-night
Baseline	0.293	0.233
CJ	0.293	0.237
MIMO ($p = 0.7$)	0.301	0.244
MIMO + CJ ($p = 0.7$)	0.3	0.241
DE (baseline)	0.3	0.24
DE (CJ)	**0.302**	**0.246**

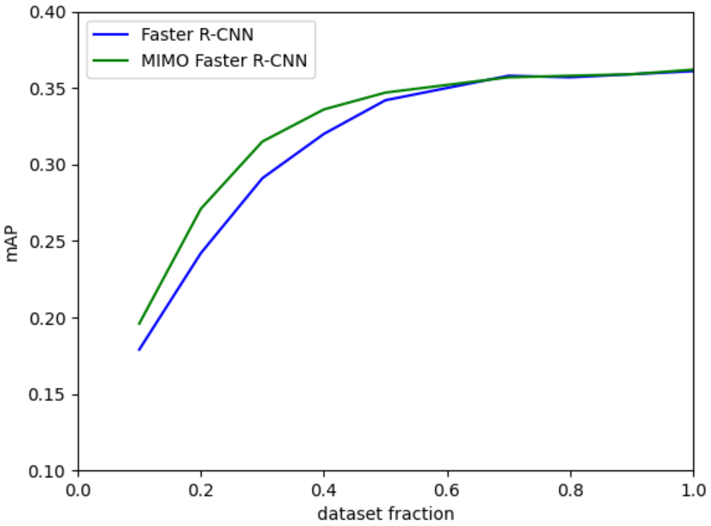


Figure 6.4: Accuracy of the standard and MIMO-based Faster R-CNN on the COCO dataset when using only a fraction of the training dataset.

0.301 and nighttime images (o.o.d. test) from 0.233 to 0.244. The probability p of sampling the same images during training also had to be further increased to observe improvements when using the MIMO model. This might be because a BDD is more challenging and includes a larger and more diverse set of images than the Cityscapes dataset. In that setting, the results of the MIMO approach are very similar to those obtained by Deep Ensemble. Looking at the single model accuracy within the MIMO method, it was found that it achieved almost the same accuracy (0.3 mAP value) as the full model. However, since the BDD dataset is very challenging, and the probability p of sampling the same images had to be increased, the outputs from single channels are no longer diverse, limiting the accuracy of the MIMO framework for this dataset. Using a larger backbone (ResNet-101), provided a similar increase over the standard model.

When evaluating the model on the COCO dataset no gains in accuracy were observed. Given the hypothesis that significant gains of the MIMO approach come from the regularization property, it should work better when using only the fraction of the training dataset. In fact, such an observation was made, and it was shown

MOST WIEDZY Downloaded from mostwiedzy.pl

that the MIMO approach was, in particular, useful in the low data regime (Fig. 6.4). Each model was trained for the same number of steps. MIMO framework was, in particular, effective when less than 50% of the training dataset was used, e.g., when using 30% of the data, using the MIMO approach improved the accuracy from 0.291 to 0.315 of the mAP score.

6.2.2 Discussion

The experiments showed that the MIMO approach could significantly improve accuracy compared to the standard training when using just $M = 2$ input/output pairs. Further, using just one output from the MIMO approach significantly gains the model accuracy. The MIMO approach was especially effective when using a fraction of the original dataset on the COCO dataset. Given those observations, we conjecture that training the model in multi-input multi-output works as a very strong regularizer, which allows the model to build a robust feature representation, and therefore even when using a small M, the model can work very well. A similar observation was made in the literature for structured pruning, which showed that model compression can also work as a regularizer [125]. This finding adds some more context to the original MIMO paper [26], which attributed its excellent performance mainly to ensembling diverse predictions.

It was also interesting to note that using specific data augmentation (for example, style transfer, color jittering) was not essential for the MIMO model to significantly improve the accuracy in the out-of-distribution setting. Again, this might indicate that using such texture-invariant data augmentation is unnecessary for the model to increase its robustness when its build representation is strongly regularized. This can be viewed as a complementary finding to a recent work [165], which shows that the increased shape bias (using the data above augmentations) does not necessarily improve model robustness.

A potential drawback of the MIMO approach is that when the task or dataset is especially challenging, it requires increasing the probability p of sampling the same images during training. This reduces the diversity between model outputs and diminishes the potential gains of having multiple outputs. The presented results could be further improved. For example, it was shown that using batch repetition during training for the MIMO framework has improved the results [26, 166], however, this came at the cost of significantly increased training time. Also, no specific optimization of the hyperparameters for MIMO was performed.

6.3 Conclusions

This chapter showed that using a multi-input multi-output approach can be generalized to object detection on real-world datasets. The MIMO Faster R-CNN model presented very competitive results in terms of model accuracy, uncertainty

MOST WIEDZY Downloaded from mostwiedzy.pl

calibration, and-out-of distribution robustness when using only $M = 2$ input/output pairs. The model adds only 0.5% of model parameters and increases the inference time by 15.9%. Similar accuracy can also be obtained when using the Deep Ensemble approach, but it requires a more significant number of models (and a significantly higher computational cost). The MIMO approach works as a regularizer during training, which significantly increases the accuracy of a single subnetwork compared to the standard training. A current limitation of the MIMO framework is that when the target dataset or task is challenging, the probability p of sampling the same image during training must be increased, limiting the diversity of the MIMO outputs. Some optimizations to the MIMO framework would be an interesting future work on scaling that approach to such a setting.

