Towards Reliable Computer Vision Feature Extraction by Novel Autoencoder Methods

Vom Fachbereich Informatik der Technischen Universität Kaiserslautern zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)

genehmigte Dissertation

von

Steve DIAS DA CRUZ

Datum der wissenschaftlichen Aussprache: 21. Dezember 2022
Dekan: Prof. Dr. Jens SCHMITT
Berichterstatter: Prof. Dr. Didier STRICKER
Berichterstatter: Prof. Dr. Marius KLOFT

DE-386

I would like to dedicate this thesis to
my loving parents, Patricia and Luis, my partner Carole and our dog Tikaani.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and acknowledgments. This dissertation contains fewer than 85,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 100 figures.

Steve DIAS DA CRUZ
December 2022

Acknowledgements

A PhD thesis is never an accomplishment of a single isolated person. It is interesting to look back and to appreciate the many connections you make during this time. Each one having an impact on the final achievement, in one way or the other. It is time to show gratitude towards those who had the greatest impact and who helped to accomplish this thesis.

I would like to express my deepest gratitude to Bertram Taetz, who was advising me for several years. Your sincere interest in my research topics, your questions and your help kept me focused, especially when I was loosing my motivation. It was great to have someone who was genuinely interested, who understood my mathematical background and who pushed me whenever I needed it. Without you, Bertram, this thesis would not have been possible.

I am thankful to Prof. Didier Stricker who supervised my thesis and integrated me in his department. The augmented vision group is a great team with a lot of kind and amazing people. Although my time there was limited, I always felt welcome.

Thomas Stifter supported me at IEE S.A. and gave me the freedom to develop myself in the areas I was most interested in. While the circumstances were often challenging, together we always managed to find the best way through.

I want to thank the following people who influenced and helped me in my scientific, but also personal development - in no particular order: Hans-Peter Beise, Jerry Dormans, Bruno Mirbach, Frédéric Grandidier, Jun Wang, Werner Bieck, Una Karahasanovic, Oliver Wasenmüller and Udo Schröder.

Further, I am grateful to those who read this thesis and provided me valuable feedback: Nareg Minaskan (and his delicious cookies and cakes), Jérôme Burelbach and Jan Sokolowski. I want to thank Krishna Kumar for making publicly available the excellent LaTeX template used to prepare this thesis.

This thesis would not have been possible without the public funding provided by the Luxembourg National Research Fund (FNR) under grant number 13043281. It is great to see that so much effort is put into supporting the research landscape in Luxembourg and I am happy to have been a part of it.

Abstract

The generally unsupervised nature of autoencoder models implies that the main training metric is formulated as the error between input images and their corresponding reconstructions. Different reconstruction loss variations and latent space regularization have been shown to improve model performances depending on the tasks to solve and to induce new desirable properties like disentanglement. Nevertheless, measuring the success in, or enforcing properties by, the input pixel space is a challenging endeavor. In this work, we want to make more efficient use of the available data and provide design choices to be considered in the recording or generation of future datasets to implicitly induce desirable properties during training. To this end, we propose a new sampling technique which matches semantically important parts of the image while randomizing the other parts, leading to salient feature extraction and a neglection of unimportant details. Further, we propose to recursively apply a previously trained autoencoder model, which can then be interpreted as a dynamical system with desirable properties for generalization and uncertainty estimation.

The proposed methods can be combined with any existing reconstruction loss. We give a detailed analysis of the resulting properties on various datasets and show improvements on several computer vision tasks: image and illumination normalization, invariances, synthetic to real generalization, uncertainty estimation and improved classification accuracy by means of simple classifiers in the latent space.

These investigations are adopted in the automotive application of vehicle interior rear seat occupant classification. For the latter, we release a synthetic dataset with several fine-grained extensions such that all the aforementioned topics can be investigated in isolation, or together, in a single application environment. We provide quantitative evidence that machine learning, and in particular deep learning methods cannot readily be used in industrial applications when only a limited amount of variation is available for training. The latter can, however, often be the case because of constraints enforced by the application to be considered and financial limitations.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The success of machine learning, and in particular deep learning based methods for solving a large variety of tasks, has become apparent. The universal applicability of the aforementioned methods in industry still goes along with a plethora of questions and challenges, particularly in case of safety critical applications [187, 254, 28, 196, 97, 6]. Special care needs to be taken to record datasets for industrial applications [243, 63, 17]. Tackling these challenges is often time consuming and relies on considerable financial support, to the detriment of fast product development. In general, the industrial actor wants to deploy reliable products while remaining cost-efficient. This requires substantial progress in the development of both model-driven and data-driven methods.

For the former, the generalization capacities and robustness of the statistical methods need to be improved. For the latter, the data collection needs to be simplified and the available data needs to be used more efficiently. This thesis scientifically investigates both approaches for computer vision tasks, yielding valuable quantitative results and insights that readily extend from the automotive application to a wide range of other topics.

The recordings of a multitude of sensors can be used to observe the surroundings of a system of interest. Each individual type of sensor usually has its advantages and disadvantages and the trade-offs to be made depend on the task to be solved and the requirements of the system. Depth maps, recorded by stereo camera systems or time-of-flight (ToF) sensors, have the advantage of neglecting most of the textures of the objects in the scene and providing the distance of each pixel to the camera. However, stereo camera systems need not only to be calibrated correctly (which can change over the lifetime of a deployed sensor) to work well, but also more than a single camera is needed. ToF cameras need to be calibrated as well and can be influenced by sunlight and multi-path, multi-reflected scattered light. The recordings of a RADAR system provide additional information about the speed of the objects in the scene by exploiting the Doppler effect. However, the resolution of the RADAR system is usually

quite low and the recording can be affected by motion, e.g. if the vehicle is in motion [52]. Camera sensors are relatively cheap and reliable, provide high resolution images and more visual details to be used by a machine learning model. Since we are adopting them in the vehicle interior, bad weather conditions barely affect the recording - except for illumination changes. Although more computing power is needed to process the image data, the potential variety of tasks to be solved by camera sensors is large. It was decided to focus on images recorded by a camera system because they are well understood, intuitive to work with and robust against most exterior effects. Further, we restrict ourselves to single independent frames instead of considering video data or multiple frames. This makes the task more challenging, but we can focus on the fundamental questions presented later. Also, the incorporation of multiple frames would make the system more sophisticated and hence more expensive.

Image data finds its use in many application areas of high industrial relevance, e.g. information provided by images can be used to determine the pose of humans [240] or even animals [164], segment the images [274], recognize activities [61], detect objects [281] or segment each instance [26], detect anomalies [127], identify people [176], answer questions [261] and generate novel images [88]. Many of these achievements require ground truth labellings different from simple classification labels. This causes a large financial and time-consuming overhead, since basically all the labelling needs to be done, or at least verified, manually. It is clear that any additional labelling information can only be helpful for training statistical methods, improve the performances or extend the functionalities if necessary. However, to *keep it simple, stupid* (KISS), it was decided to refrain from using more informative labellings and we focus on developing and improving solutions for the most basic approaches.

However, image based datasets and tasks have their downsides as well. Image based classifiers can be susceptible to small changes in the image, i.e. adversarials [97], biased towards shapes or textures [144], be sensitive to illumination [4] or overconfident about their prediction [169]. It is hence paramount to robustify the classifier before deploying it to a system [13, 221].

Due to the wide range of cutting-edge applications of image data, we argue that it is essential to understand and develop the basics of deep learning methods. In this thesis, we focus on analyzing and developing more reliable features to be used by a classifier to solve computer vision problems. Each industrial application has its potential and challenges to be considered during the data collection process, the software development phase and the training of machine learning based methods. While all of the proposed methods will be evaluated in the vehicle interior, we also take care to show their benefits on datasets commonly used by the research community.

1.1 Motivation

Autoencoder (AE), and closely related encoder-decoder models, have seen numerous successful applications partially due to their bottleneck design, which is useful for feature reduction [272] and compression [40], but also by providing beneficial properties, e.g. disentanglement [36]. The aforementioned desirable characteristics can either be achieved using dedicated priors in the latent space, e.g. by a Gaussian prior [128], triplet loss [268], discretization [245], or by non identical input-target pairs, e.g. in case of denoising and inpainting [266] or super-resolution [273]. Further, autoencoder based solutions can be used to detect anomalies [7, 83]. It is also common to pre-train autoencoder models on large datasets of general images to learn a meaningful and generic feature extraction such that the resulting encoder can be fixed during a second fine-tuning stage [186]. During the latter, an additional classifier head using the extracted features as input can be trained on the dataset of the actual task to be solved [162]. Since we were referring to safety critical application in the previous section, we also want to highlight that autoencoders have been shown to provide advantages and prove their usefulness on medical applications [35, 82, 244, 46, 171, 234].

An important argument in favor of autoencoders is the accessibility of the resulting lower-dimensional representation of the training and test data. It is hence possible to visualize the learned data manifold and to test the representation for some desirable properties after training. For example, one can test the latent space representation for disentanglement, sometimes referred to as independence of the factors of variation, or for a meaningful interpolation in-between two encoded input samples [184]. Most importantly, the representation and clusters of the samples with respect to their target label can be assessed. Due to the bottleneck design, the lower dimensional feature representation can become meaningful up to a point where it can be used for image retrieval [226]. On many occasions the decoder part can be removed and only the encoder together with the lower-dimensional representation are used for the task to be solved, in particular in case of classification [162]. This reduces the model complexity for a potential deployment. We also report arguments in favor of autoencoder models compared to other generative models in Section 2.7.

Autoencoders have been proven to work well on many applications. Their model design is intuitive and simple, can easily be extended, provide useful insights due to their lower-dimensional latent space and yet provide interesting results on many tasks. Their elegance and their mentioned properties were the main arguments to decide to focus the investigations of this thesis on the deep learning model architecture of autoencoder models. Our main goal is to build autoencoder models with some useful properties to achieve more reliable feature extraction, which in turn can be used to improve classifier performance on computer vision tasks.

1.2 Problem formulation and challenges

The problems and challenges considered in this thesis stem from the industrial application of determining seat occupancy in the vehicle interior: for each seat position in a vehicle, the system should detect whether it is empty or a child on a child seat, an infant in an infant seat or an adult is sitting on it. At first glance, the overall complexity of this task seems manageable: as in any machine learning based solution, the engineers would start with defining a test matrix, perform a data recording campaign and train several statistical methods to compare the performances against each other. While this heuristic works and provides good results, most of the challenges become apparent once the system should be commercialized [155, 107, 110].

It is expected that certain guarantees about the reliability of the deployed system can be formulated, e.g. reduce the likelihood of failures, identify failures and their causes when they occur. However, the robustness and performance of such a system might be more difficult to evaluate than anticipated. The system should be robust against new child seats appearing on the market after the system has been deployed. Even for child seats observed in the training process, the system should not be misled by everyday phenomena: what happens if the child seat is rotated slightly or placed incorrectly in the car? Some parents might put toys on the child seat to distract their children, put stickers on the seats to make them more easily recognizable or put a sun-protection on it to protect their child. Further, how should a model behave when more exotic sceneries occur, that have not been covered by the data recording campaign? Each of us has placed animals and the most random objects inside a car, e.g. backpacks, furniture, beverage, food, computers, clothes or plants. The passengers are moving inside the car, they can be reading books, sleeping, playing on their smartphone or sitting in the most (un)comfortable positions. The vehicle is moving and it is being used during day and nighttime, during all seasons and weather conditions. It is hence paramount that a deployed system is also robust against all environmental conditions. While this already sounds much harder than before, the situation becomes even more complicated due to the windows inside a vehicle interior. Since the camera is observing the seat positions, it is likely that its sensor will also perceive objects outside the vehicle. Hence, the situation and objects, i.e. the overall scenery, which could potentially mislead a classifier using the camera images as input, is even more diverse.

An example of the aforementioned problem is illustrated in Fig. 1.1. A pre-trained YOLO [211] model was used to detect objects in an image from a vehicle interior. Although it detects the passengers correctly, the model is also affected by objects inside the vehicle (a book) and outside the vehicle (a car). While it does not constitute a problem for this particular case, we want to use this example as an illustration that any classifier can potentially also be mislead by the objects in the scene - either from the inside or the outside. We also provide an example in Fig. 1.2 regarding the volatility of a pre-trained YOLO model's detections for a

Fig. 1.1 Detected objects inside a vehicle interior by a pre-trained YOLO model. The model detects the people correctly, but it is also affected by objects inside the vehicle (book - which is actually a box of a toy) and outside the vehicle (car).

scenery for which we artificially changed the illumination. The predictions are unstable, i.e. the child is sometimes wrongly classified, the dimensions of the bounding boxes vary and objects are detected which are not present in the image. A model needs to be robust against these illumination changes, since they do not alter the semantics of the image. Moreover, changes in the brightness and saturation can occur more dominantly in the field once a model is deployed.

All the aforementioned challenges are related to model robustness against situations happening inside the vehicle the model was trained on. This framework becomes even more difficult once a model should work in an unknown vehicle interior. In case the images are recorded inside a single vehicle, which is of course expected to test the feasibility of an idea, all of the images show the same, or similar background. This hinders the transferability of a model which observed images from this single vehicle interior only. However, machine learning based models, and specifically neural networks trained in a single environment, often take into account non-relevant characteristics of the specific environmental conditions in an uncontrollable manner [237]. This means in practice that the performance drops drastically in a new, unseen vehicle interior. We will highlight this problem quantitatively in Section 5 and Section 6. Even if we consider the same car model, the interior design can vary: the color and texture of the seats can change, new or different features can be included or removed. When the rear bench is turned down the vehicle interior looks different as well. This situation gets harder if we want the model to be deployed to a completely different car model, potentially of a different

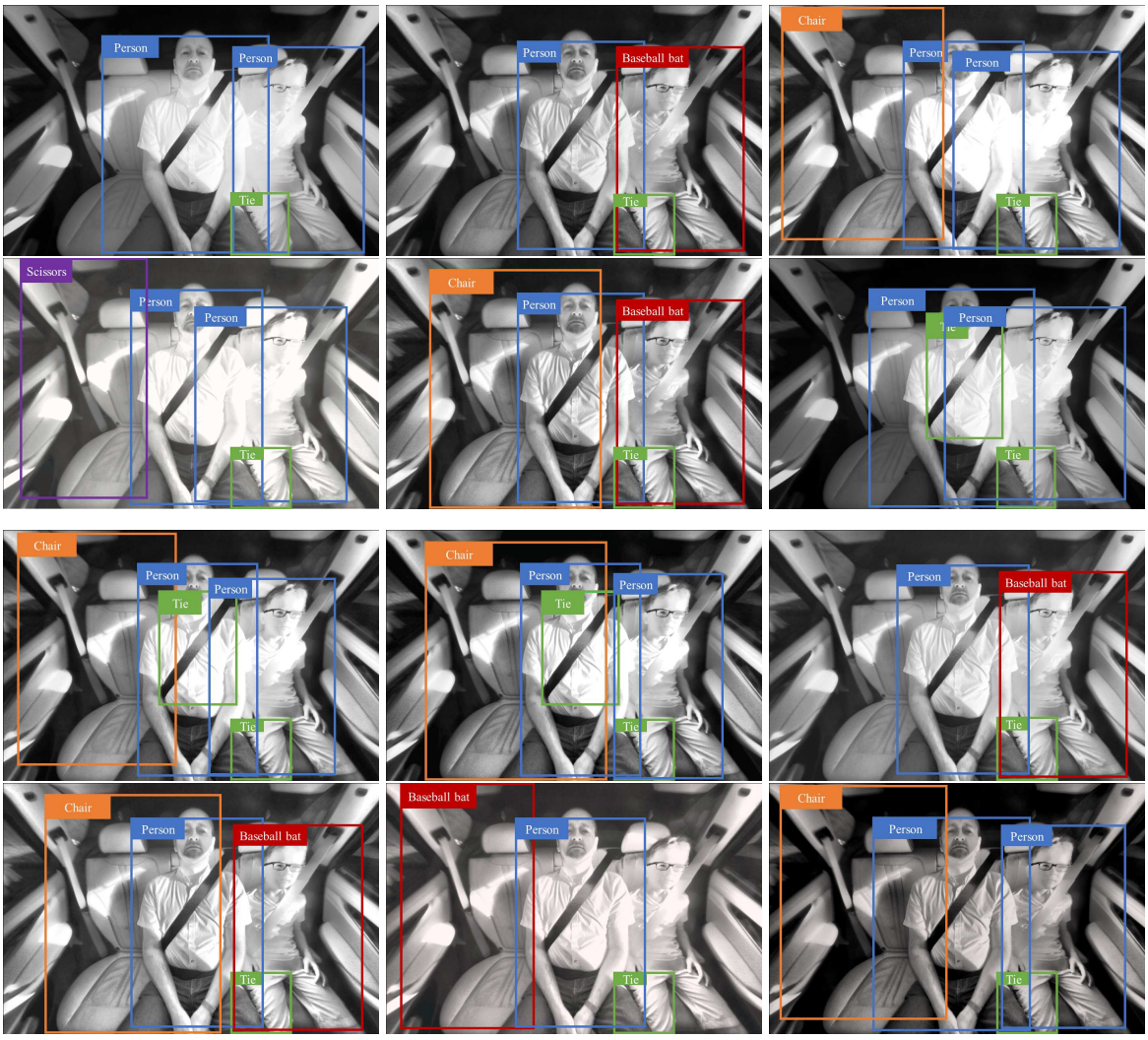


Fig. 1.2 The prediction of a pre-trained YOLO model can be volatile for the same scenery under changing illumination (achieved using transformations from Albumentations [32]). The child is sometimes wrongly classified, the sizes of the bounding boxes vary and non-present objects are detected. Ideally, a model should be robust in its prediction for the presented variations.

manufacturer. Since evaluating the model robustness against all the aforementioned cases is time-consuming and expensive, it would be desirable to minimize the number of vehicles necessary to obtain a quasi-universal classifier applicable in any vehicle interior. Otherwise, the above tests need to be repeated for each new vehicle interior resulting in a large financial overhead. As we will show in this thesis, determining the number of vehicles and the type of vehicles necessary for obtaining a quasi-universal classifier, and providing guarantees that the classifier is indeed quasi-universally applicably, is a hard challenge to solve.

While the problems explained above seem narrowed down to a fixed application, we want to highlight that the underlying questions are important for many machine learning applications, especially since they can be broken down into fundamental questions and research directions: synthetic to real generalization, robustness and invariances with respect to distributional shifts (illumination, environment, new vehicle interiors, new objects of known classes), training under a limited amount of variation and uncertainty estimation. One of the potential solutions would be to augment the training data by including synthetic images for which the ground truth labelling is basically readily available and the amount of images to generate can be arbitrarily large. This is particularly interesting for dangerous or expensive edge cases. In the best case, invariances could be learned on synthetic data and transferred to their real counterparts. However, the incorporation of synthetic data is still challenging due to the gap between synthetic and real distributions [280]. Next, the models trained for the vehicle interior should become illumination invariant. This means they should work for all possible illumination and environmental conditions such that the system's reliability is increased. It is crucial to improve the transferability to new vehicle interiors and generalization to new objects inside the vehicle when only a limited amount of variation was observed during training. This is important because it is not always possible to have a large amount and a diverse repository of objects and vehicles at one's disposal. However, since it is impossible to account for all potential variations occurring during the lifetime of a deployed system, it would be advantageous to quantify the model's predictive uncertainty. This means that the model should assess by itself the reliability of its prediction and whether the prediction can be trusted or not.

In general, the problems and challenges encourage the requirement for more reliable feature extractions to be used by a classifier. The final goal is to train a model to become more robust against distributional shifts of different origins. Overall, it is necessary to deepen our understanding of fundamental properties of deep learning models. In this thesis, we would like to propose novel training approaches for autoencoder models in order to advance the reduction of the distributional shifts regarding the aforementioned challenges. We want to make use of the available data more efficiently. We believe that the properties of autoencoder models, as mentioned in the previous section, are good candidates to alleviate at least some of the problems

described in this section. Further, we think that investigating and developing the fundamentals is important as it improves our basic understanding of the models to be used, particularly for safety critical applications. Hence, we will refrain from developing unnecessarily sophisticated models. Instead, we try to remain as simple as possible and focus on intuitive design choices, which are easier to interpret and understand, and hence to debug when it comes to failures. According to Tambon et al. [233], a machine learning-based safety-critical system should be robust, explainable and verifiable, provide uncertainty estimation and out-of-distribution detection and potentially, in the future, fulfill the ISO 26262 standard, which has not yet been defined for machine learning based solutions. With this thesis, our proposed model design choices and investigations directly address model robustness, uncertainty estimation and out-of-distribution detection. The lower dimensional latent space representation of autoencoder models might improve explainability and verifiability, as well as the fulfillment of the ISO 26262 standard, but we neither investigated nor discussed these topics in this work.

1.3 Contributions

This thesis investigates novel strategies for training neural networks based on the autoencoder model architecture with the focus on computer vision, and in particular on images of higher visual complexity, as measured and compared in Section 2.2. We consider the challenges of improving generalization, reducing the need for collecting real images and modeling uncertainty in an autoencoder model's prediction. We only consider classification tasks and focus on improving basic autoencoder models which are particularly interesting due to their bottleneck design. The new training methods presented in this work combine ideas of most of the approaches presented in Section 1.1.

Since the investigated industrial application and the provided real images are proprietary, the first achievement of the thesis was to develop a synthetic data generation pipeline in Blender [41] imitating the industrial application [53]. To this end, several datasets focusing on different challenges were generated and published to enable the reproducibility of the developed methods and the corresponding results. SVIRO [53] consists of images of 10 different vehicle interiors such that vehicle-to-vehicle generalization and robustness to novel class variations can be tested. In SVIRO-Illumination [54] each scenery is rendered under 10 different illumination conditions, which is difficult to realize with real images, so that invariances with respect to illumination can be investigated. SVIRO-Uncertainty [59] provides the means to test uncertainty estimation on several difficulty levels. In SVIRO-NoCar [58] the vehicle is removed and we show that this induces invariances with respect to the vehicle interior and its background on synthetic data which can then be transferred to real data to some extent. SVIRO-InterCar renders identical

sceneries in different vehicle interiors. The different datasets are explained and compared in Chapter 3. For most publications we released a dataset for the topic to be investigated, proposed new methods to improve the performances and tested the methods against other standard approaches using both the novel dataset and other commonly used research datasets.

The second contribution was an analysis on SVIRO, as reported in Chapter 6, to highlight to the research community that the challenges from the vehicle interior introduce interesting research questions [56]: if all training images are collected in the same vehicle interior, how could we improve the transfer to a novel vehicle interior, without collecting new data? The dominant and constant background makes this a challenging task. This can be re-formulated as a more general challenge: how can we improve the transfer to a new environment when all training images are recorded in an identical, but different environment? This is also the case when the training images are recorded in a single room, or building, and the model needs to be deployed to a different room, or building. We propose a first simple approach based on denoising autoencoder models to improve the transferability.

One of the main contributions is the introduction of a novel training strategy entitled ***"Partially Impossible Reconstruction Loss"*** (PIRL) of which we provide two variations. The first variation of the PIRL is the weaker version, because it is tailored to the task to be solved and it needs a controlled variation of the task specific unwanted features. The features we want to become invariant against are varied and the features we assess as discriminative are kept identical or similar. Hence, this sampling strategy preserves the semantics while varying the unimportant features such that the model needs to focus on what remains constant. We propose to select as target for each input image a variation of the latter: one can select the same scene under different illumination and/or with different backgrounds. In this setting, it can be used for illumination normalization [54], see Chapter 7, where identical scenes under different environmental conditions are needed.

The first variation of the PIRL will lead to good results, particularly for normalization and in case human poses need to be preserved. However, it can be challenging to apply it to a lot of commonly recorded datasets, especially when the dataset to be considered does not allow for a controlled variation of the unwanted features. To this end we introduce a second, stronger variation of the PIRL [58, 57] which can readily be applied to most existing datasets. Instead of sampling the same scene under a controlled variation, e.g. same scene under different illumination, we propose to use as target image a different image of the same class as the input. This approach implicitly uses label information in the input space, in contrast to the triplet loss in the latent space [268], and leads to a better latent space representation. This loss variation causes the model to learn invariances with respect to certain class variations that are

not important for the task at hand, e.g. clothes, human poses, textures. While illumination can still be removed with this approach, human poses will no longer be preserved.

We further show that the PIRL can be used to improve synthetic to real generalization [58] - but not without some additional considerations. We show in Chapter 8 that it is not sufficient to test generalization capacities and design choices for synthetic to real generalization on less visually complex datasets, e.g. MPI3D [81]. We propose to use a fixed pre-trained classification model as feature extractor together with an autoencoder model, where the latter is being fine-tuned on the synthetic to real task to solve. Finally, this approach can be improved by combining it with the PIRL: invariances, e.g. with respect to the background, can then be learned on synthetic data and transferred to real data to some extent, especially when real data can be integrated to reduce the synthetic gap. This improves the performances to generalize from one real vehicle interior to another one significantly.

The methods are adopted to the industrial application to illustrate that our novel design choices can improve the performance on a real application. Further, the methods are investigated on commonly used academic datasets in Chapter 5 to show that the novel training strategies are not limited to our particular application [54, 58, 57]. We show that the PIRL produces a good latent space representation on par with the triplet loss. Notwithstanding this achievement, the PIRL uses the label information in the input space and needs only two samples per batch element, in contrast to the triplet loss, which uses the labels in the latent space and needs three samples per batch element. Each experiment is combined with an ablation study to highlight the effect of the different design choices. We show the benefits and downsides of the PIRL and compare the performance against commonly used pre-trained and fine-tuned convolutional neural networks (CNN). The features that can be considered important and unimportant for the PIRL depend on the task to be solved: in case of image classification the discriminative characteristics of the different classes are important while we might not be interested in facial landmarks, human poses, illumination or backgrounds such that these features can be considered as unimportant. Our proposed sampling strategy can easily be used with any existing autoencoder model, reconstruction loss and sampling strategy.

Additionally, we show in Chapter 9 that the PIRL can also be used for uncertainty estimation and out-of-distribution detection [57], where it outperforms MC Dropout [73] and an ensemble of models [136]. This performance gain is highlighted exhaustively on several datasets. Since the PIRL improves performances on a wide range of tasks and datasets, it can hence be concluded that the PIRL leads to a more reliable feature extraction.

Lastly, we investigated a rarely used interpretation of autoencoder models [59] in Chapter 9: viewing the recursive application of a previously trained autoencoder model as a dynamical system [204]. The latter consists of attractors and basins of attraction [230] and we show

empirically that this can be used to generalize to test images. More importantly, we show that this property can be extended by means of Monte Carlo (MC) dropout to obtain a family of dynamical systems that provide good uncertainty estimations, particularly on datasets of higher visual complexity. Samples close to the training distribution converge robustly to attractors of the same class, or even to the same attractor, while exotic samples are unstable in their convergence and converge to different attractors, potentially of different classes.

The benchmark and all the datasets generated for this thesis can be found on and downloaded from our website - https://sviro.kl.dfki.de/. The website was created for hosting the datasets, reporting all results and provide information related to our investigations. The code implementations for reproducing many results presented in this work are hosted on our Github - https://github.com/SteveCruz account. A subset of our SVIRO dataset is also integrated in DomainBed - https://github.com/facebookresearch/DomainBed.

1.4 Thesis outline

We start with **Chapter 2** covering the preliminaries and background information necessary to follow the topics discussed in this thesis. We mention related work, the tools used in this thesis and provide a thorough introduction to autoencoder models.

Chapter 3 introduces the industrial application of occupant detection in the vehicle interior. We explain the provided proprietary dataset and the data generation pipeline used to generate the synthetic datasets. Further, we describe each of the different dataset extensions.

We detail most of our novel autoencoder training approaches in **Chapter 4**. The methods are derived in a common framework such that they can easily be used as standalone or combined. The remainder of the thesis follows these notations and expressions and the methods are evaluated in the following chapters.

We provide first fundamental results and investigations for some of the previously presented methods in **Chapter 5**. This should give the interested reader some guidelines to understand the effect of the second variation of the PIRL on commonly used datasets. Further, we perform a baseline evaluation on SVIRO and ORSS to highlight some of the challenges mentioned in the introduction and to establish an understanding for the problem and application.

In **Chapter 6** we investigate the problem of transferring the model from one vehicle to a new unseen one. We show that this problem formulation is indeed difficult to solve and provide a first simple approach where autoencoder models achieve results on par with commonly used classification models.

Image and illumination normalization is discussed in **Chapter 7**. We explain the problem in more detail and demonstrate that the first variation of the PIRL can be used to solve illumination normalization in some cases. This is highlighted on several datasets.

We provide a step by step investigation of different design choices for the transferability from synthetic to real images in **Chapter 8**. We start off with MPI3D and show that these insights are not sufficient for more visually complex datasets. Further, we report results using the first and second variation of the PIRL either together or without the extractor module. In case real data can be used during training, we demonstrate that it can be combined with the previous design choices to improve performance significantly: invariances can be learned on the synthetic data while the synthetic gap is reduced by integrating the real images. We extend our understanding by additional ablation studies using the multi-channel autoencoder approach and show the potential benefits of the latter, also in comparison to the extractor approach.

In **Chapter 9** we investigate uncertainty estimation and out-of-distribution detection on a large variety of different datasets. We present that SVIRO-Uncertainty is indeed challenging and that the different splits can be used to thoroughly assess a model's reliability. We use the second variation of the PIRL and the method based on attractors to show that both methods yield good uncertainty and out-of-distribution estimations.

Finally, we conclude the thesis and provide questions, ideas and guidelines for future investigations and research directions.

1.5 List of publications

This thesis is supported by the following chronological list of publications:

[53] **Steve Dias Da Cruz**, Oliver Wasenmüller, Hans-Peter Beise, Thomas Stifter, Didier Stricker. (2020). SVIRO: Synthetic Vehicle Interior Rear Seat Occupancy Dataset and Benchmark. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

[56] **Steve Dias Da Cruz**, Bertram Taetz, Oliver Wasenmüller, Thomas Stifter, Didier Stricker. (2021). Autoencoder Based Inter-vehicle Generalization for In-cabin Occupant Classification. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*.

[54] **Steve Dias Da Cruz**, Bertram Taetz, Thomas Stifter, Didier Stricker. (2021). Illumination Normalization by Partially Impossible Encoder-Decoder Cost Function. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

[58] **Steve Dias Da Cruz**, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder for Synthetic to Real Generalization: From Simple to More Complex Scenes. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR,* ***Best Industry Related Paper Award****)*.

[59] **Steve Dias Da Cruz**, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder Attractors for Uncertainty Estimation. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*.

[57] **Steve Dias Da Cruz**, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder and Partially Impossible Reconstruction Losses. In *Sensors*.

Other publications contributed to during the PhD thesis:

[19] Hans-Peter Beise, **Steve Dias Da Cruz**, Udo Schröder. (2021). On decision regions of narrow deep neural networks. In *Neural Networks*.

[18] Hans-Peter Beise, **Steve Dias Da Cruz**. (under review). Topological properties of basins of attraction and expressiveness of width bounded neural networks. In *Analysis and Applications*.

This thesis contains additional results previously unpublished:

- A novel variation of the multi-channel autoencoder approach combining real and synthetic images during training to learn removing the vehicle background of real images, see Section 4.7,
- An investigation on background removal using the multi-channel autoencoder, the extractor module and/or the second variation of the PIRL, see Section 10,
- Additional ablation and basic studies to improve the understanding of the proposed model designs, see Chapter 5,
- Results on combining the autoencoder attractor approach together with the second variation of the PIRL, see Section 9.2.5,
- A novel extension for SVIRO called SVIRO-InterCar where each scenery is rendered in different vehicle interiors, see Section 3.6.

Chapter 2

Preliminaries and related work

While the majority of the concepts presented in this thesis can be applied to a wide range of data types, e.g. images, text, time series, we limit ourselves to formulations considering image data only. Our input samples x are hence always of the size $x \in \mathbb{R}^{C \times H \times W}$, where C is the number of channels, H the height and W the width of the images. For ease of notation, the images are often flattened such that $x \in \mathbb{R}^{C \cdot H \cdot W}$, where x_i is the ith coordinate and $n = C \cdot H \cdot W \in \mathbb{N}$ is the dimension of x. We use both notations and representations interchangeably in this work. Finally, while it is common to represent image pixel values in the range $[0, 255]$, we represent image pixels to be in the range $[0, 1]$. This is common practice when working with images in deep learning and it also simplifies the reconstruction by autoencoder models since the neural networks need to output values in a smaller range.

In this chapter, we provide an introduction to the topics discussed in this thesis and describe the tools and the different datasets used. We introduce autoencoder models, several reconstruction losses and regularization techniques and provide evidence for the design choices made for the rest of the thesis. Next, we present generative models not based on the autoencoder network architecture and discuss why there are not suited for the problems investigated in this thesis. Further, we present some relevant and related work to the topics analyzed later and explain some of the challenges in more detail. Lastly, we introduce uncertainty estimation for machine learning model's predictions, dynamical systems and explain the evaluation metrics used to assess a model's performance. We assume that the reader is familiar with the basics of deep learning and neural networks, their definition and training procedure and refer to Goodfellow et al. [86] for a good overview and introduction into the latter. All experiments in this thesis were conducted using PyTorch [190]. Models which were pre-trained on ImageNet [51] and pre-defined were taken from torchvision [160].

2.1 Datasets

We provide a short overview and explanation of the datasets used in this work. Samples from the different datasets are shown in Fig. 2.1. Some of the datasets are plotted and used as grayscale even though they are available as RGB.

2.1.1 Vehicle interior

Some previous works have been investigating occupant classification [69, 197], seat-belt detection [16] or skeletal tracking [200] in the passenger compartment, but, as to best of our knowledge, no dataset was made publicly available. Publicly available realistic datasets for the vehicle interior are scarce. Some exceptions are the recently released AutoPOSE [222] and DriveAHead [220] datasets for driver head orientation and position, Drive&Act [161] a multi-modal benchmark for action recognition in automated vehicles and TICaM [123] for activity recognition and person detection. However, all these datasets have in common that they provide images for a single vehicle interior or from a single simulator.

Investigations regarding the tasks and challenges mentioned in Chapter 1 could also be performed in a different framework, as long as they reproduce the same limitations. KITTI [76] provides a wide range of different available annotations and benchmarks for vehicle exterior applications. Closely related are the Cityscapes dataset [42] for different segmentation tasks, ECP [27] for person detection in urban traffic scenes and JTA [67] for pedestrian pose estimation and tracking. On the other hand, there is COCO [147], a widely used benchmark for object detection, keypoint detection and panoptic and stuff segmentation as well as PASCAL VOC [66]. Similarly, with Open Images [3], the largest unified dataset for image classification, object detection and instance segmentation was released. Even though these datasets contribute a wide range of images and corresponding annotations, their provided data has intrinsically high background and intra-class variation due to their nature for the exterior application. These datasets can be used to benchmark models for their performance and push the state-of-the-art in specific tasks, as ImageNet [51] did for classification. It is, however, not possible to use these datasets to test the generalization to new environments and unseen intra-class variations for a larger range of tasks when only a limited amount of variability is available during training. However, deep learning-based approaches capture too much relevance between the information contained in the background and the task that the models are designed to solve [237]. In particular, those datasets cannot be used to benchmark applications for the (vehicle) interior regarding the challenges discussed in Chapter 1.

Hence, we decided to create SVIRO [53] and its extensions SVIRO-Illumination [54], SVIRO-NoCar [58], SVIRO-Uncertainty [59] and SVIRO-InterCar. Although SVIRO is a

synthetic dataset, it was designed specifically to test the transferability between different vehicles across multiple tasks. Moreover, together with all its extensions including different settings and splits, it provides a unified framework to investigate many topics and tasks together. The applicability to real infrared [53, 55] and depth images [70] was shown, but we will address this also later in this work. Further, we are using the proprietary ORSS dataset from IEE S.A. to assess some of the design choices on real images from a real application. On the other side, recent studies have shown the importance of synthetic data for the automotive industry [180, 241, 39]. With our SVIRO dataset and benchmarks we provide the means to analyze the generalization and reliability of machine learning-based approaches for different tasks when only a limited number of variations is available during training. We thereby address an important generalization issue on visually complex scenes, as defined in Section 2.2, on the example of the car interior. The SVIRO dataset and its extension will be introduced and discussed in more detail in Chapter 3.

2.1.2 Illumination

Recording identical, or similar, sceneries under different illumination or environmental conditions is a challenging task. Large-scale datasets for identical sceneries under different illumination conditions are currently scarce. The Deep Portrait Relighting Dataset [283] is based on the CelebA-HQ [120] dataset and contains human faces under different illumination conditions. However, the re-illumination has been added synthetically. We instead use the Extended Yale Face Database B [78], which is a dataset of real human faces with real illumination changes. While cross-seasons correspondence datasets prepared according to Larsson et al. [138] and based on RobotCar [158] and CMU Visual Localization [11] could be used for our investigation, the correspondences are usually not exact enough to have an identical scene under different conditions. Moreover, vehicles on the street which are dominantly visible, but changing in-between recordings, induce a large difference in the images. Alternative datasets such as St. Lucia Multiple Times of Day [79] and Nordland [183] suffer from similar problems. These datasets stem from the image correspondence search and SLAM community. We adopt the Webcam Clip Art [137] to include a dataset for the exterior environment with changing seasons and day times. The latter contains webcam images of outdoor regions from different places all over the world.

2.1.3 Synthetic to real

The annual VISDA challenge [192] hosts a benchmark for domain adaptation, including synthetic to real generalization, for different tasks, but solutions to different tasks are not com-

parable. Other common datasets for domain adaptation, e.g. Office-Home [250], DomainNet [193] and Open MIC [130], focus on a single task as well. Hence, to assess the transferability from synthetic to real images, we will consider training on MNIST [140] and applying the resulting model on real images of digits [50]. Further, we adopt the MPI3D [81] dataset which consists of objects under seven factors of variations being moved by a robotic arm. Each scenery is recorded by a real camera and rendered synthetically either by a realistic renderer or a simple (toy) renderer. This dataset allows to assess a more consistent analysis of the transferability, because each scenery is available under three conditions - toy, realistic and real. Moreover, since the environment is tractable and almost constant for all images, the setting and its conditions can be considered as similar to the vehicle interior.

2.1.4 Uncertainty and out-of-distribution

For assessing model uncertainty and out-of-distribution detection, we use several commonly used computer vision datasets for training. We then use the corresponding test data as in-distribution sets $\mathcal{D}_{in}$: MNIST, Fashion-MNIST [264], SVHN [177] and German Traffic Sign Recognition Dataset (GTSRB) [104]. The latter is an image classification dataset of photos from traffic signs. It contains 43 classes, but we limit ourselves to use 10 classes[1] only. This way the basic analyses performed later on are more manageable. For out-of-distribution $\mathcal{D}_{out}$ we use a subset of all $\mathcal{D}_{in}$ not coming from the training distribution and the test datasets from Omniglot [135], CIFAR10 [132], LSUN [270] (for which we use the train split) and Places365 [282]. These combinations of datasets are commonly used by the research community to analyze out-of-distribution detection and uncertainty estimation [95, 96, 49, 159, 149].

2.2 Visual complexity for computer vision datasets

As claimed in the introduction, in this work we are investigating extensions and contributions to autoencoder models, particularly for the case of visually more complex images from an industrial application. Consequently, to show that the datasets we are considering in this work are indeed visually more complex than other commonly used datasets, we adopt the same strategy as proposed by Rahane and Subramanian [205]. The authors propose to compute the mean Shannon Entropy and the mean grey level co-occurrence matrix over all the images to assess a dataset's complexity and compare datasets among each other. This should provide evidence that one dataset is visually more complex than another. We use scikit-image [247] to compute the mean Shannon Entropy and the mean grey level co-occurrence matrix and report

[1] We use the classes with labels 10 to 19, since the classes 0 to 8 are all speed limits

(a) CIFAR10

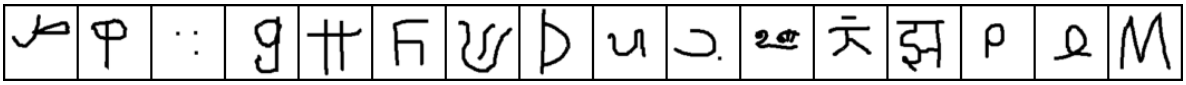
(b) Fashion-MNIST

(c) GTSRB

(d) LSUN

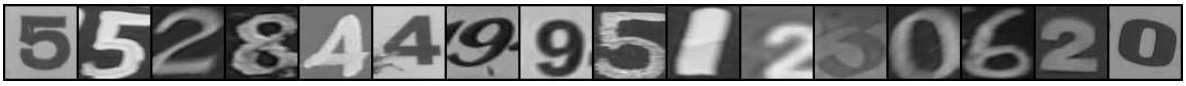
(e) MNIST

(f) MPI3D



(g) Omniglot

(h) Places365



(i) Real fonts

(j) SVHN

(k) Webcam

(l) Yale

Fig. 2.1 Overview and samples for all non-vehicle datasets used in this thesis.

Table 2.1 Overview of mean Shannon Entropy and mean grey level co-occurrence matrix for several datasets. Larger value ↑ means visually more complex. We used grayscale images, even when the dataset provides RGB images. Vehicle names in parentheses indicate which subset of the dataset was used.

Dataset	Shannon Entropy ↑	Grey level co-occurrence matrix ↑
GTSRB	6.078	9.580
ORSS (X5)	7.645	11.854
ORSS (Sharan)	7.304	11.422
SVIRO (Tesla)	7.087	11.180
SVIRO-Illumination (Cayenne)	6.464	10.403
SVIRO-Uncertainty	6.753	11.057
MNIST	2.608	3.513
Fashion-MNIST	4.116	6.110
SVHN	5.952	8.704
CIFAR10	5.935	8.755

the results for several datasets in Table 2.1. It can be observed that GTSRB and the vehicle interior datasets have a higher visual complexity compared to datasets commonly used for basic analyses. Consequently, we will start our first investigations in the next sections on GTSRB to motivate some of the first decisions and design choices.

2.3 Autoencoders

An ***autoencoder*** (AE) [14] is a particular type of neural network (NN) which tries to replicate its input $x \in \mathbb{R}^{C\times H\times W}$. It consists of an encoder $\mathrm{e}_\phi : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{d_l}$ and decoder module $\mathrm{d}_\theta : \mathbb{R}^{d_l} \to \mathbb{R}^{C\times H\times W}$ both being neural networks with network parameters θ and ϕ respectively. The encoder transforms, or encodes, the input x into $z = \mathrm{e}_\phi(x)$, while the decoder uses the encoding $z \in \mathbb{R}^{d_l}$ to reconstruct the input $\hat{x} = \mathrm{d}_\theta(z) \in \mathbb{R}^{C\times H\times W}$. A simplified visualization of an autoencoder model is represented in Fig. 2.2. Autoencoder models $f(\cdot) = \mathrm{d}_\theta(\mathrm{e}_\phi(\cdot))$ are trained similarly to standard neural networks, i.e. using (mini-)batches and (stochastic) gradient descent to minimize a loss function. The training goal of an autoencoder model is to minimize the reconstruction error

$$\mathcal{L}_R(x, f(x); \theta, \phi) = \mathcal{L}_R(x, \mathrm{d}_\theta(\mathrm{e}_\phi(x)); \theta, \phi) = \mathcal{L}_R(x, \hat{x}; \theta, \phi) = r(x, \hat{x}; \theta, \phi) \tag{2.1}$$

measuring the difference between the input image x and the autoencoder reconstruction $\hat{x}$ according to some metric. In case of the mean squared error (MSE) Eq. (2.1) could be

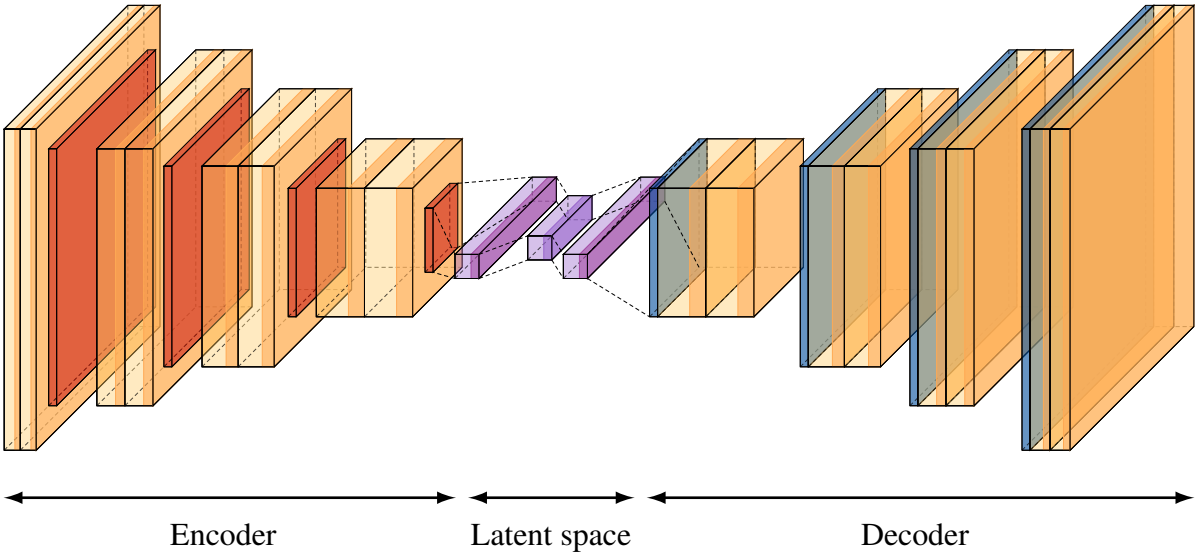


Fig. 2.2 Simplified illustration of an autoencoder model with its encoder and decoder module. The input image should be reconstructed by going through a bottleneck. The latter forms a latent space which helps to focus on the salient features. Illustrated using PlotNeuralNet [91].

formulated as

$$\mathcal{L}_R(x,\hat{x};\theta,\phi) = \frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{x}_i)^2. \tag{2.2}$$

Without any additional constraint, the autoencoder model could potentially learn to approximate the identity function. Hence, usually, the propagation through the encoder and decoder makes use of a bottleneck structure such that the dimension of z, denoted as d_l, is much smaller than the one of x. Consequently, the representation z is a compressed representation of x and the lower dimensional representation is often referred to as ***latent space***. The aforementioned bottleneck forces the autoencoder model to focus and prioritize on the salient features such that the model manages to learn more useful properties. Hence, autoencoder models have seen applications in dimensionality reduction and feature learning [272], but also in generative tasks [23]. The layers of the autoencoder can either be fully connected or (transposed) convolutional layers together with non-linear activation functions. Fully connected and convolutional layers are usually combined with max-(un)pooling, batch-normalization and dropout, but other layers are possible as well. It is common practice to use one or two fully connected layers before and after the latent space, also when convolutional layers are used for the other layers. A dense bottleneck allows to combine the features globally to find meaningful properties, while a convolutional bottleneck would continue to look for spatially local features. Lastly, we want to highlight that the reconstruction task can be considered as a regression task, where the neural

network should predict a (continuous) value for each pixel in the image. This is one of the reasons why it is preferable to consider pixel values in the range $[0, 1]$.

2.3.1 Classification

The autoencoder can be trained without adopting any classification task during training, i.e. completely in an unsupervised way. However, we are interested in using the autoencoder model as a feature extraction module and to perform a classification on a dedicated problem afterwards. It is possible to attach a classification head, or an entire network, to the latent space representation or the output of the autoencoder model respectively. Both approaches could be trained end-to-end during the previously introduced unsupervised autoencoder training. However, we will adopt a different strategy where we train a classifier on the latent space representation after the autoencoder model finished training. After training the autoencoder model in a first stage, we retrieve the latent space representation for all training samples in the second stage. Then, in the third stage, we train a classifier on the training data latent space representation, as illustrated in Fig 2.3. For the latter we use several classifiers in this work: linear classifier, k-nearest neighbour (KNN) [111], support vector machine (SVM) [232], random forest (RForest) [20] and a single hidden layer multi-layer perceptron (MLP) [86]. This means that the latent space representation needs to be in favour for training a classifier in a second stage, i.e. the better the autoencoder model learned to learn discriminative and meaningful features, the better a classifier should work on the latent space representation after the autoencoder model has finished training. The goal of this thesis to extract reliable features from the input images is hence assessed by the above heuristic: the autoencoder module is trained independently from the classifier and the former will be responsible to generate and provide features robust against distributional shifts. The quality and universality of the extracted features, i.e. the success of our autoencoder model design, is then determined by the classification accuracy of the classifier in the latent space of the test images.

2.4 Reconstruction losses

The main driving force in training an autoencoder is the reconstruction error measuring the similarity between the target image and the autoencoder reconstruction. We will present commonly used reconstruction losses in this section and conclude with a dedicated experimental section comparing the effect of using different losses during training.

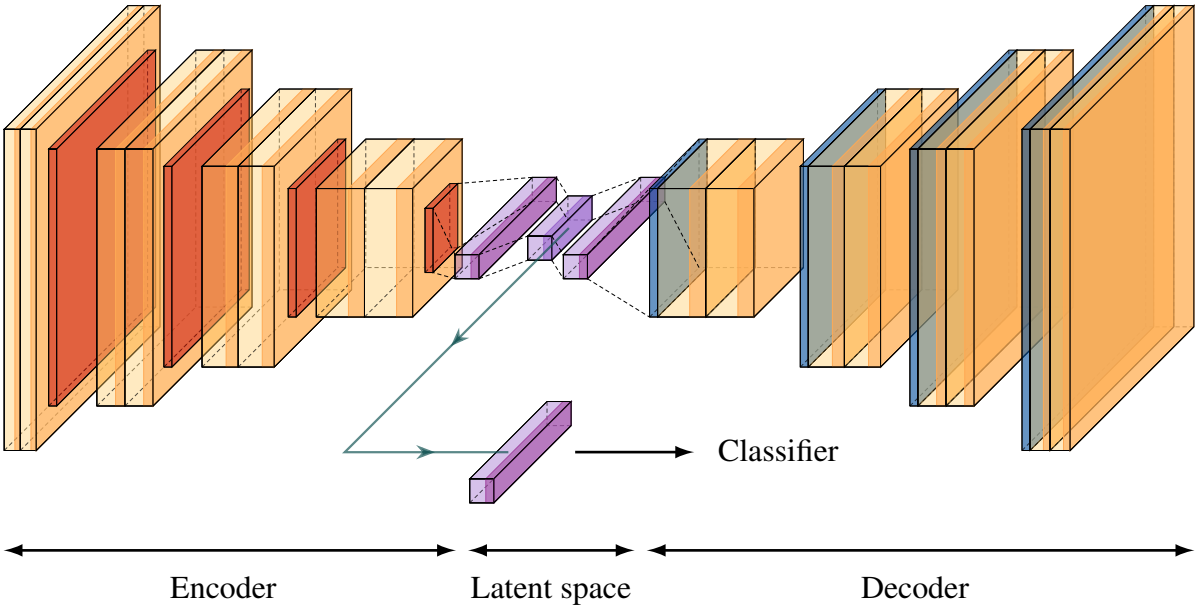


Fig. 2.3 Simplified illustration of the same autoencoder model as in Fig. 2.2, but with a classifier using the latent space representation as input. Illustrated using PlotNeuralNet [91].

2.4.1 Pixel-wise errors

One of the most used norms to measure the reconstruction error is the family of ℓ_p norms [103]. The latter measures the pixel-wise error between the reconstruction $\hat{x}$ and the target image x

$$\|x-\hat{x}\|_p = \left(\sum_{i=1}^{n} |x_i - \hat{x}_i|^p\right)^{\frac{1}{p}}, \tag{2.3}$$

for $p \geq 1$. In most cases the ℓ_1 norm

$$\|x-\hat{x}\|_1 = \sum_{i=1}^{n} |x_i - \hat{x}_i| \tag{2.4}$$

or the ℓ_2 norm

$$\|x-\hat{x}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - \hat{x}_i)^2} \tag{2.5}$$

is used. Closely related to the latter is the mean squared error (MSE)

$$\text{MSE}(x,\hat{x}) = \frac{1}{n}\sum_{i=1}^{n} (x_i - \hat{x}_i)^2. \tag{2.6}$$

The ℓ_2 norm and MSE are more sensitive to outliers, while the ℓ_1 norm is less sensitive, but not differentiable at 0. Regarding this issue, it is also possible to combine the advantages of both the ℓ_1 and MSE loss together by adopting the Huber loss [170]. The latter is less sensitive to outliers due to the ℓ_1 norm and improves smoothness around 0 due do the MSE error

$$\text{Huber}(x,\hat{x}) = \begin{cases} \frac{1}{2}(x_i - \hat{x}_i)^2, & \text{if } |x_i - \hat{x}_i| < \delta \\ \delta(|x_i - \hat{x}_i| - \frac{1}{2}\delta), & \text{otherwise} \end{cases} \tag{2.7}$$

for some $\delta > 0$. In case of binary images, e.g. MNIST, it can also make sense to use the binary cross entropy (BCE) loss to measure the reconstruction error. In this case the pixel values should either be 0 or 1 and the loss is computed by

$$\text{BCE}(x,\hat{x}) = -\frac{1}{n}\sum_{i=1}^{n}\left(x_i \log \hat{x}_i + (1-x_i)\log\left(1-\hat{x}_i\right)\right). \tag{2.8}$$

2.4.2 Structural similarity index measure (SSIM)

While in some experiments we will use the MSE and ℓ_1 norm, in most cases we will use a more advanced pixel-wise error metric: the structural similarity index measure (SSIM) [258]. The latter uses patches a and b of size $m \times m$ and computes three comparisons between the patches regarding the luminance $l(\cdot,\cdot)$, contrast $c(\cdot,\cdot)$ and structure $s(\cdot,\cdot)$

$$l(a,b) = \frac{2\mu_a\mu_b + c_1}{\mu_a^2 + \mu_b^2 + c_1} \tag{2.9}$$

$$c(a,b) = \frac{2\sigma_a\sigma_b + c_2}{\sigma_a^2 + \sigma_b^2 + c_2} \tag{2.10}$$

$$s(a,b) = \frac{\sigma_{ab} + c_3}{\sigma_a\sigma_b + c_3}, \tag{2.11}$$

where μ_a and μ_b are the means, σ_a and σ_b the variances and σ_{ab} the covariance of a and b respectively. Variables c_1 and c_2 are small-valued constants to ensure numerical stability and $c_3 = c_2/2$. The SSIM between both patches is then computed by

$$\text{SSIM}(a,b) = l(a,b)^\alpha \cdot c(a,b)^\beta \cdot s(a,b)^\gamma \tag{2.12}$$

In this work, we set $\alpha = \beta = \gamma = 1$ such that the above formula can be reduced to

$$\text{SSIM}(a,b) = \frac{2\mu_a\mu_b + c_1}{\mu_a^2 + \mu_b^2 + c_1} \frac{2\sigma_a\sigma_b + c_2}{\sigma_a^2 + \sigma_b^2 + c_2} \frac{\sigma_{ab} + c_3}{\sigma_a\sigma_b + c_3} \tag{2.13}$$

$$= \frac{2\mu_a\mu_b + c_1}{\mu_a^2 + \mu_b^2 + c_1} \frac{2\sigma_a\sigma_b + c_2}{\sigma_a^2 + \sigma_b^2 + c_2} \frac{2\sigma_{ab} + c_2}{2\sigma_a\sigma_b + c_2} \tag{2.14}$$

$$= \frac{(2\mu_a\mu_b + c_1)(2\sigma_{ab} + c_2)}{(\mu_a^2 + \mu_b^2 + c_1)(\sigma_a^2 + \sigma_b^2 + c_2)}. \tag{2.15}$$

According to commonly applied settings [258], we set $c_1 = 0.01^2$ and $c_2 = 0.03^2$. To compute the SSIM between a reconstruction $\hat{x}$ and its target x, denoted as $\text{SSIM}(\text{x}, \hat{\text{x}})$ one needs to slide a window of size $m \times m$ over both images and compute the SSIM for each pair of patches at each pixel location. The value of m can be chosen freely, but it can depend on the dataset used. In this work, we chose $m = 11$. Since $\text{SSIM}(a,b) \in [-1,1]$ and $\text{SSIM}(a,b) = 1$ if a and b are exactly the same, one needs to use

$$1 - \text{SSIM}(\text{x}, \hat{\text{x}}), \tag{2.16}$$

to define the loss function for training the autoencoder model. The benefit of adopting SSIM over vanilla pixel-wise errors is the incorporation of the previously mentioned luminance, contrast and structure over many patches. It has been shown in previous work that these induce a perceptual loss such that autoencoders can be applied to more complex real-world scenarios [21]. This can only hardly be achieved by pixel-wise errors, since each pixel is treated (spatially) independently from all others. The SSIM loss can further be extended by adopting the MS-SSIM loss [227], which advances the SSIM to incorporate patches of multiple scales, but using the same metrics as SSIM. We mostly use the SSIM loss in this work, but there are more powerful perceptual losses as introduced in the next section.

2.4.3 Perceptual loss

As briefly mentioned in the previous section, pixel-wise errors are easy to define and fast to compute during training of deep learning models. However, pixel-wise errors have the disadvantage that a lot of contextual information and semantics are being lost since each pixel is being treated independently. While the SSIM loss solves some of these disadvantages, there is still room for improvement in the case that the reconstruction needs to look as realistic as possible.

In the age of deep learning, it has become common practice to train a high capacity model on a large corpus of images, e.g. ImageNet. Since resulting models have seen a large variety

of images and learned to extract more universal features, these models can be used to extract meaningful features on novel images for downstream tasks [259]. Further, it has been shown that these features can also be used to measure the distance between images, e.g. in case of style transfer [114] or for image generation [64].

Instead of encouraging the autoencoder model to match each single pixel between target and reconstruction, we want that the resulting reconstruction should have a similar feature representation in the space of previously pre-trained models. To formalize this, let $\mathcal{E}_{fix}$ be a pre-trained model, e.g. VGG-16 [225] pre-trained on ImageNet, whose parameters are being fixed. We refer to $\mathcal{E}_{fix}$ being a feature ***extractor***. For both, the target image x and the reconstruction $\hat{x}$ we use $\mathcal{E}_{fix}$ to extract features for L layers, i.e. after each activation function, but it could also be a single layer somewhere in the middle. Let $\sigma_l(x)$ be the output of the activation after the lth layer for the input x. In case of convolutional layers, the shape of the latter extracted features is denoted as $C_l \times H_l \times W_l$. It is common practise to normalize the output of the activation $\sigma_l(x)$ along the channel dimension. The loss in the feature space of layer l is then computed by means of the squared ℓ_2 norm, but other metrics can be used as well, e.g. ℓ_1 norm or cosine similarity. The difference in the feature space can further be weighted channel-wise by dedicated vector $\omega_l \in \mathcal{R}^{C_l}$ [276]. The perceptual loss (PC) [276] for layer l using the network $\mathcal{E}_{fix}$ can be formulated as

$$\mathrm{PC}_l(x,\hat{x};\mathcal{E}_{fix}) = \frac{1}{H_l W_l} \left\| \omega_l \odot (\sigma_l(x) - \sigma_l(\hat{x})) \right\|_2^2, \tag{2.17}$$

where $\odot$ is the element-wise multiplication. Finally, the perceptual loss is the sum of all layers

$$\mathrm{PC}(x,\hat{x};\mathcal{E}_{fix}) = \sum_{l=1}^{L} \mathrm{PC}_l(x,\hat{x};\mathcal{E}_{fix}) = \sum_{l=1}^{L} \frac{1}{H_l W_l} \left\| \omega_l \odot (\sigma_l(x) - \sigma_l(\hat{x})) \right\|_2^2. \tag{2.18}$$

In case of the LPIPS loss [276], this approach is further extended by learning linear weights ω on top of each channel.

While the reconstruction quality can be largely improved by using the perceptual loss based on pre-trained networks, this does not necessarily transfer to a more beneficial latent space representation for down-stream tasks: we will show this in our quantitative investigations in Section 2.6.5. Further, using the perceptual loss slows down the training significantly: in our case training took three times longer.

2.4.4 Qualitative comparison

We conclude this overview of reconstruction losses by training different convolutional autoencoder models using the different aforementioned reconstruction losses during training. The models were trained for 1000 epochs using a 64-dimensional latent space, the AdamW

[152] optimizer with a learning rate of 0.001 and a weight decay of 0.3. For computing the perceptual loss we used the library provided by Zhang et al. [276] and for computing the SSIM the library developed by Gongfan [84]. We trained the model on GTSRB, SVIRO-Uncertainty and ORSS and evaluated the resulting models on the test images after training. The resulting reconstructions are reported in Fig. 2.4 and Fig. 2.5.

This first small experiment highlights one of the differences between SVIRO and other basic academic datasets. While GTSRB is visually already more complex than MNIST or similar datasets, SVIRO makes the task even more difficult. One of the reasons is the higher variability caused by human poses, the presence of child and infant seats and the smaller dataset size with unbalanced classes. It can also be observed that the performance on SVIRO and ORSS for different model design choices are similar: SSIM performs better than MSE and ℓ_1 and the perceptual loss produces the most clear reconstructions.

Another question arises in this context: what does generalization mean for generative, and in particular autoencoder models? Learning the identity function will not be a desirable result for all tasks since it is questionable to assume that the extracted features might be meaningful in that case. We can observe that the perceptual loss preserves the classes, but sometimes replaces adults with different (training) adults of similar poses. We argue that the latter is the most desirable scenario, since the test scene is being represented by a similar training scene. This means that the latent space representation must be similar as well such that the model learned a meaningful and beneficial feature extraction.

2.5 Regularized autoencoders

As mentioned in the previous section, the reconstruction loss, for which there are several choices, is the most important criterion during the autoencoder training. Nevertheless, a benefit of autoencoder models is the possibility of putting additional regularizations in the input space, e.g. denoising autoencoder as in Section 2.5.1, and on the latent space, e.g. Gaussian prior in case of variational autoencoders (VAE) as in Section 2.5.4. These additional regularizations should help to learn more robust and universal features and potentially invariances with respect to undesirable factors like noise. The different regularizations presented in this section can either be used standalone or combined together. Moreover, they can be used in combination with any of the aforementioned reconstruction losses.

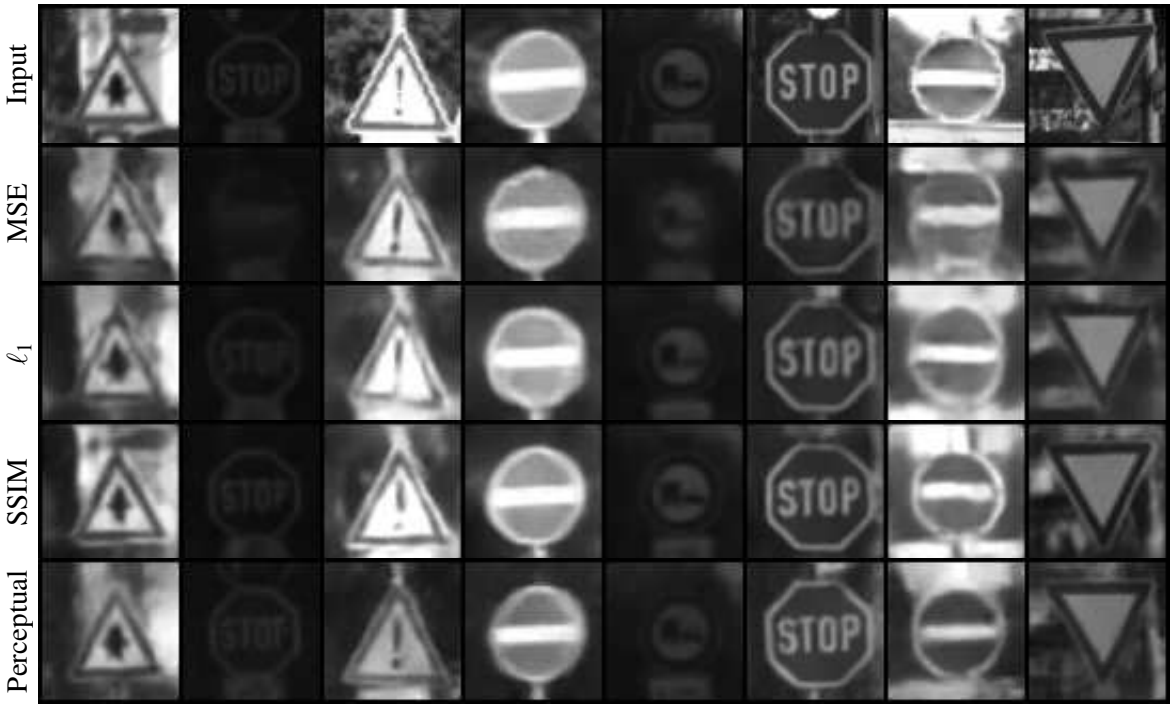


Fig. 2.4 Reconstruction results of GTSRB test images (first row) for autoencoder models being trained with different reconstruction errors (following rows).

