AUTOENCODER FOR SYNTHETIC TO REAL GENERALIZATION: FROM SIMPLE TO MORE COMPLEX SCENES

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning on synthetic data and transferring the resulting properties to their real counterparts is an important challenge for reducing costs and increasing safety in machine learning. In this work, we focus on autoencoder architectures and aim at learning 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 a new sampling technique, which matches semantically important parts of the image, while randomizing the other parts, leads to salient feature extraction and a neglection of unimportant parts. This helps the generalization to real data and we further show that our approach outperforms fine-tuned classification models.

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: Lee et al. (2020) introduced a method for training a quadrupedal robot on synthetic data by incorporating proprioceptive feedback. Akkaya et al. (2019) trained a robot hand to solve real Rubik's cubes by learning the model in a simulation only. Zhang et al. (2019) *made the robot feel at home* by translating the real world input data into synthetic data for their reinforcement learning agent. 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 because they are too dangerous, e.g. accidents. We focus on learning invariances empirically on synthetic data, which should transfer to real data, as opposed to constructing invariances as in equivariant neural networks (Romero & Hoogendoorn, 2020).

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 (Lee et al., 2020). We focus on training on synthetic data only and limit ourselves to autoencoder models which provide interesting properties due to their bottleneck design. The low-dimensional latent space of autoencoders can be subject to metric constraints (Hoffer & Ailon, 2015), allows for scene decomposition (Engelcke et al., 2020) and it is believed that latent factor disentanglement can be useful for downstream tasks (van Steenkiste et al., 2019). We 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 (Gondal et al., 2019) to generalize from synthetic (toy or realistic) to real images compared to Autoencoder, Variational Autoencoder (VAE) (Kingma & Welling, 2014), β-VAE (Higgins et al., 2017) and FactorVAE (Kim & Mnih, 2018). 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 use the partially impossible reconstruction loss (PIRL) (Dias Da Cruz et al., 2021) (matching semantically important parts while randomizing the other parts) and we propose a novel variation thereof. We extensively show that

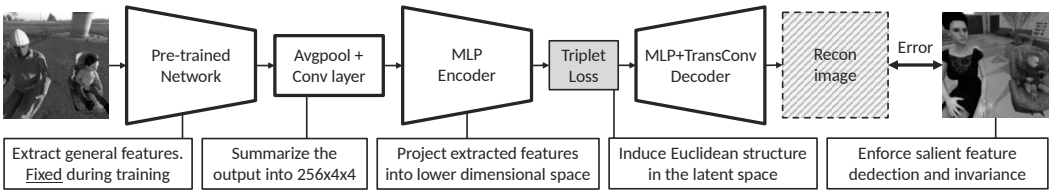


Figure 1: Impossible Instance Extractor Triplet Autoencoder (II-E-TAE) model architecture.

our variation is the driving force for the improved generalization capacities. Additionally, we induce structure in the latent space by a triplet loss regularization. We evaluate and justify the benefits of the different design choices on an automotive application focusing on occupancy classification in the vehicle interior. The challenge of training in a single vehicle interior and transferring results between different vehicle interiors has been investigated by Dias Da Cruz et al. (2021). 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.

2 Related Works

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 (Kang et al., 2019; Rao et al., 2020; Fang et al., 2018; Bewley et al., 2019). 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 in (Zhao et al., 2020). Another line of research uses generative adversarial networks (GAN) to make synthetic images look like real images or vice versa (Ho et al., 2020; Carlson et al., 2019). This requires both synthetic and real images, whereas we focus on training on synthetic images only. Part of our methodology is related to domain randomization (Tremblay et al., 2018), where the environment is being randomized, but Tremblay et al. (2018) 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 (Hinterstoisser et al., 2018), but neither with a dedicated sampling strategy nor in the context of autoencoders. While Tobin et al. (2017) focuses on localization and training on synthetic images only, the applicability is only tested on simple geometries. Although, we start our investigations on the simple dataset MPI3D, we increase the visual complexity by incorporating human models and child seats. Inoue et al. (2018) and Zhang et al. (2015) rely on the use of real images during training as well for the minimization of the synthetic to real gap for autoencoders. Recent advances on synthetic to real image segmentation (Chen et al., 2020; Yue et al., 2019; Pan et al., 2018) on the VisDA (Peng et al., 2017) 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. While our cost function variation is based on Dias Da Cruz et al. (2021), we show that our approach improves generalization while needing less demanding training data such that it can easily be applied to any commonly recorded dataset (i.e. no variations of the same scene are needed).