MOST WIEDZY Downloaded from mostwiedzy.pl

Chapter 7

Conclusions and Outlook

This thesis has explored different approaches to building efficient and robust visual recognition systems through a series of publications. This section summarizes the main results and discusses directions for future work.

7.1 Model Compression

In section 4, model compression methods are explored for the task of object detection. It has been shown in the literature that even 90% of neurons could be removed without hurting model accuracy [122]. Although we are interested in finding efficient algorithms for visual recognition in this thesis, studying methods that reduce computational cost is of great interest. In this thesis, unstructured (removing neurons) and structured (removing whole filters) pruning were studied. While unstructured pruning allows for high compression rates, it requires specialized hardware to decrease the computational cost. In the case of the Cityscapes dataset, for structured pruning, it was possible to remove around 40% of the filters without hurting model accuracy. This result shows that structured pruning is an attractive solution when one aims to reduce the computational cost. However, as was shown in section 3, measuring model accuracy using only the i.i.d. test is not enough and as a result, other measures of model accuracy were also utilized.

For that purpose, an o.o.d. benchmark, namely Cityscapes-C [79], was used. It was found that compression methods disproportionately increase the model vulnerability to different corruption types. Some of the corruptions are heavily impacted by the compression methods (i.e., additive noise), while others (blur effect) are only slightly affected. It was shown that the sensitivity of compressed models to different distortion types is nuanced and should be taken into account when deploying the model to the real world.

Naturalistic data augmentation (color jittering, noise, and blur distortions) was used to improve compressed models' robustness. The experiments section showed that, despite limited capacity, thanks to the added data augmentation, the

MOST WIEDZY Downloaded from mostwiedzy.pl

compressed models could significantly improve their robustness (even at the highest compression rates). It is worth noting that using data augmentation is a common theme across all publications in this thesis. The author believes that substantial data augmentation is a crucial component of modern visual systems, which is often omitted in publications. While in section 3, a style-transfer data augmentation was used, in section 4, even much simpler data augmentations (color jittering, noise, and blur distortions) were shown to be very effective for improving models' accuracy. In section 5, it was also shown that such data augmentation is a strong baseline in the domain adaptation setting.

In further experiments, we took a closer look at the per-class accuracy. This is another important characteristic of the models that is often omitted. Indeed, it was found that compression techniques have a disproportionate impact on different classes, even at moderate compression rates. It was hypothesized that one of the reasons for this is data imbalance, a compressed model (with smaller capacity) will firstly remove neurons responsible for the recognition of less-common data. The experimental section found that using data balancing methods helped improve the accuracy of some classes, and the effect was more striking at higher compression rates for structured pruning, which confirmed the first thesis of this work.

As can be noticed, careful evaluation and results analysis is an important part of this section, which is a common theme across all this thesis. There is no single benchmark to measure model robustness, but different approaches were studied, including cross-dataset evaluations, day-to-night-time transitions, and synthetic distortions. Additionally, in section 4, per-class accuracy was measured. The conducted experiments showed that it is crucial to test algorithms using diverse evaluations. For example, in section 3, it was demonstrated that CutOut data augmentation achieves only mediocre performance on some o.o.d. tests while having good accuracy on i.i.d. tests. Similarly, in section 4, compression methods were shown to hurt accuracy for noise-related distortions significantly, and some classes were shown to be more vulnerable to model compression.

One of the main limitations of the presented work is that a flat compression rate was used across layers; the same ratio compressed each layer. As has been shown in the literature, later layers in the networks have a greater compression capability [126]. As such, the presented compression results could be further improved. The presented work aimed to show the effect of compression methods at different compression rates (rather than achieving the best possible accuracy). As there is so single heuristic that obtains the best results for a given desired compression rate, the author decided to use a flat compression rate.

Also, some of the experiments were conducted at quite aggressive compression rates (for example, a 70% compression rate for structured pruning). Of course, it is improbable to use such a high compression factor, as it reduces model accuracy. However, the author believes that adding such experiments is worthwhile as it increases our understanding of how neural networks work.

MOST WIEDZY Downloaded from mostwiedzy.pl

7.2 Model ensembling

While the previous section focused on the robustness of models with low computational cost, model ensembles are investigated in section 5, known because of their remarkably high accuracy, which comes at a significantly increased computational cost. The first goal of the research was to find how well model ensembling copes with distributional shift for semantic segmentation, which was missing in the literature. The second goal was transferring the knowledge from the model ensemble to a single model.

Semantic segmentation models are studied in the domain adaptation setting under varying levels of the distributional shift to answer the first research question. One scenario assumed cross-dataset evaluation (Cityscapes to BDD dataset), and the other assumed a particularly challenging simulation to real transfer. When creating a model ensemble, inducing diversity between the models is a crucial [144]. In our case, this was achieved by using models which used different backbones and/or augmentation methods. The experimental section showed that a constructed ensemble improved over a single model in all scenarios and in the challenging simulation scenario to real adaptation.