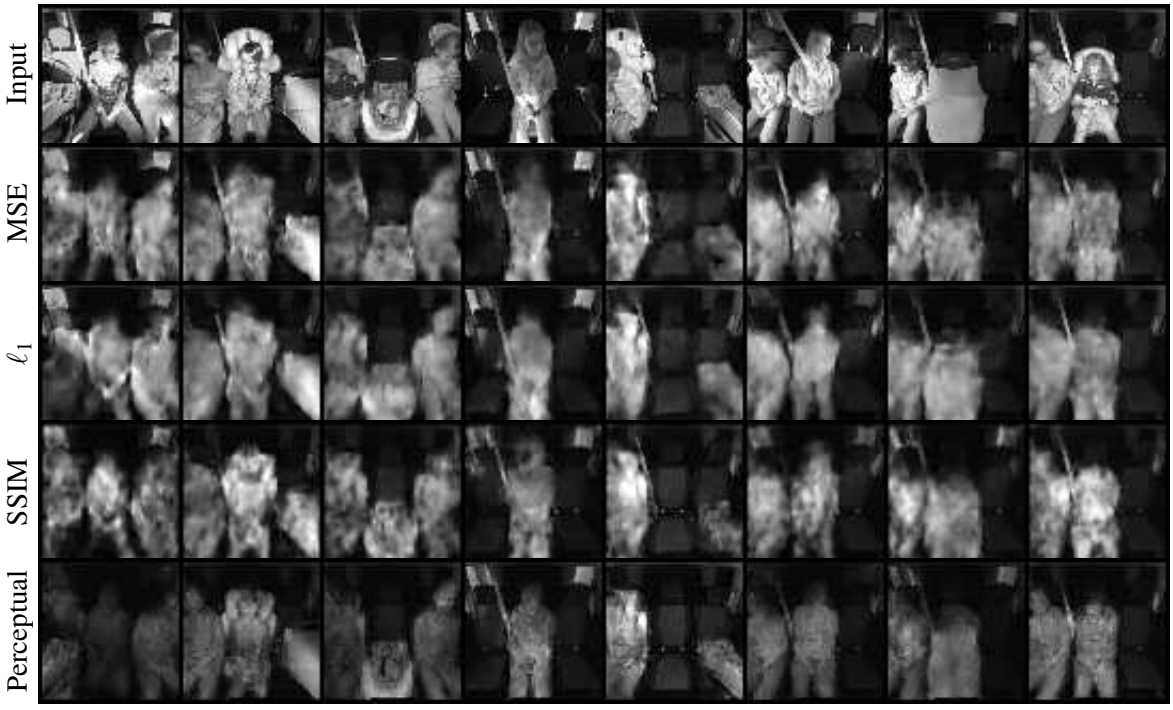
2.5.1 Denoising

One of the simpler autoencoder regularization methods is used by the denoising autoencoder (DAE) [252, 82, 266]. In this training setting the autoencoder model receives as input a corrupted version x_ε of x, where the corrupted version can be as simple as adding random Gaussian noise to it, i.e. $x_\varepsilon = x + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \Sigma)$. Instead of reconstructing the new, corrupted input image x_ε the reconstruction loss is computed against the clean input image x. This can be formulated as

$$\mathcal{L}_R(x, f(x_\varepsilon); \theta, \phi) = \mathcal{L}_R(x, f(x+\varepsilon); \theta, \phi). \tag{2.19}$$

Consequently, the denoising autoencoder needs to learn how to clean the corrupted input image instead of reconstructing the input identically. This leads to different, but more robust feature extractions, particularly with respect to noise [266]. Notice that in this case the target image and autoencoder input image are no longer the same image. This distinction is important, especially because of the methods presented later in this work.

(a) SVIRO



(b) ORSS

Fig. 2.5 Reconstruction results of SVIRO (a) and ORSS (b) test images (first rows) for autoencoder models being trained with different reconstruction errors (following rows).

2.5.2 Inpainting

Closely related to the denoising autoencoder is the training method based on inpainting. Similarly as before, the training input image x is being corrupted, but instead of noise, larger image areas (e.g. rectangles) are being removed or text can be placed on top of the image [266]. The autoencoder model should learn how to fill in the missing pixel values such that the image looks semantically correct after the reconstruction. Similarly as for the denoising autoencoder, this training strategy can lead to better generalization properties.

2.5.3 Metric losses and the triplet loss

Metric learning considers the task of learning a metric, i.e. a distance function, with respect to a dedicated task and dataset. It can, for example, be helpful that the distances between points in the latent space are separated according to the Euclidean distance such that the nearest neighbour search is more meaningful. A metric in the latent space can be enforced by dedicated losses measuring well-defined distances between samples, where the samples are chosen according to a strategy. Some of the latent space losses which can be used to induce a metric are the angular loss [256], contrastive loss [255] and n-pair loss [228]. We will compare our method against one of the most used sampling strategy: the triplet loss [15].

While the denoising autoencoder method induces a regularization in the input pixel space, the triplet loss induces a regularization in the latent space of autoencoder models. The former can only implicitly influence the latent space representation of the training data, but the latter explicitly forces certain latent space properties. In case the triplet loss is used during training, the input x to the autoencoder model is being referred to as anchor and denoted by $x_a := x$. For each encoded input sample $\mathrm{e}_\phi(x_a)$, two additional inputs are being sampled: the positive sample x_p being a different image of the same class and a negative sample x_n being of a different class. The feature used to discriminate between the positive sample x_p and the negative sample x_n is the class label, since in this work we are exclusively interested in classification tasks. However, the triplet loss can also be adapted to work with continuous variables, e.g. in case of geometric distortions [43].

The goal of the triplet loss is to enforce a similar latent space representation for samples of the same class by minimizing the distance between their latent space representations. Further, samples of different classes should be pushed away in the latent space by maximizing the distance between their latent space representations. The metric to measure the distance in the latent space can be chosen freely, but we restrict ourselves to the squared ℓ_2 norm in this work.

The triplet loss can then be formulated by

$$\mathcal{L}_T(x_a, x_p, x_n; \theta) = \max\left(0, \alpha + \left\|\mathrm{e}_\phi(x_a) - \mathrm{e}_\phi(x_p)\right\|_2^2 - \left\|\mathrm{e}_\phi(x_a) - \mathrm{e}_\phi(x_n)\right\|_2^2\right), \tag{2.20}$$

where α is the margin between positive and negative pairs. We chose $\alpha = 0.2$ in this work. It has been shown that the choice of the distance metric can have a large influence on the data manifold learned. For example, the use of the ℓ_2 norm to measure the distance in the latent space leads to a flatter, quasi Euclidean latent space [43, 172, 9].

There are several sampling strategies on how to select the positive and negative samples, e.g. hard mining [279]. We select the positive and negative samples from the currently used batch, either randomly or by considering them all. Further, we adopt the swap strategy [15] where the positive-negative distance can be used instead of the anchor-negative distance, in case that it violates the margin more. In all cases, the total loss to be optimized during training becomes

$$\mathcal{L}(x_a, x_p, x_n; \theta, \phi) = \mathcal{L}_R(x, f(x); \theta, \phi) + \mathcal{L}_T(x_a, x_p, x_n; \theta). \tag{2.21}$$

2.5.4 Variational autoencoder

The fundamental difference between the different autoencoder models presented thus far and variational autoencoders (VAE) is that the latter introduces a probabilistic view on the latent space representation and also the autoencoder process. For a complete introduction to variational autoencoders we refer to Kingma and Welling [128], since it is out of scope for this thesis to derive all properties in detail.

The true probability distribution of the training dataset is unknown. In the setting of variational autoencoder, this training distribution is approximated using a parametrized distribution p_θ with parameters θ. It is assumed that the training data is generated from an unknown latent variable, i.e. the latent vector z, such that the generative process can be formulated by a conditional distribution $p_\theta(x|z)$ representing the decoder module. The latent variables follow the prior distribution $p_\theta(z)$, which is in this work considered to be a centred isotropic multivariate Gaussian, as commonly used by the research community. The prior is hence equal to $p_\theta(z) = \mathcal{N}(0, \mathrm{I})$. Further, the likelihood of a training sample x can be formulated as

$$p_\theta(x) = \int_z p_\theta(x, z)\, \mathrm{d}z = \int_z p_\theta(x|z) p_\theta(z)\, \mathrm{d}z, \tag{2.22}$$

where $p_\theta(x, z)$ is the joint distribution between x and z. The decoder $p_\theta(x|z)$ is intractable (or very expensive) to be computed for every possible z and hence approximated by a neural

network. Using Bayes' theorem, we can also re-formulate $p_\theta(x)$ by

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}, \tag{2.23}$$

where $p_\theta(x)$ is usually intractable, i.e. impossible to compute. However, $p_\theta(z|x)$ can be approximated by the encoder module $q_\phi(z|x)$ of the autoencoder. The probabilistic encoder $q_\phi(z|x)$ is thus approximating the true posterior $p_\theta(z|x)$. The above Eq. (2.23) can be re-arranged to

$$p_\theta(x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(z|x)}, \tag{2.24}$$

out of which the data log-likelihood can be expressed[2]

$$\log p_\theta(x) = \log\left[\mathbb{E}_{z\sim q_\phi(z|x)}\left(\frac{p_\theta(x|z)p_\theta(z)}{p_\theta(z|x)}\right)\right] \tag{2.25}$$

$$\geq \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log\left(p_\theta(x|z)\right)\right] - \mathrm{KL}\left(q_\phi(z|x)\,\|\,p_\theta(z)\right), \tag{2.26}$$

where KL is the Kullback-Leibler divergence, the first term $\mathbb{E}_z\left[\log\left(p_\theta(x|z)\right)\right]$ is the reconstruction error $\mathcal{L}_R$ and the second term $\mathrm{KL}\left(q_\phi(z|x)\,\|\,p_\theta(z)\right)$ the regularization $\mathcal{L}_V$ in the latent space to make the latent space distribution close to the prior distribution. The above inequality is called the evidence lower bound (ELBO). The goal is to maximize the log-likelihood, i.e. the left-hand side of the inequality, which is equivalent to maximizing the right-hand side of the inequality. It is also common practice to minimize the the negative log-likelihood instead. The regularization in the latent space to enforce the encoding to follow a Gaussian prior is

$$\mathcal{L}_V(x;\theta,\phi) = \mathrm{KL}\left(q_\phi(z|x)\,\|\,p_\theta(z)\right). \tag{2.27}$$

For training the model, the above loss function is usually approximated using the reparameterization trick [128]. This makes it easier to calculate and optimize the regularizing loss in the latent space, which can now be approximated by

$$\mathcal{L}_V(x;\theta,\phi) \approx -\frac{1}{2}\sum_{j=1}^{d_l}\left(1+\log(\sigma_j^2)-\sigma_j^2-\mu_j^2\right), \tag{2.28}$$

where $z = \mu + \sigma\odot\varepsilon$ with $\varepsilon\sim\mathcal{N}(0,\mathrm{I})$ and all being of dimension d_l. The variables μ and σ are the resulting encodings by e_ϕ, which now needs to generate two values of size d_l instead of a single latent space vector z of size d_l. This is usually achieved by using twice

[2] We are skipping some steps here, for all details see Kingma and Welling [128]

the latent dimension and setting $\mu = [e_\phi(x)_1, ..., e_\phi(x)_{d_l/2}]$ and $\sigma = [e_\phi(x)_{d_l/2+1}, ..., e_\phi(x)_{d_l}]$. While usually the decoding is assumed to follow a Gaussian or Bernoulli distribution, we can simply adopt any of the aforementioned reconstruction losses, such that the total loss becomes

$$\mathcal{L}(x;\theta,\phi) = \mathcal{L}_R(x, f(x);\theta,\phi) + \mathcal{L}_V(x;\theta,\phi). \tag{2.29}$$

It is important to note that one needs to balance reconstruction loss and KL loss accordingly to avoid favoring one part of the equation to be minimized. This balancing term depends on the reconstruction loss used.

2.5.5 Disentanglement and scene decomposition

One possible desirable property for the latent space representation is the property that the different dimensions are independent, i.e. each dimension of the latent space is responsible for a different factor of variation (e.g. changing colors, the shape or the size of the object) with the hope of inducing interpretability for the different factors of variations [163]. Recent advances have shown that disentanglement in the latent space of autoencoders can lead to improved performance on visual downstream tasks [248]. Further, it is believed that meaningful scene decomposition [31, 65] will improve the transferability for many tasks, however, these methods still do not work well for scenes of higher visual complexity and are currently limited to toy datasets like CLEVR [115], Objects Room [31] or MPI3D. We will provide baseline results for the transferability between vehicle interiors by our proposed autoencoder approaches. Hence, future work can analyze the effect of disentanglement and scene decomposition with respect to the transferability on a task with higher visual complexity than commonly used datasets.

2.5.6 β-VAE

Closely related to the variational autoencoder is the β-VAE model. The interpretation and implementations are the same, except that an additional scaling factor is added to weight the contribution of the Kullback-Leibler divergence

$$\mathcal{L}(x;\theta,\phi) = \mathcal{L}_R(x, f(x);\theta,\phi) + \beta\mathcal{L}_V(x;\theta,\phi). \tag{2.30}$$

It has been shown that larger values for $\beta > 1$ favour disentangling factors of variation in the latent space [99].

2.5.7 FactorVAE

FactorVAE [126] tries to improve upon β-VAE by providing higher disentangling scores while maintaining the same reconstruction quality. The authors reformulated the expectation of the KL term in Eq. (2.27) by[3]

$$\mathbb{E}_z\left[\mathrm{KL}\left(q_\phi(z|x)\,\|\,p_\theta(z)\right)\right] = I(x;z) + \mathrm{KL}\left(q_\phi(z)\,\|\,p_\theta(z)\right), \tag{2.31}$$

where $I(x;z)$ is the mutual information between x and z. Optimizing for $\mathrm{KL}\left(q_\phi(z)\,\|\,p_\theta(z)\right)$ causes $q_\phi(z)$ to tend towards $p_\theta(z)$. The latter induces the dimensions of z to become independent, which under some metrics can be considered as analogous to disentanglement. However, optimizing for $I(x;z)$ reduces the information about x stored in z, hence causing worse reconstruction results. This means that higher values for β in case of β-VAE cause better disentanglement, but worse reconstructions. Instead, according to the authors Kim and Mnih [126], it is better to encourage independence directly by optimizing

$$\mathbb{E}_z\left[\log\left(p_\theta(x|z)\right)\right] - \mathrm{KL}\left(q_\phi(z|x)\,\|\,p_\theta(z)\right) - \gamma\,\mathrm{KL}\left(q_\phi(z)\,\|\,\prod_{i=1}^{d_l} q_\phi(z_i)\right), \tag{2.32}$$

where $\mathrm{TC}(z) = \mathrm{KL}\left(q_\phi(z)\,\|\,\prod_{i=1}^{d_l} q_\phi(z_i)\right)$ is intractable and called the total correlation. The authors propose to approximate the latter by training a discriminator D to estimate whether a sample comes from $q_\phi(z)$ or $\prod_{i=1}^{d_l} q_\phi(z_i)$ allowing to reformulate the previous equation as

$$\mathrm{TC}(z) = \mathrm{KL}\left(q_\phi(z)\,\|\,\prod_{i=1}^{d_l} q_\phi(z_i)\right) = \mathbb{E}_z\left[\log\frac{q_\phi(z)}{\prod_{i=1}^{d_l} q_\phi(z_i)}\right] \approx \mathbb{E}_z\left[\log\frac{D(z)}{1-D(z)}\right]. \tag{2.33}$$

Hence, the final objective and total loss to be optimized is

$$\mathcal{L}(x;\theta,\phi) = \mathcal{L}_R(x, f(x);\theta,\phi) + \mathcal{L}_V(x;\theta,\phi) - \mathbb{E}_z\left[\log\frac{D(z)}{1-D(z)}\right]. \tag{2.34}$$

Since we skipped a lot of steps, we refer to Kim and Mnih [126] for all derivations and details.

2.5.8 Qualitative comparison

Similar to the section about the different reconstruction losses, we conclude this section with a quantitative comparison of the presented regularization methods. The hyperparameters for the experiments are the same as mentioned in Sec. 2.4.4. We fixed the reconstruction loss to MSE

[3] We are skipping some steps here, for all details see Kim and Mnih [126]

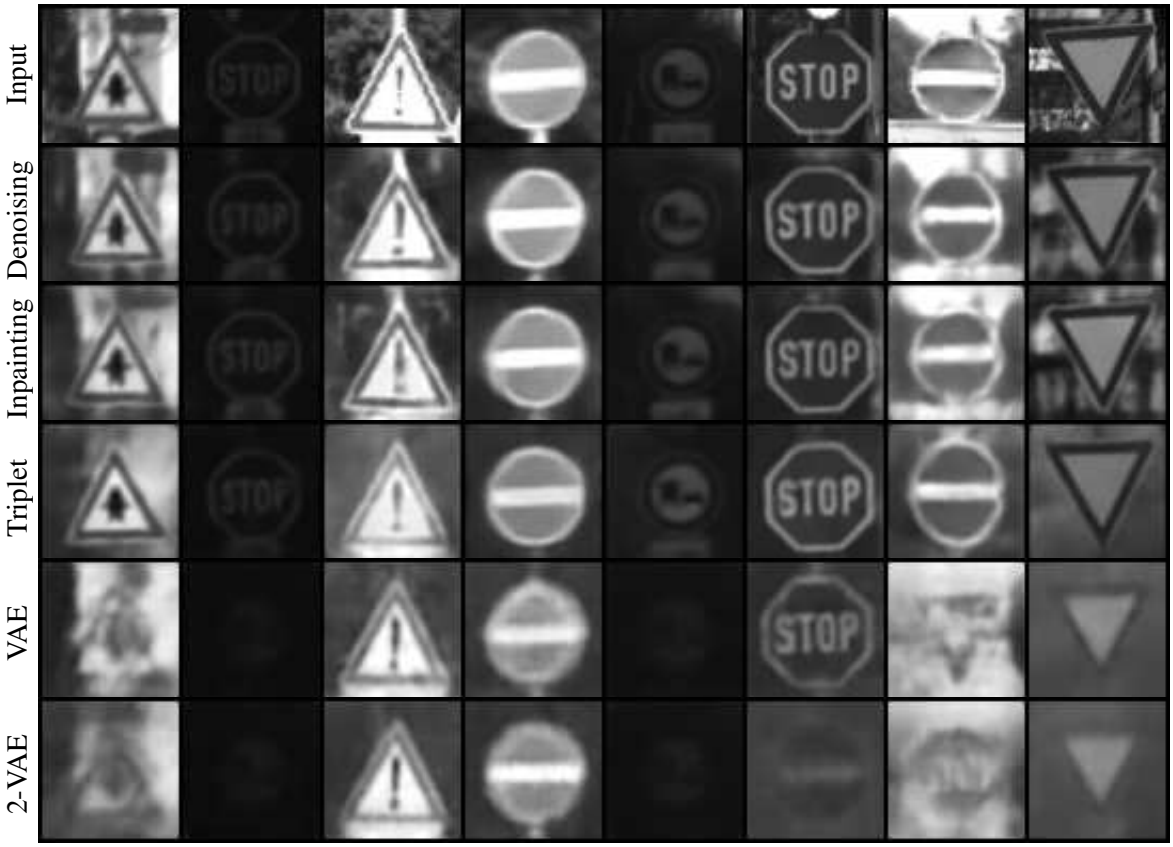


Fig. 2.6 Reconstruction results of GTSRB test images (first row) for autoencoder models being trained with different regularization methods (following rows).

for VAE based methods and used SSIM for all other methods. We use the package provided by Musgrave et al. [175] for computing the triplet loss. The resulting test image reconstructions are reported in Fig. 2.6, Fig. 2.7 and Fig. 2.8. It can be observed that the Gaussian prior has a detrimental effect on generalization capacities, at least from a purely reconstructive and semantic point of view. The latter issue has been addressed on visually simple datasets by Burgess et al. [30] though. The preservation of the semantics in the reconstructions is not being negatively influenced by the other regularizations. The triplet loss induces a smoother reconstruction, but the people become blurry. The cause for this is that the triplet loss forces images of the same class to be similarly represented in the latent space - independent of their pose. Hence, people need to be approximated and details might be reduced. It can also happen that the poses will be approximated by a generic pose, e.g. see results on SVIRO in Fig. 2.7.

Fig. 2.7 Reconstruction results of SVIRO test images (first row) for autoencoder models being trained with different regularization methods (following rows).

Fig. 2.8 Reconstruction results of ORSS test images (first row) for autoencoder models being trained with different regularization methods (following rows).

2.6 Latent space visualization

The latent space is an important part of any autoencoder model. It is usually the bottleneck part of the model with the smallest dimension across the network. Its goal should be to provide a meaningful feature reduction of the input. Hence, it is important to investigate the latent space after training. It is possible to verify the latent space for some desirable properties like meaningful interpolation [184] or disentanglement [36]. The importance of the properties depend on the dataset and the task to be solved. Visualizing the latent space of the training, test and potentially out-of-distribution data can be informative in many cases. Although the latent space is of a smaller dimension, it is usually still of dimension 8, 16, 64 or higher, depending on the task to solve. Visualizing the latent space can hence not be achieved trivially, because of the curse of dimensionality and the fact that distances behave counterintuitively in these dimensions [251]. There are three commonly used methods to transform the data into a lower dimensional representation. We will mention them shortly, but refer to other materials for detailed explanations.

2.6.1 Principal component analysis (PCA)

The principal component analysis (PCA) [260] is a linear transformation inducing a change of basis. The PCA computes the optimal basis in an iterative process, where optimal means that the first dimension (first principal component) should account for most of the data variance and the kth dimension (kth principal component) should account for the kth most variance of the data. Since the PCA is a change of basis, the different new directions need to be orthogonal to each other. To perform a dimensionality reduction, one can then select the k first principal components, where $k = 2$ is usually used for visualization purposes.

2.6.2 t-distributed stochastic neighbour embedding (t-SNE)

When using t-distributed stochastic neighbour embedding (t-SNE) [246] it is important to read the plots correctly, or to avoid concluding the wrong properties of the results: The size of the clusters and the distance between the clusters is usually *meaningless*. That being said, the belongship to a cluster means that the points are close to each other and are a good indication that the model represented them as the same class. The resulting plot depends a lot on the hyperparameter selection and the results are non-deterministic. The transformation is non-linear and the 2-dimensional projection yields usually a good representation of the high dimensional distribution, because t-SNE tries to preserve the local structure of the data.

2.6.3 Uniform manifold approximation and projection (UMAP)

Similarly to t-SNE, when using uniform manifold approximation and projection (UMAP) [166] we need to take care to not be lead to the wrong conclusions. The pitfalls are the same as for t-SNE. The reason for this is that both methods are in many aspects similar. The main difference between UMAP and t-SNE is the interpretation of the distances: t-SNE uses a Gaussian kernel and UMAP a fuzzy graph. This allows UMAP to adopt some tricks to speed up the computation. Also, UMAP seems to separate the different global clusters better one from another than t-SNE, such that it could be claimed that the global structure is more *meaningful* for UMAP.

2.6.4 Remarks

Since each of the aforementioned projection methods has its advantages and disadvantages, as well as their pitfalls, it can be informative to plot the projected latent space using all three methods to form a conclusion.

An important personal note from many performed experiments is the following: while it can be tempting to use a latent space of dimension 2 to ease visualization, the latter is usually a bad choice. The 2-dimensional latent space seems to be a special case and it often leads to detrimental performances. On many occasions, strange artefacts appeared which did not hold for larger dimensions. For example, we experienced the appearance of *tunnels* between the clusters of different classes which did not appear for higher dimensions. Its properties could potentially be interesting and important for some tasks, but for the general use case we would recommend to avoid using it. Many tasks have more than two degrees of freedom, or independent generating factors, present at the same time. Thus a two dimensional latent-space can be too restrictive. Finally, the higher the dimensionality, the easier the data is linearly separable, a property used, for example, by the kernel trick [44].

2.6.5 Experimental results

In this section we want to visualize the latent space representation of the experiments from Section 2.4.4 and Section 2.5.8. This should give a feeling for the different visualization methods, but also how the different reconstruction losses and regularization methods affect the latent space. We use the following Python libraries for the different visualization methods: scikit-learn [191] for PCA, umap-learn [167] for UMAP and opentsne [199] for t-SNE. We use the default hyperparameters for all three projection methods.

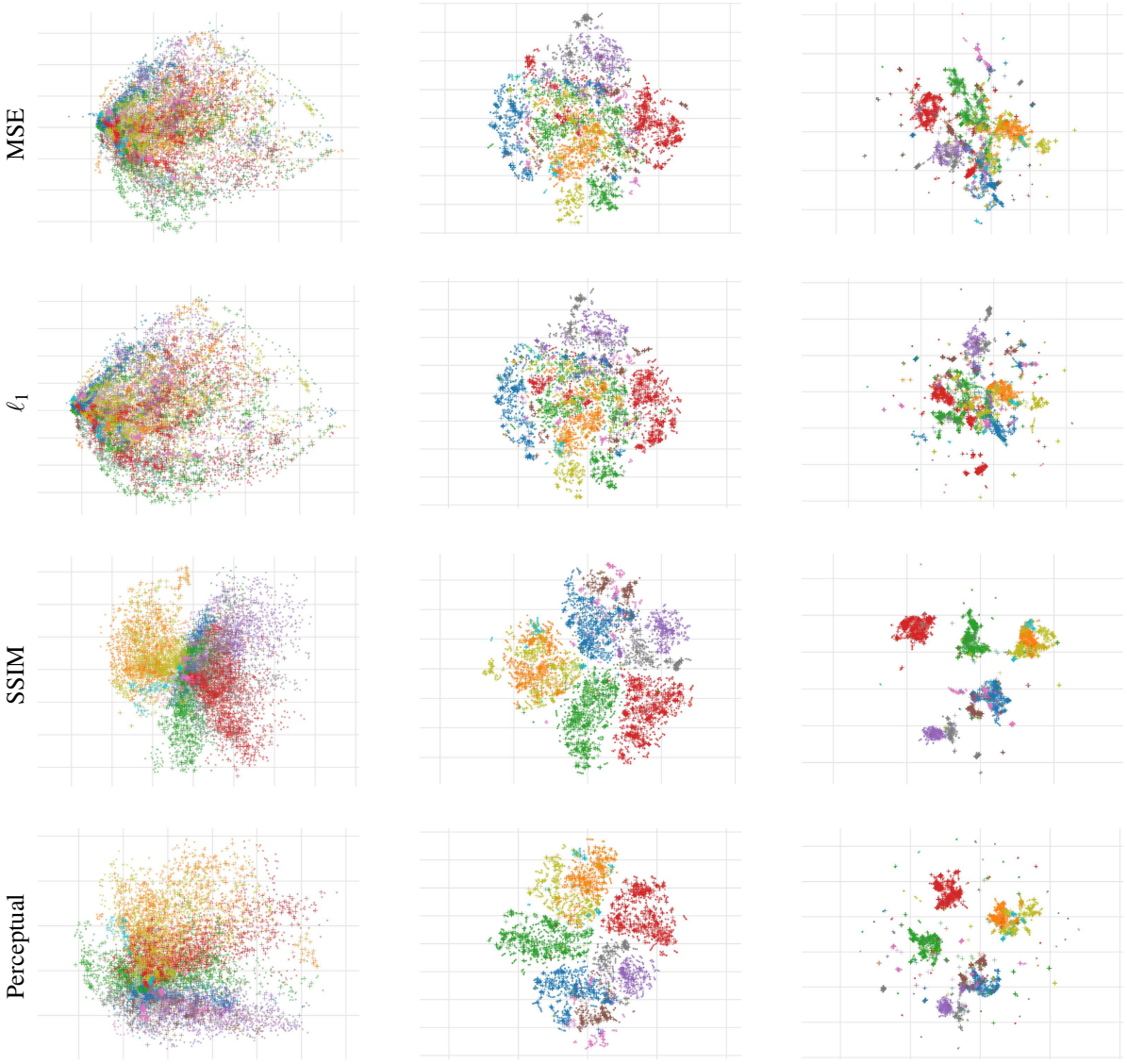
We limit the results of this section to the GTSRB dataset, because we use the reduced version of the dataset containing 10 classes only. SVIRO and ORSS have 64 classes, which makes the visualization unnecessarily complex for this basic analysis.

We report results for the models from Section 2.4.4 in Fig. 2.9. It can be observed that the best clustering across all three projection methods are achieved when SSIM and the perceptual loss are used during training. It is interesting to see that the results for SSIM are on par with the perceptual loss, even though the perceptual loss produces better reconstruction results and uses a much more powerful evaluation metric, as reported in Section 2.4.4. Since also training takes three times longer in the latter case, these arguments were in favour of focusing most of the investigations in the thesis on using the SSIM reconstruction loss.

The results for the models from Section 2.5.8 are reported in Fig. 2.10. Here it becomes obvious that the triplet loss largely influences the latent space representation due to its explicit regularization in the latent space: the training and test samples are perfectly separated. This is expected, at least for the training data, since the labels are used to define positive and negative samples. A linear classifier in the latent space would hence perform better than for the other latent spaces. Denoising and inpainting make the latent space a bit clearer while variational autoencoder variations have a detrimental effect.

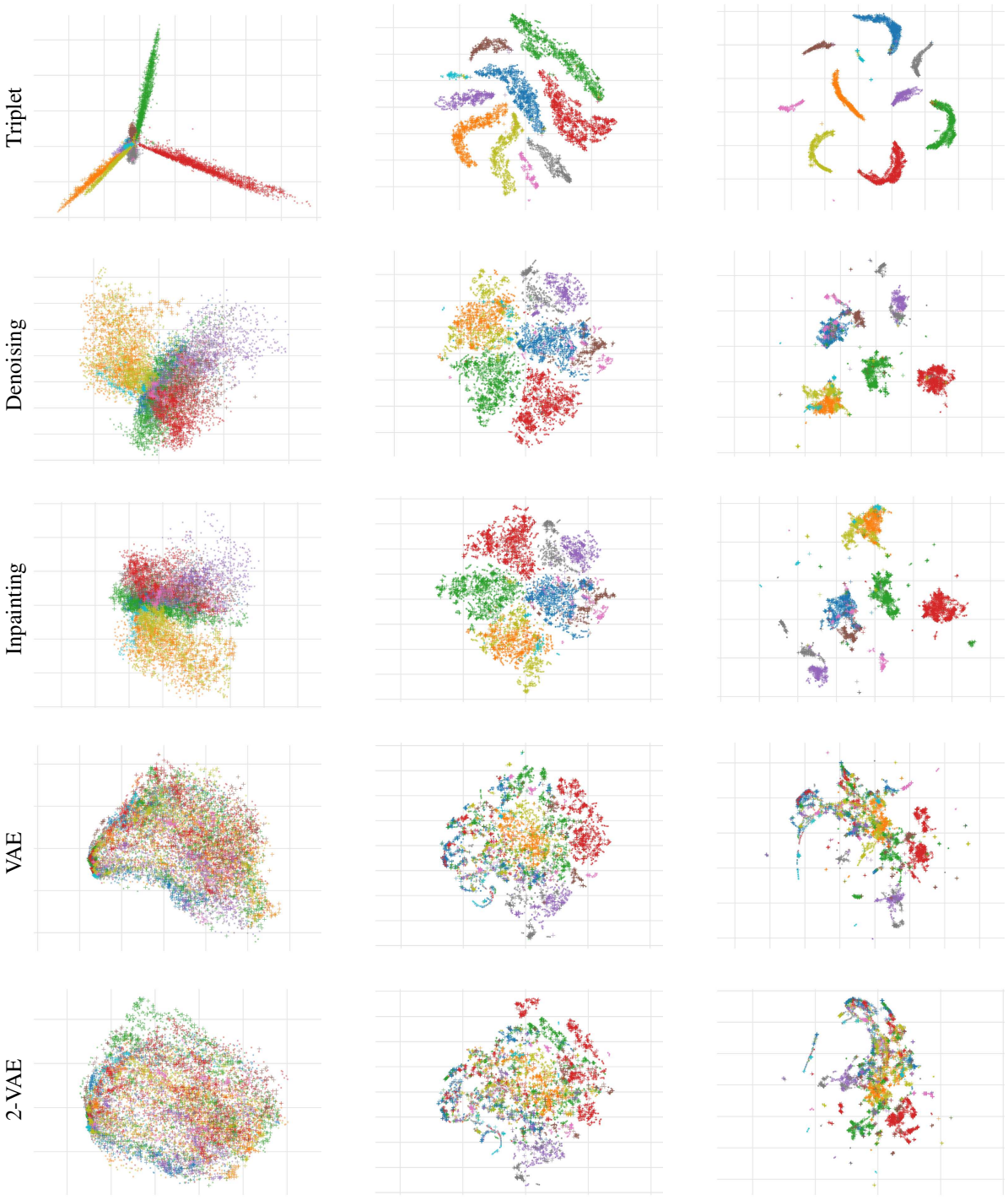
Overall, it can be concluded that the reconstruction losses and the regularization methods, implicitly or explicitly, influence the latent space structure. These results should motivate the investigations presented later in this work, where we try to induce more beneficial latent space representations and invariances for several use cases.

To assess the above qualitative conclusions quantitatively, we trained a linear SVM in the high dimensional latent space representation and test the classifier performance on the test data. A linear classifier should achieve a better performance on a high dimensional data representation if the data is better clustered. We report the results on GTSRB in Table 2.2: Similar to the latent space visualization, it can be observed that the perceptual loss induces the best feature separation in the latent space, followed by SSIM and then by ℓ_1. As expected, due to the explicit use of labels by the triplet loss, the latter performs best. Using denoising or inpainting during training also improves more universal feature extractions. These conclusions are similar to the observations we made previously on the visualizations of latent space projections. Consequently, visualizing the latent space and evaluating classification accuracies after training the autoencoder model go hand in hand and provide a good assessment about the quality of the extracted features.



(a) PCA (b) t-SNE (c) UMAP

Fig. 2.9 Comparison of different projection methods (columns - PCA, t-SNE and UMAP) on the latent spaces of autoencoder models trained with different reconstruction losses (rows - MSE, ℓ_1, SSIM and perceptual loss). The different colors represent the different classes. Circles are training samples and crosses are test samples from the GTSRB dataset. SSIM and the perceptual loss provide the best clustering across all three projection methods.



(a) PCA (b) t-SNE (c) UMAP

Fig. 2.10 Comparison of different projection methods (columns - PCA, t-SNE and UMAP) on the latent spaces of autoencoder models trained with different regularization methods (rows). The different colors represent the different classes. Circles are training samples and crosses are test samples from the GTSRB dataset.

Table 2.2 Comparison of linear SVM classifier performances (in percentage) on the high dimensional latent space representation. A higher accuracy indicates a better clustered and separated representation.

Reconstruction loss	Regularization method	Accuracy
MSE	-	84.82
ℓ_1	-	89.78
SSIM	-	94.83
Perceptual	-	96.95
SSIM	Triplet loss	98.50
SSIM	Denoising	95.17
SSIM	Inpainting	97.48
SSIM	VAE	78.77
SSIM	2-VAE	71.23

2.7 Other generative models

This work focuses on autoencoder models, variations and novel contributions thereof. Nevertheless, we could not present a work on autoencoder models without at least referring to other methods. While vanilla autoencoder models can be considered as compression models, variational autoencoders are considered to be generative models. That being said, we will limit our comparison to other generative approaches. Other approaches worth mentioning, but not considered in this thesis, are models based on compressed sensing [262], wavelets [118] or recurrent neural networks [239].

2.7.1 Autoregressive models

Autoregressive models are explicit and tractable density models (in contrast to VAE, which are not tractable) which try to predict the value of a next element based on the values of all previous elements. Applications on image data, e.g. PixelCNN++ [217], try to model the likelihood of a pixel value given the pixel values of all previous ones. This is a conditional process which means that for an image x the model can be formulated as

$$p(x) = p(x_1,...,x_n) = p(x_1)\prod_{i=2}^{n} p(x_i \,|\, x_1,...,x_{i-1}). \tag{2.35}$$

A disadvantage of this approach is that the pixels need to be generated sequentially, since each pixel needs the predicted pixel values of all the ones before. This causes the inference to be slow, especially for larger images. Usually, a starting point and a sequence of pixels (i.e. an

ordering) needs to be defined, e.g. start at the top left pixel and complete the image row by row from left to right. Another inconvenience for our investigations is that there exists no latent space for these models.

2.7.2 Generative adversarial networks

Probably the most prominent generative model architectures are the generative adversarial networks (GAN) [85]. Compared to the other methods, GANs are implicit density estimators. GANs consist of two networks: 1) the generator network G_ϕ is usually randomly initialized by sampling from a predefined simple distribution, e.g. multivariate Gaussian distribution, and generating then an image. The images are hence not generated from a specific distribution, i.e. with less control, unlike the VAE. 2) The discriminator network D_θ, which receives real images from the training distribution and images generated by the generator network as input. In both cases ϕ and θ refer to the network parameters respectively. The discriminator tries to classify whether the input is a real or generated image. The generator tries to fool the discriminator by learning to generate synthetic images, which should progressively look more and more like images from the training dataset. This training process is called a minimax loss and formulated as

$$\min_{\phi} \max_{\theta} \left[\mathbb{E}_x \log D_\theta(x) + \mathbb{E}_z \log(1 - D_\theta(G_\phi(z))) \right], \tag{2.36}$$

where x is an image from the real training distribution and z is a sample from the simple distribution used to generate a fake image. This loss can be interpreted as having two players, the discriminator and the generator, which are competing against each other. Jointly training both networks can be challenging and results often in unstable optimization procedures. Another common problem of GANs is that they are susceptible for mode collapses, where the generator would generate only the same, or a few images. The optimum is achieved on the Nash equilibrium - a state where the discriminator can no longer distinguish between real and generated images. Evaluating the model after training is more difficult than for autoencoder models, since a successful training means that the generator can create images which would potentially come from the training distribution. However, the latter can only hardly be measured. One common approach is to compute Fréchet inception distance (FID) [98] between the training data and generated images. That being said, a lot of research is being put into stabilizing and improving the training of GANs. Notwithstanding these difficulties, GANs have proven to produce impressive results on image generation, style transfer [121], image-to-image translation [285] and also latent space retrieval [62]. GANs are particularly useful for multimedia, artistic and entertainment systems where sporadic bad examples are not critical. However, for safety critical and engineering applications the complicated training procedure can have detrimental

effects. On the other side, autoencoder models are, at least in our opinion, a more elegant solution with easier inference and the possibility to add meaningful regularization without too much overhead. Using autoencoder models, one can either classify the latent space representation or the reconstruction of the input image. This is not possible in vanilla GAN, since the models do not try to compress the input image, but instead try to learn the training distribution. It is hence easier to retrieve the lower dimensional latent space representation for the training and test data using autoencoder models. Without learning to transfer an image from a new vehicle interior to the training vehicle using style transfer, see Section 2.8.1 and 2.8.3, it is not clear how GANs should generalize to new vehicle interiors. This is of course an open question and interesting research direction which we decided not to follow in this thesis.

2.7.3 Normalizing flow models

Normalizing flow based generative models are also explicit density models which use a change-of-variable to transform a complex distribution to a much simpler one (or vice-versa) using invertible functions. In our case, the complex distribution is the one from the training data. We want to learn a bijective mapping from the complex distribution to a simple one on which sampling is much easier. Since the mapping is bijective, and also deterministic, we have that

$$x = f_\theta(z) \quad \text{and} \quad z = f_\theta^{-1}(x). \tag{2.37}$$

This is similar to autoencoder models, with two differences: 1) the encoding and decoding is bijective. 2) the latent space needs to have the same dimension as the input space. The latter condition on f is necessary in order to use the change of variables formula, because otherwise the determinant is not computable. This limitation is investigated by Kim et al. [125]. The simple distribution can be chosen freely, e.g. a Gaussian distribution, and is considered to be the prior distribution $p_z(z)$. Using Eq. (2.37) and the change of variables for probability density functions formula one can derive

$$p_x(x) = p_z(z)\left|\det \frac{\partial z}{\partial x}\right| = p_z(f_\theta^{-1}(x))\left|\det \frac{\partial f_\theta^{-1}(x)}{\partial x}\right|. \tag{2.38}$$

The normalizing flow model is trained by minimizing the negative log-likelihood,

$$-\log p_x(x) = -\log p_z(f_\theta^{-1}(x)) - \log\left|\det \frac{\partial f_\theta^{-1}(x)}{\partial x}\right|. \tag{2.39}$$

In practise, instead of having a single complex function f, one can obtain a more complex function by composing multiple (simple) invertible functions $f = f_1 \circ f_2 \circ ... \circ f_k$. The different f_i can then be the layers of a neural network and the negative log-likelihood is re-written as

$$-\log p_{x_0}(x_0) = -\log p_{x_k}(f_{\theta_k}^{-1}(x_{k-1})) - \sum_{i=1}^{k-1} \log \left| \det \frac{\partial f_{\theta_i}^{-1}(x_{i-1})}{\partial x_{i-1}} \right|, \tag{2.40}$$

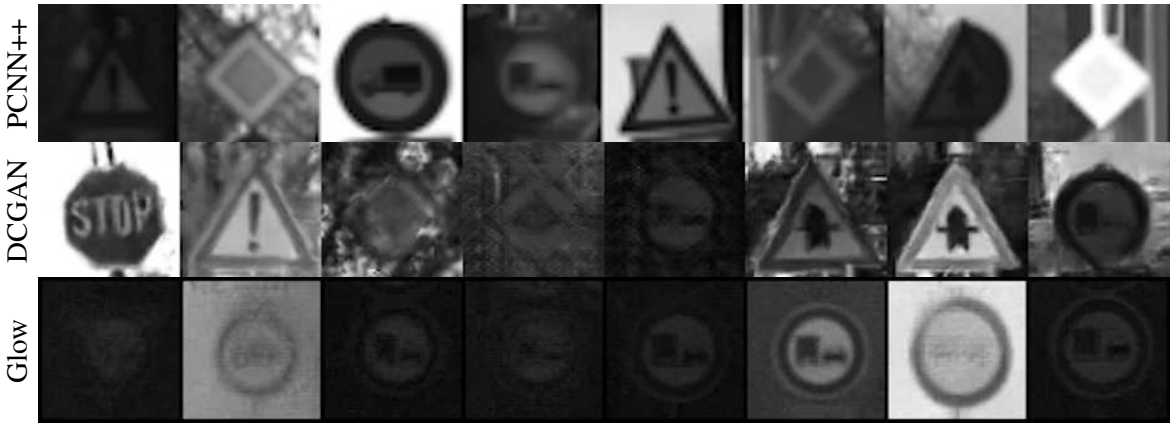
where $x_k := z$, $x_0 := x$, $x_i = f_{\theta_i}(x_{i-1})$ and $x_i = f_{\theta_i}^{-1}(x_{i+1})$. Normalizing flows converge easier than GANs and their training procedure is more stable. Since the latent spaces need to be of the same size as the input space, i.e. no dimensionality reduction is happening. This can, but does not need to, have detrimental or undesirable effects. Notice that f, or the f_i, should be easy to invert and we need to compute the determinant in order to train the model. An example of a normalizing flow based method for image data is the Glow [129] model.

2.7.4 Qualitative comparison

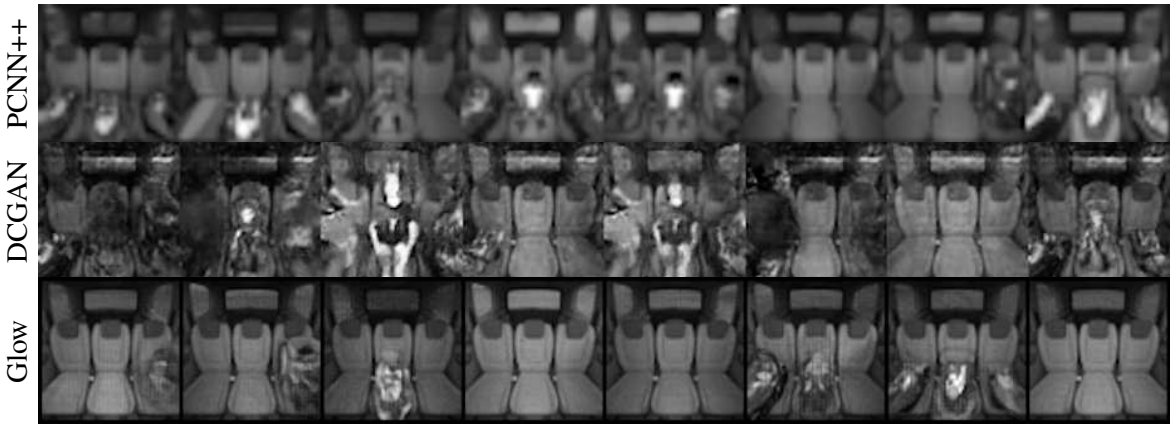
We report qualitative results for the image generation by the methods introduced in this section in Fig. 2.11. For autoregressive models we use the PixelCNN++, for GANs the DCGAN [202] and for normalizing flows the Glow approach. We use the same hyperparameters as for the autoencoder results presented earlier, i.e. we train for 1000 epochs and use a batch size of 64. Only the optimizers and learning rates were adapted to work better for the different methods, because, as mentioned earlier, convergence cannot always be achieved trivially.

When using PixelCNN++, we needed to reduce the dimension of the samples to be generated to 32 by 32 pixels, because otherwise it would not fit the memory of the GPU. Further, as mentioned before, generating samples takes a long time when using autoregressive models. While the GAN training went smooth, it can be observed that the models sometimes produce good results and sometimes noisy images, especially for the vehicle interior the blurry samples are more frequent. The Glow method seems to work well, but no latent space is available and the images contain less details. For all methods, we could only generate random images stemming from the training distribution, since their model design does not allow any straightforward and meaningful interaction with test samples[4]. Also, all of the models took significantly longer to train than the autoencoder models. All in all, the results are not better compared to using autoencoder models, while the latter has some additional advantage for the use-cases presented in this thesis.

[4]We could take the first pixel of the test images to condition the PixelCNN++, but this is uninteresting, because no semantically meaningful information of the test image is being considered.



(a) GTSRB



(b) SVIRO

(c) ORSS

Fig. 2.11 Samples generated by PixelCNN++ (PCNN++), DCGAN and Glow. Samples were generated randomly, since there is no way to condition the models equally.

2.8 Challenges

A few of the challenges investigated in this thesis consider the transfer in-between different vehicle interiors, from synthetic to real vehicle interiors, but also between different illumination conditions. In the following we give a few definitions and provide an overview of existing methods, and why they cannot be adopted to our use-case.

2.8.1 Domain shift

All of the aforementioned challenges can be considered as a shift in the domain between the training distribution and the test distribution. Existing autoencoder-based methods try to represent the information from multiple domains [8] or real-synthetic image-pairs [277] identically in the latent space by enforcing some similarity constraints, e.g. the latent vectors should be close together. However, these approaches often force networks to reconstruct some (or all) of the images correctly in the decoder part. Forcing an autoencoder to represent two images (e.g. same scenery, but different illumination) identically in the latent space, yet simultaneously forcing it to reconstruct both input images correctly implies an impossibility: The decoder cannot reconstruct two different images using the same latent space. Antelmi et al. [8] adopted a different autoencoder for each domain, but, for example, as illumination changes are continuous and not discrete, we cannot have a separate encoder or decoder for each possible illumination.

Methods from domain adaptation [188, 212, 122] are commonly used to reduce the gap between the target domain (e.g. the vehicle in which we want to use our model) and source domain (e.g. the vehicle we trained on). However, these methods usually require (often even labelled) images from the target distribution to work well. Zero-shot learning (ZSL) [263, 179], and particularly generalized zero-shot learning (GZSL) [34, 263, 71], are the most extreme cases of domain adaptation as they do not require labels for new test objects. Both setups consider the generalization to new classes, but require some additional type of information, e.g. word embeddings [179] or semantic descriptions [71]. However, we focus on the evaluation of *seen* class *instances* in *unknown* environments and *unknown* class *instances* in *known* and *unknown* environments. The main difference stems from the following constraint: the adaptation of trained models to new class instances and environments should be avoided. For example, models should be robust against new child seats appearing on the market after the model was deployed, and models should not need to be adapted for each vehicle interior variation.

Alternatively, it would be possible to transform images from an unknown vehicle back to the known vehicle, e.g. by aligning both domains [148] or by using style transfer techniques

[121, 242]. However, those techniques need images from the target distribution as well. Similar to eyeglass removal achieved by GANs [208], we could use a GAN to change the vehicle background, but this would need images from the target domain or image-pairs of what we would expect to encounter. Domain generalization considers methods to generalize to new domains without accessing images from the unknown domain during training [143, 284]. Nevertheless, these techniques use images from several domains during training in order to generalize well to unknown domains. Furthermore, the aforementioned methods often combine several datasets. To the best of our knowledge, SVIRO is the first dataset which allows to investigate the generalization on the same tasks to a new, but similar, vehicle interior. Hence, common domain shift problem formulations and proposed solutions could not consider the challenge of generalizing to an unknown domain when learning from a single domain for solving the same task.

2.8.2 Image and illumination normalization

Recent advances in portrait shadow manipulation [278] try to remove shadows cast by external objects and to soften shadows cast by the facial features of the subjects. While the proposed method can generalize to images taken in the wild, it has problems with detailed shadows and it assumes that shadows either belong to foreign or facial features. Most importantly, it assumes facial images as input and exploits the detection of facial landmarks and their symmetries to remove the shadows. Other shadow removal methods [257, 201] are limited to simpler images. The backgrounds and illumination are usually quite uniform and they contain a single connected shadow. Moreover, the availability of shadow and shadow-free image pairs provides the means of a well-defined ground truth. However, this is not possible for more complex scenes and illumination conditions for which a ground truth is not available or even impossible to define. Image relighting [231, 283] could potentially be used to change the illumination of an image to some uniform illumination. However, as noted in [231, 278] relighting struggles with foreign or harsh shadows. While it is possible to fit a face to a reference image [224], this option is limited to facial images as well. Lastly, another common approach to retrieve illumination information from an image stems from intrinsic image decomposition [75], which tries to decompose the image into reflectance and shading. Similarly as before, generating the necessary ground truth data cannot trivially be achieved and generalization needs to be enhanced. Also, it is common to adopt temporal information to learn how to retrieve illumination [145] or include much stronger priors using albedo and shading information during training [150]. However, the models tend to depend on the semantics and environmental conditions of the training data [75]. While our proposed method also suffers from the latter, we believe that future work might be able to benefit from a combination of both worlds.

2.8.3 Synthetic to real generalization

There have been successful applications of reinforcement learning systems being trained in a simulated environment and deployed to a real one, for example by combining real and synthetic data during training [119, 209, 68, 22]. However, these approaches can take into account temporal information and action-reaction causalities while in this work we use independent frames only. A good overview on reinforcement learning-based simulation to real transferability is provided by Zhao et al. [280]. Another line of research uses GANs to make synthetic images look like real images or vice versa [101, 33]. This requires both synthetic and real images, whereas we focus on training on synthetic images only. Part of methodologies presented in this work are related to domain randomization [241], where the environment is being randomized, but the authors deployed this to object detection and the resulting model needs to be fine-tuned on real data. A similar idea of freezing the layers of a pre-trained model was investigated for object detection [100], but neither with a dedicated sampling strategy nor in the context of autoencoders. Another work focuses on localization and training on synthetic images only [238], though the applicability is only tested on simple geometries. Recent advances on synthetic to real image segmentation [38, 271, 189] on the VisDA [192] dataset show a promising direction to overcome the gap between synthetic and real images. However, this cannot straightforwardly be compared against the investigation in this work, particularly, since we are focusing on autoencoder models and their generative nature. Others rely on the use of real images during training for the minimization of the synthetic to real gap for autoencoders [109, 277].

2.9 Uncertainty and out-of-distribution detection

In using any kind of classifier in their basic form to make a prediction, we force the latter to output one of the known training classes no matter the input image. This could be sufficient if you are working on a closed system in which you can guarantee that each input ever received by the classifier is similar to the training distribution. However, in open world problems, and in general in most engineering applications, particularly for the consumer market, it is impossible to account for all variations potentially occurring in the environment the model will be deployed to. This is particularly important in the case of safety critical applications. We cannot expect from a binary classifier trained on dogs and cats to raise an error when suddenly an elephant is received as input. Yet, the latter is exactly what we would want in practise. One of the open challenges of machine learning is to reliably assess the certainty, or uncertainty, of the model's prediction. This is commonly rephrased to *knowing that we don't know* [72].

Usually two tightly related tasks are being investigated: error in prediction and out-of-distribution (OOD) detection [29]. For the former the model should provide a high uncertainty in case it makes an erroneous prediction, i.e. the model should self-assess its reliability. In the latter the model should detect that an image is exotic, or from a completely different distribution than the training distribution.

When working with uncertainty there are two types of uncertainties to consider: ***aleatoric*** and ***epistemic*** uncertainty [124]. Aleatoric uncertainty describes uncertainty coming from the randomness and variability of the considered experiment. No matter how much data collected and which model is trained on it, uncertainty cannot be reduced up-to an inhered threshold: in case of a (fair) coin flipping there will always be a random outcome. This uncertainty can also be the result of noise in the observation, e.g. sensor noise. In general, aleatoric uncertainty cannot be reduced up-to a threshold no matter how much data will be collected. Epistemic uncertainty, on the other side, refers to uncertainty stemming from the model. This uncertainty can, in principle, be reduced by collecting more data and/or by improving the model and its training. This uncertainty is usually caused by lack of knowledge, i.e. the model should be able to know provided that enough and the correct data is collected. When we talk about uncertainty estimation in this work, we will always refer to epistemic uncertainty and we will try to improve the model's self assessment regarding the trustability of its predictions.

A lot of research [1] is focusing on estimating the uncertainty of a model's prediction regarding OOD or uncertainty estimation, both of which are tightly related. However, only a few works consider the use of autoencoder models for assessing uncertainty: Autoencoders can be combined with normalizing flow [25], refactor ideas from compressed sensing [89] or use properties of Variational Autoencoders [207, 265]. More commonly, autoencoders are used for non-image-based datasets [249, 267, 182]. Other deep learning approaches are based on evidential learning [223, 5], Bayesian methods [156], Variational Bayes [24] or on Hamiltonian Monte-Carlo [37]. Also non-deep-learning approaches have shown significant success, but are less scalable, as for example Gaussian Processes (GP) [210] or approaches based on support vector machines [178]. Gaussian processes are the golden standard for assessing model uncertainty in classical machine learning (i.e. pre-deep learning). Scaling these models to larger datasets and more complex data types, i.e. images, is one of their major challenges and reasons why they are seldomly used for computer vision tasks. Since one of our approaches borrows ideas from MC Dropout [73], we limit our comparison against the latter and the commonly used deep learning golden standard of using an ensemble of trained models [136, 253].

2.9.1 Ensemble of models

To obtain an ensemble of models, several models, which do not need to have the same architecture, are being trained on the same task. Then for inference, all these models together are used and each one is making a prediction for each input sample. Uncertainty is then assessed on the overall consistency of their predictions: the more models of the ensemble predict the same class the more certain the prediction is. One of the obvious down-sides of this approach is the training of many models, deploying all models onto the hardware and using them all for any prediction to make. Let's consider M models trained on solving the same classification task and the index set $\mathcal{I} = \{1, 2, ..., M\}$. The M models $\{f_{\theta_i}\}_{i\in\mathcal{I}} = \{f_{\theta_1}, f_{\theta_2}, ..., f_{\theta_M}\}$ are then interpreted as samples from the family of functions $\mathcal{F}$ all solving the same classification task: $f_{\theta_i} \in \mathcal{F}$ for $i \in \mathcal{I}$.

2.9.2 Monte Carlo dropout

The use of dropout during training and inferences, called Monte Carlo (MC) dropout, has been introduced [73] to model uncertainty in neural networks without sacrificing complexity or test accuracy for several machine learning tasks. For standard classification or regression models, an individual binary mask is sampled for each layer (except the last layer) for each new training and test sample. Consequently, neurons are dropped randomly such that during inference we sample a function f_{ϕ_i} from a family, or distribution of functions $\mathcal{F}$, i.e. $f_{\phi_i} \in \mathcal{F}$. Uncertainty and reliability can then be assessed by performing multiple runs for the same input sample x, i.e. retrieve $\{f_{\phi_i}(x)\}_{i\in\mathcal{J}}$ for $\mathcal{J} = \{1, 2, \cdots, N\}$ for some $N \geq 1$.

2.9.3 Predictive distribution and uncertainty

For both introduced methods we can use the same approach to get the predictive distribution. For an input sample x, the probability of x to be of class c by the ensemble of models $\{f_{\theta_i}\}_{i\in\mathcal{I}}$ or the family of functions obtained by using MC Dropout $\{f_{\phi_i}\}_{i\in\mathcal{J}}$, is then computed by

$$p_c(x;\theta_i) := \frac{1}{m}\sum_{i=1}^{M} p\left(y = c \,|\, x;\theta_i\right) \tag{2.41}$$

$$p_c(x;\phi_i) := \frac{1}{n}\sum_{i=1}^{N} p\left(y = c \,|\, x;\phi_i\right), \tag{2.42}$$

where

$$\left[p\left(y = 1 \,|\, x;\theta_i\right), \ldots, p\left(y = C \,|\, x;\theta_i\right)\right] := \text{Softmax}(f_{\theta_i}(x)) \tag{2.43}$$

$$[p(y=1\,|\,x;\phi_i),\ldots,p(y=C\,|\,x;\phi_i)] := \text{Softmax}(f_{\phi_i}(x)), \tag{2.44}$$

and C is the number of classes. A common metric to assess the uncertainty in the models prediction is the normalized entropy [139] of the probability vectors $p(x;\theta_i) = [p_1(x;\theta_i),\ldots,p_C(x;\theta_i)]$ and $p(x;\phi_i) = [p_1(x;\phi_i),\ldots,p_C(x;\phi_i)]$, computed by

$$H(p(x;\theta_i)) = -\frac{1}{\log(C)}\sum_{c=1}^{C} p_c(x;\theta_i)\log(p_c(x;\theta_i)) \tag{2.45}$$

$$H(p(x;\phi_i)) = -\frac{1}{\log(C)}\sum_{c=1}^{C} p_c(x;\phi_i)\log(p_c(x;\phi_i)). \tag{2.46}$$

Based on a user defined threshold and the value of the entropy for a sample x, the prediction on the latter is then either rejected or accepted. We use the latter in our experiments to compute the uncertainty of the prediction and decide based on its value whether a sample is rejected or accepted for prediction or whether the sample is in- or out-of-distribution.

2.10 Dynamical systems

It might come to a surprise that the penultimate section of the preliminaries talks about dynamical systems. A dynamical system is a system of differential equations describing the evolution of a state over time, either continuously or discrete. Dynamical systems have a wide range of applications, such as physics [168], chemistry [235] and engineering [131] in which they are used to describe and understand natural and physical phenomena. An interesting, but yet not well studied concept, is the interpretation that the recursive application of a trained autoencoder model can be viewed as a dynamical system as well. Thus far, this has been investigated in relation with associative memory for which Radhakrishnan et al. [204] provide a good overview on the basic analysis of autoencoder models, associative memory and attractors.

In case of a discrete time system with a known initial state x_0 a dynamical system is formulated by iterated maps (or difference equation) for which the kth step is formulated as

$$x_{k+1} = f(x_k), \tag{2.47}$$

for $k \in \{0,1,2,\ldots,N\}$ and some $N > 0$. This can then easily be associated with an autoencoder model. Let f_θ be an autoencoder model that finished training under the standard training regime, i.e. minimizing the reconstruction loss $\mathcal{L}_R$ between input x and target $f_\theta(x)$ and variations thereof. The above description of a dynamical system can then also be adopted to

the autoencoder model by considering

$$x_{k+1} = f_\theta(x_k), \tag{2.48}$$

where $x_0 := x$ is a training or test sample and $x_1 := \hat{x} = f_\theta(x)$ the first reconstruction by the autoencoder model. Denote by $f^k(x)$ k compositions of f applied to x, i.e.

$$f^k(x) = (f \circ f \circ \cdots \circ f)(x). \tag{2.49}$$

Adopted to a previously trained autoencoder model f_θ, this equation can be formulated as

$$f_\theta^k(x) = (f_\theta \circ f_\theta \circ \cdots \circ f_\theta)(x). \tag{2.50}$$

We then have that

$$x_k = f^k(x_0). \tag{2.51}$$

Consider an index set $\mathcal{I} = \{1, 2, \ldots, N\}$ for some $N \geq 1$ and define the sequence

$$\{f^k(x)\}_{k \in \mathcal{I}} = \{f^1(x), f^2(x), ..., f^N(x)\}. \tag{2.52}$$

We can then formulate a few terminologies frequently encountered with dynamical systems, which are now also applicable to the setting of using a trained autoencoder model.

Definition (Fixed point [181]). *A point x is a **fixed point** x^* of f if $f(x) = x$. In case of an autoencoder model f_θ, we allow the equality to be weakened, i.e. $f_\theta(x) = x + \varepsilon \approx x$ for some small ε, because the reconstruction will never be perfect.*

Definition (Attractor and basin of attraction [181]). *A fixed point x^* is an **attractor** of f if there exists an open neighborhood $\mathcal{O}$ around x^* such that for all $x \in \mathcal{O}$ the sequence $\{f^k(x)\}_{k \in \mathcal{I}}$ converges to x^* if $k \to \infty$. The set of all such points is called the **basin of attraction** of x^* for f.*

The above definitions and properties can all be applied to autoencoder models f_θ. However, to ensure that f_θ has some fixed points and attractors for the training data distribution, it is necessary to train the model for a large number of epochs and potentially reducing the number of training samples per class. It is also beneficial to train the autoencoder model using the denoising regularization, since the reconstruction will not be as clean as the input image. An interesting occurring property of autoencoder models is then the capability to retrieve the initial version of a disturbed training sample by the iterative application of a trained autoencoder model [204]. We will show that this property can be used to generalize to test samples as well, as long as they are close enough to the training distribution. In case the latter is violated,

the sample might not be stable in its convergence, which will be exploited by design choices presented later in this work.

We want to emphasize an important difference between dynamical systems from physics and engineering and the ones defined by the recursive application of a trained autoencoder model. The former is defined by observations and properties of the system engineers try to model and understand. The latter defines the dynamical system to be considered not by laws of nature, but by the hyperparameters of the autoencoder training process. This means that we can potentially influence the behavior of the autoencoder recursion and its properties by designing the training accordingly. This way it could become possible to *create* attractors and their corresponding basins of attraction with more beneficial properties.

The recursive application of autoencoder models is not the only possibility to induce associative memory. There are other types of models achieving this, e.g. discrete and continuous Hopfield Networks [206, 133, 102] and Predictive Coding [218]. The former needs an energy function to be defined, while the latter is biologically inspired. However, we focus on associative memory achieved by the recursive application of autoencoder models only, because of their elegant simplicity and analogy to dynamical systems, which has been investigated extensively in mathematics and physics [230]. While a few works investigate properties of this model design [112, 204, 203], only one [90] considers attractors for classification and uncertainty estimation. However, the latter adopts this only for speech recognition with respect to noise robustness and combines it with a hidden Markov model (HMM). We, on the contrary, apply this methodology to computer vision and assess the robustness against novel classes and unseen samples from either new datasets or the test distribution.

