Rodrigo Ferreira Berriel

On the Efforts on Training Deep Neural Networks:

Reducing Computational and Data-related Costs

Vitória, ES

2021

Rodrigo Ferreira Berriel

On the Efforts on Training Deep Neural Networks:

Reducing Computational and Data-related Costs

Tese de Doutorado submetida ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Doutor em Ciência da Computação.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Programa de Pós-Graduação em Informática

Supervisor: Prof. Dr. Thiago Oliveira dos Santos

Vitória, ES

2021

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Ferreira Berriel, Rodrigo, 1992-
F383o On the efforts on training deep neural networks : Reducing computational and data-related costs / Rodrigo Ferreira Berriel. - 2021.
123 f. : il.

Orientador: Thiago Oliveira dos Santos.
Tese (Doutorado em Informática) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Inteligência artificial. 2. Redes neurais (Computação). 3. Machine learning. 4. Aquisição de dados. 5. Complexidade computacional. I. Oliveira dos Santos, Thiago. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 004

On the Efforts on Training Deep Neural Networks: Reducing Computational and Data-related Costs

Rodrigo Ferreira Berriel

Tese submetida ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

Aprovada em 28 de setembro de 2021:

Prof. Dr. Thiago Oliveira dos Santos
Orientador

Prof. Dr. Alberto Ferreira De Souza
Membro Interno

Profª. Drª. Claudine Santos Badue Gonçalves
Membro Interno

Prof. Dr. Francisco de Assis Boldt
Membro Externo

Prof. Dr. Jurandy Gomes de Almeida Junior
Membro Externo

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
Vitória/ES, 28 de setembro de 2021.

To mom and dad, who taught me what really matters.

Acknowledgements

I would like to thank God and my family (Luci, *in memoriam* Jorge, and Bruno) for providing me with the support and solid ground I needed throughout this journey.

I would like to thank my advisor, Prof. Dr. Thiago Oliveira dos Santos, for helping me become the researcher I am today. Thank you for the many lessons, the long conversations, and your availability and understanding. Were it not for you and your guidance, this whole experience would not have been this amazing.

I would like to thank all the friends I made along this journey. I would like to thank everyone from our lab (LCAD) for the many coffee breaks, random conversations, and joint works. Thanks! Special thanks to Prof. Dr. Nicu Sebe and his team (MHUG) from Trento (Italy) for having received me so well during my one-year stay and collaboration.

Lastly, I would like to thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 for the scholarship, NVIDIA for some of the GPUs used in this research, and Microsoft for the *Microsoft Azure for Research Award*.

"An unspeakable horror seized me. There was a darkness; then a dizzy, sickening sensation of sight that was not like seeing; I saw a Line that was no Line; Space that was not Space: I was myself, and not myself. When I could find voice, I shrieked aloud in agony, "Either this is madness or it is Hell." "It is neither," *calmly replied the voice of the Sphere,* "it is Knowledge; it is Three Dimensions; open your eye once again and try to look steadily."

I looked, and, behold, a new world!"

– Edwin Abbot, "Flatland," (1884)

Resumo

Atualmente, redes neurais profundas (DNNs, do inglês *Deep Neural Networks*) estão em todos os lugares. Muitos dos nossos problemas diários têm sido resolvidos ou amenizados com a ajuda de tais modelos. No entanto, é importante ressaltar que o sucesso desses modelos se baseia em algumas inconveniências: eles requerem muitos dados e grande poder computacional. Para alcançar o desempenho necessário para usá-los no mundo real, frequentemente são necessárias bases de dados massivas para treiná-los. Essas bases de dados, por sua vez, requerem a coleta e anotação de muitas amostras. Ambos processos, coleta e anotação de dados, são caros e demandam muito tempo, i.e., eles requerem muito investimento em sensores, tempo e pessoas em uma série de tarefas eventualmente entediantes e propensas a erros. Nessa tese nós exploramos maneiras de reduzir os esforços necessários para treinar redes neurais profundas, especialmente em relação aos problemas mencionados. Nós investigamos meios de usar dados que já estão disponíveis na internet para reduzir os custos de aquisição de dados. Além disso, nós também estudamos maneiras de reduzir o custo computacional em tempo de inferência, particularmente na perspectiva da aprendizagem de múltiplos domínios. Nossos resultados mostram aplicações da coleta e anotação automática em grande escala que provaram ser úteis no treinamento de redes convolucionais profundas para ambos problemas de classificação e regressão no mundo real. Adicionalmente, nós propusemos um modelo que possui baixa complexidade computacional, requer menos armazenamento e possui um baixo consumo de memória quando comparado com as alternativas da literatura, demonstrando uma forma eficaz de reduzir os requisitos computacionais de redes profundas. Por fim, nossas propostas mostraram que os esforços envolvidos no treinamento de redes profundas podem ser reduzidas ainda mais, principalmente no que tange aos custos computacionais e aos relacionados a dados.

Palavras-chaves: aprendizagem profunda, aquisição de dados, anotação de dados, aprendizagem de múltiplos domínios.

Abstract

Nowadays, deep neural networks (DNN) are ubiquitous. Many of our daily problems have been solved or alleviated with the help of such models. It is important to note, though, that the success of these models is built on top of some drawbacks: they are data hungry and computationally intensive. In order to achieve the performance required for real-world usage, it is often necessary to train these DNNs in massive data sets. These data sets, in turn, require the acquisition and annotation of a lot of samples. Both these processes, data acquisition and annotation, are expensive and time-demanding, i.e., they require a lot of investment in sensors, time, and people in a series of sometimes tedious tasks that are error-prone. In this thesis, we explore ways of reducing the effort on training deep neural networks, especially towards the aforementioned problems. We investigate how to leverage data readily available online to reduce the data acquisition cost. Moreover, we propose to fuse data from multiple online services to reduce the data annotation cost. Furthermore, we also study ways of reducing the computational cost on inference time, particularly on the perspective of multi-domain learning. Our results show many applications of automatic large-scale data acquisition and annotation that proved to be useful for training deep convolutional networks for both classification and regression real-world problems. In addition, we proposed a model that has a low computation complexity, requires lower storage, and has a low memory footprint compared to existing alternatives, demonstrating an effective way of reducing the computational requirements of deep networks. Finally, our proposals showed that the efforts in training deep networks can be reduced even further, particularly when it comes to data-related and computational costs.

Keywords: deep learning, data acquisition, data annotation, multi-domain learning.

List of Figures

List of Tables

List of abbreviations and acronyms

ADAS	Advanced Driver Assistance Systems
API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
CNN	Convolutional Neural Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DNN	Deep Neural Network
FLOP	Floating-point Operation
GPS	Global Positioning System
GPU	Graphics Processing Unit
GSV	Google Street View
HOG	Histogram of Gradients
IARA	Intelligent Autonomous Robotic Automobile
IMU	Inertial Measurement Unit
LBP	Local Binary Pattern
LBPH	Local Binary Pattern Histogram
LCAD	Laboratório de Computação de Alto Desempenho
LiDAR	Light Detection And Ranging
MAE	Mean Absolute Error
MDL	Multi-Domain Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
OSM	OpenStreetMap
SGD	Stochastic Gradient Descent

SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TPU	Tensor Processing Unit

Contents

1 Introduction

Nowadays, deep neural networks (DNN) are ubiquitous. They can be found everywhere in the literature, from applications on Computer Vision (TAN; LE, 2019), to Autonomous Driving (CHEN et al., 2015), to Natural Language Processing (VASWANI et al., 2017) and much more. Not only that, many companies have started to develop products using this technology, and they are already shipping these products to the broader market. In fact, many of our daily problems have been solved or alleviated with the help of such models. Our smartphones are filled with examples, from your personal phone assistant, to that chat bot from your bank app that saves you some time, to that search engine in your photo app that helps you find that picture that you took from your dog years ago. These technologies are not only being delivered, but they are actually helping us out. It is important to note, though, that the success of these models is built on top of some drawback features: they are data hungry and computationally intensive.

In order to achieve the performance required for real-world usage, and actually deliver some value, it is often necessary to train these DNNs in massive data sets. These data sets, in turn, require the acquisition and annotation of a lot of samples. Both these processes, data acquisition and annotation, are expensive and time-demanding. Initially, one needs to define the scope of the data that is required to train the models. Then, often it is necessary to invest a lot of money in equipment (e.g., sensors, hardware, etc.) to enable the acquisition of such data. Not only that, but more often than not, trained personnel is also required, which means, one has to allocate a group of specialized people to go out to the world, or bring whatever is need in to the lab, and acquire the data. Note that some tasks require data that are inherently scarce (e.g., car accidents, rare diseases, etc.) and acquiring such data may require more than just money and will. Finally, the acquired data have to be organized and annotated. These processes generally require humans and are repetitive and tedious, and humans are error-prone when performing repetitive and tedious tasks. Moreover, some of these tasks may also require very well-trained professionals, people that are not always available and willing to perform such tasks (e.g., doctors to annotate imagery).

Acquiring and annotating massive datasets is only the beginning. Actually training these deep neural networks requires a lot of computational power. In fact, the modern era of deep learning is happening in part because of the advances in hardware, particularly GPUs. For effectively and efficiently training these very large deep neural networks, sometimes multiple GPUs in multiple nodes are needed. However, maintaining this infrastructure alone requires a tremendous effort from trained professionals, which is not something all research groups and companies can afford. Having the expertise to develop and train these

networks is already expensive. In addition, the deployment of these models presents a whole new set of limitations. In many real-world situations, these models will compete for resources in a very limited environment (limitations can be various, such as storage, memory, power, etc.), and they still need to keep their performance at a certain level.

In this thesis, we explore ways to reduce the effort in training deep neural networks, especially towards the aforementioned problems. Our hypothesis is that data readily available online and a generic image representation can be exploited to increase the sustainability of deep neural networks. To validate that, we investigate how to leverage data freely available on web platforms to reduce the data acquisition cost. Moreover, we propose to fuse data from multiple online services to reduce the data annotation cost. Finally, we also study ways to reduce the computational cost on inference time, particularly from the perspective of multi-domain learning.

1.1 Objectives

The overall goal of this thesis is to explore ways to reduce the effort in training deep neural networks. There are several ways to tackle this problem. Below, we pinpoint the specific goals that may help us moving in this direction:

- Investigate and develop techniques that drastically reduce the investments required in the data acquisition process and expedite the process as a whole;
- Investigate and contribute with techniques that enable the removal of humans from the loop (data acquisition and annotation) and improve the annotation process in terms of cost and time;
- Investigate and develop models with reduced computational requirements at inference time. One should be able to change these models at inference time to fit the computational budget that is currently available.

Most of these investigations took place in the context of computer vision and autonomous driving. This context is mostly because of the laboratory in which this thesis was developed, the High Performance Computing Lab (LCAD, from *Laboratório de Computação de Alto Desempenho*, in Portuguese). In our lab, we develop a self-driving car, the IARA (BADUE et al., 2021), and the computer vision problems related to it present an excellent opportunity to develop new ideas and deploy new solutions like the ones we are interested in.

1.2 Contributions

On exploring the aforementioned objectives, many contributions were made. The contributions listed below are a brief non-exhausting summary from the four papers selected to support this thesis:

- Techniques and methods to exploit imagery available through online platforms to develop models for real-world applications (e.g., worldwide crosswalk detection) with little to no human effort involved in the data acquisition and annotation processes;
- A method to train models that have adjustable (reduced) computational requirements at inference time, which is particularly useful in multi-domain problems but has shown to also be valuable for a single domain;
- Datasets (with more than 35,000 annotated images), pre-trained models, and source codes available at repositories in <https://github.com/rodrigoberriel>;
- Four published papers: Berriel et al. (2017b), IEEE Geoscience and Remote Sensing Letters, Qualis A2; Berriel et al. (2017), Elsevier Computers & Graphics, Qualis A2; Berriel et al. (2018), IJCNN, Qualis A1; and Berriel et al. (2019), ICCV, Qualis A1.

In addition to the aforementioned contributions, I had the opportunity to collaborate in other projects together with other people from our lab. Among these projects, two of them are directly connected to our line of research, but were not included in this thesis. In (ARRUDA et al., 2019), we proposed to use Generative Adversarial Networks to translate images from day (source domain) to night (target domain) and re-use the annotations that are available in source domain but scarce in the target. We showed that a model can be successfully trained using the generated images, thus removing the cost of annotating the images of the target domain. In (TORRES et al., 2019), we proposed an approach to remove almost all the costs associated with data acquisition and annotation in the problem of traffic sign detection. Our approach combines templates of traffic signs and arbitrary natural images that are freely available on the web or in general object recognition benchmarks to create a dataset that is later used to train a deep traffic sign detector. In addition to reducing the associated costs, the proposed approach helps dealing with the inherent problem of data imbalance that would happen with normal data acquisition (some traffic signs are rarer than others).

Moreover, many other research opportunities have appeared during the development of this thesis and they have allowed us to make several additional contributions that were not presented here. In fact, this thesis is a particular line of research resulting from 4 selected papers out of more than 20 ones that were published in internationally renowned conferences (e.g., ICCV, CVPR, etc.) and journals (e.g., Pattern Recognition, Expert

Systems with Applications, etc.) in our field. These papers have also allowed us to publicly release various applications, methods, models, and annotated datasets for the research community. A list of publications can be found in Appendix A. We really hope our contributions have helped others to push our field further and further, even if for just a little bit.

1.3 Structure

This thesis is structured as follows:

- After this Introduction, Chapter 2 presents a brief overview of the theoretical background, such as the foundation of deep neural networks and the efforts involved in training them;
- In Chapter 3, we present how we achieved our first two objectives, which are the reduction of the investments in data acquisition and the acceleration of the annotation process, preferably removing humans from the loop. In this context, we introduce our investigation on automatic large-scale data acquisition and annotation, along with a brief review of the relevant literature. As a result, three published methods are described, together with the experiments and results that validate them;
- In Chapter 4, we address our last objective, which is to cut down the computational costs, preferably making them adjustable to the user's budget. In this context, our work on multi-domain learning is detailed: we introduce a method that allows the reduction of computation at inference time, and we report results on several experiments on both multi-domain and single domain problems;
- Finally, Chapter 5 recalls the main points of the thesis and the conclusions are presented.

2 Theoretical Background

Deep neural networks have been around for a while now. In this chapter, we present a brief overview of what deep neural networks are and how they work, and highlight aspects of it that are of particular interest for this thesis. Moreover, we provide additional background and references on some of the efforts required in training these deep neural networks to solve real-world problems.

2.1 Deep Neural Networks

The human brain has fascinated us – humans – for a very long time. Many researchers have tried and are still trying to model the human brain, but it was only in 1943 that we saw developments in the understanding of neural activity through the use of logic and computation (MCCULLOCH; PITTS, 1943). The McCulloch-Pitts neuron has many flaws, though. Pushing it further, Rosenblatt (1958) proposed the Perceptron, the first neural network to be algorithmically described. Much later, it would be proved (CYBENKO, 1989) that single hidden layer neural networks can approximate any continuous function, which is known as the universal approximation theorem. There are limitations, nonetheless. To increase the complexity that these models can represent, one can add mode layers with more non-linear activations. Despite its power, deep neural networks remain remarkably simple (Figure 1): the network receives inputs, processes them in hidden layers that accumulate and pass the intermediate outputs through (non-)linear activations, and finally outputs one or more values.

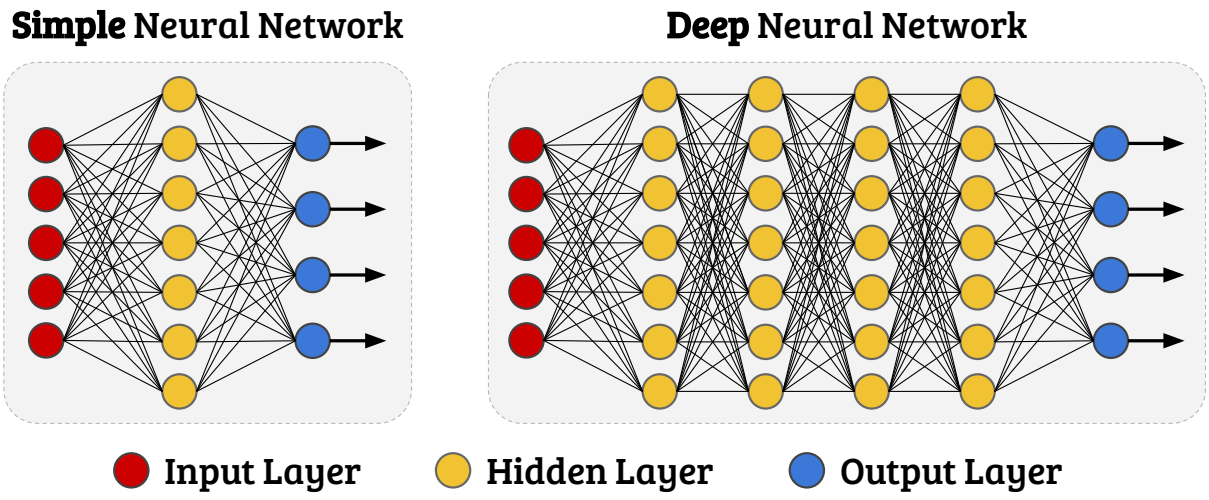


Figure 1 – Representation of Simple and Deep Neural Networks.

Although powerful, multi-layer perceptrons do not seem the most appropriate

modeling for dealing with images. Alongside the development of artificial neural networks, Hubel and Wiesel (1962) were studying visual neuroscience, and proposed a hierarchy model of the visual nervous system. This model was composed of "simple cells" and "complex cells", and they stated that a cell in a higher stage (farther from the input) generally has a larger receptive field and is more insensitive to shifts of the input stimuli. Inspired by this hierarchy model, Fukushima and Miyake (1982) proposed the Neocognitron, a neural network that self-organizes when presented with a set of stimulus patterns repeatedly. Among other things, the unsupervised learning approach used in the Neocognitron narrowed its applications. To push neural networks for vision applications even further, LeCun et al. (1989) proposed Convolutional Neural Networks (CNNs, Figure 2), which have convolutional and pooling layers that are directly inspired by the classic notions of simple cells and complex cells, and have their roots in the neocognitron.

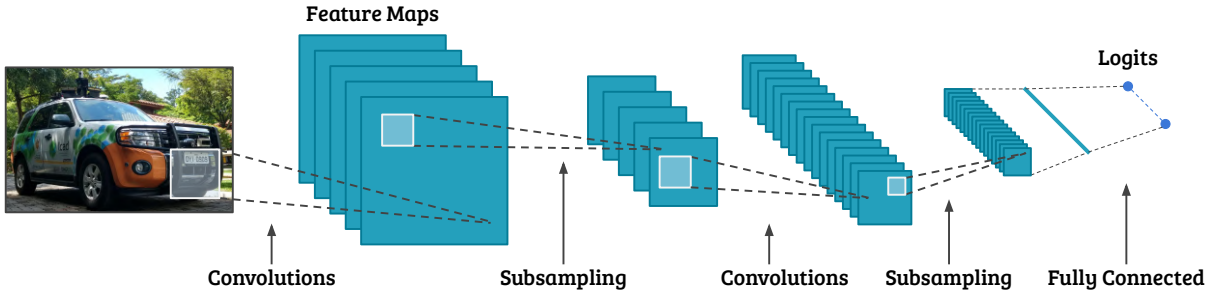


Figure 2 – Example of a Convolutional Neural Network (CNN) architecture

One of the main differences between ConvNets and Neocognitrons was how they "learned" the free parameters: Neocognitrons exploited unsupervised learning, whereas ConvNets had an end-to-end supervised-learning algorithm such as backpropagation (LINNAINMAA, 1970). Backpropagation is often referred to as the reverse mode of automatic differentiation, or as an "*efficient way of applying the chain rule to big networks with differentiable nodes*" (SCHMIDHUBER, 2015). In this algorithm, the free parameters (weights) are adjusted according to the derivatives of the output error, and the complexity of computing these derivatives is proportional to the number of weights (which was a major improvement over the previous alternatives).

Training these neural networks is an iterative process (Figure 3). They receive sample data as input (e.g., images of cats and dogs), process them through the hidden layers, and output a prediction (e.g., whether the image is from a cat or a dog). This prediction is compared with the label of the input image, and the weights are adjusted according to the error that is backpropagated. As it can be seen, the effectiveness of these models depends on the amount (and diversity) of data that is presented to the network. Therefore, to reach the demanding levels that real-world applications require, massive, large-scale data sets are needed. Training deep neural networks is no simple task. In the next section, we present some of the efforts required in this process, focusing on those that

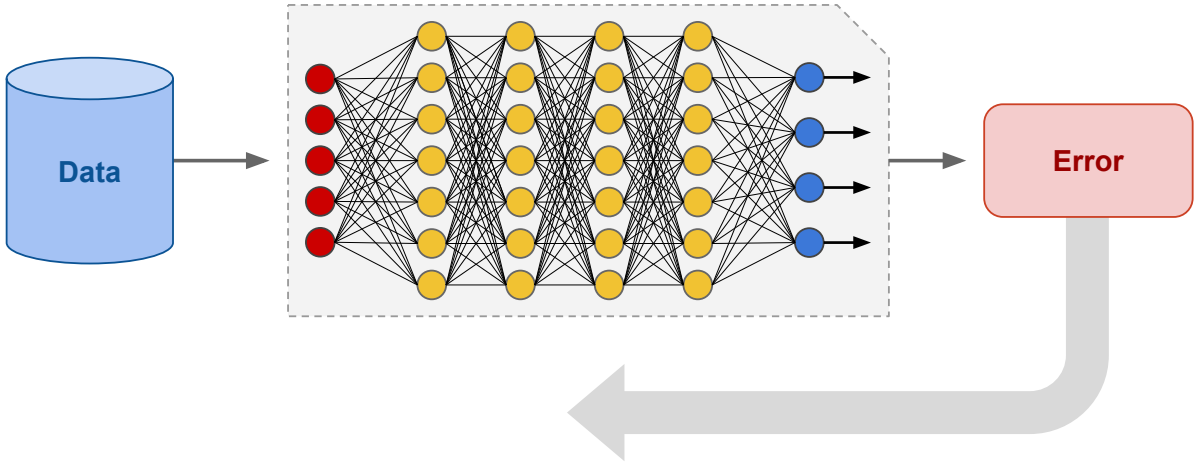


Figure 3 – Training deep neural networks with backpropagation.

are related to the scope of this thesis.

2.2 Efforts in Training Deep Neural Networks

There are several efforts involved in training deep neural networks to solve real-world problems. First, one needs to acquire a lot of data. As already described, collecting these massive data sets in the real world can be very expensive and, for some tasks, simply ineffective. After that, all these data need to be annotated. Annotating can also be very expensive, and tends to be a very tedious and error-prone task for humans. Moreover, trained experts have to develop models and code for training these deep neural networks. This is usually an iterative process and commonly requires a lot of time from expensive professionals. Together with the development of software, additional trained experts have to prepare and maintain the infrastructure require for these training sessions. Sometimes, training very large deep neural network requires multiple GPU in multiple nodes, and either maintaining that infrastructure locally or in the cloud is very expensive. To solve real-world problems, all this process may need to be iterated over and over, which shows how much human effort needs to be invested on training deep neural networks. In this section, we focus on three particular efforts: data acquisition, data annotation, and computation. These are the efforts of interest in the scope of this thesis, and they are presented with more details and in context below.

2.2.1 Data Acquisition

As already discussed, deep neural networks are data hungry. To satisfy this need and achieve good performance with deep models, researchers have to provide a lot of samples during training time. Ideally, these samples should resemble as much as possible with the environment in which the model is expected to be deployed. The process of getting

these samples can be named *data collection* or *data acquisition*. In this section, only the concepts and works more closely related to this thesis will be discussed. For a broader view on data acquisition systems, the reader is referred to (ABDALLAH; ELKEELANY, 2009).

In autonomous driving, collecting data is both expensive and difficult. In most cases, acquiring useful data involves mounting and calibrating cameras on cars, and equipping these cars with sensors and computers such that the required data can be captured. As can be noted, the process involves expensive resources and assets, such as vehicles, lasers, cameras (arguably cheaper nowadays), radars, and more (see Figure 4). Some of these sensors require specialized people to calibrate them such that the data collected are actually useful. In addition, some problems require data from inherently rare circumstances, and collecting them adds even more to the cost of the process. Moreover, to allow the models to handle some of the many edge cases in the driving context, one needs to collect data from a variety of situations that would or could damage the vehicle, sensors, and even harm the driver. Forcing such situations with real drivers is obviously not a choice, therefore there is a need for techniques to overcome this limitation as well.

Figure 4 – Some of the resources that can be required to acquire useful data: vehicle, LiDAR, IMU, GPU, cameras, radar, etc.

In a more general context, there are two main lines of research: the use of pictures available online and the use of simulators. The first one has been widely used in several image classification benchmarks. For instance, the well-known datasets CIFAR (KRIZHEVSKY, 2009), ImageNet (RUSSAKOVSKY et al., 2015), COCO (LIN et al., 2014), and many others were built on top of images crawled on the web. These images were then reviewed by humans and annotated into the categories of interest in each dataset. In the context of self-driving cars, simulators have been growing in interest, availability, and quality. While some simulators (KOSKELA et al., 2011; DOSOVITSKIY et al., 2017) have been made freely available for the research community, good and realistic ones require a lot of

investment. To circumvent that, many groups (KIM; PARK, 2017; YUE et al., 2018) have been leveraging the graphics developed to entertainment and using games such as *Grand Theft Auto* (GTA) to collect data (see Figure 5). Despite the effort on making them each time more realistic, models still have to handle the domain shift between the simulation and real-world.

(a) (b)

Figure 5 – Images from the simulator CARLA (a) and the game GTA V (b).

2.2.2 Data Annotation

Frequently, and in the context of this thesis, deep neural networks are trained in a supervised setting, i.e., at training time the model is presented with samples (e.g., images) and their corresponding labels. In this context, in addition to the collecting images, one needs to annotate them to make them useful. At times, the annotation of the data is a tedious and repetitive task. Therefore, to avoid spending the valuable and costly time of experts on tasks that may be trivial (e.g., annotating boxes around cars), many research groups have resorted to crowdsourcing as a solution to the annotation process. The most used crowdsourcing platform is the Amazon Mechanical Turk (Amazon, 2005). In this case, researchers can create tasks and distribute them to people all over the world (or to selected groups, if needed) using MTurk, usually paying the annotators a fraction of the cost of an expert's time. This solution create additional problems that have to be solved (e.g., to ensure quality, multiple people have to annotate the same sample), but all in all it tends to be a much cheaper and faster alternative than manually annotating massive data sets on a lab. However, MTurk is not fit for all problems and does not fit every pocket. Hence, many groups have developed in-house tools (see Figure 6) to help and reduce the time spent on annotating a dataset, some of them have even released these tools publicly (Berkeley DeepDrive, 2017; Microsoft, 2018). Moreover, some groups have proposed automatic and semi-automatic techniques to aid the annotation process, some of them including human-in-the-loop (WANG et al., 2018). Furthermore, there are several

works (OQUAB et al., 2015; SANGINETO et al., 2019) trying leverage cheaper forms of annotation while learning more complex tasks (e.g., using only classification annotation to perform object detection).

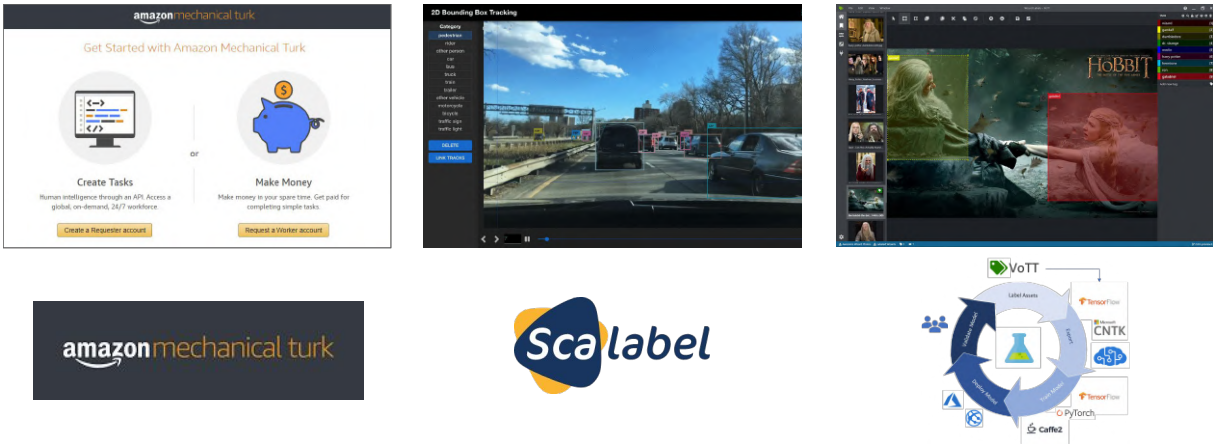


Figure 6 – Examples of a crowdsourcing marketplace (MTurk) and open-source tools (Scalabel and VoTT) commonly used for data annotation.

2.2.3 Computation

Another common limitation that practitioners and researchers encounter when deploying their models in the real-world is that there is a limited amount of computation available to their model, and they often have to compete with other requirements. This is particularly challenging in the autonomous driving context, in which a vehicle has to perform several tasks at the same time to ensure the safety of the passengers and the achievement of the goal. The computational limitation is particularly concerning because most of these tasks have been solved using DNNs, but they come at a high cost. These models require specialized hardware (e.g., GPUs, TPUs, custom ASICs; see Figure 7) for inference, and they often also demand large amounts of such hardware for training (many leading companies and universities have their own GPU clusters nowadays). To alleviate, there is an extensive research (HINTON; VINYALS; DEAN, 2014; POLINO; PASCANU; ALISTARH, 2018) being done on the model compression side (e.g., using distillation or quantization), but they usually reduce the computational requirement in a static fashion, i.e., they cannot adapt to dynamic computational budgets. Therefore, the problem of designing deep architectures which allow an adaptive accuracy-efficiency trade-off directly at runtime is of particular importance and has been recently addressed in the research community (WANG et al., 2018; WU et al., 2018; YU et al., 2019).