Further, it was observed that uncertainty calibration improved in the distributional shift scenario, which was an important finding. Firstly, well-calibrated uncertainty estimation is important for models operating in a safety-critical environment. Secondly, it can be efficiently utilized in the self-learning approach; an idea that was followed up in further experiments.

A self-learning approach was utilized to efficiently transfer knowledge from the model ensemble and improve single model accuracy in the target domain. It uses a model trained on the source domain to generate labels on the target domain. Then, those labels (usually filtered using model confidence) are used to fine-tune a baseline model. Using a model ensemble, in this case, allowed the author to correctly label a greater number of pixels in the target domain. The predictions, in general, were better calibrated, which is essential for the self-learning approach. This produced a significant gain in the final model accuracy, compared to the standard fine-tuning (without ensemble), and confirmed the second thesis of this work. Additionally, it was found (as in another section of the thesis) that using substantial data augmentation is crucial for improving final accuracy.

One of the limitations of the work is a lack of a detailed ablation study. While it was shown that using diverse ensembles consisting of models with different backbones and/or augmentation methods resulted in high accuracy, the effect of each of those interventions on final model accuracy is unknown. One of the main challenges when working with the model ensemble is the computational cost, as each experiment requires training M models (where M is the ensemble size). As the computational budget was a significant constraint in this project, the author followed a strategy that was expected to be close to optimal.

MOST WIEDZY Downloaded from mostwiedzy.pl

While model ensemble significantly improved final accuracy in the domain adaptation setting, it did not improve the accuracy of the classes for which the source model was already inferior. That is because there were fewer high-confidence predictions for those classes. Improving the accuracy of less common classes is a known problem with the self-learning approach (and in machine learning in general) and is important for future work.

7.3 Multi-input multi-output framework

Section 6 integrates the findings from two previous sections. In section 4, it was shown that modern CNN-based visual recognition models are over-parametrized, and careful strategies exist which allow them to be efficiently compressed (to some extent). On the other hand, in section 5, model ensembling is a powerful strategy for visual recognition, even under the large distributional shift between training and testing domains. Those findings motivated the research question of whether those two methods can be efficiently combined.

The starting point for the experiments is the recently introduced Multi-input Multi-output architecture, which was shown to work very well for the image classification task. It works simply by running multiple predictions (for many images) at the same time. This is possible since current CNN-based models are over-parametrized. During inference, all images are the same, and as a result, several potentially different predictions are obtained for a single image. In the thesis, a design for MIMO architecture in the object detection task was proposed and implemented. A proposed model was evaluated on three datasets, and it showed a significant improvement over the single model with a slight increase in the computational cost, which confirmed the third thesis of this work. Overall, the results (including out-of-distribution accuracy and uncertainty calibration) were similar to model ensembling, with a much smaller computational cost.

The detailed analysis showed that one of the reasons for improved performance is that training the model in the MIMO framework works as strong regularization. Thus, when using just one output, this observation is similar to compression methods that have been shown to increase the model accuracy at low compression rates.

An essential limitation of the work is that only two ($M = 2$) input-output pairs were used in the experimental section, similar to the literature [26]. This is because, when using a larger number of input-output pairs the model does not have sufficient capacity to provide accurate prediction for all images at once. Additionally, each dataset has its characteristics. For example, the independence between input images during training was relaxed for a challenging BDD dataset. With the probability $p = 0.7$, both images were the same. That reduced the output diversity and potential benefits coming from having multiple predictions.

As a result, it was concluded that a MIMO approach is undoubtedly a very efficient way to train CNN-based visual recognition models. It works as a strong

MOST WIEDZY Downloaded from mostwiedzy.pl

regularizer during training and provides more than one output for the same image. However, the independence between input images must be relaxed for more challenging datasets and tasks, limiting the method's efficiency. This is an important future work on how to effectively train MIMO models so that they provide diverse predictions, including for more challenging datasets. Also, allowing using an M greater than two would be beneficial.

7.4 Closing remarks

In this thesis, the author set a goal of finding methods for training efficient and robust models for visual recognition that could work well when deployed to the real world and potentially faced with several situations that never occurred in the training dataset. One of the methods that turned out to be effective was model ensembling, which aggregates results from many potentially diverse predictions. This improved the model accuracy and uncertainty calibration, which is essential for models operating in the real world. But a challenge remained: how to make ensembling work with much smaller computational cost. Compression methods inspired the solution, and as a result, it was shown that it is possible to obtain more than one prediction for the same image by using a specific training procedure. Nevertheless, it is expected that more work is possible in the area of efficient ensembling.

An essential aspect of that work was testing models out-of-distribution, which showed that the evaluated models still lack the required robustness. Out-of-distribution accuracy will play a crucial role in tracking progress that is made toward robust, visual-based recognition models. Certainly, new methods will be developed here that will bring machine vision closer to that of human perception. For the autonomous driving industry, extra sensors (LiDAR, depth cameras) will be used to support visual-based recognition models.

In spite of the recent progress, the generalization of trained models to the real world is often unsatisfying. Certainly, further methods will be developed in this area and the problem is being tackled from different perspectives. It is an exciting research area and the future will show which methods will turn out to be the most effective.