2.11 Evaluation metrics

After training the autoencoder model it is crucial to evaluate its performance on the task of the problem to solve. This is necessary to validate which model variations have beneficial and which have detrimental effects.

2.11.1 Accuracy

In most cases of this work, the model will be evaluated by simply computing the classification accuracy of the classifier which uses the latent space as input. The classification accuracy is the fraction of correctly predicted samples among all samples considered. This provides a straightforward evaluation metric in case classification labels are accessible. However, there

are also cases where other evaluations need to be considered, either because no classification labels are available, or as an additional criterion.

2.11.2 Semantics

Since we are using autoencoder models, we do not only have the classification as an output of our model, but we also have the reconstruction to our availability. Similar to training the autoencoder model, we can use reconstruction losses to measure whether the semantics are being preserved or not. To this end one can use the ℓ_1 norm, SSIM, perceptual loss and also the LPIPS norm on the reconstructions to compare model performances on test samples. More importantly, one could, and should, use metrics which were not used during training to give a more *objective* measure. It is important to notice that this particular research direction still lacks a universal metric. Vision is a complex problem which cannot be described and assessed by pixel accuracy. Although perceptual losses tighten the gap between human judgement and neural networks, it is still an open research question as to how to describe image similarity in the best possible way to capture all the semantics and human perception nuances.

2.11.3 AUROC, AUPR and FPRN

Classification accuracy can also be a misleading metric, especially in the case of uncertainty estimation and out-of-distribution detection. This is particularly the case when either the positive or negative samples are more likely than the others. In out-of-distribution detection we try to distinguish between samples stemming from the same distribution as the training data (positive) and samples stemming from other datasets (negative). In uncertainty estimation we want to predict whether the classifier makes an error on each of its predictions. In both cases, the models will predict a score assessing the likelihood that the input sample is OOD or wrongly classified. Based on a pre-defined threshold the score then leads to a detection or not. However, due to the latter threshold, some samples will wrongly be classified as negative or positive. This leads to a trade-off between false negatives (FN) and false positives (FP) on which our evaluation metrics will be based on. For these evaluations we adopt several commonly used metrics [95, 96, 49, 159, 149]: Area Under the Receiver Operating Characteristic curve (AUROC), Area Under the Precision-Recall curve (AUPR) and false positive rate at N% true positive rate (FPRN). The first two are threshold independent metrics, since they evaluate the model across many thresholds summarizing the performance. For each threshold, we compute the number of false negatives (FN), false positives (FP), true negative (TN) and true positive

(TP). The ROC curve plots the true positive rate (TPR)

$$\text{TPR} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{2.53}$$

against the false positive rate (FPR)

$$\text{FPR} = \frac{\text{FP}}{\text{FP}+\text{TN}}, \tag{2.54}$$

from which the area under the curve is being computed. A perfect classifier would achieve an AUROC of 100% while a random classifier would achieve an AUROC of 50%. Notice that also a performance smaller than 50% can be possible. AUPR works similarly as AUROC, but the PR plots the precision

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{2.55}$$

against recall

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}. \tag{2.56}$$

Finally, the FPRN is not threshold-independent and it operates on a pre-defined fixed threshold. As the names suggests, we fix a threshold N and get the false positive rate corresponding to a true positive rate of N. The three different metrics have their advantages and disadvantages, which is why usually all three metrics are reported. For further details and interpretations of the metrics we refer to [95, 96, 49, 159, 149].

2.12 Conclusion

The compact overview and background information of this chapter provides the necessary machine learning context to follow the derivations and investigations for the rest of the thesis. We introduced an overview of the datasets, model architectures and metrics used and established a common framework to formulate the different autoencoder models therein. We described several autoencoder reconstruction losses and regularization methods and reported experimental results to support the design choices adopted in the rest of this thesis. Some of the challenges our industrial application is facing were discussed, an overview of the related work was provided and autoencoder models were compared against other generative models.

Chapter 3

Vehicle interior rear seat occupancy detection

The questions investigated in this thesis stem from challenges of the industrial application of detecting occupancy on the rear bench in the vehicle interior. While we will always evaluate our contributions on datasets commonly used by the research community, we will also demonstrate its successes on a real application. To this end, we will explain in this chapter the provided proprietary dataset (but of course not all details can be provided) and describe the synthetic data generation process to imitate the real application. Regarding the latter, we will present the different objects and software used to reconstruct the real application in a synthetic environment and explain our thought processes to justify our design choices. Lastly, the different extensions to SVIRO will be introduced and their key features will be highlighted.

3.1 Optical Rear Seat Sensing (ORSS)

Information about the presence and location of the passengers inside a vehicle interior can be used to help reduce injuries in case of an accident, e.g. by adjusting the strength of airbag deployment [69, 197]. Seat occupancy detection can be used to remind the passengers to fasten their seat-belts or to detect children forgotten in the car [52, 60]. For autonomous driving, it will be of interest to understand the overall scenery in the car interior [200], e.g. for handover situations [165]. To this end, IEE S.A. developed a camera-based solution for rear seat occupant detection and classification referred to as Optical Rear Seat Sensing (ORSS).

The system consists of a 2D active near-infrared camera system. This means that the camera system actively emits near-infrared light to illuminate the rear bench. This should help the

(a) Camera setup (b) Close-up of camera system (c) Recorded image

Fig. 3.1 The active near-infrared camera system recording the rear bench is integrated into the roof (a). A close-up (b) of the camera setup of the red frame in (a). An example of a recorded image with lens distortion (c) contains a child on a child seat (left), an adult (middle) and an infant seat with a sun-protection (right).

system to become less susceptible to illumination changes and to ensure operability during night. The camera system is integrated into the roof, as illustrated in Fig. 3.1.

The recorded images need to undergo a few pre-processing steps to make them more suitable for machine learning-based classifiers. The images need to be rectified, i.e. remove the lens distortion, and a plateau histogram equalization [146] needs to be applied to enforce more equal illumination conditions across all recorded images and daytimes.

In this work, we consider the task of identifying for each seat position (left, middle and right) whether the seat is empty, an infant in an infant seat, a child on a child seat or an adult is occupying it. IEE S.A. provided images from two vehicle interiors: Volkswagen Sharan (2860 training and 744 test images) and BMW X5 (2832 training and 770 test images). Example images for the interior of the two vehicles are plotted in Fig 3.2.

On the one hand, one can split the image into three rectangles, one for each seat position, and classify each rectangle independently from the others resulting in a four class machine learning problem. On the other hand, one can also classify the whole image and, since for each of the three seat positions there are four possible classes, classifying the whole image results in $4^3 = 64$ classes. The benefit of the latter approach is the possibility to account for the whole scenery such that people and objects leaning over the neighboring seat can be more efficiently accounted for. The benefit of the former is the reduction in complexity and variability: only 4 classes need to be discriminated against.

(a) Sharan

(b) X5

Fig. 3.2 Example images of recorded sceneries by the ORSS camera system for two vehicle interiors: (a) Volkswagen Sharan and (b) BMW X5.

3.2 SVIRO

It was not possible to publish the ORSS dataset, because it is proprietary. Hence, we decided to develop a synthetic image generation pipeline to imitate the real application. This way, the results and design choices could be reproduced by the research community. Further, since no similar dataset existed, and since the questions and challenges of the proposed application are underrepresented in the research community, these new datasets also allow novel investigations. Other advantages of the synthetic data generation pipeline are the free and versatile labels: they are generated automatically by the simulation software. We will explain the data generation process for SVIRO and its extensions in this chapter. Baseline results and basic investigations are reported in Chapter 5. SVIRO is the base dataset and additional extensions were generated. All the extensions follow the same framework, but each extension has its own characteristics and tasks to be investigated on. All the datasets can be downloaded from our website - https://sviro.kl.dfki.de.

(a) Image of a real Maxi Cosi (b) Scanned Maxi Cosi - View 1 (c) Scanned Maxi Cosi - View 2

Fig. 3.3 Example of a real Maxi Cosi (a) which was scanned using the Artec Eva scanner. After post-processing, the infant seats can be used as a 3D object inside the simulation software such that, for example, the views can be changed (b) and (c). The scanned object has an ISOFIX base attached to it and the handle was turned up, which is not the case for the real image.

3.2.1 Synthetic objects

We used the free and open source 3D computer graphics software Blender 2.79 [41] to construct and render the synthetic 3D sceneries. We used realistic child safety seats or child restraint systems (CRS) to which we simply refer to as child seats. For our dataset, we selected a subset of available seats on the market and approved by the European New Car Assessment Programme (Euro NCAP) and the National Highway Traffic Safety Administration (NHTSA), from which we then created a 3D model so that it could be used in our simulation. The 3D models were generated using depth cameras (Kinect v1) and precise structured light scanners (Artec Eva). An example of a scanned infant seat can be found in Fig. 3.3.

One needs to define the reflection properties and visual colors for each 3D object in the scene, so that its perception by the camera under simulated illumination conditions could be calculated. For this, we used textures (Albedo, Normal and Roughness images) from Textures.com [236] (with permission) for all the objects in the scene. The environmental background and illumination were created by means of High Dynamic Range Images (HDRI) from HDRI Haven [92]. The human models (adults, children and babies) and their clothing (additional clothes were downloaded from the community assets [173]), were randomly generated by using the open source 3D graphic software MakeHuman 1.2.0 [173]. The 3D models of the cars were purchased from Hum3D [106] and everyday objects (e.g. backpacks, boxes, pillows) were downloaded from Sketchfab [10].

3.2.2 Design choices

During the data generation process we tried to simulate the conditions of a realistic application. We decided to partition the available human models, child seats and backgrounds such that

one part is only used for the training images (for all the vehicles) and the other part is used for the test images. For each of the ten different vehicle passenger compartments and available child seats, we fixed the texture as if real images had been taken. Consequently, the machine learning models need to generalize to previously unknown variations of humans, child seats and environments. In this setting, we can train models in one or several car environment(s) and test them on a different one. This is an advantage compared to common domain adaptation datasets [194, 250, 193, 174, 229] which usually focus on the transfer from synthetic to real images. Further, the photorealistic rendering and close-to-real models introduce a high visual complexity which makes them more challenging than toy examples [31, 115]. The dataset has an intrinsic dominant background and texture bias: all of the images are taken in a few passenger compartments, but generalization to new, unseen, passenger compartments and child seats should be achieved. This evaluation is currently not possible by state-of-the-art datasets [42, 27, 67, 66, 76, 3, 147].

The human models were generated randomly using MakeHuman. Their facial expression was selected to be neutral and identical. We defined a fixed set of poses for the humans represented by unit quaternions. For every human in each scenery, two poses were selected randomly and a spherical linear interpolation (Slerp) [47] was performed to get an intermediate pose. For each scenery, we randomly selected what kind of object is placed at each position, however, we avoided appearances of the same object for the same scenery. Child and infant seats can be empty. This is one of the few differences between SVIRO and ORSS: ORSS only uses occupied child and infant seats. We also decided to not allow children to be placed on the rear seat without a child seat. Infant seats were randomly rotated by 180° along the z-axis and an offset from the straight ahead orientation was randomly applied to all child seats. The handle of the infant seat was selected to be up or down. Randomly selected environmental backgrounds were rotated around the vehicle to simulate arbitrary illumination conditions. We placed everyday objects onto the rear seat to make the scenery more versatile. All cameras have the same intrinsic parameters [45] (focal length=3.4 mm, sensor width=8.5 mm, f-number= 2.5, skew coefficient= 0, focal length in terms of pixels: $\alpha_x = 514.4208$, $\alpha_y = 514.4208$, principal point: $u_0 = 640$, $v_0 = 480$), however, their pose is different in each car. Example sceneries for training and test data can be found in Fig. 3.4. An overview of the 3D objects and backgrounds are shown in Fig. 3.5 and Fig. 3.6.

We also generated a training dataset with randomly selected (partially unrealistic) textures and backgrounds from a large pool of images. Examples for this variation of the dataset are shown in Fig. 3.7. When trained on the latter, the increased variations improve the generalization for classification and semantic segmentation on the test set and to new passenger compartments, as shown in Section 5.2.1 and 5.2.2. Moreover, the difficulty can be gradually increased: one

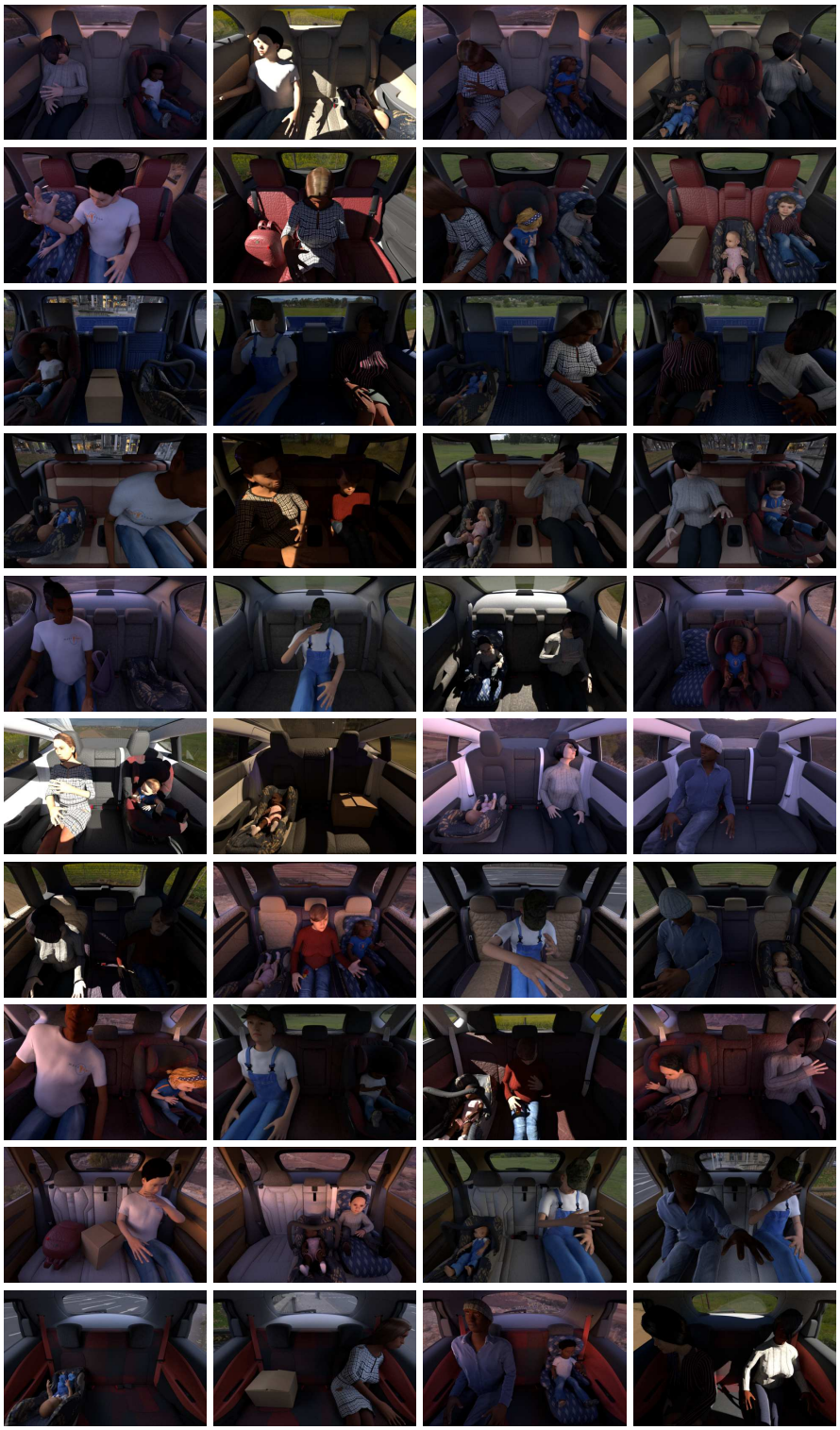


Fig. 3.4 Example sceneries from the ten vehicle interiors. Some images appear darker, which is why (also in real applications) it is preferred to use an active near-infrared camera system.

(a) Children

(b) Infants in infant seats

(c) Infant (first two) and child seats

(d) Everyday objects

Fig. 3.5 Representative selection of some of the assets used for our synthetic dataset. Some are used for training, others for generating the test images.

(a) Adults

(b) Environments

Fig. 3.6 Representative selection of some of the assets and environments used for our synthetic dataset. Some are used for training, others for generating the test images.

Fig. 3.7 Examples from the SVIRO X5 for randomized textures and environments.

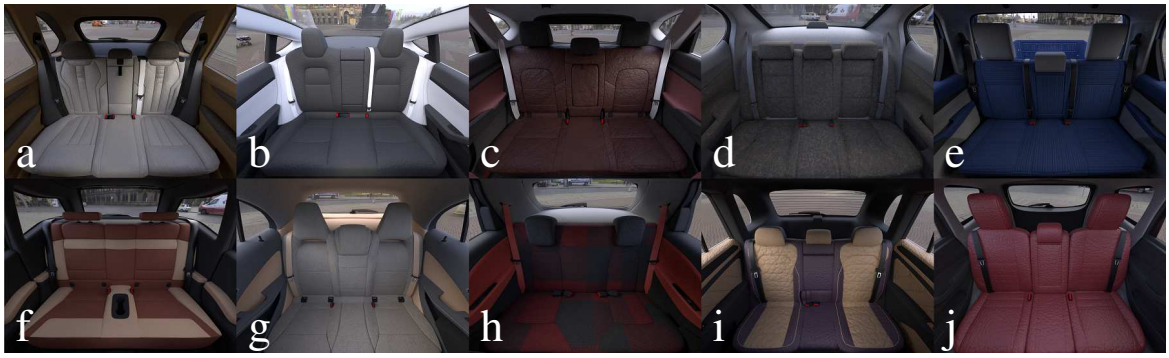


Fig. 3.8 Comparison of the different vehicle interiors. a) BMW X5, b) Tesla Model 3, c) Hyundai Tucson, d) Lexus GS F, e) Toyota Hilux, f) BMW i3, g) Mercedes A-Class, h) Renault Zoe, i) VW Tiguan and j) Ford Escape. The geometry of the rear-seat, the windows, headrest and car features differ between the cars and some have two seats instead of three.

can train on occupied child and infant seats only, train on infant seats with the handle down (or up) only or removing everyday object completely from training.

3.2.3 Statistics

SVIRO consists of ten different vehicles: BMW X5, BMW i3, Hyundai Tucson, Tesla Model 3, Lexus GS F, Mercedes A-Class, Renault Zoe, VW Tiguan, Toyota Hilux and Ford Escape. The number of windows varies, which causes different illumination conditions, and the i3 and Zoe have only two rear seats instead of three. The different vehicle interiors are compared in Fig. 3.8. We used the same people and child seats for the training set of each vehicle and the remaining ones for the test sets. This results in two child seats and one infant seat per data split. We did the same for the background: five were selected for the training and five different ones for the test set. For the everyday objects, we used two bags, a card-box and a cup for the training dataset and a different bag, a paper-bag, pillows and a box of bottles for the test set. The number of people and the distribution of the gender, age and ethnicity for the training and

Table 3.1 Number of people and distribution of gender, age and ethnicity for the training and test datasets. The same people were used for the training and test set for all the vehicles, respectively, and the same number of images were generated for each car.

	Train			Test		
	Adult	Child	Baby	Adult	Child	Baby
African	5	2	1	2	1	1
Asian	5	2	1	2	2	1
Caucasian	4	2	1	4	1	1
Female	9	3	-	5	2	-
Male	5	3	-	3	2	-
Total	14	6	3	8	4	3

test set can be found in Table 3.1. The number of images generated for each vehicle and each training and test set are identical: 2000 for each training and 500 for each test set. In total, this results in 20000 training and 5000 test sceneries. The distribution of the different classes across the vehicles and data splits is summarized in Table 3.2. The number and constellation of appearances varies between the vehicles, because all the sceneries were generated randomly.

3.2.4 Rendering

The synthetic images were generated using Blender, its Python API and the Cycles renderer. As many applications in the passenger compartment require an active near-infrared camera system to work in the dark, we decided to imitate such a system by means of a simple approach: We placed an active red lamp (R=100%, G=0%, B=0%) next to the camera inside of the car illuminating the rear seat, but overlapping with the illumination from the HDR background image. We then took the red channel only from the resulting rendered RGB image. We will refer to these images as grayscale images. This is, however, not a physically accurate simulation of a real active near-infrared camera system. The simulation of the latter is not trivial, as the perception in the infrared domain not only depends on the object's material properties, but also on the wavelength which is used [198]. We argue that this is of minor importance, because SVIRO is intended to investigate the general applicability of possible machine learning methods. Our infrared imitation helps to become less dependent on the environmental illumination. Hence, it facilitates the actual machine learning tasks: see Fig. 3.9 for a comparison between a standard RGB image and our grayscale image of a dark scenery, where a lot of information would otherwise be lost. Moreover, we report in Section 5.2.6 the

Table 3.2 Distribution of the different classes over the vehicles and data splits. As the images were generated randomly, the distribution is different for each split and vehicle. We abbreviate infant seat as IS and child seat as CS. The large difference in empty seats and everyday objects is due to the two vehicles with only two rear seat positions.

Classes	Empty		IS		CS		Adult		Object		Empty IS		Empty CS	
Vehicle	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
A-Class	2134	614	457	126	611	121	884	191	755	179	486	124	673	145
Escape	2079	569	489	133	581	143	940	215	742	187	443	108	726	145
Hilux	2218	553	457	116	560	130	847	232	769	194	510	125	639	150
i3	884	180	372	117	496	98	919	223	442	129	363	113	524	140
GS F	2127	565	465	121	579	140	907	219	791	195	468	113	663	147
Model 3	2507	613	449	121	537	107	909	224	565	196	439	105	594	134
Tiguan	2196	592	458	112	645	128	944	227	650	180	461	112	646	149
Tucson	2202	565	458	103	608	139	900	231	658	204	481	119	693	139
X5	2400	610	371	109	569	100	892	234	767	195	418	124	583	128
Zoe	909	195	380	125	518	115	816	189	438	131	392	119	547	126

Fig. 3.9 Comparison between a standard RGB image and our simple approach to imitate an active near-infrared camera system for a dark scenery. a) Standard RGB image in environmental illumination. b) RGB image of the scenery illuminated by an active red light. c) Red channel only of the RGB image of the illuminated scenery (used as infrared imitation).

evaluation of a model trained on SVIRO on real infrared images and show that it behaves similarly on real data.

3.2.5 Ground truth

For each scenery we provide a set of images and ground truth data:

1. RGB image of the scenery without an active red lamp next to the camera, e.g. Fig. 3.4,
2. grayscale image (red channel only) of the rendered RGB image using an active red lamp next to the camera, e.g. Fig. 3.10 (b),
3. instance segmentation map, where each object is color-coded depending on its position and class, e.g. Fig. 3.10 (c),

Fig. 3.10 Example scenery of SVIRO together with some of the provided ground truth data. Left seat: infant seat with an infant. Middle seat: empty. Right seat: adult passenger. a) RGB image with keypoints for human pose estimation. b) Grayscale near-infrared imitation. c) Position and class based instance segmentation. d) Depth map.

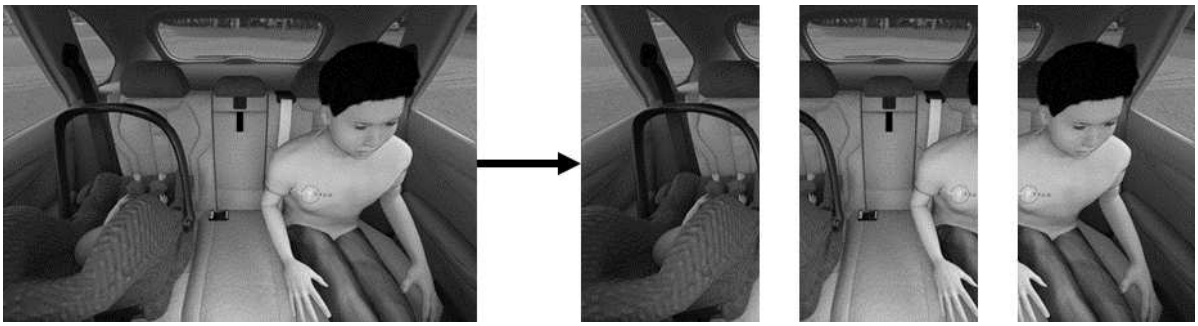


Fig. 3.11 One can split each image into three rectangles to use them for classification. The rectangles should overlap slightly, because objects are not limited to their seat position.

4. bounding boxes for all the elements in the scenery,
5. keypoints for all the human poses in the scenery, e.g. Fig. 3.10 (a),
6. depth map of the scenery, e.g. Fig. 3.10 (d).

We also split the images (RGB, grayscale, depth) into three rectangles (one for each seat position) with slight overlap between them. See Fig. 3.11 for an illustration. If a car has only two seats, then we exclude the middle rectangle. Note that people can lean over to the neighboring seat and objects from neighboring seats are overlapping to the neighboring rectangle, which makes classification more difficult. It is, however, also possible to classify the image as a whole. Both semantic segmentation and instance segmentation can be performed using the provided segmentation masks. Children on a child seat, as well as babies in an infant seat, are treated as two separate instances. We save the human poses by using keypoints, as used by the COCO dataset [147], but our skeleton is defined using partially different joints. The visibility of the keypoints are set to zero if the keypoint is outside the image, to one if it is occluded by an object or neighboring human and set to two if it is visible or occluded by the person itself. Keypoints are provided for the babies as well. For each scenery, we provide a .json file containing the 2D pixel coordinates of the keypoints of all people together with

Table 3.3 Definition of labels for the different tasks on SVIRO.

	Classification	Segmentation	Object detection	Keypoints
Empty	0	0	0	0
Infant in infant seat	1	-	-	1
Child in child seat	2	-	-	1
Adult	3	3	3	1
Everyday object	4	4	4	0
Empty infant seat	5	1	1	0
Empty child seat	6	2	2	0

the visibility flag, the bone names and their seat position. All the images are provided in .png format. The depth maps are saved in millimeters and as 16-bit .png images. The bounding boxes are given in the format $[\text{class}, x_1, y_1, x_2, y_2]$, where (x_1, y_1) is the upper left corner and (x_2, y_2) the lower right corner of the bounding box (coordinates start in the upper left image corner). An overview of the labels for the different tasks is listed in Table 3.3. We did not fasten the seat-belt for our models and let them un-attached in all our sceneries.

3.3 SVIRO-Illumination

Using a synthetic renderer and modelling software yields the possibility to generate images which are difficult, expensive or even impossible to achieve for real images. One of these additional benefits is the generation of exactly the same sceneries where only the illumination changes. The availability of such images enables the design of novel model architectures, as we will present later in this work. Based on the above presented framework used to generate SVIRO, we created additional images for three new vehicle interiors: Porsche Cayenne, Skoda Kodiaq and Hyundai Kona. For each vehicle, we randomly generated 250 training and 250 test scenes where each scenery was rendered under 10 different illumination and environmental conditions. We created two versions: one containing only people and a second one including additionally occupied child and infant seats. We used 10 different exterior environments (HDR images rotated randomly around the vehicles), 14 (or 8) human models, 6 (or 4) children and 3 babies respectively for the training and test split. The four infant and two child seats have the same geometry for each split, but different textures are being used. Consequently, the models need to generalize to new illumination conditions, humans and textures. There are four possible classes for each seat position (empty, infant seat, child seat and adult) leading to a total of $4^3 = 64$ classes for the whole image. Examples are shown in Fig. 3.12. Differently from

the SVIRO base set, this dataset does not have empty child and infant seats and no everyday objects.

3.4 SVIRO-NoCar

Similar to the possibilities presented in the previous section, it is also possible to remove the vehicle, i.e. we placed humans, child and infant seats as if they would be sitting in a vehicle interior, but instead of a vehicle, the background was replaced by selecting randomly from a pool of available HDR images. This way we can generate input-target pairs where both images are of the same scene, but differ in the properties we want to become invariant to: the dominant background. We created 2938 training and 2981 test sceneries where each scenery is rendered with 10 different backgrounds out of a pool of 450 backgrounds. The background and the corresponding illumination conditions were defined using the HDR images. As for SVIRO-Illumination, there are four possible classes for each seat position (empty, infant seat, child seat and adult) leading to a total of $4^3 = 64$ classes for the whole image. We randomly created 172 adults and we used 6 child seats and 7 infant seats which were textured using randomly one out of five textures. This dataset will help to overcome the problems for transferring between different vehicle interiors, but it also helps to learn invariances to be transferred to real images. For example, we noticed that a larger number of different human models increases the transferability to real images. Examples are visualized in Fig. 3.13.

3.5 SVIRO-Uncertainty

Particularly important for safety critical applications is the possibility to provide uncertainties together with the models' predictions. Most of the readers of this thesis have probably put the most random objects inside their vehicles (e.g. furniture and washing machine, instruments, food and beverages, animals). It is hence paramount to somehow account for these random events that a deployed system could be encountered with. Hence, we decided to create additional images for the vehicle interior which could be used to assess a model's reliability. As before, for each of the 3 seat positions in the vehicle interior rear bench the model should classify which object is occupying it, with empty being one possible choice. We created two training datasets for the Volkswagen Sharan vehicle, a new synthetic vehicle not used before, using adult passengers only (4384 sceneries and 8 classes) and one using adults, child seats and infant seats (3515 samples and 64 classes). We created fine-grained test sets to asses the reliability on several difficulty levels:

(a) Cayenne

(b) Kodiaq

(c) Kona

Fig. 3.12 Example sceneries from SVIRO-Illumination. For each vehicle, one scenery under ten different illumination and external environments is shown.

Fig. 3.13 Examples of identical sceneries with different backgrounds from SVIRO-NoCar.

Fig. 3.14 Examples of sceneries from SVIRO-Uncertainty containing everyday objects.

1. only unseen adults (2617 sceneries),
2. only unseen child and infant seats (490 sceneries),
3. unseen adults and unseen child and infant seats (896 sceneries),
4. unknown random everyday objects (e.g. dog, plants, bags, washing machine, instruments, tv, skateboard, paintings - 1622 sceneries),
5. unseen adults and unknown everyday objects (1421 sceneries),
6. unseen adults, child and infant seats and unknown everyday objects (1676 sceneries).

Unseen means that the test set uses new and different adults, child and infant seats not encountered during training. Besides the uncertainty estimation within the same vehicle interior, one can use images from unseen vehicle interiors from SVIRO (or any of the other extensions) to further test the models reliability on the same task, but in novel environments, i.e. vehicle interiors. One could also train on the SVIRO vehicles together with its everyday objects and check whether the model can generalize to the fine-grained test sets mentioned above. Example images are provided in Fig. 3.14.

3.6 SVIRO-InterCar

The last extension of SVIRO is being published with the release of this thesis. Similar to SVIRO-Illumination we generated each scenery several times by varying one key feature. While it was the illumination condition for SVIRO-Illumination, for SVIRO-InterCar we changed the entire vehicle interior. This means that not only the vehicle interior looks different, but also the perspective is adapted accordingly. Since each vehicle has different windows, this also means that the illumination changes. We generated two versions: one with adult people

Fig. 3.15 Examples of identical sceneries in different vehicle interiors from SVIRO-InterCar.

only (1000 sceneries) and one with adults, infants in infant seats and children in child seats (990 sceneries). We used the 8 vehicles from SVIRO with three seats and the Cayenne and Kona from SVIRO-Illumination. We hope that this dataset might inspire some interesting future design choices, investigations and research results. For example, would it be possible to incorporate the extrinsic camera parameters to switch from a source vehicle to another (unknown) target vehicle? Can in terms of disentanglement a dimension be used to change the vehicle? Example images are shown in Fig. 3.15.

3.7 Conclusion

We presented the proprietary ORSS dataset for the vehicle interior and explained the corresponding synthetic data generation process. SVIRO and its different extensions were introduced and its key features and differences were explained. These different splits will be used throughout this thesis to assess how the different challenges might affect the industrial application of vehicle interior occupancy detection. This step was particularly important to provide a public version of our investigated scenario such that the results and insights can be reproduced by the research community and to contribute a novel interesting application for future research.

A basic analysis on SVIRO will be reported in Chapter 5 and a much more detailed analysis for the transferability from vehicle-to-vehicle in Chapter 6. In order to investigate our novel illumination normalization method, we will adopt SVIRO-Illumination in Chapter 7. For synthetic to real generalization we will use SVIRO-NoCar in Chapter 8. Finally, SVIRO-Uncertainty will be integrated in our uncertainty estimation and out-of-distribution detection investigation in Chapter 9.

Chapter 4

Methods

We provide an overview of the novel methods proposed and used in this work. We start by formulating a framework and common language, which will be used by the rest of this thesis. We will then explain the different building blocks which are presented as standalone, but which can also be combined. First, we will introduce our proposed ***"Partially Impossible Reconstruction Losses"*** (PIRL) for autoencoder networks. The first variation of the PIRL exploits the availability of identical sceneries under different conditions, e.g. illumination. We will extend this approach by applying a triplet loss regularizer in the latent space to improve generalization. This induces some useful properties such that more robust and reliable results on unseen test samples can be achieved by adopting the nearest neighbor search. Next, we will propose a second variation of the partially impossible reconstruction loss which can readily be used with most existing datasets. The benefits and disadvantages with respect to the first variation will be explained, but also highlighted later in this thesis. We will introduce the extractor autoencoder model which uses a pre-trained model to extract features from the input images and which will be helpful to generalize to real images when trained on synthetic ones only. Further, we will explain the multi-channel autoencoder approach, which will be helpful for the transfer from synthetic to real images when real images can be included during training as well. We will show that it can be used to remove the background from real images: the synthetic gap is reduced by including real images during training and the invariance is learned on dedicated designed synthetic images. Then, we will explain the possibility to model uncertainty using MC Dropout in case of autoencoder models. Finally, we introduce the recursive application of previously trained autoencoder models and their attractors and how they can be used to model uncertainty as well. The methods will be introduced using examples from the SVIRO datasets and its extensions, as presented in Chapter 3, but they will also be tested on other datasets later in this work.

4.1 Framework

Consider N_s sceneries and N_v variations of the same scenery, e.g. same scenery under different illuminations, with different backgrounds or under different data augmentation transformations. Most commonly used datasets will have $N_v = 1$ unless they are cleverly augmented by transformations (i.e. preserving the important semantics) to adopt our proposed sampling strategies. Let $\mathcal{X} = \{X_i^j \,|\, i \in \{1,\dots,N_v\}, j \in \{1,\dots,N_s\}\}$ denote the training data, where each $X_i^j \in \mathbb{R}^{C\times H\times W}$ is the ith variation of scene j. Hence, the dataset consists of a total of $|\mathcal{X}| = Nv \cdot N_s$ images. Let $X^j = \{X_i^j \,|\, i \in \{1,\dots,N_v\}\}$ be the set of all variations i of scenery j and $\mathcal{Y} = \{Y^j \,|\, j \in \{1,\dots,N_s\}\}$ be the corresponding target classes of the scenes of $\mathcal{X}$. Notice that the classes remain constant for the variations i of each scene j. This is important for classification tasks and it needs to be adapted in case our sampling strategies are adopted for other tasks, e.g. landmark detection. Using the above framework, the vanilla autoencoder cost function presented in Eq. (2.1) is slightly modified

$$\mathcal{L}_R(X_i^j, f(X_i^j); \theta, \phi) = \mathcal{L}_R(X_i^j, \mathrm{d}_\theta(\mathrm{e}_\phi(X_i^j)); \theta, \phi) = \mathrm{r}(X_i^j, \hat{X}_i^j; \theta, \phi), \tag{4.1}$$

where f is the autoencoder model with e_ϕ being the encoder with parameters θ and d_θ the decoder with parameters ϕ, X_i^j the input image and $\hat{X}_i^j$ the reconstruction and r the reconstruction loss of choice.

4.2 First sampling strategy: Partial impossible

One of the main contributions of this thesis is the introduction of a novel sampling strategy for which we provide two variations. Our proposed ***"Partially Impossible Reconstruction Losses"*** (PIRL) can be applied to any autoencoder neural network architecture. Both variations formulate a novel reconstruction loss based on the autoencoder reconstruction and a different variation of the input image used as target.

The first variation of the PIRL is the weaker version, because it is tailored to the task to be solved and it needs a controlled variation of the task specific unwanted features. For example, we can select the same scene under different illumination conditions and/or with different backgrounds. It is important that the features we want to become invariant against are being varied and the features we assess as discriminative are being kept identical or similar. This also means that the target of the image should not be changed (e.g. if flipping the images would change the label). In this work, for sampling the individual elements of a batch, we randomly select for each scene two images, one as input and the other one as target. This sampling strategy preserves the semantics while varying the unimportant features such that the model

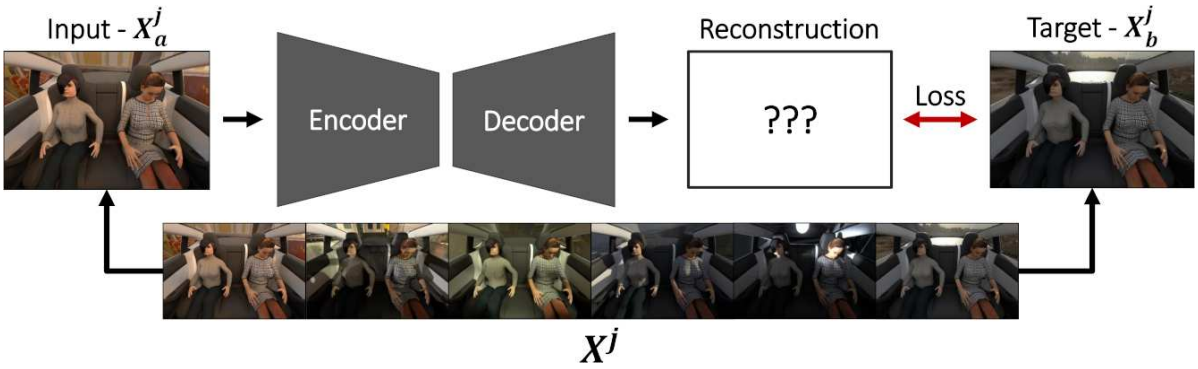


Fig. 4.1 Variation I of the partially impossible reconstruction loss together with a sampling example: input X_a^j and target X_b^j are images of the same scenery X^j, but under different illumination conditions.

needs to focus on what remains constant. More precisely, during training, the batches iterate over the X^j and for each X^j we randomly select $a,b \in \{1,2,\ldots,N_v\}$, $a \neq b$ to get $X_a^j, X_b^j \in X^j$. Finally, X_a^j is considered input to the autoencoder network and X_b^j is considered as the target for the reconstruction loss. The aforementioned method is illustrated in Fig. 4.1. This leads to the loss formulation:

$$\mathcal{L}_{R,I}(X_a^j, X_b^j; \theta, \phi) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{e}_\phi(X_a^j)), X_b^j\right), \tag{4.2}$$

for a reconstruction loss r of choice. We refer to models using this variation of the PIRL by prepending an I, i.e. we call it I-PIRL and the models using it, for example, I-AE.

4.3 Second sampling strategy: Partial impossible class instance

The first variation of the PIRL will lead to a better removal of illumination, shadow and environmental information and improve the accuracies by the classifier in the latent space, particularly in case human poses need to be preserved. However, it can be challenging to apply it to a lot of commonly recorded datasets, especially when the dataset to be considered does not allow for a controlled variation of the unwanted features. To this end we introduced a second, stronger variation of the previously introduced PIRL which can readily be applied to most existing datasets. Instead of sampling the same scene under a controlled variation, e.g. same scene under different illumination, we propose to use as target image a different image of the same class as the input. This approach implicitly uses label information in the input space, however, as we will show, this leads to a better latent space representation. This loss variation causes the model to learn invariances with respect to certain class variations which are

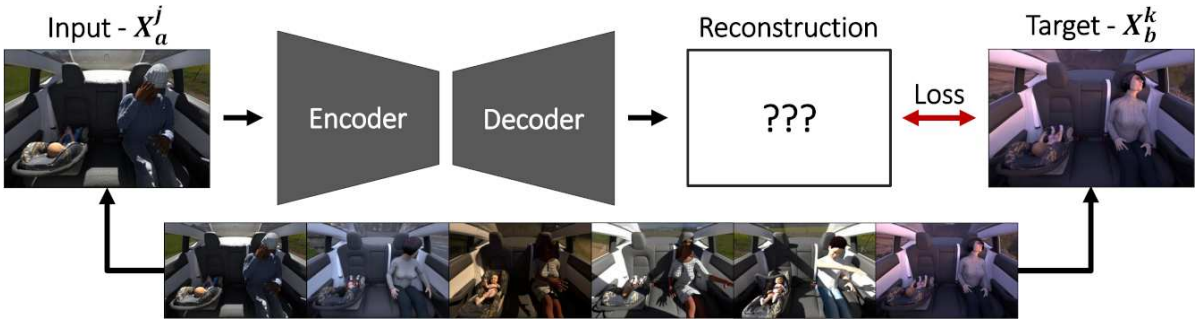


Fig. 4.2 Variation II of the partially impossible reconstruction loss together with a sampling example: for the input X_a^j a new scenery X_b^k of the same class label is selected as target.

(a) Vanilla: Eq. (4.1) (b) I-PIRL: Eq. (4.2) (c) II-PIRL: Eq. (4.3)

Fig. 4.3 Comparison of different input-target pairs for the different reconstruction losses.

not important for the task at hand, e.g. clothes, human poses, textures. While illumination can still be removed with this approach, human poses will no longer be preserved. This sampling variation is reflected in the reconstruction loss as follows:

$$\mathcal{L}_{R,II}(X_a^j, X_b^k; \theta, \phi) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{e}_\phi(X_a^j)), X_b^k\right), \tag{4.3}$$

for random $a, b \in \{1, 2, \ldots, N_v\}$, $j \neq k$ and $Y^j = Y^k$. We refer to this method as partially impossible class instance sampling marked by prepending *II* when it is used, i.e. referring to it as II-PIRL and the models using it as, for example, II-AE. This sampling variation is visualized in Fig. 4.2. The first and second sampling variations are compared against the vanilla autoencoder sampling in Fig. 4.3.

4.4 Structure in the latent space: Triplet loss and nearest neighbour search

While the I-PIRL works well on the training data, generalizing to unseen test images can remain a challenging task if no additional precautions are taken. For example, the illumination is still removed from test samples, but the reconstruction of the objects of interest can be less stable. In

case training data is limited, the autoencoder network is mostly used as a compression method instead of a generative model. Consequently, generalizing to unseen variations with respect to reconstruction quality cannot trivially be achieved. It turns out that a blurry reconstruction is in fact a blurry version of the reconstruction of its nearest neighbor in the latent space (or a combination of several nearest neighbors). Interestingly, in terms of classification accuracy, the II-PIRL does not suffer from the generalization to test samples problem of the I-PIRL, probably because of the implicit use of label information in the pixel space. However, on the other side, the II-PIRL cannot conserve human poses in its reconstructions, not even on the training images. Nevertheless, incorporating the triplet loss with II-PIRL does not have a detrimental effect.

Consequently, instead of reconstructing the encoded test sample when the I-PIRL is used, it is more beneficial to reconstruct its nearest neighbor. However, applying nearest neighbor search in the latent space of a vanilla autoencoders or variational autoencoders will not provide robust results. This is due to the fact that there is no guarantee that the learned latent space representation follows an L^2 metric [9]. As the nearest neighbor search is (usually) based on the L^2 norm, the latter will hence not always work reliably. To this end, and as stated in Eq. (2.20), we incorporated a triplet loss in the latent space of the autoencoder model (TAE) instead. This can lead to a better nearest neighbour retrieval. The triplet loss can be used with or without the PIRL. In case of the former, using the same notations as before, the triplet loss is formulated here as

$$\begin{aligned}&\mathcal{L}_T(X^j_{a,a}, X^k_{b,p}, X^l_{c,n}; \theta)\\ &\qquad = \max\left(0, \alpha + \left\|\mathrm{e}_\phi(X^j_{a,a}) - \mathrm{e}_\phi(X^k_{b,p})\right\|_2^2 - \left\|\mathrm{e}_\phi(X^j_{a,a}) - \mathrm{e}_\phi(X^l_{c,n})\right\|_2^2\right),\end{aligned} \tag{4.4}$$

for random $a, b, c \in [0, N_v]$, $j \neq k \neq l$ and $Y_j = Y_k \neq Y_l$. where $X^j_{a,a}$ is the anchor using scenery j, $X^k_{b,p}$ is the positive sample using a different scenery k and $X^l_{c,n}$ is the negative sample using another scenery l. An illustration for the triplet loss is given in Fig. 4.4 and for the nearest neighbor inference in Fig. 4.5.

Our sampling strategies could easily be combined with other commonly used sampling strategies, but in this work we limit ourselves to the triplet loss to induce a regularization and structure in the latent space of autoencoder models. Notice that we take full advantage of the triplet selection by choosing the triplets from the same batch such that the positive and negative samples are reconstructed as well.

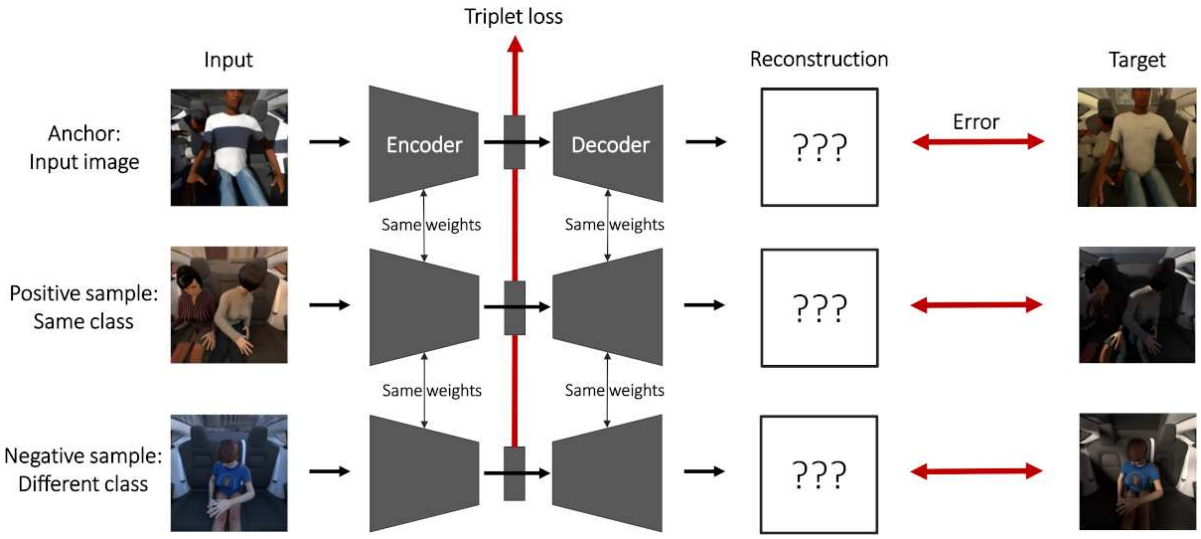


Fig. 4.4 Illustration of the triplet loss when applied to SVIRO-Illumination. We chose the positive sample to be of the same class as the anchor image (but from a different scenery) and the negative sample to differ only on one seat (i.e. change only the class on a single seat w.r.t. the anchor image). Notice the difference in illumination of the target image w.r.t. input image in order to apply our proposed I-PIRL.

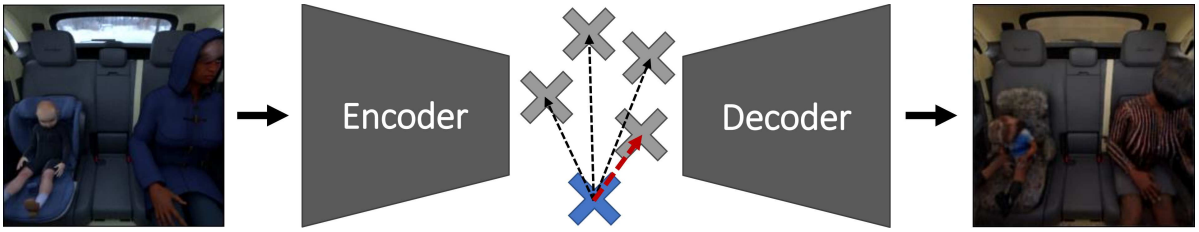


Fig. 4.5 During inference, we choose the nearest neighbor (red arrow) of the latent space vector of the input image (blue cross) from all the training latent space vectors (grey crosses). This can be used to reconstruct a clean image or as classification prediction using its label.

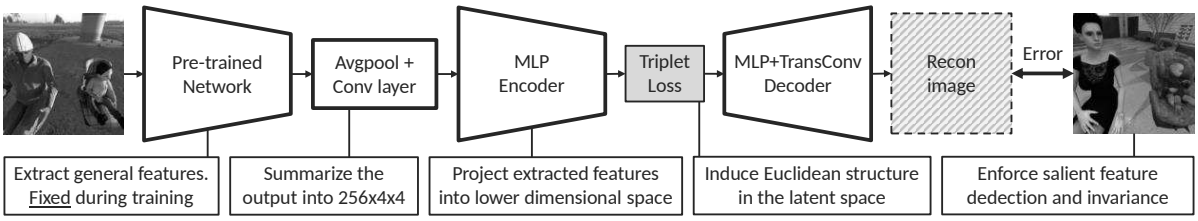


Fig. 4.6 Impossible Instance Extractor Triplet Autoencoder (II-E-TAE) model architecture.

4.5 Model architecture: Extractor autoencoder

Additionally to the proposed variations of the PIRL, we will also highlight the benefit of using a feature extractor in case of generalizing from synthetic to real images. This model architecture can be combined with the PIRL or be used as a standalone design choice. We propose to apply ideas from transfer learning and the perceptual loss, see Section 2.4.3, by using a pre-trained classification model to extract more general features from the input images. Instead of using the images itself, the extracted features are used as input. Our autoencoder consists of a summarization module (average pooling and convolutional layers) which reduces the number of convolutional filters. This is fed to a simple MLP encoder which is then decoded by a transposed convolutional network. We refer to this model as *extractor autoencoder* (E-AE). Let e_ϕ be the encoder, d_θ the decoder and $\mathcal{E}_{fix}$ be a pre-trained classification model, referred to as *extractor*. For ease of notation, we define $\mathrm{e}\mathcal{E}_\phi(\cdot) := \mathrm{e}_\phi(\mathcal{E}_{fix}(\cdot))$. The model, using the vanilla reconstruction loss, can be formulated for a single input sample as

$$\mathcal{L}_R(X_i^j, f(X_i^j); \theta, \phi) = \mathcal{L}_R\left(X_i^j, \mathrm{d}_\theta(\mathrm{e}_\phi(\mathcal{E}_{fix}(X_i^j))); \theta, \phi\right) \tag{4.5}$$

$$= \mathrm{r}\left(X_i^j, \mathrm{d}_\theta(\mathrm{e}_\phi(\mathcal{E}_{fix}(X_i^j)))\right) \tag{4.6}$$

$$= \mathrm{r}\left(X_i^j, \mathrm{d}_\theta(\mathrm{e}\mathcal{E}_\phi(X_i^j))\right). \tag{4.7}$$

This loss can easily be combined with both PIRL variations and with the triplet loss. This design choice combined with the II-PIRL is visualized in Fig. 4.6.

4.6 Model architecture: Uncertainty estimation

Instead of training the autoencoder model under a standard training regime, we train the model using dropout and enable dropout during inference as well, i.e. using MC Dropout as introduced in Section 2.9. Hence, we obtain different, but similar, autoencoder models for inference which should behave similarly for training and test samples, but differently and not consistently for novel feature variations in the input space. Let us formulate this intuition more precisely: Let

x be an input sample and $\mathcal{F}$ be the family of functions consisting of the autoencoder models learned by using dropout during training and enabling the latter during inference. We repeat inference M times, sampling each time a new $f_j \in \mathcal{F}$ for $j \in \mathcal{J} = \{1,2,\ldots,M\}$. This results in a predictive distribution $\{f_j(x)\}_{j\in\mathcal{J}}$, i.e. M realizations with different dropout masks. Since we are interested in the variation of the latent space representation, we refrain from using dropout in the latent space. Finally, this approach can easily be extended using, for example, the II-PIRL. We refer to the latter as MC-II-AE and the former as MC-AE. Particularly in the case when the II-PIRL is used, uncertainty estimation and OOD detection can be significantly improved.

4.7 Model architecture: Multi-channel

The multi-channel approach for autoencoder models (MuCh-AE) tries to exploit the availability of real and synthetic image pairs during training [277] to reduce the synthetic gap and learn more efficiently from synthetic data. Instead of a single decoder module, the MuCh-AE uses two decoders - one for synthetic d_{θ_s} and one for real d_{θ_r} input images. There is, however, only one encoder e_ϕ which is used for all input images. For each synthetic input image $X_i^j := X_{i,s}^j$ being sampled, we randomly sample a real image $X_{k,r}^l$ being of the same class as $X_{i,s}^j$. Since the datasets do not need to be of the same size and since the sampling is random, it can happen that a same real image is sampled multiple times per epoch. The MuCh-AE encodes both input samples using the same encoder e_ϕ to get two latent vectors z_r and z_s. Next, the decoder for real images d_{θ_r} reconstructs the encoding of $X_{k,r}^l$ and the one for synthetic images d_{θ_s} the one of $X_{i,s}^j$. In summary, we have $f_r(X_{k,r}^l) = \mathrm{d}_{\theta_r}(\mathrm{e}_\phi(X_{k,r}^l)) = \mathrm{d}_{\theta_r}(z_r)$ and $f_s(X_{i,s}^j) = \mathrm{d}_{\theta_s}(\mathrm{e}_\phi(X_{i,s}^j)) = \mathrm{d}_{\theta_s}(z_s)$. Formalizing the reconstruction error leads to

$$\begin{aligned}
&\mathcal{L}_R(X_{i,s}^j, f_s(X_{i,s}^j), X_{k,r}^l, f_r(X_{k,r}^l); \theta_s, \theta_r, \phi) && (4.8)\\
&\quad := \mathcal{L}_R(X_{i,s}^j, f_s(X_{i,s}^j); \theta_s, \phi) + \mathcal{L}_R(X_{k,r}^l, f_r(X_{k,r}^l); \theta_r, \phi) && (4.9)\\
&\quad = \mathcal{L}_R(X_{i,s}^j, \mathrm{d}_{\theta_s}(\mathrm{e}_\phi(X_{i,s}^j))); \theta_s, \phi) + \mathcal{L}_R(X_{k,r}^l, \mathrm{d}_{\theta_r}(\mathrm{e}_\phi(X_{k,r}^l))); \theta_r, \phi). && (4.10)
\end{aligned}$$

However, related work [277] thus far reports that it is beneficial to balance both reconstructions by adding a penalizing term leading to a total error of

$$\begin{aligned}
&\mathcal{L}_T(X_{i,s}^j, f_s(X_{i,s}^j), X_{k,r}^l, f_r(X_{k,r}^l); \theta_s, \theta_r, \phi) && (4.11)\\
&\quad = \mathcal{L}_R(X_{i,s}^j, f_s(X_{i,s}^j); \theta_s, \phi) + \mathcal{L}_R(X_{k,r}^l, f_r(X_{k,r}^l); \theta_r, \phi) && (4.12)\\
&\qquad + \gamma \frac{1}{2} \left(\mathcal{L}_R(X_{i,s}^j, f_s(X_{i,s}^j); \theta_s, \phi) - \mathcal{L}_R(X_{k,r}^l, f_r(X_{k,r}^l); \theta_r, \phi) \right)^2, && (4.13)
\end{aligned}$$

with $\gamma \geq 0$. The latter should ensure that both decoders achieve a more or less equal performance such that not one decoder is favoured while neglecting the second one. However, in our experiments we found that the balance term negatively influences the performance. Hence, in this work we choose $\gamma = 0$. In practice, it is beneficial to choose a same target for both reconstructions [277]. In our case, we decided to reconstruct synthetic images only, such that the previous reconstruction losses become

$$\mathcal{L}_R(X^j_{i,s}, f_s(X^j_{i,s}); \theta_s, \phi) \qquad \text{and} \qquad \mathcal{L}_R(X^l_{k,r}, f_r(X^j_{i,s}); \theta_r, \phi). \tag{4.14}$$

Due to the latter choice, we can exploit the availability of synthetic images with random backgrounds such that invariances can be learned using both variations of the PIRL. We will exploit this model design choice later in this work.

While related work has adopted this method successfully to roof classification of satellite images and face-sketch recognition, our task is more challenging. In both aforementioned applications, the method exploits the existence of one-to-one real-synthetic image pairs. However, in our case no such pairs exist since real and synthetic images have been generated independently. This means that for each synthetic image for each epoch a new real image of the same class is randomly selected. While this makes the application of the MuCh method more challenging, we will show that combining it with the extractor and II-PIRL improves the performance significantly. Further, when adopting the triplet loss with the MuCh, we made sure to use the latent space representation of the synthetic and real data interchangeably. This enforces a similar representation for both distributions and improves the performance.

4.8 Inference strategy: Attractor autoencoder

This method consists of the inference strategy of applying a previously trained autoencoder model recursively to itself several times. However, in contrast to Radhakrishnan et al. [204] where the authors trained the autoencoder model under a standard training regime, we train the model using dropout and enabling dropout during inference as well, i.e. using MC Dropout as presented in the previous section. This causes an interesting model feature: if we repeat the recursive application of the trained autoencoder model several times for the same input sample x, then each iteration uses a different function f_j from the same distributions of functions $\mathcal{F}$. Hence, we obtain different, but similar, dynamical systems for inference which should behave similarly for training and test samples, but differently and not consistently for novel feature variations in the input space. Each iteration can hence potentially converge to a different attractor, potentially of different classes. The latter is useful to detect inconsistencies and hence

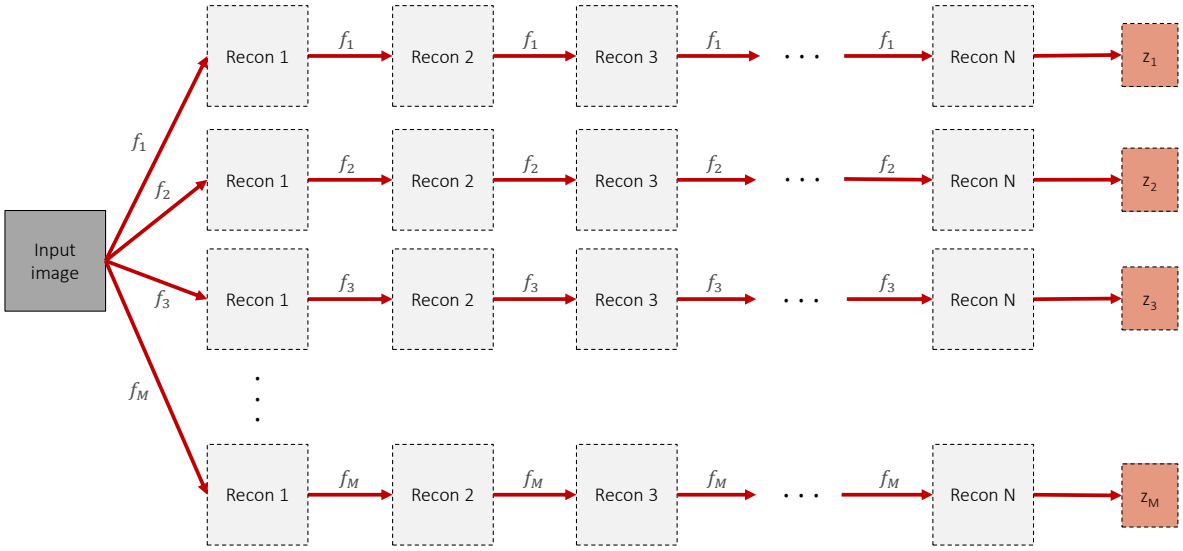


Fig. 4.7 Visualization of MCA-AE. For each of the M samples of autoencoder functions $f_j \in \mathcal{F}$ for $j \in \mathcal{J} = \{1,2,\cdots,M\}$, the recursion is repeated N times. Coherence and inconsistencies in the resulting latent space vectors z_j for $j \in \mathcal{J} = \{1,2,\cdots,M\}$ is used to assess the model's reliability for the input image. Each z_j is the encoding used to generate the nth reconstruction.

uncertainty: if the model converges to attractors of the same class we can assume a trustful prediction, if it converges to attractors of different classes the convergence is unstable and unreliable.

Let x be an input sample and $\mathcal{F}$ be the family of functions consisting of the autoencoder models learned by using dropout during training and enabling the latter during inference. We repeat the recursion M times, sampling each time a new f_j for each recursion $\mathcal{J} = \{1,2,\cdots,M\}$. Up to this point, the method is identical to the one presented in the previous section. However, for a fixed function f_j, the latter is applied recursively k times to itself, i.e. $f^k(x)$ as in Eq. (2.49). This results in a predictive distribution $\{f_j^k(x)\}_{j\in\mathcal{J}}$, where k is the number of compositions performed for each recursion. As a reminder, for a fixed f_j the dropout mask is the same for each recursive step $i \in \{1,\ldots,k\}$. The latter implies that the dropout mask needs to be implemented manually such that it can be fixed for multiple inferences. Since we are adopting this strategy for autoencoder models, we refrain from using a dropout in the latent space. To perform a classification of the resulting iteratively reconstructed sample, we perform the classification in the latent space of the kth iteration. This heuristic is summarized in Algorithm 1 and visualized in Fig. 4.7.

Algorithm 1 MCA-AE algorithm

Train autoencoder model using dropout to get $\mathcal{F}$
Enable dropout for inference
Define the number of recursions N
Train classifier $g(\cdot)$ in latent space after N recursions
Define the number of inferences per sample M
Define uncertainty threshold U
for each input sample x **do**
  for $j \leftarrow 1$ to M **do**
    for $k \leftarrow 1$ to N **do**
      if $k = 1$ **then**
        Sample a new dropout mask and keep it fixed
        This gives you $f_j \in \mathcal{F}$, where $f_j(x) = d_j(e_j(x))$
      end if
      $z = e_j(x)$ {encoding}
      $x = d_j(z)$ {decoding}
    end for
    $y_j = g(z)$ {probability distribution of classification}
  end for
  $p(y) = \frac{1}{M}\sum_{j=1}^{M} y_j$
  $H(p(y)) = -\frac{1}{\log(C)}\sum_{c=1}^{C} p_c(y)\log(p_c(y))$
  if $H(p(y)) \leq U$ **then**
    $y = \text{argmax}(y_j)$ {class prediction}
  else
    Reject sample
  end if
end for

Since our method is based on both aforementioned design choices, we call it ***Monte Carlo Attractor Autoencoder (MCA-AE)***. For training samples to become attractors it is necessary to train the autoencoders for a large number of epochs, i.e. we used 25000. A benefit of this approach is that a lot of hyperparameters are defined for the inference process instead of the training process. The number of recursions and the number of different runs is independent from the training. The classifier can be chosen after the autoencoder model training and the uncertainty threshold can also be fine-tuned.

4.9 Conclusion

We introduced several novel, or novel variations of existing, design choices, training approaches as well as sampling and inference strategies for autoencoder models in a common framework. The different approaches will be adopted in the upcoming chapters using the dataset splits presented in Chapter 3, as well as other commonly used academic datasets. The I-PIRL will be used for the illumination normalization presented in Chapter 7, but we will also highlight that it has some beneficial effect for the transfer of synthetic to real images. The latter challenge, together with the II-PIRL and the extractor module will be reported in Chapter 8. These investigations will be corroborated by comparing them against the multi-channel autoencoder approach, for which we also show that it can be used to remove the background from real images. The II-PIRL will also be evaluated on illumination normalization, but its main additional benefit is highlighted on commonly used academic datasets, as shown in Chapter 5, and on uncertainty estimation and out-of-distribution detection, as shown in Chapter 9. Lastly, the attractor autoencoder is going to be discussed exhaustively in Chapter 9 as well. Most of the aforementioned methods can, and will be considered with and without the triplet loss.