Figure 7 – Examples of accelerators (GPUs from NVIDIA, TPU from Google, and D1 from Tesla) and three of the fastest supercomputers with GPUs in the world (Selene from NVIDIA, Summit from ORNL, and Perlmutter from Berkeley).

2.3 Discussion

As already discussed, deep neural networks have indeed taken over many areas of research and are already powering many commercial applications. These advances, nonetheless, brought with them a set of new challenges and limitations. Humans are still heavily involved in the data-related processes like data acquisition and annotation. This dependence on human effort drastically increases the cost of these processes and the time it takes to complete them. Moreover, to achieve the level of performance required by certain real-world applications, the network architectures have grown more and more over time, requiring more computational power than ever. Despite the advances in both fronts (data-related and computational costs), there is still much more to do. In this thesis, we explore alternative to continue pushing these frontiers even further.

3 Automatic Large-scale Data Acquisition and Annotation

As discussed in the previous chapter, deep neural networks are data-intensive, and this fact poses various challenges for those who need to train and deploy DNN-based models in the real world. In this thesis, we investigate how to reduce or eliminate the need for human effort in the processes of large-scale data acquisition and annotation in the context of specific problems. We hypothesize that online platforms can be exploited to effortlessly acquire large amounts of data from around the world to solve specific problems with unprecedented performance. Furthermore, we also hypothesize that data from multiple platforms can be fused to reduce the inherent noise of automatic processes and remove humans altogether from the annotation step, enabling the acquisition of a dataset that is large and diverse enough to overcome its noisy nature. In this chapter, we first present an overview of the literature for the problems we investigated. Then, we introduce the methods we developed, and the experiments and results validating their efficacy. Finally, we discuss the main findings of our contributions and present the conclusion.

3.1 Literature Review

In this thesis, we investigate and propose techniques for automatic[1] large-scale data acquisition and annotation in the context of specific problems: crosswalk classification (based on satellite and streetview imagery) and heading direction estimation. In this section, an overview of the relevant literature is presented.

Satellite Crosswalk Classification. There are several approaches to the crosswalk classification problem in the literature. In the aerial (top-view) perspective, Herumurti et al. (2013) employed a circle mask template matching and SURF method to detect zebra crossing on aerial images. Their method was validated on a single region in Japan and the method that detects most crosswalks took 739.2 seconds to detect 306 crosswalks. Riveiro et al. (2015) developed an algorithm for automatic detection of zebra crossings from mobile LiDAR data. The algorithm comprises image segmentation followed by several image processing techniques. From a total of 30 crosswalks, 25 crosswalks were correctly classified (83.33%). The authors analyzed the missed crosswalks and the failures came from painting deterioration, very common on developing countries; and occlusion produced by

[1] In this thesis, we adopted the nomenclature used by the literature. Note, however, that the *automatic* processes used by the literature and in this thesis often require *crowd-sourced* data which are usually manually annotated by volunteers.

other vehicles, also very common in traffic. Ghilardi, Junior and Manssour (2016) proposed a model that used satellite imagery to perform crosswalk detection and localization in order to help the visually impaired to approach intersections. The best model uses a Local Binary Pattern (LBP) feature extraction method and a SVM classifier. The database comprises only 900 image patches (600 for the training and 300 for the testing) from 4 different cities. Koester, Lunt and Stiefelhagen (2016) proposed an SVM based on HOG and LBPH features to detect zebra crossings in aerial imagery. Although very interesting, their dataset (not publicly available) was gathered manually from satellite photos, therefore it is relatively small (3119 zebra crossings and $\approx$ 12500 negative samples) and local. In addition, their method shows a low generalization capability, because a model trained in one region shows low recall when evaluated in another (known as cross-based protocol), e.g., the recall goes from 95.7% to 38.4%.

Street View Crosswalk Classification. In the perspective of helping people with disabilities, specially the visually impaired people, using cameras available on the phones is a strategy commonly used. A prototype for a cell phone application was developed by Ivanchenko, Coughlan and Shen (2008) to help the visually impaired people. When approaching an intersection, the application detects crosswalks in the images (if any crosswalk is visible) and helps the user to align in the direction of the crosswalk to safely cross the streets. The feature extraction used in the application is based on simple image processing (edge detection using derivative filters). A limitation of this work is related to the scope of the dataset used in the evaluation: only 90 images with only 30 of them containing crosswalks. In addition, there are some thresholds that may require a tuning phase. Poggi, Nanni and Mattoccia (2015) also investigated this problem from the perspective of a pedestrian. The authors developed a wearable mobility aid system that captures RGB-D images. The RGB image is filtered based on the depth (point cloud) information to remove noise. Their system predicts 4 classes of crosswalks (considering different angles) and one class for the other (negative) cases. The authors captured about 2500 images to train a Convolutional Neural Network (approximately 500 for each class), and evaluated on a validation set of 10,165 frames. They reported an accuracy of 88.97%, and 91.59% after the head and pose refinement. As the authors neither described the process used while capturing the dataset nor the location where it was captured nor the diversity of the validation set, it is difficult to assess the quality of the results. Wang et al. (2014) also used RGB-D images to propose a wayfinding and navigation aid to improve autonomy of the visually impaired people. For the stair and crosswalk detection and recognition part, their system uses image processing on the RGB image and extract depth feature that are both used on a SVM classifier to distinguish stairs from crosswalks. Like many of the systems in the literature, their dataset is small (228 images with only 30 crosswalks) and local. The authors report an accuracy of 78.90% for the crosswalk class. Considering only the crosswalks and the negative class, the overall accuracy of the system

is of 90.98%.

Finally, Ahmetovic et al. (2015) presented a method combining two perspectives: aerial (satellite) and street view. Their system is targeted to blind travelers and comprises two steps. Initially, crosswalks are searched in the satellite imagery. Subsequently, only the crosswalks candidates detected for the satellite processing are further validated. Finally, Google Street View panoramas are iteratively acquired and the image is either confirmed as containing a crosswalk or rejected. The Google Street View method was evaluated in only 406 portions of Street View images, which comprises a limitation to the results. Moreover, only images from a single city (San Francisco, CA) were used, and the authors tuned the parameters of the system using images from the same region.

Heading Direction Estimation. There are several solutions proposed for lane detection that could be used for heading direction estimation, and the most common ones rely only on images. Berriel et al. (2017) propose a system that works on a temporal sequence of images and for each image, one at time, applies an inverse perspective mapping estimating the lane based on a combination of methods (Hough lines with Kalman filter and spline with particle filter). The final lane is represented as a cubic spline. Jung, Youn and Sull (2016) propose an aligned spatiotemporal image generated by accumulating the pixels on a scanline along the time axis and aligning consecutive scanlines. In that image, the trajectory of the lane points appears smooth and forms a straight line. Lee et al. (2017) built a dataset with 20,000 images with labeled lanes and trained a Convolutional Neural Network (CNN) to detect and track the lane. Lane detection methods could be used to compute the heading direction, but are generally dependent of labeled datasets (for image processing adjustments or CNNs training) that are expensive to generate.

Brahmbhatt and Hays (2017) collected a large-scale dataset of street-view images organized in a graph where nodes are connected by roads. They applied an A* algorithm to generate the labels and trained a CNN to reach a destination by deciding which direction to take at intersections. This method also uses an automatic data acquisition and labeling system, but it is more complex to generate the dataset labels because of the A* algorithm used in the processes. Another issue is that their CNN finds the path instead of the heading direction. Bojarski et al. (2016) trained a CNN to map a single front image to steering commands. Their system learned to drive on roads with or without lane markings, with traffic, unpaved roads, parking lots and highways. The data were acquired using their autonomous car. For each image the steering wheel angle, recorded while a human was driving, was used as the image label. Gupta et al. (2017) implemented a similar approach but only for indoor environment robots. This approach is the most similar to the one presented in this work, but the CNN predicts the commands for actuation in the robot instead of the heading direction. Another issue is that their approach depends on a robot to generate the dataset, and that is sometimes difficult and expensive to obtain.

In the set of solutions that rely on LIDAR data, Veronese et al. (2016) employ a particle filter localization by matching the online 2D projection of LIDAR point cloud with offline 2D projection. Using this technique, they are able to compute the car orientation in the world but they cannot find the road. Hernandez et al. (2015) estimate the heading angle by identifying lane marks on the road surface using the reflection of a laser system. They employed the DBSCAN, but they could not achieve good precision. The biggest disadvantages of working with LIDAR is their prohibitive cost (which is coming down over time) and they generally require additional drivers.

Summary. Despite the several advances made by the research community, there are still many issues to be addressed. For instance, the data-related processes (data acquisition and annotation) are still dependent of humans, which causes these processes to be expensive and slow, especially the annotation process. This dependence is, at least in part, responsible for the small scale and little diversity of the datasets used in most of these works. The consequence of using limited datasets is seen in the performance of the proposed solutions. Therefore, proposing new ways to automate the data-related processes (possibly removing humans from the loop) will result in the decrease in cost and the increase in size and diversity of the collected datasets, which will likely lead to better models. In this thesis, we investigate and develop methods and applications in this direction.

3.2 Methods & Applications

The investigation of ways to mitigate the efforts in acquiring and annotating large-scale data sets led us to develop three methods and real-world applications. In the subsections below, these contributions are described in detail with the experiments and results that validate their efficacy. These contributions were published in the following papers: (BERRIEL et al., 2017b), (BERRIEL et al., 2017), and (BERRIEL et al., 2018).

3.2.1 Satellite Crosswalk Classification

Zebra crossing classification and detection are important tasks for mobility autonomy. Even though, there are few data available on where crosswalks are in the world. The automatic annotation of crosswalks' locations worldwide can be very useful for online maps, GPS applications and many others. In addition, the availability of zebra crossing locations on these applications can be of great use to people with disabilities, to road management, and to autonomous vehicles. However, automatically annotating this kind of data is a challenging task. They are often aging (painting fading away), occluded by vehicle and pedestrians, darkened by strong shadows, and many other factors.

In this section, we present a system able to automatically acquire and annotate zebra

crossings satellite imagery, and train deep-learning-based models for crosswalk classification in large scale. The proposed system can be used to automatically annotate crosswalks worldwide, helping systems used by the visually impaired, autonomous driving technologies, and others. This system is the result of a comprehensive study. This study assesses the quality of the available data, the most suitable model and its performance in several imagery levels (city, country, continent and global). In fact, this study is performed on real-world satellite imagery (almost 250,000 images) acquired and annotated automatically. Given the noisy nature of the data, results are also compared with human annotated data. The proposed system is able to train models that achieve 97.11% on a global scale.

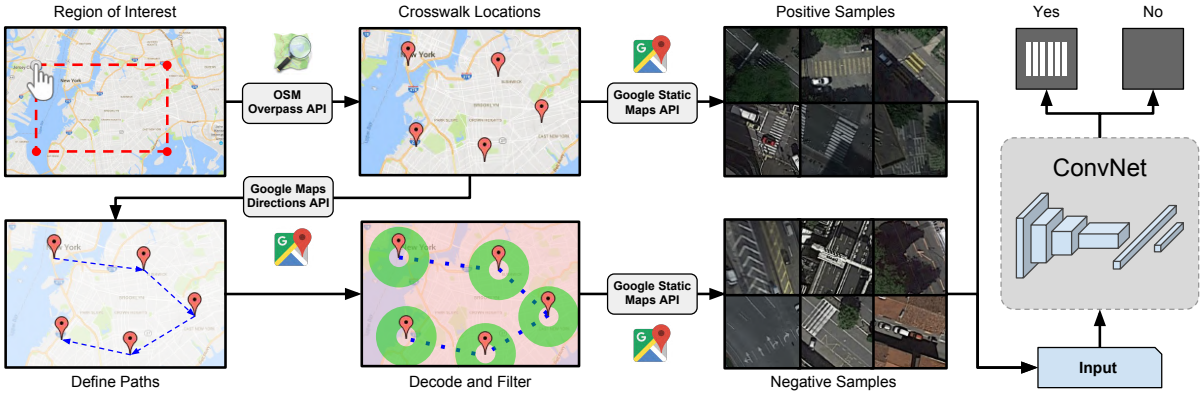


Figure 8 – System architecture. The input is a region (red dashed rectangle) or a set of regions of interest. Firstly, known crosswalk locations (red markers) are retrieved using the OpenStreetMap (OSM). Secondly, using Google Maps Directions API, paths (blue dashed arrows) between the crosswalk locations are defined. Thirdly, these paths are decoded and the result locations are filtered (only locations within the green area are accepted) in order to decrease the amount of wrongly annotated images. At this point, positive and negative samples can be downloaded from Google Static Maps API. Finally, this large-scale satellite imagery is used to train Convolutional Neural Networks (ConvNets) to perform zebra crossing classification.

The system comprises two parts: Automatic Data Acquisition and Annotation, and Model Training and Classification. An overview of the proposed method can be seen in the Figure 8. Firstly, the user defines the regions of interest (regions where he wants to download crosswalks). The region of interest is given by the lower-left and the upper-right corners. After that, crosswalk locations within these regions are retrieved from the OpenStreetMap[2]. Subsequently, using the zebra crossing locations, positive and negative images (i.e., images that contain and do not contain crosswalks on it, respectively) are downloaded using the Google Static Maps API[3]. As the location of the crosswalks are known, the images are automatically annotated. Finally, these automatically acquired and annotate images are used to train a Convolutional Neural Network to perform classification.

[2] <http://www.openstreetmap.org>
[3] <http://developers.google.com/maps/documentation/static-maps/>

Each process is described in details in the following subsections.

Automatic Data Acquisition and Annotation. To automatically create a model for zebra crossing classification, the first step is the image acquisition. Initially, regions of interest are defined by the user. These regions can either be defined manually or automatically (e.g. based on a given address, city, etc.). The region is rectangular (with the True North up) and is defined by four coordinate points: minimum latitude, minimum longitude, maximum latitude, maximum longitude (or South-West-North-East), i.e., the bottom-left and top-right corners (e.g. 40.764498, -73.981447, 40.799976, -73.949402 defines the Central Park, NY, USA region). For each region of interest, zebra crossing locations are retrieved from the OpenStreetMap (OSM) using the Overpass API[4]. Regions larger than 1/4 degree in either dimension are likely to be split into multiple regions, as OpenStreetMap servers may reject these requests given it is the maximum allowed size of the API. Even though, most of the settled part of the cities around the world meets this limitation (e.g. the whole city of Niterói, RJ, Brazil fits into a single region).

To download the zebra crossings, the proposed system uses the tag `highway=crossing` of the OSM, which is one of the most reliable and most used tag for this purpose. In possession of these crosswalk locations, the system needs to automatically find locations without zebra crossing to serve as negative samples. As simple as it may look, negative samples are tricky to find because of several factors. One of these is the relatively low coverage of the zebra crossing around the world. Another is the fact that crosswalks are susceptible to changes over the years. They may completely fade away, they can be removed and streets can change. Alongside these potential problems, OSM is built by volunteers, therefore open to contributions that may not be as accurate as expected. Altogether these factors indicate how noisy the zebra crossing locations may be. Therefore, picking up no-crosswalk locations must be done cautiously. The first step to acquire good locations for the negative samples is to filter only regions that contain roads. For that, the system queries Google Maps Directions API for directions from a crosswalk to another to ensure that the points will be within a road. In order to lower the number of requests to the Google Maps Directions API, our system adds 20 other crosswalks as waypoints between two crosswalk points (the API has a limit of up to 23 waypoints and 20 ensures this limit). Increasing the number of waypoints would not affect the accuracy performance of the final system because the same images would be downloaded but requiring more time. Google Maps Directions API responds to the request with an encoded polyline. The decoded polyline comprises a set of points in the requested path. The second step consists of virtually augmenting the number of locations using a fixed spacing of 1.5×10^{-4} degrees (approximately 16 meters). All duplicate points are removed on the third step. Finally, the system also filters out all images too close or too far away. Too close locations

[4] <http://wiki.openstreetmap.org/wiki/Overpass_API>

may contain crosswalks and could create false positives; and too far locations may have non-annotated crosswalks and could create false negatives. Therefore, locations closer than 3×10^{-4} degrees or farther than 6×10^{-4} degrees or locations outside the region requested are removed to decrease the occurrence of false positives and false negatives.

After acquiring both positive and negative sample locations, the proposed system dispatches several threads to download the images using the Google Static Maps API. Each requested image is centered on the location to be requested. Also, the images are requested with a zoom factor equal to 20 and size of 200×225 pixels. This size was empirically defined to have a good trade-off between the image size and the field of view of the area. Bigger sizes would increase the probability of having crosswalks away of the queried location, while smaller images would decrease the robustness of the automatic data acquisition process w.r.t. the imprecision of the annotated crosswalk locations, hence adding more noise into the collected datasets. As a result, each image covers $\approx 22 \times 25$ meters. Some positive and negative samples can be seen in the Figure 9.

Model training and classification. Before initializing the model training, an automatic pre-processing operation is required. Every image downloaded from the Google Static Maps API contains the Google logo and a copyright message on the bottom. In order to remove these features, 25 pixels are removed from the bottom of each image, cropping the original images from 200×225 to 200×200. In possession of all cropped images, both positive and negative samples, the training of the model can begin.

To tackle this large-scale problem, the proposed system uses a deep-learning-based model: a Convolutional Neural Network (LECUN et al., 1989). The architecture of the model used by the system is the VGG (SIMONYAN; ZISSERMAN, 2014). It was chosen after the evaluation of three different architectures: AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), VGG and GoogLeNet (SZEGEDY et al., 2015) with 5, 16, and 22 convolutional layers, respectively. All models started from pre-trained models on the ImageNet (RUSSAKOVSKY et al., 2015), i.e., fine-tuning. In addition, the input images were upsampled from 200×200 to 256×256 using bilinear interpolation and the subtraction of the mean of the training set was performed. More details on the training configuration are described in Section 3.2.1.1. Also, the last layer of VGG was replaced by a fully-connected layer comprising two neurons with randomly initialized weights, one for each class (crosswalk or no-crosswalk), and 10 times higher learning rate when compared to the previous layers (due to fine-tuning).

3.2.1.1 Experimental Methodology

In this section, we present the methodology used to evaluate the proposed system. First, the dataset is properly introduced and described. Then, the metrics used to evaluate the proposed system are presented. Finally, the experiments are detailed.

3.2.1.1.1 Dataset

The dataset used in this work was automatically acquired and annotated using the system hereby presented. The system downloads satellite images using the Google Static Maps API and acquires the annotations using the OpenStreetMap. In total, the dataset comprises 245,768 satellite images, 74,047 images of which contain crosswalks (positive samples) and 171,721 do not contain zebra crossings (negative samples). To the best of our knowledge, this is the largest satellite dataset for crosswalk-related tasks in the literature. In the wild, crosswalks can vary across different cities, different countries and different continents. Alongside the design variations, they can be presented in a variety of conditions (e.g. occluded by trees, cars, pedestrians; with painting fading away; with shadows; etc.). In order to capture all this diversity, this dataset comprises satellite imagery from 3 continents, 9 countries, and at least 20 cities. The cities were chosen considering the density of available annotations and the size of the city. It was given preference to big cities assuming that they are better annotated. In total, these images add up to approximately 135,000 square kilometers, even though different images may partially contain a shared area. Some samples of crosswalks are shown in the Figure 9. A summary of the dataset can be seen at the Table 1. It is worth noting that, even though each part of the dataset is named after a city, some of the selected regions were large enough to partially include neighboring towns. A more detailed description of each part of the dataset, region locations and scripts used for the data acquisition are publicly available[5].

Table 1 – Number of Images on the Datasets Grouped by Continents

Description	**Crosswalks**	**No-Crosswalks**	**Total**
Europe	42,554	99,461	142,015
America	15,822	36,811	52,633
Asia	15,671	35,449	51,120
Total	**74,047**	**171,721**	**245,313**

3.2.1.1.2 Metrics

On this classification task, we reported the global accuracy (Equation 3.1) and the F_1 score (Equation 3.2).

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} \tag{3.1}$$

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{3.2}$$

[5] <http://github.com/rodrigoberriel/satellite-crosswalk-classification>

Figure 9 – First row presents positive samples with varying layouts of crosswalks. Second row has some challenging positive cases (crosswalks with strong shadows, occlusions, aging, truncated, etc.). Third row presents different negative examples. Last row has some challenging negative cases.

where TP, TN, P and N means the number of True Positive, True Negative, Positive and Negative, respectively. Precision is defined by $\frac{\text{TP}}{\text{TP+FP}}$ and recall by $\frac{\text{TP}}{\text{P}}$, where FP means False Positive.

3.2.1.1.3 Experiments

Several experiments were designed to evaluate the proposed system. Initially, three well-known Convolutional Neural Network architectures were evaluated: AlexNet, VGG and GoogLeNet. The images downloaded from the Google Static Maps API were upsampled from 200×200 to 256×256 using bilinear interpolation. As a data augmentation procedure, the images were randomly cropped (224×224 to the VGG and GoogLeNet, and 227×227 to the AlexNet – as in the original networks), and the images were randomly mirrored on-the-fly.

It is known that crosswalks vary across different cities, countries and continents. In this context, some experiments were designed based on the expectation that crosswalks belonging to the same locality (e.g., same city, same country, etc.) are more likely to present similar features. Therefore, in these experiments, some models were trained and evaluated on the same locality, and they were named with the prefix "intra": intra-city, intra-country, intra-continent and intra-world. At the smaller scales (e.g., city and country) only some of the datasets were used (assuming the performance may be generalized),

at higher levels all available datasets were used. These datasets were randomly chosen considering all cities with high annotation density. Ultimately, the intra-world experiment was also performed.

Besides these intra-based experiments, in which samples tend to have high correlation (i.e., present more similarities), cross-based experiments were performed. The experiments named with the prefix "cross" are those in which the model was trained with a dataset and evaluated in another with the same level of locality, i.e., trained using data from a city and tested in another city. Cross-based experiments were performed on the three levels of locality: city, country and continent. Another experiment performed was the cross-level, i.e., the model trained using an upper level imagery (e.g., the world model) was evaluated in lower levels (e.g., continents).

All the experiments aforementioned share a common setup. Each dataset was divided into train, validation and test sets, with 70%, 10% and 20%, respectively. For all the experiments, including the cross-based ones (cross-level included), none of the images in the test set were seen by the models during training or validation, i.e., train, validation and test sets were exclusive. In fact, the cross experiments used only the test-set of the respective dataset on which they were evaluated. This procedure enables fairer comparisons and conclusions about the robustness of the models, even though the entire datasets could have been used on the cross-based models. Regarding the training, all models were trained during 30 epochs and the learning rate was decreased three times by a factor of 10. The initial learning rate was set to 10^{-4} to all three networks.

Lastly, the annotations from the OpenStreetMap may not be as accurate as required due to many factors (e.g., human error, zebra crossing was removed, etc.). Even though, we assume the vast majority of them are correct and accurate. To validate that, an experiment using manually labeled datasets from the three continents (America, Europe, and Asia – 44,175 images in total) was performed. The experiment was designed to have three results: error of the automatic annotation; performance increase when switching from automatic labeled training data to manually labeled; and, correlation between the error of the automatic annotation and the error of the validation results.

3.2.1.2 Results

Several experiments were performed and their accuracy and F_1 score were reported. Initially, different architectures were evaluated on two levels of locality. As can be seen in the Table 2, VGG achieved the best results. It is interesting to notice that AlexNet, a smaller model that can be loaded into smaller GPUs, also achieved competitive results.

Regarding the intra-based experiments, the chosen model showed to be very consistent across the different levels of locality. The model achieved 96.9% of accuracy (on average) and the details of the results can be seen in the Table 3.

Table 2 – Different Architectures using Intra-Based Protocol

Architecture	**Level**	**Dataset**	**Accuracy**	F_1 **score**
AlexNet	City	Milan	96.69%	94.56%
	City	Turim	95.39%	92.51%
	Country	Italy	96.06%	93.53%
VGG	City	Milan	**97.00%**	**95.10%**
	City	Turim	**96.37%**	**94.15%**
	Country	Italy	**96.70%**	**94.66%**
GoogLeNet	City	Milan	96.04%	93.41%
	City	Turim	94.27%	90.78%
	Country	Italy	95.22%	92.17%

Table 3 – Intra-Based Results for the VGG Network

Level	**Dataset**	**Accuracy**	F_1 **score**
City	Milan	97.00%	95.10%
	Turim	96.37%	94.15%
Country	Italy	96.70%	94.66%
	France	95.87%	93.12%
Continent	Europe	96.72%	94.50%
	America	96.77%	94.55%
	Asia	98.61%	97.71%
World	World	97.11%	95.17%

Table 4 – Cross-Based Results for the VGG Network

Level	**Train/Val**	**Test**	**Accuracy**	F_1 **score**
City	Milan	Turim	93.47%	89.80%
	Turim	Milan	95.40%	92.36%
Country	Italy	France	94.78%	91.16%
	France	Italy	94.78%	91.61%
Continent	Europe	Asia	96.62%	94.33%
	Asia	Europe	93.65%	88.94%

As expected, the cross-based models achieved a lower overall accuracy when compared to the intra-based. Nevertheless, these models were still able to achieve high accuracies (94.8% on average, see Table 4). This can be partially explained by inherent differences on the images between places far apart, i.e., different solar elevation angles cause notable differences on the images; the quality of the images may differ between cities; among other factors that tend to be captured by the model during the training phase.

Cross-level experiments reported excellent results. As already discussed, none of

these models had contaminated test sets. Yet, on average, they achieved 96.1% of accuracy. The robustness of these models can be seen in the Table 5, where all the cross-level results were summarized.

Table 5 – Cross-Level Results for the VGG Network
Train/Val→Test

Cross-Level	Train/Val	Test	Accuracy	F_1 score
Country→City	Italy	Milan	97.17%	95.37%
	Italy	Turim	96.28%	94.04%
Continent→Country	Asia	Portugal	92.71%	86.04%
	Asia	Italy	94.69%	91.43%
World→Continent	World	Europe	96.72%	94.50%
	World	America	96.66%	94.35%
	World	Asia	98.65%	97.79%

Lastly, results of manual annotations showed the proposed system can automatically acquire and annotate satellite imagery with an average accuracy of 95.41% (4.04% false positive and 4.83% false negative samples), see Table 6 for detailed automatic annotation errors. In addition, results of the manual annotation also showed a small improvement (2.00% on average, see Table 6) on the accuracy when switching from automatic labeled training data to manually labeled. This improvement is due to the decrease in noise of models trained using the manual labels. Some failure cases are shown in Figure 10. As can be seen in the Table 6, there is a correlation between the error of the automatic annotation and the accuracy of the resulting models, i.e., an increase in the annotation error implies in a decrease in the accuracy of models using automatic data. Table 6 also shows that the absolute differences between models validated with automatic data and manual data are not very high, at most 1.75% which is the difference for New York. This indicates that all our previous results of experiments evaluated with automatic data are valid, and would not be much different if they were evaluated with manually annotated data.

Table 6 – Impact of Manual Annotation on the Accuracy for the VGG
A: Automatic – M: Manual

Dataset	Annotation Error	TrainVal / Test		
		A / A	A / M	M / M
Milan	2.58%	97.00%	97.71%	98.91%
Turim	6.57%	96.37%	94.69%	98.32%
New York	6.77%	95.49%	93.74%	96.62%
Toyokawa	1.47%	98.16%	99.04%	99.33%

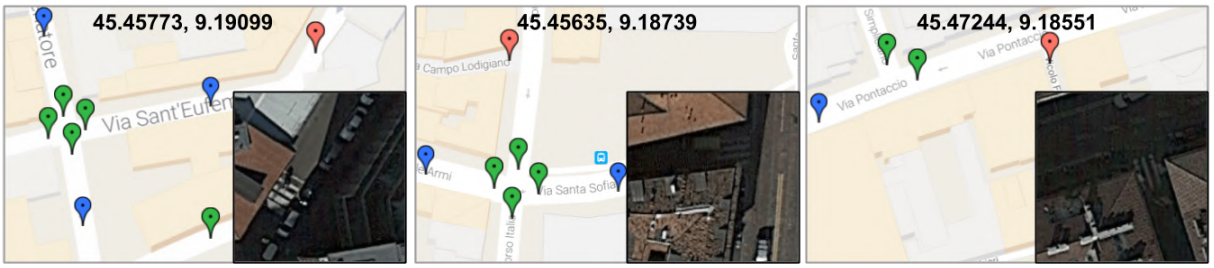


Figure 10 – Failure cases. True Positive and True Negative are represented by green and blue markers, respectively. False Positive and False Negative are both represented by the red markers.

3.2.2 Street View Crosswalk Classification

As already said, correctly identifying crosswalks is an essential task for the driving activity and mobility autonomy. Moreover, walking in the sideways and crossing streets may seem like easy and effortless activities. However, people with disabilities, especially the almost 285 million visually impaired people worldwide (WHO: World Health Organization, 2014), have several challenges regarding mobility autonomy. Nonetheless, solving this problem is not easy. There are many challenges, such as: crosswalks are often aging, i.e. the painting is fading away; occluded by vehicle and pedestrians; darkened by strong shadows and many other factors. These factors are especially challenging in developing countries, such as Brazil, where maintenance of the roads and paintings is worse than in developed countries.

As we have seen in this thesis, crosswalks can be detected in several perspectives: top-view and satellite imagery; from behind the windshield and from the top of the car (cockpit view); and from the pedestrian perspective. Each perspective has a different purpose, i.e. target different potential applications. Although the general problem of crosswalk classification can be tackled in all these perspectives, we now focus on the perspective of the cockpit view only. Although the problem has been widely studied in the literature over the years, there are still limitations. In most of these works, the data sets are small and local, i.e. they are limited to a specific neighborhood, city or small (developed) country. Developing countries, such as Brazil, present a wide range of challenges that may not be encountered in those small data sets of developed countries.

In this section, we present a streetview crosswalk classification system based on deep learning. The system exploits crowdsourcing platforms to automatically train a ConvNet to perform crosswalk classification in a developing country: Brazil. The proposed system can be used to classify crosswalks in real-time, such as in advanced driver assistance systems and autonomous vehicles. Additionally, we propose a comparison study of models trained using the fully-automatic data acquisition and annotation against models that were partially annotated manually. Moreover, focus is given on evaluating the system on data from real-world applications, i.e. comparing to images from others sources (such

as different cameras) and not only to images coming from the crowdsourcing platform. The results show that the models can accurately classify crosswalks using the automatic training (average overall accuracy of $94.12 \pm 0.21\%$) and that manually annotating a specific part of the data can boost the performance of the system (average overall accuracy of $96.30 \pm 0.14\%$). Additionally, despite of being developed and evaluated with focus in Brazil, these models presented robustness when applied to different image sources, being scalable to use in other countries. Finally, the results indicate that the proposed system is able to automatically train models to be used on real-world applications.