MOST WIEDZY Downloaded from mostwiedzy.pl

Bibliography

[1] D. Silver *et al.*, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] T. B. Brown *et al.*, “Language models are few-shot learners,” in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS, virtual*, Dec. 2020.

[3] A. Esteva *et al.*, “Dermatologist-level classification of skin cancer with deep neural networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[4] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Boston, USA*, pp. 815–823, June 2015.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25, Lake Tahoe, USA*, pp. 1106–1114, Dec. 2012.

[6] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain.,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[7] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[8] R. McAllister, Y. Gal, A. Kendall, M. van der Wilk, A. Shah, R. Cipolla, and A. Weller, “Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, Melbourne, Australia*, pp. 4745–4753, Aug. 2017.

[9] C. Michaelis *et al.*, “Benchmarking robustness in object detection: Autonomous driving when winter is coming,” in *Machine Learning for Autonomous Driving Workshop, NeurIPS*, July 2019.

[10] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Colorado Springs, USA*, pp. 1521–1528, IEEE Computer Society, June 2011.

MOST WIEDZY Downloaded from mostwiedzy.pl

[11] S. Beery, G. V. Horn, and P. Perona, "Recognition in terra incognita," in *15th European Conference European Conference on Computer Vision, Munich, Germany, Proceedings, Part XVI*, vol. 11220, pp. 472–489, Springer, Sept. 2018.

[12] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 5389–5400, PMLR, 2019.

[13] J. Jo and Y. Bengio, "Measuring the tendency of cnns to learn surface statistical regularities," *preprint, arXiv:1711.11561*, 2017.

[14] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," in *7th International Conference on Learning Representations, ICLR, New Orleans, USA*, May 2019.

[15] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Advances in Neural Information Processing Systems, Vancouver, Canada*, pp. 125–136, Dec. 2019.

[16] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever, "Deep double descent: Where bigger models and more data hurt," in *8th International Conference on Learning Representations, ICLR, Addis Ababa, Ethiopia*, April 2020.

[17] R. Geirhos *et al.*, "Shortcut learning in deep neural networks," *Nat. Mach. Intell.*, vol. 2, no. 11, pp. 665–673, 2020.

[18] F. C. Borlino, A. D'Innocente, and T. Tommasi, "Rethinking domain generalization baselines," in *25th International Conference on Pattern Recognition, ICPR, Virtual / Milan, Italy*, Jan. 2021.

[19] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2, NIPS, Denver, USA*, pp. 598–605, Nov. 1989.

[20] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, USA*, pp. 2704–2713, June 2018.

[21] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, USA*, June 2018.

[22] S. Hooker, A. Courville, G. Clark, Y. Dauphin, and A. Frome, "What do compressed deep neural networks forget?," *preprint arXiv:1911.05248*, 2019.

MOST WIEDZY Downloaded from mostwiedzy.pl

[23] V. Feldman and C. Zhang, "What neural networks memorize and why: Discovering the long tail via influence estimation," in *Advances in Neural Information Processing Systems 33, NeurIPS, virtual*, Dec. 2020.

[24] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[25] Q. Xie, M. Luong, E. H. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, USA*, pp. 10684–10695, June 2020.

[26] M. Havasi *et al.*, "Training independent subnetworks for robust prediction," in *International Conference on Learning Representations, ICLR, Virtual*, May 2021.

[27] M. Wortsman *et al.*, "Supermasks in superposition," in *Annual Conference on Neural Information Processing Systems, NeurIPS, virtual*, Dec. 2020.

[28] B. Cheung, A. Terekhov, Y. Chen, P. Agrawal, and B. A. Olshausen, "Superposition of many models into one," in *Annual Conference on Neural Information Processing Systems, NeurIPS, Vancouver, Canada*, Dec. 2019.

[29] R. Geirhos *et al.*, "Partial success in closing the gap between human and machine vision," in *Advances in Neural Information Processing Systems 35, virtual*, Dec. 2021.

[30] S. Dodge and L. Karam, "A study and comparison of human and deep learning recognition performance under visual distortions," in *26th international conference on computer communication and networks (ICCCN), Vancouver, Canada*, pp. 1–7, IEEE, Aug. 2017.

[31] R. Geirhos, C. R. M. Temme, J. Rauber, H. H. Schütt, M. Bethge, and F. A. Wichmann, "Generalisation in humans and deep neural networks," in *Advances in Neural Information Processing Systems 31, NeurIPS, Montréal, Canada*, pp. 7549–7561, Dec. 2018.

[32] K. P. Murphy, *Machine learning : a probabilistic perspective.* MIT Press, 2013.

[33] K. Hornik, M. B. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[34] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, pp. 947–951, June 2000.

[35] H. J. Kelley, "Gradient theory of optimal flight paths," *ARS j.*, vol. 30, pp. 947–954, Oct. 1960.

MOST WIEDZY Downloaded from mostwiedzy.pl

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[37] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[38] D. J. Felleman and D. C. Van Essen, “Distributed Hierarchical Processing in the Primate Cerebral Cortex,” *Cerebral Cortex*, vol. 1, pp. 1–47, 01 1991.

[39] O. S. Kayhan and J. C. van Gemert, “On translation invariance in cnns: Convolutional layers can exploit absolute spatial location,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, Seattle, USA*, pp. 14262–14273, PMLR, June 2020.