Chapter 5

Basic analyses and baseline results

In the first part of this chapter, we start with a basic analysis on datasets commonly used by the research community. This should build an entry point to get familiar with the second variation of the newly introduced loss function from Chapter 4 and to understand its effect on well known datasets and learning tasks, also in comparison to other methods. We trained vanilla autoencoders (AE), autoencoders with the triplet loss (TAE) and autoencoders with the II-PIRL (II-AE) on MNIST, Fashion-MNIST , CIFAR10 and GTSRB. We then compare the test accuracy, reconstructions and latent space representations of the different models.

In the second part of this chapter, we present baseline results for training and evaluating on SVIRO for different tasks. These results show that the investigated application in this work is indeed challenging with interesting questions to be explored. Further, we show that training on SVIRO and applying the resulting model on ORSS is possible, such that insights made on SVIRO are transferable to the real application as well. Lastly, we report baseline results on the test sets, but also for transferring in-between vehicles, on ORSS to highlight that the previously mentioned challenges are indeed valid.

5.1 Commonly used datasets

For training on MNIST and Fashion-MNIST we used a latent space dimension of 16, while for training on CIFAR10 and GTSRB we used a latent space dimension of 64. We treated all datasets (even RGB ones) as grayscale images and all images were center-cropped and resized to 64 by 64 pixels. We used a batch size of 64, trained our models for 1000 epochs and did not perform any data augmentation. Classification is performed by means of a linear SVM classifier in the latent space, which was trained after the autoencoder model finished training. The latter gives hints about the quality and separability of the learned latent space representation and hence feature extraction.

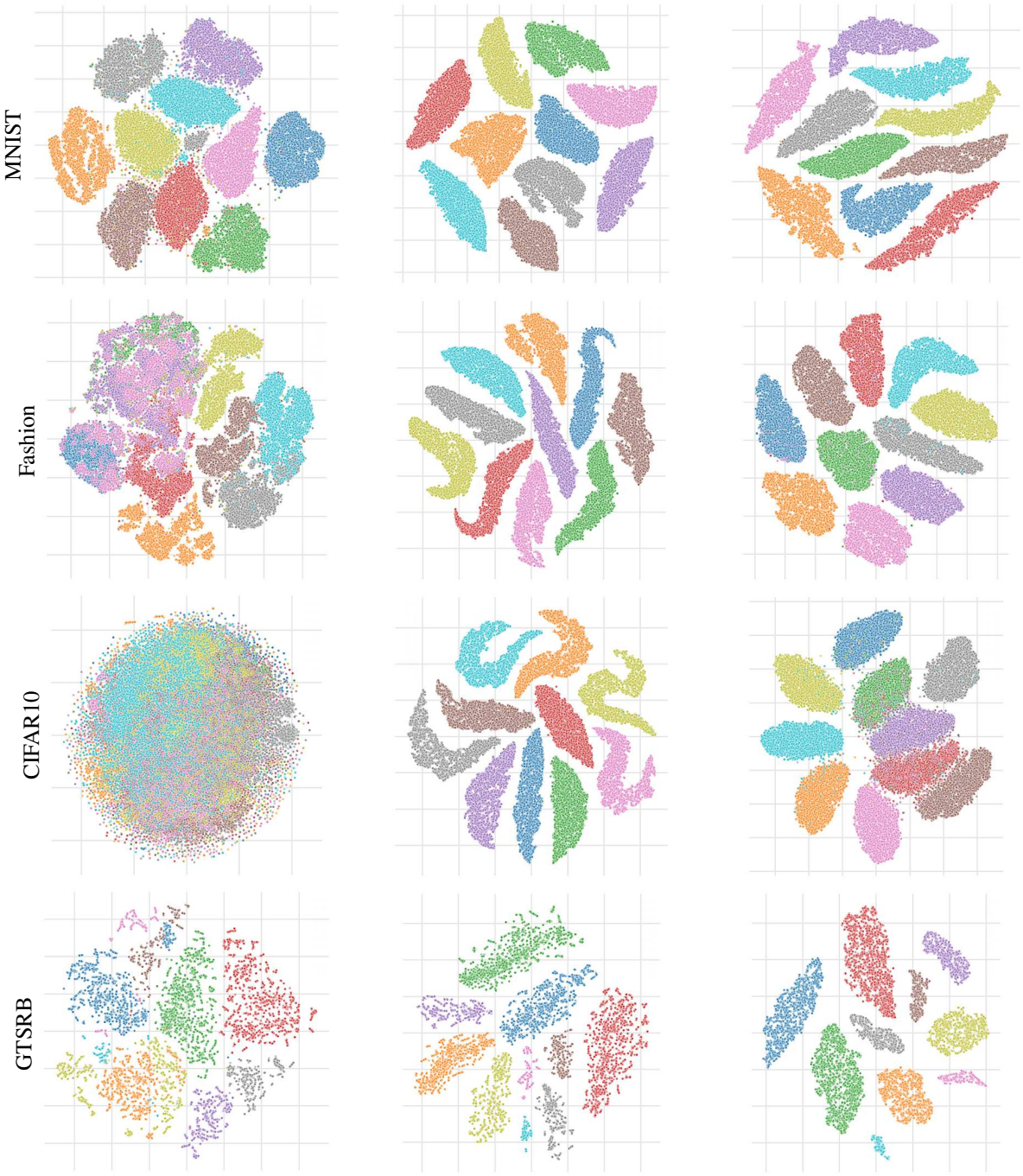
Table 5.1 Mean and standard deviation test accuracy over 10 runs by a linear SVM classifier trained in the latent space of different autoencoder models after the latter finished training.

Dataset	II-AE (Ours)	TAE	AE
MNIST	99.3 ± 0.1	99.1 ± 0.1	93.2 ± 0.5
Fashion-MNIST	91.9 ± 0.2	90.7 ± 0.3	78.9 ± 0.3
CIFAR10	65.9 ± 0.6	60.4 ± 1.4	18.6 ± 2.0
GTSRB	98.9 ± 0.4	98.8 ± 0.3	95.7 ± 0.5

In Fig. 5.1 we compare the t-SNE projection of the training data latent space representation for the different autoencoder models for different datasets. It can be observed that the II-PIRL produces very clear clusters for the different classes. This is an interesting phenomenon, since the label information is only used implicitly in the pixel space and not explicitly in the latent space as done by the triplet loss. In order to assess the representation quality quantitatively, we consider the test accuracy performance by a linear SVM classifier, when trained on the aforementioned latent spaces. The results are reported in Table 5.1. The linear classifier can exploit the latent space representation obtained by the II-PIRL at least as good as if the triplet loss is used: the test accuracy is slightly better. Finally, to get an idea about the input-target pairs used for training and the resulting reconstruction by the different methods, we report in Fig. 5.2 reconstructions of the training data after the model finished training. It can be observed that the II-AE removes all unnecessary background information (see GTSRB) and learns a mean class representation. However, the model is not able to learn a meaningful class representation for CIFAR10, due to its large intra-class variability. Nevertheless, as shown by the latent space representation and quantitative results, the model can still learn a somewhat meaningful separation in the latent space and achieve a better test accuracy compared to using the triplet loss.

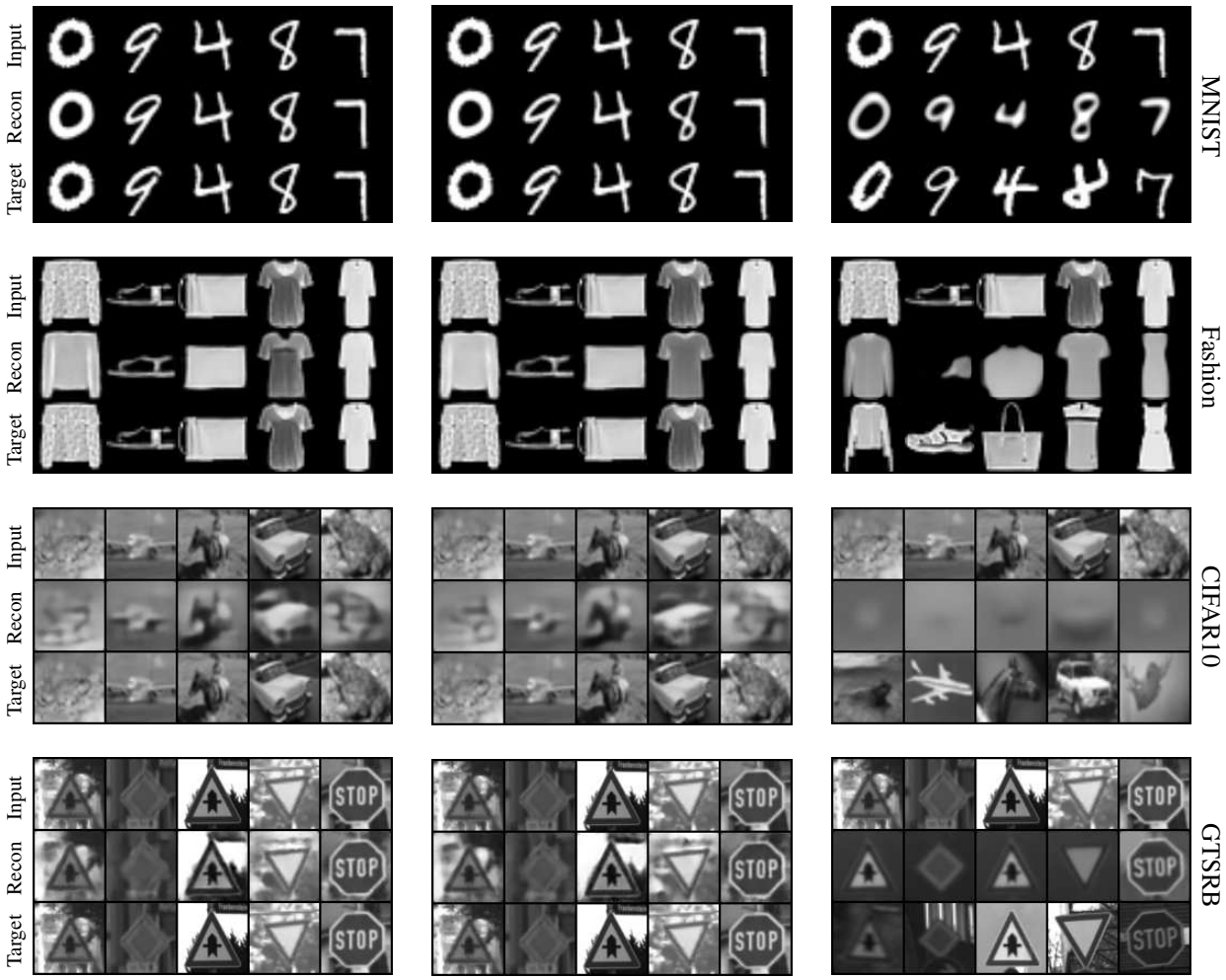
Lastly, we compared two properties in the latent space of the different autoencoder models: nearest neighbor retrieval and interpolation between training samples. For the encoding of several training input images, we searched in the latent space for the ten nearest neighbors and decoded them. The results in Fig. 5.3 show that the neighbors are decoded identically in case the II-PIRL is used. For the AE it can be observed that the nearest neighbors are not as robust as for the other model variations: there are neighbors being of a different class in the second and fourth columns. For the interpolation, we encoded two training input samples and decoded each of the intermediate latent space representations. Let x_1 and x_2 bet two training samples and $z_1 = \mathrm{e}_\phi(x_1)$ and $z_2 = \mathrm{e}_\phi(x_2)$ be their encodings. The linear interpolation is then defined by

$$z_i = (1 - \alpha_i) z_1 + \alpha_i z_2, \tag{5.1}$$



(a) AE (b) TAE (c) II-AE (Ours)

Fig. 5.1 Comparison of training latent space representations (t-SNE projection) by different autoencoder models (AE, TAE and II-AE) for different datasets: MNIST (first row), Fashion-MNIST (second row), CIFAR10 (third row) and GTSRB (fourth row). Different colors represent different classes.



(a) AE (b) TAE (c) II-AE (Ours)

Fig. 5.2 Comparison of the input (first rows), reconstruction (second rows) and target (third rows) of the training data after the last epoch. The results are from different autoencoder models (AE, TAE and II-AE) for different datasets: MNIST (first block), Fashion-MNIST (second block), CIFAR10 (third block) and GTSRB (fourth block).

for $0 \leq \alpha_i \leq 1$. We chose $\alpha_i \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and reconstructed each z_i. The results are reported in Fig. 5.4. For II-PIRL, the background for each intermediate value is being removed. This highlights the robustness of the background removal and image normalization obtained by adopting the II-PIRL during training. Overall, the smoothness of the transitioning between samples and the reconstruction quality is similar for all model designs. Notice that we did not enforce any interpolation properties during training. Nevertheless, the latter has been learned, to some extend, as a by-product of the autoencoder bottleneck structure which should induce desirable properties in case a meaningful feature extraction is learned.

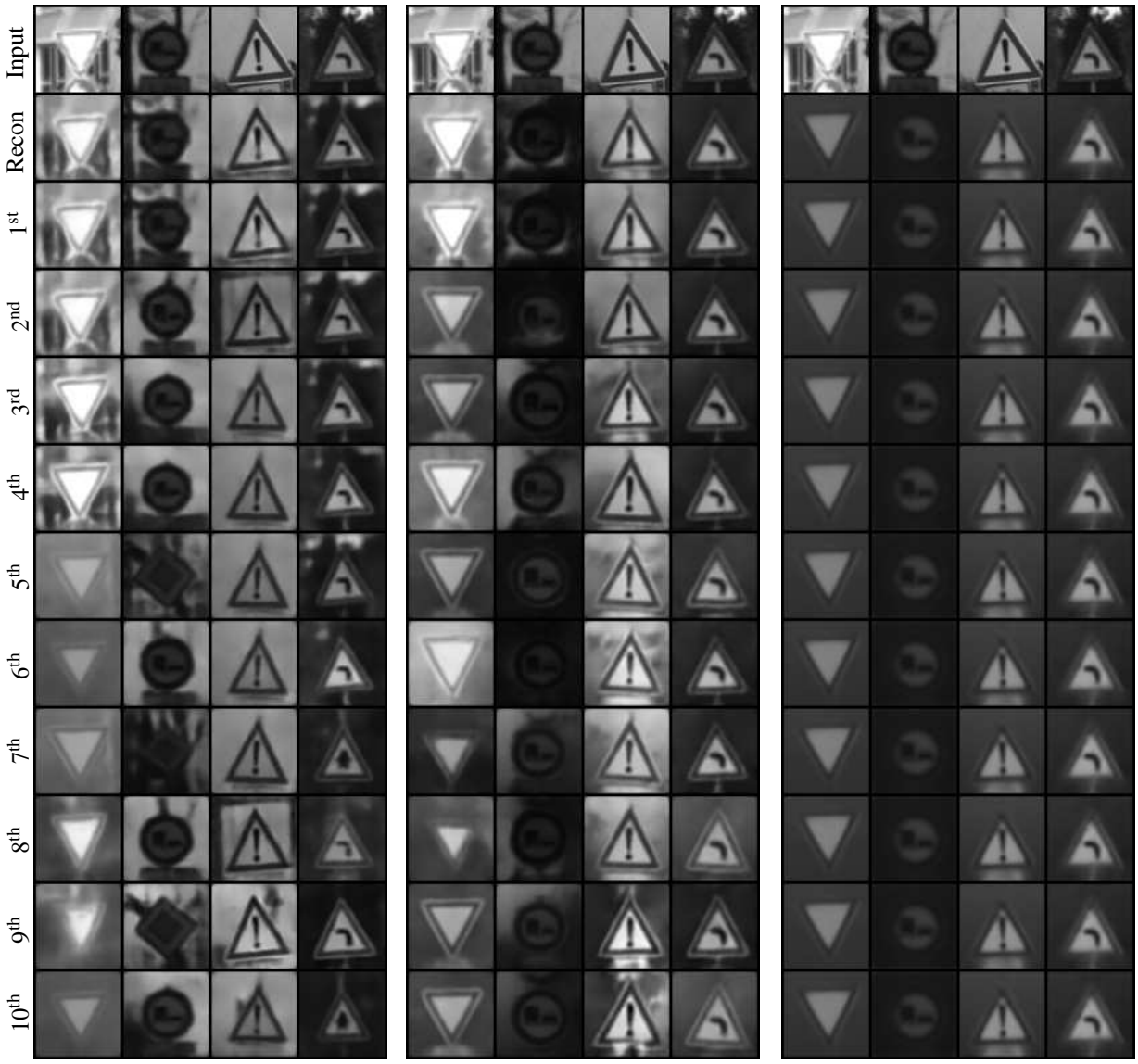
It is important and interesting to notice the following observation: The triplet loss needs for each batch instance three samples - the input image, a positive and a negative sample. On the other side, our PIRL only needs two samples per instance - the input image and a positive sample. Although not having a negative sample at its disposal, using the II-PIRL causes better results than using the triplet loss. Further, we believe that the II-PIRL can potentially be improved by incorporating more information, and hence constraints or regularizations, in its cost function formulation.

5.1.1 Conclusion

The baseline results presented in this section show that the II-PIRL is a beneficial design choice for several commonly used research datasets. Not only do linear classifiers achieve better classification results in the latent space compared to the triplet loss, but it also induces well separated clusters. Our loss formulation causes the reconstructions to be smoother and unimportant information, like background, is removed. The latter is even the case for all the interpolation values between training samples. Although the reconstruction might not always be as good, or even unrecognizable, this does neither have detrimental effects on the classification performance nor on the clustering.

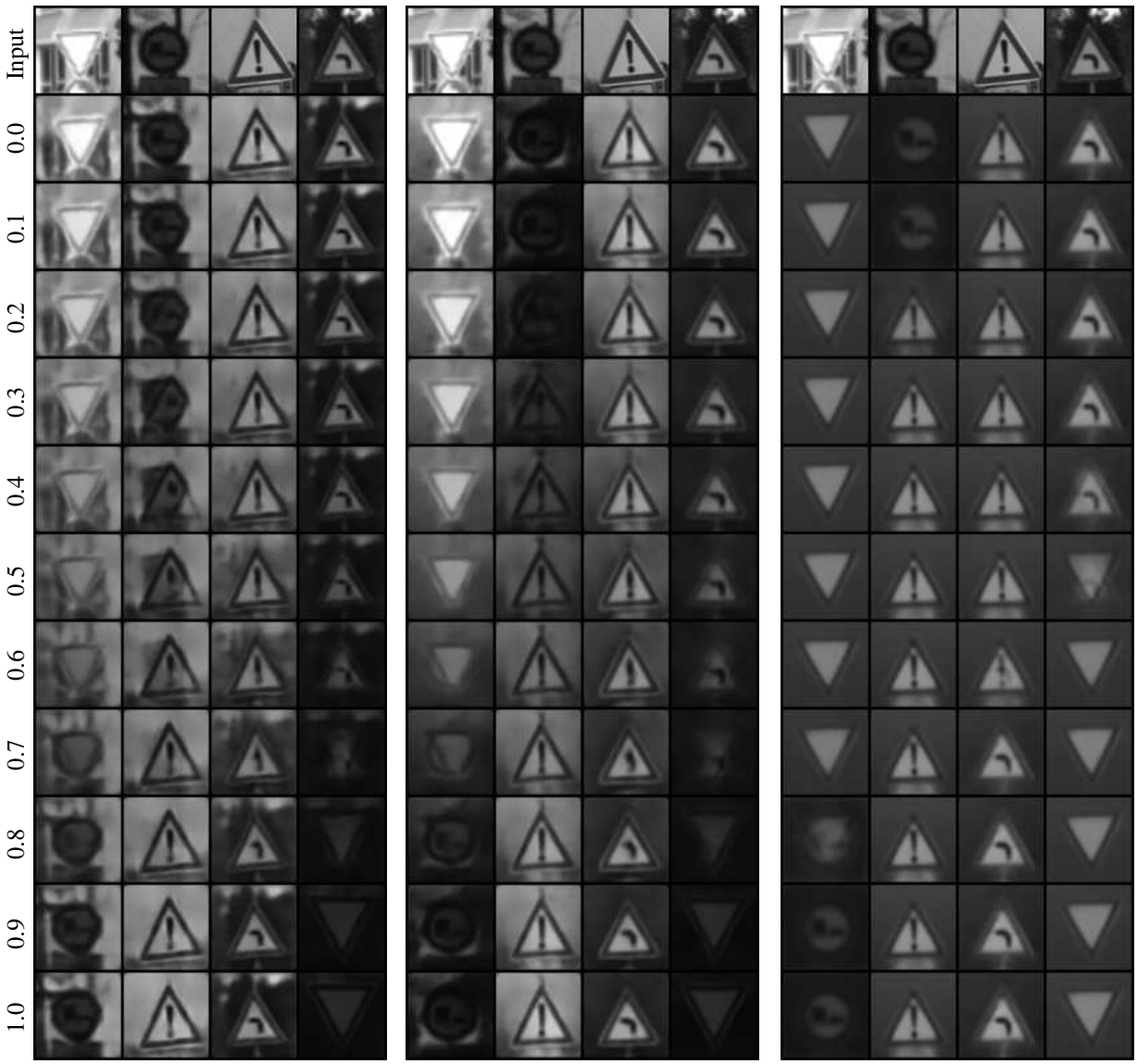
5.2 SVIRO

In the baseline evaluation presented in this section, we will show that SVIRO provides the means to analyze the performance of common machine learning methods under new conditions. We tested some widely used models and approaches for their robustness and reliability, when trained on limited number of variations only. Specifically, we will show that state-of-the-art models cannot generalize well to new environments and textures when trained under these conditions. For this first evaluation, we limited ourselves to training on the X5 (three seats) vehicle and testing on the Tucson (three seats) and i3 (two seats) vehicle. For all tasks, we



(a) AE (b) TAE (c) II-AE

Fig. 5.3 Reconstruction of the ten nearest neighbors (rows three to twelve) of the encoded training input images (first row) together with the reconstructions of the encodings (second row). Different autoencoder models are compared against each other. In case the II-PIRL is used, all the neighbors are reconstructed equally. In case of the AE some of the neighbors are of a different, wrong class.



(a) AE (b) TAE (c) II-AE

Fig. 5.4 Reconstructions of the interpolations (rows two to twelve) from the latent space representation of the current training input (first row) and the training input of the next column for a step size of 0.1. Different autoencoder models are compared against each other. It can be observed that all intermediate values also remove all background information in case the II-PIRL is used. The smoothness of the transition and the quality of the reconstruction are similar across all model designs.

considered two training data versions, for which we used the exact same hyperparameters: 1) the standard X5 training data with fixed textures and backgrounds (F), 2) half of the standard X5 training data was replaced by randomly textured X5 training data with random backgrounds (F&R). The latter dataset and training design choice is supported by recent results by Li et al. [144].

We used the grayscale images (infrared imitation) for all evaluations. For the deep learning-based approaches, we used the pre-defined models implemented in PyTorch and torchvision. For classification, we used pre-trained models on ImageNet. For semantic and instance segmentation, object detection and keypoint detection, the models were pre-trained on COCO train 2017. The pre-trained models were fine-tuned on the X5 only and then evaluated on the test sets of all three cars. Using this approach, we could test the generalization capacities on two difficulty levels. The training dataset was partitioned randomly according to a 75:25 split for training and evaluation, where the latter was used to perform early stopping when fine-tuning the models. As we considered our F&R dataset as data augmentation, the only additional data augmentation performed was a random horizontal flip to increase the number of sceneries.

5.2.1 Classification

As introduced in Section 3.2.5, we used the rectangular grayscale images for classification with seven different classes. One could decide to classify a seat with an everyday object (and an empty infant/child seat) as empty as well. We trained a single classifier for the three seats, but other setups are possible as well, e.g. train one classifier for each seat or one classifier using the whole image as input, as reported later in this work. In the following, we will report results on different deep learning models, as they are commonly used for visual classification problems. These results will be compared against a traditional method using a support vector machine (SVM) and handcrafted features. We will show that both methods suffer from the same problems and including the randomized F&R dataset overall improves the results.

Convolutional neural networks

We used the ResNet [93], DenseNet [105], SqueezeNet V1.1 [108] and MobileNet V2 [219] architectures and considered four different training approaches: 1) Training from scratch, 2) only fine-tuning the last fully connected layer, 3) additionally fine-tuning the last residual block, 4) allowing all weights to be trainable. We tried different combinations of weight decay, weighted costs and imbalanced sampling and report results for the best models only. In Table 5.2, we compare the results across the different models and training approaches and against the SVM. The deep learning-based approaches have problems to generalize to the test

set, especially for new cars. The randomized backgrounds and textures help to improve the accuracy on the same car, which gives hint that models trained on the (F) dataset mostly use the texture as a classification criterion. However, the models can still not generalize well to new vehicle interiors, probably because of the different interior structures, see Fig. 3.8.

HOG and SVM

For comparison, we also wanted to test at least one *traditional* machine learning-based approach for the classification task. To this end, we computed the histogram of oriented gradients (HOG) [87] features of all the training images, and their horizontally flipped versions for data augmentation. These features were then used to train a SVM, using the *one vs. rest* [269] approach and balanced class weights. We performed a grid search on different kernels (linear, polynomial and radial basis) and their hyperparameters and used a 5-fold cross validation for hyperparameter selection. We used scikit-learn for the training and scikit-image for the feature generation. The results for the best hyperparameters are reported in Fig. 5.2. Overall, the traditional approach has similar problems as the deep learning approach when the standard X5 data is used, and can sometimes even generalize better. However, it cannot exploit the additional information when random textures and backgrounds are included in the training.

5.2.2 Semantic segmentation

It could be beneficial to take spatial information into account to improve the transfer to new instances and environments. Further, the model might consider overlapping objects from neighboring seats more efficiently when the entire scene is used. To this end, we evaluated semantic segmentation and considered the five classes as introduced in Section 3.2.5. The model should separate the child from the child seat and the baby from the infant seat and classify them as person. We fine-tuned all layers of a Fully Convolutional Network (FCN) [151] with a ResNet-101 backbone and report the results in Fig. 5.3. As for the classification results of the previous section, the model's performance decreases drastically on the child and infant seats on the test set for the same car and it performs even worse in previously unknown cars. Using the F&R training data, the generalization performance largely increases, although the geometry of the child seats of the test sets was never observed during training. It seems that the texture has a larger influence on the performance of classification and semantic segmentation models than the geometry. This observation seems to be in line with recent results by Geirhos et al. [77]. However, using SVIRO, we can additionally show that the model cannot perform as good on new environments, even though the textures are randomized and the objects of the different test sets are the same.

Table 5.2 Comparison of classification accuracies (in percentage). We trained several models from scratch or fine-tuned pre-trained models, where all the layers, the last block or the last layer were trainable. Further, we trained a SVM using HOG features. We used the X5 training data (F) or replaced half of it with the randomized data (F&R). After training, we retained models with the best total accuracy on the X5 test data and evaluate them on the i3 and Tucson test data. The models have difficulties to generalize to the test data and perform even worse in unknown vehicles. Including the randomized data helps to generalize to unseen objects.

	Scratch		All layers		Last layer		Last block	
	F	F&R	F	F&R	F	F&R	F	F&R
ResNet-18	69	78	75	88	70	73	76	88
DenseNet-121	64	77	81	85	69	78	77	92
SqueezeNet	67	45	80	85	62	69	73	83
MobileNet	66	72	82	88	64	74	71	86
HOG + SVM	69	70	-	-	-	-	-	-

(a) X5

	Scratch		All layers		Last layer		Last block	
	F	F&R	F	F&R	F	F&R	F	F&R
ResNet-18	31	34	44	58	50	60	42	64
DenseNet-121	32	43	50	60	51	71	43	76
SqueezeNet	37	28	50	54	45	63	48	65
MobileNet	36	31	57	70	47	61	40	71
HOG + SVM	41	52	-	-	-	-	-	-

(b) i3

	Scratch		All layers		Last layer		Last block	
	F	F&R	F	F&R	F	F&R	F	F&R
ResNet-18	20	31	31	46	44	55	34	47
DenseNet-121	14	32	39	75	35	48	50	60
SqueezeNet	22	44	32	44	28	33	46	37
MobileNet	21	43	48	71	32	44	33	51
HOG + SVM	35	53	-	-	-	-	-	-

(c) Tucson

Table 5.3 Mean intersection over union (IoU), in percent, for semantic segmentation for a fine-tuned pre-trained FCN. (F) represents the model performance when trained on the X5 training dataset and (F&R) when we included the random X5 data. The models were evaluated on the test dataset for the X5, Tucson and i3. Using the randomized version largely improves the generalization capacities of the model, especially for identifying infant seats and child seats.

	Background		Infant seat		Child seat		Person		Object		Total	
	F	F&R	F	F&R	F	F&R	F	F&R	F	F&R	F	F&R
X5	95.6	96.5	29.6	82.0	63.3	80.2	80.6	88.2	14.2	61.0	56.7	81.6
Tucson	82.1	88.5	16.5	77.9	55.9	74.5	68.0	79.2	5.2	15.3	45.6	67.1
i3	89.9	93.6	14.5	70.0	47.1	58.2	77.3	86.5	12.9	25.2	48.3	66.7

5.2.3 Object detection

We also wanted to evaluate an object detection based approach, as they are commonly used to detect and classify objects in diverse surroundings. An object detection model might generalize better to new environments than standard classification models. The reason for the latter is that such models either first detect the objects of interest and then perform the classification or perform detection and classification simultaneously. We considered the same classes as for semantic segmentation. For this task, we used the Faster R-CNN [213] model architecture with a ResNet-50 backbone and fine-tuned all layers of the model. We used the COCO evaluation metrics and report the results in Table 5.4. Focusing on the primary challenge metric for COCO, i.e. mean average precision (mAP) on intersection over union (IoU) 0.50:0.05:0.95, the results are similar across the different test sets and are slightly improved when we included the randomly textured X5 images (F&R). A difference is noticeable on the PASCAL VOC metric, i.e. mAP on IoU 0.50, but in general the performance drop is not as drastic as for classification and semantic segmentation. Overall, the object detection model seems to be better suited for the transfer between different vehicles.

5.2.4 Instance segmentation

For instance segmentation, we used the same classes as for the semantic segmentation and used the mask mAP COCO evaluation metrics. We fine-tuned the Mask R-CNN [94] model with a ResNet-50 backbone and considered the same training approach as for object detection. The results are summarized in Table 5.4. For this setting, including the randomly textured X5 data improves the generalization and model performance across all IoU evaluations.

Table 5.4 We report the average precision (AP), in percent, for different tasks and two different training approaches: Standard X5 dataset (F) and the combination with randomized textures (F&R). According to the COCO evaluation standard, the trained model is evaluated on the different test sets of the different cars for different intersection over union (IoU) and object keypoint similarities (OKS). Top rows: Object detection (mAP). Middle rows: Instance segmentation (mask mAP). Bottom rows: Keypoint estimation (OKS). Columns: Different IoU and OKS metric evaluations

X5			Tucson			i3			Dataset
0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	
43.1	79.9	38.8	37.9	71.7	33.2	38.6	77.2	31.6	F
41.3	77.4	36.1	40.8	71.8	40.8	43.8	80.4	44.8	F&R

(a) Object detection

X5			Tucson			i3			Dataset
0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	
32.6	59.0	32.5	22.0	49.2	15.7	27.5	51.0	27.0	F
44.9	73.5	49.1	34.3	64.3	31.0	46.9	74.6	52.2	F&R

(b) Instance segmentation

X5			Tucson			i3			Dataset
0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	0.50:0.95	0.50	0.75	
36.0	84.7	12.3	30.6	81.7	5.8	37.9	86.7	17.8	F
35.9	84.7	11.0	31.3	83.4	5.5	37.0	86.9	18.1	F&R

(c) Keypoint detection

5.2.5 Keypoints for human pose estimation

We retained only two classes for the keypoint estimation for human poses. Either we have a person (infant in an infant seat, child on a child seat or an adult), or we treat it as background. We considered a subset of 17 different keypoints, although more are available, e.g. finger joints. We used the Keypoint R-CNN [94] with a ResNet-50 backbone and considered the same training approach as for object detection. Again, we use the standard COCO evaluation metric and report the average precision over different object keypoint similarities (OKS). Including randomly textured images did not improve the results in contrast to Section 5.2.4.

5.2.6 Comparison with real images

We tested the transferability of a model trained on SVIRO to real infrared images and report results for instance segmentation to illustrate this. We fine-tuned all layers of a pre-trained Mask R-CNN model with a ResNet-50 backbone and considered the same classes as for semantic segmentation. We combined the training images of the i3, Tucson and Model 3 and compare results on synthetic and real images in the X5 in Fig. 5.5. We reproduced the real ORSS images in Blender such that the performance can be compared qualitatively between similar real and synthetic sceneries. Only bounding boxes and masks with a confidence of at least 0.5 are plotted. The model performs similarly across real and synthetic images and sometimes fails to detect objects. This is expected as the model has only seen a limited amount of variation. However, the similar child seat is detected in the real images, but not in the synthetic ones. We believe that investigations on SVIRO are transferable to real applications as the resulting model behaves similarly on real and synthetic images. Additional realistic effects could be applied to close the synthetic gap even further [142]. Since only class labels are available for ORSS, we could not evaluate the model performance quantitatively.

In Fig. 5.6 we provide additional examples of the instance segmentation mask predictions. Some of the infant seats (red) with sun-protection are classified as an everyday object (light blue), however, this is understandable as we did not have any infant seats with sun-protection in the training data. Nevertheless, as some of those infant seats with sun-protection are classified correctly, we believe that this is in favor of our assumption that insights on SVIRO are transferable to real images. As the model was trained on a very limited amount of variation, some objects are not detected at all. We believe this will be improved once generalization on SVIRO is enhanced as well.

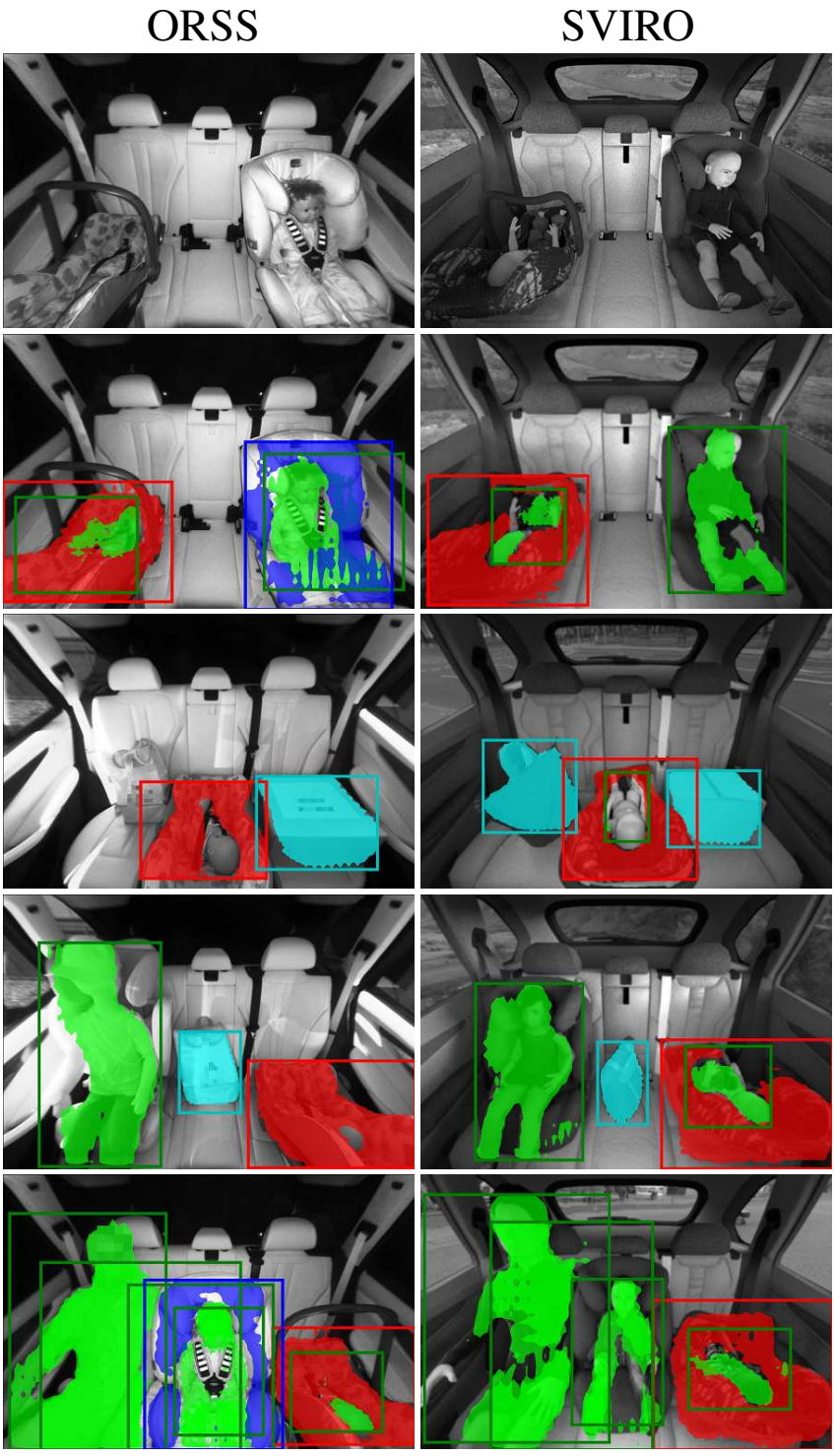


Fig. 5.5 We compare results for similar scenes from real active infrared images (first column) in an X5 and reproduced the same sceneries in Blender (second column). The first row compares real and synthetic images. The remaining rows compare instance segmentation mask predictions. The model performs similarly on both setups and the similar child seat is detected in the real images, but not in the synthetic ones.

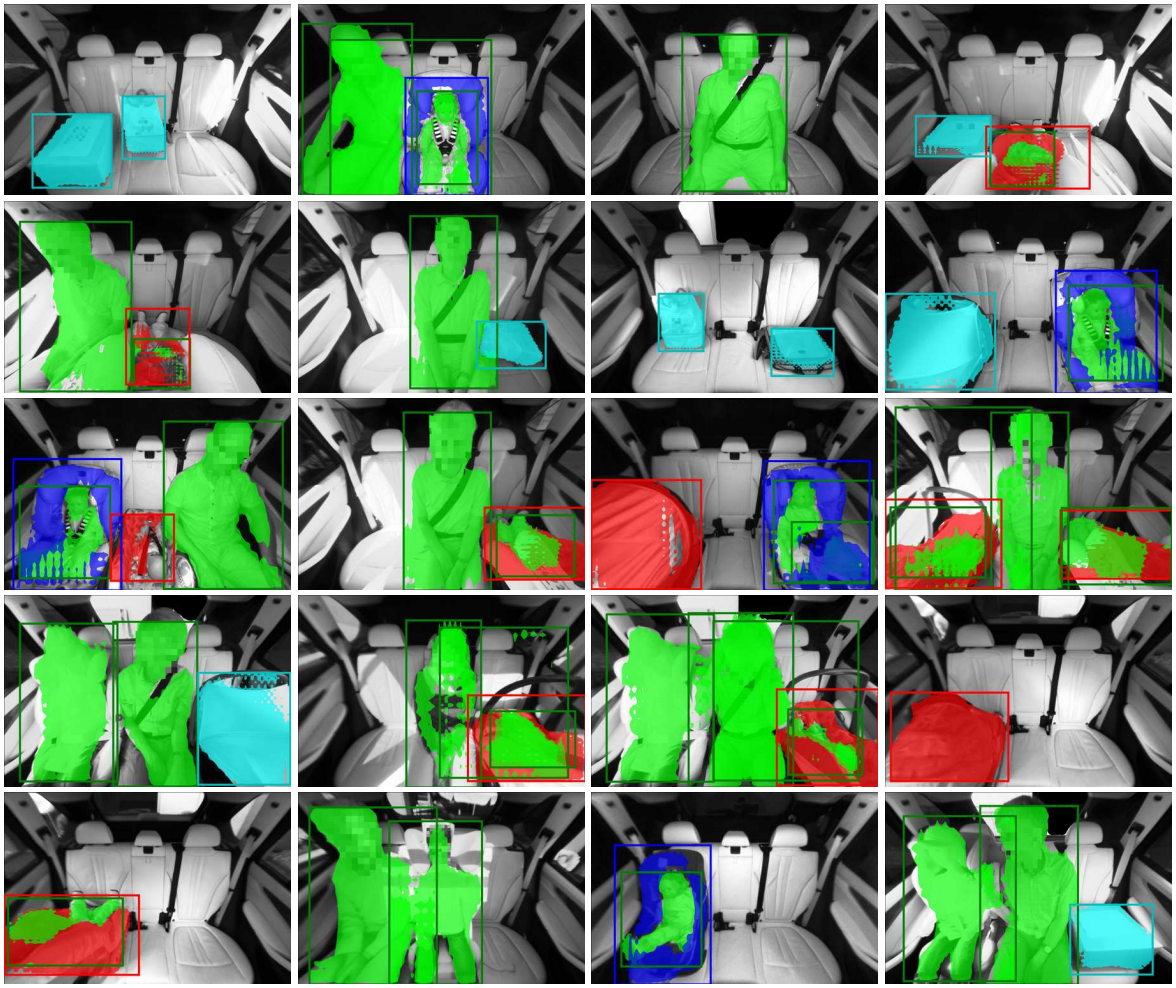


Fig. 5.6 Instance segmentation predictions on real images by a model fine-tuned on SVIRO.

5.2.7 Conclusion

Our benchmark addresses real-world engineering obstacles regarding the robustness and generalization of machine learning models. Using SVIRO and ORSS, we showed in our baseline evaluation that traditional and deep learning approaches, although commonly used in practice, drastically decrease in performance for solving the same task in a new vehicle interior when trained in a setting with limited variations without taking additional precautions. Models cannot generalize well to new intra-class variations, even in the car they were trained on. The performance fluctuates a lot: a model with the same performance on the test set of the training vehicle can have a very different behavior on unseen vehicles. Sometimes a model performs equally well on two unseen vehicles, sometimes it performs significantly better on one of both, and sometimes the best model on unseen vehicles is not the best one for the test set of the training vehicle. Consequently, no reliability can be guaranteed and the presented approaches do not fully grasp the underlying task, although the environment and the objects are similar. Including randomized images increases the performance, but to be applicable in real-world applications further improvements need to be investigated and developed. Hence, additional investigations need to be performed to better understand the behavior of machine learning models, which will be presented throughout the rest of this thesis.

5.3 ORSS

In this last section, we will provide some baseline results on ORSS to get a feeling for the test accuracy and vehicle-to-vehicle transferability for different commonly used CNNs, but also for some baseline autoencoder models.

Although these baseline results could probably be improved by a dedicated hyperparameter search, we decided to use for the autoencoder models the same hyperparameters as for the results presented in Section 5.1 and Chapter 2: the SSIM reconstruction loss, a batch size of 64, a learning rate of 0.001 with a weight decay of 0.3 using the AdamW optimizer for 1000 epochs. No augmentation but random horizontal flipping was applied to the training images. After the autoencoder training, we trained support vector machines, nearest neighbour classifiers or single hidden layer MLPs in the latent space using both the input images and their flipped versions. The CNNs were either trained from scratch, or the last block or all the layers were fine-tuned. We used a batch size of 64, a learning rate of 0.0001 and trained for 200 epochs.

The models were either trained on the Sharan or the X5 vehicle from the ORSS dataset. In both cases, the resulting models were then evaluated on the test sets of both vehicles. This way we obtain the performance on the test set in the same vehicle the model was trained on, but we also get an assessment of its transferability to a novel vehicle interior.

Table 5.5 Comparison of classification accuracies (in percentage) for different CNNs trained under different training regimes: the models were trained from scratch, or all layers or the last block only were fine-tuned. We also report results for several autoencoder variations for which we train different classifiers in the latent space: support vector machine, nearest neighbour (1-NN) and a single layer MLP. The models were either trained on the Sharan or the X5 vehicle of the ORSS dataset and then evaluated on the test sets of both vehicles. The experiments were repeated 10 times and the mean and standard deviation is reported. We highlight the best results for the different training-test cases for both CNNs and autoencoders respectively in gray.

		Trained on Sharan		Trained on X5	
Model	Variation	Sharan	X5	Sharan	X5
VGG-11	Scratch	78.4 ± 2.0	4.9 ± 1.5	13.3 ± 1.8	66.2 ± 3.9
VGG-11	Fine-tune	84.6 ± 1.7	15.0 ± 4.7	66.6 ± 5.7	85.9 ± 2.2
VGG-11	Last block	78.8 ± 1.0	30.8 ± 3.7	60.6 ± 4.5	79.9 ± 3.4
ResNet-50	Scratch	61.9 ± 4.3	11.8 ± 0.8	11.7 ± 0.8	73.7 ± 1.4
ResNet-50	Fine-tune	80.8 ± 6.4	4.2 ± 3.4	50.4 ± 4.8	87.6 ± 2.9
ResNet-50	Last block	66.3 ± 5.2	20.0 ± 2.4	52.8 ± 2.1	81.1 ± 2.0
DenseNet-121	Scratch	78.7 ± 1.2	12.6 ± 2.0	19.5 ± 2.6	75.5 ± 3.7
DenseNet-121	Fine-tune	79.5 ± 10.0	22.5 ± 5.9	53.2 ± 7.1	88.3 ± 2.5
DenseNet-121	Last block	62.9 ± 2.8	16.0 ± 1.9	47.2 ± 2.5	70.0 ± 2.0
AE	SVM	45.8 ± 14.1	5.1 ± 2.2	12.4 ± 2.1	31.9 ± 5.4
TAE	SVM	66.5 ± 2.8	6.0 ± 1.9	16.6 ± 2.6	71.6 ± 2.1
II-AE	SVM	67.7 ± 1.6	7.1 ± 0.9	10.6 ± 1.4	60.4 ± 4.0
AE	1-NN	46.0 ± 7.8	4.0 ± 0.4	11.3 ± 2.0	33.5 ± 1.7
TAE	1-NN	69.8 ± 2.3	7.3 ± 1.2	17.7 ± 2.3	80.8 ± 2.1
II-AE	1-NN	65.9 ± 2.4	8.0 ± 1.6	9.6 ± 1.3	64.2 ± 2.4
AE	MLP	40.5 ± 4.7	5.0 ± 1.3	10.0 ± 2.6	32.2 ± 3.0
TAE	MLP	69.3 ± 2.2	7.2 ± 1.1	17.3 ± 2.1	81.3 ± 2.2
II-AE	MLP	61.0 ± 3.7	6.5 ± 1.6	10.7 ± 1.6	60.1 ± 4.2

The experiments were repeated for ten runs and the mean performance and its standard deviations are reported in Table 5.5. The results show that obtaining a very high test performance cannot trivially be achieved, but some CNN variations, as well as the TAE, do obtain an acceptable classification accuracy. Further, the transfer to a novel vehicle does not work at all and the performance decreases drastically. Under these circumstances, it would not be possible to deploy a model trained in a single vehicle interior to a novel vehicle without further adaptations. It can also be observed that some vehicles are more beneficial for training than others: training on the X5 and evaluating on the Sharan results in a better performance than the other way around.

Although the autoencoder models perform worse than the pre-trained CNNs, we want to emphasize that the autoencoder models were not pre-trained on a large amount of data. Comparing the autoencoders against CNNs trained from scratch, we can observe that the former perform better. It is expected that the performance can be improved by dedicated regularization for the autoencoder model, or when additional data is being integrated. Further, these results provide evidence that using pre-trained CNNs is a beneficial design choice to extract features which transfer better. Consequently, the combination with an autoencoder could be beneficial, as proposed by the extractor autoencoder presented in Section 4.5. We hence believed that autoencoders had promising prospects and we will show in the next chapters their benefits when novel training strategies are being adopted.

5.3.1 Conclusion

Similarly to the results on SVIRO, we showed that the vehicle-to-vehicle transferability is indeed challenging if no additional precautions are taken. Neither the deployment of commonly used CNNs nor the one of autoencoders can be expected to work reliably in a novel vehicle interior. Overall, we showed that the industrial application investigated in this thesis is indeed challenging and that improvements are needed. We will hence present in the following chapters further investigations to deepen our understanding of the tasks at hand and inspect our proposed novel design choices to alleviate some of the challenges at least partially.

Chapter 6

Vehicle-to-vehicle generalization

Regarding the generalization of machine learning models between different car interiors, we formulate the criterion of training in a single vehicle: without access to the target distribution of the vehicle the model would be deployed to, neither with access to multiple vehicles during training. This is in contrast to common domain shift problem formulations [195], which consider the integration of multiple source domains, or the target domain during training. In this chapter, we perform a detailed investigation on SVIRO and propose a first, simple autoencoder-based approach as a baseline model to improve the transferability. These results will highlight the usefulness of the autoencoder model design for our problem formulation and motivate the design choices of the next chapters. The autoencoder is on par with commonly used classification models when trained from scratch and sometimes outperforms models pre-trained on a large amount of data. This is critical in case pre-trained models cannot be used due to licensing constraints. Moreover, the autoencoder can transform images from unknown vehicles into the vehicle it was trained on. These results are corroborated by an evaluation on real infrared images from ORSS.

6.1 Introduction

The deployment of deep learning-based approaches in the automotive industry needs to be corroborated by robustness and generalization guarantees, especially when the models' predictions would be used for safety critical applications, e.g. the adjustment of the strength of the airbag deployment in case of an accident [69, 197]. In this chapter, we will highlight some of the unique challenges for the vehicle interior regarding the robustness and generalization of machine learning models and we will thoroughly extend the baseline results from the previous chapter: we will report on a more elaborated investigation on classification models and develop

Fig. 6.1 Synthetic images from SVIRO. Left: infant seat with a baby, Right: child seat with a child. Training on images from a ***single*** vehicle interior (a): How can we improve robustness on same class instances, but to a new vehicle (b)? Can the model generalize to new class instances in the vehicle it was trained on (c)? How can we improve generalization to new class instances in a new vehicle (d)?

and evaluate a first approach on autoencoder models to motivate the choice of the latter for the rest of the thesis.

Machine learning models trained in a single vehicle interior take non-relevant characteristics of the background into account for their decision [237], because the training data contains similar backgrounds for all images. Consequently, the performance drops drastically if models trained in a single car interior are used in a different vehicle. Repeating the data recording and annotation generation process for each new car model and automotive manufacturer implies a time-consuming and costly development pipeline. Alternative sensor data (e.g. depth maps computed by time-of-flight sensors and RADAR [52]) and the inclusion of data from different vehicles (e.g. domain generalization) would improve the transferability. However, improving the foundations of deep learning models is paramount for safety critical applications. As illustrated in Fig. 6.1, we formulate the challenge of training in a single vehicle interior and generalizing to unseen cars without using images from the target distribution or from multiple vehicles. Even more difficult, the models' generalization to new objects in new environments should be examined as well. This can be considered as an extreme form of domain adaptation [188] and is related to (generalized) zero-shot learning [263, 34]. We demonstrate that transfer learning is not sufficient to ensure consistency between different vehicles for the aforementioned challenging training conditions. Autoencoders with a classifier in the latent space achieve accuracies on par with classification models. Moreover, the autoencoders can transform images from unknown vehicles to the one they were trained on. We compare different reconstruction cost functions for the training of autoencoders which lead to different behaviors for classification and reconstruction transferability. We corroborate the inter-vehicle transformation by showing that it works on real ORSS infrared images as well, even when trained on synthetic images. The resulting advantages are two-fold: we need less data to achieve similar performances, which is important when pre-trained models cannot be used due to licensing constraints. Further, as we

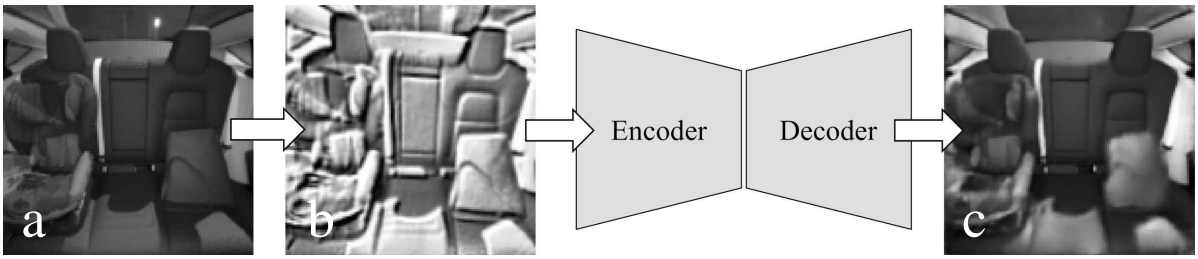


Fig. 6.2 Image (a) from the SVIRO dataset is augmented using different random transformations to form image (b). The latter is used as input to the autoencoder, which should learn to output image (c), i.e. transform (b) back to (a).

will show later in this work, the inclusion of prior knowledge in the latent space of autoencoders further improves the robustness and transferability.

6.2 Method

The autoencoder baseline method presented in this chapter is based on exploiting the disadvantage of our problem formulation to our advantage. As all the images from a single vehicle interior contain similar backgrounds, an autoencoder should be able to learn to reconstruct the vehicle interior robustly. If we augment the training images, then the autoencoder should easily learn to clean the augmented images, because the clean backgrounds do not undergo a high variability. Example images are shown in Fig. 6.2: we augment the training images (a) to obtain a modified version (b) by changing colors and the perspective and adding random noise. The autoencoder then reconstructs (c), which should be the initial clean version (a). This can be considered as a more extreme version of de-noising and in-painting [266, 252], see Section 2.5.1, and can be considered as a weak version of the PIRL. The autoencoder needs to map different backgrounds and perspectives to similar latent space representations to correctly reconstruct the background. Hence, a classifier using the latent space vector as input should learn to neglect background information more easily than using the input image.

6.2.1 Architecture details

Our autoencoder network architecture is inspired by SegNet [12]: we keep track of the indices from the max-pooling in the encoder part and use them for the max-unpooling in the decoder part. The network architecture is illustrated in Fig. 6.3: All convolutional layers use 3×3 kernels with a padding of 1 and a stride of 1. Each convolutional layer is followed by batch normalization and a ReLU activation function. After each block of two convolutional layers we apply a max-pooling (or max-unpooling) layer with a kernel size of 2×2 and a stride of 2

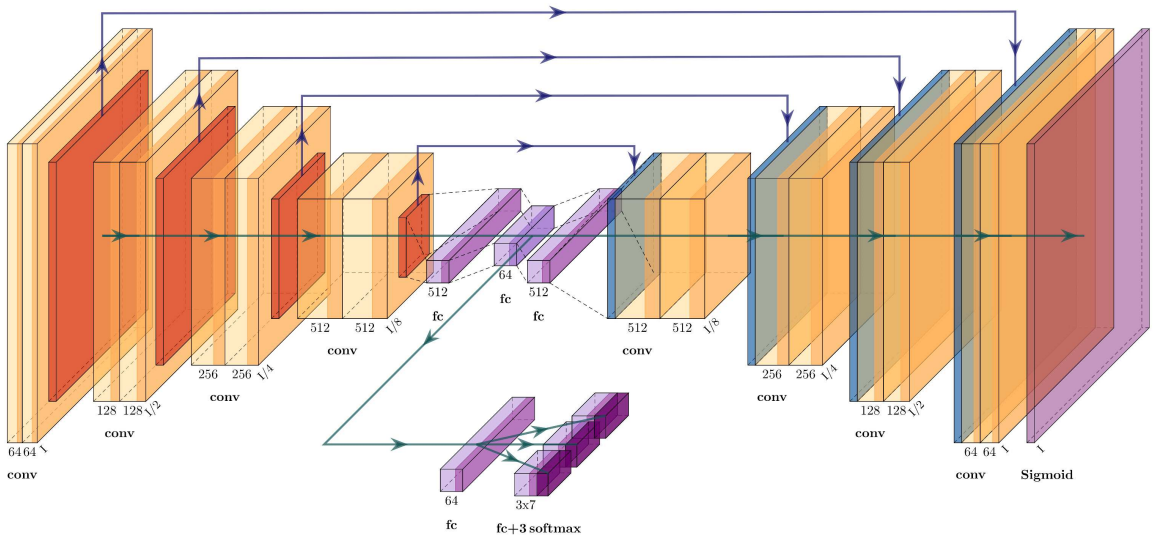


Fig. 6.3 Our model consists of a simple autoencoder model with a classifier in the latent space. We use four down-sample and four up-sample blocks: each consisting of two convolutional layers. The number of filters and features are specified for each layer. Max-unpooling (blue) uses the indices from the down-sampling counter-part (red). We use three independent softmax layers for the classification. Illustrated using PlotNeuralNet [91].

and we double (or halve) the amount of filters. The four down-sampling blocks are followed by two fully connected layers (+ intermediate ReLU) where the last one generates the latent vector. The decoder is exactly the reverse of the encoder network with a final sigmoid function. The latent vector is used as input for the decoder network and the classification network. The latter consists of two fully connected layers (+ intermediate ReLU) where the last one outputs three predictions: one for each seat position (left, middle and right). Hence, we need to apply three independent softmax functions to the output of the last layer. This way, we implicitly force the network to learn to use each classifier for a different seat position. Moreover, we can use the whole image as input compared to using an individual image for each seat position. Thus, passengers leaning over to the neighboring seat can be easier classified. Additionally, the middle prediction can be dropped in case of a two-seated car, which is not possible if you classify the image as a whole. This design choice is necessary for this chapter and its investigated benchmarks in order to compare the models across all available cars, i.e. two- and three-seated. Further, we will also report results for the same architecture with nearest neighbor up-sampling instead of max-unpooling, i.e. without using skip connections. This will provide some comparisons and insights such that we can motivate the selection of one of the two design choices for the rest of the thesis.

We define the cost function $\mathcal{L}(x,\hat{x};\theta,\phi,\omega)$ as a combination of the reconstruction loss for the whole image and the classification loss for each seat position:

$$\mathcal{L}(x,\hat{x};\theta,\phi,\omega) = \mathrm{r}\left(x,\mathrm{d}_\theta(\mathrm{e}_\phi(\hat{x}));\theta,\phi\right) - \gamma\sum_{i=0}^{2} p_i(x)\log\left(\mathrm{c}_{\omega_i}(\mathrm{e}_\phi(\hat{x}))\right), \tag{6.1}$$

where e_ϕ is the encoder, d_θ the decoder, c_{ω_i} the classifier for seat i where i corresponds to the left, right and middle seat position, $p_i(x)$ is the true probability distribution for x for seat i, and $\gamma \geq 0$ is a hyperparameter to weight the classification loss. The reconstruction loss $\mathrm{r}(\cdot,\cdot,\theta,\phi)$ is computed between the clean input image x (Fig. 6.2.a) and the reconstruction $\mathrm{d}_\theta(\mathrm{e}_\phi(\hat{x}))$ (Fig. 6.2.c) of the augmented image $\hat{x}$ (Fig. 6.2.b). In this chapter, we consider for the reconstruction loss the MSE, SSIM, MS-SSIM and the perceptual loss (PC).

6.3 Experiments and results

We will present a comparison between a representative selection of classification models. We start by establishing a baseline regarding the transferability between different vehicle interiors and their generalization to new class instances. These results will be compared against our autoencoder approach for which we will conduct additional investigations to highlight some of the advantages compared to classification models: the autoencoder is able to transform sceneries from unknown vehicles back to the vehicle it was trained on. The latter is corroborated by a qualitative and quantitative evaluation on real infrared images to show the applicability to a real application, even when trained on synthetic data.

6.3.1 Training data

We limited our training on the eight vehicles with three seat positions, but the evaluation will be performed on all ten vehicles such that the models also need to generalize to the two-seated cars. For the latter, the model output for the middle seat will be discarded. Each vehicle consists of a train (2500 images) and a test (500 images) split: SVIRO uses different objects (but identical ones across the vehicles) for the train and test split to evaluate the generalization to unknown vehicle interiors either using known or unknown class instances. When referring in the following to training images from unknown vehicles, those images had not been used during training, but they contain class instances of seen objects. We conducted our experiments on the grayscale images (simplified infrared imitations) which helps to become less susceptible to changing illumination.

6.3.2 Training details

All models were trained using the following data augmentations: we applied a random horizontal flip (labels need to be adopted as well) and randomly used transformations from the imgaug library [116] (PerspectiveTransform, Emboss, Invert, SigmoidContrast, AllChannelsCLAHE and AdditiveLaplaceNoise). The perspective transformation has the largest impact on improving the generalization between the vehicles. Images were center-cropped to 640x640 and then resized to 128×128 for the autoencoder and 224×224 for the classification models to match the pre-defined architectures and the resolution of the data used for pre-training. For the latter, the grayscale images were repeated across the channel dimension to form a 3-channel image. The training dataset was partitioned randomly according to an 80:20 split for training and evaluation (the evaluation data was augmented for better transferability assessment), where the latter was used to perform early stopping with respect to classification accuracy. The latter is necessary to exclude overfitting, particularly because the dataset sizes are rather small. The models were trained for 1000 epochs with a batch size of 64. The random seeds of all libraries were fixed for all experiments. Even though the classes are imbalanced, we did not consider a weighted loss or imbalanced sampling, because some models are capable of achieving a good performance on the training objects in unknown vehicles. While we evaluated weighted loss and imbalanced sampling in a pre-study, it did not have a significant impact on the model performances. Since SVIRO contains seven classes, all the classification networks output 21 values which corresponds to three predictions: one for each seat position (left, middle and right). The classification loss is computed by adding up three cross-entropy losses (one for each seat position) as formulated in Eq. (6.1).

6.3.3 Representative classification models

We trained six classification models (DenseNet-121 [105], MobileNet V2 [219], ResNet-18 [93], ResNet-50 [93], SqueezeNet V1.1 [108] and VGG-16 [225]) as implemented and pre-trained in torchvision from scratch or fine-tuned all layers. We replaced the last fully-connected layer with an output similar to the autoencoder classification network explained in Section 6.2. Fine-tuning the classification layer or last-block only behaves similarly as shown in the baseline evaluation in Chapter 5. We used the Adam optimizer with a learning rate of 0.0001. While we used weight decay for the autoencoder training, we decided not to use it for the classification models because of the conducted hyperparameter search reported in Table 6.1.

The training images were augmented using the same transformations as for the autoencoder approach. For each classification model and each training method (scratch and fine-tuned), we trained an individual model on the training images of each vehicle and then evaluated it on the

Table 6.1 We tested the hyperparameters used for the autoencoder training on the classification models for a fair comparison between both models' architectures. The classification models were trained on the Tesla vehicle and evaluated on all training and test images of the nine unknown vehicles. We tested the models when trained from scratch or when fine-tuning all layers from a pre-trained one. There is no clear winner, but we decided not to use weight decay as it performs slightly better.

Model	Weight decay	Approach	Train Accuracy	Test Accuracy
VGG-16	0	Scratch	78.3	50.6
		Fine-tune	94.0	61.8
	0.01	Scratch	82.5	47.0
		Fine-tune	92.9	61.4
DenseNet-121	0	Scratch	76.2	47.3
		Fine-tune	88.8	64.5
	0.01	Scratch	62.8	42.7
		Fine-tune	84.5	59.2
MobileNet	0	Scratch	76.7	49.4
		Fine-tune	91.5	60.6
	0.01	Scratch	79.8	50.8
		Fine-tune	91.7	58.9
ResNet-50	0	Scratch	80.1	51.9
		Fine-tune	88.0	57.4
	0.01	Scratch	75.0	46.8
		Fine-tune	84.2	51.9
ResNet-18	0	Scratch	75.4	47.8
		Fine-tune	86.3	57.2
	0.01	Scratch	80.4	52.9
		Fine-tune	86.8	54.9
SqueezeNet	0	Scratch	69.6	46.1
		Fine-tune	83.3	50.2
	0.01	Scratch	71.3	45.3
		Fine-tune	83.2	50.6

training and test images of all the other nine unknown vehicles. The results are summarized in Table 6.5 and Table 6.6. A performance comparison for models trained from scratch on the Tesla vehicle and evaluated on the training images of all unknown vehicles is reported in Table 6.2 and on the test images in Table 6.3. The models' performances vary a lot across the different vehicle interiors: higher accuracy performance on ImageNet does not guarantee a better inter-vehicle transferability (e.g. DenseNet compared to MobileNet). Moreover, a performance evaluation on one vehicle does not necessarily transfer to a similar performance on other vehicles (e.g. ResNet-50 vs. MobileNet on the A-Class compared to most other vehicles). Hence, it is difficult to assess in advance how well a model might perform in a new vehicle, even if only class instances seen during training are considered. Moreover, evaluating a model's performance on a subset of vehicles does not guarantee a similar behavior in a different car. Table 6.2 provides a comparison between training from scratch and fine-tuning classification models against the autoencoder approach. The fine-tuned models achieve overall a higher accuracy, however, the performance still fluctuates between different vehicle interiors. The performance on the Tiguan is worse compared to all other vehicles, most likely because of the unique pattern and color difference in the texture of the rear seats: e.g. see last row of Fig. 6.6.