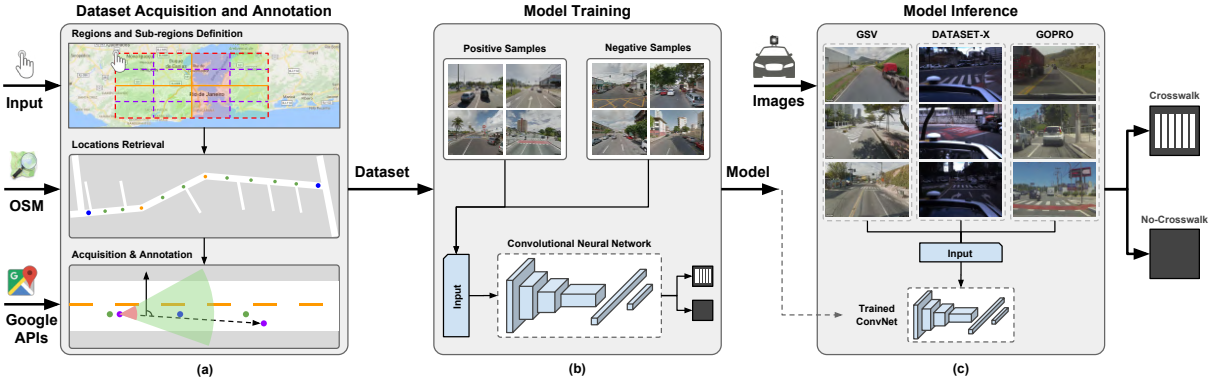


Figure 11 – Overview of the proposed system. Firstly, in the dataset acquisition and annotation (a), the system receives the user input to define the regions of interest, then retrieve the locations (both positive and negative) of interest using the OpenStreetMap (OSM) and Google Maps Directions API, and, finally, automatically acquires (using Google Street View Image API) and annotates the samples outputting a dataset. Secondly, in the model training (b), this dataset is used to train a convolutional neural network to predict whether or not there is a crosswalk in a given image, outputting a trained model. Finally, during the model inference (c), images from different sources (such as different cameras) are given to the model and it yields the probability of a given image to contain crosswalks or not.

The system comprises two main parts: automatic data acquisition and CNN model training/inference. The automatic data acquisition can be performed having just a selection of the area to be acquired. The data is treated, downloaded, and automatically annotated. Alternatively, the data can be partially annotated if desired. With the data, the CNN model can be trained and later used for inference of a given image. The output of the system is a label indicating the presence or not of the crosswalk in a given image. An overview of the proposed system can be seen in the Figure 11.

Dataset Acquisition and Annotation.

Acquisition of a large dataset is primordial for automatically training a Convolutional Neural Network. Therefore, the first step of the proposed system is the automatic acquisition and annotation of a large-scale database of images using the APIs of the Google

Street View, Google Directions and OpenStreetMap. This acquisition process starts with the user defining the regions of interest. Subsequently, the system automatically downloads the data (positive and negative samples) to identify the presence of crosswalks. Finally, the actual imagery is downloaded and automatically annotated. Alternatively, the user can manually annotate part of the data to correct errors of the automatic system and increase accuracy of the final crosswalk classification.

Regions and Sub-regions Definition. The only user interaction necessary is the definition of the region of interest. A region of interest is defined as a rectangular region in world-coordinate system in which the user aims acquiring images (both positive and negative samples). The user can provide one or multiple regions of interest. Each region is defined by two points and must be defined regarding world coordinates. The points required are the bottom-left and the top-right of the region.

For each region of interest, the proposed system retrieves crosswalk locations using the OpenStreetMap[6] (OSM) via Overpass API[7]. One of the limitations imposed by the Overpass API is the size of each of these regions. The maximum allowed size, in any dimension (width or height), of a requested region to the Overpass API is 1/4 degrees. Whenever one of the dimensions of a region is greater than 1/4 degrees, the request is likely to be refused. Therefore, regions greater than this limit are automatically divided into multiple sub-regions. This process is illustrated in the Figure 12, where the region in black solid line has both dimensions greater than the limitation and the orange solid line are the boundaries of the sub-regions A, B, C and D. At this point, each sub-region dimension is smaller than 1/4 degrees and all sub-regions have the same dimension, i.e. the same height and the same width, although regions may be rectangular (i.e. width and height may be different from one another). When the sub-regions are determined, i.e all sub-regions are complying with the Overpass API constraint, the requests for crosswalk locations are dispatched. OpenStreetMap has several tags referring to pedestrian crossings. In this work, the tag `highway=crossing` was used. This tag has almost two times more nodes associated with it than one of the "useful combinations" (`crossings=*`). Moreover, this tag is the recommended one when it comes to crossing infrastructure for the convenience of pedestrians. Therefore, crosswalk locations are requested using the tag `highway=crossing`. If the request is successful, the responses carry the crosswalk locations of the regions of interest.

With the crosswalk locations in hand, the first strategy to avoid noise is performed. This strategy, illustrated in the Figure 12, comprises splitting each sub-region that contains more than 2000 crosswalks into multiple other sub-regions with equal dimensions. This split follows the same procedure performed on the regions. If the resulting sub-region contains

[6] <http://www.openstreetmap.org>
[7] <http://wiki.openstreetmap.org/wiki/OverpassAPI>

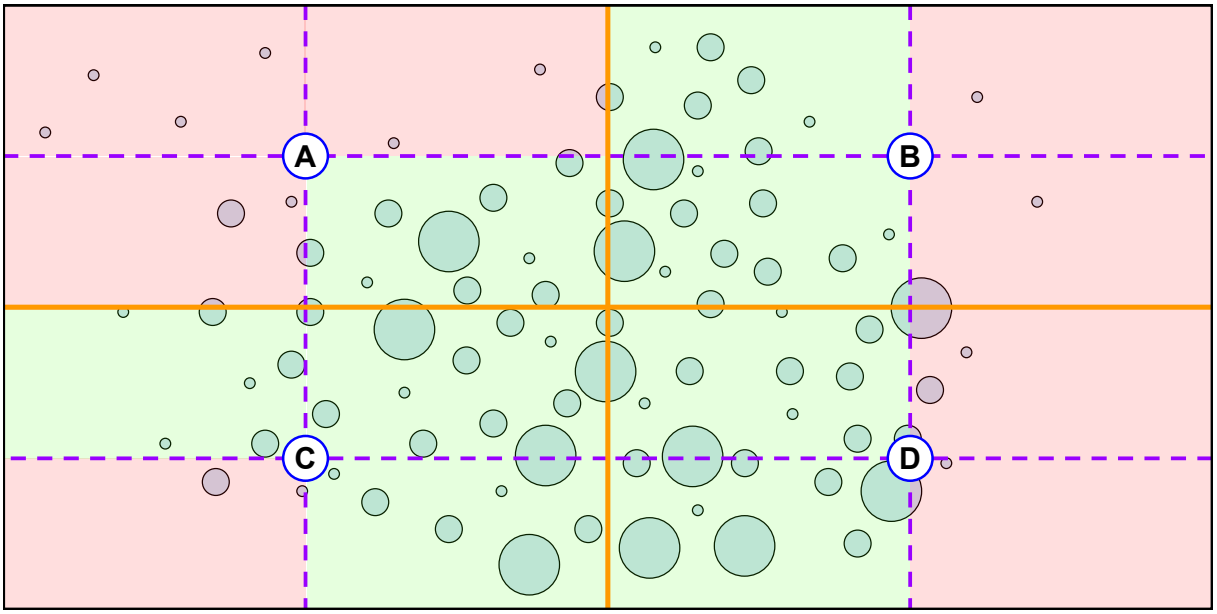


Figure 12 – Regions and sub-regions definition. A region (black solid line) must be compliant with Overpass API limitations. If any dimension is larger than 1/4 degrees, the region is split into sub-regions (A, B, C and D – orange solid lines) that comply the limitation. If any of these sub-regions contain between 50 and 2000 crosswalks (green sub-regions), the sub-region is kept. Otherwise, if it contains less than 50 crosswalks (red sub-region), it is discarded. If it contains more than 2000, the split is applied recursively, until all regions have been either discarded or kept. Blue circles indicates crosswalk density, i.e. larger the circle, more crosswalks in that area.

less than 50 crosswalks (red sub-regions in the Figure 12), it is discarded. Otherwise, if it contains less than 2000 and more than 50 crosswalks (green sub-regions in the Figure 12), it is kept. If the sub-region still has more than 2000 crosswalks, the process is applied recursively until all sub-regions have more than 50 and less than 2000 crosswalks. These values were empirically defined. This process is an attempt to remove areas of the regions of interest that present low density of crosswalk annotations. These areas may have been inconsistently annotated or they simply contain fewer crosswalks. In both cases, these areas are likely to contain crosswalks that were not annotated, which might result in false negative samples.

Locations Retrieval. At this point, all crosswalk locations of all regions of interest were retrieved. Nevertheless, neither positive nor negative samples can be acquired yet. From the perspective of a car or a driver (cockpit view), images that contain crosswalks are taken from locations near to crosswalks, i.e. not at the crosswalk locations themselves. In other words, there is still need to discover places nearby crosswalk locations. The location of these nearby places are discovered using the following procedure. Initially, a path between two known crosswalk locations (both randomly chosen within a sub-region) is requested to the Google Maps Directions API. To reduce the number of requests to this

API, each request contains up to 23 subsequent crosswalk locations (blue circles in the Figure 13), besides the origin and destination. The Google Maps Directions API returns by default the directions in the driving mode, i.e. that a car could follow to go from one place to the other. These directions are returned as encoded polylines. The system decodes these polylines into a list of points (orange circles in the Figure 13), i.e. locations. However, these points are distributed in order to optimize the encoding process, resulting in points that are not evenly distributed.

Positive samples are necessarily near to the crosswalk locations, therefore the list of locations derived from the polylines needs to be augmented in order to generate more samples to be collected near by the crosswalk locations. The augmentation (see Figure 13) is performed by sampling locations between two subsequent points at equal distances. The sampling strategy ensures that each sampled location (green circles in the Figure 13) is at most 1×10^{-4} degrees (≈ 11 meters) from another location. At the end of the augmentation process, the proposed system has a much larger list of locations that can be used to retrieve both positive and negative samples, and capture images with larger diversity.

Finally, this list needs to be cleared of duplicate locations. Duplicate locations may appear because the paths provided by the Google Maps Directions API may be partially shared across different requests, i.e. there may be partial path overlap between two or more path requests. Besides that, the augmentation process can also generate points in locations already existent in another paths.

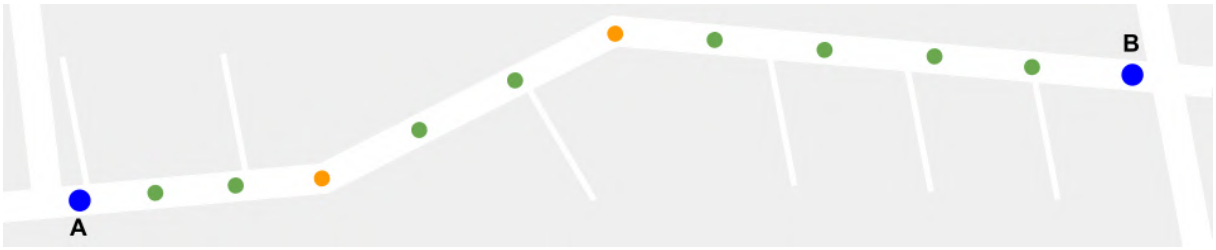


Figure 13 – Location Augmentation. Given two crosswalk locations (A and B), a path is requested to the Google Maps Directions API. The path delivered by the API comprises a list of points (orange points, besides the origin and destination) encoded into a polyline. Finally, this list of locations is augmented by evenly sampling locations (green circles) between the points.

Samples Acquisition and Automatic Annotation. To request positive and negative samples (images) to the Google Static Maps API[8], there are some parameters to be set. Two parameters are mandatory: location and size. The location (in world coordinates, i.e. latitude and longitude) varies according to the position of the sample, and the size is fixed as 640×520 pixels. Besides these mandatory parameters, there are three others that influence the resulting image and are optional: field of view, pitch and heading.

[8] <http://developers.google.com/maps/documentation/static-maps>

In the first two parameters (field of view and pitch), the default values were used: 90 and 0 degrees, respectively. As the documentation says, the field of view (fov) can be seen as a zoom factor, i.e. all the images will present the same "zoom factor", and the pitch is the vertical angle of the camera relative to the vehicle, i.e. the default angle is often flat horizontal. The last one (heading) is basically the direction that the camera is pointed to. Differently than the previous ones, the direction needs to be carefully calculated in order to retrieve useful images.

Before defining the direction for each location, an additional step is performed. This additional step is required because the Street View images come from specific locations, i.e. the locations defined so far are likely not to be the exact locations of available Street View images. To retrieve the location of the nearest Street View image of a given location, the proposed system needs to dispatch a request to the Street View Image Metadata API. This API requires a location and returns the latitude and longitude of the nearest Street View panorama to the given location. Besides that, the id of the panorama, the date the photo was taken, and copyright information are also provided by the Street View Image Metadata API. If there is no Street View image available near (i.e. in a radius of 50 meters) to the requested location, the requested location is discarded by our system. As a result of this step, the input list of locations is turned into a list of valid Street View locations. There are three important notes about this step. First, if this step is not performed, the direction for each location can be wrongly calculated, and the resulting image may differ from the expected. Second, two different locations in the input list may be represented by the same Street View image location, and this may lead to duplicate locations. For that reason, only unique locations remain in the resulting list of valid Street View locations. Third, when the requests of the proposed system were dispatched, there was a bug that caused requests to Street View Image Metadata API to consume the quota, and this quota is shared with the Google Street View Image API (used to request the images). Therefore, the least requests, the better. However, this bug has been fixed by Google[9].

After the generation of the list of valid Street View image locations, the proposed system is ready to calculate the heading for each location and request the images. The heading (α) for each location is defined by the Equation 3.3.

$$\alpha = \text{atan2}(y_{i+1} - y_i, x_{i+1} - x_i) \cdot \frac{180}{\pi} \tag{3.3}$$

where α is defined in degrees, and x_i and y_i are the latitude and longitude of the i-th valid Street View location (e.g. A' and C' in the Figure 14), respectively. After the definition of the heading for every location, the system is ready to dispatch the requests of both positive and negative samples to the Google Street View Image API.

[9] <https://issuetracker.google.com/issues/35830093>

Finally, the proposed system has all the images and it needs to properly annotate every single of them. Basically, the samples are separated between positive and negative using a simple rule (illustrated in the Figure 14). The image is only considered as positive sample if a crosswalk location is within the field of view of the image and in a certain range. The proposed system uses a narrower field of view (70°, from $\alpha - 35°$ to $\alpha + 35°$) than the one requested to the API (90°), i.e. it intends to reduce false positives by being more restrictive. Besides that, a crosswalk must be located farther than $5° \times 10^{-5}$ (≈ 5.6 meters) and closer than $2.5° \times 10^{-4}$ (≈ 27.8 meters). Note that these values depend on the field of view. If any crosswalk is located within this region (green region in Figure 14), the image is annotated as a positive sample. Otherwise, it will be labeled as a negative sample.

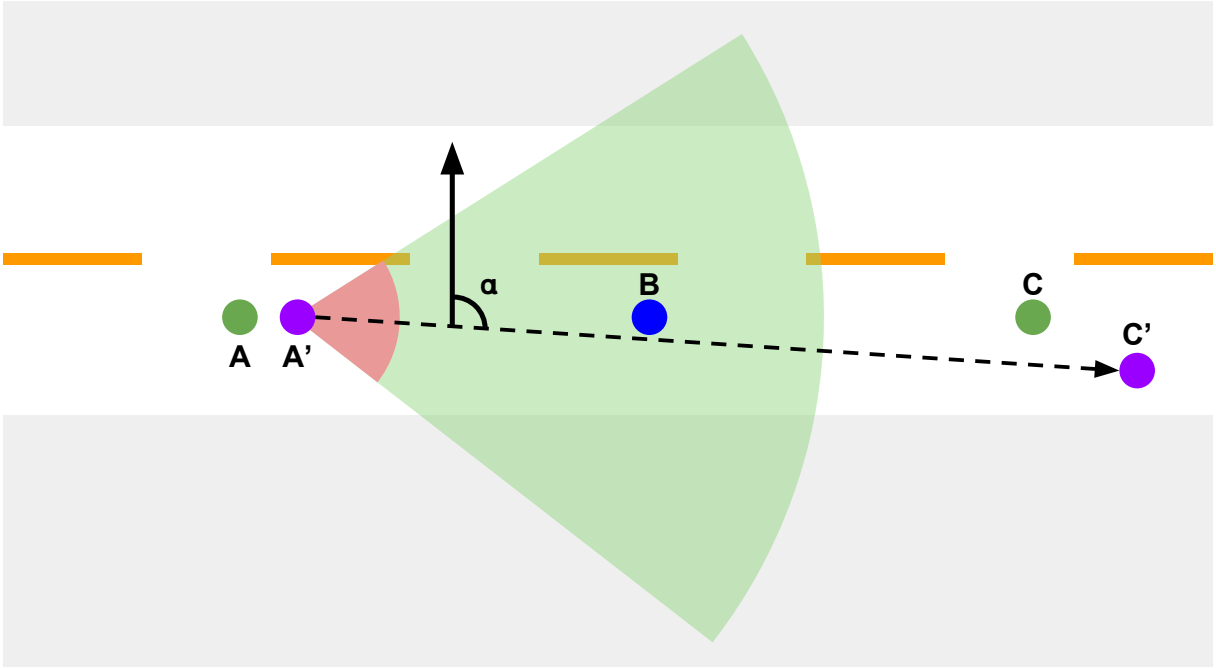


Figure 14 – Sample Acquisition and Annotation. Given three locations (A, B and C), where only B is a crosswalk (blue circle), valid Street View panorama locations are requested to the Street View Image Metadata API. The nearest panorama locations (A' and C') are represented by the purple circles. The heading of a location is defined by α. If a crosswalk location is in the green area, the image from A' is considered as a positive sample. Otherwise, it is a negative sample.

Partial Annotation. The crosswalk locations were retrieved using the Overpass API (OpenStreetMap data). The OpenStreetMap is a mapping initiative based on crowdsourcing, i.e. the data provided by the OpenStreetMap is the result of the contributions of their voluntary users. The quality of such data is based on self-regulation. Nevertheless, data provided by voluntary users may not be as accurate as required by our system. Therefore, despite of the efforts to automatically reduce false positive and negative samples, the dataset automatically acquired can be noisy. In this context, the user has the opportunity to manually annotate part of the data in order to increase the final accuracy of the system. To show the increase in accuracy with manual annotation, part of the dataset

was re-annotated by a human expert. The amount of negative samples is far greater than the amount of positive samples, as observed in the real world. Because of that, only the crosswalks were re-annotated, thence are refereed to as partial annotation. The annotations were made by a human which used two subjective criteria: *i)* the crosswalk should be occupying a reasonable area of the image, i.e. crosswalks too far should be removed (see Figure 15 (a) and (b) for positive and negative cases using this criterion); *ii)* the crosswalk in the image should be preferably in the traffic direction of the vehicle (see Figure 15 (c) and (d) for positive and negative cases using this criterion). In total, 17.36% of the 27,241 manually annotated images had their labels changed, i.e., the human expert considered them as wrongly annotated by the automatic procedure. The majority of these samples erroneously annotated were automatically annotated as positive samples: 38.24% changed from negative to positive and 61.76% from positive to negative.

(a) (b) (c) (d)

Figure 15 – Sample images manually annotated during the partial annotation. The images (a) and (c) were manually labeled as positive samples. The sample (b) was labeled as negative because the crosswalk occupies a very small area in the image, i.e. it is too far. The sample (d) was labeled as negative because the crosswalk is not in the traffic direction of the vehicle.

Model Training

In the proposed system, the acquired dataset is used to train a Convolutional Neural Network (ConvNet). The system uses the VGG architecture (SIMONYAN; ZISSERMAN, 2014), where the last layer was changed to only 2 neurons, one for each class of the crosswalk classification problem: crosswalk (positive), i.e. the image contains a crosswalk; and no-crosswalk (negative), i.e. the image does not contain a crosswalk. For the training of the model, all the images automatically acquired were used as input. The model expects an input of size 256×256 pixels. For that reason, all images were rescaled from their original size (640×520) down to 256×256 using bilinear interpolation. Moreover, a data augmentation procedure is also performed. The data augmentation is performed on-the-fly, and comprises two operations: *i)* cropping 224×224 pixels, *ii)* mirroring horizontally the image. Both operations are performed randomly, i.e. a random position is chosen to the crop, and there is a 50% chance of mirroring the image. The result of this data

augmentation is provided as input to the ConvNet. During the training, the network weights were initialized randomly using the Xavier (GLOROT; BENGIO, 2010) algorithm. Although fine-tuning procedures are widely used in the literature, preliminary analysis showed they achieved: i) results on pair with the randomly initializing the weights, i.e., there was no significant statistical difference; *ii)* no significant improvement in the learning time, i.e., the convergence rate was only slightly improved; and *iii)* worse results when they were evaluated in the cross-database (especially in the IARA dataset) compared to the ones with randomly initialized weights. Therefore, fine-tuning is not employed in the models hereby presented. All the models were trained during 10 epochs using an initial learning rate of 1×10^{-2} and a Step Down policy that decreases the learning rate by a factor of 10 three times during the training, i.e. at the end of the training process, the learning rate was equal to 1×10^{-4}. After the 10 epochs, the model is considered trained. Two models were trained: GSV-FA and GSV-PA. The former was trained using the fully-automatic (hence FA) dataset, i.e. all the images were automatically annotated by the system. The latter was trained using the partially-automatic (hence PA) dataset, i.e. all the crosswalks were manually re-annotated by an expert. In both cases, the system used all the crosswalks available and two times the amount of positive samples as negative samples. The negative samples were randomly chosen.

Model Inference

At this point, the dataset was automatically acquired and annotated, and a model is already trained. At inference, images from different image sources may be used. Therefore, all input images are downsampled to 256×256 pixels using bilinear interpolation. After that, because of the data augmentation procedure performed during the training, a centralized crop of 224×224 pixels has to be performed. This cropped image is forwarded through the ConvNet. Finally, the ConvNet yields a probability for each class: crosswalk (positive) and no-crosswalk (negative).

3.2.2.1 Experimental Methodology

In this section, the methodology and materials used in the experimentation process are described. Firstly, the datasets used for the training and evaluation of the system are detailed. Secondly, the metrics used for the quantitative experimentation are presented. Finally, the experiments are described in details.

3.2.2.1.1 Datasets

Three datasets from different sources were used during the training and evaluation of the proposed system: the dataset from the Google Street View including the automatically acquired and annotated versions; one acquired by the camera system of an autonomous

car; and, one acquired by a camera placed inside a standard car. These datasets are also described below. The scripts used for the acquisition of the Google Street View dataset will be made publicly available. In addition, the image sequences of the IARA and GOPRO datasets (more than 23,500 images) are publicly available.

Google Street View (GSV). The Google Street View dataset was automatically acquired and annotated by the proposed system. The images of this dataset came from Google StreetView (via Google Static Maps API) using the procedure described previously in the Section 3.2.2. It comprises non-sequential images of two classes: crosswalks and no-crosswalks. All the images (both positive and negative) are from Brazil. Brazilian roads present all sorts of challenges expected to be encountered in the roads worldwide. Crosswalks alone can be presented in a variety of ways: the painting can be fading away, aging; there can be pedestrians, vehicles, and other objects partially occluding the crosswalks; shadows of building, trees, large vehicles and other objects may be partially or completely darkening the road; crosswalks themselves have different styles (e.g. background colored in red, blue; with arrows on it; etc.). All these challenges can be seen in this dataset. Because of the process of retrieving locations used by the system, most images are within a road heading in the traffic direction. The size of the images was defined to 640×520. The default values of the API for both field of view and pitch were used: 90 and 0 degrees, respectively. Their documentation states that the default pitch is often, but not always, flat horizontal.

This dataset comprises 657,691 images from 20 states of the Brazil plus the federal district (Brazil has 26 states, in total). In total, there are 33,959 images of crosswalks (i.e. positive samples) and 623,732 images of no-crosswalks (i.e. negative samples). The dataset is divided into 24 regions. Even though each region is named after a city, regions may include nearby towns besides the city they are named after. The cities chosen are mainly capitals and big cities. Many factors contributed to this decision: the images on the Google Street View in the capitals are more likely to be recent and capitals are also more likely to contain crosswalk annotations in the OpenStreetMap. Some samples of the GSV dataset can be seen in the Figure 16.

In order to properly evaluate on this dataset, part of the database was separated as test set. To build the test set, some regions were chosen to provide the images. These regions were not in the training set, therefore ensuring that images in the test set are not in the training set. The test set was entirely manually labeled (27,241 images in total). Firstly, all positive samples were manually labeled following the same criteria used on the partial annotation of the dataset. Secondly, the number of negative samples to be annotated was defined as twice the number of manually annotated samples (a typical proportion of real-world use case (IVANCHENKO; COUGHLAN; SHEN, 2008)). Finally, negative samples were randomly drawn and manually annotated until the amount of truly negative

samples reach twice the number of truly positive samples. This fully-manual annotated set is refereed as test set, and it was used in the evaluation of the GSV-FA and GSV-PA models on the first experiment (Evaluation on the GSV dataset). This proportion (2:1) between the negative and positive samples was also ensured in the training and validation sets. The training and validation sets comprise all positive samples of the training regions plus randomly sampled negative ones (ensuring the 2:1 factor). The dataset splits (training, validation, and test sets) used in the experimentation are detailed in the Subsection 3.2.2.1.4.

(a) (b) (c) (d)

Figure 16 – Sample images of the GSV dataset. First two images are positive samples (contain crosswalks) and the last two are negative samples.

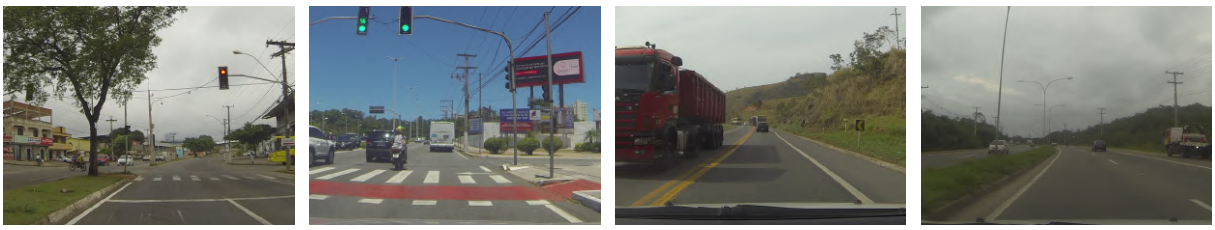
IARA. In order to assess the generalization of the trained model, the proposed system was evaluated in datasets from different sources. The IARA dataset is named after the vehicle used to capture the images: the Intelligent Autonomous Robotic Automobile. IARA is an autonomous vehicle that is being developed in the High Performance Computing Lab (LCAD) of the Universidade Federal do Espírito Santo. The vehicle contains many sensors, but only one camera was used to record this dataset. The camera, a Bumblebee XB3, was mounted on the top of the car facing forward. The images generated by this camera are very different from the ones in the GSV dataset, used during training. Therefore, the evaluation on this dataset is challenging. This dataset was recorded during the day in a week-day, i.e. it contains usual traffic. In total, there are 12,441 images from 4 different sequences recorded in the capital of the Espírito Santo, Vitória. Differently from the GSV dataset, the IARA dataset had to be manually labeled. An image was annotated as positive sample following the same criteria used in the partial annotation of the GSV dataset. In total, 2,637 images (21.2%) were labeled as positive samples and 9,804 images (78.8%) as negative samples in the IARA dataset. Some samples of the IARA dataset can be seen in the Figure 17.

GOPRO. The GOPRO dataset comprises 11,070 images recorded in the city of Vitória, Vila Velha and Guarapari, Espírito Santo, Brazil. The videos were recorded using a GoPRO HERO 3 camera in Full HD (1920×1080 pixels) at 29.97 frames per second (FPS) in different days. Some of them were recorded in city roads and the others in the highways connecting these cities. The images are divided into 29 sequences, from which

(a) (b) (c) (d)

Figure 17 – Sample images of the IARA dataset. First two images are positive samples (contain crosswalks) and the last two are negative samples.

23 of them are short sequences (up to 15 seconds) of a vehicle passing by crosswalks and the other 6 are longer sequences (up to 90 seconds) of a vehicle driving without any crosswalk in the field of view. All the videos were manually annotated in order to enable both qualitative and quantitative evaluation. The annotation followed the same criteria used on the partial annotation of the GSV dataset. In total, crosswalks were manually annotated in 17.34% of the images. It is worth noting that the process of annotating a crosswalk in a sequence is subjective, i.e. when a vehicle is approaching a crosswalk, it is really hard to tell when the annotation of that crosswalk should begin. For that reason, qualitative evaluation is essential to support the quantitative results. The crosswalks in these sequences are presented in a variety of ways, such as with pedestrians, occluded by cars, painting fading away (i.e. aging), etc. Some images of both positive and negative samples are shown in the Figure 18.



(a) (b) (c) (d)

Figure 18 – Sample images of the GOPRO dataset. First two images are positive samples (contain crosswalks) and the last two are negative samples.