[40] J. F. Canny, “A computational approach to edge detection.,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[41] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision - ECCV - 13th European Conference, Zurich, Switzerland, Proceedings, Part I*, vol. 8689, pp. 818–833, Sept. 2014.

[42] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba, “Object detectors emerge in deep scene cnns,” in *3rd International Conference on Learning Representations, ICLR, San Diego, USA, Conference Track Proceedings*, May 2015.

[43] K. He, R. Girshick, and P. Dollar, “Rethinking ImageNet Pre-Training,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, South Korea*, IEEE, Oct. 2019.

[44] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016.

[45] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[46] D. Yin, R. G. Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, “A fourier perspective on model robustness in computer vision,” in *Advances in Neural Information Processing Systems, Vancouver, Canada*, pp. 13255–13265, Dec. 2019.

[47] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, “Exploring the landscape of spatial robustness,” in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, (Long Beach, USA), pp. 1802–1811, PMLR, June 2019.

[48] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems, Montreal, Canada*, pp. 91–99, Dec. 2015.

MOST WIEDZY Downloaded from mostwiedzy.pl

[49] P. A. Viola and M. J. Jones, “Rapid object detection using a boosted cascade of simple features.,” in *CVPR, Kauai USA*, pp. 511–518, IEEE Computer Society, Dec. 2001.

[50] Z. Cai and N. Vasconcelos, “Cascade R-CNN: delving into high quality object detection,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, USA,*, pp. 6154–6162, June 2018.

[51] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” *Computer Vision and Pattern Recognition, CVPR 2005*, vol. 1, pp. 886–893.

[52] P. Felzenszwalb, D. Mcallester, and D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” *Computer Vision and Pattern Recognition, CVPR*, June 2008.

[53] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587, June 2014.

[54] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, “Selective search for object recognition,” *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, 2013.

[55] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, USA*, pp. 779–788, June 2016.

[56] W. Liu *et al.*, “Ssd: Single shot multibox detector.,” in *ECCV (1), Amsterdam, The Netherlands*, vol. 9905, pp. 21–37, Springer, Oct. 2016.

[57] R. Girshick, “Fast R-CNN,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015.

[58] N. Otsu, “A Threshold Selection Method from Gray-level Histograms,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[59] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis.,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[60] C. Rother, V. Kolmogorov, and A. Blake, “Grabcut: interactive foreground extraction using iterated graph cuts.,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[61] E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional networks for semantic segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, 2017.

MOST WIEDZY Downloaded from mostwiedzy.pl

[62] S. Cygert, B. Wróblewski, K. Woźniak, R. Slowiński, and A. Czyżewski, "Closer look at the uncertainty estimation in semantic segmentation under distributional shift," in *International Joint Conference on Neural Networks (IJCNN), virtual*, IEEE, July 2021.

[63] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI, Munich, Germany, Proceedings, Part III*, vol. 9351, pp. 234–241, Springer, Oct. 2015.

[64] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *3rd International Conference on Learning Representations, ICLR, San Diego, USA, Conference Track Proceedings*, May 2015.

[65] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *The European Conference on Computer Vision (ECCV), Munich, Germany*, Sept. 2018.

[66] Y.-H. Liao, A. Kar, and S. Fidler, "Towards good practices for efficiently annotating large-scale image classification datasets," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), virtual*, IEEE, June 2021.

[67] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *IEEE Conference on Computer Vision and Pattern Recognition, Miami, USA*, pp. 304–311, IEEE, June 2009.

[68] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robotics Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[69] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, USA*, pp. 3213–3223, June 2016.

[70] S. Zhang, R. Benenson, and B. Schiele, "Citypersons: A diverse dataset for pedestrian detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3213–3221, 2017.

[71] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, "Eurocity persons: A novel benchmark for person detection in traffic scenes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1844–1861, 2019.

[72] L. Neumann *et al.*, "Nightowls: A pedestrians at night dataset," in *Asian Conference on Computer Vision, Perth, Australia*, pp. 691–705, Springer, Dec. 2018.

MOST WIEDZY Downloaded from mostwiedzy.pl

[73] F. Yu *et al.*, “BDD100K: A diverse driving dataset for heterogeneous multitask learning,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, USA*, pp. 2633–2642, IEEE, June 2020.

[74] P. Sun *et al.*, “Scalability in perception for autonomous driving: Waymo open dataset,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, USA*, IEEE, June 2020.

[75] H. Caesar *et al.*, “nuscenes: A multimodal dataset for autonomous driving,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2020.

[76] T. Lin *et al.*, “Microsoft COCO: common objects in context,” in *Computer Vision - ECCV - 13th European Conference, Zurich, Switzerland, Proceedings, Part V*, vol. 8693, pp. 740–755, Springer, Sept. 2014.

[77] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *International Conference on Learning Representations, Banff, Canada*, April 2014.

[78] A. Demontis *et al.*, “Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks,” in *28th USENIX Security Symposium, USENIX Security, Santa Clara, USA*, pp. 321–338, USENIX Association, Aug. 2019.