6.3.4 Proposed autoencoder classification

We will present results on the autoencoder architecture introduced in Section 6.2 and compare them against the models obtained in Section 6.3.3. All models use a latent space of dimension 64, the AdamW optimizer with a learning rate of 0.0001 and a weight decay of 0.01. We used $\gamma = 75$ in Eq. (6.1) to weight the classification loss with respect to the MSE reconstruction loss according to the conducted hyperparameter search in Table 6.4. All other reconstruction losses used $\gamma = 1$.

We trained an individual autoencoder on each vehicle and for each reconstruction cost function (MSE, SSIM, MS-SSIM and perceptual loss). If not stated otherwise, we used max-unpooling for up-sampling. The different autoencoder models are compared in Table 6.2 against the results of the classification models obtained in Section 6.3.3: the models were trained on the Tesla vehicle and compared on the training images of the nine vehicles not seen during training. Different reconstruction losses influence the transferability of the reconstruction quality, but also the classification accuracy. The MS-SSIM cost function yielded the best accuracy when trained on the augmented training images while the perceptual loss generated the best reconstruction quality: see Section 6.3.7 for a visual comparison. Comparing the mean performance across all vehicles, all the different autoencoder models outperform the classification models trained from scratch and sometimes even exceed the performance of the fine-tuned models pre-trained on a large amount of data. Autoencoders with nearest neighbor

Table 6.2 Comparison of the accuracies (in percentage) across different vehicles. The classification models were trained from ***scratch*** (S) or ***fine-tuned*** (F) and the autoencoders with and without (W) max-unpooling were trained from scratch only with different reconstruction losses. The models were trained on the augmented training images of the ***Tesla*** vehicle and tested on the ***training*** images of all vehicles not seen during training. Abbreviations: Dense is DenseNet121, Mobile is MobileNet, Res-50 is ResNet-50, Res-18 is ResNet-18, Squeeze is SqueezeNet and PC is perceptual loss. Best results without pre-training are highlighted for a fairer comparison between autoencoder and classification model.

		Trained on Tesla. Tested on vehicle:										
		A-Class	Escape	Hilux	Lexus	Tiguan	Tucson	X5	i3	Zoe	Mean	Std
VGG-16	F	93.6	93.6	92.5	95.3	78.8	98.6	96.5	99.1	98.2	94.0	5.8
	S	73.0	88.8	69.9	87.2	59.4	89.7	83.4	80.8	72.9	78.3	9.6
Dense	F	81.3	90.2	91.4	93.4	83.2	86.4	96.7	99.4	77.4	88.8	6.9
	S	65.1	86.2	63.7	71.1	69.5	84.8	84.1	91.0	70.2	76.2	9.7
Mobile	F	80.1	86.1	93.3	94.9	81.4	94.5	98.0	98.6	96.2	91.5	6.7
	S	82.6	76.2	71.8	80.4	65.4	77.7	78.1	77.6	80.4	76.7	4.9
Res-50	F	76.0	89.0	84.8	91.1	79.2	93.5	98.0	96.8	83.5	88.0	7.2
	S	75.8	81.8	75.8	72.8	63.3	86.0	89.8	89.3	86.6	80.1	8.4
Res-18	F	82.1	88.4	79.0	89.3	77.7	86.0	98.3	98.6	77.1	86.3	7.7
	S	61.2	78.5	70.1	70.1	63.0	84.2	85.6	94.4	71.2	75.4	10.4
Squeeze	F	76.7	84.8	69.2	76.8	85.5	86.5	93.6	95.2	81.8	83.3	7.8
	S	69.0	72.8	56.6	58.4	69.8	75.6	81.6	80.1	62.4	69.6	8.5
SSIM		78.4	85.5	82.5	76.0	70.9	93.5	90.2	87.2	87.1	83.5	6.8
	W	81.6	86.0	85.9	85.0	65.7	92.0	88.7	90.0	91.5	85.1	7.6
MSSSIM		84.0	86.1	87.9	86.8	65.7	94.7	89.6	93.7	91.0	86.6	8.1
	W	73.9	82.7	82.1	84.1	58.3	89.8	88.5	88.4	84.4	81.4	9.3
PC		80.8	86.8	82.3	87.0	63.1	92.9	84.5	94.2	90.8	84.7	8.8
	W	81.1	80.7	83.9	82.5	61.1	91.3	83.4	90.3	89.6	82.7	8.5
MSE		78.8	86.4	82.0	80.6	60.0	92.5	84.6	93.9	92.4	83.5	9.8
	W	81.7	81.3	78.1	83.3	61.1	93.0	81.8	88.8	91.6	82.3	8.9

Table 6.3 Additional results for Table 6.2. Comparison of the accuracies (in percentage) across the different vehicles for several classification models and autoencoders with different reconstruction losses. The classification models were trained from ***scratch*** (S) or ***fine-tuned*** (F) and the autoencoders with and without (W) max-unpooling were trained from scratch only. The models were trained on the augmented training images of the ***Tesla*** vehicle and tested on the ***test*** images (Table 6.2 reported results for training images) of all vehicles not seen during training. The autoencoders outperform all classification models when trained from scratch. Abbreviations: Dense is DenseNet121, Mobile is MobileNet, Res-50 is ResNet-50, Res-18 is ResNet-18, Squeeze is SqueezeNet and PC is perceptual loss. Best results without pre-training are highlighted for a fairer comparison between autoencoder and classification model.

		Trained on Tesla. Tested on vehicle:										
		A-Class	Escape	Hilux	Lexus	Tiguan	Tucson	X5	i3	Zoe	Mean	Std
VGG-16	F	64.9	67.7	55.1	67.1	50.4	65.1	67.7	62.5	55.4	61.8	6.1
	S	46.9	56.2	42.7	59.9	39.2	61.7	57.7	49.7	41.5	50.6	8.0
Dense	F	61.4	69.5	67.0	74.9	57.7	57.3	77.0	66.8	49.1	64.5	8.5
	S	42.3	55.6	40.2	44.9	47.1	53.9	58.6	48.2	34.6	47.3	7.3
Mobile	F	54.8	57.8	62.2	68.4	54.4	58.8	72.1	63.4	53.2	60.6	6.2
	S	55.1	48.1	49.9	55.2	46.0	53.1	53.9	40.5	43.1	49.4	5.1
Res-50	F	47.5	61.8	51.7	61.6	53.7	60.9	70.1	61.2	48.1	57.4	7.1
	S	53.1	56.5	47.7	50.9	41.8	56.9	60.6	49.9	49.7	51.9	5.3
Res-18	F	54.6	68.8	50.3	64.9	50.6	55.3	70.9	59.8	39.5	57.2	9.4
	S	39.5	56.9	43.4	46.0	38.0	54.3	60.3	54.8	37.2	47.8	8.4
Squeeze	F	47.3	55.1	40.7	47.1	56.5	51.6	62.0	54.1	37.3	50.2	7.4
	S	47.0	51.0	37.3	42.7	50.9	49.1	59.3	45.5	32.0	46.1	7.6
SSIM		52.9	61.3	53.0	49.8	50.9	62.1	66.5	58.9	47.8	55.9	6.1
	W	54.9	60.3	50.3	59.5	49.7	61.4	62.5	53.6	49.3	55.7	5.0
MSSSIM		60.1	63.9	56.8	60.5	49.8	65.0	63.0	56.5	46.2	58.0	6.1
	W	50.4	59.9	54.3	61.2	48.5	60.7	63.1	53.7	47.8	55.5	5.5
PC		51.0	59.3	47.9	56.7	49.3	61.1	59.5	56.9	47.3	54.3	5.1
	W	62.5	59.6	57.1	61.1	52.3	60.5	63.8	58.5	49.7	58.3	4.4
MSE		57.8	61.4	57.4	61.1	50.5	62.4	61.7	61.7	49.8	58.2	4.6
	W	61.3	60.4	53.3	62.1	50.6	61.8	65.1	59.6	49.5	58.2	5.3

Table 6.4 Comparison of different γ values to weight the classification loss accordingly with the MSE reconstruction loss. The autoencoder was trained on the Tesla vehicle and evaluated on the training and test images of all unknown vehicles. The effect of γ on the performance does not seem to follow a regularity, but 75 seems to provide a good trade-off.

	γ							
	1	25	50	75 (used)	100	125	150	175
Training Accuracy	61.6	76.7	77.9	83.5	82.8	82.5	83.6	82.7
Test Accuracy	47.2	55.2	51.2	58.2	54.8	55.5	55.8	58.1

up-sampling perform slightly better with respect to accuracy, but cannot compete with the domain transformation presented in Section 6.3.7.

6.3.5 SVIRO benchmark results

The official SVIRO benchmark on the website reports results on the mean test classification accuracy: a model should be trained in a single vehicle and the mean accuracy on the test images across all nine vehicles not seen during training is taken as the score. Each of the aforementioned models was trained individually on each vehicle and then evaluated on the nine unknown vehicles: an overview of the performances is reported in Table 6.5. We report the mean accuracy on training on the different vehicles and evaluating on all remaining nine vehicles: Autoencoders perform better than training classification models from scratch, but no method outperforms consistently.

6.3.6 Ablation study

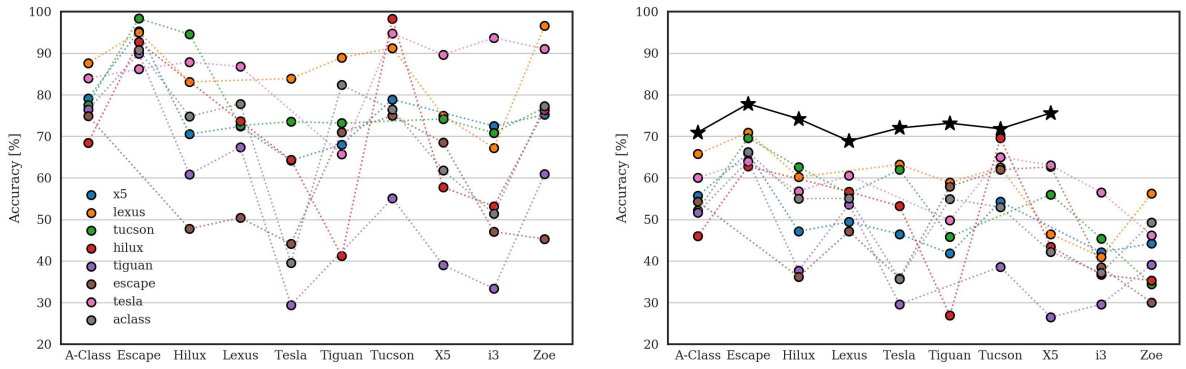
Apart from comparing the autoencoder based approach against the classification models, we also investigated the method itself in more details. We trained an individual MS-SSIM autoencoder on each of the eight vehicles. In Fig. 6.4a, the resulting models are compared individually on the training images of each of the nine vehicles not seen during training. The results on the test images are reported in Fig. 6.4b. Depending on which vehicle the models were trained on, the trend in their overall performance can be quite different and fluctuate a lot: some vehicles are more advantageous while others lead to worse results. The fluctuation for the generalization to unknown class instances in the vehicle the models were trained on ($\star$) is smaller than for the generalization between different vehicles. Although generalizing to new class instances is hard, this gives a hint that the changing background has a large influence on the models' robustness. As we did not train on two-seated cars no data points are available for those.

Table 6.5 Overview of the SVIRO leaderboard accuracies (in percentage). The models were trained on ***different*** vehicles and then evaluated on the ***test*** images of all nine unknown vehicles. Using the augmented images, the autoencoders with and without (W) max-unpooling and the classification models were trained from ***scratch*** (S) or ***fine-tuned*** (F). Abbreviations: Dense is DenseNet121, Mobile is MobileNet, Res-50 is ResNet-50, Res-18 is ResNet-18, Squeeze is SqueezeNet and PC is perceptual loss. Best results without pre-training are highlighted for a fairer comparison between autoencoder and classification model.

		Trained on									
		A-Class	Escape	Hilux	Lexus	Tesla	Tiguan	Tucson	X5	Mean	Std
VGG-16	F	61.3	62.6	54.1	58.1	61.8	48.5	66.3	58.3	58.9	5.2
	S	51.9	49.3	49.7	60.6	50.6	44.6	45.9	52.0	50.6	4.5
Dense	F	59.2	58.1	60.9	67.4	64.5	49.2	68.4	52.0	60.0	6.4
	S	42.4	49.4	47.6	48.2	47.3	35.5	46.6	40.0	44.6	4.5
Mobile	F	51.8	47.9	58.3	57.0	60.6	45.5	53.0	51.2	53.2	4.9
	S	46.5	50.9	45.1	53.6	49.4	44.7	47.9	47.4	48.2	2.8
Res-50	F	49.1	49.7	52.5	49.1	57.4	50.0	60.0	49.2	52.1	4.0
	S	48.3	47.1	41.3	48.5	51.9	35.7	44.7	43.1	45.1	4.7
Res-18	F	56.6	52.3	56.2	59.5	57.2	51.2	58.7	42.4	54.3	5.2
	S	47.6	49.4	45.6	52.8	47.8	40.9	49.9	46.8	47.6	3.3
Squeeze	F	55.0	51.1	54.5	56.4	50.2	53.4	51.4	45.7	52.2	3.2
	S	52.4	48.8	48.4	51.2	46.1	43.4	50.6	39.2	47.5	4.1
SSIM		54.8	52.0	49.1	58.2	55.9	42.1	50.4	53.3	52.0	4.6
	W	49.9	53.5	51.5	61.6	55.7	41.9	53.6	59.1	53.3	5.6
MSSSIM		49.8	47.2	47.8	58.4	58.0	41.2	53.8	50.0	50.8	5.4
	W	51.6	53.7	48.5	57.6	55.5	43.2	52.1	52.0	51.8	4.1
PC		52.5	56.0	45.2	61.2	54.3	39.1	46.1	52.9	50.9	6.6
	W	52.3	53.1	51.2	59.7	58.3	44.8	55.1	53.9	53.6	4.3
MSE		45.3	49.5	51.1	59.0	58.2	46.8	44.5	55.0	51.2	5.3
	W	55.0	52.9	49.9	60.6	58.2	46.2	47.6	52.9	52.9	4.7

Table 6.6 Additional results for Table 6.5. The models were trained on different vehicles and then evaluated on the ***training*** images (Table 6.5 reported results for test images) of all nine unknown vehicles. We compare the accuracy (in percentage) for different models presented in this work. The autoencoders with and without (W) max-unpooling and the classification models were trained from ***scratch*** (S) or ***fine-tuned*** (F). The main paper compared performances on the test images. Abbreviations: Dense is DenseNet121, Mobile is MobileNet, Res-50 is ResNet-50, Res-18 is ResNet-18, Squeeze is SqueezeNet and PC is perceptual loss. Best results without pre-training are highlighted for a fairer comparison between autoencoder and classification model.

		Trained on vehicle									
		A-Class	Escape	Hilux	Lexus	Tesla	Tiguan	Tucson	X5	Mean	Std
VGG-16	F	86.0	90.7	86.8	92.2	94.0	75.4	97.0	83.5	88.2	6.4
	S	76.3	80.5	82.4	89.5	78.3	66.9	78.5	82.5	79.4	6.0
DenseNet	F	85.5	86.6	94.0	93.9	88.8	75.8	93.6	81.2	87.4	6.2
	S	65.6	76.2	70.6	77.4	76.2	53.1	76.0	64.5	70.0	7.9
Mobile	F	80.8	77.2	90.2	86.5	91.5	77.1	85.6	81.5	83.8	5.2
	S	69.8	69.1	69.2	76.8	76.7	60.3	65.9	73.9	70.2	5.2
Res-50	F	82.9	78.6	80.7	80.4	88.0	74.1	89.4	81.4	81.9	4.6
	S	70.9	71.3	63.2	74.0	80.1	51.6	68.8	72.6	69.1	8.0
Res-18	F	84.3	84.4	86.2	93.1	86.3	75.1	87.6	75.7	84.1	5.6
	S	72.4	72.9	69.1	83.9	75.4	59.6	74.8	72.8	72.6	6.3
Squeeze	F	84.2	77.3	83.0	84.3	83.3	76.9	80.7	74.3	80.5	3.6
	S	78.5	72.9	68.3	76.3	69.6	60.6	72.7	64.2	70.4	5.6
SSIM		76.2	70.5	72.0	82.6	83.5	59.7	73.5	80.0	74.8	7.3
	W	68.3	71.4	72.4	85.2	85.1	56.5	76.4	86.4	75.2	9.6
MSSSIM		70.2	58.2	69.5	85.4	86.6	56.9	79.1	75.1	72.6	10.5
	W	71.9	73.5	69.4	81.6	81.4	58.4	77.2	76.8	73.8	7.1
PC		78.6	77.5	66.1	86.3	84.7	55.9	69.7	77.8	74.6	9.5
	W	72.3	74.6	71.9	85.9	82.7	60.2	77.3	78.5	75.4	7.3
MSE		68.1	70.0	72.0	81.5	83.5	63.8	69.2	77.1	73.2	6.4
	W	75.1	71.5	68.8	86.7	82.3	59.4	68.7	76.0	73.6	8.0



(a) Training images - unknown vehicles

(b) Test images - unknown vehicles

Fig. 6.4 We trained an individual MS-SSIM autoencoder on each of the eight vehicles. The models were evaluated on the training images (left) and the test images (right) of the nine vehicles not seen during training. Different colors represent the vehicles each model was trained on. The performances of each model across all unseen vehicles are connected by dotted lines to ease the visualization. The performances on the test images of the vehicle the models were trained on ($\star$) are plotted as well.

In Fig. 6.5, we compare the confusion matrices from three unknown vehicles with similar mean accuracy by the model trained on the Tesla vehicle: in the Escape the model performs worse on adults and everyday objects and it misclassified many samples as empty while in the Hilux more infant seats are misclassified as unoccupied. On the Lexus, the model performs better on child seats, but more samples are wrongly predicted as people. Hence, a same model can behave quite differently (even with similar mean accuracy) on different vehicles such that no guarantees can be provided without additional precautions.

6.3.7 Vehicle domain transformation

An additional advantage of the autoencoder approach is the possibility to exploit the disadvantage of our training environment: since all images are from the same vehicle interior, the sceneries will not undergo many changes (besides the objects on the seat). Hence, by augmenting the training images and using the autoencoder as a denoiser, the model learns to transform images back to the original environment. This leads to useful properties when images from an unknown vehicle are used as input. In Fig. 6.6 we compare input images from six unknown vehicles against their transformed versions computed by autoencoders from Table 6.2 trained with different cost functions on images from the Tesla. The same experiment is repeated for autoencoder using nearest neighbor up-sampling in Table 6.7, but the results are blurrier and less stable. The autoencoders are able to transform the input images to a scenery in the vehicle the models were trained on by replacing the rear bench and adapting the perspective

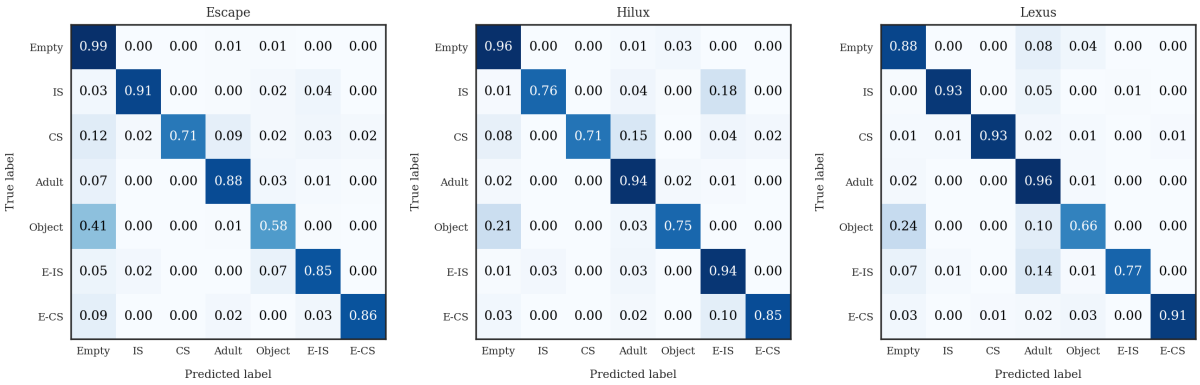


Fig. 6.5 Confusion matrices for the MS-SSIM autoencoder from Fig. 6.4 trained on the Tesla. Evaluation was done on the training images from the Escape (left), Hilux (middle) and Lexus (right). Although the model achieves a similar overall accuracy on all three vehicles, the mis-classifications are quite different. Abbreviations used: CS = child seat, IS = infant seat and E-* stands for empty.

accordingly. The different cost functions have an influence on the transferability and quality of the transformations. Overall, SSIM and the perceptual loss perform best. The reconstruction of the objects is not perfect, especially humans often appear blurrier, probably because of the higher visual complexity and variability due to different poses. Notice that it is much harder for the image from the Tiguan, because of the unique texture pattern. The reconstructions for the objects need further investigations regarding the delivery of valid and robust features that can be used from a classifier. In Fig. 6.8 we show several examples of the reconstruction of the same sceneries by autoencoders trained on different vehicles using the perceptual loss. For nearest neighbor up-sampling, the models have more trouble for the reconstruction in unseen vehicles, because all the information goes through the latent space in contrast to the indices when max-unpooling is used.

6.3.8 Applicability to real infrared images

We tested the transferability of the vehicle domain transformation presented in Section 6.3.7 on the real images from ORSS. Individual autoencoders were then trained on the real X5 and Sharan vehicles and the synthetic Tesla images respectively. We used the same architecture as presented in Section 6.2 and the perceptual loss for training. In Fig. 6.9, we report results on the autoencoder transformations for images of the vehicle not used during training. The backgrounds and rear seats from the Sharan are replaced by the ones from the X5 and vice versa. Albeit the results are not as detailed as if trained on real images, the transfer from synthetic to real images is possible and the background is replaced by the synthetic one. Additionally, we trained individual classification models and autoencoders using several reconstruction cost

Fig. 6.6 Autoencoders were trained on the Tesla using the de-noising approach. The first column contains input training images from unknown vehicles. The other columns show the corresponding transformations by the autoencoder for different cost functions: MSE, SSIM, MS-SSIM and perceptual loss.

Fig. 6.7 Reconstruction of the same examples as in Fig. 6.6 for autoencoders trained using the nearest neighbor up-sampling instead of max-unpooling with indices. The first column contains input training images from unknown vehicles. The other columns show the corresponding transformations by the autoencoder for different cost functions used: MSE, SSIM, MS-SSIM and perceptual loss. The model was trained on the Tesla vehicle. The results are blurrier and less stable as for max-unpooling.

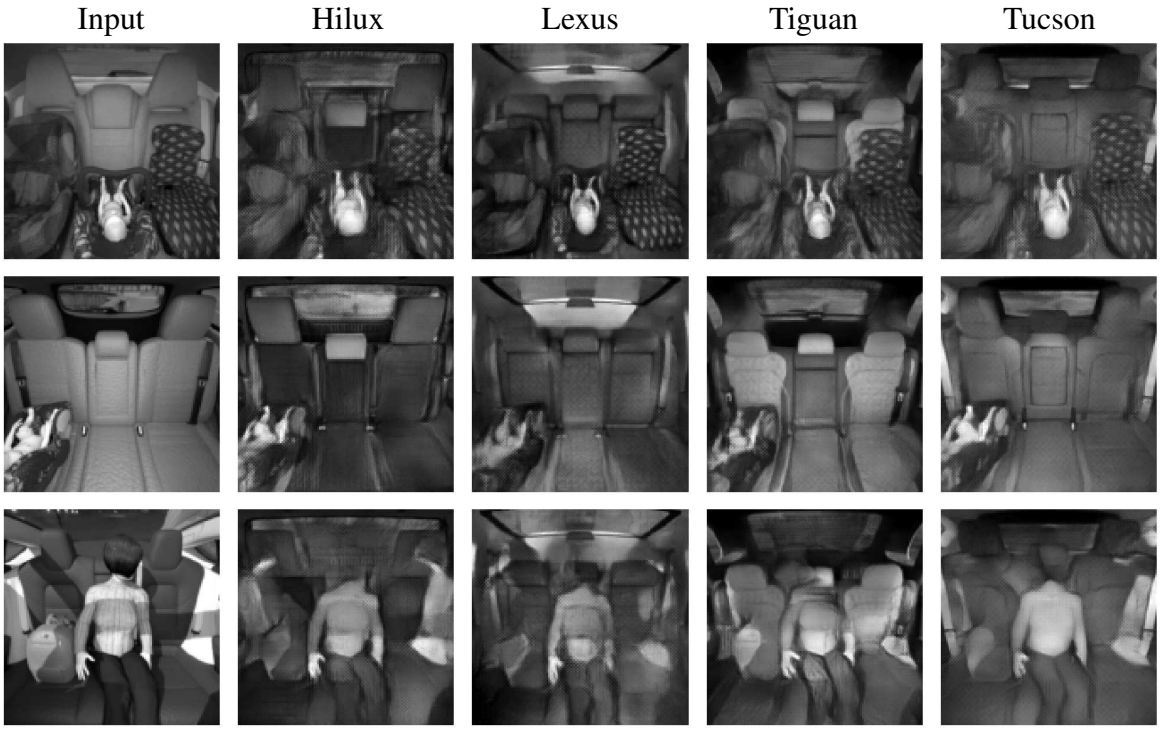


Fig. 6.8 Reconstruction of the same sceneries (first column) by autoencoders trained on different vehicles (remaining columns) using the perceptual loss during training.

functions in order to compare their generalization accuracy. The results are summarized in Table 6.7 and they are similar as for the synthetic data: our autoencoder approach generalizes better than classification models trained from scratch. Fine-tuned classification models usually perform best, but we think that improvements on the autoencoder will further close the gap and yield additional helpful properties. This highlights that design choices and insights made on SVIRO are transferable to ORSS.

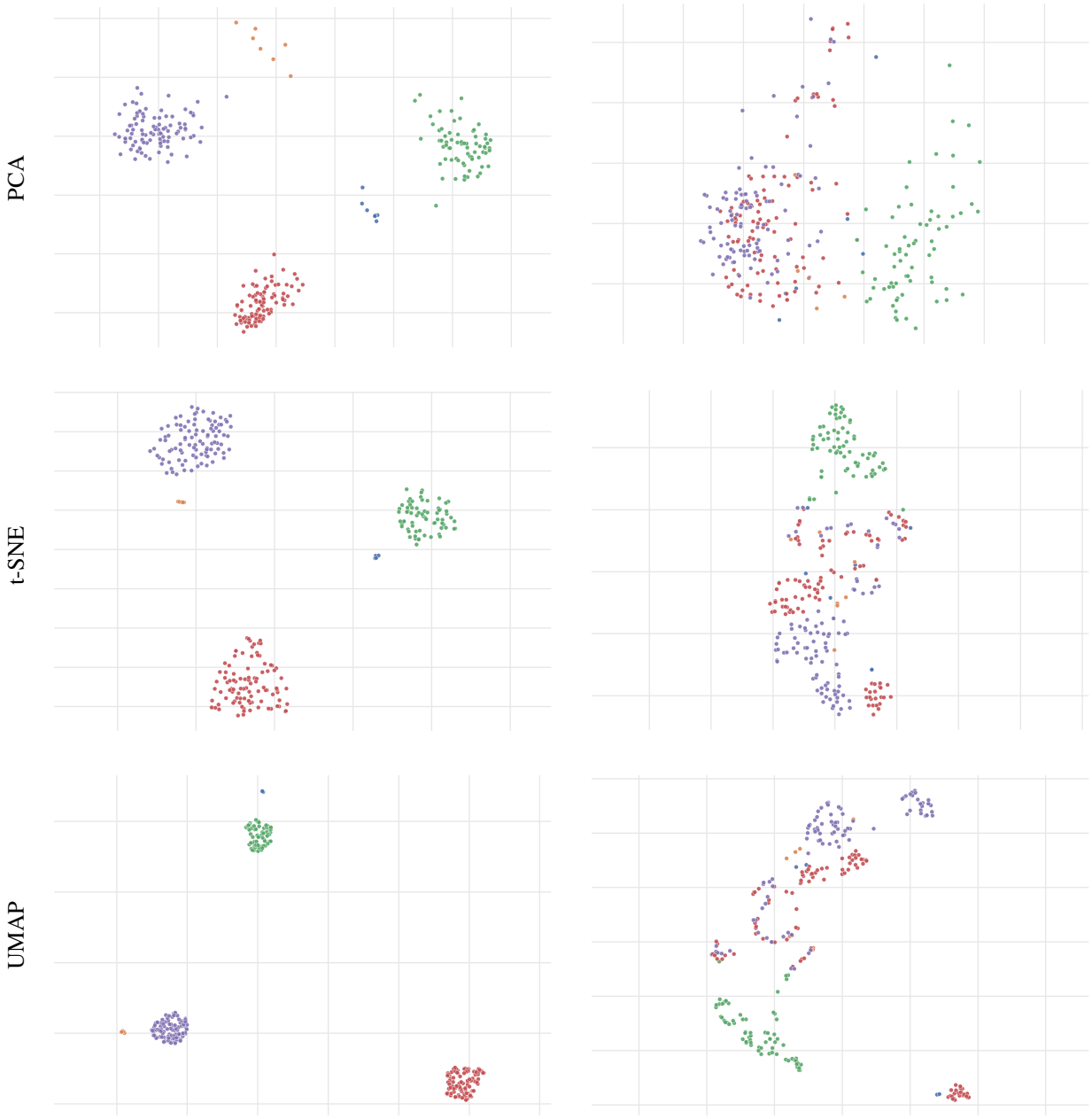
6.4 Discussion and limitations

The use of skip connections in the model design allows the model to skip the latent space and hence bottleneck. This makes it easier for the model to replace the background, since the model can learn to use the identity function for the objects and people on the seat. On the other side, this means that the latent space representation will loose on importance once the classifier in the latent space is not trained end-to-end anymore. This is problematic, since the latent space is important for inspections and to induce regularizations. We compare the latent space for different projection methods for models using skip connections either with or without a classifier in the latent space in Fig. 6.10. It can be observed that the clusters are well separated in case a classifier is used, but convoluted when the classifier is removed. Consequently, we

Fig. 6.9 Individual models were trained on *real* images from an X5 (first row) and Sharan (fourth row) and on *synthetic* images from the Tesla using the perceptual loss. The models were then applied to real images from the vehicle not used during training. Recon-R are reconstructions when trained on real images and Recon-S when trained on synthetic ones.

Table 6.7 Classification models were trained from ***scratch*** (S) or ***fine-tuned*** (F) and autoencoders with and without (W) max-unpooling were trained using different reconstruction losses. The models were trained on the augmented real images from the Sharan and X5 respectively and evaluated [%] on the images from the vehicle not seen during training. Best results without pre-training are highlighted for a fairer comparison between autoencoder and classification model.

		Tested on		
		Sharan	X5	Mean
VGG-16	F	90.2	87.9	89.1
	S	77.5	74.7	76.1
DenseNet-121	F	95.4	85.2	90.3
	S	68.1	68.2	68.2
MobileNet	F	91.9	89.3	90.6
	S	70.1	60.1	65.1
ResNet-50	F	91.2	93.0	92.1
	S	65.6	66.8	66.2
ResNet-18	F	87.7	93.0	90.4
	S	68.7	74.4	71.6
SqueezeNet	F	79.9	84.3	82.1
	S	70.6	57.1	63.8
SSIM		84.9	76.2	80.6
	W	75.3	81.7	78.5
MS-SSIM		76.8	82.1	79.4
	W	78.3	78.1	78.2
PC		75.7	81.8	78.8
	W	73.9	88.4	81.2
MSE		77.2	76.9	77.1
	W	71.6	72.5	72.0



(a) Classifier (b) No classifier

Fig. 6.10 Latent space projection of training samples using different projection methods (rows). The models using skip connections were either trained with a classifier (left) or without a classifier (right) on the latent space. We only considered classes with empty seats (0) or adults (3) to ease visualization by reducing the amount of classes to plot. It can be observed that using a classifier induces well separated clusters. Removing the classifier makes the clusters much more convoluted. Different colors represent different classes.

decided to drop the model design of skip-connections and max-unpooling for the rest of the thesis and stick to the normal decoder design choice. As the results in this section showed, the performance does not suffer significantly in case the skip connections are dropped.

6.5 Conclusion

We introduced the challenge of training in a single vehicle interior and improving generalization to unknown vehicles and class instances. Our results showed that commonly used classifiers do not behave reliably across different vehicle interiors and our introduced autoencoder approach outperforms classification models trained from scratch. This is important when pre-trained models cannot be used for commercialized applications due to licensing constraints. Although the applicability to real images has been shown, the generalization to new class instances needs to be improved and the transfer to new vehicles needs to be robustified to be applicable to safety critical applications. None of the investigated classification and autoencoder methods can guarantee a similar behavior on potentially new vehicles, even when multiple vehicles are available to test the models' behaviors during the design process. The models' predictions should be accompanied with uncertainty estimations to quantify the model's self-assessment with respect to its capability to generalize to new vehicle interiors or new sceneries. The approaches presented in next chapters will alleviate some of these challenges and point out the benefit of considering a model design based on autoencoders.

Chapter 7

Image and illumination normalization

We showed in the previous chapter the challenge of transferring from one vehicle to another and proposed a first baseline investigation and autoencoder approach as a starting point. Images recorded during the lifetime of computer vision based systems undergo a wide range of illumination and environmental conditions affecting the reliability of previously trained machine learning models. Hence, even within a same vehicle, classification performance can degrade in case the test images undergo a distributional shift like illumination or environmental changes. Image normalization is hence a valuable pre-processing component to enhance the models' robustness. To this end, we propose to use the first variation of the PIRL to average out all the unimportant information in the input images (e.g. environmental features and illumination changes) to focus on the reconstruction of the salient features (e.g. class instances). I-PIRL exploits the availability of identical sceneries under different illumination and environmental conditions by formulating a partially impossible reconstruction target: the input image will not convey enough information to reconstruct the target in its entirety. We combine the triplet loss as a regularizer in the latent space representation and a nearest neighbour search to improve the generalization to unseen illuminations and class instances. We will propose in the next chapter to learn invariances on synthetic data, which should then be transferred to their real counterparts. This way, improvements of this chapter, combined with the ones from the next chapter, can be used together to alleviate some of the challenges from the previous chapter to robustify the overall system of occupant detection in the vehicle interior.

7.1 Introduction

Recording a sufficient amount of images to train and evaluate computer vision algorithms is usually a time consuming and expensive challenge. This is aggravated when the acquisition of images under various illumination and weather conditions needs to be considered as well.

Notwithstanding the aforementioned data collection challenges, the performance of many machine learning algorithms suffers from changing illumination or environmental conditions, e.g. SLAM [80], place recognition [183], localization and classification [157], semantic segmentation [4], 3D human pose estimation [214] and facial expression recognition [216]. Since it is impracticable to wait for different weather conditions, day times and seasons to record images under as many variations as possible, it would be beneficial to train machine learning models to become invariant with respect to illumination and the exterior environment. Particularly for safety critical applications, as is common in the automotive industry, it would be of interest to reduce the amount of different illumination conditions necessary to guarantee reliable inference of machine learning models. Improvements on the aforementioned invariances would reduce the amount of mileage and images needed to be recorded and hence reduce the financial risk and time investment while improving the overall robustness of the deployed system.

In this chapter, we aim to transform the input image by removing illumination and environmental features instead of computing more robust and invariant feature descriptors like SIFT [154] or enforcing illumination invariance in deep neural networks through data augmentation. We achieve this by exploiting the availability of sceneries under different illumination and/or environmental conditions. We will use the first variation of the partially impossible reconstruction loss which enforces similarity in the latent space of autoencoder models implicitly, in opposition to an explicit constraint [8, 277]. In contrast to shadow removal [257, 201] or relighting [231, 278], our method removes all the illumination and environmental features together. Our method is neither limited to a specific application where prior knowledge, e.g. about faces [278, 224], needs to be included, nor does it need shadow and shadow-free image pairs [257, 201] to define a ground truth target. Further, it preserves the human poses for training images, compared to the II-PIRL which approximates humans by a mean representation. We highlight its applicability on multiple datasets and provide evidence for the usefulness of collecting images under these more challenging conditions.

Our cost function formulation, according to Eq. (4.2), implies a partially impossible task to solve. The input image X_a^j, i.e. variation a of scenery j, does not convey enough information to perfectly reconstruct the same scene under different environmental conditions X_b^j in its entirety. While X_a^j contains, usually, all the information of the objects in the scene, it does not contain any information about the illumination or environmental condition of X_b^j. However, both images are sufficiently close to each other such that the autoencoder model can learn to focus on what is important, i.e. the salient features (e.g. people). Consequently, the only possibility for the neural network to minimize the loss is to focus on the objects in the scene which remain constant and neglect all the illumination and environment information. The latter cannot be reconstructed, because the input images do not include information on how to handle

it correctly. This implies that the neural network implicitly learns to focus the reconstruction on people, objects and vehicle interior and to average out all the other information which changes between the similar scenes, e.g. the illumination and environment. This can later be observed in Fig. 7.2, Fig. 7.3 and Fig. 7.4 where we compare the reconstruction of similar sceneries after training: all background information and illumination conditions have either been removed or replaced by constant values. The encoder learns to remove the illumination information. The decoder is light invariant and cannot produce different illuminations, since the information has already been removed in the latent space representation.

Example of failures are plotted in Fig. 7.2 and Fig. 7.4: it can be observed, that the application on test images can cause blurry reconstructions. Hence, our proposition to combine I-PIRL with the triplet loss, which acts as a regularizer. Due to its definition, it will also induce an L^2 norm in the latent space. This effect is highlighted in Fig. 7.6, where we compare the five nearest neighbors of the AE, VAE and TAE.

Our proposed method is not limited to having the same scenery under different illumination conditions though. One could use different augmentation transformations on the same input image X^j to form X_a^j and X_b^j and hence create the images on the fly. Alternatively, one could apply a *reverse* denoising approach where only X_b^j is augmented and X_a^j is the clean input image. See Fig. 7.1 and Fig. 7.8 examples for both approaches. However, as shown on the ORSS images, in this case it might be more beneficial to use the II-PIRL.

7.2 Experiments and results

We will present an analysis of the aforementioned properties, problems and improvements on the SVIRO-Illumination dataset to highlight the benefit of our design choices. Further, we will present results on two additional publicly available datasets, Yale and Webcam, to show the applicability of our proposed cost function to other problem formulations as well.

7.2.1 Training details

We centre-cropped the images to the smallest image dimension and then resized it to a size of 224×224. We used a batch size of 16, trained our models for 1000 epochs and did not perform any data augmentation. We used a latent space of dimension 16 and the Adam optimizer with a learning rate of 0.0001. Image similarity between target image and reconstruction was computed using SSIM [21]. The model uses the VGG-11 architecture [225] for the encoder and reverses the layers together with nearest neighbour up-sampling for the decoder, i.e. the model is similar to the one from Chapter 6 without skip connections.

(a) Input and target are augmented differently

(b) Only target is augmented

Fig. 7.1 One can augment the input and target images differently (a) or augment the target images only (b), instead of using identical sceneries under different illumination conditions for the input-target pairs. Our proposed cost function still provides valid reconstructions, though the objects are blurrier in this case. This is expected, as augmentations are random and a consistent representation is hence more difficult to obtain. Nevertheless, the images are smoothed and averaged out, but the illumination invariance is not as good. We used the following augmentations: Gaussian noise, random contrast change, invert image, emboss, random hue and saturation change and random brightness change [32].

7.2.2 Extended Yale Face Database B

The Extended Yale Face Database B contains images of 28 human subjects under nine poses. For each pose and human subject, the same image was recorded under 64 illumination conditions. We considered the full-size image version, used 25 human subjects for training and three for testing. We removed some of the extreme dark (no face visible) illumination conditions. For the triplet sampling we chose as a positive sample an image with the same head pose and for the negative sample an image with a different head pose. We report qualitative results in Fig. 7.2. The model is able to remove illumination and shadows from the training images, but the vanilla reconstruction on test samples can be blurry. We were not using the center-cropped images, which made the task more complicated, because the head is not at the same position for each human subject. Nevertheless, the model is able to provide a nearest neighbour with a similar head pose and position.

7.2.3 Webcam Clip Art

The Webcam Clip Art dataset consists of images from 54 webcams from places all over the world. The images are recorded continuously such that a same scenery is available for different day times, seasons and weather conditions. We randomly selected 100 sceneries for each region. For the triplet sampling, we chose as positive sample an image from the same location and for the negative sample an image from a different location. Each landscape and building arrangement undergoes unique shadow, illumination and reflection properties. The generalization to unknown places under unknown illumination conditions is thus too demanding to be deduced from a single input image. Hence, we report results in Fig. 7.3 on training samples only. The model removes the illumination variations and shadows from the images. Moreover, rivers, oceans and skies as well as beaches are smoothed out. Most of the people and cars are removed and replaced by the background of the scenery. The features of the salient objects are preserved when their position remains constant in each image, e.g. see Fig. 7.3 for vehicles being removed if not contained in each image.

7.2.4 SVIRO-Illumination

For the triplet loss sampling, we chose the positive sample to be of the same class as the anchor image (but from a different scenery) and the negative sample to differ only on one seat (i.e. change only the class on a single seat w.r.t. the anchor image). Images of three empty seats do not contain any information which could mislead the network, so to make it more challenging, we did not use them as negative samples. After training, the autoencoder learned to remove

(a) Training samples

(b) Test samples

Fig. 7.2 Extended Yale Face Database B. Illumination is removed from the training samples (first row) to form the reconstruction (second row). The test samples (third row) cannot always be reconstructed reliably (fourth row). Reconstructing the nearest neighbour (NN - fifth row) preserves the head pose and position and removes the illumination.

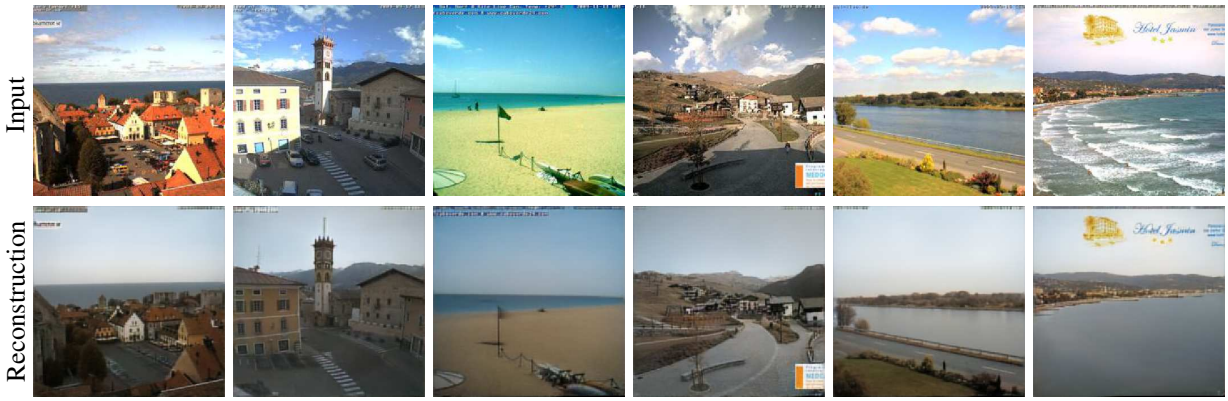


Fig. 7.3 Examples for the Webcam Clip Art dataset. The autoencoder removes the environmental features from the images (first row) to form the output images (second row). Vehicles and people are removed from the scenery and skies, rivers and beaches are smoothed out.

all the illumination and environmental information from the training images. See Fig. 7.4 for an example on how images from the same scenery, but under different illumination, are transformed. Sometimes, test samples are not reconstructed reliably. However, due to the triplet loss and nearest neighbour search, we can preserve the correct classes and reconstruct a clean image. The reconstruction from the test image latent vector produces a blurry person, which is usually a combination of several nearest neighbors.

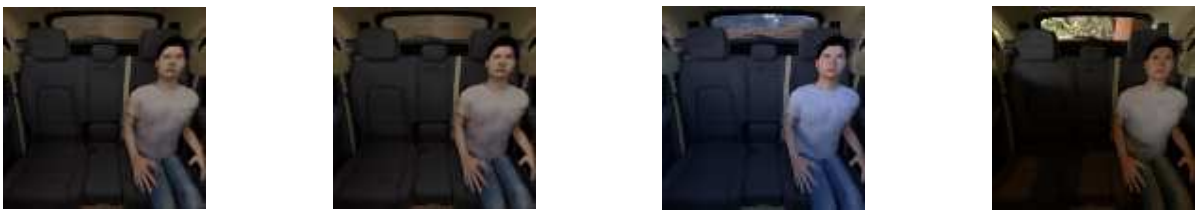
We want to emphasize that the model is not learning to focus the reconstruction to a single training image for each scenery. In Fig. 7.5 we searched for the closest and furthest (w.r.t. SSIM) input images of the selected scenery w.r.t to the reconstruction of the first input image. Moreover, we selected the reconstruction of all input images which is furthest away from the first one to get an idea about the variability of the reconstructions inside a single scenery. While the reconstructions are stable for all images of a scenery, it can be observed that the reconstructions are far, w.r.t. SSIM, from all training images. Hence, the model did not learn to focus the reconstruction to a single training sample, but instead learned to remove all the unimportant information from the input image. Finally, the texture of the salient objects is uniformly lit and smoothed out.

We compared the classification accuracy of our proposed method together with the nearest neighbour search against vanilla classification models when the same training data is being used. This way, we can quantitatively estimate the reliability of our proposed method against commonly used models. To this end, we trained baseline classification models (ResNet-50 [93], VGG-11 [225] and MobileNet V2 [219]) as pre-defined in torchvision on SVIRO-Illumination. For each epoch, we randomly selected one variation X_i^j for each scenery X^j. The classification models were either trained for 1000 epochs or we performed early stopping with a 80:20 split on the training data.

(a) Training samples

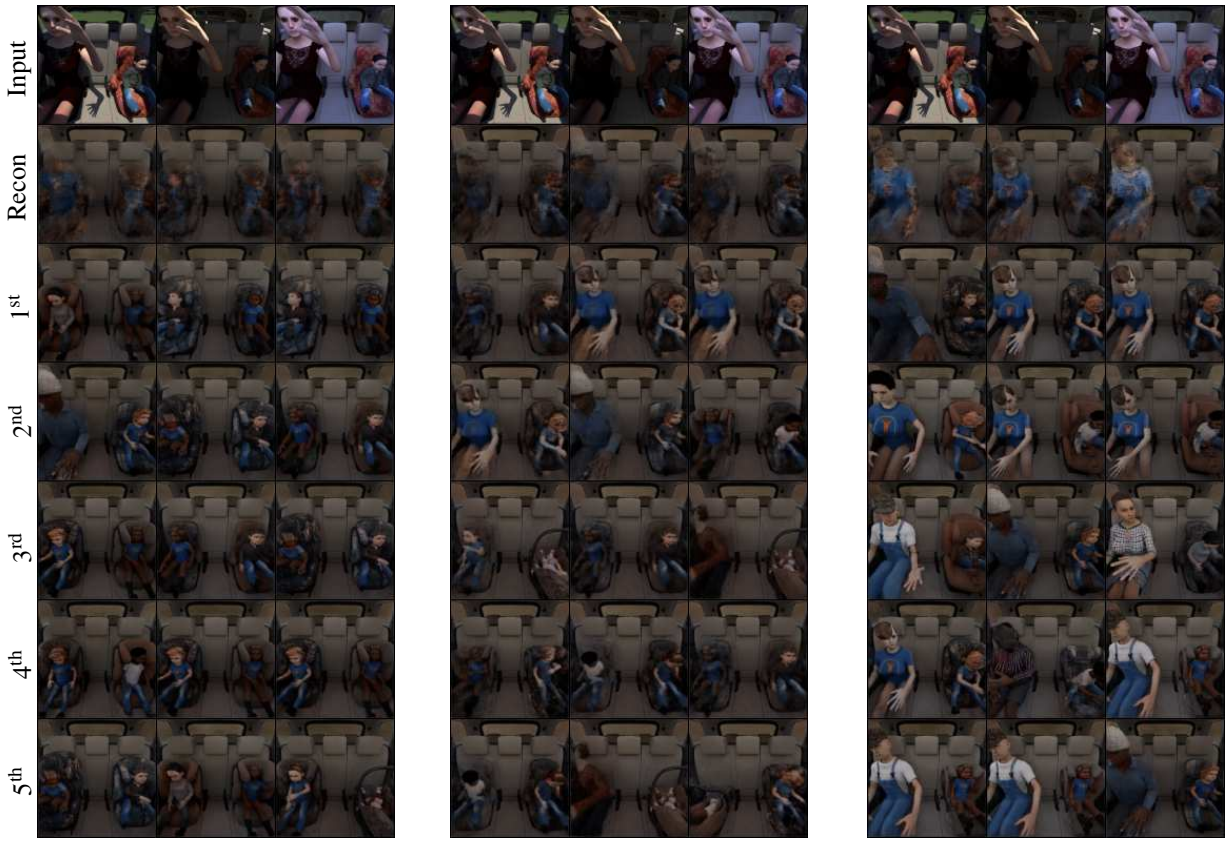
(b) Test samples

Fig. 7.4 The autoencoder model transforms the input images of a same scenery (first row) into a cleaned version (second row) by removing all illumination and environment information (see the background through the window). The test image (third row) cannot be reconstructed perfectly (fourth row). Choosing the nearest neighbour in the latent space and reconstructing the latter leads to a class preserving reconstruction (fifth row).



First recon | Max recon | Closest | Furthest

Fig. 7.5 Reconstruction of the first image of a scenery (first recon) is compared against the furthest reconstruction of all images of that scenery (max recon). First recon is also used to determine the closest and furthest images of the scenery. The model does not learn to reconstruct a training sample.



(a) I-AE (b) I-VAE (c) I-TAE

Fig. 7.6 Comparison of the reconstructions of the five nearest neighbors (rows 3 to 7) for different autoencoder latent spaces (a), (b) and (c). The reconstruction (second row) of the test sample (first row) is also reported. The I-TAE is by far the most reliable and consistent one across all five neighbors. Notice how the class changes across neighbors for I-AE and I-VAE.

Table 7.1 Mean accuracy and variance over five repeated training runs on each of the three vehicle interiors. Classification models were trained from scratch or fine-tuned and pre-trained, used early stopping with 80:20 split or no early stopping. I-PIRL improves the vanilla version and with the nearest neighbour search outperforms all other models.

			Vehicle		
Model	Pre-trained	Early stopping	Cayenne	Kodiaq	Kona
MobileNet		✓	62.9 ± 3.1	71.8 ± 4.3	73.0 ± 0.8
VGG11		✓	64.4 ± 35.0	74.0 ± 19.0	75.5 ± 5.7
ResNet50		✓	72.3 ± 3.7	77.9 ± 35.0	76.6 ± 9.9
MobileNet			72.7 ± 3.8	77.0 ± 4.1	77.4 ± 2.2
VGG11			74.1 ± 5.8	71.2 ± 14.0	78.4 ± 2.6
ResNet50			76.2 ± 18.0	83.1 ± 1.1	82.0 ± 3.2
MobileNet	✓		85.8 ± 2.0	90.6 ± 1.2	88.6 ± 0.6
VGG11	✓		90.5 ± 2.0	90.3 ± 1.2	89.2 ± 0.9
ResNet50	✓		87.9 ± 2.0	89.7 ± 6.1	88.5 ± 1.0
AE			74.1 ± 0.7	80.1 ± 1.8	73.3 ± 0.9
VAE			73.4 ± 1.3	79.5 ± 0.6	73.0 ± 0.9
TAE			90.8 ± 0.3	91.7 ± 0.2	89.9 ± 0.6
I-AE (Ours)			86.8 ± 0.3	86.7 ± 1.5	86.7 ± 0.9
I-VAE (Ours)			81.4 ± 0.5	86.6 ± 0.9	85.9 ± 0.8
I-TAE (Ours)			92.4 ± 1.5	93.5 ± 0.9	93.0 ± 0.3

We further fine-tuned pre-trained models for 1000 epochs. The triplet-based autoencoder model is being trained exactly as before. During inference, we take the label of the nearest training sample as the classification prediction. Each setup was repeated five times with five different (but the same ones across all setups) seeds. Moreover, the experiments are repeated for all three vehicle interiors. The mean classification accuracy over all five runs together with the variance is reported in Table 7.1. Our proposed method significantly outperforms vanilla classification models trained from scratch and the models' performances undergo a much smaller variance. Moreover, our proposed method outperforms fine-tuned pre-trained classification models, despite the advantage of the pre-training of these models. Additionally, we trained the autoencoder models using the vanilla reconstruction error between input and reconstruction, but using the nearest neighbour search as a prediction. Again, including our proposed reconstruction loss improves the models' performance significantly.

For visualization purposes, we trained an autoencoder, variational autoencoder and triplet autoencoder on the SVIRO-Illumination dataset with people and empty seats only. Similarly as

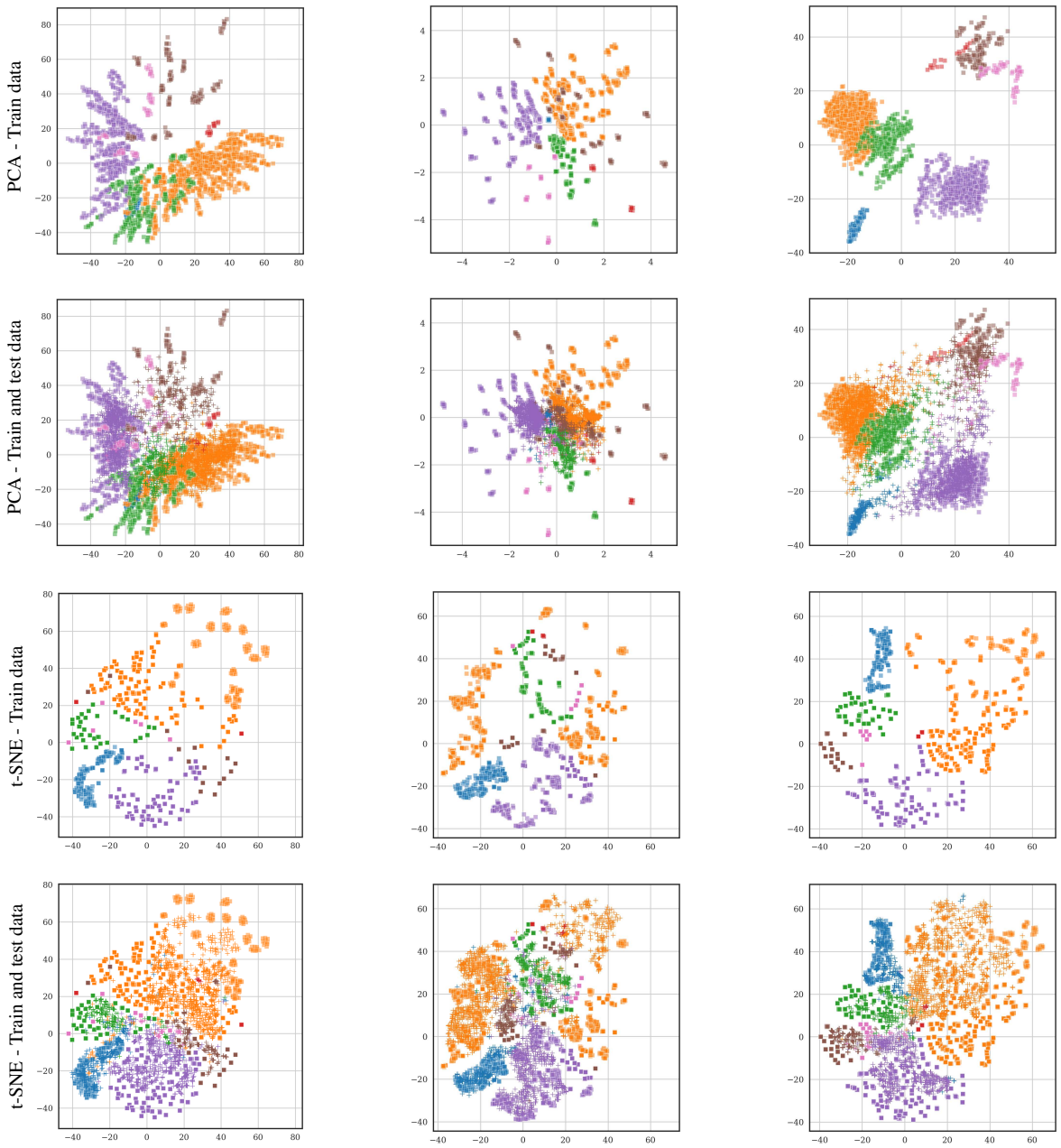
before, we chose a latent space dimension of 16. After training, we computed the latent space representation for all training samples, computed the first two principal components and the t-SNE projection and plotted the resulting projections in Fig. 7.7. The triplet-based autoencoder model separates and clusters the classes best. Both the I-AE and I-VAE contain wrong classes inside other clusters. The test data latent space representation is plotted as well. It is important to note that the comparison between I-AE, I-VAE and I-TAE is not entirely fair, because the latter implicitly uses labels during the positive and negative sample selection. Nevertheless, for the problem formulations at hand, it is beneficial to collect the classification labels considering the additional advantage of the induced L^2 norm in the latent space and improved classification accuracy.

7.2.5 ORSS

Although neither the training nor the test set of the ORSS dataset allows for a rigorous investigation of the illumination normalization, we still wanted to report some qualitative results. To this end, we trained several model variations on the training images of the X5 and applied the resulting models to the test images. Since no multiple variations per scene are available, we adopted the method of augmenting both input and target image differently. Additionally, we used the second variation of the PIRL to verify whether illumination can be removed as well. The results are reported in Fig. 7.8. Illumination is removed best when II-PIRL is used during training, however, the reconstructions are blurrier. This can be helpful in case the data does not allow the I-PIRL to be used. The training variations using different augmentations for input and target pairs did not improve normalization compared to the II-PIRL, but the reconstructions are less blurrier. Overall, it can be concluded that the II-PIRL is still a good choice to be adopted in case the I-PIRL cannot be used.

7.3 Discussion and limitations

The I-PIRL works well for image normalization on the training data, which can be sufficient for some applications, e.g. when a fixed dataset is available on which some post-processing needs to be done only. Since the generalization to test images can be achieved by a nearest neighbour search, the latter will only be useful for a subset of machine learning tasks. Our method preserves the classes for a given problem formulation, which will be fine for classification and object detection. Our method can preserve head poses to some extend, e.g. Fig. 7.2, when it is dominantly present in the images. Our approach will likely not preserve complex human poses, e.g. Fig. 7.4, or detailed facial landmarks, because the body poses and key features



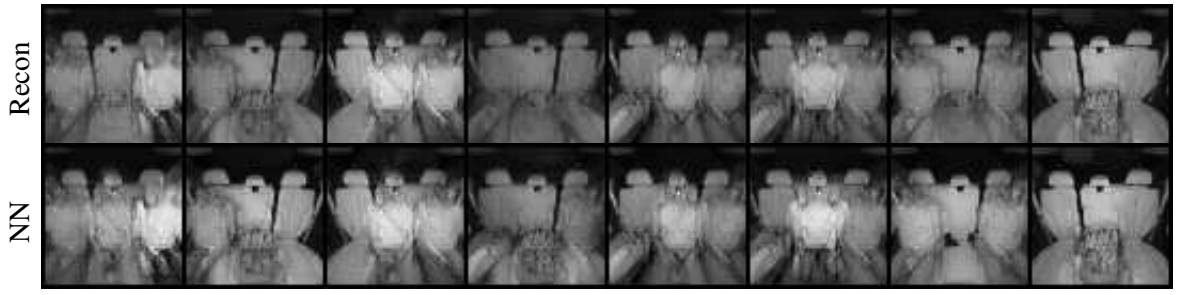
(a) I-AE (b) I-VAE (c) I-TAE

Fig. 7.7 Different autoencoders were trained with a latent dimension of 16. We report the first two principal components (PCA) and the t-SNE projection. The training and test distribution were computed and projected together. We plot the training distribution only ■ (first and third row - test points are made invisible) and the training distribution ■ together with the test distribution **+** (second and fourth row) to ease visualization. Different colors represent different classes. The projection of the I-TAE provides a good separation with almost no wrong test samples.

(a) Input images from the X5 ORSS vehicle

(b) AE with augmentations

(c) TAE with augmentations



(d) II-AE

(e) II-TAE

Fig. 7.8 Reconstructions of test images (Recon - first rows) and their nearest neighbour reconstructions (NN - second rows) for different model and training variations.

are not necessarily preserved by the nearest neighbour search. Future work should try to incorporate constraints such that the poses and landmarks of test samples are preserved as well. This is currently not working with our approach, since we only cluster images of the same class together, without differentiating between different poses. Hence the model learns to approximate the poses, i.e. by averaging them out, such that in most cases a single pose per class is learned.

In practise, it will be challenging to record identical sceneries under different illumination conditions. However, as the Extended Yale Face Database B [78] and Webcam Clip Art [137] dataset have shown, it is also feasible. Since we have highlighted the benefit of the acquisition of said datasets, the investment of recording under similar conditions in practise can be worthwhile for some applications. We believe that future work will develop possibilities to facilitate the data acquisition process. Moreover, the possibility to incorporate images taken for the same scene, but in less perfect conditions, should be explored, e.g. Fig. 7.1. Last but not least, as the results on ORSS showed, in the latter case it might be more beneficial to use the II-PIRL instead: the results will then be blurrier, but the normalization is reduced more efficiently. Although successful, I-PIRL would be a better choice, in case poses need to be preserved and the dataset allows for its applicability.

7.4 Conclusion

Our results show the benefit of recording identical sceneries under different illumination and environmental conditions such that unwanted features can be removed by a partially impossible reconstruction loss function: without the need for a ground truth target image. Our method works well for classification and post-processing tasks due to an enhanced nearest neighbour search induced by a triplet loss regularization in the latent space of an autoencoder model. We demonstrated the universal applicability of our proposed method, as long as the correct data, i.e. same scenery under different conditions, is available, on three different tasks and datasets. Moreover, our proposed method improves classification accuracy significantly compared to standard autoencoder and classification models, even when the latter was a fine-tuned pre-trained model.

Chapter 8

Synthetic to real generalization

Learning on synthetic data and transferring the resulting properties to their real counterparts is an important challenge for reducing costs and increasing safety in applications using machine learning. In this chapter, we focus on latent space representations that are invariant to inductive biases caused by the domain shift between simulated and real images showing the same scenario. We train on synthetic images only, present approaches to increase generalizability and improve the preservation of the semantics to real datasets of increasing visual complexity. We show that pre-trained feature extractors (e.g. VGG) can be sufficient for generalization on images of lower complexity, but additional improvements are required for visually more complex scenes. To this end, we demonstrate that the second variation of the PIRL leads to an improved salient feature extraction and a neglection of unimportant parts for the reconstruction and classification. This helps the generalization to real data and we further show that our approach outperforms fine-tuned classification models. Lastly, we show that combining real and synthetic images during training improves the generalization between real vehicle interiors significantly. Invariances are learned on synthetic data, in case they are modeled correctly, while the synthetic gap is reduced due to the integration of real images.