3 Method

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. Let $\mathcal{X} = \{X_i^j \,|\, 1 \leq i \leq N_v, 1 \leq j \leq N_s\}$ denote the training data, where each $X_i^j \in \mathbb{R}^{C \times H \times W}$ is the ith variation of scene j consisting of C channels and being of height H and width W. Let $X^j = \{X_i^j \,|\, 1 \leq i \leq N_v\}$ be the set of all variations i of scenery j and $\mathcal{Y} = \{Y^j \,|\, 1 \leq j \leq N_s\}$ 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. In the following, we will present the final model architecture as illustrated in Fig. 1 and we provide evidences for each design choice in Section 4.

(a) Vanilla reconstruction: Eq. 1 (b) Impossible: Eq. 2 (c) Impossible instance: Eq. 3

Figure 2: Illustration of the different input-target pairs for the autoencoder reconstruction loss.

3.1 Model Architecture: Extractor Autoencoder

By an abuse of terminology, we will refer to our method as a variation of vanilla autoencoders, although an encoder-decoder formulation would strictly speaking be more correct, because the goal will not be to reconstruct the input image exactly. We propose to apply ideas from transfer learning and use 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 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 ext_ω be a pre-trained classification model, referred to as *extractor*. For ease of notation, we define $\mathrm{e}_\phi(\mathrm{ext}_\omega(\cdot)) = \mathrm{ee}_{\phi,\omega}(\cdot)$. The model, using the vanilla reconstruction loss, can be formulated for a single input sample as

$$\mathcal{L}_R(X_i^j;\theta,\phi) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{e}_\phi(\mathrm{ext}_\omega(X_i^j))), X_i^j\right) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{ee}_{\phi,\omega}(X_i^j)), X_i^j\right), \tag{1}$$

where $\mathrm{r}(\cdot,\cdot)$ computes the error loss between target and reconstruction. We use the structural similarity index measure (SSIM) (Bergmann et al., 2018) and binary cross entropy (BCE), but our method is not limited to them. Model details are provided in the appendix A.2.1.

3.2 Sampling Strategy: Partial Impossible

An additional improvement to the autoencoder training approach is a dedicated sampling strategy for which we provide two variations. The first one is the partially impossible reconstruction loss (PIRL) as introduced by Dias Da Cruz et al. (2021) for illumination normalization. As our results will show, this also helps the transfer between synthetic and real images. 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 needs to focus on what remains constant. For random $a, b \in [0, N_v]$ and $a \neq b$:

$$\mathcal{L}_{R,I}(X_a^j;\theta,\phi) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{ee}_{\phi,\omega}(X_a^j)), X_b^j\right). \tag{2}$$

We refer to models using the PIRL by prepending an *I*, e.g. I-E-AE.

3.3 Sampling Strategy: Partial Impossible Class Instance

We propose a novel variation to further improve this strategy by sampling a target image of a different scene, but of the same class. This should cause the model to learn invariances with respect to certain class variations which are not important for the task at hand, e.g. clothes, human poses, textures. This sampling variation would be reflected in the reconstruction loss as follows

$$\mathcal{L}_{R,II}(X_a^j;\theta,\phi) = \mathrm{r}\left(\mathrm{d}_\theta(\mathrm{ee}_{\phi,\omega}(X_a^j)), X_b^k\right), \tag{3}$$

for random $a, b \in [0, N_v]$, $j \neq k$ and $Y^j = Y^k$. We refer to this method as impossible class instance sampling marked by prepending *II*, e.g. II-E-AE. It is important to notice that our novel variation can easily be applied to any common dataset. The sampling variations are visualized in Fig. 2.