[79] D. Hendrycks and T. G. Dietterich, “Benchmarking neural network robustness to common corruptions and perturbations,” in *7th International Conference on Learning Representations, ICLR, New Orleans, USA*, May 2019.

[80] K. Gu, B. Yang, J. Ngiam, Q. Le, and J. Shlens, “Using videos to evaluate image model robustness,” *Safe Machine Learning workshop at ICLR*, 2019.

[81] A. Barbu *et al.*, “Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models,” in *Advances in Neural Information Processing Systems, Vancouver, Canada*, pp. 9453–9463, Dec. 2019.

[82] B. Carter, S. Jain, J. Mueller, and D. K. Gifford, “Overinterpretation reveals image classification model pathologies,” in *Advances in Neural Information Processing Systems 35, NeurIPS, virtual*, Dec. 2021.

[83] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Las Vegas, USA*, pp. 2414–2423, IEEE Computer Society, June 2016.

[84] T. Devries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” *preprint, arXiv:1708.04552*, 2017.

MOST WIEDZY Downloaded from mostwiedzy.pl

[85] E. Rusak *et al.*, “A simple way to make neural networks robust against diverse image corruptions,” in *Computer Vision – ECCV*, (Cham), pp. 53–69, Springer International Publishing, 2020.

[86] R. G. Lopes, D. Yin, B. Poole, J. Gilmer, and E. D. Cubuk, “Improving robustness without sacrificing accuracy with patch gaussian augmentation,” *preprint, arXiv:1906.02611*, 2019.

[87] D. Mahajan *et al.*, “Exploring the limits of weakly supervised pretraining,” in *Computer Vision - ECCV - 15th European Conference, Munich, Germany, Proceedings, Part II*, vol. 11206, pp. 185–201, Springer, Sept. 2018.

[88] I. Hasan, S. Liao, J. Li, S. U. Akram, and L. Shao, “Generalizable pedestrian detection: The elephant in the room,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR virtual*, pp. 11328–11337, Computer Vision Foundation / IEEE, June 2021.

[89] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, “Using self-supervised learning can improve model robustness and uncertainty,” in *Advances in Neural Information Processing Systems, Vancouver, Canada*, pp. 15637–15648, Dec. 2019.

[90] P. W. Koh *et al.*, “WILDS: A benchmark of in-the-wild distribution shifts,” in *Proceedings of the 38th International Conference on Machine Learning, ICML, Virtual*, vol. 139, pp. 5637–5664, PMLR, July 2021.

[91] R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt, “Measuring robustness to natural distribution shifts in image classification,” in *Advances in Neural Information Processing Systems 33, NeurIPS, virtual*, Dec. 2020.

[92] T. DeVries, I. Misra, C. Wang, and L. van der Maaten, “Does object recognition work for everyone?,” in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops, Long Beach, USA*, pp. 52–59, Computer Vision Foundation / IEEE, June 2019.

[93] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang, “Distributionally robust neural networks,” in *8th International Conference on Learning Representations, ICLR, Addis Ababa, Ethiopia*, April 2020.

[94] N. S. Sohoni, J. Dunnmon, G. Angus, A. Gu, and C. Ré, “No subclass left behind: Fine-grained robustness in coarse-grained classification problems,” in *Advances in Neural Information Processing Systems 33, NeurIPS, virtual*, Dec. 2020.

[95] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, 2015.

MOST WIEDZY Downloaded from mostwiedzy.pl

[96] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International Journal of Computer Vision*, vol. 128, pp. 261–318, Oct. 2019.

[97] B. Cheng, R. B. Girshick, P. Dollár, A. C. Berg, and A. Kirillov, "Boundary iou: Improving object-centric image segmentation evaluation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, virtual*, pp. 15334–15342, Computer Vision Foundation / IEEE, June 2021.

[98] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70, Sydney, Australia*, pp. 1321–1330, JMLR. org, August 2017.

[99] J. Snoek *et al.*, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift," in *Advances in Neural Information Processing Systems, Vancouver, Canada*, pp. 13969–13980, Dec. 2019.

[100] S. Cygert and A. Czyżewski, "Toward robust pedestrian detection with data augmentation," *IEEE Access*, vol. 8, pp. 136674–136683, 2020.

[101] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *IEEE International Conference on Computer Vision, ICCV, Santiago, Chile*, pp. 1026–1034, IEEE Computer Society, Dec. 2015.

[102] D. Dai and L. Van Gool, "Dark model adaptation: Semantic image segmentation from daytime to nighttime," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA*, pp. 3819–3824, IEEE, Dec. 2018.

[103] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy*, pp. 1501–1510, Oct. 2017.

[104] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. P. Breckon, and B. Obara, "Style augmentation: Data augmentation via style randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, USA*, pp. 83–92, June 2019.

[105] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE International Conference on Computer Vision, Seoul, South Korea*, pp. 6023–6032, Oct. 2019.

[106] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data processing method to improve robustness and uncertainty," in *8th International Conference on Learning Representations, ICLR, Addis Ababa, Ethiopia*, April 2020.

MOST WIEDZY Downloaded from mostwiedzy.pl

[107] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *Proceedings of the IEEE conference on computer vision and pattern recognition, Long Beach, USA*, pp. 113–123, June 2019.