8.1 Introduction

The generation of synthetic data constitutes a cost efficient way for acquiring machine learning training data together with exact and free annotations. Notwithstanding this obvious advantage, bridging the gap between synthetic and real data remains an open challenge, in particular for camera-based applications. Learning from synthetic data is an important tool in robotics: for example, to train a quadrupedal robot on synthetic data by incorporating proprioceptive feedback [141], to train a robot hand to solve real Rubik's cubes by learning the model in a simulation only [2] or by translating the real world input data into synthetic data for a

reinforcement learning agent [275] and to *make the robot feel at home*. In view of safety critical applications, synthetic data can provide the means to reduce costs related to acquiring samples for edge cases, or which are difficult to obtain since they are too dangerous, e.g. accidents. We focus on learning invariances empirically on synthetic data, which should transfer to real data, opposed to constructing invariances as in equivariant neural networks [215].

We investigate the case of single independent images for which consistency between frames and physical interactions cannot be taken advantage of. The latter is commonly used by reinforcement learning methods [141]. We focus on training on synthetic data only, assess to what extend we can generalize to real images and we highlight which design choices improve the autoencoder models performance with respect to accuracy and reconstruction quality. To this end, we first develop a method using features of pre-trained classifiers and show that we achieve better results on MPI3D to generalize from synthetic (toy or realistic) to real images compared to autoencoder, variational autoencoder (VAE), β-VAE and FactorVAE. Although successful, we highlight that insights and design choices on a simple dataset do not necessarily transfer to real applications of higher visual complexity. To improve generalization, we propose to combine the latter design choice with the second variation of the PIRL. We extensively show that the second variation is the driving force for the improved generalization capacities. Additionally, we induce structure in the latent space by a triplet loss regularization. While we start off with MPI3D, we evaluate and justify the benefits of the different design choices on SVIRO. The latter and similar industrial applications suffer from the limited availability and variability of training data. A successful transfer from synthetic to real data would avoid the necessity of collecting real data for each vehicle interior: the invariances could be learned and improved on synthetic data only. Especially regarding the latter case, we show on ORSS that combining real and synthetic data can lead to an improved transferability towards a novel, real vehicle interior. To this end, we performed an ablation study using different synthetic datasets and compared the extractor autoencoder against the multi-channel autoencoder, both of which have their advantages and disadvantages. The extractor autoencoder achieves higher classification accuracies, but the multi-channel autoencoder removes the background on real images more efficiently, which could potentially be exploited by down-stream tasks.

8.2 Experiments and results

This section is organized in observations, formulated as subsections, which are built on one another and contain results highlighting the improvements. This provides explanations for the design choices leading to our final model architecture and cost function formulations. Improvements regarding the transfer to real images when only being trained on synthetic

images are assessed qualitatively based on reconstruction quality and latent space structure and quantitatively on classification accuracy.

We perform a baseline evaluation on MPI3D, which provides simple and realistic renderings and real counterparts, see Section 8.2.2 for more details. We reduced the dataset to contain only large objects, since even for humans the small objects cannot always be distinguished reliably. For a higher visual complexity, we use as synthetic images the SVIRO dataset and its extensions. TICaM is used to evaluate the performance on a real, publicly available dataset of a similar application, but we also report results on ORSS later in this chapter. The design choices made on MPI3D and the available synthetic images are not sufficient to obtain a good transferability to real images from the vehicle interior. We introduce step by step modifications to the autoencoder architecture, leading to steady quantitative and qualitative improvements. MPI3D and the vehicle interior share interesting properties: they have almost identical backgrounds and the environment is more tractable than many computer vision datasets. The transfer from SVIRO to TICaM is further complicated by new unseen attributes, e.g. steering wheel, since in TICaM the authors recorded images for the front while SVIRO contains images for the rear seat. An additional ablation study shows that the second variation of PIRL is the driving force for the improved generalization capacity. Finally, to be in line with common benchmark datasets, we show that our design choices also improve the transfer from training on MNIST to generalizing to real images of digits.

8.2.1 Training details

We used the same hyperparameters for all training experiments and for all autoencoder and classification models respectively. We used the AdamW optimizer with a learning rate of 0.0001 and weight decay of 0.00001. We used a batch size of 64 and the only augmentation performed was a random horizontal flip for the classification task. All models were trained for 100 epochs on MPI3D and 250 epochs for the other datasets. We used a latent space dimension of 64 for all models trained on the vehicle interior and a latent space dimension of 10 for the MPI3D dataset. In case of a triplet loss, we used the swap parameter of PyTorch to make the negative mining more challenging [15]. As a positive sample, we selected an image of a different scenery of the same class, i.e. the same objects are at the same seat position. For the negative sample we selected a scenery which differs in a single seat position and we did not allow sceneries with empty seats only. In case the II-PIRL was used, the target images for the positive and negative samples are chosen to be partially impossible as well.

Both the extractor autoencoder and the classification models used the same layer for extracting the features from the pre-trained models. In both cases and for all pre-trained models we used layer level -3 in our implementation: those features were used to fine-tune the rest of

the pre-trained classification model or to train from scratch our added autoencoder layers. In all cases, we interpolated the input images to be of size 224×224 and copied the single grayscale image channel twice along the channel dimension.

8.2.2 MPI3D

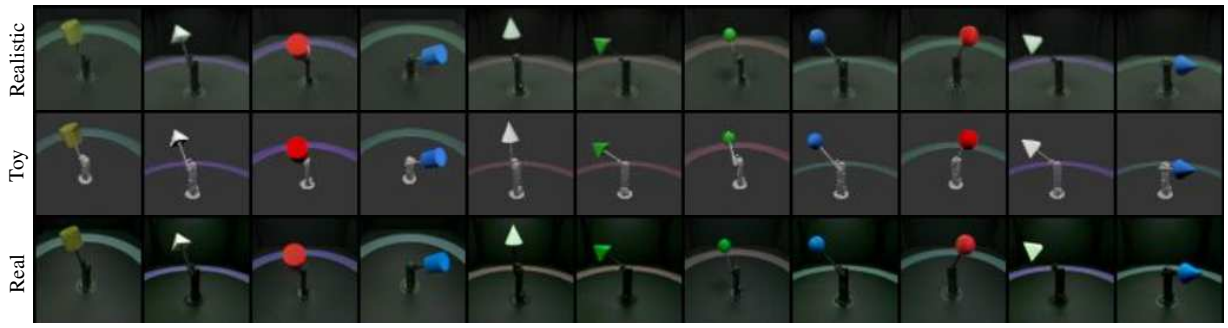
MPI3D consists of three dataset variations: 1) real images recorded on a mechanical platform, 2) realistic rendering of the same scenes as in 1) and 3) simplified (toy) renderings of the same scenes as in 1). The sceneries show a robotic arm moving 3D objects under seven factors of variation: 1) six different shapes, 2) six colors, 3) two sizes (although we consider the large size only), 4) with three different colors for the background circle, 5) from three different camera heights, 6) for forty different positions for the x-axis and 7) forty positions for the y-axis. Since the images were recorded in a controlled setting, it was possible to replicate each scenery in the simulation software. For the latter, the rendering was either chosen to be realistic and hence close to the real recordings, or simplified and hence an abstraction of reality. This enables detailed investigations for representation learning across simulated and real environments. Since the sceneries are the same across the three different levels of realism, the dataset allows to compare rigorously whether a model trained on one dataset variation can transfer to a different level of realism for exactly the same scenes. Hence, the dataset provides a good starting point to test the transfer from synthetic to real images. Example images and comparisons of the same scenes across the three dataset variations are reported in Fig 8.1a.

Autoencoders struggle on real images when trained on synthetic images

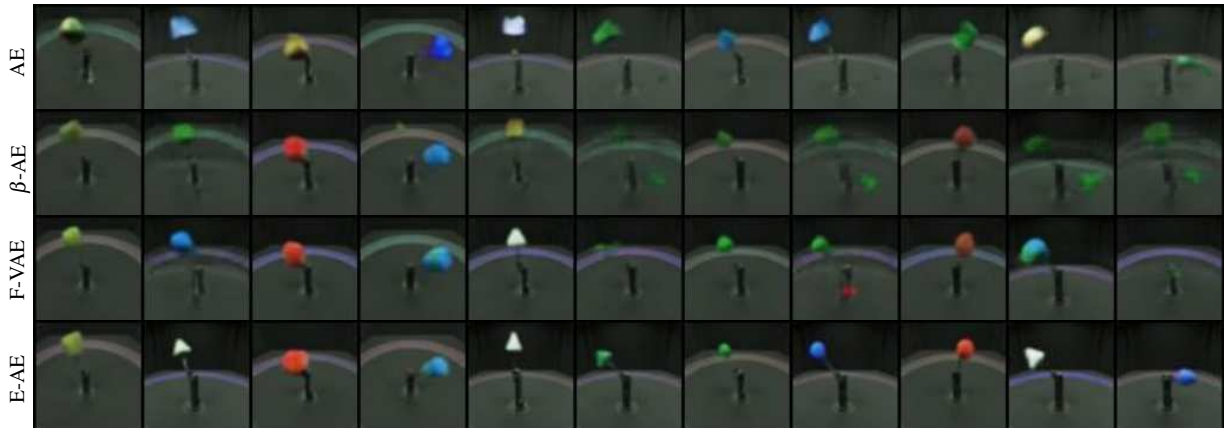
In the first, albeit naïve experiment we assumed that due to the bottleneck of autoencoders, the latter should generalize to some extent to real images when trained on synthetic ones. We trained convolutional autoencoders on the toy and realistic MPI3D images, respectively, and evaluated the resulting models on the real recordings. The first row of Fig. 8.1b shows the reconstruction of real images when trained on the realistic synthetic images: the model preserves some of the semantics. The model fails to perform senseful reconstructions when trained on toy images, see Fig. 8.1c.

Autoencoders overfit to the synthetic distribution

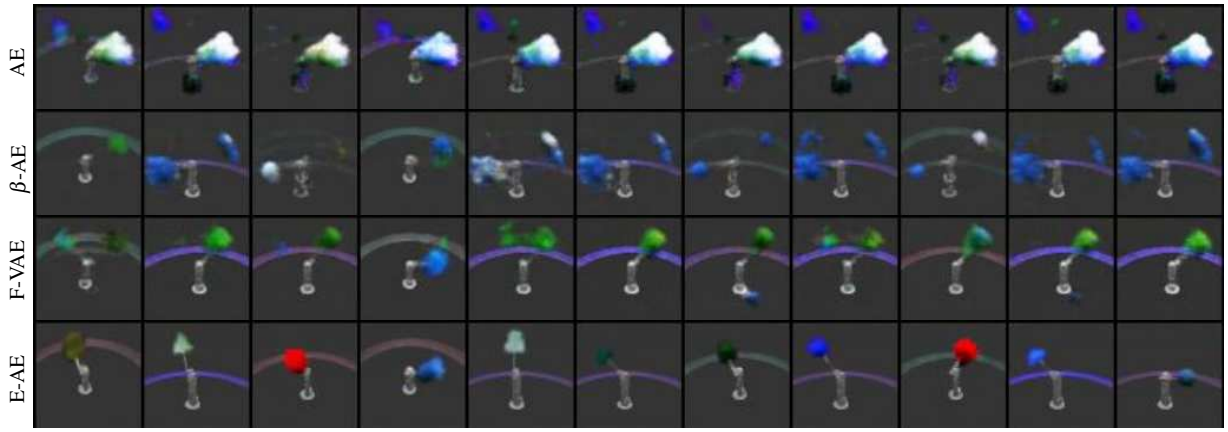
A consequence of the results of the previous section is the assumption that the autoencoder overfits to the synthetic distribution and takes into consideration some artefacts (e.g. rendering noise). We followed the idea of the MPI3D authors Gondal et al. [81] and trained variational



(a) Realistic and toy are splits of synthetic datasets used for training, respectively. Real are real images of the same scenes used as input after training and used for evaluating the synthetic to real transfer.



(b) Reconstruction of real data when being trained on realistic data.



(c) Reconstruction of real data when being trained on toy data.

Fig. 8.1 Reconstruction of unseen real data for different autoencoders: Autoencoder (AE), β Variational Autoencoder (β-VAE), FactorVAE (F-VAE), Extractor Autoencoder (E-AE).

autoencoder (VAE), β-VAE and FactorVAE on the same data as before using the BCE reconstruction loss. The results in the second (β-VAE with $\beta = 8$) and third (FactorVAE with $\gamma = 50$) row of Fig. 8.1b show that the models reconstruct real images better and more of the semantics are preserved. If trained on toy renderings, the representation gap is too large, causing the reconstruction of the real images still to be bad: see Fig. 8.1c. This is in line with the results from Gondal et al. [81].

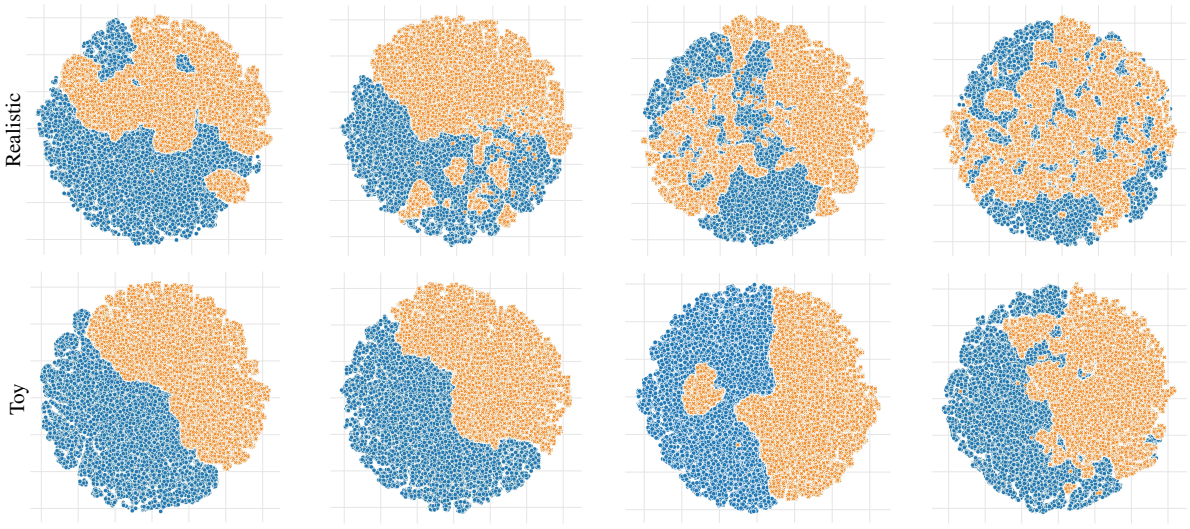
More general input features improve reconstructions

A small gap between the synthetic and real distribution can potentially be closed by a dedicated data augmentation approach to avoid overfitting to synthetic artefacts. Nevertheless, an abstraction from toy to real images cannot be achieved by means of simple data transformations or model constraints (e.g. denoising autoencoder). To this end we propose to use a pre-trained feature extractor as presented in Section 4.5 and as defined by Eq. (4.7). We used the VGG-11 model pre-trained on ImageNet as the extractor if not stated otherwise.

The results from the fourth row of Fig. 8.1b and Fig. 8.1c, respectively, show that the proposed modifications enable the model to generalize to real images when trained on synthetic ones. Much more of the semantics are preserved even when the model was only trained on toy images. Our method produces semantically more correct and less noisy reconstructions compared to the VAE and FactorVAE baseline results. Additional qualitative improvements are highlighted by visualizing the latent space: both the 10-dimensional training (synthetic) and test (real) data latent spaces are projected together into a 2-dimensional representation using t-SNE. In Fig. 8.2 (first row) we can observe that VAE and FactorVAE improve the representation of real and synthetic images in the same region in the latent space, however, only partially, indicating a different representation for real and synthetic images. When using E-AE, real and synthetic images are represented more similarly in the latent space and the clusters are completely overlapping. Even when trained on the toy dataset, the latent space representation for synthetic and real images produced by E-AE overlaps partially as visualized in Fig. 8.2 (second row). We report in Table 8.1 a quantitative evaluation between the reconstructions of the real images against their synthetic training counterparts across all dataset images for different norms. We also compute the error between the real input images and their reconstruction to measure whether the semantics are being preserved: in all cases E-AE performs best. Reconstructions of training data are reported in Fig 8.3. The latter shows that all models perform similarly well on the training data, hence the training was successful, but our proposed design choices generalize best to the real images.

Table 8.1 We report the L1, SSIM and LPIPS norm between the reconstructions of the real images (which are unknown) and themselves (Real), or between the corresponding synthetic (Synth.) training images. For the latter we abbreviate realistic (R) or toy (T). We report the mean of the norms across the entire dataset: for SSIM larger ↑ and for the others smaller ↓ is better. E-AE performs best across all evaluations.

			L1 ↓		SSIM ↑		LPIPS ↓	
	Model	Variant	Synth.	Real	Synth.	Real	Synth.	Real
T	AE	SSIM	932	1763	0.56	0.42	0.35	0.40
T	VAE	BCE	659	1497	0.50	0.33	0.34	0.42
T	β-VAE	BCE, $\beta = 4$	710	1542	0.53	0.38	0.31	0.44
T	β-VAE	BCE, $\beta = 8$	406	1321	0.71	0.48	0.26	0.37
T	FactorVAE	BCE, $\gamma = 10$	521	1288	0.66	0.45	0.26	0.39
T	FactorVAE	BCE, $\gamma = 50$	430	1295	0.71	0.51	0.22	0.35
T	E-AE (Ours)	SSIM	177	1165	0.90	0.58	0.10	0.28
R	AE	SSIM	568	1133	0.83	0.62	0.20	0.24
R	VAE	BCE	482	890	0.74	0.61	0.20	0.23
R	β-VAE	BCE, $\beta = 4$	372	833	0.81	0.64	0.18	0.20
R	β-VAE	BCE, $\beta = 8$	384	854	0.79	0.64	0.19	0.21
R	FactorVAE	BCE, $\gamma = 10$	218	734	0.88	0.68	0.15	0.19
R	FactorVAE	BCE, $\gamma = 50$	391	830	0.78	0.64	0.16	0.18
R	E-AE (Ours)	SSIM	251	841	0.92	0.70	0.08	0.14



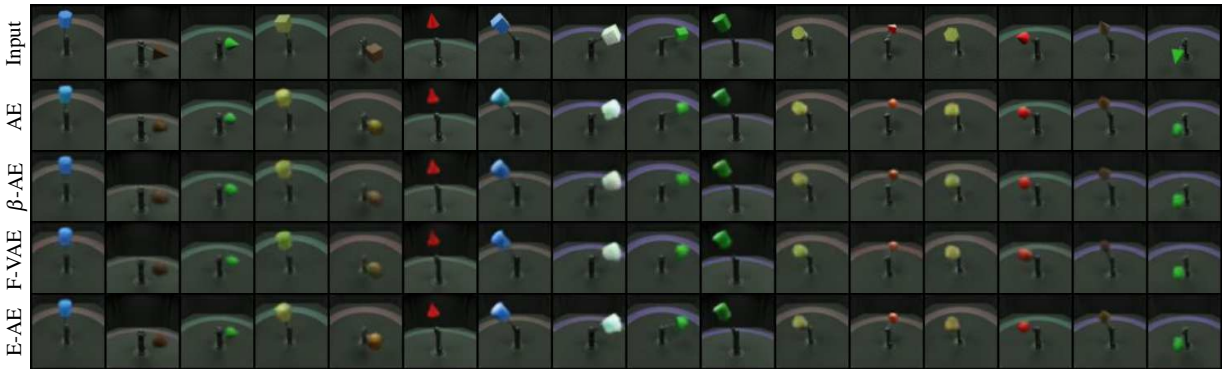
(a) AE (b) β-VAE (c) FactorVAE (d) E-AE (Ours)

Fig. 8.2 t-SNE projection of the 10 dimensional latent space representation by different autoencoders when trained on the realistic (first row) or toy (second row) training (blue circle) images. The resulting representation for real (orange cross) images is plotted as well. The extractor approach is the only method clustering synthetic and real images together when trained on realistic renderings. When trained on toy images, our extractor approach performs still best although the synthetic-real distributions are not as overlapped as if trained on realistic images.

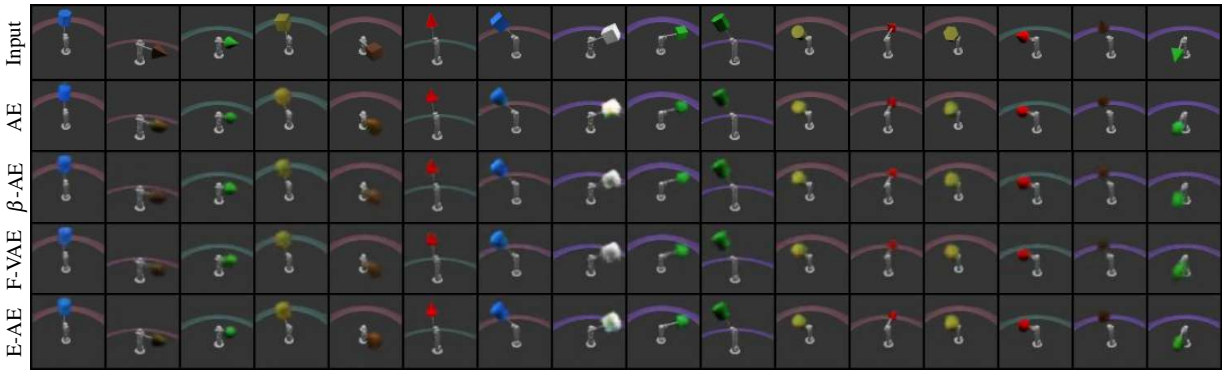
8.2.3 SVIRO to TICaM

We only used the grayscale training images from the SVIRO dataset. We considered everyday objects as background and removed all images containing empty child and infant seats. For the classification evaluation we used all the images from all the different vehicles, but we used training images only. Occupancy classification is performed on the entire image such that all three seats need to be classified simultaneously.

For TICaM, we used all training and test images and also flipped the images for the classification evaluation. This was done, because otherwise the class variability is quite low and there is a strong bias towards people sitting on the right driver seat. Moreover, the steering wheel would always be placed at the same right position. We also needed to perform some pre-processing to make the real TICaM images compatible with the synthetic images. First, we adapted the labels to be in accordance with the ones from SVIRO reported in Table 3.3: we extracted the labels for the left and right seat from the filename. The file name is split at the character _ after which the third (right seat) and ninth (left seat) part is responsible for the class definition. If the latter was a 0 or contained an o, we kept it as a 0. If it contained a p, it was changed into a 3. We changed the value to 2 if it was one of the child seats $s03$, $s13$, $s04$, $s14$



(a) Reconstruction of training data when being trained on realistic data.



(b) Reconstruction of training data when being trained on toy data.

Fig. 8.3 Reconstruction of realistic and toy training data for different autoencoders: AE, β-VAE, FactorVAE (F-VAE) and Extractor Autoencoder (E-AE). All models perform equally well on the training data, yet our approach generalizes best to real images.

or the variation $g00$ for the child seats $s01$, $s11$, $s02$, $s12$. In all other cases, it was transformed to a 1, i.e. for the child seats $s05$, $s15$, $s06$, $s16$ and variations $g01$, $g11$, $g10$ for $s01$, $s11$, $s02$, $s12$. Second, the illumination of the images was normalized using a histogram equalization. After that the images were cropped at height position 120 with height 300 and left position 106 with width 300. Finally, the images were resized to 128 pixels.

It works for visually simple images - More is needed on more complex data

Since the method introduced in the previous section achieved good results, even when being trained on toy images, we wanted to apply it to images of higher visual complexity, e.g. a vehicle interior. We trained the same model architecture, but with a 64-dimensional latent space, on images from the Tesla vehicle from SVIRO and the Kodiaq vehicle from SVIRO-Illumination, respectively, and evaluated the model on the real TICaM images. Examples of the resulting model's reconstructions are plotted in Fig. 8.4 (b). In both cases only blurry human models are reconstructed, which is similar to the mode collapse in the first row of Fig. 8.1c. We concluded that more robust features are needed.

PIRL helps generalization

As defined in Eq. (4.2) the first variation of the PIRL has proven to work well for image normalization [54]. We hypothesized that the same approach could lead to a better generalization to real vehicle interiors. We applied this strategy to variations of the same scene under different illumination conditions, but realized that the learned invariances are not suitable for the transfer between synthetic and real. An example is provided in Fig. 8.4 (c) where we trained on the Kodiaq images from SVIRO-Illumination.

We concluded that, for learning more general features by applying the PIRL, we needed input-target pairs where both images are of the same scene, but differ in the properties we want to become invariant to: the dominant background. To this end we created and used SVIRO-NoCar, as presented in Section 3.4. During training, we randomly select two images per scene and use one as input and the other as target, i.e. as defined in Eq. (4.2). When applied to real images, see Fig. 8.4 (e), the model preserves the semantics better of the real images: the model starts to reconstruct child seats and not people only, anymore. We also trained a model without the PIRL to show that the success is not due to the design choice of the dataset: in Fig. 8.4 (d) the model performs worse.

Finally, we extended this idea further with our second PIRL formulation: instead of taking the same scene with a different background as target image, we randomly selected a different scene of the same class, e.g. if a person is sitting at the left seat position, we would take

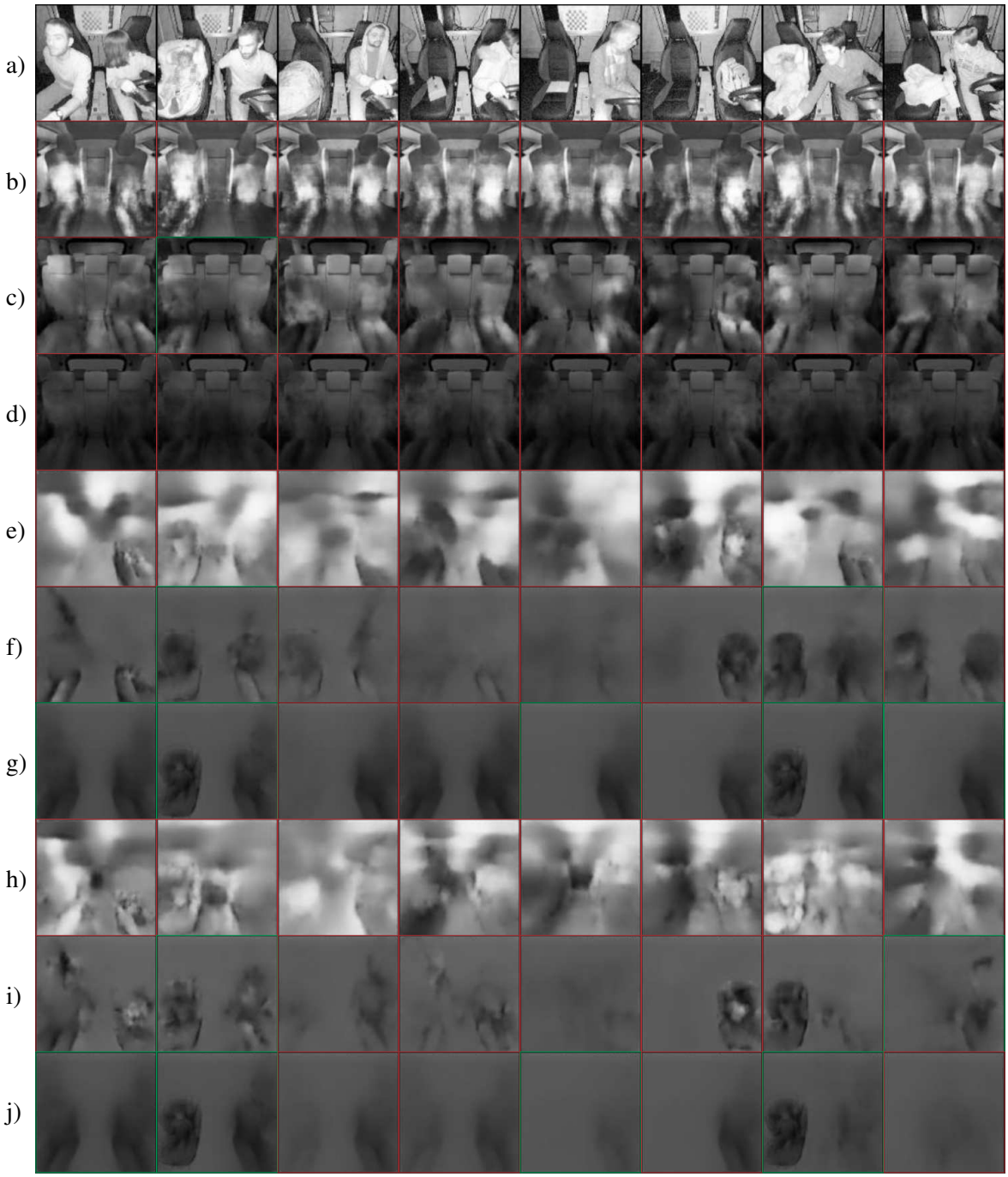


Fig. 8.4 Reconstruction results of unseen real data a) from the TICaM dataset: b) E-AE Trained on Tesla SVIRO, c) E-AE Trained on Kodiaq SVIRO-Illumination , d) I-E-AE Trained on Kodiaq SVIRO-Illumination , e) E-AE, f) I-E-AE, g) II-E-AE, h) E-TAE, i) I-E-TAE, j) II-E-TAE. Examples e)-j) are all trained on SVIRO-NoCar. A red (wrong) or green (correct) box highlights whether the semantics are preserved by the reconstruction.

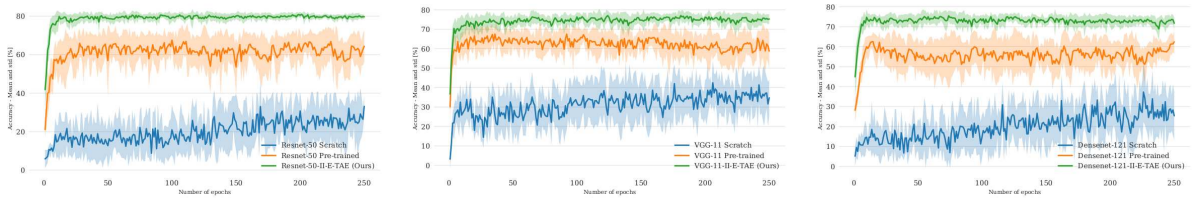
another image with a person on the left seat, potentially a different person with a different pose. This approach is formulated in Eq. (4.3). While this leads to a blurrier object reconstruction, which is expected because the autoencoder needs to learn an average class representation, the classes are preserved more robustly and the reconstructions look better than before, see Fig. 8.4 (f). Moreover, this additional randomization improves classification accuracy as discussed in Section 8.2.3 and in Section 8.3.

Structure in the latent space helps generalization

The final improvement is based on the assumption that structure in the latent space should help the model performance. Class labels are included by formulating a triplet loss regularization to the latent space representation as defined by Eq. (4.4): images of the same class should be mapped closely together and images of different classes should be pushed away. The triplet loss induces a more meaningful L^2-norm in the latent space [54] such that a k-nearest neighbour (KNN) classifier can be used in the next section. As the results of Fig. 8.4 (g) show, these final improvements, together with the previous changes, yield the semantically most correct reconstructions. The triplet loss without the PIRL is not sufficient and in Section 8.3 we show that the II-PIRL is the driving force for the improved performance.

KNN with triplet loss outperforms classification models

We investigated whether the qualitative improvements also transfer to a quantitative improvement. We took the most basic approach: we combined the E-TAE with a k-nearest neighbour classifier in the latent space and used SVIRO-NoCar for training. We retrieve the latent space vectors for all flipped training images as well and used only a single image per scene (i.e. not all 10 variations). According to a common rule of thumb, we choose $k = \sqrt{N} = 115$, where N is the size of the training data together with its flipped version [113]. The model should classify occupancy (empty, infant, child or adult) for each seat position and we used the same hyperparameters for all methods and variations thereof. We froze the same layers of the pre-trained models for fine-tuning the later layers in case of classification models or to train our autoencoder using it as an extractor. We evaluated the model performance after each epoch on the real TICaM images (normal and flipped images of the training and test splits) for both the autoencoder and the corresponding classification model. This provides a measure on the best possible result for each method, but is of course not a valid approach for model selection. We report in Fig. 8.5 the training results for seeds 1 to 10 and summarize the training performance by plotting the mean and standard deviation per epoch per method. Our approach converges more robustly and consistently to a better mean accuracy. For each experiment,



(a) ResNet-50 (b) VGG-11 (c) DenseNet-121

Fig. 8.5 Comparison of the training performance distribution for each epoch over 250 epochs. II-E-TAE (green) is compared against training the corresponding extractor from scratch (blue) or fine-tuning (orange) the layers after the features used by the extractor in our autoencoder approach. Our approach converges much faster to a better accuracy with a smaller standard deviation.

we retrieve the best accuracy across all epochs and compute the mean, standard deviation and maximum of these values across all runs: these statistics are reported in Table 8.2. The model weights corresponding to the epochs selected by the previous heuristics were applied on the SVIRO dataset to verify whether the learned representations are universally applicable to other vehicle interiors. For SVIRO, we used the training images and excluded all images containing empty child seats or empty infant seats and treated everyday objects as background. The results show that our E-AE significantly outperforms the classification models across three different pre-trained models and across all datasets. A consistent improvement for the different modifications is achieved: I-E-TAE outperforms E-TAE and II-E-TAE outperforms I-E-TAE.

8.2.4 MNIST to real digits

The improvements reported in this chapter are not limited to the application in the vehicle interior. To this end, we trained models using the same design choices on MNIST [140] and evaluate the generalization onto real digits [50] in Fig. 8.6 and Table 8.3: the quantitative and qualitative results show that the different design choices induce similar improvements as for the vehicle interior. The autoencoder models were trained for 20 epochs using a latent dimension of 64 and MSE reconstruction loss.

8.2.5 SVIRO to ORSS

Since the design choices developed in the previous sections improved the performances on MNIST and TICaM, we also wanted to evaluate it on the ORSS images. To this end we trained the models exactly as in the previous section on SVIRO-NoCar and evaluated it after 250 epochs on the Sharan images from the ORSS dataset. All hyperparameters are identical to the investigation on TICaM and we used the ResNet-50 backbone as extractor, since it was the best

Table 8.2 For each experiment, the best accuracy on real TICaM images across all epochs is taken and the mean, standard deviation and maximum of those values across all 10 runs is reported. The model weights achieving maximum performance per run on TICaM are evaluated on SVIRO. Our approach outperforms the classification models significantly.

		TICaM		SVIRO	
Model	Variant	Mean	Max	Mean	Max
VGG-11	Scratch	58.5 ± 4.0	64.6	65.6 ± 5.4	72.7
ResNet-50	Scratch	53.3 ± 3.5	60.4	56.4 ± 2.6	59.3
DenseNet-121	Scratch	56.3 ± 5.5	62.1	68.8 ± 2.4	74.9
VGG-11	Pre-trained	75.5 ± 1.5	78.0	78.7 ± 2.9	84.0
ResNet-50	Pre-trained	78.1 ± 1.7	80.4	83.5 ± 2.7	88.1
DenseNet-121	Pre-trained	72.2 ± 4.2	77.4	85.0 ± 2.3	88.0
VGG-11	E-TAE (Ours)	76.7 ± 2.3	81.5	78.6 ± 2.6	82.3
ResNet-50	E-TAE (Ours)	83.8 ± 1.3	86.0	85.8 ± 2.4	89.1
DenseNet-121	E-TAE (Ours)	78.5 ± 2.4	81.8	86.7 ± 1.3	88.2
VGG-11	I-E-TAE (Ours)	79.7 ± 2.1	82.2	80.9 ± 4.0	85.6
ResNet-50	I-E-TAE (Ours)	83.5 ± 1.3	85.6	89.2 ± 1.0	90.3
DenseNet-121	I-E-TAE (Ours)	77.2 ± 1.7	79.3	90.4 ± 1.3	92.1
VGG-11	II-E-TAE (Ours)	81.0 ± 0.6	82.0	79.1 ± 3.9	84.8
ResNet-50	II-E-TAE (Ours)	83.7 ± 0.5	84.5	93.0 ± 0.8	94.1
DenseNet-121	II-E-TAE (Ours)	79.3 ± 1.3	81.5	89.9 ± 1.8	92.3

Table 8.3 Different model architecture variations were trained on MNIST. Then different classifiers were trained on the latent space representation of the training data and evaluated on real images of digits.

Model	KNN	Random Forest	SVM
AE	15.7	12.5	11.6
TAE	11.1	11.6	8.4
II-AE (Ours)	27.8	20.2	23.6
II-TAE (Ours)	21.8	17.9	23.9
E-AE (Ours)	27.3	23.1	26.5
E-TAE (Ours)	26.1	19.1	23.3
II-E-AE (Ours)	65.	61.9	65.6
II-E-TAE (Ours)	64.1	63.7	63.7

Fig. 8.6 Reconstruction of real input images of digits by models trained on MNIST. Similar to the vehicle interior, the II-PIRL provides the best class preserving reconstructions.

performing one in the previous section. The results are presented in Table 8.4. Out of the box, the design choices of the previous section do have a positive effect on the transferability. This highlights once again that investigations and insights made on SVIRO, and here combined with TICaM, provide good directions for the real industrial application. However, the performance is, unfortunately, still far from satisfactory, which is why we performed a more detailed ablation study by incorporating real data as well.

Including real images

Learning from synthetic data only and generalizing to real images is a hard task, particularly if invariances and the robustness to be learned require an extensive and expensive collection of real images. However, usually at least some real data is available during training, otherwise the transferability can not be evaluated. In our use-case the framework is special: we could use real data from one vehicle interior, combine it with synthetic data and hope for a good performance on real images from a different vehicle interior. In that setting, it would be desirable to learn invariances, e.g. with respect to the background and vehicle interior, by the synthetic data and learn to close the gap between synthetic and real images by using the real images from one vehicle interior only. This way one could reduce the necessity of collecting real images for each new vehicle interior. It is possible to combine synthetic and real images naively by shuffling them together or to use the multi-channel autoencoder approach, as presented in Section 4.7.

Table 8.4 Ablation study for different training dataset combinations and autoencoder model designs. We report results for extractor autoencoders (E-AE) and multi-channel autoencoder (MuCh) model variations. The mean accuracy and standard deviation on the previously unseen Sharan vehicle from ORSS over 10 runs is reported. We also report results when the child seat class is excluded (No CS). Incorporating SVIRO-NoCar (SVIRO-NC) always leads to better results than using SVIRO-Uncertainty (SVIRO-U).

				Sharan		Sharan - No CS	
Model	SVIRO-NC	SVIRO-U	ORSS	1-NN	MLP	1-NN	MLP
E-AE	✓			4.8 ± 0.9	7.4 ± 1.6	6.9 ± 1.5	12.6 ± 3.4
E-TAE	✓			37.2 ± 3.2	38.3 ± 2.8	67.6 ± 6.1	69.9 ± 4.2
I-E-AE	✓			38.8 ± 1.4	40.7 ± 1.9	68.0 ± 2.6	67.2 ± 3.8
I-E-TAE	✓			36.7 ± 1.1	39.9 ± 1.4	66.2 ± 3.4	69.9 ± 2.2
E-AE			✓	13.7 ± 2.4	16.0 ± 1.6	18.9 ± 4.3	13.5 ± 4.9
E-TAE			✓	72.4 ± 3.0	72.1 ± 3.9	83.3 ± 4.8	82.1 ± 6.1
I-E-AE			✓	40.8 ± 7.0	50.2 ± 7.1	48.2 ± 8.5	59.0 ± 7.5
I-E-TAE			✓	74.7 ± 5.7	75.7 ± 5.8	88.2 ± 4.4	88.9 ± 3.6
E-AE		✓	✓	13.0 ± 1.6	18.1 ± 3.9	21.1 ± 4.5	18.6 ± 7.2
E-TAE		✓	✓	62.7 ± 5.3	66.2 ± 5.6	80.5 ± 2.6	85.3 ± 3.3
I-E-AE		✓	✓	30.2 ± 8.0	41.6 ± 9.1	55.2 ± 14.4	61.5 ± 12.1
I-E-TAE		✓	✓	69.4 ± 4.1	73.8 ± 4.0	82.9 ± 3.3	90.5 ± 3.0
E-AE	✓		✓	4.0 ± 0.7	11.5 ± 2.3	6.2 ± 1.2	12.5 ± 2.4
E-TAE	✓		✓	75.0 ± 2.4	75.8 ± 2.5	93.0 ± 1.5	93.9 ± 1.1
I-E-AE	✓		✓	35.6 ± 9.4	64.4 ± 7.6	53.9 ± 14.4	84.6 ± 10.4
I-E-TAE	✓		✓	75.2 ± 3.4	76.1 ± 3.4	94.4 ± 1.5	94.8 ± 1.4
MuCh		✓	✓	12.0 ± 1.4	11.7 ± 0.8	13.0 ± 2.8	11.9 ± 1.5
T-MuCh		✓	✓	13.1 ± 1.3	12.3 ± 1.1	21.2 ± 3.9	23.3 ± 5.2
I-MuCh		✓	✓	10.5 ± 0.9	10.7 ± 1.1	21.1 ± 3.0	21.6 ± 2.5
I-T-MuCh		✓	✓	13.0 ± 1.3	12.1 ± 1.4	23.0 ± 3.3	27.2 ± 4.2
MuCh	✓		✓	4.6 ± 0.7	8.9 ± 1.5	6.7 ± 0.9	10.2 ± 2.5
T-MuCh	✓		✓	28.0 ± 1.8	29.0 ± 1.7	47.7 ± 2.8	49.3 ± 3.0
I-MuCh	✓		✓	20.1 ± 2.4	26.0 ± 2.6	31.7 ± 3.3	40.2 ± 3.6
I-T-MuCh	✓		✓	29.1 ± 3.3	29.4 ± 3.2	49.4 ± 5.4	49.7 ± 4.8
E-MuCh		✓	✓	21.7 ± 3.2	24.1 ± 4.2	35.3 ± 5.8	36.0 ± 6.7
E-T-MuCh		✓	✓	65.3 ± 3.6	67.3 ± 4.2	81.3 ± 4.0	86.5 ± 3.4
I-E-MuCh		✓	✓	36.0 ± 11.3	36.5 ± 11.7	59.0 ± 16.4	66.8 ± 12.0
I-E-T-MuCh		✓	✓	67.3 ± 4.1	69.4 ± 4.0	86.2 ± 2.8	91.0 ± 2.1
E-MuCh	✓		✓	9.6 ± 1.6	16.8 ± 2.9	9.9 ± 2.3	17.0 ± 4.0
E-T-MuCh	✓		✓	73.9 ± 3.4	75.0 ± 3.3	89.4 ± 1.0	90.7 ± 1.0
I-E-MuCh	✓		✓	66.7 ± 2.6	70.5 ± 2.4	88.9 ± 1.4	89.3 ± 1.8
I-E-T-MuCh	✓		✓	72.8 ± 3.0	74.1 ± 3.4	90.1 ± 1.8	90.8 ± 1.6

The latter can also be combined with the extractor and II-PIRL design choices. In the following we will report results for both cases.

Shuffling real and synthetic images

In this first approach, we re-used the same extractor model architecture as in Section 8.2.5, but we shuffled the real images from the X5 of ORSS together with SVIRO-NoCar or SVIRO-Uncertainty. For the latter case, we combined all the different training and test splits of SVIRO-Uncertainty without everyday objects and used them for training. Lastly, as a baseline result, we also trained the same model using real images only. We use exactly the same hyperparameters as for the extractor autoencoder approach from the previous sections. This way we can easily compare the performances across the different design choices.

We trained a nearest neighbor (1-KNN) and single hidden layer MLP classifier in the latent space being of the same dimension. The latter is trained after the autoencoder model has finished training and it uses both the training data and the corresponding flipped versions for the synthetic and real images. The models are then evaluated on the Sharan images from ORSS and reported in Table 8.4. The results indicate a few interesting observations. The design choice of SVIRO-NoCar indeed induces, at least partially, invariances with respect to the vehicle background. This becomes particularly obvious by comparing against the results when SVIRO-Uncertainty is used instead. Hence, a correctly designed synthetic dataset can improve important properties when applied on real images. The latter becomes even more apparent in case the child seat class is left out - we will discuss this important point in Section 8.2.6. The results indicate that the extractor autoencoder model, when combined with a correctly designed synthetic dataset, can improve the transferability to a new vehicle interior. Nevertheless, we wanted to compare this approach against a model architecture dedicated to combine both real and synthetic images during training.

Multi-channel autoencoder

Many methods combining real and synthetic images expect real-synthetic pairs being close to one another. However, in our case, we randomly sample for each epoch a real image of the same class for each synthetic image. Moreover, except for the class, nothing else needs to match, e.g. the poses can be different. While this is much more challenging, since for each epoch we see a different real image for each synthetic image, it is not always possible to have matching real-synthetic image pairs. Our approach requires a less demanding synthetic-real correspondence and yet still performs well. It is also possible to fix a real-synthetic image pair and use it for the entire training process, however, this had a detrimental effect for our use case.

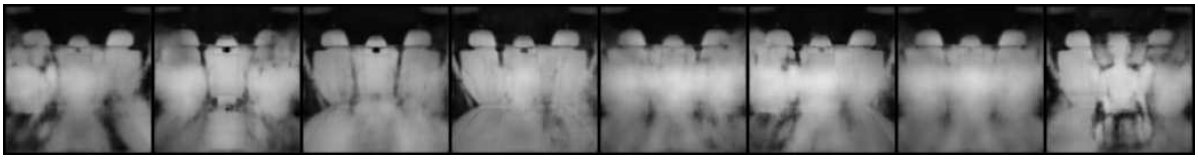
The model architecture and the training details are the same as before: we only define the decoder twice, one for the real images and one for the synthetic ones. The models were trained by combining the X5 images from ORSS with the SVIRO-NoCar or the SVIRO-Uncertainty images. For each synthetic image being sampled, we select a real image being of the same class. Both the reconstruction of the synthetic and real input image are compared against a same synthetic image. The latter target depends on the design choice, e.g. it can be the same as the input image or a new one in case of II-PIRL. In case the triplet loss is used, we use the latent space representation for real and synthetic images interchangeably. The latter design choice enforces both distributions to be represented closer and more similarly. For each model design choice, the experiment is repeated for 10 runs. The results of our exhaustive ablation study are reported in Table 8.4. As the results show, the multi-channel autoencoder combined with the extractor and the II-PIRL improves the performance up to a point where it is similar to the vanilla extractor autoencoder model. This shows that our previously defined extractor autoencoder model can already exploit the availability of real and synthetic images sufficiently well.

The multi-channel autoencoder allows for different choices to be considered as the target image for the reconstruction loss. As a reminder, we use a single encoder for encoding both synthetic and real images and we use two decoders, one for the synthetic and one for the real input images. While it would be possible to force both decoders to reconstruct real images only, or to reconstruct real and synthetic images respectively, this has a detrimental effect, as highlighted by the reconstructions later on. The reason for the latter is the fact that the autoencoder could then learn to represent the real and synthetic image slightly differently in the latent space and exploit this effect during the reconstruction. Particularly in the case where the II-PIRL is used to learn invariances using the synthetic images, it is more beneficial to use both times the synthetic images as target. This is illustrated in Fig. 8.7: we compare the reconstruction performance in case different target distributions were used during training. If the extractor multi-channel autoencoder with the II-PIRL and triplet loss is used during training (II-E-T-MuCh), then the background is only removed in case real and synthetic images were reconstructed to synthetic images only. The availability of synthetic data with changing backgrounds can then be exploited best by the II-PIRL.

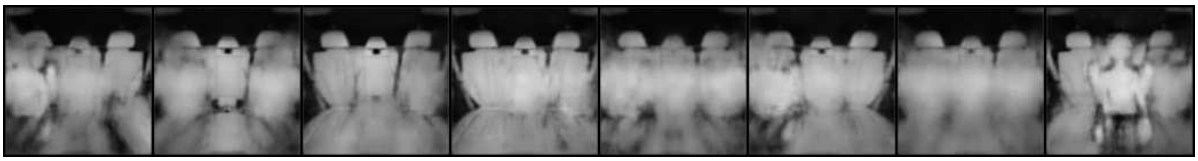
ResNet-50 CNN classifier

Lastly, we evaluated the performance of vanilla ResNet-50 CNN classification models in case real and synthetic images can be shuffled together for training. We either trained the whole network from scratch or we fine-tuned the last block when pre-trained on ImageNet. Fine-tuning all layers did not improve the performance, as reported already several times in

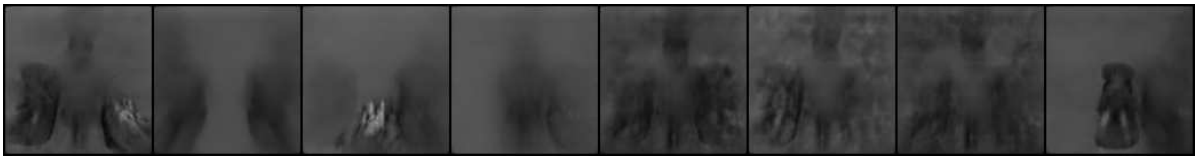
(a) Input images from the Sharan ORSS vehicle



(b) Reconstruction to real and synthetic respectively



(c) Reconstruction to real only



(d) Reconstruction to synthetic only

Fig. 8.7 Reconstruction of real input images from the Sharan vehicle from the ORSS dataset for the extractor multi-channel autoencoder when the II-PIRL and triplet loss is used during training (II-E-T-MuCh). We compare the reconstruction performance when different target distributions were used during training. The background is only removed if real and synthetic images were reconstructed to synthetic images only. The availability of synthetic data with changing backgrounds can then be exploited best in case the II-PIRL is used.

Table 8.5 Ablation study for different training dataset combinations and training approaches. We fine-tuned the last block or trained all layers from scratch of ResNet-50 models. The mean accuracy and standard deviation on the previously unseen Sharan vehicle from ORSS over 10 runs is reported. We also report results when the child seat class is excluded (No CS). Combining a real vehicle and SVIRO-NoCar yields the best results.

Variation	SVIRO-NoCar	SVIRO-Uncertainty	ORSS	Sharan	Sharan - No CS
Scratch	✓			15.3 ± 3.3	30.0 ± 6.4
Scratch			✓	15.0 ± 1.6	17.2 ± 3.3
Scratch		✓	✓	13.0 ± 1.5	18.2 ± 1.1
Scratch	✓		✓	35.3 ± 7.2	53.1 ± 7.3
Fine-tune	✓			42.5 ± 2.6	61.0 ± 5.8
Fine-tune			✓	57.2 ± 2.2	74.6 ± 2.1
Fine-tune		✓	✓	64.8 ± 3.9	84.6 ± 6.2
Fine-tune	✓		✓	81.5 ± 2.2	94.3 ± 1.7

this thesis. The training data was the same as for the autoencoder models and the training hyperparameters for the CNNs are the same as well. The results are reported in Table 8.5. It can be observed that the ResNet-50 model performs slightly better than the extractor models from the previous section, except when the child seat class is left out.

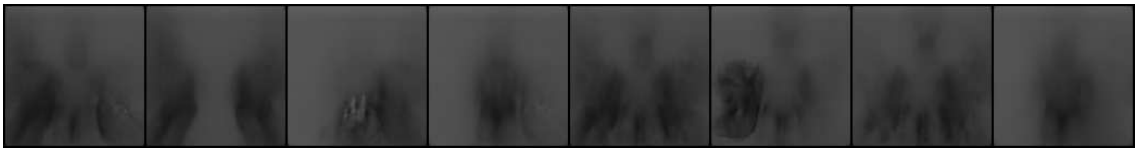
Reconstruction results

In addition to the quantitative improvements, we also want to report interesting insights regarding the reconstruction performance of the previously discussed methods. We report reconstruction results for the real Sharan images from the ORSS dataset in Fig. 8.8, Fig. 8.9, Fig. 8.10 and Fig. 8.11.

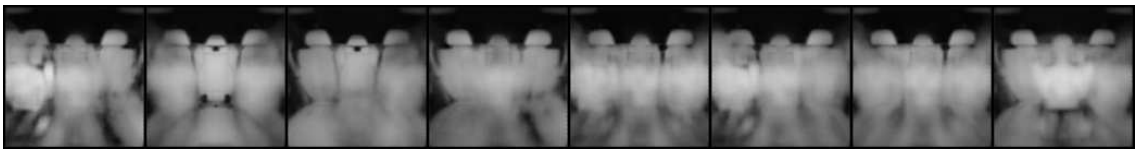
The perceptual loss can be used to achieve better reconstruction results, but the overall conclusion is the same, independently of which reconstruction loss was used. The classification accuracy for the multi-channel autoencoder either using the extractor or not was performing worse, compared to the extractor autoencoder model. However, the reconstruction results by the multi-channel autoencoder approach highlight a different advantage of the latter design choice. All the extractor autoencoder models, except when synthetic images only are used for training, do not remove the vehicle background entirely when images from the Sharan images are used as input.

Although the classification performance is good, the model did not learn to remove the background. The extractor autoencoder model can use two regions in the latent space, one for the synthetic images and one for the real images. Hence, the model does not need to learn to remove undesired features from the real images. In contrast, in the multi-channel autoencoder

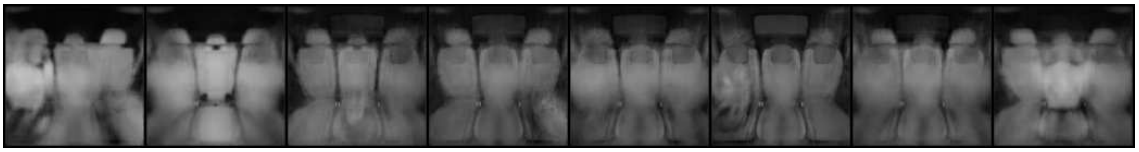
(a) Input images from the Sharan ORSS vehicle



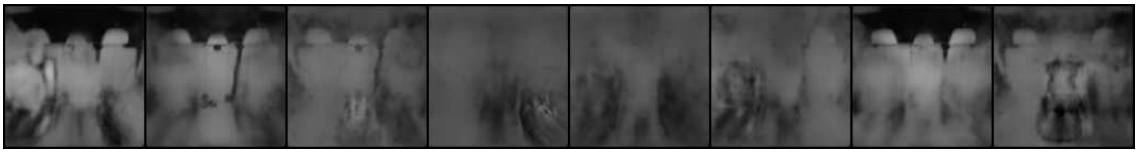
(b) II-E-TAE SVIRO-NoCar



(c) II-E-TAE ORSS



(d) II-E-TAE ORSS and SVIRO-Uncertainty

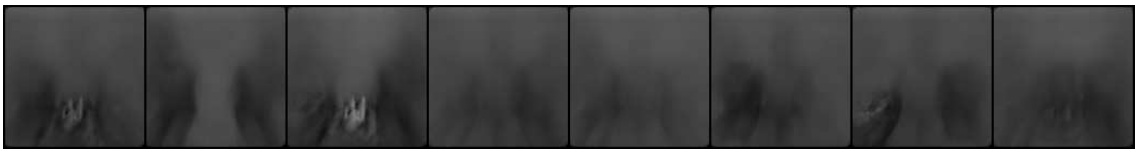


(e) II-E-TAE ORSS and SVIRO-NoCar

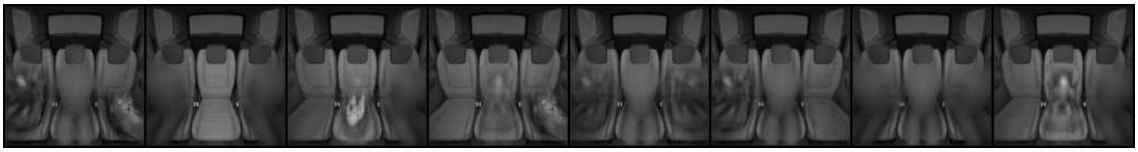
Fig. 8.8 Reconstruction of real input images from the unseen Sharan vehicle from the ORSS dataset for different autoencoder model architectures using the SSIM for the reconstruction error, the II-PIRL, the triplet loss and different training dataset variations. We abbreviate extractor (E) and autoencoder (AE).

(a) Input images from the Sharan ORSS vehicle

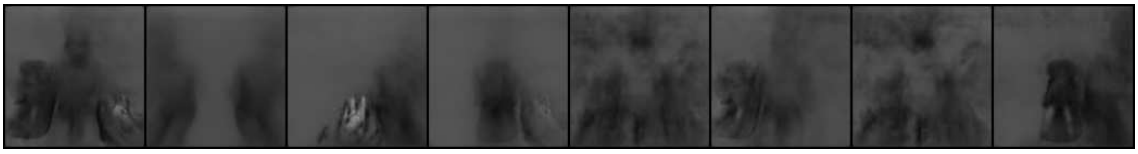
(b) II-T-MuCh ORSS and SVIRO-Uncertainty



(c) II-T-MuCh ORSS and SVIRO-NoCar



(d) II-E-T-MuCh ORSS and SVIRO-Uncertainty



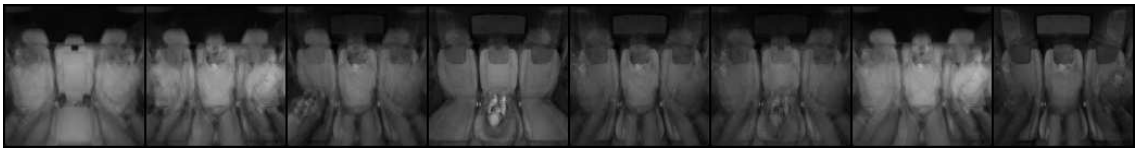
(e) II-E-T-MuCh ORSS and SVIRO-NoCar

Fig. 8.9 Reconstruction of real input images from the unseen Sharan vehicle from the ORSS dataset for different autoencoder model architectures using the SSIM for the reconstruction error, the II-PIRL, the triplet loss and different training dataset variations. We abbreviate extractor (E) and multi-channel (MuCh) autoencoder.

(a) Input images from the Sharan ORSS vehicle

(b) II-E-TAE SVIRO-NoCar

(c) II-E-TAE ORSS

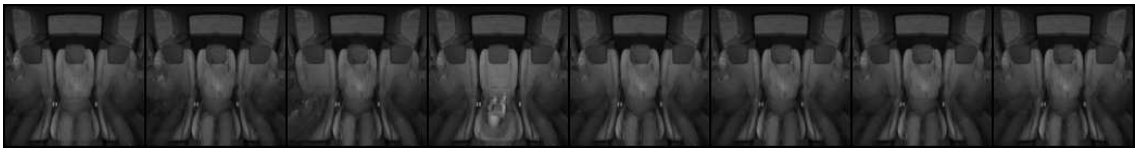


(d) II-E-TAE ORSS and SVIRO-Uncertainty

(e) II-E-TAE ORSS and SVIRO-NoCar

Fig. 8.10 Reconstruction of real input images from the unseen Sharan vehicle from the ORSS dataset for different autoencoder model architectures using the perceptual loss for the reconstruction error, the II-PIRL (II), the triplet loss (T) and different training dataset variations. We abbreviate extractor (E) and autoencoder (AE).

(a) Input images from the Sharan ORSS vehicle



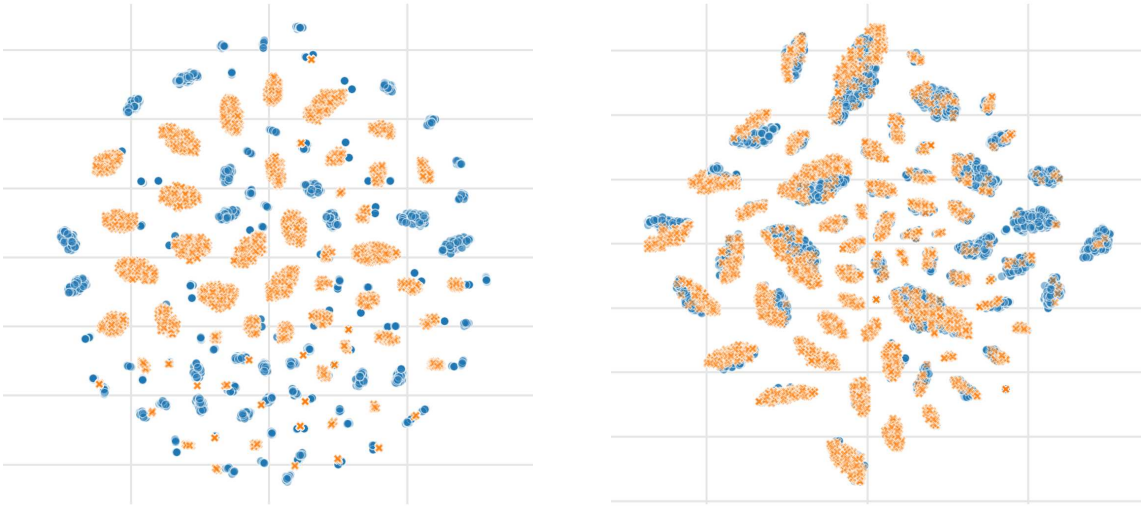
(b) II-T-MuCh ORSS and SVIRO-Uncertainty

(c) II-T-MuCh ORSS and SVIRO-NoCar

(d) II-E-T-MuCh ORSS and SVIRO-Uncertainty

(e) II-E-T-MuCh ORSS and SVIRO-NoCar

Fig. 8.11 Reconstruction of real input images from the unseen Sharan vehicle from the ORSS dataset for different autoencoder model architectures using the perceptual loss for the reconstruction error, the II-PIRL (II), the triplet loss (T) and different training dataset variations. We abbreviate extractor (E) and multi-channel (MuCh) autoencoder.



(a) II-E-TAE (b) II-E-T-MuCh

Fig. 8.12 Comparison of the t-SNE latent space projection between II-E-TAE and II-E-T-MuCh. The synthetic training images (blue) and real training images (orange) are clustered together in case of the II-E-T-MuCh, but both distributions are much more separated for the II-E-TAE. Properties like background removal are hence more likely to be adopted for real images in case they are learned on synthetic images when the multi-channel autoencoder is used.

approach, we forced the model to reconstruct both real and synthetic images using synthetic images as target. Further, we force the model to represent real and synthetic images similarly in the latent space by a dedicated triplet loss. This difference becomes apparent in the latent space projections reported in Fig. 8.12. Since the latter design choice was combined with the II-PIRL, the model needed to learn to remove the background also for real images. Properties like background removal are hence more likely to be adopted for real images in case they are learned on synthetic images when the multi-channel autoencoder is used. Regarding the latter property, the availability of the synthetic data could, however, not be exploited as efficiently in case we constrained the model to reconstruct real images only or real and synthetic images respectively. This was previously discussed and compared in Fig. 8.7. The multi-channel autoencoder approach using the SVIRO-NoCar images removes the background entirely for real images. Although the classification performance is slightly worse, this model design might still be useful for downstream tasks or design choices to be investigated in future work. Regarding the latter, we report an interesting observation in Chapter 10 which allows to remove the background.

8.2.6 The importance of synthetic data generation design choices

A very important point needs to be mentioned regarding the children in the child seats class, i.e. the class with label 2. We report results when the class is left out from the test set in Table 8.4 and Table 8.5. It can be observed that the performance increases significantly in case the class is left out. The reason for this is a miss-alignment between real and synthetic images. Most of the children on the child seats will be classified as adults by the model when synthetic images are included during training. The reason for this is two-fold: on the one hand side, there is a large difference between the synthetic and real class characteristics. In the synthetic images the child seat is dominantly visible and the children are rather small in comparison to the seat. On the real images, the child seat is quite often barely visible and the children appear much larger - sometimes they are close to the size of adults. This is not represented well in the synthetic images, because it is challenging to avoid intersections between children and child seats if the children become to large. This should be addressed in future work. On the other side, the labeling on the real images is not consistent. There are child seat variations, e.g. boosters, which are classified as 3, i.e. as an adult. While this decision was done internally at IEE S.A. , it aggravates the aforementioned problems between synthetic and real miss-alignments. In both cases, we believe that an improved version of the synthetic data would also solve the problem with respect to the children on child seats. Overall, it can be observed that our introduced model and data design choices improve the performances. Since the final performance using the II-PIRL and the extractor are above 90% when children in child seats are neglected, we argue that the success on child seats is only a matter of improving the synthetic data generation process.