3.4 Structure in the Latent Space: Triplet Loss

The final adjustment to our training strategy is the incorporation of the triplet loss regularization in the latent space (Hoffer & Ailon, 2015) to induce structure. This can be integrated by

$$\mathcal{L}_T(X_a^j;\phi) = \max\left(0, \left\|\mathrm{ee}_{\phi,\omega}(X_a^j) - \mathrm{ee}_{\phi,\omega}(X_b^k)\right\|^2 - \left\|\mathrm{ee}_{\phi,\omega}(X_a^j) - \mathrm{ee}_{\phi,\omega}(X_c^l)\right\|^2 + 0.2\right), \tag{4}$$

for random $a, b, c \in [0, N_v]$, $j \neq k \neq l$ and $Y^j = Y^k \neq Y^l$. We refer to this model as *triplet autoencoder* (TAE) either with or without using the PIRL. We can sample impossible target instances for the positive and negative triplet samples such that the total loss becomes (for some α and β):

$$\mathcal{L}(X_a^j; \theta, \phi) = \alpha \mathcal{L}_T(X_a^j; \phi) + \beta \left(\mathcal{L}_{R,II}(X_a^j; \theta, \phi) + \mathcal{L}_{R,II}(X_b^k; \theta, \phi) + \mathcal{L}_{R,II}(X_c^l; \theta, \phi) \right). \quad (5)$$

4 EXPERIMENTS

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 presented in Section 3. 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. All experiments use the same hyperparameters whenever possible. Training details and additional results are provided in the appendix and in our implementation.

We perform a baseline evaluation on MPI3D (Gondal et al., 2019), which provides simple and realistic renderings and real counterparts. We reduced the dataset to contain only the large objects. For a higher visual complexity, we use as synthetic images the SVIRO (Dias Da Cruz et al., 2020) dataset. TICaM (Katrolia et al., 2021) is used to evaluate the performance on a real dataset of a similar application. The latter datasets are grayscale images from the vehicle interior and consider the task of classification (empty, infant, child or adult) for each seat position. 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. Hence, we release an additional dataset, see Section 4.5 and A.1.4. 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. An additional ablation study shows that our novel 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 (LeCun et al., 1998) to generalizing to real images of digits (De Campos et al., 2009).

4.1 AUTOENCODERS STRUGGLE ON REAL IMAGES WHEN TRAINED ON SYNTHETIC IMAGES

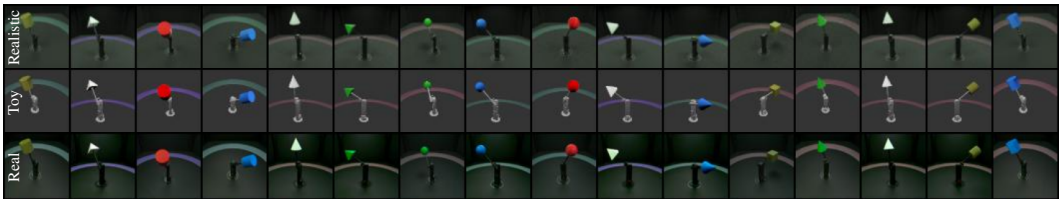
In the first, albeit naïve experiment we assumed that due to the bottleneck of autoencoders, the latter should generalize to some extent to real images when trained on synthetic ones. We trained convolutional autoencoders (AE) on the toy and realistic MPI3D images, respectively, and evaluated the resulting models on the real recordings. The first row of Fig. 3b 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. 3c.