[108] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *6th International Conference on Learning Representations, ICLR, Vancouver, Canada, Conference Track Proceedings*, April 2018.

[109] D. Hendrycks, K. Lee, and M. Mazeika, "Using pre-training can improve model robustness and uncertainty," *Safe Machine Learning workshop at ICLR*, 2019.

[110] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, 2020.

[111] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen, "Repulsion loss: Detecting pedestrians in a crowd," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, USA*, pp. 7774–7783, June 2018.

[112] W. Liu, S. Liao, W. Hu, X. Liang, and X. Chen, "Learning efficient single-stage pedestrian detectors by asymptotic localization fitting," in *The European Conference on Computer Vision (ECCV), Munich, Germany*, Sept. 2018.

[113] X. Huang, Z. Ge, Z. Jie, and O. Yoshie, "Nms by representative region: Towards crowded pedestrian detection by proposal pairing," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), virtual*, June 2020.

[114] W. Liu, S. Liao, W. Ren, W. Hu, and Y. Yu, "High-level semantic feature detection: A new perspective for pedestrian detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, USA*, pp. 5187–5196, June 2019.

[115] K. Piniarski, P. Pawłowski, and A. Dąbrowski, "Improved pedestrian detection by adjustment of segmented roi in thermal night vision," in *2020 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), Poznań, Poland*, pp. 92–97, Sept. 2020.

[116] K. Piniarski, P. Pawłowski, and A. Dąbrowski, "Tuning of classifiers to speed-up detection of pedestrians in infrared images," *Sensors*, vol. 20, p. 4363, Aug. 2020.

[117] S. Cygert and A. Czyzewski, "Style transfer for detecting vehicles with thermal camera," in *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), Poznań, Poland*, pp. 218–222, Sept. 2019.

MOST WIEDZY Downloaded from mostwiedzy.pl

[118] D. Feng, Y. Cao, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging uncertainties for deep multi-modal object detection in autonomous driving," in *IEEE Intelligent Vehicles Symposium, IV, Las Vegas, USA*, pp. 877–884, IEEE, Oct. 2020.

[119] S. Y. Duck, "Painter by numbers," *available online https://www.kaggle.com/c/painter-by-numbers, accessed 8.12.2021*, 2016.

[120] K. Chen *et al.*, "Mmdetection: Open mmlab detection toolbox and benchmark," *preprint, arXiv:1906.07155*, 2019.

[121] S. Kohl *et al.*, "A probabilistic u-net for segmentation of ambiguous images," in *Advances in Neural Information Processing Systems, Montréal, Canada*, pp. 6965–6975, Dec. 2018.

[122] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems, Montreal, Canada*, vol. 28, pp. 1135–1143, Dec. 2015.

[123] S. Cygert and A. Czyzewski, "Robustness in compressed models for object detection," in *International Joint Conference on Neural Networks (IJCNN), virtual*, IEEE, July 2021.

[124] M. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *6th International Conference on Learning Representations, ICLR, Workshop Track Proceedings, Vancouver, Canada*, May 2018.

[125] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Advances in Neural Information Processing Systems 29, Barcelona, Spain*, pp. 2074–2082, Dec. 2016.

[126] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *5th International Conference on Learning Representations, ICLR, Toulon, France*, April 2017.

[127] C. H. Tu, J. H. Lee, Y. M. Chan, and C. S. Chen, "Pruning depthwise separable convolutions for mobilenet compression," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2020.

[128] G. Tzelepis, A. Asif, S. Baci, S. Cavdar, and E. E. Aksoy, "Deep neural network compression for image classification and object detection," in *18th IEEE International Conference On Machine Learning And Applications, ICMLA*, pp. 1621–1628, 2019.

[129] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *5th International Conference on Learning Representations, ICLR, Toulon, France*, April 2017.

MOST WIEDZY Downloaded from mostwiedzy.pl

[130] R. Entezari and O. Saukh, “Class-dependent compression of deep neural networks,” *preprint, arXiv:1909.10364*, 2020.

[131] L. Liebenwein, C. Baykal, B. Carter, D. Gifford, and D. Rus, “Lost in pruning: The effects of pruning neural networks beyond test accuracy,” in *Proceedings of Machine Learning and Systems 2021, MLSys 2021*.

[132] T. Gale, E. Elsen, and S. Hooker, “The state of sparsity in deep neural networks,” *preprint, arXiv:1902.09574*, 2019.

[133] K. L. Hermann, T. Chen, and S. Kornblith, “The origins and prevalence of texture bias in convolutional neural networks,” in *Advances in Neural Information Processing Systems 33, NeurIPS, virtual*, Dec. 2020.

[134] M. Buda, A. Maki, and M. A. Mazurowski, “A systematic study of the class imbalance problem in convolutional neural networks,” *Neural Networks*, vol. 106, pp. 249–259, 2018.

[135] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” *J. Artif. Int. Res.*, vol. 16, p. 321–357, June 2002.

[136] C. Huang, Y. Li, C. C. Loy, and X. Tang, “Learning deep representation for imbalanced classification,” in *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA*, pp. 5375–5384, IEEE Computer Society, June 2016.

[137] Y. Cui, M. Jia, T. Lin, Y. Song, and S. J. Belongie, “Class-balanced loss based on effective number of samples,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Long Beach, USA*, pp. 9268–9277, June 2019.