3.2.2.1.2 Metrics

For the evaluation of the proposed system, two metrics were used: global accuracy (ACC, hits per number of images) and the F_1 score (harmonic mean of precision and recall). In addition to that, another metric is proposed, because these two metrics (ACC and F_1) are image-based metrics, i.e. they do not consider the temporal dimension of a dataset. In order to cope with that, an instance-based metric is proposed. This metric ($ACC_{INSTANCE}$) considers crosswalks as instances. Therefore, a hit is considered only if the

proposed system correctly classified the crosswalk at least half of its time in the sequence. In other words, if a crosswalk is shown in 20 sequential images (frames) of a sequence, the crosswalk classification will only be considered as correct if the proposed system correctly classified it for more than 10 of these frames. Therefore, the $\text{ACC}_{INSTANCE}$ reports how many of the crosswalks the proposed system correctly identified in a given temporal sequence of images (e.g., a video). This metric was only used in the videos datasets, the IARA and the GOPRO, because they are temporal sequences of images.

3.2.2.1.3 Statistical Analysis

To properly compare the proposed models (GSV-FA and GSV-PA), a pairwise statistical comparison was carried out. For the statistical analysis, we have used the paired t-test (CASELLA; BERGER, 2002). In this test, our null hypothesis is that these models (GSV-FA and GSV-PA) are equivalent. Therefore, the lower the p-value, the lower the probability of these models to be equivalent, i.e., the distributions are indeed statistically significantly different. To define whether or not this is the case, a cutoff value has to be specified. It is common to use a p-value of 5×10^{-2} as a threshold, however, in this work we used 1×10^{-4} to be more restrictive. Using this cutoff value means that, if the models were equivalent, there would be a 0.1% chance of obtaining the observed results. That is, if the p-value is lower than 1×10^{-4}, the difference can be considered significant.

3.2.2.1.4 Experiments

Some experiments were designed for the evaluation of the proposed system. These experiments intend to assess the generalization of the system and to measure its performance in terms of the proposed metrics. The quantitative experimentation can be unfolded in two experiments, one for each dataset. In the first experiment, the models were trained on several different combinations of the GSV-FA and GSV-PA datasets and evaluated on the GSV test set. Then, the best models of the first experiment are used on the cross-database experiments: the model trained on the GSV dataset is evaluated on the IARA and GOPRO datasets. Besides that, qualitative experimentation was also performed. These experiments are detailed below.

Quantitative

Quantitative experiments intend to report the accuracy and the F_1 score for the evaluation on the GSV dataset, and all three metrics (ACC, F_1, and $\text{ACC}_{INSTANCE}$) for the cross-database experiments.

Evaluation on the GSV dataset. In the first part, the proposed system is used to train two models: one using the fully automatically annotated images (GSV-FA) and

another using the partially automatically annotated images (GSV-PA), where FA and PA stands for fully-automatic and partially-automatic, respectively. The difference between them is that in the former all the images are labeled according to the automatic procedure of the proposed system, and in the latter only the crosswalks images (i.e. positive samples) were manually labeled. This experiment intends to measure the advantage of having part of the data manually annotated. The dataset splits used for the training was carefully chosen to ensure a fairer evaluation protocol. First, the GSV dataset was split into three sets: train, validation and test. The test set was chosen region-wise, i.e. none of the regions in the test set was seen during training or validation. The test regions were randomly chosen trying to keep approximately 25% of the database in the test set. The train and validation splits comprise the rest of the data, where 90% of it was used to train the models and only 10% for the validation, and together they comprise 17 regions. Second, during the training, the number of negative samples was equivalent to ≈ 2 times the number of positive samples in both models. The negative samples were randomly sampled, but as the test set comprises mutually exclusive regions, none of the negative samples used in any of the training sets were used in the test set. This imbalance was used to approximate real-world use cases, where most of the time there are no crosswalks in the field of view of the driver. As a result, each fully automatic training set contains 65,095 images and each validation set contains less than 7,000 images. Finally, in order to enable a fairer pairwise comparison of the experiments, the GSV-FA dataset was generated based on the GSV-PA, i.e. GSV-PA is always a subset of GSV-FA. As the number of crosswalks in the GSV-FA is greater than in the GSV-PA, several other no-crosswalks images (complementary set) were randomly chosen to create GSV-FA. This complementary set comprises non-duplicate images from the training/validation split. Each training set of the GSV-PA models contains 39,302 images and each validation set contains less than 4,500 images.

Cross-Database The IARA and GOPRO datasets were used for the cross-database experiments. For these experiments, the best models trained in the evaluation on the GSV dataset are used to be evaluated in the other datasets. These models are referenced as GSV-FA* and GSV-PA* (the best model trained on fully-automatic and partially-automatic dataset, respectively). This experiment was planned to evaluate the generalization of the model. In fact, the models hereby evaluated are trained on images from Google Street View and evaluated on real-world scenarios from different sources. It was ensured that the regions of the IARA and GOPRO datasets were not in the training set of the evaluated models. As these datasets comprise sequential images, the $\text{ACC}_{INSTANCE}$ is also reported.

Qualitative

In addition to the quantitative experiments, the cross-database experiments (IARA and GOPRO datasets) were recorded and are available as supplementary to allow a

qualitative evaluation. As already discussed, the qualitative evaluation is essential to support the quantitative evaluation, specially on sequences of images where the decision on when a crosswalk begin is fairly subjective. In total, more than 23,500 images (more than 30 sequences) were used on the qualitative evaluation.

3.2.2.1.5 Setup

The experiments were carried out in an Intel Core i7-4770 3.40 GHz with 16GB of RAM, and 1 Tesla K40 GPU with 12GB of memory. The machine was running Linux Ubuntu 14.04 with NVIDIA CUDA 7.5 and cuDNN 5.1 (CHETLUR et al., 2014) installed. The training and inference steps were done using a NVIDIA fork of the Caffe framework (JIA et al., 2014). In this setup, the training sessions took 4 hours, on average; and the inference can be done in more than 30 frames per second. The code for the dataset acquisition and the experiments were written in Python, using the OpenCV library. The source-code of the proposed system and the experiments performed in this work are publicly available[10].

3.2.2.2 Results and Discussions

The results of the proposed experiments are presented in this section. At first, the results of both models (GSV-FA and GSV-PA) in the evaluation on the GSV dataset are shown to identify the models with the higher overall accuracy. After that, the results of these models (GSV-FA* and GSV-PA*) on the cross-database experiments are presented. Qualitative results are shown for all the experiments.

3.2.2.2.1 Results of Evaluation on the GSV dataset

In this experiment, two models are evaluated. Initially, the results of the model trained on the fully-automatic dataset (GSV-FA) is presented. The results of the 10 runs (each run with a different negative subset) can be seen in the Table 7. On average, the model achieves an overall accuracy of 94.12%, and 89.00% of F_1 score.

As can be seen in the Table 7, the GSV-FA models achieved high performance results. Besides these results, the very low standard deviation (0.21%) indicates that the models are very robust to the noise in the dataset, specially to the many variations of no-crosswalk subsets presented to it, and to the initialization of the weights. Moreover, Table 7 shows the GSV-FA model presents a balanced accuracy between the positive and negative classes.

After evaluating the fully-automatic model, the results of model trained on the partially-automatic dataset (GSV-PA) are shown. The results of the 10 runs can be seen

[10] <https://github.com/rodrigoberriel/streetview-crosswalk-classification>

Table 7 – Accuracy and F_1 score of the Model Trained on the Fully Automatic Dataset (GSV-FA)

#	Accuracy (%)			F_1 score (%)
	Overall	**Negative**	**Positive**	
1	93.85	93.21	95.76	88.67
2	94.21	94.02	94.75	89.15
3	**94.38**	94.27	94.70	89.43
4	94.15	94.06	94.43	89.03
5	94.22	93.97	94.96	89.19
6	94.15	93.91	94.89	89.08
7	94.27	94.67	93.09	89.09
8	93.74	93.43	94.67	88.38
9	94.32	94.24	94.56	89.32
10	93.95	93.81	94.39	88.69
Average	**94.12 ± 0.21**	**93.96 ± 0.42**	**94.62 ± 0.66**	**89.00 ± 0.33**

in Table 8. For each run, the variations are generated by the different subsets of negative samples and the random initialization of the weights. On average, the GSV-PA model achieves 96.30% of overall accuracy and 92.78% of F_1 score.

Table 8 – Accuracy and F_1 score of the Model Trained on the Partially Automatic Dataset (GSV-PA)

#	Accuracy (%)			F_1 score (%)
	Overall	**Negative**	**Positive**	
1	96.42	97.18	94.15	92.97
2	96.31	96.90	94.58	92.80
3	96.41	97.22	94.01	92.94
4	96.36	96.88	94.83	92.91
5	**96.51**	97.14	94.62	93.16
6	96.06	96.43	94.94	92.37
7	96.26	96.94	94.21	92.68
8	96.18	96.59	94.94	92.58
9	96.14	96.53	94.97	92.51
10	96.39	97.16	94.08	92.90
Average	**96.30 ± 0.14**	**96.90 ± 0.29**	**94.53 ± 0.39**	**92.78 ± 0.24**

Table 8 shows that the GSV-PA models were also able to achieve very high performance results. The low standard deviation (0.14%) indicates that these models were also robust to the training variations.

In addition to the experiments with the VGG network, we performed experiments

with the AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) architecture. The results of the AlexNet models are on pair with the VGG network, which demonstrates it is possible to use a smaller network (e.g., in GPUs with lower memory) without losing much performance. We also performed an experiment on the GSV-FA with a simpler learning method (Multilayer Perceptron – MLP) by naively plugging it in the input image (since hand-crafted features are not within the scope of this study). The MLP models yielded much worse results when compared to the ConvNets (both VGG and AlexNet). It shows the benefits of a representation learning (end-to-end classification) approach.

We further explored the errors of the models, both positive and negative. We noticed that most of the positive errors (i.e., manually labeled as positive and predicted as negative) presented mainly four types of errors: *i)* genuine mistakes; *ii)* strong occlusions; *iii)* aging crosswalks (i.e., painting fading away); and *iv)* different types of crosswalks (i.e., types that are naturally less frequent such as with blue and red backgrounds, dashed style, etc.). In addition, most of the negative errors (i.e., manually labeled as negative and predicted as positive) also presented four types of errors, in general: *i)* genuine mistakes; *ii)* crosswalks that are too far; *iii)* arrows, writings, and other road markings; and *iv)* crosswalks not in the traffic direction. As can be seen in the videos of the qualitative analysis, the models predict correctly many samples in these situations (e.g., road markings, crosswalks with red background, etc.), but these are the ones they are more likely to fail.

In comparison, the GSV-PA models presented an improvement of the overall accuracy from 94.12% to 96.30% (+2.18%), as expected. Furthermore, the high performance results presented by both partially-automatic (GSV-PA) and fully-automatic (GSV-FA) models show that the GSV-FA models are very robust to the noise in the dataset. The difference is relatively small when the amount of human effort is accounted. Despite of it, this effort may be worth depending on the application. Statistical analysis shows that the difference between these two models (GSV-FA and GSV-PA) are indeed significant, i.e. p-value $\approx 5.5 \times 10^{-10}$ for the overall accuracy and p-value $\approx 2.3 \times 10^{-10}$ for the F_1 score.

3.2.2.2.2 Results of the Cross-Database Experiment

In the cross-database experiment, the best models (GSV-FA* and GSV-PA*) chosen in the evaluation on the GSV dataset are evaluated in other datasets from different sources (e.g., different cameras). The first of the cross-database experiments was carried out on the IARA dataset. The GSV-FA* model achieved 95.65% of $\text{ACC}_{INSTANCE}$. Evaluating frame-by-frame, the GSV-FA* model achieved an overall accuracy of 95.46% (F_1 score of 90.03%). The GSV-PA* model achieved an overall accuracy of 90.20% (F_1 score of 75.61%), which is slightly worse than the GSV-FA*. In addition, the GSV-PA* achieved 89.13% considering the $\text{ACC}_{INSTANCE}$. An example of the performance of the GSV-FA* and GSV-PA* models on the IARA dataset can be seen in the Figure 19. It is important to note

that GSV-PA* reported lower F_1 score when compared to the GSV-FA* mainly because of two long sequences where the vehicle was stopped in a red traffic light with vehicles highly occluding the crosswalk. These sequences were annotated as positive samples, despite of the high occlusion. On the other hand, surprisingly, the GSV-FA* was able to predict these highly occluded crosswalks consistently.

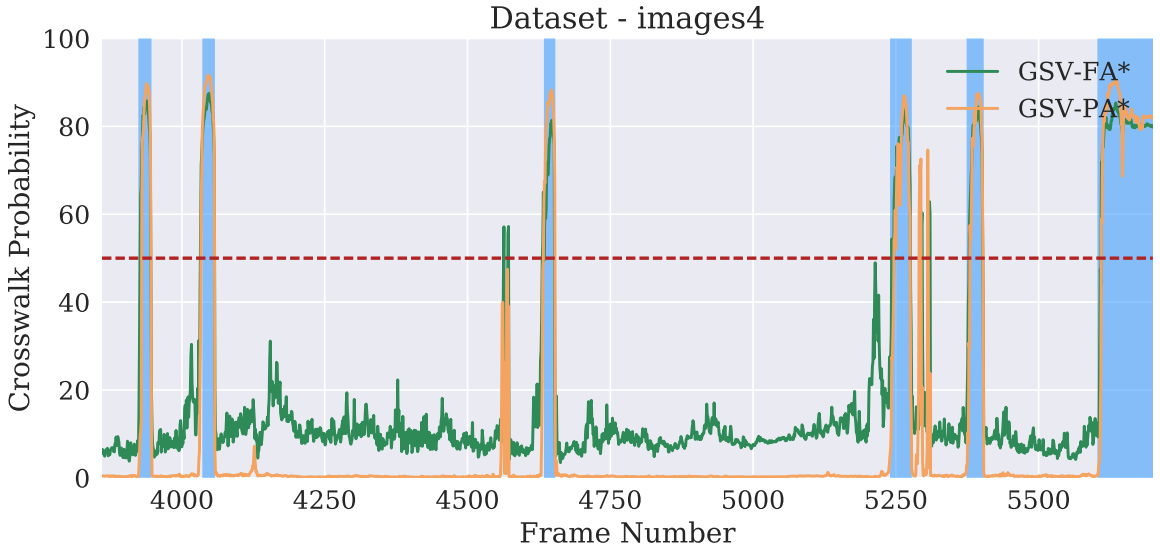


Figure 19 – Performance of the GSV-FA* and GSV-PA* models on a sequence of the IARA dataset. The red dashed line at 50% represents the threshold for the prediction: if it is below the threshold, the model is predicting no-crosswalk. Otherwise, the model predicts that there is a crosswalk in the image. Blue regions are those manually labeled as having crosswalks. The failure (false positive) case near the frame #4500 is the first image of the Figure 20.

As can be seen in the Figure 19, the GSV-PA* model is fairly more robust in the negative predictions. When it comes to the positive predictions, the GSV-PA* predicts correctly with an average of 90.30% of confidence compared to 91.88% of the GSV-FA*. Additionally, failure cases on the IARA dataset are shown in the Figure 20.

(a) (b) (c) (d)

Figure 20 – Failure samples on the IARA dataset. The first two images are false positives, and the last two are false negative predictions.

For the second database of the cross-database experiments, the models were evaluated on the GOPRO dataset. On this dataset, the GSV-FA* model achieved an

overall accuracy of 88.16%, and 92.31% on the $\text{ACC}_{INSTANCE}$ (F_1 score of 72.01%). As expected, the GSV-PA* model achieved slightly better results: overall accuracy of 92.85% and $\text{ACC}_{INSTANCE}$ of 96.15% (F_1 score of 82.07%). An example of the performance of the GSV-FA* and GSV-PA* models on the GOPRO dataset can be seen in the Figure 21.

The GSV-PA* model achieved better results on both the overall accuracy (frame-based metric) and on the $\text{ACC}_{INSTANCE}$ (crosswalk-based metric). This can be explained by the fact that the GSV-FA* model is less stable than the GSV-PA*: during the crosswalks, the GSV-PA* predicted crosswalk for 82.28% of the samples, on average, against 88.56% of the GSV-FA*. Additionally, failure cases on the GOPRO dataset are shown in the Figure 22.

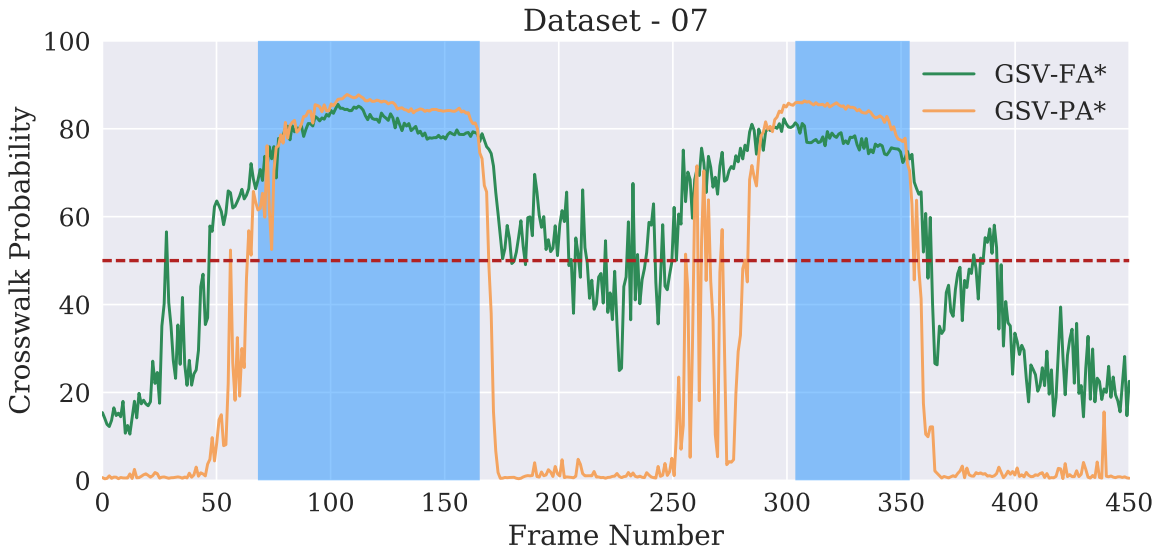


Figure 21 – Performance of the GSV-FA* and GSV-PA* models on a sequence of the GOPRO dataset. The red dashed line at 50% represents the same threshold as in the Figure 19. Between the frames #250 and #300 both models begin to report a crosswalk. However, the human annotator only labeled it after the frame #300. This illustrates the importance of the qualitative evaluation to support the quantitative results. The frame #275 of this dataset can be seen in the first image of the Figure 22.

As can be seen in the Table 9, the models trained using the proposed system can indeed achieve good performance results even on datasets from different sources, i.e. images with clear differences on the brightness, contrast, size, etc. Based on the results, the IARA dataset seems more difficult than the GOPRO dataset. Looking at the images, the camera used on the IARA dataset seems to be calibrated very differently than the other datasets. This difference may have lead to unexpected inconsistencies, such as the GSV-FA* achieving better results than the GSV-PA* for the IARA dataset.

Comparing the results hereby presented to the literature is very difficult. Most of the works in the literature do not release publicly the dataset used in their experimentation

(a) (b) (c) (d)

Figure 22 – Failure samples on the GOPRO dataset. The first two images are false positives, and the last two are false negative predictions. The second image is particularly difficult, because there is a crosswalk in the intersection and it does not fit into the criteria used for the manual annotation.

Table 9 – Performance overview of the best fully-automatic (FA) and partially-automatic (PA) models using both intra-database (GSV) and cross-database (IARA and GOPRO) evaluation protocols.

Model	**Dataset**	**ACC**	**ACC$_{INSTANCE}$**
GSV-FA*	GSV	94.21%	–
	IARA	93.06%	81.25%
	GOPRO	89.30%	92.31%
GSV-PA*	GSV	96.51%	–
	IARA	92.04%	89.58%
	GOPRO	93.48%	96.15%

nor their methods, therefore it is hardly possible to replicate their results. Even though, given the appropriate considerations, some results of the literature can provide context to ours. Wang et al. (2014) proposed an SVM classifier and achieved an overall accuracy of 90.98%, and 78.90% for the positive class. The evaluation of their method was performed in a small (122 images, in total) and local dataset. Riveiro et al. (2015) proposed the combination of several image processing techniques to detect crosswalks. They evaluated their method in a small (30 crosswalks) and local dataset, and achieved 83.33% of accuracy for the positive class. Poggi, Nanni and Mattoccia (2015) proposed the use of CNNs to detect crosswalks in the perspective of a person. They proposed two versions: without any refinement (88.97%) and with head pose refinement (91.59%). These results were achieved evaluating both of their models in a local and smaller (10,165 frames) dataset compared to ours. As previously discussed, most of the results in the literature rely on an evaluation made on small and local datasets. On the other hand, our models were evaluated in large-scale databases and still presented higher performance results (in absolute terms) than the literature. In addition, none of the methods in the literature performed cross-database evaluation. Nonetheless, our cross-database results also presented performance results on pair with the other experiments in the literature. It is worth noticing that both our databases (IARA and GOPRO) and the scripts to automatically acquire the GSV dataset

are publicly available to enable future fair comparisons.

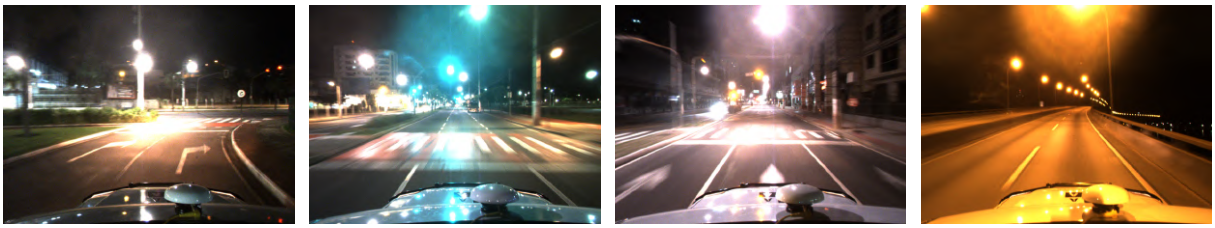
3.2.2.2.3 Results of the Qualitative Experiments

Qualitative results can be viewed on the publicly available videos: using the IARA[11] and the GOPRO[12] datasets. The videos, in total, contain the evaluation on more than 23,500 images from these two datasets. As can be seen in the videos, the model presents robustness to various factors. In addition, the failure cases are consistent, i.e. they tend to happen in some specific situations and not randomly, which indicates the need for more images of those cases during training. The two most common failure cases are when the crosswalk is too far or there are arrows or writings on the road. The first case leads to two failures: false positive, when the human annotator judged that the crosswalk was too far but the model could detect it; and false negative, when the model misses crosswalks that indeed are too far. The problem is that the definition of "too far" is subjective. In those cases, the qualitative analysis may provide useful insights. The second case leads to a much more difficult problem. Currently, there is no way to automatically retrieve images of that particular type (i.e. with arrows or writing in the lane) using the services that were used in this work. Nevertheless, as a future work, we think that we could synthetically augment the dataset by systematically adding arrows and writings to images using a procedure similar to (PAULA; JUNG; SILVEIRA, 2014). Note that several post-processing techniques could have been used to increase the performance of the system in those datasets (i.e. temporal sequence of images), such as hysteresis-based technique to remove the few noisy unstable predictions. However, these post-processing techniques were not in the scope of this work. In addition, the same procedure of automatic acquisition and annotation of large sets of images could be extended to other tasks, specially driving related tasks. Some of these extensions may require human-in-the-loop to achieve good performance (e.g., traffic light detection), but others may not. As a future work, we plan to investigate such extensions and the impact of having a human-in-the-loop. In general, qualitative analysis suggests that the models are able to perform very well in real-world scenarios.

In addition to the above mentioned experiments, we also performed an experiment with a video recorded during the night using the same camera used in the IARA dataset. The sequence comprises 12,114 frames (in a temporal sequence) covering part of the IARA dataset and more (including a toll, a bridge, etc.) during the night (Figure 23). It is important to note that only daytime images were presented to the models during the training (Google Street View only provides images in daylight). We also ensured that the regions in this night sequence were not in the training sets as well. Although not as stable as in the other datasets, the models were able to correctly predict the existence of the

[11] <https://youtu.be/zrKU3duNwuo>
[12] <https://youtu.be/jmYmQFiqY3c>

crosswalks in most of the cases, as can be seen in the publicly available video[13].



(a) (b) (c) (d)

Figure 23 – Samples of the dataset used to evaluate the models during the night.

3.2.2.2.4 Limitations

The analysis of the qualitative experiments already presented some of the limitations of the proposed system. In addition to those limitations, three others are important to mention. Firstly, the availability of crosswalk annotations in the OpenStreetMap is large but limited, i.e., the system is constrained to acquire positive samples from these locations. Systems such as the one proposed in this work can be used to cope with this fact (helping to annotate other areas) and with the inconsistencies in the data already available (helping to filter the noise). Secondly, the images are acquired from Google Street View, therefore all images are in daylight. Despite of that, experimental results show the model performed reasonably well in night sequences. Thirdly, given the cost to manually annotate large amounts of images, the manual labels were assigned by a single human annotator, i.e., the subjectivity is constrained to a single individual. To cope with that, subjectivity can be decreased at the high cost of employing a group of annotators. Even though, subjectivity would still apply. Besides that, one application of the models hereby proposed is precisely to avoid this manual labor.

3.2.3 Heading Direction Estimation

Advanced Driver Assistance Systems (ADAS) have experienced major advances in the past few years. These systems are increasing the safety of traffic and helping drivers by monitoring the surrounds, and warning or acting to avoid accidents. Three of the most common ADAS are stability control, velocity control, and lane keeping. These technologies are also part of systems that will be applied in fully autonomous cars. The main objective of ADAS includes keeping the vehicle in the correct road direction, and avoiding collision with other vehicles or obstacles around. For this purpose, it is important to estimate the current localization of the car in relation to the lane or road. A common representation of the robot localization is the position and orientation, e.g., a 2D location (x, y) and its

[13] <https://youtu.be/afCBi1Pj1NE>

heading direction (θ) in relation to the world representation.

In this section, we address the problem of heading direction estimation that keeps the vehicle in the road direction. Not only essential for ADAS as well as for autonomous cars navigation, this information can be used in precise localization, road and lane keeping, driver snooze warning (ZHANG et al., 2015), lane change detection (WANG; MURPHEY; KOCHHAR, 2016), lane departure warning (BERRIEL et al., 2017), and others. We propose an end-to-end approach that uses a DNN to estimate the heading direction angle in relation to the road, using only a forward-looking front-view camera raw image as input. The proposed system leverages public available platforms to automatically acquire and annotate a large-scale database. Then, this database is used to train a CNN model end-to-end to perform the task of interest. The proposed approach was evaluated in an intra-database test and in a cross-database test, using an urban road dataset from our autonomous car (IARA). The experiments showed that our model can estimate the heading direction to maintain the car aligned to the road using only raw images as input with a mean absolute error (MAE) of 2.359° for the intra-database experiment and 2.524° of MAE in the cross-database experiments, on average.

The proposed system is two-fold: *i)* the automatic data acquisition and annotation of a large-scale database; and *ii)* training and testing a Convolutional Neural Network (CNN) to estimate the heading direction. The system receives as input a single image and predicts the difference ($\Delta\theta$) in the current car orientation (θ) and the "ideal" car orientation 4 meters ahead (θ_{+4}) in the road. The automatic data acquisition and annotation exploits online mapping platforms (such as Google Street View and OpenStreetMap) to acquire road definition points and imagery of a set of regions. The annotated large-scale database is, then, fed into a CNN model that estimates the required $\Delta\theta$ to keep the vehicle aligned to the road. The overview for the proposed system is presented in the Figure 24.

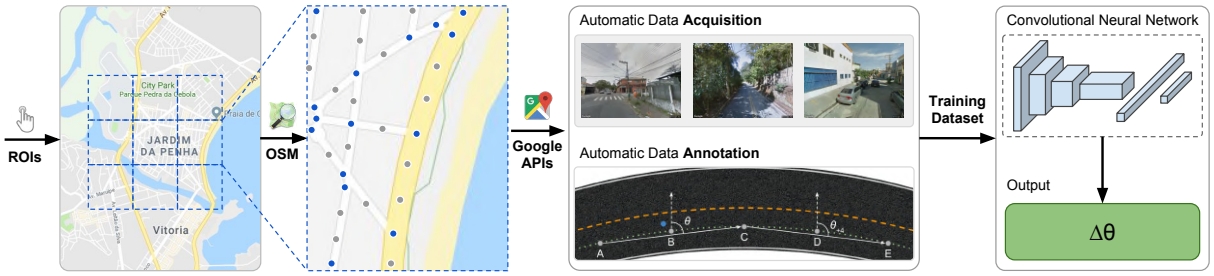


Figure 24 – Overview of the proposed system. Regions of interest (ROIs) are given to the OpenStreetMap (OSM) that returns a list of points (blue circles) for each street in the ROIs. A linear interpolation is applied to generate new samples every 2 meters (gray circles). All the points are used to request images of the streets using the Google Street View API. After that, an automatic annotation process is used to generate the training dataset of our model (a ConvNet). The model predicts the $\Delta\theta$.

Automatic Data Acquisition and Annotation. In the past few years, CNNs

have become very popular to solve different kinds of problems, but a major drawback is that they need large amounts of labeled data which may be difficult and expensive to obtain. To overcome this issue, inspired by (BERRIEL et al., 2017), publicly available platforms are exploited to automatically acquire and annotate a large-scale database. Firstly, the data acquisition uses OpenStreetMap via the Overpass API to obtain road definition points from a set of regions. Secondly, images from the streets are acquired using Google Street View Image APIs. Lastly, an automatic labeling process is applied to generate the dataset.

The OpenStreetMap via the Overpass API receives an input region and outputs the lists of roads within that region, where a road is represented by a list of sequential points (latitude and longitude). To cope with an API limitation, only rectangular regions with sides smaller than 1/4 degrees (≈ 27.75 km) were used. The list of points the API outputs is not evenly distributed along the roads. Therefore, each road (sequence of points) is upsampled (using linear interpolation) to enforce a ≈ 2 meters distance between two consecutive points. Points outside the region of interest are discarded.