8.3 Discussion and limitations

We want to highlight that most of the contribution to the success of our introduced model variations when no real images are included during training stems from the II variation of the PIRL. To this end we trained several types of classifiers in the latent space of different autoencoder model variations and report the results in Table 8.6. The II variation of the PIRL largely improves the classification accuracy compared to the I variation. Moreover, the performance is better compared to the triplet loss variation which uses the label information explicitly as a latent space constraints, compared to the implicit use by the II-PIRL.

The II variation of the PIRL implicitly assumes that the classes are uni-modal, i.e. objects of the same class should be mapped onto a similar point in the latent space. This characteristic can either improve generalization or have a detrimental effect on the performance depending on the task to be solved. It is clear that the reconstruction is far from perfect and hence it

Table 8.6 For each of the 10 experimental runs per method after 250 epochs (i.e. not the best model weights per training were selected) and using the VGG-11 extractor we trained different classifiers in the latent space: k-nearest neighbour (KNN), random forest (RForest) and support vector machine with a linear kernel (SVM). The results show that most of the contribution to the synthetic to real generalization on TICaM is due to the II variation of the PIRL cost function.

	TICaM			SVIRO		
Variant	KNN	RForest	SVM	KNN	RForest	SVM
E-AE	17.1 ± 6.7	24.2 ± 4.1	40.6 ± 8.5	38.7 ± 2.9	58.2 ± 2.0	72.9 ± 2.3
I-E-AE	18.2 ± 7.3	42.4 ± 6.5	50.1 ± 3.7	61.0 ± 3.5	72.2 ± 2.5	73.8 ± 2.3
II-E-AE	73.2 ± 3.9	68.8 ± 5.7	66.9 ± 6.7	83.7 ± 1.9	79.8 ± 2.7	81.4 ± 2.2
E-TAE	69.2 ± 3.4	66.4 ± 4.0	68.7 ± 2.2	76.2 ± 2.3	71.2 ± 2.5	75.3 ± 2.5

might not be suitable for some applications. Under its current form there is no guarantee that, for example, facial landmarks or poses would be preserved. Nevertheless, we believe that extensions of our proposed loss, for example based on constraints (e.g. preservation of poses) could be an interesting direction for future work. It can be observed that our model is not perfect and sometimes struggles: in case an object (e.g. backpack) is located on the seat and for more complex human poses (e.g. people turning over). However, we believe that these problems are related to the training data: a more versatile synthetic dataset would probably improve the model performance on more challenging real images. This is also in line with the remarks from Section 8.3.

Finally, we want to highlight that the vehicle-to-vehicle generalization challenge can potentially be solved by the methods presented in this chapter. Invariances with respect to background and illumination can be learned from synthetic data. It is, however, paramount that the synthetic data generation is performed with care such that it aligns correctly with the real data class distribution and definitions. Afterwards, there is basically no limit regarding the amount of randomness and the number of sceneries which can be generated and achieved. Some amount of real data is required to close the synthetic to real gap though.

8.4 Conclusion

We introduced an autoencoder model which uses a pre-trained classification model as a feature extractor. Our results showed that the our novel model design choices produce superior reconstructions for synthetic to real generalization compared to vanilla autoencoder models, common variations thereof or FactorVAE. However, we highlighted that design choices made

on simple datasets do not necessarily transfer to visually more complex tasks. We performed a step-by-step investigation of additional model changes and showcased the improvements of each change. Our proposed autoencoder model outperforms consistently and more robustly all classification model counterparts. If real data can be incorporated during training, it is possible to tighten the gap between synthetic and real images using the extractor approach. The invariances are then learned from the synthetic data while the transfer to real images is improved by the integration of real images.

Chapter 9

Uncertainty estimation and out-of-distribution detection

The reliability assessment of a machine learning model's prediction is an important feature for the deployment in safety critical applications. Not only can it be used to detect novel sceneries, either as out-of-distribution or anomaly sample, but it also helps to determine deficiencies in the training data distribution. A lot of promising research directions have either proposed traditional methods like Gaussian processes or extended deep learning-based approaches, for example, by interpreting them from a Bayesian point of view. In this chapter we propose two novel approaches for uncertainty estimation based on autoencoder models. The first one uses the second variation of the PIRL: we show that it can also be used for uncertainty estimation where its performance exceeds MC Dropout and an ensemble of models. The second one uses the recursive application of a previously trained autoencoder model. This can be interpreted as a dynamical system storing training examples as attractors. While input images close to known samples will converge to the same or similar attractor, input samples containing unknown features are unstable and converge to different training samples by potentially removing or changing characteristic features. The use of dropout during training and inference leads to a family of similar dynamical systems, each one being robust on samples close to the training distribution, but unstable on samples further away from the training distribution. Either the model reliably removes these features or the resulting instability can be exploited to detect problematic input samples.

In the previous chapters we developed, analyzed and proposed methods to alleviate the challenges of this thesis - at least to some extend. It is, however, utopian to expect a system, and a method, to work for all possible circumstances. To this end, this penultimate chapter investigates and proposes methods to identify those edge cases and endows an autoencoder approach with an uncertainty measure.

9.1 Introduction

Assessing the reliability of machine learning models' predictions is an important challenge for the deployment and applicability of statistical methods, particularly in the case of safety critical applications. This additional information allows the possibility to detect novel and exotic sceneries during the lifetime of a deployed model on which the model's predictions trustability can be determined. This knowledge also gives hints whether the collected training data needs to be extended or modified, e.g. in the case of active learning [74] and continuous learning [117]. Recent activities investigated the possibility for estimating the uncertainty in the case of deep learning-based methods [48, 134, 153, 5]. Monte Carlo (MC) dropout, i.e. using dropout during training and enabling the latter during inference for multiple runs, has been shown to produce good uncertainty quantification [73] on several tasks while limiting the additional overhead during training and inference.

It has been shown that recursive applications of autoencoder models, which are trained under the standard training regime, can be viewed as a dynamical system [204]. From a mathematical and physical point of view [230] the analysis of fixed points and attractors and their basins of attraction are important tools to analyze and understand dynamical systems and their behavior. This iterative process can further be viewed as associative memory [204] to retrieve perturbed training samples, but the models need to be trained long enough to ensure that the training samples become fixed points and attractors. To the best of our knowledge, the recursive application of autoencoder models and their attractors have not been investigated in view of generalization capacities and uncertainty estimation.

As explained in Section 4.8, we extend the recursive application of autoencoder models, thus dynamical systems and attractors, in view of generalization capacities. We combine this strategy with MC Dropout and we exploit characteristics of both design choices to determine whether new input samples are close or far from the training distribution by analyzing the behavior of multiple inferences, see Fig. 9.2 for an example. In the latter, the test sample is converging to a similar attractor, while the out-of-distribution sample converges to different attractors of different classes. We show that uncertainty estimation is improved compared to vanilla MC Dropout and deep ensemble models across three metrics and in view of the entropy distribution. Our ablation study shows that the recursive application is key to the success of our approach. Our analysis is performed on several commonly used OOD dataset combinations as well as on SVIRO-Uncertainty. While introduced independently in this work, the method based on the recursive application of autoencoder models (MCA-AE) and the one using the second variation of the PIRL (MC-II-AE) can also be used together.

9.2 Experiments and results

We evaluated our methods on two scenarios: First, we want to assess the predictive uncertainty where the model should provide a high uncertainty in case it wrongly classifies a test sample. This is made more difficult in the case of the vehicle interior: unseen new objects should be classified as empty seats, i.e. the model should only identify known classes and neglect everything else. For example, if a bag is placed on the rear-bench, the model should classify it as an empty seat, since no airbag needs to be deployed in case of an accident. Our results will show that this is a challenging task. Second, the model should differentiate between in- and out-of-distribution samples. In the case, for example, of training on MNIST and evaluating on Fashion-MNIST, the model cannot perform a correct prediction and it should detect the OOD as such. This is also the case when images from a new vehicle interior are provided as input to the model. If a model can reliably work on a new vehicle interior, there should not be a high uncertainty and vice versa. Again, our results will underline the challenge of this task.

For our investigations, we combined several commonly used datasets for $\mathcal{D}_{in}$ (in-distribution) and $\mathcal{D}_{out}$ (out-of-distribution), as introduced in Section 2.1.4. We used approximately the same number of samples from $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ by sampling each class uniformly. In addition to these commonly used datasets, we use SVIRO-Uncertainty, which enables several analyses as highlighted in Section 3.5. An overview of the number of classes and samples used for each dataset is provided in Table 9.1.

9.2.1 Training and evaluation details

We compare our method against MC Dropout and an ensemble of models using the same architecture as the autoencoder encoder part, but with an additional classification head. We trained our MCA-AE models for 25000 epochs to make sure training samples become attractors, but fewer epochs might produce good results as well. We did not perform an ablation study with respect to the number of epochs needed. Further, we did not check whether the training samples are truly fixed point and attractors because of the computational overhead: This could be done by computing the largest eigenvalue of the Jacobian matrix for each training sample and checking whether it is greater than 1. The autoencoder model was trained as a denoiser, see Section 2.5.1, using blur, random noise, brightness and contrast to augment the images. This facilitates and robustifies the recursive autoencoder application, since the reconstructions will never be perfect. Consequently, to have a fair benchmark, MC Dropout and ensemble models used the same augmented images during training. The latter were trained for 1000 epochs. All methods used Adam, a learning rate of 0.0001 and a batch size of 64. For training on MNIST and Fashion-MNIST we used a latent space of 10, while for all others we used a latent space of

Table 9.1 Overview of the number of classes and samples for OOD or uncertainty estimation for the different datasets used.

Dataset	Classes	$\mathcal{D}_{in}$ and $\mathcal{D}_{out}$	Uncertainty
MNIST	10	2500	10000
Fashion	10	2500	26032
SVHN	10	2500	10000
GTSRB	10	2006	3208
CIFAR10	10	2500	-
Omniglot	660	2636	-
LSUN	10	2500	-
Places365	365	2555	-
SVIRO-U Adults (A)	7	1337	2617
SVIRO-U Seats (S)	8	-	490
SVIRO-U Objects (O)	8	-	1622
SVIRO-U A,S	26	-	896
SVIRO-U A,O	7	-	1421
SVIRO-U A,S,O	30	-	1676
SVIRO Tesla	21	-	2000

64. We used SSIM for computing the reconstruction loss. We used 250 samples per class for training and treat all datasets (even RGB ones) as grayscale images. The latter design choice induces mostly balanced datasets and hence balanced dataset combinations. All images were centre-cropped and resized to 64 by 64 pixels. We used a dropout rate of 0.33 for all methods, because the results are slightly worse for MCA-AE and MC Dropout in case of a 0.1 dropout rate.

For MCA-AE and MC Dropout we used 20 inferences and we used an ensemble of 10 models to assess uncertainty and the OOD estimation. We used 2 recursions for MCA-AE, but this value depends on the dataset used and it might be subject to a hyperparameter search. In our case, the models converged fast for test samples and slow for OOD samples, see for example Fig. 9.2. Hence, iterating more often did not provide a meaningful improvement w.r.t. the metrics mentioned in Section 2.11. We fixed the seeds for all experiments and we repeated each training for 10 runs for MCA-AE and MC Dropout and for 100 runs to get the ensembles of models, such that we can report mean and standard deviation.

9.2.2 Uncertainty estimation and out-of-distribution detection

We report the summary of our results for uncertainty estimation and OOD detection in Tables 9.2, 9.3 and 9.4. An interesting observation is the result that our approach performs significantly

better when the visual complexity is increased (GTSRB, SVIRO), while the performance of MC Dropout and ensemble of models decreases on those setups. On the other side, on visually much simpler datasets (MNIST, Fashion-MNIST, SVHN) the performance of MC Dropout and ensemble of models performs best. There seems to be a correlation between the performance on uncertainty estimation in case of mis-classifications and out-of-distribution detection on unknown images. Another interesting observation is that our approach can much better provide estimations in the unseen Tesla vehicle from SVIRO. It can be observed that the different SVIRO-Uncertainty splits undergo a large performance gap between all methods. This underlines the difficulty of the dataset and the necessity of the different fine grained splits.

We also kept track of all normalized entropies, see Eq. (2.45), used for determining the threshold for all $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$. We computed the histograms of the entropies for each dataset and report results for MCA-AE, MC Dropout and an ensemble of models in Fig. 9.1 when trained on GTSRB. The results show that the entropy distribution between $\mathcal{D}_{in}$ and several $\mathcal{D}_{out}$ are best separated by our approach. The distributions of the different $\mathcal{D}_{out}$ are more similar then for the other models. To quantify this, we computed the sum of the Wasserstein distances between $\mathcal{D}_{in}$ and all $\mathcal{D}_{out}$ (TD, larger is better, because we want them to be different) separately and the sum of the distances between $\mathcal{D}_{out}$ CIFAR10 and all other $\mathcal{D}_{out}$ (OD, smaller is better, because we want them to be similar). We then computed the mean and standard deviation across 10 runs. The results in Table 9.5 for MCA-AE, MC Dropout and an ensemble of models show that our method separates best uncertainty between $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$, and all $\mathcal{D}_{out}$ are most similar between each other. This investigations shows that the different $\mathcal{D}_{out}$ datasets and their uncertainty are treated similarly by our approach, because the distances between the distributions are small. Further, the separation between $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ is also performed best by our approach, since the distances between the $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ distributions are the largest.

9.2.3 Ablation study

While the results of the previous section have shown that the MCA-AE yields good uncertainty and OOD estimations, we want to highlight that the performance is improved due to the recursive application of the previously trained autoencoder model. To this end we provide some additional results where we compare the performance if no recursion is applied. We repeat the evaluation from the previous section and report the performance in Table 9.6. By comparing the results against Tables 9.2, 9.3 and 9.4, it becomes apparent that the recursive application significantly improves uncertainty and OOD estimation.

We also provide examples for several recursive steps and their reconstructions. In Fig 9.2 we report the reconstructions after 1, 2, 3 and 4 iterative steps together with the input image.

Table 9.2 AUROC performance comparison (in percentage, larger is better). We repeated the experiments 10 times and report the mean and standard deviation. If $\mathcal{D}_{in} = \mathcal{D}_{out}$, the result on the test set of $\mathcal{D}_{in}$ only is reported.

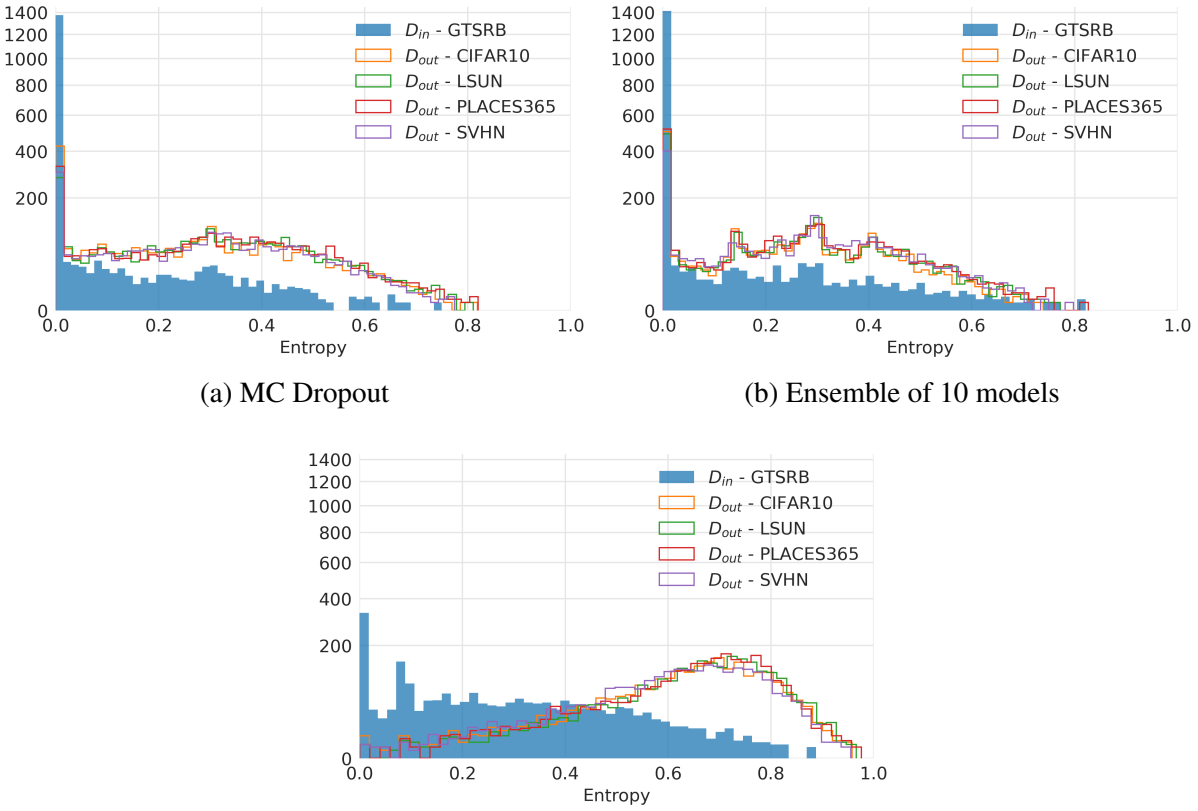
$\mathcal{D}_{in} \rightarrow \mathcal{D}_{out}$	MCA-AE (Ours)	MC Dropout	Ensemble of 10 models
MNIST→MNIST	79.9 ± 1.6	90.1 ± 0.6	85.8 ± 1.4
MNIST→CIFAR10	88.2 ± 2.3	91.2 ± 1.3	91.5 ± 1.1
MNIST→Fashion	74.5 ± 3.2	90.0 ± 1.6	89.5 ± 1.1
MNIST→Omniglot	64.4 ± 5.0	93.4 ± 2.8	95.5 ± 1.0
MNIST→SVHN	92.2 ± 2.0	94.2 ± 1.7	94.9 ± 0.9
Fashion→Fashion	81.0 ± 1.0	82.5 ± 0.4	81.7 ± 0.7
Fashion→CIFAR10	93.9 ± 1.8	88.7 ± 1.9	91.6 ± 0.9
Fashion→MNIST	87.8 ± 4.0	85.4 ± 1.8	90.2 ± 0.5
Fashion→Omniglot	86.8 ± 3.8	93.6 ± 2.0	97.9 ± 0.4
Fashion→SVHN	93.7 ± 2.0	90.8 ± 1.0	94.8 ± 0.5
SVHN→SVHN	77.6 ± 0.8	84.0 ± 0.6	83.7 ± 0.5
SVHN→CIFAR10	77.5 ± 1.2	74.9 ± 0.9	77.6 ± 0.7
SVHN→GTSRB	75.4 ± 2.2	74.0 ± 1.1	75.3 ± 0.7
SVHN→LSUN	78.4 ± 0.9	77.0 ± 0.7	79.2 ± 0.7
SVHN→Places365	78.5 ± 0.8	77.1 ± 0.6	79.2 ± 0.5
GTSRB→GTSRB	85.1 ± 0.9	89.3 ± 2.4	84.6 ± 1.7
GTSRB→CIFAR10	91.4 ± 0.6	81.2 ± 0.9	76.3 ± 0.5
GTSRB→LSUN	93.0 ± 0.7	83.4 ± 0.8	77.7 ± 0.8
GTSRB→Places365	92.3 ± 0.7	82.8 ± 0.7	77.5 ± 0.6
GTSRB→SVHN	91.3 ± 0.7	85.6 ± 1.5	79.4 ± 0.6
SVIRO-U→CIFAR10	95.4 ± 0.6	74.6 ± 3.5	77.7 ± 1.5
SVIRO-U→GTSRB	95.8 ± 1.0	69.9 ± 2.7	74.7 ± 2.5
SVIRO-U→LSUN	94.8 ± 0.5	67.6 ± 2.0	72.0 ± 1.1
SVIRO-U→Places365	95.4 ± 0.5	73.2 ± 2.6	77.4 ± 1.0
SVIRO-U→SVHN	92.4 ± 1.6	81.0 ± 3.4	81.0 ± 1.3
SVIRO-U→Adults (A)	87.8 ± 1.3	95.2 ± 1.7	91.1 ± 1.9
SVIRO-U→Seats (S)	54.0 ± 7.5	17.5 ± 13.1	28.1 ± 6.8
SVIRO-U→Objects (O)	68.9 ± 3.1	64.7 ± 3.2	57.4 ± 2.3
SVIRO-U→A,S	58.8 ± 2.6	36.3 ± 2.3	39.5 ± 1.7
SVIRO-U→A,O	78.8 ± 1.5	70.1 ± 1.6	71.1 ± 0.7
SVIRO-U→A,S,O	62.2 ± 2.0	42.1 ± 2.7	45.8 ± 1.9
SVIRO-U→Tesla (OOD)	88.6 ± 2.0	52.1 ± 2.9	28.6 ± 28.6

Table 9.3 AUPR performance comparison (in percentage, larger is better). We repeated the experiments 10 times and report the mean and standard deviation. If $\mathcal{D}_{in} = \mathcal{D}_{out}$, the result on the test set of $\mathcal{D}_{in}$ only is reported.

$\mathcal{D}_{in} \rightarrow \mathcal{D}_{out}$	MCA-AE (Ours)	MC Dropout	Ensemble of 10 models
MNIST→MNIST	94.9 ± 0.5	99.6 ± 0.1	99.0 ± 0.1
MNIST→CIFAR10	87.4 ± 2.2	92.2 ± 1.1	92.4 ± 0.9
MNIST→Fashion	72.7 ± 3.3	91.1 ± 1.3	90.6 ± 0.9
MNIST→Omniglot	70.4 ± 5.7	94.2 ± 2.5	96.0 ± 0.8
MNIST→SVHN	91.3 ± 1.5	94.9 ± 1.4	95.4 ± 0.7
Fashion→Fashion	94.4 ± 0.3	96.4 ± 0.1	96.4 ± 0.1
Fashion→CIFAR10	95.8 ± 1.1	89.6 ± 1.8	92.1 ± 0.8
Fashion→MNIST	88.2 ± 3.4	86.7 ± 1.5	90.6 ± 0.5
Fashion→Omniglot	91.2 ± 2.5	94.1 ± 1.8	98.1 ± 0.3
Fashion→SVHN	95.6 ± 1.3	91.7 ± 0.9	95.1 ± 0.4
SVHN→SVHN	80.8 ± 1.0	93.1 ± 0.4	92.9 ± 0.3
SVHN→CIFAR10	80.4 ± 1.1	78.0 ± 0.8	80.5 ± 0.6
SVHN→GTSRB	80.5 ± 1.9	80.1 ± 1.0	81.2 ± 0.7
SVHN→LSUN	81.5 ± 0.8	79.8 ± 0.7	81.9 ± 0.7
SVHN→Places365	81.0 ± 0.7	79.4 ± 0.6	81.5 ± 0.4
GTSRB→GTSRB	95.6 ± 0.5	98.8 ± 0.3	97.4 ± 0.3
GTSRB→CIFAR10	90.3 ± 0.8	81.4 ± 0.9	77.7 ± 0.5
GTSRB→LSUN	92.2 ± 0.7	83.3 ± 0.7	78.7 ± 0.6
GTSRB→Places365	91.3 ± 0.7	82.4 ± 0.6	78.2 ± 0.6
GTSRB→SVHN	90.7 ± 0.8	85.5 ± 1.5	80.3 ± 0.5
SVIRO-U→CIFAR10	93.3 ± 1.0	73.6 ± 2.0	75.0 ± 1.1
SVIRO-U→GTSRB	94.9 ± 1.1	74.2 ± 1.2	76.2 ± 1.5
SVIRO-U→LSUN	92.7 ± 0.7	70.1 ± 1.0	71.6 ± 0.6
SVIRO-U→Places365	93.3 ± 0.7	72.5 ± 1.3	74.5 ± 0.7
SVIRO-U→SVHN	88.6 ± 2.3	77.8 ± 2.4	77.3 ± 1.1
SVIRO-U→Adults (A)	99.1 ± 0.3	99.9 ± 0.1	99.8 ± 0.1
SVIRO-U→Seats (S)	8.9 ± 4.2	0.4 ± 0.2	2.6 ± 1.6
SVIRO-U→Objects (O)	83.7 ± 2.4	85.3 ± 3.3	80.6 ± 1.4
SVIRO-U→A,S	48.6 ± 6.4	16.5 ± 2.8	23.6 ± 1.8
SVIRO-U→A,O	93.0 ± 0.5	93.5 ± 0.9	92.3 ± 0.6
SVIRO-U→A,S,O	56.4 ± 4.7	18.6 ± 2.7	33.0 ± 1.7
SVIRO-U→Tesla (OOD)	97.4 ± 0.5	90.9 ± 0.5	45.8 ± 45.8

Table 9.4 FPR95 % performance comparison (in percentage, smaller is better). We repeated the experiments 10 times and report the mean and standard deviation. If $\mathcal{D}_{in} = \mathcal{D}_{out}$, the result on the test set of $\mathcal{D}_{in}$ only is reported.

$\mathcal{D}_{in} \rightarrow \mathcal{D}_{out}$	MCA-AE (Ours)	MC Dropout	Ensemble of 10 models
MNIST→MNIST	74.2 ± 4.8	28.0 ± 2.4	41.5 ± 3.0
MNIST→CIFAR10	44.1 ± 8.3	39.8 ± 5.1	34.0 ± 4.7
MNIST→Fashion	72.3 ± 4.4	40.5 ± 5.9	37.0 ± 3.2
MNIST→Omniglot	99.4 ± 0.7	35.4 ± 12.2	22.0 ± 6.0
MNIST→SVHN	28.2 ± 12.4	30.5 ± 9.4	22.4 ± 5.1
Fashion→Fashion	80.2 ± 2.4	64.4 ± 4.3	64.7 ± 2.2
Fashion→CIFAR10	47.0 ± 20.0	45.7 ± 5.8	34.3 ± 3.1
Fashion→MNIST	48.7 ± 15.6	53.5 ± 5.1	35.7 ± 2.4
Fashion→Omniglot	87.3 ± 9.7	32.6 ± 10.1	9.3 ± 2.3
Fashion→SVHN	48.9 ± 17.1	40.7 ± 3.6	23.0 ± 2.6
SVHN→SVHN	79.5 ± 2.1	69.3 ± 2.4	68.7 ± 2.0
SVHN→CIFAR10	83.6 ± 3.0	85.8 ± 1.8	83.3 ± 1.4
SVHN→GTSRB	80.7 ± 5.4	84.9 ± 2.9	84.0 ± 3.0
SVHN→LSUN	82.7 ± 4.8	81.9 ± 2.1	80.1 ± 1.9
SVHN→Places365	82.6 ± 3.7	80.9 ± 2.5	79.5 ± 1.9
GTSRB→GTSRB	69.3 ± 3.3	50.9 ± 6.0	62.1 ± 3.2
GTSRB→CIFAR10	42.0 ± 3.3	69.5 ± 3.7	83.4 ± 1.3
GTSRB→LSUN	36.5 ± 4.4	65.3 ± 3.9	81.3 ± 1.6
GTSRB→Places365	38.8 ± 3.4	65.1 ± 3.6	80.6 ± 1.7
GTSRB→SVHN	44.5 ± 3.7	60.7 ± 5.1	80.1 ± 1.7
SVIRO-U→CIFAR10	26.9 ± 3.4	60.4 ± 7.2	57.1 ± 3.2
SVIRO-U→GTSRB	25.1 ± 6.9	68.8 ± 4.7	63.8 ± 1.3
SVIRO-U→LSUN	31.5 ± 2.7	72.3 ± 4.2	64.4 ± 2.3
SVIRO-U→Places365	27.3 ± 2.8	63.5 ± 6.8	57.0 ± 2.5
SVIRO-U→SVHN	40.1 ± 7.6	49.5 ± 8.9	51.6 ± 4.1
SVIRO-U→Adults (A)	62.9 ± 3.9	8.9 ± 3.1	28.8 ± 8.8
SVIRO-U→Seats (S)	88.8 ± 10.8	95.7 ± 5.1	98.0 ± 2.5
SVIRO-U→Objects (O)	84.1 ± 5.5	85.3 ± 4.6	86.5 ± 3.3
SVIRO-U→A,S	93.2 ± 1.1	97.4 ± 1.4	96.9 ± 1.1
SVIRO-U→A,O	76.1 ± 3.4	77.4 ± 2.8	77.8 ± 2.4
SVIRO-U→A,S,O	88.7 ± 3.1	96.4 ± 0.9	95.5 ± 0.9
SVIRO-U→Tesla (OOD)	58.0 ± 6.1	94.1 ± 3.7	44.4 ± 44.4



(a) MC Dropout

(b) Ensemble of 10 models

(c) MCA-AE (Ours)

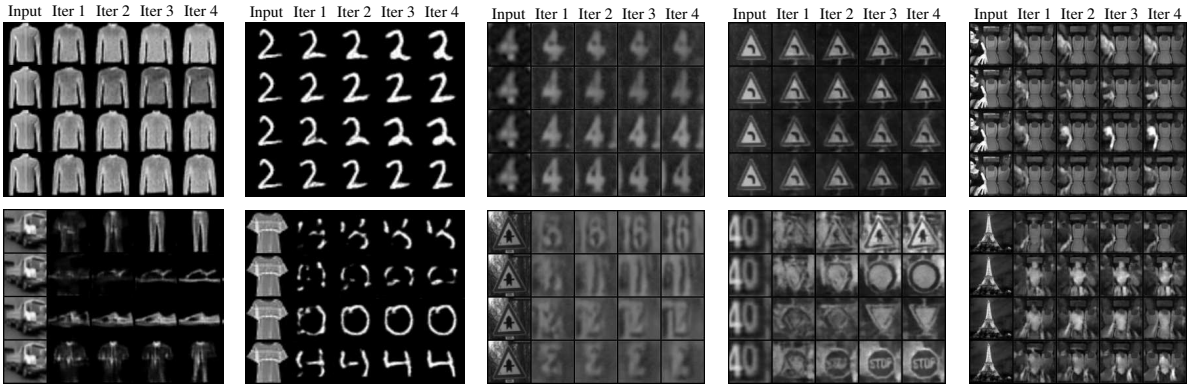
Fig. 9.1 Comparison of entropy histograms between $\mathcal{D}_{in}$ (GTSRB, filled bars with blue) and several $\mathcal{D}_{out}$ (not filled bars and coloured according to dataset used) for different model architectures (a), (b) and (c) . MCA-AE provides the best separation between $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ across the entire datasets. Moreover, the different $\mathcal{D}_{out}$ have a more similar distribution as compared to MC Dropout or an ensemble of models, which we also evaluate quantitatively. Notice the non-linear scale on the y-axis to ease visualization for smaller values.

Table 9.5 We calculated the sum of the Wasserstein distances between $\mathcal{D}_{in}$ and all $\mathcal{D}_{out}$ (TD, larger is better) separately and the sum of the distances between $\mathcal{D}_{out}$ CIFAR10 and all other $\mathcal{D}_{out}$ (OD, smaller is better) for different methods over 10 runs. We report the mean and standard deviation.

		MCA-AE (Ours)	MC Dropout	Ensemble of 10 models
OD	$\downarrow$	0.049 ± 0.007	0.080 ± 0.016	0.050 ± 0.007
TD	$\uparrow$	1.551 ± 0.044	0.854 ± 0.028	0.686 ± 0.017

Table 9.6 OOD and uncertainty estimation when no recursion is applied. In most cases the results are worse compared to 2 recursions - see Tables 9.2, 9.3 and 9.4. In case they are better, we mark the result in grey.

$\mathcal{D}_{in} \rightarrow \mathcal{D}_{out}$	AUROC ↑	AUPR ↑	FPR95 % ↓
MNIST → MNIST	73.5 ± 1.6	91.3 ± 0.8	82.6 ± 2.7
MNIST → CIFAR10	80.4 ± 3.9	77.8 ± 4.5	58.9 ± 8.6
MNIST → Fashion	61.5 ± 5.8	59.3 ± 5.4	82.7 ± 4.2
MNIST → Omniglot	32.6 ± 10.5	44.0 ± 6.5	99.9 ± 0.1
MNIST → SVHN	87.2 ± 3.4	83.3 ± 4.9	38.8 ± 15.5
Fashion → Fashion	77.2 ± 0.9	91.6 ± 0.5	82.0 ± 1.7
Fashion → CIFAR10	88.2 ± 5.0	89.6 ± 6.2	63.0 ± 15.9
Fashion → MNIST	88.2 ± 3.3	89.3 ± 3.0	55.7 ± 8.9
Fashion → Omniglot	60.9 ± 25.4	71.3 ± 19.5	98.5 ± 2.8
Fashion → SVHN	87.2 ± 6.7	88.2 ± 8.8	61.0 ± 11.6
SVHN → SVHN	67.2 ± 1.2	54.6 ± 2.3	87.9 ± 2.1
SVHN → CIFAR10	56.0 ± 1.0	57.2 ± 1.1	94.8 ± 0.9
SVHN → GTSRB	53.6 ± 3.0	60.4 ± 2.7	94.9 ± 1.9
SVHN → LSUN	57.5 ± 1.7	59.2 ± 1.7	94.4 ± 1.4
SVHN → Places365	57.9 ± 1.3	58.6 ± 1.4	93.9 ± 1.5
GTSRB → GTSRB	85.7 ± 1.3	95.9 ± 0.6	67.3 ± 2.9
GTSRB → CIFAR10	82.2 ± 2.3	81.0 ± 2.5	69.9 ± 6.5
GTSRB → LSUN	83.2 ± 2.2	82.0 ± 2.3	68.6 ± 6.1
GTSRB → Places365	82.8 ± 2.1	81.3 ± 2.3	68.2 ± 5.9
GTSRB → SVHN	79.8 ± 2.8	78.7 ± 3.1	76.4 ± 5.5
SVIRO-U → CIFAR10	73.4 ± 2.8	60.9 ± 3.3	76.4 ± 4.7
SVIRO-U → GTSRB	70.5 ± 7.8	63.6 ± 7.2	82.4 ± 6.6
SVIRO-U → LSUN	70.8 ± 2.7	58.2 ± 2.6	81.2 ± 3.6
SVIRO-U → Places365	73.5 ± 2.9	60.4 ± 3.2	76.4 ± 4.5
SVIRO-U → SVHN	79.9 ± 3.5	66.7 ± 5.7	59.8 ± 4.6
SVIRO-U → Adults (A)	86.7 ± 2.2	98.6 ± 0.5	66.6 ± 8.8
SVIRO-U → Seats (S)	19.5 ± 13.1	1.2 ± 1.0	74.6 ± 37.6
SVIRO-U → Objects (O)	58.6 ± 4.9	56.6 ± 6.2	88.2 ± 6.2
SVIRO-U → A,S	43.4 ± 2.9	11.0 ± 2.0	95.1 ± 2.3
SVIRO-U → A,O	65.9 ± 1.8	75.1 ± 1.6	88.5 ± 1.4
SVIRO-U → A,S,O	48.4 ± 3.2	16.9 ± 2.1	91.6 ± 2.4
SVIRO-U → Tesla (OOD)	54.2 ± 10.8	86.2 ± 4.1	94.8 ± 3.7



(a) $\mathcal{D}_{in}$: Fashion (b) $\mathcal{D}_{in}$: MNIST (c) $\mathcal{D}_{in}$: SVHN (d) $\mathcal{D}_{in}$: GTSRB (e) $\mathcal{D}_{in}$: SVIRO-U

Fig. 9.2 Multiple recursive reconstructions (rows - from left to right) of identical samples (first column) from $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ by our novel MCA-AE model. Notice the evolution in the reconstruction results over each iterative step for the OOD samples. $\mathcal{D}_{in}$ converge more robustly compared to $\mathcal{D}_{out}$ reconstructions.

We repeat this for models trained on different $\mathcal{D}_{in}$ and show that $\mathcal{D}_{out}$ reconstructions converge over time (and much slower) to training samples. We hence believe that considering the trajectory of the latent space representation over several steps can be an additional appropriate indicator whether an input sample is in- or out-of-distribution. It becomes also visible that the reconstruction converges robustly to similar classes for $\mathcal{D}_{in}$ samples, but to different classes for $\mathcal{D}_{out}$.

9.2.4 II-PIRL

In this section, we compare the MC Dropout approach on different autoencoder model architectures without the recursive application used in the previous section. We compare vanilla autoencoders (MC-AE) against triplet autoencoders (MC-TAE) and autoencoder using the II-PIRL (MC-II-AE). Further, in this section we do not perform a data augmentation since none of the models is trained in a denoising approach, mainly because no recursion is necessary. This means that the results from this section cannot be compared one-to-one with the ones from the previous section, but instead it provides a more fundamental baseline. Both methods are compared under fair conditions in Section 9.2.5 though. Additionally, we use the training set with adults and child seats from SVIRO-Uncertainty, such that we can report results for the second training split as well in this work. Besides that, all training details and hyperparameters are the same as in the previous section.

The AUROC results in Table 9.7 show that the autoencoder model using the II-PIRL always performs better than the vanilla MC-AE or when a triplet loss is used. Notwithstanding this achievement, it can also be observed that the MC-II-AE usually significantly outperforms MC

Dropout and an ensemble of models. While the ensemble of models outperforms our method only thrice, it is worth noting that our approach only uses a single model while the ensemble uses 10 models.

We also compare the reconstructions of the different methods on OOD images when trained on GTSRB in Fig. 9.3. It can be observed that the reconstructions by MC-II-AE are the most clear and diverse in case of OOD input images.

Finally, as in the previous section, we also kept track of all normalized entropies for all $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$. We computed the histograms of the entropies for each dataset and each method and report results in Fig. 9.4 when trained on GTSRB. The results show that the entropy distribution between $\mathcal{D}_{in}$ and several $\mathcal{D}_{out}$ are best separated when the II-PIRL is used. The distributions of the different $\mathcal{D}_{out}$ are more similar in case an ensemble of models is used though. We computed the sum of the Wasserstein distances between $\mathcal{D}_{in}$ and all $\mathcal{D}_{out}$ (TD, larger is better, because we want them to be different) separately and the sum of the distances between $\mathcal{D}_{out}$ CIFAR10 and all other $\mathcal{D}_{out}$ (OD, smaller is better, because we want them to be similar). We then computed the mean and standard deviation across the 10 runs. The results for MC-AE, MC-TAE, MC-II-AE, MC Dropout and an ensemble of 10 models show that our method best separates uncertainty between $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$. This is, however, not the case for the similarity between all $\mathcal{D}_{out}$. In the latter case, an ensemble of models causes the $\mathcal{D}_{out}$ to be most similar between each other.

9.2.5 Attractors vs. II-PIRL

In the previous sections, we presented two novel approaches to assess uncertainty estimation and out-of-distribution detection using autoencoders. As part of an ablation study and to have a valid comparison between both methods, we want in this section to compare autoencoder attractors and autoencoders using the II-PIRL against each other, but also to combine them.

It is important to note that the combination of the II-PIRL with the attractor approach does not make sense from the point of view of a fixed point - at least not for training samples. By definition of the II-PIRL, a fixed point property cannot be expected since the target image is not the same as the input image. Although the fixed point property is violated for training samples, there can still be other fixed points and attractors. These attractors will not be original training samples, but they will rather be versions of training samples for which the unimportant information has been normalized. This does not constitute a problem because we know from other results presented in this thesis, that the resulting properties, when the PIRL is used, are beneficial for the classification task.

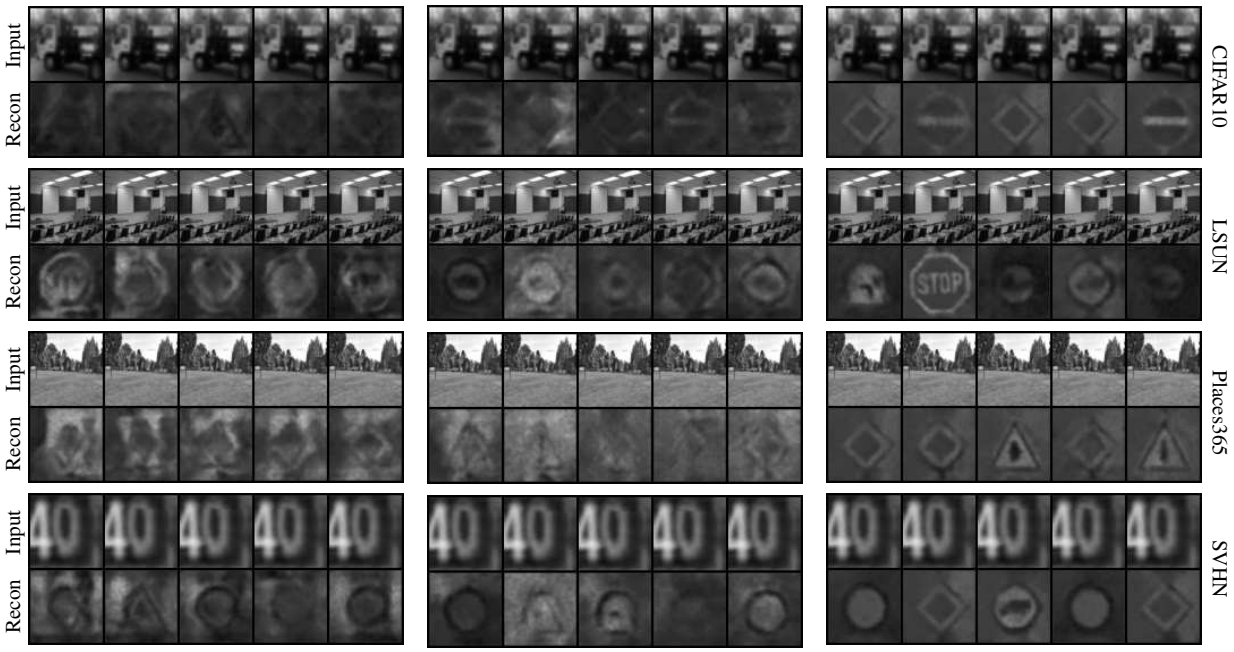
The experiments for our ablation study were conducted on GTSRB. We re-used the same models from the experiments in Section 9.2.2 and Section 9.2.4. The models were trained

Table 9.7 Comparison of AUROC (in percentage, larger is better) of our method against MC Dropout and an ensemble of models as well as vanilla and triplet autoencoders. We repeated the experiments for 10 runs and report the mean and standard deviation. If $\mathcal{D}_{in} = \mathcal{D}_{out}$, then we report the result on the test set of $\mathcal{D}_{in}$ only. Best results are highlighted in grey.

$\mathcal{D}_{in} \rightarrow \mathcal{D}_{out}$	MC-II-AE (Ours)	MC-TAE	MC-AE	MC Dropout	Ensemble
MNIST → MNIST	93.6 ± 0.4	93.3 ± 1.0	84.9 ± 0.8	90.2 ± 0.8	81.0 ± 1.6
MNIST → CIFAR10	99.1 ± 0.8	97.5 ± 1.1	81.0 ± 5.1	91.8 ± 1.8	91.9 ± 1.8
MNIST → Fashion	97.3 ± 0.7	95.0 ± 1.1	77.2 ± 6.1	88.5 ± 2.4	82.6 ± 2.5
MNIST → Omniglot	99.4 ± 0.4	97.6 ± 0.7	82.6 ± 9.0	93.2 ± 4.0	95.8 ± 2.3
MNIST → SVHN	99.1 ± 1.1	98.1 ± 1.0	81.5 ± 7.1	94.9 ± 1.9	94.2 ± 1.6
Fashion → Fashion	86.2 ± 0.5	85.8 ± 0.8	83.5 ± 0.8	82.1 ± 0.4	79.8 ± 0.8
Fashion → CIFAR10	96.6 ± 1.3	91.7 ± 1.9	91.2 ± 3.2	88.6 ± 1.2	91.0 ± 1.0
Fashion → MNIST	91.5 ± 1.7	87.2 ± 2.3	76.4 ± 6.4	83.2 ± 2.0	88.4 ± 0.8
Fashion → Omniglot	97.7 ± 1.2	89.0 ± 3.0	77.7 ± 8.7	91.7 ± 2.4	96.9 ± 0.9
Fashion → SVHN	95.7 ± 2.5	90.5 ± 2.7	92.1 ± 3.4	90.0 ± 1.1	93.6 ± 1.1
GTSRB → GTSRB	94.6 ± 0.9	93.4 ± 0.8	87.9 ± 1.6	85.7 ± 1.2	83.2 ± 0.9
GTSRB → CIFAR10	92.6 ± 3.3	80.7 ± 1.8	75.4 ± 2.5	79.0 ± 0.8	69.2 ± 1.0
GTSRB → LSUN	93.9 ± 3.5	81.2 ± 1.9	76.6 ± 2.3	80.3 ± 0.6	68.3 ± 0.8
GTSRB → Places365	93.6 ± 3.6	82.0 ± 1.7	76.2 ± 2.1	79.4 ± 0.5	68.7 ± 0.8
GTSRB → SVHN	92.8 ± 3.0	83.6 ± 2.4	76.1 ± 3.7	82.8 ± 1.2	72.7 ± 0.7
SVIRO-U → CIFAR10	87.2 ± 7.2	71.5 ± 21.8	78.5 ± 4.9	70.8 ± 10.6	83.8 ± 2.2
SVIRO-U → GTSRB	74.5 ± 9.4	68.1 ± 18.4	82.5 ± 4.8	76.6 ± 6.2	87.0 ± 1.9
SVIRO-U → LSUN	84.4 ± 8.2	71.0 ± 21.2	77.5 ± 4.4	74.0 ± 8.5	82.7 ± 1.8
SVIRO-U → Places365	85.7 ± 7.8	71.2 ± 21.4	79.4 ± 3.8	75.3 ± 7.4	83.7 ± 1.5
SVIRO-U → SVHN	92.7 ± 4.9	72.4 ± 22.6	79.0 ± 4.8	66.2 ± 13.0	84.4 ± 3.0
SVIRO-U → Adults (A)	97.6 ± 0.8	48.7 ± 48.7	88.4 ± 1.1	91.9 ± 0.8	87.3 ± 0.7
SVIRO-U → Seats (S)	93.3 ± 2.8	46.4 ± 46.4	74.9 ± 2.1	89.7 ± 4.2	89.0 ± 2.8
SVIRO-U → Objects (O)	75.5 ± 4.2	39.5 ± 39.6	73.4 ± 1.6	73.7 ± 2.5	73.4 ± 4.5
SVIRO-U → A,S	92.4 ± 2.0	45.7 ± 45.7	75.0 ± 1.7	87.8 ± 1.7	79.0 ± 1.7
SVIRO-U → A,O	81.8 ± 2.3	66.2 ± 16.2	76.4 ± 1.0	80.5 ± 1.9	81.8 ± 0.8
SVIRO-U → A,S,O	77.1 ± 2.6	37.9 ± 37.9	69.6 ± 1.2	75.3 ± 1.9	77.8 ± 1.3
SVIRO-U → Tesla (OOD)	79.0 ± 11.8	65.3 ± 16.5	72.7 ± 4.8	81.3 ± 3.8	80.3 ± 2.2

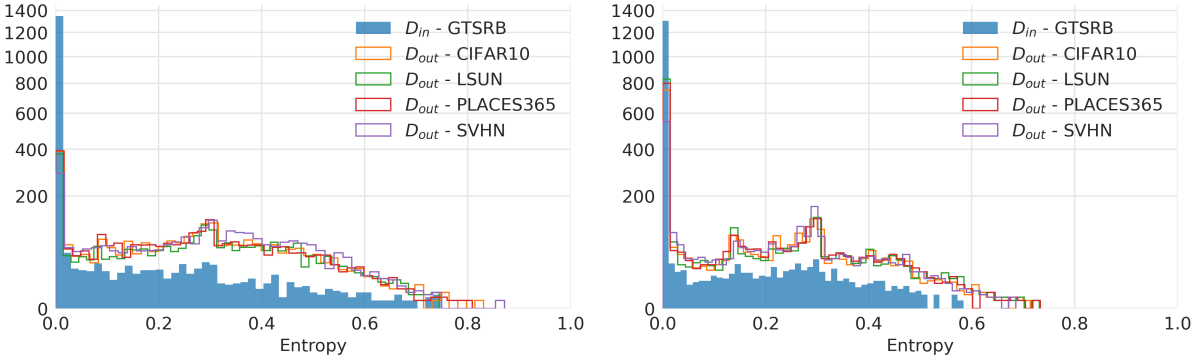
Table 9.8 We calculated the sum of the Wasserstein distances between $\mathcal{D}_{in}$ and all $\mathcal{D}_{out}$ (TD, larger is better) separately and the sum of the distances between $\mathcal{D}_{out}$ CIFAR10 and all other $\mathcal{D}_{out}$ (OD, smaller is better) over 10 runs. We report the mean and standard deviation.

		MC-II-AE (Ours)	MC-TAE	MC-AE	MC Dropout	Ensemble
OD	↓	0.082 ± 0.026	0.086 ± 0.025	0.041 ± 0.014	0.047 ± 0.014	0.030 ± 0.004
TD	↑	1.916 ± 0.332	1.261 ± 0.124	0.782 ± 0.112	0.749 ± 0.014	0.436 ± 0.020



(a) MC-AE (b) MC-TAE (c) MC-II-AE (Ours)

Fig. 9.3 Comparison of OOD images (first rows) with the corresponding reconstructions (second rows) for several inferences (columns) when dropout is enabled. The results are for different autoencoder models for different datasets: CIFAR10 (first block), LSUN (second block), Places365 (third block) and SVHN (fourth block).



(a) MC Dropout

(b) Ensemble

(c) MC-AE

(d) MC-TAE

(e) MC-II-AE (Ours)

Fig. 9.4 Comparison of entropy histograms between $\mathcal{D}_{in}$ (GTSRB, filled bars with blue) and several $\mathcal{D}_{out}$ (not filled bars and coloured according to dataset used) for different methods. MCA-II-AE provides the best separation between $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$. The different $\mathcal{D}_{out}$ have the most similar distribution in case of an ensemble of models, which we also evaluated quantitatively. Notice the non-linear scale on the y-axis to ease visualization for smaller values.

with and without augmenting the training images, with and without the II-PIRL and either for 1000 or 25000 epochs. This enables us to assess the effect of the different hyperparameters. The training for each hyperparameter combination is repeated for 10 runs and the results are reported in Table 9.9 in case an MLP and in Table 9.10 in case a linear SVM was used to perform the classification in the latent space. For MCA model variations we report results when three iteration were used. We want to remind that MCA-II-AE with no iteration is MC-II-AE and MCA-AE with no iteration is MC-AE. Further, we also report results in case the dropout mask is variable for each iterative step, in comparison to a fixed dropout mask for the entire iteration. An ablation study over several number of iterations is reported in Table 9.11.

Several interesting observations can be made. In general, using an MLP is more robust and leads to better performances compared to using a linear SVM classifier, which is expected, because of the higher approximation properties of the MLP. In case the II-PIRL is used during training, augmenting the images and training the model as a denoiser does not improve the performance. However, on the other side, for the MCA-AE approach denoising the images does have a significant positive impact. In all cases, fixing or varying the dropout mask for each iterative step did not lead to a significant difference. Training the models for 25000 epochs instead of 1000 epochs always leads to a better AUROC. Using the II-PIRL improves the results, either combining it with MCA or not. In case the II-PIRL is used, adopting a single iteration is sufficient to obtain the best result. In general, iterating for more than three steps does not contribute to a better uncertainty and OOD assessment.

Lastly, reconstruction results for the model variations investigated in this section are reported in Fig. 9.5 in case of denoising and in Fig. 9.6 when no augmentation was performed during training. It can be observed that using augmented images during training leads to a more versatile reconstruction in case of an OOD sample. If no II-PIRL is used, then the denoising approach leads to visually clearer reconstructions. In general though, using the II-PIRL, training the model for 25000 epochs and iterating several times leads to the best qualitative uncertainty assessment, as supported by the quantitative results. In most cases, the limit images become clear after a few steps only, i.e. three steps. This is also supported by the quantitative ablation study over several iterative steps.

9.3 Discussion and limitations

From a mathematical point of view dynamical systems are defined by natural phenomena or mechanical systems one wants to investigate and understand. Hence, designing or influencing the dynamical system of interest is usually not a possibility. An interesting observation is that the latter phenomenon is not necessarily the case for the recursive application of a trained

Table 9.9 AUROC (in percentage) by single hidden layer MLP classifiers trained in the latent space of different autoencoder models. We used three iterations for the MCA models. The dropout mask for each iteration is either fixed (F) or variable (V). The models were trained for 1000 or 25000 epochs and some were trained as denoising autoencoders. The models were trained on GTSRB and assessed on the latter's test set for uncertainty estimation and on the $\mathcal{D}_{out}$ for OOD detection. Mean and standard deviation over 10 runs are reported.

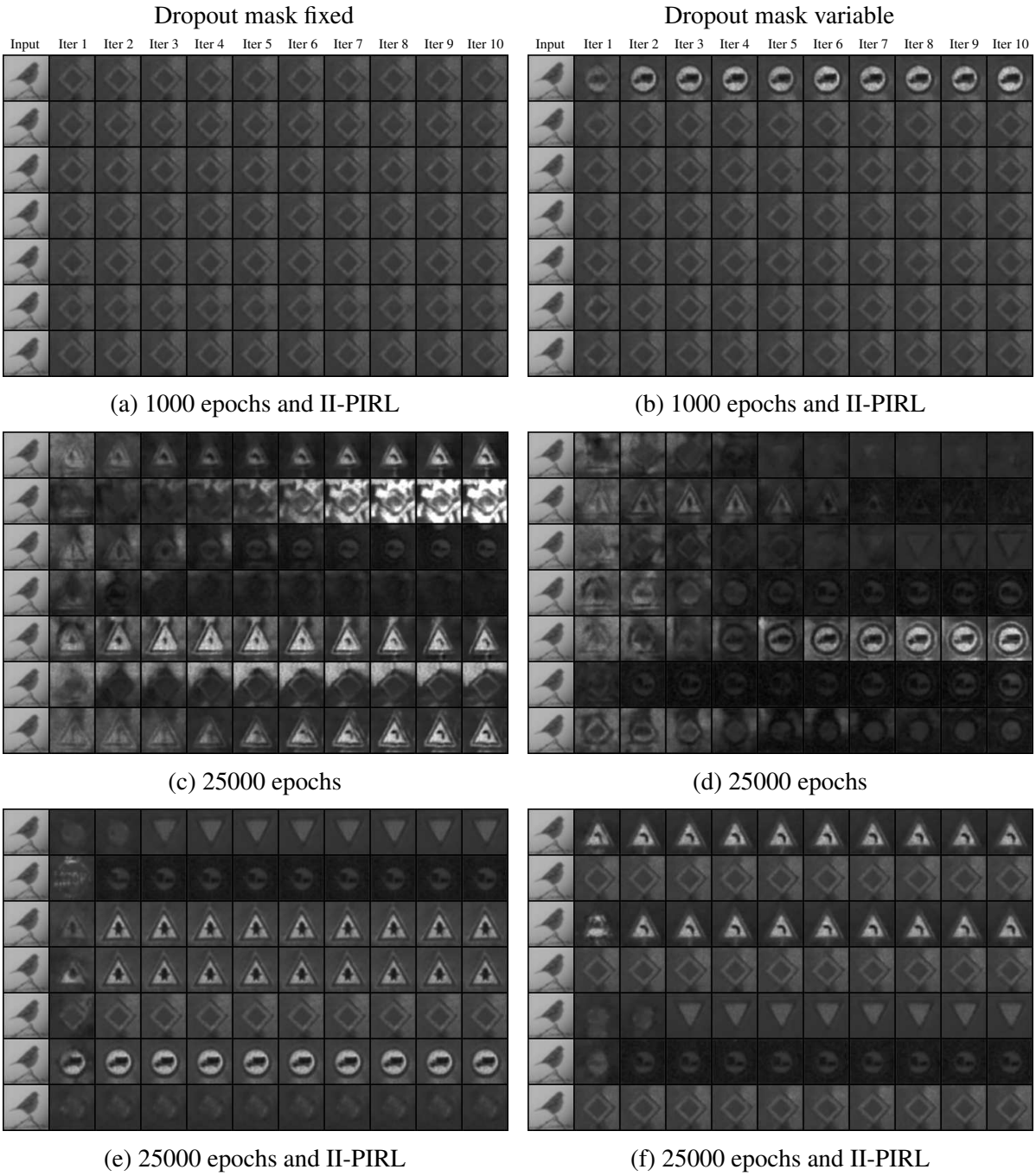
				GTSRB →				
Model	Epochs	Denoising	Mask	GTSRB	CIFAR10	LSUN	Places365	SVHN
MCA-AE	1000		F	83.9 ± 0.7	83.0 ± 1.0	85.7 ± 1.3	84.5 ± 1.3	82.2 ± 1.6
MCA-AE	1000	✓	F	85.7 ± 1.0	85.4 ± 1.4	87.6 ± 1.4	86.8 ± 1.3	84.9 ± 2.1
MCA-AE	25000		F	83.1 ± 0.8	87.9 ± 1.8	91.6 ± 1.3	90.4 ± 1.2	85.3 ± 3.0
MCA-AE	25000	✓	F	85.2 ± 1.6	91.3 ± 0.7	92.8 ± 0.8	92.2 ± 0.9	90.3 ± 1.0
MCA-AE	1000		V	83.4 ± 1.3	82.7 ± 0.9	85.7 ± 1.3	84.5 ± 1.3	81.2 ± 1.5
MCA-AE	1000	✓	V	85.5 ± 0.9	85.7 ± 1.6	87.8 ± 1.4	86.9 ± 1.5	84.9 ± 2.2
MCA-AE	25000		V	83.1 ± 0.8	87.7 ± 1.8	92.0 ± 1.4	90.6 ± 1.4	84.3 ± 3.1
MCA-AE	25000	✓	V	85.1 ± 1.3	91.8 ± 0.8	93.2 ± 0.8	92.6 ± 0.8	90.7 ± 0.9
MC-II-AE	1000		-	94.0 ± 1.5	93.3 ± 2.4	94.8 ± 2.4	94.4 ± 2.5	93.2 ± 2.2
MC-II-AE	1000	✓	-	90.3 ± 7.1	90.6 ± 2.1	92.9 ± 1.9	92.3 ± 1.9	91.9 ± 2.0
MC-II-AE	25000		-	97.6 ± 0.5	95.4 ± 3.0	95.6 ± 3.2	95.4 ± 3.4	96.0 ± 2.5
MC-II-AE	25000	✓	-	96.8 ± 0.5	94.0 ± 1.3	95.2 ± 1.0	94.9 ± 1.0	93.7 ± 1.4
MCA-II-AE	1000		F	92.1 ± 3.2	89.3 ± 1.7	91.7 ± 1.7	91.0 ± 2.0	89.3 ± 1.4
MCA-II-AE	1000	✓	F	91.9 ± 2.7	87.4 ± 3.5	89.9 ± 3.5	89.4 ± 3.5	87.8 ± 4.7
MCA-II-AE	25000		F	96.0 ± 3.3	95.4 ± 3.3	96.6 ± 3.1	96.6 ± 3.2	93.8 ± 4.6
MCA-II-AE	25000	✓	F	96.1 ± 1.5	94.6 ± 1.8	95.7 ± 1.5	95.6 ± 1.8	93.3 ± 2.3
MCA-II-AE	1000		V	91.5 ± 3.8	89.1 ± 1.8	91.5 ± 1.8	90.9 ± 2.1	89.1 ± 1.4
MCA-II-AE	1000	✓	V	92.0 ± 2.8	87.2 ± 3.4	89.7 ± 3.4	89.2 ± 3.4	87.7 ± 4.6
MCA-II-AE	25000		V	96.1 ± 3.6	95.3 ± 3.4	96.6 ± 3.2	96.5 ± 3.3	93.6 ± 4.8
MCA-II-AE	25000	✓	V	96.0 ± 1.2	94.5 ± 1.9	95.7 ± 1.7	95.6 ± 1.8	93.3 ± 2.4

Table 9.10 AUROC (in percentage) by linear classifiers trained in the latent space of different autoencoder models. We used three iterations for the MCA models. The dropout mask for each iteration is either fixed (F) or variable (V). The models were trained for 1000 or 25000 epochs and some were trained as denoising autoencoders. The models were trained on GTSRB and assessed on the latter's test set for uncertainty estimation and on the $\mathcal{D}_{out}$ for OOD detection. Mean and standard deviation over 10 runs are reported.

				GTSRB →				
Model	Epochs	Denoising	Mask	GTSRB	CIFAR10	LSUN	Places365	SVHN
MCA-AE	1000		F	80.5 ± 1.5	77.4 ± 2.8	79.5 ± 2.9	78.8 ± 2.9	75.8 ± 3.7
MCA-AE	1000	✓	F	82.5 ± 1.6	81.6 ± 1.8	83.7 ± 1.9	83.1 ± 2.0	80.7 ± 2.4
MCA-AE	25000		F	80.0 ± 1.5	86.1 ± 2.1	89.2 ± 2.1	88.1 ± 2.0	83.5 ± 3.7
MCA-AE	25000	✓	F	82.5 ± 1.1	88.3 ± 1.4	90.0 ± 1.3	89.3 ± 1.3	86.9 ± 1.4
MCA-AE	1000		V	79.0 ± 1.6	77.4 ± 2.8	79.7 ± 2.7	79.1 ± 3.0	75.8 ± 3.3
MCA-AE	1000	✓	V	81.8 ± 2.0	82.4 ± 1.7	84.5 ± 1.7	83.9 ± 1.8	80.8 ± 2.5
MCA-AE	25000		V	79.7 ± 1.2	85.7 ± 2.1	88.8 ± 2.0	87.7 ± 2.0	82.8 ± 3.7
MCA-AE	25000	✓	V	82.0 ± 1.6	88.3 ± 1.5	90.0 ± 1.5	89.4 ± 1.5	86.7 ± 1.2
MC-II-AE	1000		-	66.3 ± 16.3	58.5 ± 17.7	54.9 ± 21.1	56.9 ± 19.1	56.8 ± 16.2
MC-II-AE	1000	✓	-	88.3 ± 4.6	82.6 ± 5.7	83.9 ± 6.1	83.4 ± 6.2	82.2 ± 5.7
MC-II-AE	25000		-	95.4 ± 4.4	96.5 ± 1.4	97.1 ± 1.0	97.1 ± 1.2	96.4 ± 1.6
MC-II-AE	25000	✓	-	87.1 ± 6.8	83.8 ± 7.2	83.3 ± 9.2	82.9 ± 8.8	83.3 ± 8.4
MCA-II-AE	1000		F	91.2 ± 4.3	88.9 ± 1.7	91.3 ± 1.9	90.9 ± 2.0	88.9 ± 1.6
MCA-II-AE	1000	✓	F	92.2 ± 2.6	86.8 ± 4.8	89.4 ± 4.8	88.8 ± 4.9	87.3 ± 6.2
MCA-II-AE	25000		F	97.0 ± 1.9	94.8 ± 4.5	96.0 ± 4.5	95.9 ± 4.6	93.3 ± 5.5
MCA-II-AE	25000	✓	F	96.4 ± 0.9	94.5 ± 1.6	95.6 ± 1.5	95.5 ± 1.6	93.2 ± 2.0
MCA-II-AE	1000		V	90.9 ± 3.9	88.7 ± 1.7	91.4 ± 2.0	90.7 ± 2.0	88.6 ± 1.6
MCA-II-AE	1000	✓	V	91.6 ± 2.8	86.6 ± 4.8	89.1 ± 4.8	88.6 ± 4.8	87.1 ± 6.1
MCA-II-AE	25000		V	96.5 ± 1.8	94.6 ± 4.7	95.8 ± 4.7	95.7 ± 4.7	93.2 ± 5.8
MCA-II-AE	25000	✓	V	96.2 ± 1.2	94.5 ± 1.6	95.7 ± 1.5	95.5 ± 1.5	93.2 ± 2.0

Table 9.11 Comparison of AUROC (in percentage) for different numbers of iterations for MCA-AE and MCA-II-AE. The results are in case the MLP classifier is used in the latent space. The models are the same as in Table 9.9. The AUROCs over all the different $\mathcal{D}_{out}$ (GTSRB, CIFAR10, LSUN, Places365, SVHN) are averaged to get a unique number.