[138] A. Gupta, P. Dollár, and R. B. Girshick, “LVIS: A dataset for large vocabulary instance segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Long Beach, USA*, pp. 5356–5364, June 2019.

[139] Y. Li *et al.*, “Overcoming classifier imbalance for long-tail object detection with balanced group softmax,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, Seattle, USA*, pp. 10988–10997, June 2020.

[140] N. Zmora, G. Jacob, L. Zlotnik, B. Elharar, and G. Novik, “Neural network distiller: A python package for dnn compression research,” *preprint, arXiv:1910.12232*, 2019.

[141] J. Platt *et al.*, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” *Advances in large margin classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

MOST WIEDZY Downloaded from mostwiedzy.pl

[142] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33nd International Conference on Machine Learning, ICML, New York City, USA*, vol. 48, pp. 1050–1059, JMLR, June 2016.

[143] F. K. Gustafsson, M. Danelljan, and T. B. Schön, "Evaluating scalable bayesian deep learning methods for robust computer vision," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops, virtual*, June 2020.

[144] B. Neyshabur, H. Sedghi, and C. Zhang, "What is being transferred in transfer learning?," in *Advances in Neural Information Processing Systems 30 (NeurIPS), virtual*, Dec. 2020.

[145] A. Abramov, C. Bayer, and C. Heller, "Keep it simple: Image statistics matching for domain adaptation," in *CVPR workshop on Scalability in Autonomous Driving*, 2020.

[146] M. Kim and H. Byun, "Learning texture invariant representation for domain adaptation of semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, USA*, pp. 12975–12984, June 2020.

[147] Y. Zou, Z. Yu, V. Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *The European Conference on Computer Vision (ECCV), Munich, Germany*, vol. 11207, pp. 297–313, Springer, Sept. 2018.

[148] S. James *et al.*, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, USA*, pp. 12627—12637, June 2019.

[149] H. Zhang, Y. Tian, K. Wang, H. He, and F. Wang, "Synthetic-to-real domain adaptation for object instance segmentation," in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2019.

[150] A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Advances in Neural Information Processing Systems, Montréal, Canada*, pp. 3235—3246, Dec. 2018.

[151] S. Cygert and A. Czyżewski, "Vehicle detection with self-training for adaptative video processing embedded platform," *Applied Sciences*, vol. 10, p. 5763, Aug. 2020.

[152] A. Mehrtash, W. M. Wells, C. M. Tempany, P. Abolmaesumi, and T. Kapur, "Confidence calibration and predictive uncertainty estimation for deep medical

MOST WIEDZY Downloaded from mostwiedzy.pl

image segmentation," *IEEE Transactions on Medical Imaging*, vol. 39, no. 12, pp. 3868–3878, 2020.

[153] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, USA*, pp. 535–541, ACM, 2006.

[154] C. Liu *et al.*, "Progressive neural architecture search," in *Computer Vision - ECCV - 15th European Conference, Munich, Germany*, pp. 19–35, Sept. 2018.

[155] G. Zhou, S. Dulloor, D. G. Andersen, and M. Kaminsky, "EDF: ensemble, distill, and fuse for easy video labeling," *preprint arXiv:1812.03626.*

[156] Y. Wen, D. Tran, and J. Ba, "Batchensemble: an alternative approach to efficient ensemble and lifelong learning," in *8th International Conference on Learning Representations, ICLR, Addis Ababa, Ethiopia*, April 2020.

[157] S. Cygert and A. Czyżewski, "Robust object detection with multi-input multi-output faster r-cnn," *preprint, arXiv:2111.13065*, 2021.

[158] S. Lee, S. Purushwalkam, M. Cogswell, D. J. Crandall, and D. Batra, "Why M heads are better than one: Training a diverse ensemble of deep networks," *preprint arXiv:1511.06314*, 2015.

[159] A. Ashukha, A. Lyzhov, D. Molchanov, and D. P. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," in *8th International Conference on Learning Representations, ICLR, Addis Ababa, Ethiopia*, April 2020.

[160] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in neural information processing systems, Long Beach, USA*, pp. 6402–6413, Dec. 2017.

[161] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *7th International Conference on Learning Representations, ICLR, New Orleans, USA*, May 2019.

[162] R. A. Solovyev, W. Wang, and T. Gabruseva, "Weighted boxes fusion: Ensembling boxes from different object detection models," *Image Vis. Comput.*, 2021.

[163] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020.

[164] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA*, June 2016.

MOST WIEDZY Downloaded from mostwiedzy.pl

[165] C. K. Mummadi, R. Subramaniam, R. Hutmacher, J. Vitay, V. Fischer, and J. H. Metzen, “Does enhanced shape bias improve neural network robustness to common corruptions?,” in *Proceedings of the 38th International Conference on Machine Learning, ICML, virtual*, May 2021.

[166] A. Ramé, R. Sun, and M. Cord, “Mixmo: Mixing multiple inputs for multiple outputs via deep subnetworks,” in *IEEE International Conference on Computer Vision, ICCV, virtual*, Oct. 2021.

MOST WIEDZY Downloaded from mostwiedzy.pl