In order to acquire the images, three steps are performed: *i)* for each point on the road, the location of the closest image is retrieved by requesting it to the Google Street View Image Metadata API; then, *ii)* for each point on the road, the angle (θ, based on the geodetic north) between its predecessor (2m behind) and its successor (2m ahead) is computed, assuming it represents the direction of the road at that point; finally, *iii)* an image can be requested to the Google Street View Image API using the location acquired in the first step and using the angle (heading) computed in the second step. As the database needs to have images associated with different angles, a random noise $\alpha \sim \mathcal{U}(-25°, 25°)$ is added to θ when requesting the image in the third step. Therefore, the final heading angle of a given image is equal to $\theta + \alpha$. In addition, to increase to robustness of the model in respect to vertical oscillations, the image is requested with a random pitch $\gamma \sim \mathcal{U}(-10°, 10°)$.

An example of the angle calculation process can be seen in the Figure 25, where the gray points are the road definition points (labeled A-E), and the blue circle represents an image position. For example, point B is the closest one to an image (blue circle) and the previous and next points 2m away (A and C, respectively) form the vector $\overrightarrow{AC}$. In this context, θ is given by the orientation of $\overrightarrow{AC}$ in respect to the True North.

To generate a dataset to the task of interest, it is required to annotate all images, i.e., to associate every image with its corresponding $\Delta\theta$ (not the θ itself). Therefore, in possession of the vector $\overrightarrow{AC}$, the same process is performed to the point 4m ahead of the point of interest (point B, in this example), which is the point D. Using the vector $\overrightarrow{CE}$, the "ideal" angle 4 meters ahead (θ_{+4}) can be calculated, as shown in Figure 25. As a result, the $\Delta\theta$ associated with the image represented by the blue circle can be defined as

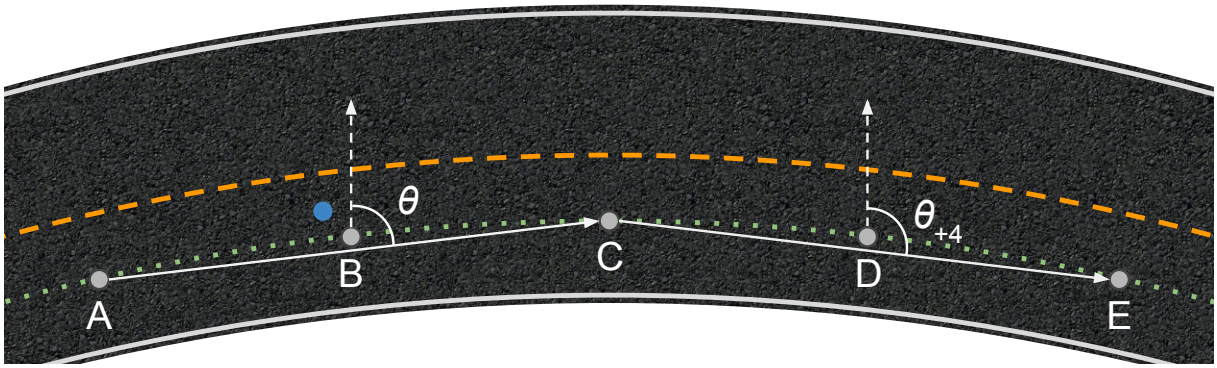


Figure 25 – Angle calculation. The gray circles A, B, C, D and E are road definition points with 2 meters of distance between each other. The blue circle represents an image location and B its closest point. The angle to acquire a forward-looking image is denoted as θ, derived from $\overrightarrow{AC}$ (upwards dashed arrows represent the geodetic north). The "ideal" angle 4 meters ahead is denoted as θ_{+4}, derived from $\overrightarrow{CE}$. The difference between θ and θ_{+4} gives $\Delta\theta$, i.e., how much the heading direction should change to keep the car aligned 4 meters ahead.

$\Delta\theta = \theta_{+4} - \theta - \alpha$, where α is the random noise added when requesting the image. This process is performed for every image in the dataset.

Deep Neural Network for Heading Estimation. In possession of a annotated dataset, i.e., a set of images and the corresponding difference in orientation ($\Delta\theta$) the car must achieve 4m ahead, it is feasible to train a model. In this work, a Convolutional Neural Network (CNN) was chosen. During the training, the CNN receives an image as input and predicts the $\Delta\theta$. The architecture of the CNN was inspired in the AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) with some modifications, such as using less neurons on the fully-connected layers; using Batch Normalization instead of Local Response Normalization; replacing the output layer by just one neuron with linear activation function to perform the regression; and changing the input size to 128×128 pixels.

The dataset was split into 3 sets: train, validation and test. None of the regions used in the test set were seen during training. Details of the datasets are presented in the Section 3.2.3.1. During training, all images were downsampled to the input size of the model: 128×128 pixels. In addition, the images were converted to grayscale mainly for two reasons: *i)* color is assumed not to be important for this task, and *ii)* performance is a must (less channels means less computation) for this kind of application. The weights were randomly initialized using (GLOROT; BENGIO, 2010). The objective of the training procedure was to minimize the Mean Squared Error (MSE). For that, the Adam optimizer (KINGMA; BA, 2015) (with $\beta_1 = 0.9$ and $\beta_1 = 0.999$) was used with initial learning rate equal to 10^{-3}, decaying by 10 every time the validation loss stops improving for 3 consecutive epochs. Moreover, the model was trained until the validation loss reached a plateau for 10 consecutive epochs. After training, the network is ready for inference and

experiments.

3.2.3.1 Experiments

In order to validate the proposed system, two experiments are carried out. Each experiment was evaluated in a database that was collected automatically using the proposed system, and described below. Moreover, the metric used in the experimentation is shown. Finally, the experiments and setup are presented in details.

3.2.3.1.1 Datasets

To validate the proposed system, the experiments were performed using two datasets: Google Street View (GSV) and IARA. The GSV dataset was used to train the model responsible for the prediction and to validade the performance of the system. The IARA dataset was used exclusively for performance evaluation. Both datasets are described below.

Google Street View (GSV). The GSV dataset was generated automatically using the system previously described. Therefore, the dataset contains images and the corresponding $\Delta\theta$ that will guide the vehicle to the heading direction of the road. The images were collected from several cities in Brazil, acquiring images from every single road within a region of interest. These paths include samples of highways, paved roads with and without lane markings, and unpaved roads in both urban and rural areas; different road widths, number of lanes, lane marking and curvatures; streets with strong shadows; and many other scenarios. Some samples can be seen in Figure 26.

(a) (b) (c)

Figure 26 – Sample images of the GSV dataset.

The dataset has a total of 1,035,524 images of 640×480 pixels and was split into train, validation, and test sets. The train set has 782,123 images of nine different regions in five different states of Brazil (Manaus-AM, Recife-PE, Rio de Janeiro-RJ, Salvador-BA, São Paulo-SP, Piracicaba-SP, and Teresina-PI). Even though a region is named after a city, some of them include parts of neighbor cities as well. The validation set has 45,534 images of one region (Natal-RN). The test set has 207,867 images from three different

cities in three different states (Maceió-AL, Porto Alegre-RS, Vitória-ES). The region of Vitória-ES was specifically chosen to be in the test set, because of the experiments using IARA dataset that was recorded there.

IARA dataset. Another dataset was acquired using the Intelligent Autonomous Robotic Automobile (IARA), shown in Figure 27, that is developed by the High Performance Computing Laboratory (LCAD, Laboratório de Computação de Alto Desempenho in Portuguese). IARA's software is composed of many modules and the five main ones are: the Mapper (MUTZ et al., 2016) computes a map of the environment around IARA; the Localizer (VERONESE et al., 2015; VERONESE et al., 2016) estimates IARA's state (position $\{x, y\}$ and orientation $\{\theta\}$), relative to the origin of the map; the Motion Planner (CARDOSO et al., 2017) computes a trajectory from the current IARA's state to the next goal state; the Obstacle Avoider (GUIDOLINI et al., 2016) verifies and eventually changes the current trajectory in case it becomes necessary to avoid a collision; the Controller (GUIDOLINI et al., 2017) converts the control commands of the trajectories into acceleration, brake and steering efforts to actuate in IARA's hardware.

Figure 27 – Intelligent Robotic Autonomous Automobile (IARA).

IARA's localizer module computes the car orientation relative to the map, and the map orientation is aligned to world, therefore directly providing the car orientation (θ) computed approximately at 20Hz (i.e., every 50ms). The difference in orientation the car must achieve 4m ahead ($\Delta\theta$) is obtained in the same way it is done for the training dataset: the difference between the orientation (θ) of the closest point of an image and the orientation (θ_{+4}) of the point 4m ahead. The only difference, though, is that in this case the orientations are computed by IARA's Localizer, directly from sensors data. After performing this process in every image, the dataset is automatically annotated.

The IARA dataset was built to evaluate the robustness of the model to different images from a different camera, with different field-of-view, and other factors. To generate the dataset, two routes were covered (both routes are in the same city, Vitória-ES, and

can be seen in Figure 31). The first route has 6.2 km in an urban environment that crosses avenues with single and multiples lanes and neighborhood areas with narrow streets. The second route is the ring-road of the Universidade Federal do Espírito Santo (UFES) main campus with 3.5 km. This route has sharp and wide curves, varying road widths and variations in road pavement, such as cobbles and asphalt, most of it without lane marking.

To automatically annotate this dataset, a human driver had to drive IARA through each route twice. In the first lap (i.e., ground-truth lap), the car is carefully driven trying to keep the vehicle aligned to the road direction. This first lap is used as ground truth for indicating the road direction. In the second lap (i.e., maneuver lap), in order to generate more variability, the driver intentionally performs maneuvers like sudden lane changes and abrupt turns. These maneuvers cause the heading direction of the car to differ from the heading direction of the road, therefore creating a rich dataset. Finally, to compute the $\Delta\theta$ for each image in the maneuver lap, the ground-truth and the maneuver laps had to be synchronized. Since the maneuver lap does not necessarily point to the direction of the lane, it is not reliable to be used to calculate the ground truth angle 4m ahead (θ_{+4}). Therefore, they are synchronized so that the corresponding θ_{+4} can be calculated for every position in the maneuver lap (see Figure 28). The ground truth calculation is as follows. For each image of the maneuver lap, the closest pose of the car (B_{ma}, point B in the maneuver lap) is used as the location of the image. Then, the θ for the B_{ma} position is calculated using the information of the maneuver lap following the procedure illustrated in the Figure 25. The corresponding θ_{+4} is calculated using the ground-truth lap. Therefore, the point B_{gt} in the ground-truth lap (i.e., the closest point to B_{ma}) is used to find the point D_{gt} that is 4m ahead in the ground-truth lap. The θ_{+4} of D_{gt} is calculated using the information of the ground-truth lap following the procedure illustrated in Figure 25. Finally, $\Delta\theta$ is computed using θ of B_{ma} and θ_{+4} of D_{gt}.

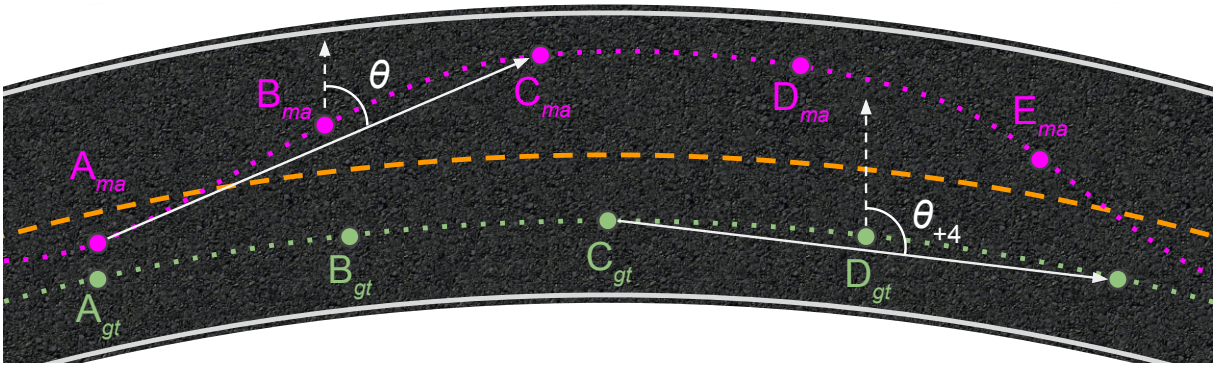


Figure 28 – Ground-truth calculation in the IARA dataset. The green and pink dotted lines are the ground-truth and maneuver laps, respectively. The circles A, B, C, D and E are road definition points with 2 meters of distance between each other. The ground-truth ($\Delta\theta$) for a point (e.g., B_{ma}) is given by θ of B_{ma} and θ_{+4} of D_{gt}, where D_{gt} is the point 4m ahead of the closest point of B_{ma} in the ground-truth lap (i.e., B_{gt}).

(a) (b) (c)

Figure 29 – Sample images of the IARA dataset (first route).

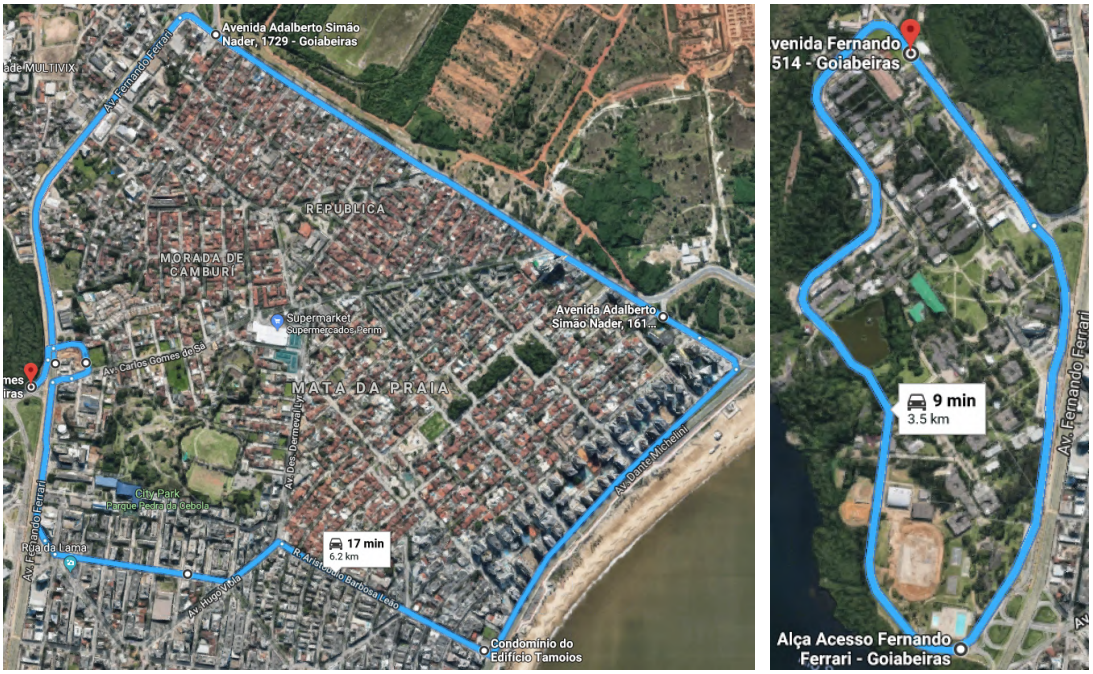
(a) (b) (c)

Figure 30 – Sample images of the IARA dataset (second route).

Some samples of the IARA dataset can be seen in the Figure 29. The images were collected in daylight on a week day with usual traffic. The first route has a total of 10,040 images. The second route has a total of 7,182 images. Both route images were recorded by a ZED stereo camera installed in the back of the rearview mirror of IARA. Images of IARA dataset were recorded at approximately 15 frames per second with 1920×1080 pixels, each. As the images from the GSV dataset have a ratio of $4 : 3$, a centralized crop of 1440×1080 pixels was performed in the original images from the ZED camera to achieve the same ratio. All three datasets (GSV and the two routes of the IARA dataset) are very different between one another (see Figure 26, Figure 29, and Figure 30): differences in brightness, contrast, position, field of view, and others. All these factors make this dataset more challenging.

3.2.3.1.2 Experiments

In order to evaluate the proposed system, two experiments were performed. The experiments are differentiated by the evaluation protocol: *i)* intra-database, where the model is evaluated in part of the same database used during the training (train and test regions are mutually exclusive); and *ii)* cross-database, where the model is trained in one database and evaluated in a different one. In both experiments, the model was only trained with the train set of the GSV dataset.



(a) (b)

Figure 31 – Aerial view of the first (a) and second (b) routes of the IARA dataset.

In the intra-database experiment, the trained model was used to predict the heading direction in the test set of the GSV dataset. Then, the heading direction estimated by the model is compared with the ground truth of the GSV dataset test set automatically acquired. In the cross-database method, the same model used in the intra-database experiment is evaluated in the datasets from IARA. This aims to evaluate the resulting model in a real application, given that a model is likely to perform (or expected to be robust, at least) on a different and unpredictable new setting: camera height, field-of-view, brightness, contrast, etc.

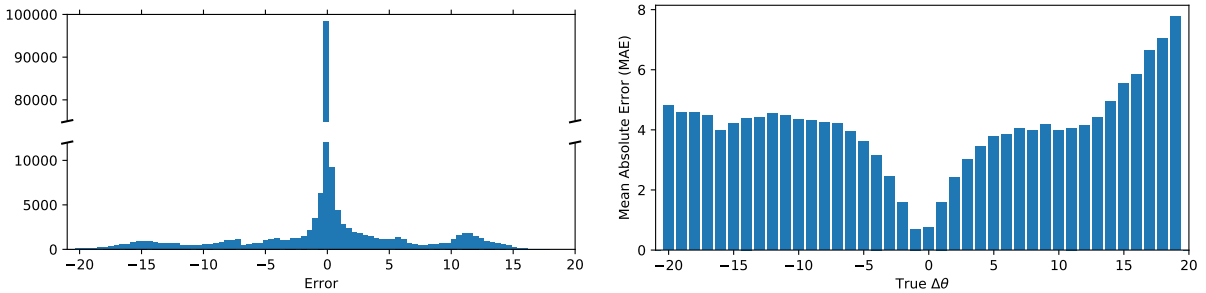
3.2.3.1.3 Setup

The experiments were carried out in an Intel Core i7-4770 3.40 GHz with 16GB of RAM, and a Titan Xp GPU with 12GB of memory. The machine was running Linux Ubuntu 16.04 with NVIDIA CUDA 9.0 and cuDNN 7.0. The training and inference steps were performed using Keras (CHOLLET et al., 2015) with Tensorflow 1.4 (ABADI et al., 2015) as the backend.

3.2.3.2 Results

The first experiments were performed by both training and testing on the GSV dataset. Training and test sets were region-wise mutually exclusive, i.e., none of the regions used for training were in the test sets. The results are reported in terms of MAE. The

MAE for the GSV dataset was equal to 2.359°. As can be seen in Figure 32a that depicts the frequency of the errors, most of the errors are close to zero.

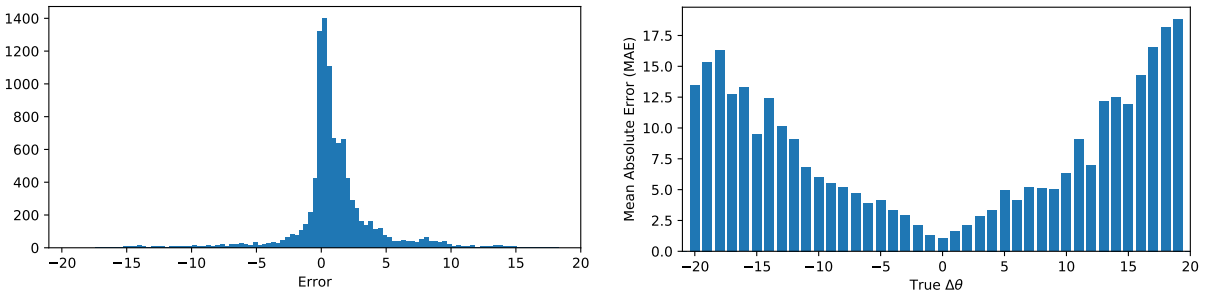


(a) (b)

Figure 32 – Results for the GSV dataset: *a)* presents the frequency of the errors (in °) and *b)* the MAE (in °) per true $\Delta\theta$ (in °).

Besides the frequency of the errors, it can be seen in Figure 32b that the MAE is stable for every heading angle to be predicted, except for the angles that are closer to zero. This indicates that the model is more reliable closer to zero, where most of the driving activity usually happens.

In the second experiment, the same model was evaluated using IARA datasets. None of the regions of the IARA datasets were used to train the model. As can be seen in Figure 33a, the frequency of the error of the first route is also larger around zero. Furthermore, despite the differences between the images of the GSV dataset, the MAE also tends to be smaller near zero (see Figure 33b). The mean average error on the first route was 2.358°.

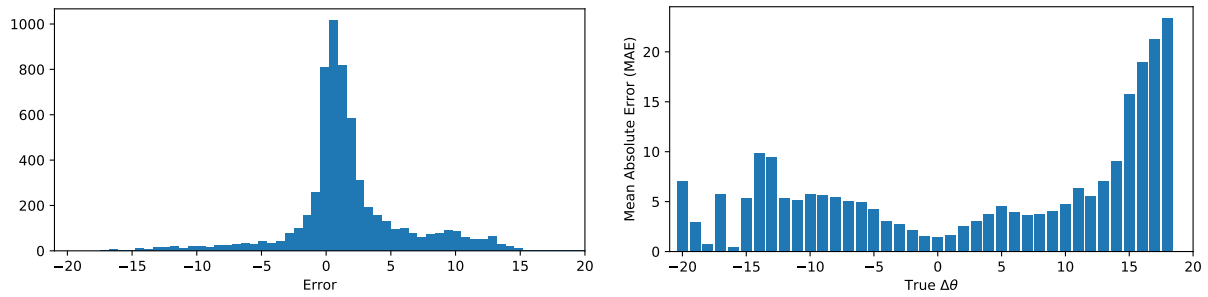


(a) (b)

Figure 33 – Results on the first route of the IARA dataset: *a)* presents the frequency of the errors (in °) and *b)* the MAE (in °) per true $\Delta\theta$ (in °).

The distribution of the error of the second route (see Figure 34a) is on par with the results of both GSV dataset and the first route of the IARA dataset. This fact indicates that the model is robust to several factors that vary between these datasets. In Figure 34b, it can be seen that the model struggled to predict $\Delta\theta$ greater than 15. However, despite the fact that most of the second route has no asphalt and no lane markings on the street

(only cobble), the mean average was 2.756°. On average, the model achieved a MAE of 2.524° in the IARA experiments.



(a) (b)

Figure 34 – Results on the second route of the IARA dataset: *a)* presents the frequency of the errors (in °) and *b)* the MAE (in °) per true $\Delta\theta$ (in °).

Another important fact that is worth noting is that, unlike (BOJARSKI et al., 2016), we neither balanced the training set with road curves nor excluded lane changes and turns from one road to another. Both these changes could improve the performance of the model even further. Even though there are other works related to ours, fair comparisons are difficult for many reasons: *i)* the variables of interest are different; *ii)* oftentimes, the databases are private (i.e., we cannot run our experiments on their data); and *iii)* neither the models nor a way to reproduce their work are provided (i.e., we cannot run their models on our data). To enable future fair comparisons, we released our database (first and second routes of the IARA dataset) and the pre-trained models.

Qualitative results on the IARA datasets can be seen in the videos of the first[14] and second[15] routes. As can be seen in the video, the images are very different, there are many lane changes, turns from one road to another and intersections. In addition, the dataset was recorded with the usual traffic of a weekday. A top-view is also depicted in the videos to facilitate the quality assessment. Moreover, the model can predict in more than 75 frames per second using the GPU, i.e., the system is suitable for real-time applications.

3.3 Discussion & Conclusion

There are many interesting results and insights in the three aforementioned contributions. In the first one, the satellite crosswalk classification problem, the results showed that exploiting online platforms is an effective alternative to reduce the cost of acquiring and annotating satellite imagery w.r.t crosswalks. They also showed that, for this problem, the automatic processes have a low labeling error and that it is possible to have a reliable global-scale model trained by this automatically acquired and annotated data. In the

[14] <https://youtu.be/htl_eJP5oMo>
[15] <https://youtu.be/XXckftkJLGc>

second one, the street view crosswalk classification, the results showed that, despite of the larger error of the automatic process, it still can be used to effortlessly acquire and annotate datasets that are useful. The models trained on this large-scale dataset also presented robustness to several factors (e.g., different cameras, during the night, etc.). Lastly, in the heading direction estimation problem, we observed that similar techniques can be used to automatically acquire and annotate data to solve a regression problem (compared to the previous two classification ones). Moreover, the proposed techniques helped to solve the data imbalance problem that would have happened otherwise (e.g., by driving around in a car), which would likely degrade the generalization of the neural network.

Future works. There are interesting works one can build on top of what we proposed that could also help to cope with the limitations. For instance, the process described in Figure 12 is very simple and it attempts to reduce noise by avoiding low-density regions. Although effective, a better heuristics to select dense regions, preferably a non-parametric one, could help to reduce the noise even further and improve the performance of the models trained on a potentially cleaner dataset. Another interesting direction is to exploit the model trained for satellite crosswalk detection to train on non-annotated regions. These models have shown to be robust in the cross-dataset experiments, therefore they could be useful to bypass the limitation of the proposed approach in which training is only possible where OSM data is available. Two more future works are related to the heading direction estimation problem. First, we proposed a model that directly predicts the change in heading to align the car to the road, however, it could be useful to not only regress this value but have some confidence associated with it. To add this feature, one could simply extend our model by training it with a quantile loss, thus performing quantile regression instead. Moreover, we assess the performance of our model using an image-based metric. However, it could be useful to bring the temporal aspect of the problem into the evaluation protocol by considering the aggregate error over time. One could extend it even further by proposing a metric that encourages smoother driving behavior, which is usually associated with a better driving experience.

These findings validate our hypotheses. They support the argument that automatic large-scale data acquisition and annotation are useful for training deep neural networks to solve real-world problems. Moreover, our results show that the large scale and diversity of these data sets reduce the impact of the inherent error of these automatic processes, even when trained naively. In summary, the proposed methods are good alternatives for almost eliminating the human effort from the data acquisition and annotation processes, thus greatly reducing the cost of training deep neural networks for the investigated problems.

4 Multi-Domain Learning

Training deep neural networks does not only require massive datasets, as previously discussed, but also a lot of computational power. In many cases, training deep learning models require multiple GPUs, sometimes even clusters with them, and not all research groups and companies can afford this infrastructure. Moreover, in most cases, these models are deployed in an environment with limited access to memory and computation, and they will have to compete for resources with other applications. This is particularly concerning when a model has to achieve high recognition accuracy on several different domains. Therefore, in this thesis, we investigate how to reduce the computational costs of a deep learning model at inference time in the context of multiple domains. We hypothesize that a generic image representation can be exploited to reduce the computational requirements of a model when learning additional domains. Furthermore, we hypothesize that exploiting this feature can also allow a model with an adjustable budget at inference time, providing a trade-off between performance and model complexity. In this chapter, we first introduce the multi-domain learning problem and how we approached it. Then, a review of the relevant literature is presented. Next, the proposed method is detailed. After that, we describe the experiments we used to validate our method together with their results. Finally, we conclude by summarizing the key findings.

4.1 Introduction

Deep learning methods have brought revolutionary advances in computer vision, setting the state of the art in many tasks such as object recognition (HE et al., 2016; KRIZHEVSKY; SUTSKEVER; HINTON, 2012), detection (GIRSHICK et al., 2014), semantic segmentation (CHEN et al., 2018), depth estimation (XU et al., 2017), and many more. Despite these progresses, a major drawback with deep architectures is that when a novel task is addressed typically a new model is required. However, in many situations it may be reasonable to learn models which perform well on data from different domains. This problem, referred as Multi-Domain Learning (MDL) and originally proposed in (REBUFFI; BILEN; VEDALDI, 2017), has received considerable attention lately (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2018). An example of MDL is the problem of image classification when the data belong to several domains (e.g., natural images, paintings, sketches, etc.) and the categories in the different domains do not overlap.

Previous MDL approaches (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2017; REBUFFI; BILEN; VEDALDI, 2018)

utilize a common backbone architecture (i.e., a pre-trained model) and learn a limited set of domain-specific parameters. This strategy is advantageous with respect to building several independent classifiers, as it guarantees a significant saving in terms of memory. Furthermore, it naturally deals with the *catastrophic forgetting* issue, as when a new domain is considered the knowledge on the previously learned ones is retained. Existing approaches mostly differ from the way domain-specific parameters are designed and integrated within the backbone architecture. For instance, binary masks are employed in (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018) in order to select the parameters of the main network that are useful for a given task. Differently, in (REBUFFI; BILEN; VEDALDI, 2017; REBUFFI; BILEN; VEDALDI, 2018) domain-specific residual blocks are embedded in the original deep architecture. While the different approaches are typically compared in terms of classification accuracy, their computational and memory requirements are not taken into account.

In this work we argue that an optimal MDL algorithm should not only achieve high recognition accuracy on all the different domains but should also permit to keep the number of parameters as low as possible. In fact, as in real world applications the number of domains and tasks can be very large, it is desirable to limit the models' complexity (both in terms of memory and computation). Furthermore, it is very reasonable to assume that different domains and tasks may correspond to a different degree of difficulty (e.g., recognizing digits is usually easier than classifying flowers) and may require different models: small networks should be used for easy tasks, while models with a large number of parameters should be employed for difficult ones.