				Number of iterations				
Model	Epochs	Denoising	Mask	2	3	4	5	6
MCA-AE	1000		F	81.0 ± 1.7	83.8 ± 1.3	83.9 ± 1.8	83.7 ± 2.0	83.2 ± 1.6
MCA-AE	1000	✓	F	84.6 ± 1.4	86.2 ± 1.5	86.6 ± 1.2	85.8 ± 1.3	85.6 ± 1.5
MCA-AE	25000		F	88.1 ± 2.1	88.8 ± 1.8	89.1 ± 2.2	88.9 ± 1.9	88.6 ± 1.8
MCA-AE	25000	✓	F	90.7 ± 1.1	91.7 ± 0.9	91.5 ± 1.0	91.4 ± 0.8	91.1 ± 0.9
MCA-AE	1000		V	80.9 ± 1.7	83.5 ± 1.2	83.9 ± 1.9	83.9 ± 2.0	84.3 ± 2.0
MCA-AE	1000	✓	V	84.7 ± 1.5	86.3 ± 1.7	87.3 ± 1.4	86.9 ± 1.4	87.1 ± 1.7
MCA-AE	25000		V	88.1 ± 2.1	88.7 ± 1.9	88.3 ± 2.5	87.4 ± 2.3	86.4 ± 2.5
MCA-AE	25000	✓	V	90.9 ± 1.1	92.1 ± 0.8	92.0 ± 0.9	91.9 ± 0.8	91.7 ± 1.0
MCA-II-AE	1000		F	91.4 ± 2.0	90.3 ± 1.7	90.1 ± 1.8	90.1 ± 1.8	90.2 ± 1.8
MCA-II-AE	1000	✓	F	90.9 ± 1.3	88.6 ± 3.8	88.2 ± 5.1	88.1 ± 5.4	88.2 ± 5.3
MCA-II-AE	25000		F	97.4 ± 0.6	95.6 ± 3.5	94.9 ± 4.9	94.9 ± 4.9	94.9 ± 4.8
MCA-II-AE	25000	✓	F	95.6 ± 0.7	94.8 ± 1.9	94.7 ± 2.1	94.8 ± 2.0	94.7 ± 2.1
MCA-II-AE	1000		V	91.2 ± 2.0	90.1 ± 1.8	89.9 ± 1.9	89.9 ± 1.9	90.0 ± 1.9
MCA-II-AE	1000	✓	V	90.8 ± 1.3	88.4 ± 3.7	88.0 ± 5.3	87.9 ± 5.6	88.0 ± 5.4
MCA-II-AE	25000		V	97.4 ± 0.6	95.5 ± 3.7	94.8 ± 5.0	94.7 ± 5.1	94.8 ± 5.0
MCA-II-AE	25000	✓	V	95.6 ± 0.7	94.8 ± 1.9	94.7 ± 2.2	94.7 ± 2.1	94.7 ± 2.2



(a) 1000 epochs and II-PIRL

(b) 1000 epochs and II-PIRL

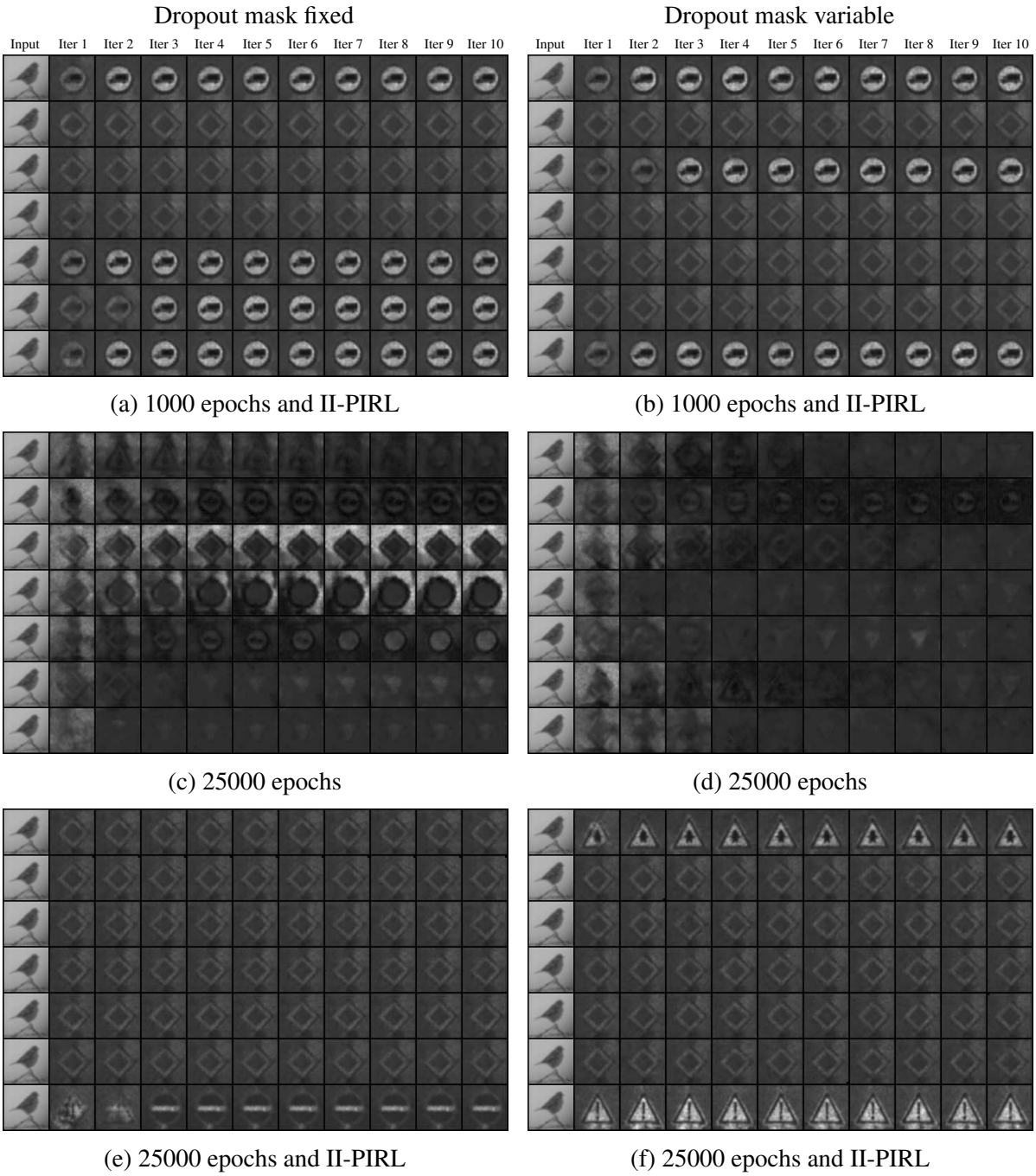
(c) 25000 epochs

(d) 25000 epochs

(e) 25000 epochs and II-PIRL

(f) 25000 epochs and II-PIRL

Fig. 9.5 Ten iterative reconstructions (second to eleventh columns) of a CIFAR10 sample (first columns) when trained on GTSRB. The iterative reconstruction of the same input sample is repeated seven times (rows). Left: the dropout mask is fixed for each entire iteration. Right: a new dropout mask is sampled for each step. The MCA-AE models were either trained using II-PIRL or not, for 1000 or 25000 epochs. The models were trained using augmented images and the denoising method. Using only one iteration is equal to MC-II-AE.



(a) 1000 epochs and II-PIRL

(b) 1000 epochs and II-PIRL

(c) 25000 epochs

(d) 25000 epochs

(e) 25000 epochs and II-PIRL

(f) 25000 epochs and II-PIRL

Fig. 9.6 Ten iterative reconstructions (second to eleventh columns) of a CIFAR10 sample (first columns) when trained on GTSRB. The iterative reconstruction of the same input sample is repeated seven times (rows). Left: the dropout mask is fixed for each entire iteration. Right: a new dropout mask is sampled for each step. The MCA-AE models were either trained using II-PIRL or not, for 1000 or 25000 epochs. The models were trained using the clean input images, i.e. without denoising. Using only one iteration is equal to MC-II-AE.

autoencoder model which is then interpreted as a dynamical system. Since we train the autoencoder in a first step, the resulting dynamical behavior and its attractors can be influenced by our previously defined autoencoder training procedure. We believe that it is an interesting direction for future work to analyze this interrelationship. The effect of the number of epochs needed to obtain good results should be further investigated. The basins of attraction can be studied after the autoencoder model is trained, such that potentially this information could be used to further improve robustness, interpretability and uncertainty estimation and provide guarantees. We believe that the trajectory of the latent space representation over several iterations can give hints about the model robustness. It could be assessed whether an input and/or model can be considered as good, or known, if the convergence is occurring in the first N iterations, where N would need to be fine-tuned as well. It would also be interesting to consider the distance and evolution of the latent space representations to their nearest attractors and to assess how often the nearest attractors change. Finally, the advantages and disadvantages between fixing and sampling a new dropout mask for each iterative step should be investigated in more detail. Especially the mathematical interpretation of sampling a new dropout mask for each step cannot trivially be assessed.

As our results show, the II-PIRL can improve OOD and uncertainty estimation compared to other autoencoder variations. Additionally, our approach can also outperform commonly used uncertainty estimation approaches for deep learning models. II-PIRL can easily be adopted to commonly used datasets and combined with other approaches to further improve the performance. Training the models for a large number of epochs, as we did for MCA, improves the performance up to a point where it outperforms using MCA only.

Training the autoencoder as a denoiser or using the II-PIRL during training violates the fixed point property. However, as we showed, fixed points and attractors do still exist and the iterative reconstruction of an input sample does converge. We did not check this property numerically by computing the eigenvalue of the Jacobian matrix of the network function w.r.t. the input sample, but our experimental results support our assumption. The second variation of the PIRL changes the basins of attractions and potentially reduces the number of attractors of the considered dynamical system. It would hence be interesting to explore the similarities and dissimilarities between the II-PIRL and MCA. Does it make sense to assume that the sample generated by the II-PIRL is actually already a special type of attractor, because the information has been normalized and a single iteration is sufficient to obtain good results? Last but not least, would it make sense to combine the recursive application of autoencoder models with skip connections to see how would this influence the recursive application?

9.4 Conclusion

Our results on several datasets show that the recursive application of autoencoder models, viewed as dynamical systems, together with a MC Dropout approach provides good uncertainty and out-of-distributions estimations. Our model design choices improve the performance, particularly for computer vision datasets of higher visual complexity. Our ablation study highlights that the success is mainly due to the recursion and the entropy histograms underline the improved separability compared to MC Dropout and an ensemble of models. We believe that our introduced method is an interesting research direction for future work and we provide a baseline and several ideas to be investigated.

Notwithstanding this achievement, it can be observed that the II-PIRL provides good uncertainty estimations when combined with MC Dropout as well. If trained sufficiently long, it can even outperform MCA on GTSRB. Although impressive, we believe that future developments of MCA can further improve uncertainty estimation. Particularly due to its analogy to dynamical systems, properties of the recursion in the latent space and combinations with II-PIRL can and should be considered.

Chapter 10

Background removal

In Section 8.2.5 we reported that the multi-channel autoencoder approach combined with the extractor performed only slightly worse with respect to accuracy compared to standard autoencoders utilizing the extractor module. We observed that the multi-channel extractor approach was the only one removing the vehicle interior for real images reliably, but only in case both decoders were constrained to reconstruct synthetic images only. We made the suggestion that one of the potential downstream application could be the removal of the background.

To this end we developed a heuristic to remove the background automatically and we re-used the models trained with the perceptual loss utilized to report the results in Section 8.2.5. It can be observed in Fig. 8.11 that the background is quite uniform due to the use of the II-PIRL. We will exploit this observation in Algorithm 2, which proposes a method to remove the background.

Algorithm 2 Background removal

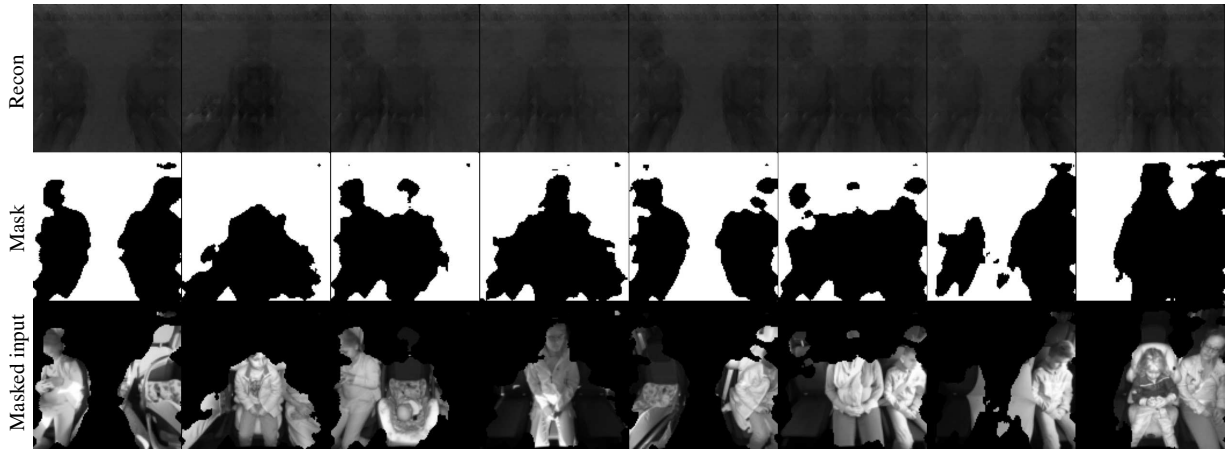
Train autoencoder f with II-PIRL using a dataset allowing to remove the background
for each input sample x **do**
    Perform a reconstruction using the autoencoder model f to get the output $\hat{x}$
    Apply a median blur to $\hat{x}$ using a kernel size of 7
    Use Otsu's method to obtain a binary mask m from $\hat{x}$
    Invert the mask m to get $\bar{m}$
    Find contours in $\bar{m}$
    Draw contours in $\bar{m}$ and fill them to get $\hat{\bar{m}}$
    Invert the mask $\hat{\bar{m}}$ to get $\hat{m}$
    Use the processed mask $\hat{m}$ to mask the input x and/or reconstruction $\hat{x}$
end for

Each input sample is first reconstructed by the previously trained autoencoder model. The resulting reconstruction should hence have a background which is averaged out. This property

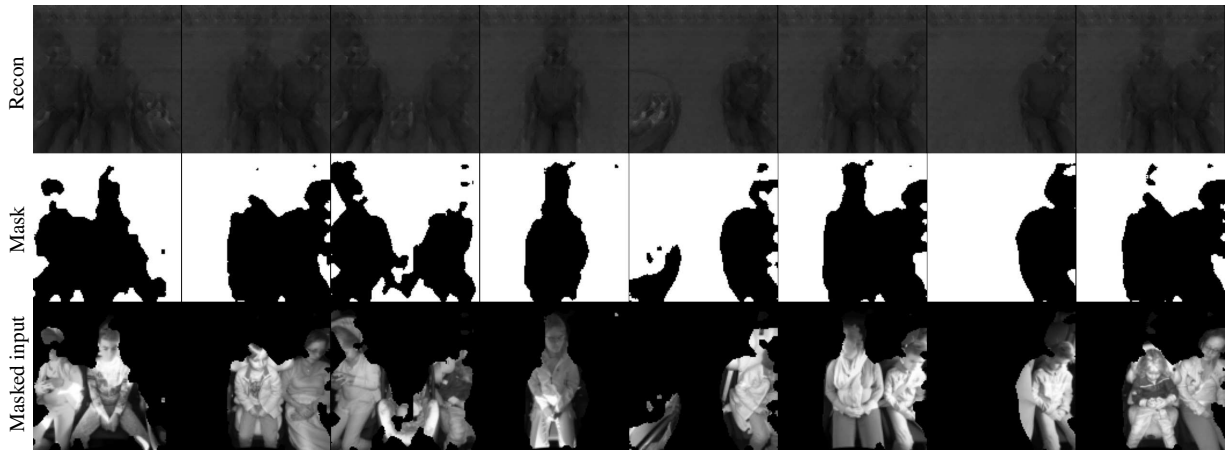
is being exploited by the next steps. To reduce image complexity, we next apply a median blur to the image. This will further smooth out the background, but also the objects on the seat. We will then use an image thresholding method to binarize the image into foreground and background. To this end we adopted Otsu's method [185] - an automatic threshold free approach separating the pixels into two classes. Of course other thresholding methods could be used as well. The resulting mask is then cleaned by filling in potential holes: we find the contours in the image and then fill them with zero. This final mask can then be used to mask the input image and/or the reconstruction to get the objects on the seat or the background. Examples using the multi-channel autoencoder approach are shown in Fig. 10.1 and using the extractor autoencoder in Fig 10.2. Of course the method is not working perfectly, but as can be observed, most of the background is removed such that the foreground objects only are highlighted. Our results also show that the extractor autoencoder performs worse than the multi-channel autoencoders. The proposed approach helps to identify which objects are being reconstructed and what is unimportant, e.g. belonging to the background. It can be observed that some objects are being missed by our approach, even though they appear in the reconstruction. This can probably be improved by more sophisticated thresholding approaches, potentially by means of a neural network dedicated and trained to solve this specific task. Since this approach allows us to retrieve which objects are being reconstructed on novel test images, it could potentially be used to assess the model's reliability and quality for the proposed input sample. For example, we can now assess which regions in the input and output should be compared against.

We adopted this approach to the transfer from synthetic to real sceneries and on a novel vehicle, but this idea could also be applied to a whole range of other tasks and training settings. It is, however, paramount to use, at least for our application, the II-PIRL and the SVIRO-NoCar dataset in order to remove the background reliably. We need the vehicle to be removed, i.e. smoothed out or normalized, such that the presented heuristic can work. The method presented in this section is of course only a first basic approach and we believe that additional improvements are required and possible. Further, we believe that more interesting ideas could be developed by combining the PIRL with some downstream processes.

(a) Input images from the unseen Sharan ORSS vehicle



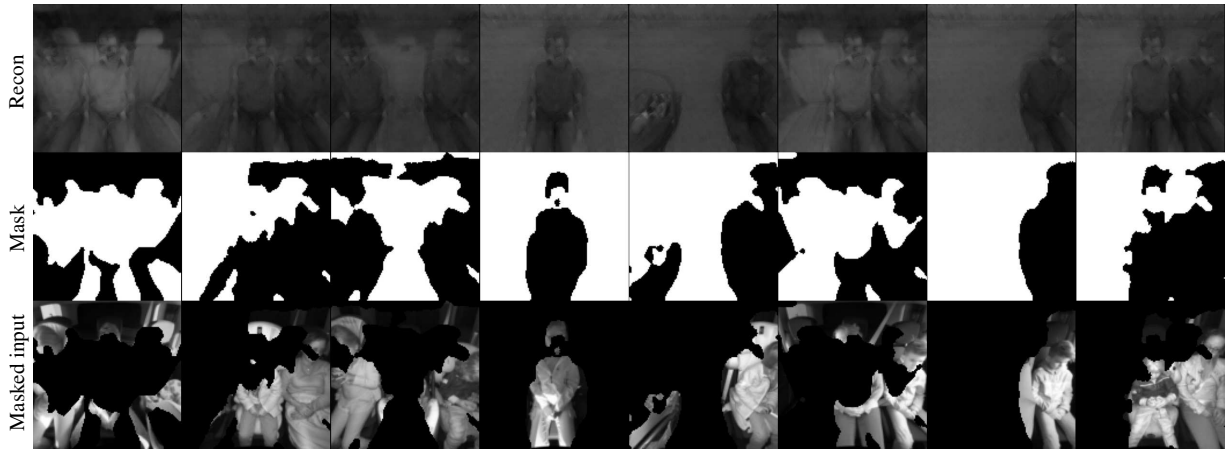
(b) Multi-channel autoencoder trained on X5 ORSS and SVIRO-NoCar



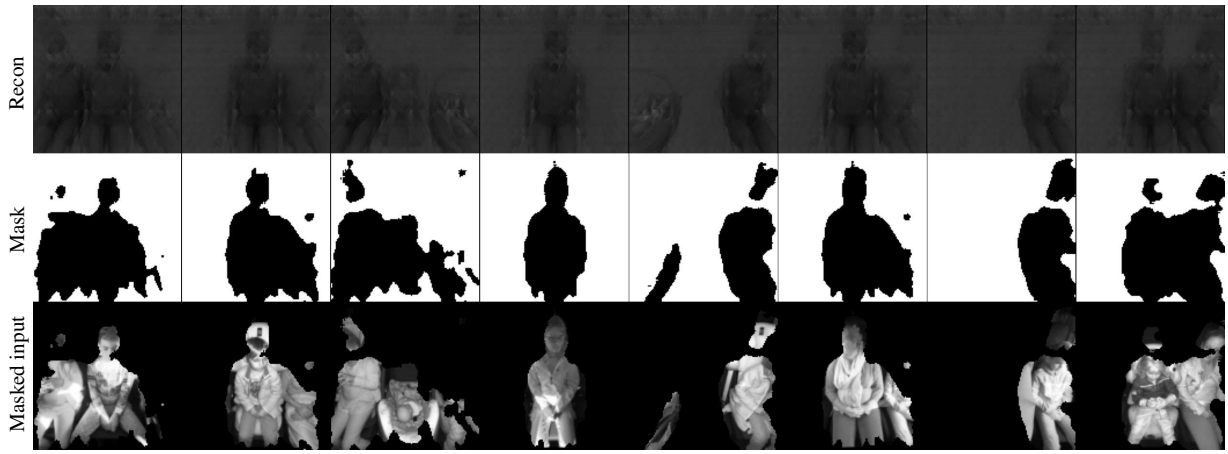
(c) Extractor multi-channel autoencoder trained on X5 ORSS and SVIRO-NoCar

Fig. 10.1 Results for the background removal heuristic presented in Algorithm 2. The reconstructions (first rows) of the input images (a) is used to compute a binary mask (second rows) used to mask the input (third rows). All multi-channel autoencoders used the triplet loss and II-PIRL during training.

(a) Input images from the unseen Sharan ORSS vehicle



(b) Extractor autoencoder trained on X5 Sharan and SVIRO-NoCar



(c) Extractor autoencoder trained on SVIRO-NoCar

Fig. 10.2 Results for the background removal heuristic presented in Algorithm 2. The reconstructions (first rows) of the input images (a) is used to compute a binary mask (second rows) used to mask the input (third rows). All extractor autoencoders used the triplet loss and II-PIRL during training.

Chapter 11

Concluding remarks

In this thesis, we investigated several fundamental questions stemming from industry, but with repercussions on many machine learning applications. We tried to break down the problems into fundamental questions and research directions: invariances with respect to illumination and background, the transfer from synthetic to real sceneries, uncertainty estimation and, in general, the extraction of reliable features for downstream computer vision classification tasks. We created and released several synthetic datasets for the vehicle interior to imitate the automotive application of vehicle interior occupant detection. With our SVIRO dataset and its extensions, we provide a common framework to investigate the aforementioned topics in a single environment. Each extension focuses on a specific task, but since all datasets are situated in the same framework, it was shown that the different problem formulations can be studied either in isolation or combined.

We focused our investigations and model design choices on autoencoder models, because of their elegancy of compressing information by their bottleneck design and simplicity to incorporate different regularization constraints either in the input or the latent space. A lower dimensional representation of the task and input can be useful for safety critical applications, because the overall complexity is reduced. We proposed several improvements to alleviate some of the aforementioned problems and provide evidence that the resulting insights transfer to other datasets commonly used by the research community as well.

The results of an exhaustive ablation study on several computer vision tasks and datasets show that autoencoders using both of our proposed partially impossible reconstructions losses have a beneficial effect on the model performance. We discuss the benefits and downsides of both loss variations and show that they produce good latent space representations on par with the triplet loss. Our novel loss variations also induce interesting reconstruction phenomena: most of the unimportant information is removed or smoothed out. This can potentially be exploited by downstream tasks. Our proposed sampling strategies can easily be combined with

any existing autoencoder model, reconstruction loss and potential latent space regularization. Overall, a partially impossible reconstruction loss seems beneficial to learn more general and reliable features on a wide range of tasks.

The first, weaker variation of the PIRL exploits the availability of certain dataset characteristics: if for each scenery several variations regarding the invariances to be learned are available, e.g. same scenery under different illumination conditions, then the unwanted features can be removed reliably without affecting human poses. The second, stronger variation only uses label information: for each input image, a target image of the same class is sampled randomly. This induces robust feature extraction, allowing for higher classification accuracies. The latter design choice can also be used to improve the transfer from synthetic to real images and to provide more reliable uncertainty estimations. However, human poses cannot be preserved under its current form.

In order to learn more general features, we proposed to use pre-trained convolutional neural networks to extract features from the input images, which are then used as input by the autoencoder model. This improves synthetic to real generalization, particularly if combined with the second variation of the partially impossible reconstruction loss and the triplet loss. Further, in case real images can be used during training, we showed that the synthetic gap can be reduced by incorporating the real images and invariances can be learned by dedicated designed synthetic images. It is hence possible to improve a model's invariances with respect to the car environment and transfer more efficiently to novel real vehicle interiors.

We showed that the recursive application of a previously trained autoencoder model can be viewed as a dynamical system when trained long enough. Combining this model design with Monte Carlo dropout results in a family of similar, but different dynamical systems. Using fixed point and attractor properties, each of the resulting dynamical systems should behave similarly for training and test samples, but differently and not consistently for inputs far away from the training distribution. This improves uncertainty and out-of-distribution detection while maintaining a good generalization performance on test samples. Moreover, we discussed that this design choice opens many directions for future work. Notwithstanding this achievement, we provide evidence that the second variation of the PIRL can outperform the recursive application of autoencoder models either as standalone or when combined with the latter.

Finally, we believe that the results presented in this work show that autoencoders are a promising and versatile deep learning architecture, also for safety critical applications. Autoencoders can be regularized by many different techniques, during training or during inference. We hope that this thesis motivates further research in the direction of autoencoder

models, because we believe that there are many more improvements to be made and benefits to be discovered.

11.1 Future work

During the experiments performed for this thesis, many potential questions were left out. We want to inspire future research by mentioning some of the possible topics to be investigated.

The second variation of the partially impossible reconstruction loss implicitly assumes that the classes are unimodal, i.e. objects of the same class should be mapped onto a similar point in the latent space. This characteristic can either improve generalization or have a detrimental effect on the performance depending on the task to be solved. Under its current form there is no guarantee that, for example, facial landmarks or poses would be preserved. We only ask the model to match the classification labels such that input and target images might contain humans of different poses. This means that the model is hence expected to represent humans of different poses similarly in the latent space. It makes then sense for the model to learn a mean pose approximation such that the reconstruction cannot be expected to preserve the poses anymore. However, we believe that extensions of our proposed loss, for example based on incorporating constraints (preservation of poses and landmarks by sampling images of similar poses) could be an interesting direction for future work. This would also help to reduce the blurriness of the reconstructions, since currently we do not punish bad poses. An analogy of this phenomenon can be observed in the MNIST experiments: since we only ask the classes to be the same, the model learns to approximate each digit by a *mean digit*. However, instead of simply sampling the same class, one could additionally constraint the sampling to use digits of the same style.

Another interesting direction for future research would be the investigation of resulting properties when the partially impossible reconstruction loss is used during training. It would be interesting to see whether the partially impossible reconstruction loss implicitly forces the learned training data manifold in the latent space to follow some beneficial properties, e.g. Euclidean space or disentanglement. This could be combined with and compared against other sampling strategies as to better understand the advantages and disadvantages.

Regarding the recursive application of a trained autoencoder model, it would be of large interest to investigate how the attractors and the basins of attraction are influenced by different training and model hyperparameters. Further, since iterating with dropout means that the latent space will be different at each step, the evolution, or trajectory, of the latent space representation over the different recursive steps can also give hints about the model robustness and uncertainty for a given input sample. Thus far, we have not exploited the latent space information across

different recursive steps and a first simple approach could use a classifier taking the latent space representations across multiple steps as input. It could also be interesting to analyze in more details the behavior of the dynamical system when each recursive iteration samples a new function, i.e. instead of fixing the dropout mask, each recursive step uses a different mask. While this can trivially be conducted experimentally, and result in a similar AUROC performance, the mathematical interpretation is not clear at all. Since the function changes for each iterative step, the recursive application can no longer trivially be considered as a dynamical system. Hence, it is not clear how fixed points, attractors and basins of attractions should be defined anymore, because their properties might no longer hold.

Another possibility would be to exploit the definition of a fixed point. This property should be satisfied for samples close to the training data, but it should be violated for large deviations from the training distribution. One could try to formulate and enforce divergence in case of novel variations and convergence for samples close to the training distribution. We further believe that it should be examined to what extend and under which additional conditions the Banach fixed-point theorem can be applied to our setting. Would it be possible to induce a complete metric space using the triplet loss or regularizations based on the Lipschitz continuity, and what would be the benefit of the latter? Lastly, it could be investigated whether the inspection of the basins of attraction can yield information about the model and the dataset, and hence provide some sort of explainability for the subsequent classification.

References

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

[2] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[3] Kuznetsova Alina, Hassan Rom, Alldrin Neil, Uijlings Jasper, Krasin Ivan, Pont-Tuset Jordi, Kamali Shahab, Stefan Popov, Malloci Matteo, Alexander Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.

[4] Naif Alshammari, Samet Akcay, and Toby P Breckon. On the impact of illumination-invariant image pre-transformation for contemporary automotive semantic scene understanding. In *Intelligent Vehicles Symposium (IV)*, 2018.

[5] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[6] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[7] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.

[8] Luigi Antelmi, Nicholas Ayache, Philippe Robert, and Marco Lorenzi. Sparse multi-channel variational autoencoder for the joint analysis of heterogeneous data. In *International Conference on Machine Learning (PMLR)*, 2019.

[9] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations (ICLR)*, 2018.

[10] Elinor Quittner (Backpack), cjohn259 (Bag), costorella (3Dx bag), andree (Mochila), and Blender3D (Box Of Bottles). Sketchfab. http://www.sketchfab.com, 2022.

[11] Hernán Badino, D Huber, and Takeo Kanade. Visual topometric localization. In *Intelligent Vehicles Symposium (IV)*, 2011.

[12] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 39(12):2481–2495, 2017.

[13] Lucas Baier, Fabian Jöhren, and Stefan Seebacher. Challenges in the deployment and operation of machine learning in practice. In *ECIS*, 2019.

[14] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012.

[15] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *British Machine Vision Conference (BMVC)*, 2016.

[16] Michael Baltaxe, Ruben Mergui, Kobi Nistel, and Gila Kamhi. Marker-less vision-based detection of improper seat belt routing. In *Intelligent Vehicles Symposium (IV)*, 2019.

[17] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems (NeurIPS*, 2019.

[18] Hans-Peter Beise and Steve Dias Da Cruz. Topological properties of basins of attraction and expressiveness of width bounded neural networks. *Analysis and Applications*, under review.

[19] Hans-Peter Beise, Steve Dias Da Cruz, and Udo Schröder. On decision regions of narrow deep neural networks. *Neural Networks*, 140:121–129, 2021.

[20] Mariana Belgiu and Lucian Drăguţ. Random forest in remote sensing: A review of applications and future directions. *ISPRS journal of photogrammetry and remote sensing*, 114:24–31, 2016.

[21] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *arXiv preprint arXiv:1807.02011*, 2018.

[22] Alex Bewley, Jessica Rigley, Yuxuan Liu, Jeffrey Hawke, Richard Shen, Vinh-Dieu Lam, and Alex Kendall. Learning to drive from simulation without real world labels. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[23] Thomas Blaschke, Marcus Olivecrona, Ola Engkvist, Jürgen Bajorath, and Hongming Chen. Application of generative autoencoder in de novo molecular design. *Molecular informatics*, 37(1-2):1700123, 2018.

[24] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning (ICML)*, 2015.

[25] Vanessa Böhm and Uroš Seljak. Probabilistic auto-encoder. *arXiv preprint arXiv:2006.05479*, 2020.

[26] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, 2019.

[27] Markus Braun, Sebastian Krebs, Fabian B. Flohr, and Dariu M. Gavrila. Eurocity persons: A novel benchmark for person detection in traffic scenes. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2019.

[28] Miles Brundage, Shahar Avin, Jasmine Wang, Haydn Belfield, Gretchen Krueger, Gillian Hadfield, Heidy Khlaaf, Jingying Yang, Helen Toner, Ruth Fong, et al. Toward trustworthy ai development: mechanisms for supporting verifiable claims. *arXiv preprint arXiv:2004.07213*, 2020.

[29] Saikiran Bulusu, Bhavya Kailkhura, Bo Li, Pramod K Varshney, and Dawn Song. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347, 2020.

[30] Christopher P. Burgess, Irina Higgins, Arka Pal, Loïc Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in beta-vae. *CoRR*, 2018.

[31] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[32] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. ISSN 2078-2489. doi: 10.3390/info11020125. URL https://www.mdpi.com/2078-2489/11/2/125.

[33] Alexandra Carlson, Katherine A Skinner, Ram Vasudevan, and Matthew Johnson-Roberson. Sensor transfer: Learning optimal sensor effect image augmentation for sim-to-real domain adaptation. *IEEE Robotics and Automation Letters (RA-L)*, 2019.

[34] Wei-Lun Chao, Soravit Changpinyo, Boqing Gong, and Fei Sha. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *European Conference on Computer Vision (ECCV)*, 2016.

[35] Min Chen, Xiaobo Shi, Yin Zhang, Di Wu, and Mohsen Guizani. Deep feature learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*, 7(4):750–758, 2017.

[36] Ricky TQ Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in vaes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[37] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning (ICML)*, 2014.

[38] Wuyang Chen, Zhiding Yu, Zhangyang Wang, and Animashree Anandkumar. Automated synthetic-to-real generalization. In *International Conference on Machine Learning (ICML)*, 2020.

[39] Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[40] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Deep convolutional autoencoder-based lossy image compression. In *IEEE Picture Coding Symposium (PCS)*, 2018.

[41] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL http://www.blender.org.

[42] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[43] Luca Cosmo, Antonio Norelli, Oshri Halimi, Ron Kimmel, and Emanuele Rodolà. LIMP: Learning Latent Shape Representations with Metric Preservation Priors. *European Conference on Computer Vision (ECCV)*, 2020.

[44] Thomas M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, 14:326–334, 1965.

[45] Boguslaw Cyganek and J Paul Siebert. *An introduction to 3D computer vision techniques and algorithms*. John Wiley & Sons, 2011.

[46] Yinglong Dai and Guojun Wang. Analyzing tongue images using a conceptual alignment deep autoencoder. *IEEE Access*, 6:5962–5972, 2018.

[47] Erik B Dam, Martin Koch, and Martin Lillholm. *Quaternions, interpolation and animation*, volume 2. Citeseer, 1998.

[48] Andreas Damianou and Neil D Lawrence. Deep gaussian processes. In *Artificial intelligence and statistics (AISTATS)*, 2013.

[49] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, 2006.

[50] Teófilo Emídio De Campos, Bodla Rakesh Babu, Manik Varma, et al. Character recognition in natural images. *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, 2009.

[51] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[52] Steve Dias Da Cruz, Hans-Peter Beise, Udo Schröder, and Una Karahasanovic. A theoretical investigation of the detection of vital signs in presence of car vibrations and radar-based passenger classification. *Transactions on Vehicular Technology (TVT)*, 2019.

[53] Steve Dias Da Cruz, Oliver Wasenmüller, Hans-Peter Beise, Thomas Stifter, and Didier Stricker. Sviro: Synthetic vehicle interior rear seat occupancy dataset and benchmark. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[54] Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, and Didier Stricker. Illumination normalization by partially impossible encoder-decoder cost function. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[55] Steve Dias Da Cruz, Bertram Taetz, Oliver Wasenmüller, Thomas Stifter, and Didier Stricker. Autoencoder based inter-vehicle generalization for in-cabin occupant classification. In *IEEE Intelligent Vehicles Symposium (IV)*, 2021.

[56] Steve Dias Da Cruz, Bertram Taetz, Oliver Wasenmüller, Thomas Stifter, and Didier Stricker. Autoencoder based inter-vehicle generalization for in-cabin occupant classification. In *IEEE Intelligent Vehicles Symposium (IV)*, 2021.

[57] Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, and Didier Stricker. Autoencoder and partially impossible reconstruction losses. *Sensors*, 22(13), 2022.

[58] Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, and Didier Stricker. Autoencoder for synthetic to real generalization: From simple to more complex scenes. *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, 2022.

[59] Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, and Didier Stricker. Autoencoder attractors for uncertainty estimation. *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, 2022.

[60] Andreas R Diewald, Jochen Landwehr, Dimitri Tatarinov, Patrick Di Mario Cola, Claude Watgen, Catalin Mica, Mathieu Lu-Dac, Peter Larsen, Oscar Gomez, and Thierry Goniva. Rf-based child occupation detection in the vehicle interior. In *International Radar Symposium (IRS)*, 2016.

[61] Tushar Dobhal, Vivswan Shitole, Gabriel Thomas, and Girisha Navada. Human activity recognition using binary motion image and deep learning. *Procedia computer science*, 58:178–185, 2015.

[62] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[63] Samuel Dooley, George Z Wei, Tom Goldstein, and John P Dickerson. Are commercial face detection models as biased as academic models? *arXiv preprint arXiv:2201.10047*, 2022.

[64] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. *Advances in neural information processing systems (NeurIPS)*, 2016.

[65] Martin Engelcke, Adam R Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv:1907.13052*, 2019.

[66] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 2010.

[67] Matteo Fabbri, Fabio Lanzi, Simone Calderara, Andrea Palazzi, Roberto Vezzani, and Rita Cucchiara. Learning to detect and track visible and occluded body joints in a virtual world. In *European Conference on Computer Vision (ECCV)*, 2018.

[68] Kuan Fang, Yunfei Bai, Stefan Hinterstoisser, Silvio Savarese, and Mrinal Kalakrishnan. Multi-task domain adaptation for deep learning of instance grasping from simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[69] Michael E Farmer and Anil K Jain. Occupant classification system for automotive airbag suppression. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[70] Hartmut Feld, Bruno Mirbach, Jigyasa Katrolia, Mohamed Selim, Oliver Wasenmüller, and Didier Stricker. Dfki cabin simulator: A test platform for visual in-cabin monitoring functions. *Commercial Vehicle Technology (CVT) Symposium*, 2021.

[71] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[72] Yarin Gal. Uncertainty in deep learning. 2016.

[73] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on machine learning (ICML)*, 2016.

[74] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning (ICML)*, 2017.

[75] Elena Garces, Carlos Rodriguez-Pardo, Dan Casas, and Jorge Lopez-Moreno. A survey on intrinsic images: Delving deep into lambert and beyond. *International Journal of Computer Vision (IJCV)*, 130(3):836–868, 2022.

[76] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[77] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *International Conference on Learning Representations (ICLR)*, 2018.

[78] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2001.

[79] Arren Glover, Will Maddern, Michael Milford, and Gordon Wyeth. FAB-MAP + RatSLAM: Appearance-based SLAM for Multiple Times of Day. In *Conference on Robotics and Automation (ICRA)*, 2010.

[80] Arren J Glover, William P Maddern, Michael J Milford, and Gordon F Wyeth. Fab-map+ ratslam: Appearance-based slam for multiple times of day. In *International Conference on Robotics and Automation (ICRA)*, 2010.

[81] Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[82] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. In *IEEE 16th international conference on data mining workshops (ICDMW)*, 2016.

[83] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[84] Fang Gongfan. Pytorch ms-ssim. https://github.com/VainF/pytorch-msssim, 2019.

[85] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[86] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[87] P Govardhan and Umesh C. Pati. Nir image based pedestrian detection in night vision with cascade classification and validation. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies (ICACCCT)*, 2014.

[88] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning (ICML)*, 2015.

[89] Aditya Grover and Stefano Ermon. Uncertainty autoencoders: Learning compressed representations via variational information maximization. In *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[90] Amir Hossein Hadjahmadi and Mohammad Mehdi Homayounpour. Robust feature extraction and uncertainty estimation based on attractor dynamics in cyclic deep denoising autoencoders. *Neural Computing and Applications*, 2019.

[91] Iqbal Haris. Plotneuralnet. https://github.com/HarisIqbal88/PlotNeuralNet, 2018.

[92] HDRI Haven. Hdri haven. http://www.hdrihaven.com, 2022.

[93] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[94] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2017.

[95] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations (ICLR)*, 2017.

[96] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations (ICLR)*, 2018.

[97] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[98] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems (NeurIPS)*, 2017.

[99] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*, 2017.

[100] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.

[101] Daniel Ho, Kanishka Rao, Zhuo Xu, Eric Jang, Mohi Khansari, and Yunfei Bai. Retina-gan: An object-aware approach to sim-to-real transfer. *arXiv preprint arXiv:2011.03148*, 2020.

[102] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences (PNAS)*, 1982.

[103] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[104] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks (IJCNN)*, 2013.

[105] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[106] Hum3D. Hum3d. http://www.hum3d.com, 2022.

[107] Ben Hutchinson, Andrew Smart, Alex Hanna, Emily Denton, Christina Greer, Oddur Kjartansson, Parker Barnes, and Margaret Mitchell. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2021.

[108] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[109] Tadanobu Inoue, SLINEMOD ubhajit Choudhury, Giovanni De Magistris, and Sakyasingha Dasgupta. Transfer learning from synthetic to real images using variational autoencoders for precise position detection. In *IEEE International Conference on Image Processing (ICIP)*, 2018.

[110] Fuyuki Ishikawa and Nobukazu Yoshioka. How do engineers perceive difficulties in engineering of machine-learning systems?-questionnaire survey. In *2019 IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry (CESI) and 6th International Workshop on Software Engineering Research and Industrial Practice (SER&IP)*. IEEE, 2019.

[111] Liangxiao Jiang, Zhihua Cai, Dianhong Wang, and Siwei Jiang. Survey of improving k-nearest-neighbor for classification. In *Fourth international conference on fuzzy systems and knowledge discovery (FSKD 2007)*, 2007.

[112] Yibo Jiang and Cengiz Pehlevan. Associative memory in iterated overparameterized sigmoid autoencoders. In *International Conference on Machine Learning (ICML)*, 2020.

[113] Marcel Jirina, MJ Jirina, and K Funatsu. Classifiers based on inverted distances. In *New fundamental technologies in data mining*, volume 1, pages 369–387. InTech, 2011.

[114] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision (ECCV)*. Springer, 2016.

[115] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[116] Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. imgaug. https://github.com/aleju/imgaug, 2020.

[117] Christoph Käding, Erik Rodner, Alexander Freytag, and Joachim Denzler. Fine-tuning deep neural networks in continuous learning scenarios. In *Asian Conference on Computer Vision (ACCV)*, 2016.

[118] Eunhee Kang, Junhong Min, and Jong Chul Ye. A deep convolutional neural network using directional wavelets for low-dose x-ray ct reconstruction. *Medical physics*, 2017.

[119] Katie Kang, Suneel Belkhale, Gregory Kahn, Pieter Abbeel, and Sergey Levine. Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[120] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018.

[121] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019.

[122] Jigyasa Katrolia, Lars Krämer, Jason Rambach, Bruno Mirbach, and Didier Stricker. An adversarial training based framework for depth domain adaptation. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2021.

[123] Jigyasa Singh Katrolia, Bruno Mirbach, Ahmed El-Sherif, Hartmut Feld, Jason Rambach, and Didier Stricker. Ticam: A time-of-flight in-car cabin monitoring dataset, 2021.

[124] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems (NeurIPS)*, 2017.

[125] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems (NeurIPS*, 2020.

[126] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning (ICML)*, 2018.

[127] Seong Soo Kim and AL Narasimha Reddy. Image-based anomaly detection technique: algorithm, implementation and effectiveness. *IEEE Journal on Selected Areas in Communications*, 24(10):1942–1954, 2006.

[128] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[129] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems (NeurIPS*, 2018.

[130] Piotr Koniusz, Yusuf Tas, Hongguang Zhang, Mehrtash Harandi, Fatih Porikli, and Rui Zhang. Museum exhibit identification challenge for domain adaptation and beyond. *arXiv preprint arXiv:1802.01093*, 2018.

[131] Erwin Kreyszig, K Stroud, and G Stephenson. Advanced engineering mathematics. *Integration*, 9:4, 2008.

[132] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[133] Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems (NeurIPS*, 2016.

[134] Matthew A Kupinski, John W Hoppin, Eric Clarkson, and Harrison H Barrett. Ideal-observer computation in medical imaging with use of markov-chain monte carlo techniques. *Journal of the Optical Society of America A: Optics and Image Science, and Vision*, 2003.

[135] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 2015.

[136] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[137] Jean-François Lalonde, Alexei A. Efros, and Srinivasa G. Narasimhan. Webcam clip art: Appearance and illuminant transfer from time-lapse sequences. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 2009.

[138] Mans Larsson, Erik Stenborg, Lars Hammarstrand, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[139] Max-Heinrich Laves, Sontje Ihler, Karl-Philipp Kortmann, and Tobias Ortmaier. Calibration of model uncertainty for dropout variational inference. *arXiv preprint arXiv:2006.11584*, 2020.

[140] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[141] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), 2020.

[142] Andreas Ley, Ronny Hänsch, and Olaf Hellwich. Syb3r: A realistic synthetic benchmark for 3d reconstruction from images. In *European Conference on Computer Vision (ECCV)*, 2016.

[143] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision (ICCV)*, 2017.

[144] Yingwei Li, Qihang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, et al. Shape-texture debiased neural network training. In *International Conference on Learning Representations (ICLR)*, 2020.

[145] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[146] Kun Liang, Yong Ma, Yue Xie, Bo Zhou, and Rui Wang. A new adaptive contrast enhancement algorithm for infrared images based on double plateaus histogram equalization. *Infrared Physics & Technology*, 55(4):309–315, 2012.

[147] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[148] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[149] Si Liu, Risheek Garrepalli, Thomas Dietterich, Alan Fern, and Dan Hendrycks. Open category detection with pac guarantees. In *International Conference on Machine Learning (ICML)*, 2018.

[150] Yunfei Liu, Yu Li, Shaodi You, and Feng Lu. Unsupervised learning for intrinsic image decomposition from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR*, 2020.

[151] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2015.

[152] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2018.

[153] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

[154] David G Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV*, 1999.

[155] Lucy Ellen Lwakatare, Aiswarya Raj, Ivica Crnkovic, Jan Bosch, and Helena Holmström Olsson. Large-scale machine learning systems in real-world industrial settings: A review of challenges and solutions. *Information and software technology*, 127, 2020.

[156] David JC MacKay. Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks. *Network: computation in neural systems*, 1995.

[157] Will Maddern, Alex Stewart, Colin McManus, Ben Upcroft, Winston Churchill, and Paul Newman. Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles. In *International Conference on Robotics and Automation (ICRA)*, 2014.

[158] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 2017.

[159] Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.

[160] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, 2010.

[161] Manuel Martin, Alina Roitberg, Monica Haurilet, Matthias Horne, Simon Reiß, Michael Voit, and Rainer Stiefelhagen. Drive&act: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles. In *International Conference on Computer Vision (ICCV)*, 2019.

[162] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks (ICANN)*, 2011.

[163] Emile Mathieu, Tom Rainforth, Nana Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In *International Conference on Machine Learning (ICML)*. PMLR, 2019.

[164] Mackenzie Weygandt Mathis and Alexander Mathis. Deep learning tools for the measurement of animal behavior in neuroscience. *Current opinion in neurobiology*, 60:1–11, 2020.

[165] Roderick McCall, Fintan McGee, Alexander Meschtscherjakov, Nicolas Louveton, and Thomas Engel. Towards a taxonomy of autonomous vehicle handover situations. In *International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutoUI)*, 2016.

[166] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software (JOSS)*, 3(29), 2018.

[167] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29): 861, 2018.

[168] Paul Melby, Nicholas Weber, and Alfred Hübler. Dynamics of self-adjusting systems with noise. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 15(3):033902, 2005.

[169] Gledson Melotti, Cristiano Premebida, Jordan J Bird, Diego R Faria, and Nuno Gonçalves. Reducing overconfidence predictions for autonomous driving perception. *arXiv preprint arXiv:2202.07825*, 2022.

[170] Gregory P Meyer. An alternative probabilistic interpretation of the huber loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2021.

[171] Ibomoiye Domor Mienye, Yanxia Sun, and Zenghui Wang. Improved sparse autoencoder based artificial neural network approach for prediction of heart disease. *Informatics in Medicine Unlocked*, 18:100307, 2020.

[172] Renqiang Min, David A Stanley, Zineng Yuan, Anthony Bonner, and Zhaolei Zhang. A deep non-linear feature mapping for large-margin knn classification. In *International Conference on Data Mining (ICDM)*, 2009.

[173] Mindfront, punkdunk, MargaretToigo, Sonntag78, and Elvaerwyn. Makehuman. http://www.makehumancommunity.org, 2022.

[174] Boris Moiseev, Artem Konev, Alexander Chigorin, and Anton Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. In *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 2013.

[175] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. Pytorch metric learning, 2020.

[176] Ryota Nakatani, Daichi Kouno, Kazutaka Shimada, and Tsutomu Endo. A person identification method using a top-view head image from an overhead camera. *J. Adv. Comput. Intell. Intell. Informatics*, 16(6):696–703, 2012.

[177] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[178] Roohollah Noori, Hund-Der Yeh, Maryam Abbasi, Fatemeh Torabi Kachoosangi, and Saber Moazami. Uncertainty analysis of support vector machine for online prediction of five-day biochemical oxygen demand. *Journal of Hydrology*, 527:833–843, 2015.

[179] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650*, 2013.

[180] Farzan Erlik Nowruzi, Prince Kapoor, Dhanvin Kolhatkar, Fahed Al Hassanat, Robert Laganiere, and Julien Rebut. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. *arXiv preprint arXiv:1907.07061*, 2019.

[181] Helena E Nusse and James A Yorke. *Dynamics: numerical explorations: accompanying computer program dynamics*, volume 101. Springer, 2012.

[182] Dong Yul Oh and Il Dong Yun. Residual error based anomaly detection using auto-encoder in smd machine sound. *Sensors*, 2018.

[183] Daniel Olid, José M. Fácil, and Javier Civera. Single-view place recognition under seasonal changes. In *IROS Workshop on Planning, Perception, Navigation for Intelligent Vehicle (PPNIV)*, 2018.

[184] Alon Oring, Zohar Yakhini, and Yacov Hel-Or. Autoencoder image interpolation by shaping the latent space. In *International Conference on Machine Learning (ICML)*. PMLR, 2021.

[185] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[186] Sebastian Palacio, Joachim Folz, Jörn Hees, Federico Raue, Damian Borth, and Andreas Dengel. What do deep networks like to see? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[187] Andrei Paleyes, Raoul-Gabriel Urma, and Neil D Lawrence. Challenges in deploying machine learning: a survey of case studies. *arXiv preprint arXiv:2011.09926*, 2020.

[188] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[189] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[190] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.

[191] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[192] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge, 2017.

[193] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *arXiv preprint arXiv:1812.01754*, 2018.

[194] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*, 2018.

[195] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[196] Ana Pereira and Carsten Thomas. Challenges of machine learning applied to safety-critical cyber-physical systems. *Machine Learning and Knowledge Extraction*, 2(4): 579–602, 2020.

[197] Toby Perrett and Majid Mirmehdi. Cost-based feature transfer for vehicle occupant classification. In *Asian Conference on Computer Vision (ACCV)*, 2016.

[198] Helmut Piazena, Hans Meffert, and Ralf Uebelhack. Spectral remittance and transmittance of visible and infrared-a radiation in human skin—comparison between in vivo measurements and model calculations. *Photochemistry and photobiology*, 93(6): 1449–1461, 2017.

[199] Pavlin G. Poličar, Martin Stražar, and Blaž Zupan. opentsne: a modular python library for t-sne dimensionality reduction and embedding. *bioRxiv*, 2019. doi: 10.1101/731877. URL https://www.biorxiv.org/content/early/2019/08/13/731877.

[200] Erwin Jose Lopez Pulgarin, Guido Herrmann, and Ute Leonards. Drivers' manoeuvre classification for safe hri. In *Conference Towards Autonomous Robotic Systems*, 2017.

[201] Liangqiong Qu, Jiandong Tian, Shengfeng He, Yandong Tang, and Rynson WH Lau. Deshadownet: A multi-context embedding deep network for shadow removal. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[202] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[203] A Radhakrishnan, KD Yang, M Belkin, and C Uhler. Memorization in overparameterized autoencoders. In *Deep Phenomena Workshop, International Conference on Machine Learning (ICML)*, 2019.

[204] Adityanarayanan Radhakrishnan, Mikhail Belkin, and Caroline Uhler. Overparameterized neural networks implement associative memory. *Proceedings of the National Academy of Sciences (PNAS)*, 2020.

[205] Ameet Annasaheb Rahane and Anbumani Subramanian. Measures of complexity for large scale image datasets. In *IEEE International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2020.

[206] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, et al. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2020.

[207] Xuming Ran, Mingkun Xu, Lingrui Mei, Qi Xu, and Quanying Liu. Detecting out-of-distribution samples via variational auto-encoder with reliable uncertainty estimation. *Neural Networks*, 2021.

[208] Akshay Rangesh, Bowen Zhang, and Mohan M Trivedi. Driver gaze estimation in the real world: Overcoming the eyeglass challenge. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[209] Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[210] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[211] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016.

[212] Simon Reiß, Alina Roitberg, Monica Haurilet, and Rainer Stiefelhagen. Deep classification-driven domain adaptation for cross-modal driver behavior recognition. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[213] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems (NeurIPS*, 2015.

[214] Nadia Robertini, Florian Bernard, Weipeng Xu, and Christian Theobalt. Illumination-invariant robust multiview 3d human motion capture. In *Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[215] David W. Romero and Mark Hoogendoorn. Co-attentive equivariant neural networks: Focusing equivariance on transformations co-occurring in data. In *International Conference on Learning Representations (ICLR)*, 2020.

[216] Ariel Ruiz-Garcia, Vasile Palade, Mark Elshaw, and Ibrahim Almakky. Deep learning for illumination invariant facial expression recognition. In *International Joint Conference on Neural Networks (IJCNN)*, 2018.

[217] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *International Conference on Learning Representations (ICLR)*, 2016.

[218] Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative memories via predictive coding. *Advances in Neural Information Processing Systems (NeurIPS*, 2021.

[219] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[220] Anke Schwarz, Monica Haurilet, Manuel Martinez, and Rainer Stiefelhagen. Driveahead-a large-scale driver head pose dataset. In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.

[221] Vikash Sehwag, Arjun Nitin Bhagoji, Liwei Song, Chawin Sitawarin, Daniel Cullina, Mung Chiang, and Prateek Mittal. Analyzing the robustness of open-world machine learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2019.

[222] Mohamed Selim, Ahmet Firintepe, Alain Pagani, and Didier Stricker. Autopose: Large-scale automotive driver head pose and gaze dataset with deep head pose baseline. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2020.

[223] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[224] Zhixin Shu, Sunil Hadap, Eli Shechtman, Kalyan Sunkavalli, Sylvain Paris, and Dimitris Samaras. Portrait lighting transfer using a mass transport approach. *ACM Transactions on Graphics (TOG)*, 2017.

[225] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations (ICLR)*, 2015.

[226] Indah Agustien Siradjuddin, Wrida Adi Wardana, and Mochammad Kautsar Sophan. Feature extraction using self-supervised convolutional autoencoder for content based image retrieval. In *IEEE 3rd International Conference on Informatics and Computational Sciences (ICICoS)*, 2019.

[227] Jake Snell, Karl Ridgeway, Renjie Liao, Brett D Roads, Michael C Mozer, and Richard S Zemel. Learning to generate images with perceptual similarity metrics. In *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017.

[228] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems (NeurIPS)*, 2016.

[229] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 2012.

[230] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering*. Westview Press, 2000.

[231] Tiancheng Sun, Jonathan T Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul Debevec, and Ravi Ramamoorthi. Single image portrait relighting. *ACM Transactions on Graphics (TOG)*, 2019.

[232] Shan Suthaharan. Support vector machine. In *Machine learning models and algorithms for big data classification*, pages 207–235. Springer, 2016.

[233] Florian Tambon, Gabriel Laberge, Le An, Amin Nikanjam, Paulina Stevia Nouwou Mindom, Yann Pequignot, Foutse Khomh, Giulio Antoniol, Ettore Merlo, and François Laviolette. How to certify machine learning based safety-critical systems? a systematic literature review. *Automated Software Engineering*, 29, 2022.

[234] Kaiwen Tan, Weixian Huang, Jinlong Hu, and Shoubin Dong. A multi-omics supervised autoencoder for pan-cancer clinical outcome endpoints prediction. *BMC Medical Informatics and Decision Making*, 20(3):1–9, 2020.

[235] Hiroshi Teramoto, Mikito Toda, and Tamiki Komatsuzaki. Understandings of chemical reaction dynamics in terms of dynamical systems theory. In *AIP Conference Proceedings*, volume 1702, page 090042. AIP Publishing LLC, 2015.

[236] Textures.com. Textures.com. http://www.textures.com, 2022.

[237] Maoqing Tian, Shuai Yi, Hongsheng Li, Shihua Li, Xuesen Zhang, Jianping Shi, Junjie Yan, and Xiaogang Wang. Eliminating background-bias for robust person re-identification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[238] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[239] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[240] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2014.

[241] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018.

[242] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[243] Michal Uricár, David Hurych, Pavel Krizek, and Senthil Yogamani. Challenges in designing datasets and validation for autonomous driving. *arXiv preprint arXiv:1901.09270*, 2019.

[244] Hristina Uzunova, Sandra Schultz, Heinz Handels, and Jan Ehrhardt. Unsupervised pathology detection in medical images using conditional variational autoencoders. *International journal of computer assisted radiology and surgery*, 14(3):451–461, 2019.

[245] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[246] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research (JMLR)*, 9(11), 2008.

[247] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

[248] Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. Are disentangled representations helpful for abstract visual reasoning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[249] Ali Moradi Vartouni, Saeed Sedighian Kashi, and Mohammad Teshnehlab. An anomaly detection method to detect web attacks using stacked auto-encoder. In *2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*, 2018.

[250] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[251] Michel Verleysen and Damien François. The curse of dimensionality in data mining and time series prediction. In *International work-conference on artificial neural networks*, pages 758–770. Springer, 2005.

[252] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research (JMLR)*, 11(Dec):3371–3408, 2010.

[253] Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[254] Anthony Yu-Tung Wang, Mahamad Salah Mahmoud, Mathias Czasny, and Aleksander Gurlo. Crabnet for explainable deep learning in materials science: bridging the gap between academia and industry. *Integrating Materials and Manufacturing Innovation*, 2022.

[255] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[256] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[257] Jifeng Wang, Xiang Li, and Jian Yang. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[258] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[259] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[260] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[261] Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40, 2017.

[262] Yan Wu, Mihaela Rosca, and Timothy Lillicrap. Deep compressed sensing. In *International Conference on Machine Learning (ICML)*, 2019.

[263] Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning-the good, the bad and the ugly. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[264] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[265] Zhisheng Xiao, Qing Yan, and Yali Amit. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. *Advances in Neural Information Processing Systems*, 2020.

[266] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems (NeurIPS)*, 2012.

[267] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference (WWW)*, 2018.

[268] Yao Yang, Haoran Chen, and Junming Shao. Triplet enhanced autoencoder: Model-free discriminative network embedding. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

[269] Cao Yu, Li Jinxu, Zhao Fudong, Bian Ran, and Liu Xia. Comparative study on face recognition based on svm of one-against-one and one-against-rest methods. In *2014 8th International Conference on Future Generation Communication and Networking (FGCN)*. IEEE, 2014.

[270] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[271] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[272] Jaime Zabalza, Jinchang Ren, Jiangbin Zheng, Huimin Zhao, Chunmei Qing, Zhijing Yang, Peijun Du, and Stephen Marshall. Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. *Neurocomputing*, 185:1–10, 2016.

[273] Kun Zeng, Jun Yu, Ruxin Wang, Cuihua Li, and Dacheng Tao. Coupled deep autoencoder for single image super-resolution. *IEEE Transactions on Cybernetics*, 2015.

[274] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[275] Jingwei Zhang, Lei Tai, Peng Yun, Yufeng Xiong, Ming Liu, Joschka Boedecker, and Wolfram Burgard. Vr-goggles for robots: Real-to-sim domain adaptation for visual control. *IEEE Robotics and Automation Letters (RA-L)*, 2019.

[276] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[277] Xi Zhang, Yanwei Fu, Shanshan Jiang, Leonid Sigal, and Gady Agam. Learning from synthetic data using a stacked multichannel autoencoder. In *IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2015.

[278] Xuaner Zhang, Jonathan T. Barron, Yun-Ta Tsai, Rohit Pandey, Xiuming Zhang, Ren Ng, and David E. Jacobs. Portrait shadow manipulation. In *ACM Transactions on Graphics (TOG)*, 2020.

[279] Cairong Zhao, Xinbi Lv, Zhang Zhang, Wangmeng Zuo, Jun Wu, and Duoqian Miao. Deep fusion feature representation learning with hard mining center-triplet loss for person re-identification. *IEEE Transactions on Multimedia*, 22(12):3180–3195, 2020.

[280] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020.

[281] ZQ Zhao, P Zheng, ST Xu, and X Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019.

[282] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[283] Hao Zhou, Sunil Hadap, Kalyan Sunkavalli, and David W Jacobs. Deep single-image portrait relighting. In *International Conference on Computer Vision (ICCV)*, 2019.

[284] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European Conference on Computer Vision (ECCV)*, 2020.

[285] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2017.

Appendix A

List of publications

Steve Dias Da Cruz, Oliver Wasenmüller, Hans-Peter Beise, Thomas Stifter, Didier Stricker. (2020). SVIRO: Synthetic Vehicle Interior Rear Seat Occupancy Dataset and Benchmark. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Hans-Peter Beise, **Steve Dias Da Cruz**, Udo Schröder. (2021). On decision regions of narrow deep neural networks. In *Neural Networks*.

Steve Dias Da Cruz, Bertram Taetz, Oliver Wasenmüller, Thomas Stifter, Didier Stricker. (2021). Autoencoder Based Inter-vehicle Generalization for In-cabin Occupant Classification. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*.

Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, Didier Stricker. (2021). Illumination Normalization by Partially Impossible Encoder-Decoder Cost Function. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder for Synthetic to Real Generalization: From Simple to More Complex Scenes. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR,* ***Best Industry Related Paper Award****)*.

Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder Attractors for Uncertainty Estimation. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*.

Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, Didier Stricker. (2022). Autoencoder and Partially Impossible Reconstruction Losses. In *Sensors*.

Appendix B

Curriculum Vitae

Work experience

2018-2022 **Associate Scientist - Applied Mathematician**
IEE S.A. - Basics and Mathematical Models

2017 **Master student**
IEE S.A. - Basics and Mathematical Models

2015 **Intern**
SES Engineering s.à r.l - Space Operating Center

2014 **Intern**
SES Engineering s.à r.l - Space Operating Center

2011-2012 **Helpdesk Agent E-Banking**
Banque Raiffeisen

Education

2015-2017 **Master of Science in Mathematics - Grade: Very Good (17.2/20)**
University of Luxembourg
Master thesis: Detection of vital signs in presence of car vibrations - Mathematical modelling for simulation and signal processing

2012-2015 **Bachelor of Science in Mathematics - Grade: Very Good (16.4/20)**
University of Luxembourg
Bachelor thesis: Elliptic Curve Cryptography and the Weil pairing

Awards

2017 Mathematical Society of Luxembourg Student **Prize for outstanding performance in the Master of Mathematics Programme**

2017 Germain Dondelinger **Prize for the best Master's thesis** in the Faculty of Science, Technology and Communication

2015 **Prize for the most successful semester abroad** in the Faculty of Science, Technology and Communication