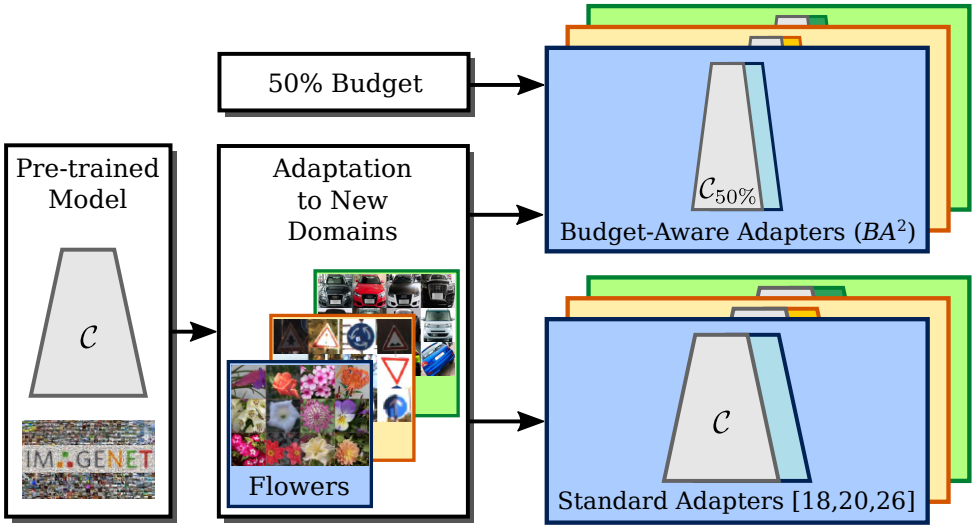


Figure 35 – In Multi-Domain Learning, a pre-trained model is usually adapted to solve new tasks in new domains. When using standard approaches, the complexity $\mathcal{C}$ of the domain-specific models is dependent on the pre-trained model complexity. In this work we propose a novel approach to learn specialized models while imposing budget constraints in terms of the number of parameters for each new domain.

Following these ideas, we propose the first MDL approach which derives a set of domain-specific classifiers from a common backbone deep architecture under a budget constraint, where the budget is specified by a user and is expressed as the number of network parameters (see Figure 35). This idea is realized by designing a new network module, called *Budget-Aware Adapters* (BA^2), which embeds switch variables that can select the feature channels relevant to a domain. By dropping feature channels in each convolutional layer, BA^2 both adapt the image representation of the network and reduce the computational complexity. Furthermore, we propose a constrained optimization problem formulation in order to train domain-specific classifiers that respect budget constraints provided by the user. The proposed approach has been evaluated on two publicly available benchmarks, the ten datasets of the Visual Decathlon Challenge (REBUFFI; BILEN; VEDALDI, 2017) and the six-dataset benchmark proposed in (MALLYA; DAVIS; LAZEBNIK, 2018). Our results show that the proposed method is competitive with state-of-the-art baselines and requires much less storage and computational resources.

4.2 Literature Review

Multi-domain Learning. The problem of adapting deep architectures to novel tasks and domains has been extensively studied in the past. Earlier works considered simple strategies, such as fine-tuning existing pre-trained models, with the drawback of incurring to *catastrophic forgetting* and of requiring the storage of multiple specialized models. More recent studies address the problem proposing methods for extending the capabilities of existing deep architectures by adding few task-specific parameters. In this way, as the parameters of the original network are left untouched, the catastrophic forgetting issue is naturally circumvented. For instance, Rebuffi, Bilen and Vedaldi (2017) introduced residual adapters, i.e., a novel design for residual blocks that embed task-specific components. In a subsequent work (REBUFFI; BILEN; VEDALDI, 2018), they proposed an improved architecture where the topology of the adapters is parallel rather than series. Rosenfeld and Tsotsos (2018) employed controller modules to constrain newly learned parameters to be linear combinations of existing ones. Weight-based pruning has been considered in (MALLYA; LAZEBNIK, 2018) to adapt a single neural network to multiple tasks. Aiming at decreasing the overhead in terms of storage, more recent works proposed to adopt binary masks (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018) as task-specific parameters. In particular, while in (MALLYA; DAVIS; LAZEBNIK, 2018) simple multiplicative binary masks are used to indicate which parameters are and which are not useful for a new task, Mancini et al. (2018) propose a more general formulation considering affine transformations. Guo et al. (2019) proposed an adaptive fine-tuning method and derive specialized classifiers by fine-tuning certain layers according to a given target image.

While these works considered a supervised learning setting, the idea of learning task-specific parameters has also been considered in reinforcement learning. For instance Rusu et al. (2016) proposed an approach where each novel task is addressed by adding a side branch to the main network. While our approach also aims at developing architectures which adapts a pretrained model to novel tasks, we target for the first time the problem of automatically adjusting the complexity of the task-specific models.

Incremental and Life-long learning. In the last few years several works have addressed the problem of incremental (BENDALE; BOULT, 2016; REBUFFI et al., 2017) and life-long learning (ALJUNDI et al., 2018; KIRKPATRICK et al., 2017; LI; HOIEM, 2017), considering different strategies to avoid catastrophic forgetting. For instance, Li and Hoiem (2017) proposed to adopt knowledge distillation to ensure that the model adapted to the new tasks is also effective for the old ones. Kirkpatrick et al. (2017) demonstrated that a good strategy to avoid forgetting on the old tasks is to selectively slow down learning on the weights important for those tasks. Aljundi et al. (2018) presented Memory Aware Synapses, where the idea is to estimate the importance weights for the network parameters in an unsupervised manner in order to allow adaptation to unlabeled data stream. However, while these works are interested in learning over multiple tasks in sequence, in this work we focus on a different problem, i.e., re-configuring an existing architecture under some resource constraints.

Adaptive and Resource-aware Networks. The problem of designing deep architectures which allow an adaptive accuracy-efficiency trade-off directly at runtime has been recently addressed in the research community. For instance, Wu et al. (2018) proposed BlockDrop, an approach that learns to dynamically choose which layers of a Residual Network to drop at test time to reduce the computational cost while retaining the prediction accuracy. Wang et al. (2018) introduced novel gating functions to automatically define at test time the computational graph based on the current network input. Slimmable Networks have been introduced in (YU et al., 2019) with the purpose of adjusting the network width according to resource constraints. While our approach is inspired by these methods, in this work we show that the idea of dynamically adjusting the network according to resource constraints is especially beneficial in the multi-domain setting.

4.3 Budget-Aware Adapters for Multi-Domain Learning

In Multi-Domain Learning (MDL), the goal is to learn a single model that can work for diverse visual domains, such as pictures from the web, medical images, paintings, etc. Importantly, when the visual domains are very different, the model has to adapt its image representation. To address MDL, we follow the common approach (MALLYA; DAVIS; LAZEBNIK, 2018; REBUFFI; BILEN; VEDALDI, 2017) that consists in learning

Convolutional Neural Networks (ConvNets) that share the vast majority of their parameters but employ a very limited number of additional parameters specifically trained for each domain.

Formally, we consider an arbitrary pre-trained ConvNet $\Psi_0(\cdot;\theta_0) : \mathcal{X} \rightarrow \mathcal{Y}_0$ with parameters θ_0 that assigns class labels in $\mathcal{Y}_0$ to elements of an input space $\mathcal{X}$ (e.g., images). Our goal is to learn for each domain $d \in \{1, \ldots, D\}$, a classifier $\Psi_d(\cdot;\theta_0,\theta_a^d) : \mathcal{X} \rightarrow \mathcal{Y}_d$ with a possibly different output space $\mathcal{Y}_d$ that shares the vast majority of its parameters θ_0 but exploits additional domain-specific parameters θ_a^d to adapt Ψ_d to the domain d.

In this work, we claim that an effective approach for MDL should require a low number of domain-specific parameters. In other words, the cardinality of each θ_a^d parameter set should be negligible with respect to the cardinality of θ_0. In addition, we argue that one major drawback of previous MDL methods is that the network computational complexity directly ensues from the initial pre-trained network Ψ_0. More precisely, the networks Ψ_d for the new domains usually have computational complexities at best equal to the one of the initial pre-trained network. Moreover, such models lack flexibility for deployment since the user cannot adjust the computational complexity of Ψ_d depending on its needs or on hardware constraints.

To address this issue, we introduce novel modules, the *Budget-Aware Adapters* (BA^2) that are designed both for enabling a pre-trained model to handle a new domain and for controlling the network complexity. The key idea behind BA^2 is that the parameters θ_a^d control the use of the convolution operations parameterized by θ_0. Therefore, BA^2 can learn to drop parts of the computational graph of Ψ_0 and parts of the parameters θ_0 resulting in a model Ψ_d with a lower computational complexity and fewer parameters to load at inference time. In the following, we first describe the proposed *Budget-Aware Adapters* (Subsection 4.3.1) and then present the training procedure we introduced to learn domain-specific models with budget constraints (Subsection 4.3.2).

4.3.1 Adapting Convolutions with BA^2

We now describe our *Budget-Aware Adapters* illustrated in Figure 36. Since, BA^2 acts on the elementary convolution operation, it can be employed in any ConvNet but, for the sake of notation simplicity, we consider the case of 2D convolutions. Let $\boldsymbol{K} \in \mathbb{R}^{2K_H+1 \times 2K_W+1 \times C}$ be a kernel of a standard convolutional layer of Ψ_0. Here $2K_H + 1$ and $2K_W + 1$ denote the kernel size and C the number of input channels. Note that $\boldsymbol{K}$ is a subset of the parameters θ_0 introduced in Chapter 4. Considering a standard 2D convolution, an input feature map $\boldsymbol{I} \in \mathbb{R}^{H \times W \times C}$ and an activation function g, the output

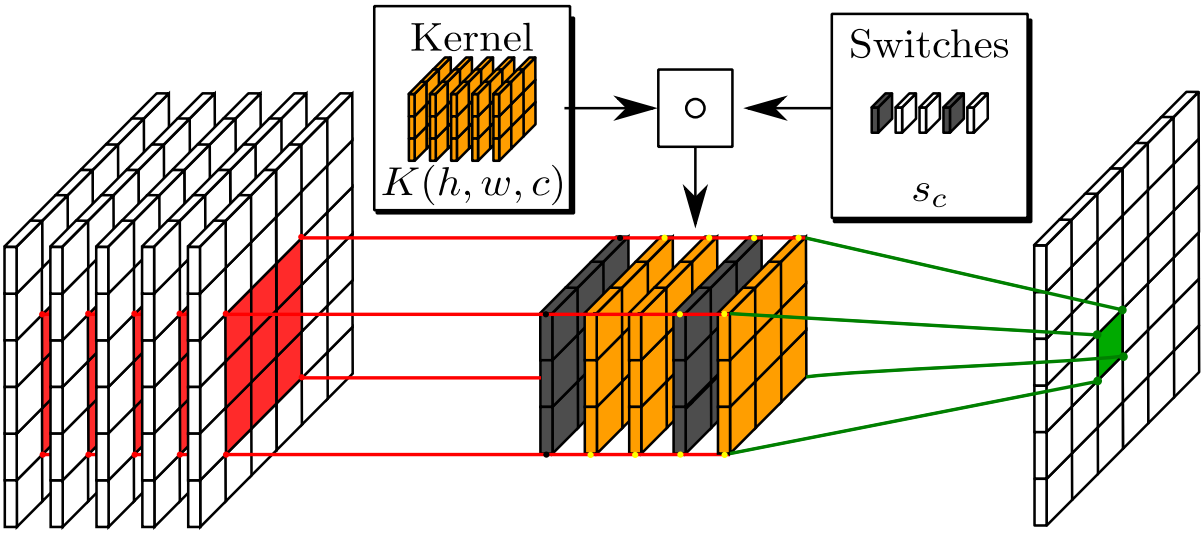


Figure 36 – Budget-Aware Adapters (BA^2): a switch vector controls the activation of convolution channels in order to both adapt the network to a new domain and adjust its computational complexity. Dark grey arrays represents channels that are "turned off" by the switches.

value at the location $(i, j) \in [1..H] \times [1..W]$ is given by:

$$\boldsymbol{x}(i,j) = g\Big(\sum_{c=1}^{C} \phi_c(i,j)\Big), \tag{4.1}$$

where ϕ_c is given by:

$$\phi_c = \sum_{h=-K_h}^{K_h} \sum_{w=-K_w}^{K_w} \boldsymbol{K}(h,w,c)\boldsymbol{I}(i-h, j-w, c). \tag{4.2}$$

For the sake of simplicity, the kernel parameter tensor $\boldsymbol{K}$ is indexed from $-K_h$ to K_h and from $-K_w$ to K_w. When learning a new domain d, we propose to adapt the convolution by controlling the use of each channel of the convolution layer. To this aim, we introduce an additional binary switch vector $s \in \{0,1\}^C$. This vector s is a subset of the θ_a^d introduced in Chapter 4. As shown in Figure 4.3.1, each switch value is used for an entire channel. As a consequence, BA^2 results in a limited number of additional parameters. Formally, the output of the adapted convolution at location (i, j) is given by:

$$\boldsymbol{x}(i,j) = g\Big(\sum_{c=1}^{C} s_c \phi_c(i,j)\Big) \tag{4.3}$$

Note that, when $s_c = 0$, the tensor ϕ_c in Equation (4.3) does not need to be computed. In this context, by adjusting the proportion of zeros in s_c, we can control the computational complexity of the convolution layer. Furthermore, in Equation (4.2), when ϕ_c is not computed, the kernel weights values $\boldsymbol{K}(h,w,c)$ can be removed from the computational graph and do not need to be stored. Therefore, in scenarios where the parameters $\boldsymbol{K}$ of the initial networks are not further used, these $\boldsymbol{K}(h,w,c)$ weights can be dropped resulting in

a lower number of parameters to store. Thus, s_c can also control the number of parameters of the new domain networks.

In order to obtain a model that can be trained via Stochastic Gradient Descent, we follow (KAISER et al., 2018; MALLYA; DAVIS; LAZEBNIK, 2018) and obtain binary values using a threshold function τ:

$$s_c = \tau(\tilde{s}_c) = \begin{cases} 0 & \tilde{s}_c \leq 0.0 \\ 1 & \text{otherwise} \end{cases} \tag{4.4}$$

where $\tilde{s}_c \in \mathbb{R}$ are continuous scalar parameters. Similarly to (KAISER et al., 2018; MALLYA; DAVIS; LAZEBNIK, 2018), during backward propagation, the τ function is replaced by the identity function to be able to back-propagate the error and update $\tilde{s}_c$. Even though we learn $\tilde{s}_c$ at training time, we only need to store the binary s_c values to use at testing time, leading to a small storage requirement (1-bit per s_c). Compared to other multi-domain methods that generally use additional 32-bit floating-point numbers (REBUFFI; BILEN; VEDALDI, 2017; REBUFFI; BILEN; VEDALDI, 2018; ROSENFELD; TSOTSOS, 2018), BA^2 results in a much lighter storage.

The proposed BA^2 have four main features:

Adapting image representation: In BA^2, the ϕ_c features can be interpreted as a filter bank and the switch vectors can be understood as a filter selector. Depending on the domain, different switch values can be employed to select features relevant for the considered domain.

Low computational complexity: After training, all the tensors $\{\phi_c \mid s_c = 0\}$ can be removed of the computational graph, resulting in a lower computational complexity. More precisely, the computational complexity is proportional to:

$$\mathcal{C} = \frac{1}{C} \sum_{c=1}^{C} s_c. \tag{4.5}$$

Note that, the uncomputed operations are grouped in channels allowing fast GPU implementation.

Lower storage: First, the number of additional parameters is rather small compared to the number of kernel parameters of the base network. Second, at testing time, the additional switch parameters can be stored with a binary representation to obtain a lightweight storage (1-bit per kernel channel). Finally, the weight values $\{\boldsymbol{K}(h, w, c) \mid s_c = 0\}$ can be dropped, obtaining models with fewer parameters for the new domains. Again, the number of parameters is proportional to $\mathcal{C}$ in Equation (4.5).

Low Memory footprint: Reducing the computational complexity, does not necessarily reduces the memory footprint at testing time. In order to properly reduce the memory footprint, one needs to reduce the memory requirements of all operations across the

computational graph, as stated in (SANDLER et al., 2018). Given that BA^2 works on the level of the convolution operation, it can also control the memory footprint.

4.3.2 Training Budget-aware adapters for MDL

We now detail how BA^2 is used for MDL. As explained in Subsection 4.3.1, we follow a strategy of adapting a pre-trained model to novel domains. Therefore, when learning a new domain, we consider that θ_0 is provided and we keep it fixed for the whole training procedure.As a consequence the learning procedure can be subdivided in independent training for each domain resulting in simpler training procedure. Note that, similarly to (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2017), we use batch-normalization parameters specific for each domain. Furthermore, as shown in (YU et al., 2019), using different number of channels leads to different feature mean and variance and, as a consequence, sharing Batch Normalization layers performs poorly. Therefore, we use different batch-normalization layers for each budget. Note that, since the number of parameters in a batch-normalization layer is much lower than in convolution layer, this solution does not increase significantly the number of additional parameters with respect to the size of θ_0.

Following the notations introduced in Chapter 4, θ_a^d now denotes the set of all the switch values s_c and the additional batch-normalization parameters. Considering a new domain d, Ψ_d is trained using a loss $\mathcal{L}$. In the case of classification, we employ the cross-entropy loss for all the domains. In the context of BA^2, we aim at training Ψ_d with budget constraints. Formally, we formulate the optimization problem as follows: we minimize $\mathcal{L}$ with respect to the BA^2 parameters θ_a^d such that the network complexity satisfies a target budget $\beta \in [0, 1]$. For each new domain, we obtain the following constrained optimization problem:

$$\theta_a^{d*} = \arg\min_{\theta_a^d} \mathcal{L}(\theta_0, \theta_a^d) \tag{4.6}$$

$$s.t.\ \bar{\theta_a^d} \leq \beta \tag{4.7}$$

where $\bar{\theta_a^d}$ denotes the mean value of the switches in θ_a^d. From Equation (4.6), we construct the generalized Lagrange function and the associated optimization problem:

$$\theta_a^{d*} = \arg\min_{\theta_a^d} \Big[\mathcal{L}(\theta_0, \theta_a^d) + \max_{\lambda \geq 0}(\lambda(\bar{\theta_a^d} - \beta))\Big]. \tag{4.8}$$

The λ is known as the Karush-Kuhn-Tucker (KKT) multiplier. Equation (4.8) is optimized via stochastic gradient descent (SGD). When the budget constrained is respected, $\lambda = 0$ and Equation (4.8) corresponds to $\mathcal{L}$ minimization. When the constraint is not satisfied, in addition to $\mathcal{L}$ minimization, the SGD steps also lead to an increase of λ which in turn increases the impact of the budget constraint on $\mathcal{L}$.

In order to obtained networks with different budgets, training is performed independently for each β value. When β is set to 1, the constraint in Equation (4.6) is satisfied for any θ_a^d. Therefore, the problem consists in a loss minimization problem over the parametric network family defined by θ_0. This scenario corresponds to a standard multi-domain scenario without considering budget as in (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2017; ROSENFELD; TSOTSOS, 2018). When $\beta < 1$, we combine both re-parametrization and budget-adjustable abilities of *Budget-Aware Adapters*. In this case, the goal is to obtain the best performing model that respects the budget constrain. It is important to note that the actual complexity of the network, after training, can be lower than the one defined by the user, including the $\beta = 1$ case.

Note that in Equation (4.6), the budget constraint is formulated as a constraint on the total network complexity. In practice, it can be preferable to constrain each BA^2 to satisfy independent budget constraints in order to both spread computation over the layers and obtain a lower memory footprint. In this case, KKT multipliers are added in Equation (4.8) for each convolution layer.

4.4 Experiments & Results

In this section we present the experimental methodology and metrics used to evaluate our approach. Moreover, we report the results and comparisons with state of the art MDL approaches (Subsection 4.4.1). In addition, we also conduct further experiments on the usual single-domain setting and demonstrate the effectiveness of BA^2 (Subsection 4.4.3) in reducing complexity while learning accurate recognition models.

4.4.1 Multi-Domain Learning

Datasets. In order to evaluate our MDL approach, we adopt two different benchmarks. We first consider the Visual Decathlon Challenge (REBUFFI; BILEN; VEDALDI, 2017). The purpose of this challenge is to compare methods for MDL over 10 different classification tasks: ImageNet (RUSSAKOVSKY et al., 2015), CIFAR-100 (KRIZHEVSKY, 2009), Aircraft (MAJI et al., 2013), Daimler pedestrian (DPed) (MUNDER; GAVRILA, 2006), Describable Textures (DTD) (CIMPOI et al., 2014), German Traffic Signs (GTSR) (STALLKAMP et al., 2012), Omniglot (LAKE; SALAKHUTDINOV; TENENBAUM, 2015), SVHN (NETZER et al., 2011), UCF101 Dynamic Images (BILEN et al., 2016; SOOMRO; ZAMIR; SHAH, 2012) and VGG-Flowers (NILSBACK; ZISSERMAN, 2008). For more details about the challenge, please refer to (REBUFFI; BILEN; VEDALDI, 2017).

As for the second benchmark, we follow previous works (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018) and consider the union of six different datasets: Ima-

geNet (RUSSAKOVSKY et al., 2015), VGG-Flowers (NILSBACK; ZISSERMAN, 2008), Stanford Cars (KRAUSE et al., 2013), Caltech-UCSD Birds (CUBS) (WELINDER et al., 2010), Sketches (EITZ; HAYS; ALEXA, 2012), and WikiArt (SALEH; ELGAMMAL, 2016). These datasets are very heterogeneous, comprising a wide range of the categories (e.g., cars (KRAUSE et al., 2013) vs birds (WELINDER et al., 2010)) and a large variety of image appearance (i.e., natural images (RUSSAKOVSKY et al., 2015), art paintings (SALEH; ELGAMMAL, 2016), sketches (EITZ; HAYS; ALEXA, 2012)).

Accuracy Metrics. Both benchmarks are designed to address classification problems. Therefore, as common practice (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018), we report the accuracy for each domain and the average accuracy over the domains. In addition, the score function S, as introduced in (REBUFFI; BILEN; VEDALDI, 2017), is considered to jointly account for the N domains. The test error E_d of the model on the domain d is compared to the test error of a baseline model $E_d^{\max}$. The score is given by $S = \sum_{d=1}^{N} \alpha \max\{0, E_d^{\max} - E_d\}^2$, where α is a scaling parameter ensuring that the perfect score for each domain is 1000. The baseline error is given by doubling the error of 10 independent models fine-tuned for each domain. Importantly, this metric favors models with good performances over all domains, while penalizing those that are accurate only on few domains.

Complexity Metrics. Furthermore, since in this work we argue that MDL methods should also be evaluated in terms of model complexity, we consider two other metrics which account for the number of network parameters and operations. First, following (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018), we report the total number of parameters relative to the ones of the initial pre-trained model (counting all domains and excluding the classifiers). Note that, when computing the model size, we consider that all float numbers are encoded in 32 bits and switches in 1 bit only. Second, we propose to report the average number of floating-point operations (FLOP) over all the domains (including the pre-training domain $d = 0$, i.e., ImageNet) relative to the number of operations of the initial pre-trained network. Interestingly, for the Budget-Aware Adapters, this ratio is also equal to the average number of parameters used at inference time for each individual domain, relative to the number of parameters of Ψ_0.

These complexity measures lead us to two variants of the score S: the score per parameter S_P and the score per operation S_O. These two metrics are able to assess the trade-off between performance and model complexity.

Networks and training protocols. Concerning the Visual Decathlon, we consider the Wide ResNet-28 (ZAGORUYKO; KOMODAKIS, 2016) adopted in previous works (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2017; ROSENFELD; TSOTSOS, 2018) and employ the same data pre-processing protocol. In term of hyper-parameters, we follow (REBUFFI; BILEN;

VEDALDI, 2017) when pre-training on ImageNet. For other domains, we employ the hyper-parameters used in (MALLYA; DAVIS; LAZEBNIK, 2018).

For the second benchmark, we use a ResNet-50 (HE et al., 2016). Note that, since the performance of the method in (MALLYA; LAZEBNIK, 2018) relies on the order of the domains, we report the performances for two orderings as in (MALLYA; DAVIS; LAZEBNIK, 2018): starting from the model pre-trained on ImageNet, the first ($\rightarrow$) corresponds to CUBS-Cars-Flowers-WikiArt-Sketch, while the second ($\leftarrow$) corresponds to reversed order. We also followed the pre-processing, hyper-parameters and training schedule of (MALLYA; DAVIS; LAZEBNIK, 2018), as we did in the Visual Decathlon Challenge.

We chose to use the same setting (network, pre-processing, and training schedules) employed by previous works seeking fairer analyses regarding the impact of the proposed approach.

Budget Constraints. Even if our training procedure is formulated as a constrained optimization problem (see Equation 4.7), the stochastic gradient descent algorithm we employ does not guarantee that all the constraints will be satisfied at the end of training. Therefore, in all our experiments, we check whether the final models respect the specified budget constraints. All the scores reported in this work were obtained with models that respect the specified budget constraints, unless explicitly specified otherwise.

4.4.1.1 Visual Decathlon Challenge

Training and Evaluation Protocols. Following previous works (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018; REBUFFI; BILEN; VEDALDI, 2017; ROSENFELD; TSOTSOS, 2018), we employ the Wide ResNet WRN-28-4-B(3,3), i.e., 28 convolutional layers with widening factor of 4 and the original "basic" block (2 convolutions using 3×3 kernel size). As in (REBUFFI; BILEN; VEDALDI, 2017; MANCINI et al., 2018), random crops of 64×64 pixels are used to feed the network during training. The optimizer parameters are set following (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018), where SGD with momentum is employed for the classifier and Adam for the rest of the architecture with initial learning rates of 0.001 and 0.0001, respectively. The model is trained with batch size of 32 for 60 epochs; after 45 epochs, the learning rates are decayed by a factor of 10. The real-valued switches ($\tilde{s}_c$) are initialized with a value of 0.001. In addition to random cropping, in regards to training data augmentation, horizontal mirroring is also applied with a 50% probability, except for four datasets (DTD, Omniglot, SVHN, and GTSR) on which it could be either harmful or useless. During training, the models for all the domains are initialized using the ImageNet pre-trained weights and these weights are kept fixed, i.e., only the domain-specific parameters ($\tilde{s}_c$, Batch Normalization and classifiers) are learned. At test time, we follow the procedure

proposed in (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) and used in (MANCINI et al., 2018). We employ a ten-crop strategy for the datasets in which horizontal mirroring was applied and five-crop otherwise. In the five-crop strategy, a crop is performed on each corner of the image in addition to a central one; the ten-crop adds horizontally mirrored versions of each one of the five crops. The final prediction is based on the average of the predictions over all the crops.

Results. We first evaluate our methods on the Visual Decathlon Challenge. Results are reported in Table 10. We report the BA^2 scores with respect to four different budgets $\beta \in \{0.25, 0.50, 0.75, 1.00\}$. We first observe that for most of the domains, BA^2 without budget constrains is competitive with state-of-the-art methods in terms of accuracy. Our method is the second best performing for three domains (GTSR, VGG-Flowers, and SVHN). In terms of score, among lightweight methods, only Parallel Adapters (PA) and Weight Transformations using Binary Masks (WTPB) perform better than ours. However, both methods require significantly more additional parameters to achieve these performances. Concerning the models where a budget constraint is considered, we observe that the scores still outperform RA (REBUFFI; BILEN; VEDALDI, 2017), DAM (ROSENFELD; TSOTSOS, 2018) and PB (MALLYA; DAVIS; LAZEBNIK, 2018) in terms of score when targeting a budget of 50% or 75% of the initial network parameters, i.e., $\beta = 0.50$ or 0.75.

Table 10 – Results in terms of accuracy and S-Score, for the Visual Decathlon Challenge. Best model in bold, second best underlined.

Method	Params	ImNet	Airc.	C100	DPed	DTD	GTSR	Flwr.	Oglt.	SVHN	UCF	Mean	S-Score
Feature (REBUFFI; BILEN; VEDALDI, 2017)	**1**	59.7	23.3	63.1	80.3	45.4	68.2	73.7	58.8	43.5	26.8	54.3	544
Finetune (REBUFFI; BILEN; VEDALDI, 2017)	10	59.9	60.3	82.1	92.8	55.5	97.5	81.4	87.7	96.6	51.2	76.5	2500
SpotTune (GUO et al., 2019)	11	60.3	63.9	80.5	96.5	57.13	**99.5**	85.22	88.8	96.7	**52.3**	**78.1**	**3612**
RA(REBUFFI; BILEN; VEDALDI, 2017)	2	59.7	56.7	81.2	93.9	50.9	97.1	66.2	89.6	96.1	47.5	73.9	2118
DAM (ROSENFELD; TSOTSOS, 2018)	2.17	57.7	64.1	80.1	91.3	56.5	98.5	86.1	**89.7**	96.8	49.4	77.0	2851
PA (REBUFFI; BILEN; VEDALDI, 2018)	2	60.3	64.2	**81.9**	94.7	58.8	99.4	84.7	89.2	96.5	50.9	**78.1**	3412
PB (MALLYA; DAVIS; LAZEBNIK, 2018)	1.28	57.7	**65.3**	79.9	**97.0**	57.5	97.3	79.1	87.6	**97.2**	47.5	76.6	2838
WTPB (MANCINI et al., 2018)	1.29	**60.8**	52.8	**82.0**	96.2	58.7	99.2	**88.2**	89.2	96.8	48.6	77.2	3497
BA^2 (Ours) ($\beta = 1.00$)	1.03	56.9	49.9	78.1	95.5	55.1	99.4	86.1	88.7	96.9	50.2	75.7	3199
BA^2 (Ours) ($\beta = 0.75$)	1.03	56.9	47.0	78.4	95.3	55.0	99.2	85.6	88.8	96.8	48.7	75.2	3063
BA^2 (Ours) ($\beta = 0.50$)	1.03	56.9	45.7	76.6	95.0	55.2	99.4	83.3	88.9	96.9	46.8	74.5	2999
BA^2 (Ours) ($\beta = 0.25$)	1.03	56.9	42.2	71.0	93.4	52.4	99.1	82.0	88.5	96.9	43.9	72.6	2538

Interestingly, it can be seen in Table 10 that, when we impose a tighter budget to BA^2, the total number of parameters do not decrease. Indeed, all the parameters of the pre-trained network Ψ_0 are still required at testing time to handle the 10 domains. Only the number of parameters used for each domain and the number of floating-point operations are reduced. Therefore, we propose to complete this evaluation in order to further understand the performance/complexity trade-off achieved by each method. More precisely, we report the number of parameters and FLOPs in Table 11, and their corresponding scores. First, we observe that only BA^2 models report FLOPs lower than 1. In other words, only BA^2 provide models with fewer operations than the initial network Ψ_0. We see that BA^2 achieve the best performance in terms of S_P when using 100% budget. As mentioned above, the total number of parameters for the 10 domains do not decrease with a smaller budget.

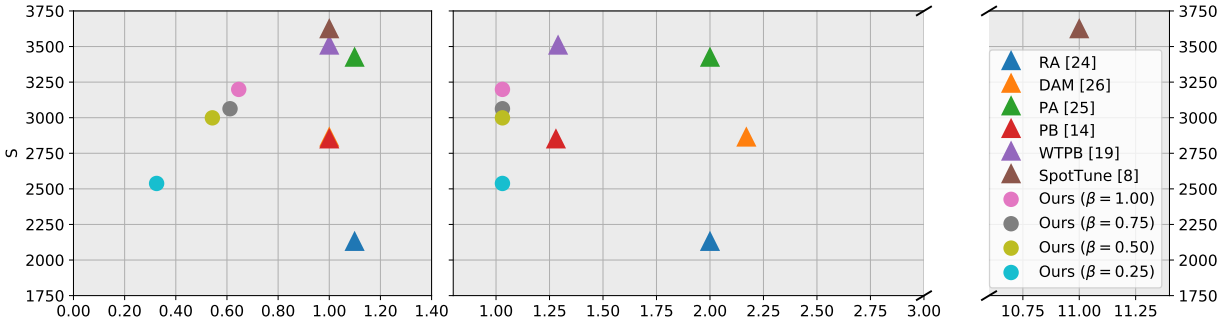
Consequently, smaller budgets obtain lower S_P values. Nevertheless, the 75% and 50% models rank second and third, respectively.

Table 11 – Performance/Complexity trade-off comparison on the visual decathlon challenge.

Method	FLOP	Params	Score	S_O	S_P
Feature	1	1	544	544	544
Finetune	1	10	2500	2500	250
SpotTune	1	11	3612	3612	328
RA	1.099	2	2118	1926	1059
DAM	1	2.17	2851	2851	1314
PA	1.099	2	3412	3102	1706
PB	1	1.28	2838	2838	2217
WTPB	1	1.29	**3497**	3497	2710
BA^2 (Ours) ($\beta = 1.00$)	0.646	**1.03**	<u>3199</u>	4952	**3106**
BA^2 (Ours) ($\beta = 0.75$)	0.612	**1.03**	3063	5005	<u>2974</u>
BA^2 (Ours) ($\beta = 0.50$)	<u>0.543</u>	**1.03**	2999	<u>5523</u>	2912
BA^2 (Ours) ($\beta = 0.25$)	**0.325**	**1.03**	2538	**7809**	2464

Concerning the FLOP, only BA^2 return models with fewer floating-point operations than the initial network. As a consequence, BA^2 clearly outperforms other approaches in terms of S_O. In addition, we note that S_O increases when using tighter budgets. It illustrates the potential of our approach in order to obtain a good performance/complexity trade-off. Interestingly, even the models with $\beta = 100\%$ report a FLOP value lower than 1 since convolutional channels can be dropped to adapt to each domain. Note that for all our models, the reported FLOP numbers are smaller than the specified budget. The reason for this is that we impose budget constraints independently to each convolutional layer in order to obtain a low memory footprint (see Subsection 4.3.2). Therefore, the average percentage of channels that are dropped can be smaller than the specified budget and, in fact, this is what we observed in ours models.

For better visualization, we illustrate in Figure 37 the performance/complexity trade-off for each method on the Visual Decathlon Challenge. More precisely, in Figure 37a, we plot the score obtained as a function of the computation complexity in FLOPs. When comparing with other methods, we see that BA^2 lead to much lighter models that have, as a consequence, better performance/computation trade-offs. In Figure 37b, we report the obtained score as a function of the total number of parameters. BA^2 is the method that requires the lowest number of additional parameters to adapt to the 10 domains. Furthermore, this plot clearly show that our 100% model has an interesting trade-off between the performance and the number of additional parameters. Note that WTPB (MANCINI et al., 2018) also obtained a good trade-off but stores, in total, 29% more parameters (approximately $\times 9$ per new domain). Interestingly, the best performing



(a) Comp. complexity (FLOPs) (b) Network weight (Params)

Figure 37 – Performance/Complexity Trade-Off on the visual decathlon challenge: the total score is displayed as a function of the two considered complexity metrics *FLOPs* and *Params*. Note that DAM is below PB in (a).

approach in terms of score, i.e., *SpotTune* (GUO et al., 2019), requires a much larger number of parameters that would restrict the use of this method when increasing the number of domains.

Visualization. There are several methods to visualize CNNs. Some of these recent methods (e.g., (ZHOU et al., 2018)) are class-specific and it is not clear how to efficiently use them in our multi-domain setting. Thus, to visualize our models we employed VisualBackProp (BOJARSKI et al., 2018) and compared the activation maps obtained without and with BA^2 (β=1 and β=0.25) in Figure 38. As it can be seen, when using BA^2, we obtain much sharper and object-specific attention maps than with ImageNet pretrained model.

Figure 38 – Visualization of the activation maps without (top row) and with BA^2: β=1 (middle row) and β=0.25 (bottom row).

4.4.1.2 ImageNet-to-Sketch

Training and Evaluation Protocols. We use the ResNet-50 as in (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018) feeding a random crop of 224 × 224 pixels after resizing the images to 256 × 256 pixels. In addition to random cropping,

horizontal mirroring is applied to all datasets during training. The networks are initially trained using the same schedule as in (MALLYA; DAVIS; LAZEBNIK, 2018; MANCINI et al., 2018), i.e., 30 epochs with a learning rate drop (with a factor of 10) after 15 epochs. For the lowest budget ($\beta = 0.25$), we add another learning rate drop at 30 epochs and train for additional 15 epochs in order to fully satisfy the budget constraints. All the other steps are performed as in the Visual Decathlon Challenge.

Results. We now compare our method with state-of-the-art approaches on the ImageNet-to-Sketch setting using the ResNet-50 architecture. Results are reported in Table 12. First, we see that BA^2 achieve the second best score among methods that employ only a small number of additional parameters. Only WTPB (MANCINI et al., 2018) reports better scores at a higher cost in terms of additional parameters per domain. Interestingly, the 50% and 75% models report similar performances. Second, the S_O and S_P values clearly confirm the conclusions drawn on the first experiments on the Visual Decathlon Challenge. The four different BA^2 models outperform all the other methods in terms of S_O and our model with 100% budget slightly outperforms WTPB in terms of S_P.

Table 12 – State of the art comparison on the ImageNet-to-Sketch benchmark using ResNet-50 architecture. Best model in bold, second best underlined.

	FLOP	Params	ImageNet	CUBS	Cars	Flowers	WikiArt	Sketch	Score	S_O	S_P
Classifier Only (MALLYA; DAVIS; LAZEBNIK, 2018)	1	**1**	76.2	70.7	52.8	86.0	55.6	50.9	533	533	533
Individual Networks (MALLYA; DAVIS; LAZEBNIK, 2018)	1	6	76.2	82.8	91.8	**96.6**	75.6	**80.8**	**1500**	1500	250
SpotTune (GUO et al., 2019)	1	7	76.2	**84.03**	**92.40**	96.34	**75.77**	80.2	**1526**	1526	218
PackNet → (MALLYA; LAZEBNIK, 2018)	1	1.10	75.7	80.4	86.1	93.0	69.4	76.2	732	732	665
PackNet ← (MALLYA; LAZEBNIK, 2018)	1	1.10	75.7	71.4	80.0	90.6	70.3	78.7	620	620	534
Piggyback (MALLYA; DAVIS; LAZEBNIK, 2018)	1	1.16	76.2	80.4	88.1	93.5	73.4	79.4	934	934	805
Piggyback+BN (MALLYA; DAVIS; LAZEBNIK, 2018)	1	1.17	76.2	82.1	90.6	95.2	74.1	79.4	1184	1184	1012
WTPB (MANCINI et al., 2018)	1	1.17	76.2	82.6	91.5	96.5	74.8	80.2	1430	1430	1222
BA^2 (Ours) ($\beta = 1.00$)	0.700	1.03	76.2	81.19	92.14	95.74	72.32	79.28	1265	1807	**1228**
BA^2 (Ours) ($\beta = 0.75$)	0.600	1.03	76.2	79.44	90.62	94.44	70.92	79.38	1006	1677	977
BA^2 (Ours) ($\beta = 0.50$)	0.559	1.03	76.2	79.34	90.80	94.91	70.61	78.28	1012	1810	983
BA^2 (Ours) ($\beta = 0.25$)	**0.375**	1.03	76.2	78.01	88.15	93.19	67.99	77.85	755	**2013**	733

In addition to results with ResNet-50 models, results using the DenseNet-121 are also reported in several works (MALLYA; DAVIS; LAZEBNIK, 2018; MALLYA; LAZEBNIK, 2018; MANCINI et al., 2018). For that reason, we also provide these results in Table 13. First, we see that our method achieves the best scores in two domains and the second best in three other domains. Interestingly, in the *Cars* domain, our method with $\beta = 0.75$ is the second best model (only after our method with $\beta = 1.00$), achieving results better than all the other methods using only 57.8% of the FLOP, on average. Concerning the number parameters, our approach is the second best in terms of Params. Indeed the number of batch normalization and 1×1 convolutions parameters in the DenseNet-121 model with respect to the total number of parameters is higher than in the ResNet-50 model. Nevertheless, BA^2 is still the only one with FLOP less than 1. Finally, it can be noted that our method with $\beta = 1.00$ still achieves a good trade-off between performance and complexity.

Table 13 – State-of-the-art comparison on the ImageNet-to-Sketch benchmark using DenseNet-121 architecture. (*) Even though the average sparsities are greater than 75%, these models did not satisfy the constraint for every single layer.

	FLOP	Params	ImageNet	CUBS	Cars	Flowers	WikiArt	Sketch	Score	S_O	S_P
Classifier Only (MALLYA; DAVIS; LAZEBNIK, 2018)	1	1	74.4	73.5	56.8	83.4	54.9	53.1	328	328	328
Individual Networks (MALLYA; DAVIS; LAZEBNIK, 2018)	1	6	74.4	81.7	91.4	96.5	76.4	80.5	1500	1500	250
PackNet → (MALLYA; LAZEBNIK, 2018)	1	**1.11**	74.4	80.7	84.7	91.1	66.3	74.7	691	691	623
PackNet ← (MALLYA; LAZEBNIK, 2018)	1	**1.11**	74.4	69.6	77.9	91.5	69.2	78.9	610	610	550
Piggyback (MALLYA; DAVIS; LAZEBNIK, 2018)	1	1.15	74.4	79.7	87.2	94.3	72.0	**80.0**	951	951	827
Piggyback+BN (MALLYA; DAVIS; LAZEBNIK, 2018)	1	1.21	74.4	81.4	90.1	95.5	73.9	79.1	1215	1215	1004
WTPB (MANCINI et al., 2018)	1	1.21	74.4	81.7	91.6	**96.9**	**75.7**	79.8	**1540**	1540	**1268**
BA^2 (Ours) ($\beta = 1.00$)	0.687	1.17	74.4	**82.4**	**92.9**	96.0	71.5	79.9	1440	2096	1230
BA^2 (Ours) ($\beta = 0.75$)	0.578	1.17	74.4	81.2	91.9	94.9	68.9	79.9	1193	2064	1019
BA^2 (Ours) ($\beta = 0.50$)	0.543	1.17	74.4	78.2	89.2	95.0	66.2	78.8	925	1703	790
BA^2 (Ours) ($\beta = 0.25$)	**0.375**	1.17	74.4	76.2*	88.4*	94.7*	67.9*	78.4	840	**2240**	717

4.4.2 Ablation study of BA^2

In order to further understand the performance of BA^2, we propose to compare the drop in accuracy when imposing different budget constraints. In Figure 39, we perform an experiment on the Visual Decathlon Challenge with varying budgets $\beta = \{0.1, 0.2, \ldots, 1.0\}$. We display the accuracy drop relative to the performance of the model with 100% budget. Because of the restrictions of the Challenge in terms of number of submissions (per day and in total), we report results on the validation set. To decrease the impact of the training stochasticity, we report the median performance over 4 runs. As mentioned previously, the stochastic gradient descent algorithm we employ for training our model does not guarantee to provide a solution that respect the budget constraints. Therefore, in Figure 39, we display only points corresponding to models that satisfy the specified budget. We first observe that for all the domains, our method returns models that respect the specified budget when the budget is greater than 30%. For some domains, we obtain models that respect even tighter budgets, such as the *GTSR* dataset where we obtain a 10%-budget model.

Interestingly, we notice that the domains where our models fail to respect the 20% budget are the same in which the drop in performance is more clearly visible from the 100% to 30% budgets. Furthermore, we observe that the domains where the performance drop is small correspond to those where the 100% model reaches excellent performance. For instance, in Table 10 the models for the *GTSR* (traffic signs) and *DPed* (pedestrians) datasets reach accuracy over 95% with our 100% model and do not significantly lose performance when imposing a tighter budget. Conversely, we observe that the *DTD* and the *aircraft* datasets, that show the largest performance drop, correspond also to the most challenging datasets according the accuracies of all methods reported in Table 10.

4.4.3 Evaluating BA^2 for single-domain problems

In order to further demonstrate the effectiveness of our proposed BA^2, we perform experiments on a standard single-domain classification problem training a single ConvNet

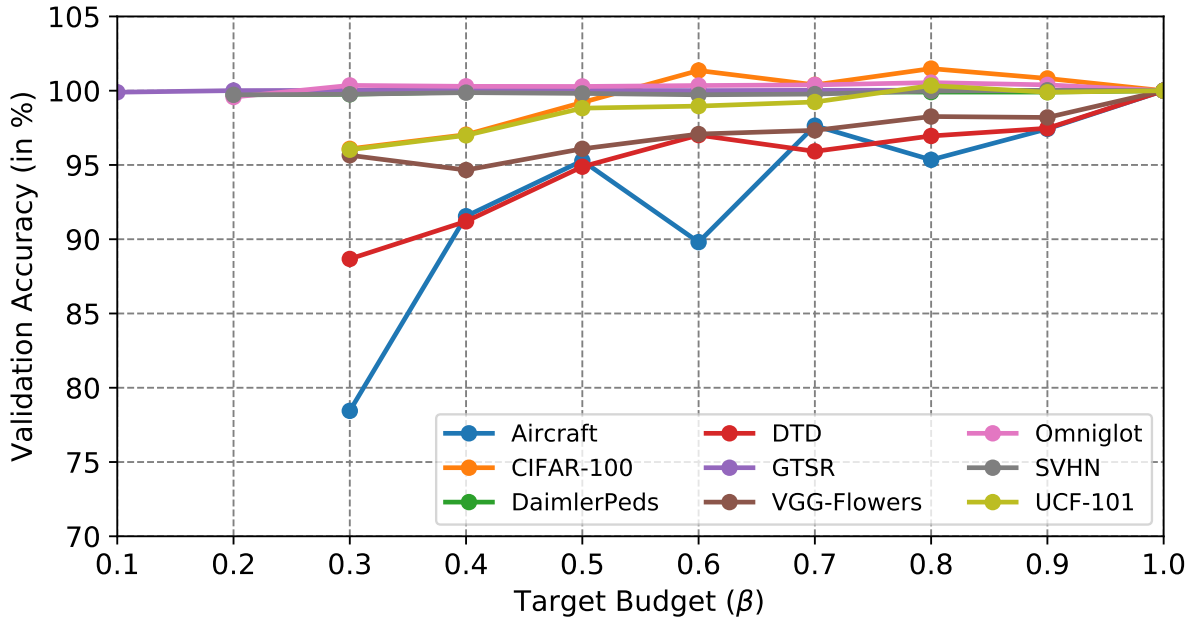
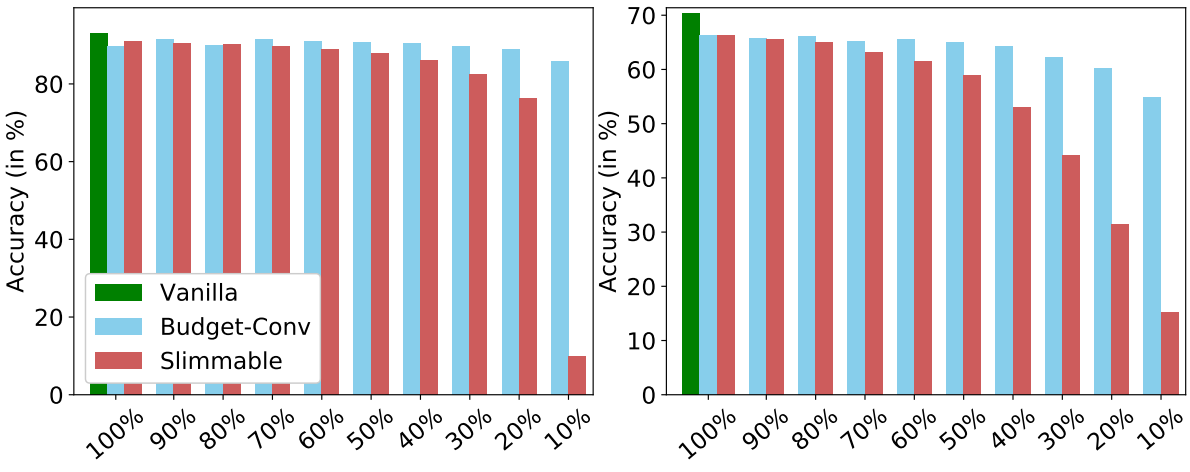


Figure 39 – Relative accuracy drop compared to the 100% model accuracy for the 10 domains of the Visual Decathlon challenge datset (validation set). Median score over 4 runs is reported. Missing points correspond to models where the budget constraint was not satisfied in the four runs.



(a) Cifar-10 (b) Cifar-100

Figure 40 – Single-domain classification with adaptive budget. BA^2 is compared with Slimmable Networks (YU et al., 2019).

with different budget constraints.

Datasets and Experimental Protocol. We perform experiments on two well-known classification datasets: CIFAR-10 and CIFAR-100 (KRIZHEVSKY, 2009). In these experiments, we use the same variant of the ResNet proposed in the original Residual Networks (HE et al., 2016) paper for the CIFAR-10 dataset with $n = 9$, which results in a ResNet with 56 layers. Then, we employ the following procedure. First, we train the

baseline model, which is the model without *switches* (equivalent to setting all the switch vectors to 1). Subsequently, we use different switches θ_s for each budget, but we share the convolution parameters θ_k among budgets. We fine-tune all the parameters optimizing Equation (4.8) w.r.t. θ_k and all the different θ_s parameters jointly. In other words, since we are only interested in single-domain learning for this experiment, we do not need to freeze the weights as in the multi-domain experiments. Therefore, we jointly train the switches and the weights using the baseline pre-trained weights as initialization.

Results. We compare our approach with the recently proposed Slimmable Networks (YU et al., 2019). We consider this approach as this is the most closely related to our method in literature. In (YU et al., 2019), different budgets are obtained by gradually dropping filters, imposing that filters dropped for a given budget are also dropped for lower budgets. Conversely, in BA^2 we do not impose any constraints between the switches at different budgets. It can be observed in Figure 40 that, in the case of CIFAR-10, both models achieve an accuracy similar to a vanilla network trained without any budget constraint. On the more challenging CIFAR-100, both BA^2 and Slimmable Networks perform slightly worse than the vanilla network. Interestingly, BA^2 is able to maintain a constant accuracy on both datasets when decreasing the budget constraint up to 50% whereas Slimmable Networks begins to perform poorly. The difference between the two methods becomes larger with tighter budgets, until 10% where Slimmable Networks accuracy collapses. These experiments show that imposing constraints between budgets as in (YU et al., 2019) harms the performance and illustrate the potential of our approach even for single-domain problems.

4.5 Discussion & Conclusion

In this work, we proposed to investigate the multi-domain learning problem with budget constraints. We propose to adapt a pre-trained network to each new domain with constraints on the network complexity. Our *Budget-Aware Adapters* (BA^2) select the most relevant feature channels using trainable switch vectors. We impose constraints on the switches to obtain networks that respect to user-specified budget constraints. From an experimental point of view, BA^2 show performances competitive with state-of-the-art methods even with small budget values.

Future works. There are a couple of directions to improve on what we proposed, and here we highlight two. First, in BA^2 we propose a way in which the user can change the computational cost of a model at inference time according to a set of budgets defined a priori. In this context, it would be interesting to extend our adapter to allow the user to control the computational cost of a model in a continuous fashion. The second direction aims at effectively decreasing the number of parameters by better exploiting the multiple

domains. Instead of training the BA^2 independently for each domain, assuming multiple domains are available at the same time, we could extend the training protocol and train the adapters jointly among the domains, also encouraging the budget constraint to be satisfied for all the domains at the same time, which would allow for an effective reduction in the number of parameters for the MDL context.

In summary, in the context of multi-domain learning, our results show that exploiting a generic image representation can help to not only reduce but also control the computational and memory requirements of a model at inference time. Moreover, our approach provides the user with options to balance between performance and model complexity, which validates our hypotheses.

5 Final Remarks

In this thesis, we set out to investigate ways of reducing the efforts in training deep neural networks, especially those concerning data-related and computational costs. First, we developed techniques that exploited data readily available through online platforms to acquire massive worldwide datasets for specific problems. These techniques have allowed a drastic reduction in data acquisition cost while also expediting the whole process, which was the first objective of this thesis. Then, after the acquisition of these large-scale datasets, we contributed with techniques to fuse data available in crowdsourced platforms such that we could annotate the collected data either reducing or removing the human effort altogether, which was the second objective of this thesis. The proposed techniques were employed in three applications of automatic large-scale data acquisition and annotation that proved to be useful for training deep convolutional networks for both classification and regression problems. Our results show that the ability to collect a really large and diverse dataset overcomes the inherent noise that these processes carry, and allows the training of deep learning models that have shown to be effective in solving real-world problems. Next, achieved our last objective by developing a model in which the user can reduce and control the computational requirements at inference time, particularly in the context of multiple domains. Our proposed model has a low computation complexity, requires lower storage, and has a low memory footprint compared to the alternatives. The results showed that not only it is possible to have a model with an adjustable (reduced) computational budget at inference time, but this can be beneficial to both multi- and single-domain problems. Finally, the techniques, models, and applications we developed showed that the efforts in training deep networks can be further reduced, particularly when it comes to data-related and computational costs, thus helping to increase the sustainability of deep neural networks.

5.1 Future works

There are several directions to continue reducing data-related and computational costs, and some specific ones were provided in Chapters 3 and 4. In this section, we provide a few additional recommendations for future works. A promising direction to greatly reduce data acquisition and annotation costs is to work with completely artificial datasets (e.g., from simulators). Moreover, as realistic simulators are difficult and sometimes expensive to develop, it would be interesting if we could exploit more basic simulators (which would provide annotated data) together with large amounts of non-annotated data from the domain of interest. Both these directions will require significant developments in the

current domain adaptation techniques. A different direction is the improvement of few-shot techniques, which tackles both data-related and computational costs. Although few-shot learning has been widely studied for classification problems, there seems to be still a long road to go for other applications (e.g., detection, segmentation, etc.). These future works would help to further reduce the efforts in training deep neural networks, making them more accessible to the broader research community and more industrial applications.

Bibliography

ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANé, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIéGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; ZHENG, X. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Available at: <https://www.tensorflow.org/>. Cited on page 78.

ABDALLAH, M.; ELKEELANY, O. A Survey on Data Acquisition Systems DAQ. In: *International Conference on Computing, Engineering and Information*. USA: IEEE, 2009. p. 240–243. Cited on page 32.

AHMETOVIC, D.; MANDUCHI, R.; COUGHLAN, J. M.; MASCETTI, S. Zebra Crossing Spotter: Automatic Population of Spatial Databases for Increased Safety of Blind Travelers. In: ACM. *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*. Portugal, 2015. p. 251–258. Cited on page 39.

ALJUNDI, R.; BABILONI, F.; ELHOSEINY, M.; ROHRBACH, M.; TUYTELAARS, T. Memory Aware Synapses: Learning what (not) to forget. In: *European Conference on Computer Vision (ECCV)*. Germany: Springer, 2018. p. 144–161. Cited on page 86.

Amazon. *Amazon Mechanical Turk*. 2005. <https://mturk.com>. Visited on 08/19/2021. Cited on page 33.

ARRUDA, V. F.; PAIXãO, T. M.; BERRIEL, R. F.; SOUZA, A. F. D.; BADUE, C.; SEBE, N.; OLIVEIRA-SANTOS, T. Cross-Domain Car Detection Using Unsupervised Image-to-Image Translation: From Day to Night. In: *International Joint Conference on Neural Networks (IJCNN)*. Budapest, Hungary: IEEE, 2019. Cited 2 times on pages 27 and 120.

BADUE, C.; GUIDOLINI, R.; CARNEIRO, R. V.; AZEVEDO, P.; CARDOSO, V. B.; FORECHI, A.; JESUS, L.; BERRIEL, R.; PAIXãO, T.; MUTZ, F.; VERONESE, L.; OLIVEIRA-SANTOS, T.; SOUZA, A. F. D. Self-Driving Cars: A Survey. *Expert Systems with Applications*, v. 165, 2021. ISSN 0957-4174. Cited 2 times on pages 26 and 119.

BENDALE, A.; BOULT, T. E. Towards Open Set Deep Networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, USA: IEEE, 2016. Cited on page 86.

Berkeley DeepDrive. *Scalabel*. 2017. <https://www.scalabel.ai>. Visited on 08/19/2021. Cited on page 33.

BERRIEL, R. F.; AGUIAR, E. de; SOUZA, A. F. de; OLIVEIRA-SANTOS, T. Ego-Lane Analysis System (ELAS): Dataset and Algorithms. *Image and Vision Computing*, v. 68, p. 64–75, 2017. Cited 2 times on pages 39 and 71.

BERRIEL, R. F.; LATHUILIèRE, S.; NABI, M.; KLEIN, T.; OLIVEIRA-SANTOS, T.; SEBE, N.; RICCI, E. Budget-Aware Adapters for Multi-Domain Learning. In: *The IEEE International Conference on Computer Vision (ICCV)*. Seoul, South Korea: CVF/IEEE, 2019. p. 382–391. Cited 2 times on pages 27 and 120.

BERRIEL, R. F.; LOPES, A. T.; RODRIGUES, A.; VAREJãO, F. M.; OLIVEIRA-SANTOS, T. Monthly Energy Consumption Forecast: A Deep Learning Approach. In: *International Joint Conference on Neural Networks (IJCNN)*. Anchorage, USA: IEEE, 2017. p. 4283–4290. Cited on page 121.

BERRIEL, R. F.; LOPES, A. T.; SOUZA, A. F. de; OLIVEIRA-SANTOS, T. Deep Learning Based Large-Scale Automatic Satellite Crosswalk Classification. *IEEE Geoscience and Remote Sensing Letters*, 2017. ISSN 1545-598X. Cited 3 times on pages 27, 40, and 121.

BERRIEL, R. F.; PAIXAO, T. M.; NICCHIO, I.; OLIVEIRA-SANTOS, T. Controlling First-Person Character Movement: A Low-Cost Camera-based Tracking Alternative for Virtual Reality. In: *Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. Foz do Iguacu, Brazil: IEEE, 2018. p. 418–425. Cited on page 121.

BERRIEL, R. F.; ROSSI, F. S.; SOUZA, A. F. de; OLIVEIRA-SANTOS, T. Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning Approach. *Computers & Graphics*, v. 68, p. 32–42, 2017. ISSN 0097-8493. Cited 4 times on pages 27, 40, 72, and 121.

BERRIEL, R. F.; TORRES, L. T.; CARDOSO, V. B.; GUIDOLINI, R.; BADUE, C.; SOUZA, A. F. de; OLIVEIRA-SANTOS, T. Heading Direction Estimation Using Deep Learning with Automatic Large-scale Data Acquisition. In: *International Joint Conference on Neural Networks (IJCNN)*. Rio de Janeiro, Brazil: IEEE, 2018. Cited 3 times on pages 27, 40, and 120.

BILEN, H.; FERNANDO, B.; GAVVES, E.; VEDALDI, A.; GOULD, S. Dynamic Image Networks for Action Recognition. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, USA: IEEE, 2016. Cited on page 91.

BOJARSKI, M.; CHOROMANSKA, A.; CHOROMANSKI, K.; FIRNER, B.; JACKEL, L.; MULLER, U.; ZIEBA, K. VisualBackProp: efficient visualization of CNNs. In: *ICRA*. Brisbane, Australia: IEEE, 2018. Cited on page 96.

BOJARSKI, M.; TESTA, D. D.; DWORAKOWSKI, D.; FIRNER, B.; FLEPP, B.; GOYAL, P.; JACKEL, L. D.; MONFORT, M.; MULLER, U.; ZHANG, J.; ZHANG, X.; ZHAO, J.; ZIEBA, K. End to End Learning for Self-Driving Cars. *arXiv preprint arXiv:1604.07316*, 2016. Cited 2 times on pages 39 and 80.

BRAHMBHATT, S.; HAYS, J. DeepNav: Learning to Navigate Large Cities. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, USA: IEEE, 2017. p. 3087–3096. Cited on page 39.

CARDOSO, V.; OLIVEIRA, J.; TEIXEIRA, T.; BADUE, C.; MUTZ, F.; OLIVEIRA-SANTOS, T.; VERONESE, L.; SOUZA, A. F. D. A Model-Predictive Motion Planner for the IARA autonomous car. In: *International Conference on Robotics and Automation (ICRA)*. Singapore: IEEE, 2017. p. 225–230. Cited on page 75.

CASELLA, G.; BERGER, R. L. *Statistical Inference*. Boston, USA: Cengage Learning, 2002. v. 2. Cited on page 61.

CHEN, C.; SEFF, A.; KORNHAUSER, A.; XIAO, J. DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving. In: *The IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, 2015. Cited on page 25.

CHEN, L.; PAPANDREOU, G.; KOKKINOS, I.; MURPHY, K.; YUILLE, A. L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 40, n. 4, p. 834–848, 2018. Cited on page 83.

CHETLUR, S.; WOOLLEY, C.; VANDERMERSCH, P.; COHEN, J.; TRAN, J.; CATANZARO, B.; SHELHAMER, E. cuDNN: Efficient primitives for Deep Learning. *arXiv preprint arXiv:1410.0759*, 2014. Cited on page 63.

CHOLLET, F. et al. *Keras*. 2015. <https://github.com/fchollet/keras>. Cited on page 78.

CIMPOI, M.; MAJI, S.; KOKKINOS, I.; MOHAMED, S.; VEDALDI, A. Describing Textures in the Wild. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Columbus, USA: IEEE, 2014. Cited on page 91.

CORREIA-SILVA, J. R.; BERRIEL, R. F.; BADUE, C.; SOUZA, A. F. de; OLIVEIRA-SANTOS, T. Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data. In: *International Joint Conference on Neural Networks (IJCNN)*. Rio de Janeiro, Brazil: IEEE, 2018. Cited on page 120.

CORREIA-SILVA, J. R.; BERRIEL, R. F.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Copycat cnn: Are random non-labeled data enough to steal knowledge from black-box models? *Pattern Recognition*, Elsevier, v. 113, p. 107830, 2021. Cited on page 119.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, Springer, v. 2, n. 4, p. 303–314, 1989. Cited on page 29.

DOSOVITSKIY, A.; ROS, G.; CODEVILLA, F.; LOPEZ, A.; KOLTUN, V. CARLA: An Open Urban Driving Simulator. In: *Conference on Robot Learning*. Mountain View, USA: PMLR, 2017. Cited on page 32.

EITZ, M.; HAYS, J.; ALEXA, M. How Do Humans Sketch Objects? *ACM Transactions on Graphics*, v. 31, n. 4, p. 44:1–44:10, 2012. Cited on page 92.

FREITAS, R.; PAIXÃO, T.; BERRIEL, R.; SOUZA, A.; BADUE, C.; SANTOS, T. Relevant Traffic Light Localization via Deep Regression. In: SBC. *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional (BRACIS)*. Salvador, Brazil, 2019. p. 460–471. Cited on page 120.

FUKUSHIMA, K.; MIYAKE, S. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern recognition*, Elsevier, v. 15, n. 6, p. 455–469, 1982. Cited on page 30.

GHILARDI, M.; JUNIOR, J.; MANSSOUR, I. Crosswalk localization from low resolution

satellite images to assist visually impaired people. *IEEE computer graphics and applications*, IEEE, 2016. Cited on page 38.

GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Columbus, USA: The CVF, 2014. Cited on page 83.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Sardinia, Italy: PMLR, 2010. v. 9, p. 249–256. Cited 2 times on pages 57 and 73.

GUIDOLINI, R.; BADUE, C.; BERGER, M.; VERONESE, L. de P.; SOUZA, A. F. D. A simple yet effective obstacle avoider for the IARA autonomous car. In: *International Conference on Intelligent Transportation Systems (ITSC)*. Rio de Janeiro, Brazil: IEEE, 2016. p. 1914–1919. Cited on page 75.

GUIDOLINI, R.; SOUZA, A. F. D.; MUTZ, F.; BADUE, C. Neural-based model predictive control for tackling steering delays of autonomous cars. In: *International Joint Conference on Neural Networks (IJCNN)*. Anchorage, USA: IEEE, 2017. p. 4324–4331. Cited on page 75.

GUO, Y.; SHI, H.; KUMAR, A.; GRAUMAN, K.; ROSING, T.; FERIS, R. SpotTune: Transfer Learning through Adaptive Fine-tuning. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, USA: IEEE, 2019. Cited 4 times on pages 85, 94, 96, and 97.

GUPTA, S.; DAVIDSON, J.; LEVINE, S.; SUKTHANKAR, R.; MALIK, J. Cognitive Mapping and Planning for Visual Navigation. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, USA: IEEE, 2017. p. 7272–7281. Cited on page 39.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep Residual Learning for Image Recognition. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, USA: IEEE, 2016. Cited 3 times on pages 83, 93, and 99.

HERNANDEZ, D. C.; FILONENKO, A.; SEO, D.; JO, K.-H. Laser scanner based heading angle and distance estimation. In: *International Conference on Industrial Technology (ICIT)*. Seville, Spain: IEEE, 2015. p. 1718–1722. Cited on page 40.

HERUMURTI, D.; UCHIMURA, K.; KOUTAKI, G.; UEMURA, T. Urban Road Network extraction based on Zebra Crossing Detection from a very high resolution RGB aerial image and DSM data. In: *Signal-Image Technology & Internet-Based Systems (SITIS), 2013 International Conference on*. Kyoto, Japan: IEEE, 2013. p. 79–84. Cited on page 37.

HINTON, G.; VINYALS, O.; DEAN, J. Distilling the Knowledge in a Neural Network. In: *Conference on Neural Information Processing Systems – Deep Learning Workshop*. Vancouver, Canada: NIPS, 2014. Cited on page 34.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, Wiley Online Library, v. 160, n. 1, p. 106–154, 1962. Cited on page 30.

IVANCHENKO, V.; COUGHLAN, J.; SHEN, H. Detecting and Locating Crosswalks using a Camera Phone. In: *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. Anchorage, USA: IEEE, 2008. p. 1–8. Cited 2 times on pages 38 and 58.

JIA, Y.; SHELHAMER, E.; DONAHUE, J.; KARAYEV, S.; LONG, J.; GIRSHICK, R.; GUADARRAMA, S.; DARRELL, T. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*, 2014. Cited on page 63.

JUNG, S.; YOUN, J.; SULL, S. Efficient lane detection based on spatiotemporal images. *Transactions on Intelligent Transportation Systems*, v. 17, n. 1, p. 289–295, 2016. Cited on page 39.

KAISER, Ł.; ROY, A.; VASWANI, A.; PARMAR, N.; BENGIO, S.; USZKOREIT, J.; SHAZEER, N. Fast Decoding in Sequence Models Using Discrete Latent Variables. In: *International Conference on Machine Learning (ICML)*. Stockholm, Sweden: PMLR, 2018. Cited on page 89.

KIM, J.; PARK, C. End-to-end ego lane estimation based on sequential transfer learning for self-driving cars. In: *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. Honolulu, USA: IEEE, 2017. Cited on page 33.

KINGMA, D. P.; BA, J. Adam: A Method for Stochastic Optimization. In: *International Conference on Learning Representations (ICLR)*. San Diego, USA: ICLR, 2015. Cited on page 73.

KIRKPATRICK, J.; PASCANU, R.; RABINOWITZ, N.; VENESS, J.; DESJARDINS, G.; RUSU, A. A.; MILAN, K.; QUAN, J.; RAMALHO, T.; GRABSKA-BARWINSKA, A. et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, National Academy of Sciences, v. 114, n. 13, p. 3521–3526, 2017. Cited on page 86.

KOESTER, D.; LUNT, B.; STIEFELHAGEN, R. Zebra crossing detection from aerial imagery across countries. In: *International Conference on Computers Helping People with Special Needs*. Linz, Austria: Springer, 2016. p. 27–34. Cited on page 38.

KOSKELA, K.; NURKKALA, V.; KALERMO, J.; JÄRVILEHTO, T. Low-cost driving simulator for driver behavior research. In: *Proceedings of the International Conference on Computer Graphics and Virtual Reality (CGVR)*. Las Vegas, USA: CSREA Press, 2011. p. 83–86. Cited on page 32.

KRAUSE, J.; STARK, M.; DENG, J.; FEI-FEI, L. 3D Object Representations for Fine-Grained Categorization. In: *International Conference on Computer Vision Workshops (ICCVW)*. Sydney, Australia: IEEE, 2013. Cited on page 92.

KRIZHEVSKY, A. *Learning multiple layers of features from tiny images*. Dissertation (Master of Science) — University of Toronto, 2009. Cited 3 times on pages 32, 91, and 99.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: *Advances in Neural Information Processing Systems*. Nevada, USA: NeurIPS, 2012. p. 1097–1105. Cited 5 times on pages 43, 65, 73, 83, and 94.

LAKE, B. M.; SALAKHUTDINOV, R.; TENENBAUM, J. B. Human-level concept learning through probabilistic program induction. *Science*, v. 350, n. 6266, p. 1332–1338, 2015. Cited on page 91.

LECUN, Y.; BOSER, B.; DENKER, J.; HENDERSON, D.; HOWARD, R.; HUBBARD, W.; JACKEL, L. Handwritten digit recognition with a back-propagation network. In: *Advances in neural information processing systems*. Denver, USA: NIPS, 1989. Cited 2 times on pages 30 and 43.

LEE, S.; KWEON, I. S.; KIM, J.; YOON, J. S.; SHIN, S.; BAILO, O.; KIM, N.; LEE, T.-H.; HONG, H. S.; HAN, S.-H. VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition. In: *International Conference on Computer Vision (ICCV)*. Venice, Italy: IEEE, 2017. p. 1965–1973. Cited on page 39.

LI, Z.; HOIEM, D. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 40, n. 12, p. 2935–2947, 2017. Cited on page 86.

LIN, T.-Y.; MAIRE, M.; BELONGIE, S.; HAYS, J.; PERONA, P.; RAMANAN, D.; DOLLÁR, P.; ZITNICK, C. L. Microsoft COCO: Common Objects in Context. In: *European Conference on Computer Vision*. Zurich, Switzerland: Springer, 2014. p. 740–755. Cited on page 32.

LINNAINMAA, S. The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors. *Master's Thesis (in Finnish), Univ. Helsinki*, 1970. Cited on page 30.

MAJI, S.; RAHTU, E.; KANNALA, J.; BLASCHKO, M.; VEDALDI, A. Fine-Grained Visual Classification of Aircraft. *arXiv preprint arXiv:1306.5151*, 2013. Cited on page 91.

MALLYA, A.; DAVIS, D.; LAZEBNIK, S. Piggyback: Adapting a Single Network to Multiple Tasks by Learning to Mask Weights. In: *European Conference on Computer Vision (ECCV)*. Munich, Germany: IEEE, 2018. Cited 13 times on pages 83, 84, 85, 86, 89, 90, 91, 92, 93, 94, 96, 97, and 98.

MALLYA, A.; LAZEBNIK, S. PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City: IEEE, 2018. Cited 4 times on pages 85, 93, 97, and 98.

MANCINI, M.; RICCI, E.; CAPUTO, B.; BULÒ, S. R. Adding New Tasks to a Single Network with Weight Transformations using Binary Masks. In: *European Conference on Computer Vision Workshops (ECCVW)*. Munich, Germany: IEEE, 2018. Cited 12 times on pages 83, 84, 85, 90, 91, 92, 93, 94, 95, 96, 97, and 98.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. Cited on page 29.

MELLO, J. P. V. de; PAIXÃO, T. M.; BERRIEL, R.; REYES, M.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Deep learning-based type identification of volumetric mri sequences. In: *International Conference on Pattern Recognition (ICPR)*. Virtual-Milano, Italy: IEEE, 2020. Cited on page 119.

MELLO, J. P. V. de; TABELINI, L.; BERRIEL, R. F.; PAIXÃO, T. M.; SOUZA, A. F.

D.; BADUE, C.; SEBE, N.; OLIVEIRA-SANTOS, T. Deep traffic light detection by overlaying synthetic context on arbitrary natural images. *Computers & Graphics*, Elsevier, v. 94, p. 76–86, 2021. Cited on page 119.

Microsoft. *Visual Object Tagging Tool*. 2018. <https://github.com/Microsoft/VoTT>. Visited on 08/19/2021. Cited on page 33.

MUNDER, S.; GAVRILA, D. M. An Experimental Study on Pedestrian Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 28, n. 11, p. 1863–1868, 2006. Cited on page 91.

MUTZ, F.; VERONESE, L. P.; OLIVEIRA-SANTOS, T.; AGUIAR, E. de; CHEEIN, F. A. A.; SOUZA, A. F. D. Large-scale mapping in complex field scenarios using an autonomous car. *Expert Systems with Applications*, v. 46, p. 439–462, 2016. Cited on page 75.

NETZER, Y.; WANG, T.; COATES, A.; BISSACCO, A.; WU, B.; NG, A. Y. Reading Digits in Natural Images with Unsupervised Feature Learning. In: *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*. Granada, Spain: NeurIPS, 2011. Cited on page 91.

NILSBACK, M.-E.; ZISSERMAN, A. Automated Flower Classification over a Large Number of Classes. In: *Indian Conference on Computer Vision, Graphics & Image Processing*. Bhubaneswar, India: IEEE, 2008. Cited 2 times on pages 91 and 92.

OQUAB, M.; BOTTOU, L.; LAPTEV, I.; SIVIC, J. Is Object Localization for Free? - Weakly-Supervised Learning With Convolutional Neural Networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, USA: IEEE, 2015. Cited on page 34.

PAIXãO, T. M.; BERRIEL, R. F.; BOERES, M. C. S.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. A Deep Learning-Based Compatibility Score for Reconstruction of Strip-Shredded Text Documents. In: *SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. Foz do Iguaçu, Brazil: IEEE, 2018. Cited on page 120.

PAIXãO, T. M.; BERRIEL, R. F.; BOERES, M. C. S.; KOERICH, A. L.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Fast(er) Reconstruction of Shredded Text Documents via Self-Supervised Deep Asymmetric Metric Learning. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Virtual: IEEE, 2020. Cited on page 119.

PAIXãO, T. M.; BERRIEL, R. F.; BOERES, M. C. S.; KOERICH, A. L.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Self-supervised Deep Reconstruction of Mixed Strip-shredded Text Documents. *Pattern Recognition*, 2020. ISSN 0031-3203. Cited on page 119.

PAULA, M. B. de; JUNG, C. R.; SILVEIRA, L. D. Automatic on-the-fly extrinsic camera calibration of onboard vehicular cameras. *Expert Systems with Applications*, Elsevier, v. 41, n. 4, p. 1997–2007, 2014. Cited on page 69.

PAULUCIO, L. S.; PAIXÃO, T. M.; BERRIEL, R. F.; SOUZA, A. F. D.; BADUE, C.; OLIVEIRA-SANTOS, T. Product Categorization by Title Using Deep Neural Networks

as Feature Extractor. In: *International Joint Conference on Neural Networks (IJCNN)*. Glasgow, UK: IEEE, 2020. Cited on page 120.

POGGI, M.; NANNI, L.; MATTOCCIA, S. Crosswalk Recognition Through Point-Cloud Processing and Deep-Learning Suited to a Wearable Mobility Aid for the Visually Impaired. In: *International Conference on Image Analysis and Processing*. Genova, Italy: Springer, 2015. p. 282–289. Cited 2 times on pages 38 and 68.

POLINO, A.; PASCANU, R.; ALISTARH, D. Model Compression via Distillation and Quantization. In: *International Conference on Learning Representations (ICLR)*. Vancouver, Canada: ICLR, 2018. Cited on page 34.

POSSATTI, L. C.; GUIDOLINI, R.; CARDOSO, V. B.; BERRIEL, R. F.; PAIXãO, T. M.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars. In: *International Joint Conference on Neural Networks (IJCNN)*. Budapest, Hungary: IEEE, 2019. Cited on page 120.

REBUFFI, S.; KOLESNIKOV, A.; SPERL, G.; LAMPERT, C. H. iCaRL: Incremental Classifier and Representation Learning. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, USA: IEEE, 2017. Cited on page 86.

REBUFFI, S.-A.; BILEN, H.; VEDALDI, A. Learning multiple visual domains with residual adapters. In: *Advances in Neural Information Processing Systems*. Long Beach, USA: NeurIPS, 2017. Cited 10 times on pages 83, 84, 85, 86, 89, 90, 91, 92, 93, and 94.

REBUFFI, S.-A.; BILEN, H.; VEDALDI, A. Efficient parametrization of multi-domain deep neural networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA: IEEE, 2018. Cited 5 times on pages 83, 84, 85, 89, and 94.

RIVEIRO, B.; GONZÁLEZ-JORGE, H.; MARTÍNEZ-SÁNCHEZ, J.; DÍAZ-VILARIÑO, L.; ARIAS, P. Automatic detection of zebra crossings from mobile LiDAR data. *Optics & Laser Technology*, Elsevier, v. 70, p. 63–70, 2015. Cited 2 times on pages 37 and 68.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958. Cited on page 29.

ROSENFELD, A.; TSOTSOS, J. K. Incremental Learning Through Deep Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. Early Access. Cited 6 times on pages 85, 89, 91, 92, 93, and 94.

RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATHY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; FEI-FEI, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, v. 115, n. 3, p. 211–252, 2015. Cited 4 times on pages 32, 43, 91, and 92.

RUSU, A. A.; RABINOWITZ, N. C.; DESJARDINS, G.; SOYER, H.; KIRKPATRICK, J.; KAVUKCUOGLU, K.; PASCANU, R.; HADSELL, R. Progressive Neural Networks. *arXiv preprint arXiv:1606.04671*, 2016. Cited on page 86.

SALEH, B.; ELGAMMAL, A. Large-scale Classification of Fine-Art Paintings: Learning The Right Metric on The Right Feature. *International Journal for Digital Art History*, 2016. Cited on page 92.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA: IEEE, 2018. Cited on page 90.

SANGINETO, E.; NABI, M.; CULIBRK, D.; SEBE, N. Self Paced Deep Learning for Weakly Supervised Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 41, n. 3, p. 712–725, March 2019. Cited on page 34.

SARCINELLI, R.; GUIDOLINI, R.; CARDOSO, V. B.; PAIXãO, T. M.; BERRIEL, R. F.; AZEVEDO, P.; SOUZA, A. F. D.; BADUE, C.; OLIVEIRA-SANTOS, T. Handling pedestrians in self-driving cars using image tracking and alternative path generation with Frenét frames. *Computers & Graphics*, v. 84, p. 173–184, 2019. Cited on page 120.

SCHMIDHUBER, J. Deep Learning. *Scholarpedia*, v. 10, n. 11, p. 32832, 2015. Revision #184887. Cited on page 30.

SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2014. Cited 2 times on pages 43 and 56.

SOOMRO, K.; ZAMIR, A. R.; SHAH, M. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *arXiv preprint arXiv:1212.0402*, 2012. Cited on page 91.

STALLKAMP, J.; SCHLIPSING, M.; SALMEN, J.; IGEL, C. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, Elsevier, v. 32, p. 323–332, 2012. Cited on page 91.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. Going deeper with convolutions. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, USA: IEEE, 2015. p. 1–9. Cited on page 43.

TABELINI, L.; BERRIEL, R.; PAIXãO, T. M.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. PolyLaneNet: Lane Estimation via Deep Polynomial Regression. In: *International Conference on Pattern Recognition (ICPR)*. Virtual-Milano, Italy: IEEE, 2020. Cited on page 119.

TABELINI, L.; BERRIEL, R.; PAIXãO, T. M.; BADUE, C.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Keep your Eyes on the Lane: Real-time Attention-guided Lane Detection. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Virtual: IEEE, 2021. Cited on page 119.

TAN, M.; LE, Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In: *International Conference on Machine Learning (ICML)*. Long Beach, California: PMLR, 2019. Cited on page 25.

TORRES, L. T.; PAIXãO, T. M.; BERRIEL, R. F.; SOUZA, A. F. D.; BADUE, C.; SEBE, N.; OLIVEIRA-SANTOS, T. Effortless Deep Training for Traffic Sign Detection Using Templates and Arbitrary Natural Images. In: *International Joint Conference on Neural Networks (IJCNN)*. Budapest, Hungary: IEEE, 2019. Cited 2 times on pages 27 and 120.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: *Advances in neural information processing systems*. Long Beach, USA: NeurIPS, 2017. p. 5998–6008. Cited on page 25.

VERONESE, L. de P.; AGUIAR, E. de; NASCIMENTO, R. C.; GUIVANT, J.; CHEEIN, F. A. A.; SOUZA, A. F. D.; OLIVEIRA-SANTOS, T. Re-emission and satellite aerial maps applied to vehicle localization on urban environments. In: *International Conference on Intelligent Robots and Systems (IROS)*. Hamburg, Germany: IEEE, 2015. p. 4285–4290. Cited on page 75.

VERONESE, L. de P.; GUIVANT, J.; CHEEIN, F. A. A.; OLIVEIRA-SANTOS, T.; MUTZ, F.; AGUIAR, E. de; BADUE, C.; SOUZA, A. F. D. A light-weight yet accurate localization system for autonomous cars in large-scale and complex environments. In: *International Conference on Intelligent Transportation Systems (ITSC)*. Rio de Janeiro, Brazil: IEEE, 2016. p. 520–525. Cited 2 times on pages 40 and 75.

WANG, K.; YAN, X.; ZHANG, D.; ZHANG, L.; LIN, L. Towards human-machine cooperation: Self-supervised sample mining for object detection. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA: IEEE, 2018. Cited on page 33.

WANG, S.; PAN, H.; ZHANG, C.; TIAN, Y. Rgb-d image-based detection of stairs, pedestrian crosswalks and traffic signs. *Journal of Visual Communication and Image Representation*, Elsevier, v. 25, n. 2, p. 263–272, 2014. Cited 2 times on pages 38 and 68.

WANG, X.; MURPHEY, Y. L.; KOCHHAR, D. S. MTS-DeepNet for lane change prediction. In: *International Joint Conference on Neural Networks (IJCNN)*. Vancouver, Canada: IEEE, 2016. p. 4571–4578. Cited on page 71.

WANG, X.; YU, F.; DOU, Z.-Y.; DARRELL, T.; GONZALEZ, J. E. SkipNet: Learning Dynamic Routing in Convolutional Networks. In: *European Conference on Computer Vision (ECCV)*. Munich, Germany: Springer, 2018. Cited 2 times on pages 34 and 86.

WELINDER, P.; BRANSON, S.; MITA, T.; WAH, C.; SCHROFF, F.; BELONGIE, S.; PERONA, P. *Caltech-UCSD Birds 200*. Pasadena, USA, 2010. Cited on page 92.

WHO: World Health Organization. *Visual impairment and blindness*. 2014. Available at: <http://www.who.int/mediacentre/factsheets/fs282/en/>. Cited on page 49.

WU, Z.; NAGARAJAN, T.; KUMAR, A.; RENNIE, S.; DAVIS, L. S.; GRAUMAN, K.; FERIS, R. BlockDrop: Dynamic Inference Paths in Residual Networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, USA: IEEE, 2018. Cited 2 times on pages 34 and 86.

XU, D.; RICCI, E.; OUYANG, W.; WANG, X.; SEBE, N. Multi-Scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, USA: IEEE, 2017. Cited on page 83.

YU, J.; YANG, L.; XU, N.; YANG, J.; HUANG, T. Slimmable Neural Networks. In: *International Conference on Learning Representations*. New Orleans, USA: ICLR, 2019. Cited 6 times on pages 17, 34, 86, 90, 99, and 100.

YUE, X.; WU, B.; SESHIA, S. A.; KEUTZER, K.; SANGIOVANNI-VINCENTELLI, A. L. A lidar point cloud generator: From a virtual world to autonomous driving. In: *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*. Yokohama, Japan: ACM, 2018. p. 458–464. Cited on page 33.

ZAGORUYKO, S.; KOMODAKIS, N. Wide Residual Networks. In: *British Machine Vision Conference (BMVC)*. York, UK: BMVA Press, 2016. Cited on page 92.

ZAVAREZ, M. V.; BERRIEL, R. F.; OLIVEIRA-SANTOS, T. Cross-Database Facial Expression Recognition Based on Fine-Tuned Deep Convolutional Network. In: *SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. Niterói, Brazil: IEEE, 2017. p. 405–412. Cited on page 121.

ZHANG, W.; MURPHEY, Y. L.; WANG, T.; XU, Q. Driver yawning detection based on deep convolutional neural learning and robust nose tracking. In: *International Joint Conference on Neural Networks (IJCNN)*. Killarney, Ireland: IEEE, 2015. p. 1–8. Cited on page 71.

ZHOU, B.; SUN, Y.; BAU, D.; TORRALBA, A. Interpretable basis decomposition for visual explanation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Munich, Germany: Springer, 2018. p. 119–134. Cited on page 96.

Appendix

APPENDIX A – List of Publications

During the development of this thesis, many research opportunities were presented to us, and allowed us to make several contributions. Many of these contributions were published in internationally renowned conferences and journals. A list of all publications can be found below (newest first).

- (BADUE et al., 2021): Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., **Berriel, R. F.**, Paixão, T., Mutz, F., Veronese, L., Oliveira-Santos, T., De Souza, A. F.; *Self-driving cars: A survey* (Expert Systems with Applications, 2021, Qualis A1);

- (TABELINI et al., 2021): Tabelini, L., **Berriel, R. F.**, Paixão, T. M., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Keep your Eyes on the Lane: Real-time Attention-guided Lane Detection* (CVPR'21, Qualis A1);

- (CORREIA-SILVA et al., 2021): Correia-Silva, J. R., **Berriel, R. F.**, Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Copycat CNN: Are Random Non-Labeled Data Enough to Steal Knowledge from Black-box Models?* (Pattern Recognition, 2021, Qualis A1);

- (MELLO et al., 2021): De Mello, J. P. V., Tabelini, L., **Berriel, R. F.**, Paixão, T. M., De Souza, A. F., Badue, C., Sebe, N., Oliveira-Santos, T.; *Deep traffic light detection by overlaying synthetic context on arbitrary natural images* (Computers & Graphics, Qualis A2);

- (PAIXãO et al., 2020a): Paixão, T. M., **Berriel, R. F.**, Boeres, M. C. S., Koerich, A. L., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Fast(er) Reconstruction of Shredded Text Documents via Self-Supervised Deep Asymmetric Metric Learning* (CVPR'20, Qualis A1);

- (PAIXãO et al., 2020b): Paixão, T. M., **Berriel, R. F.**, Boeres, M. C. S., Koerich, A. L., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Self-supervised Deep Reconstruction of Mixed Strip-shredded Text Documents* (Pattern Recognition, 2020, Qualis A1);

- (TABELINI et al., 2020): Tabelini, L., **Berriel, R. F.**, Paixão, T. M., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *PolyLaneNet: Lane Estimation via Deep Polynomial Regression* (ICPR'20, Qualis A2);

- (MELLO et al., 2020): De Mello, J. P. V., Paixão, T. M., **Berriel, R. F.**, Reyes,

M., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Deep Learning-based Type Identification of Volumetric MRI Sequences* (ICPR'20, Qualis A2);

- (PAULUCIO et al., 2020): Paulucio, L. S., Paixão, T. M., **Berriel, R. F.**, De Souza, A. F., Badue, C., Oliveira-Santos, T.; *Product Categorization by Title Using Deep Neural Networks as Feature Extractor* (IJCNN'20, Qualis A1);

- (BERRIEL et al., 2019): **Berriel, R. F.**, Lathuilière S., Nabi, M., Klein, T., Oliveira-Santos, T., Sebe, N., Ricci, E.; *Budget-Aware Adapters for Multi-Domain Learning* (ICCV'19, Qualis A1);

- (SARCINELLI et al., 2019): Sarcinelli, R., Guidolini, R., Cardoso, V. B., Paixão, T. M., **Berriel, R. F.**, Azevedo, P., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Handling pedestrians in self-driving cars using image tracking and alternative path generation with Frenét frames* (Computer & Graphics, 2019, Qualis A2);

- (TORRES et al., 2019): Tabelini, L., Paixão, T. M., **Berriel, R. F.**, De Souza, A. F., Badue, C., Sebe, N., Oliveira-Santos, T.; *Effortless Deep Training for Traffic Sign Detection Using Templates and Arbitrary Natural Images* (IJCNN'19, Qualis A1);

- (ARRUDA et al., 2019): Arruda, V. F., Paixão, T. M., **Berriel, R. F.**, De Souza, A. F., Badue, C., Sebe, N., Oliveira-Santos, T.; *Cross-Domain Car Detection Using Unsupervised Image-to-Image Translation: From Day to Night* (IJCNN'19, Qualis A1);

- (POSSATTI et al., 2019): Possatti, L. C., Guidolini, R., Cardoso, V. B., **Berriel, R. F.**, Paixão, T. M., Badue, C., De Souza, A. F., Oliveira-Santos, T.; *Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars* (IJCNN'19, Qualis A1);

- (FREITAS et al., 2019): De Freitas, R. H., Paixao, T. M., **Berriel, R. F.**, De Souza, A. F., Badue, C., Oliveira-Santos, T.; *Relevant Traffic Light Localization via Deep Regression* (BRACIS'19, Qualis B2);

- (BERRIEL et al., 2018): **Berriel, R. F.**, Tabelini, L., Cardoso, V. B., Guidolini, R., Badue, C., De Souza, A. F., Oliveira-Santos T.; *Heading Direction Estimation Using Deep Learning with Automatic Large-scale Data Acquisition* (IJCNN'18, Qualis A1);

- (CORREIA-SILVA et al., 2018): Correia-Silva, J. R., **Berriel, R. F.**, Badue, C., De Souza, A. F., Oliveira-Santos T.; *Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data* (IJCNN'18, Qualis A1);

- (PAIXãO et al., 2018): Paixao, T. M., **Berriel, R. F.**, Boeres, M. C., Badue, C.,

De Souza, A. F., Oliveira-Santos, T.; *A Deep Learning-Based Compatibility Score for Reconstruction of Strip-Shredded Text Documents* (SIBGRAPI'18, Qualis B1);

- (BERRIEL et al., 2018): **Berriel, R. F.**, Paixao, T. M., Nicchio, I., Oliveira-Santos, T.; *Controlling First-Person Character Movement: A Low-Cost Camera-based Tracking Alternative for Virtual Reality* (SBGames'18, Qualis B2);
- (BERRIEL et al., 2017b): **Berriel, R. F.**, Lopes, De Souza, A. F., Oliveira-Santos, T.; *Deep Learning Based Large-Scale Automatic Satellite Crosswalk Classification* (IEEE Geoscience and Remote Sensing Letters, 2017, Qualis A2);
- (BERRIEL et al., 2017): **Berriel, R. F.**, Rossi, F. S., De Souza, A. F., Oliveira-Santos T.; *Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning Approach* (Computers & Graphics, 2017, Qualis A2)
- (ZAVAREZ; BERRIEL; OLIVEIRA-SANTOS, 2017): Zavarez, M. V., **Berriel, R. F.**, Oliveira-Santos, T.; *Cross-database facial expression recognition based on fine-tuned deep convolutional network* (SIBGRAPI'17, Qualis B1);
- (BERRIEL et al., 2017a): **Berriel, R. F.**, Lopes, A. T., Rodrigues, A., Varejao, F. M., Oliveira-Santos, T.; *Monthly energy consumption forecast: A deep learning approach* (IJCNN'17, Qualis A1).