4.2 AUTOENCODERS OVERFIT TO THE SYNTHETIC DISTRIBUTION

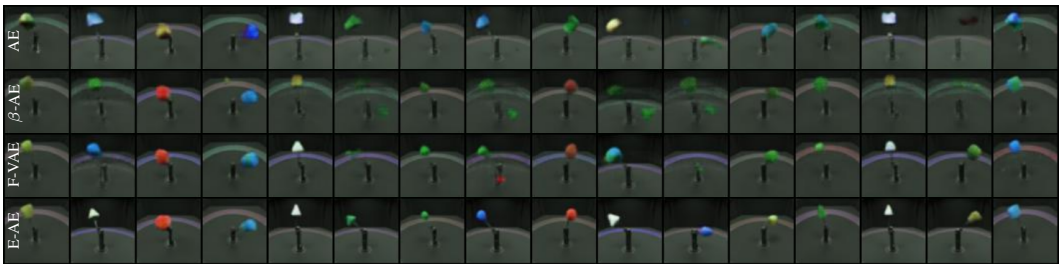
An immediate consequence of the results of the previous section is the assumption that the autoencoder overfits to the synthetic distribution and takes into consideration some artefacts (e.g. rendering noise). We followed the idea of Gondal et al. (2019) and trained Variational Autoencoder (VAE) (Kingma & Welling, 2014), β-VAE (Higgins et al., 2017) and FactorVAE (Kim & Mnih, 2018) 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. 3b 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 to be bad: see Fig. 3c.

4.3 MORE GENERAL INPUT FEATURES IMPROVE AUTOENCODER RECONSTRUCTIONS

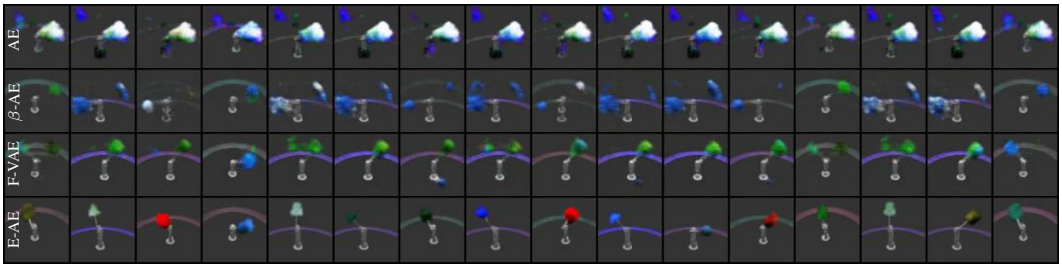
A small gap between the synthetic and real distribution can potentially be closed by a dedicated data augmentation approach to avoid overfitting to synthetic artefacts. Nevertheless, and while making sense, an abstraction from toy to real images cannot be achieved by means of simple data transformations or model constraints (e.g. denoising autoencoder). To this end we propose to use a



(a) Synthetic realistic and toy training data as well as real data used as input after training

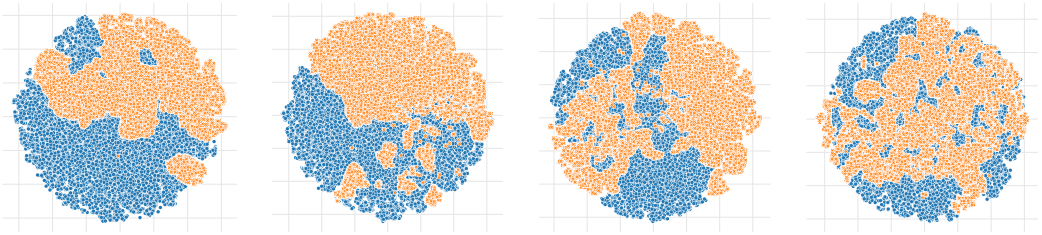


(b) Reconstruction of real data when being trained on realistic data.



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

Figure 3: Reconstruction of unseen real data for different autoencoders: Autoencoder (AE), β Variational Autoencoder (β-VAE), FactorVAE (F-VAE) and Extractor Autoencoder (E-AE).



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

Figure 4: t-SNE projection of the 10 dimensional latent space representation of the realistic training (blue circle) together with the real (orange cross) images. Autoencoder (AE), β Variational Autoencoder (β-VAE), FactorVAE and Extractor Autoencoder (E-AE). The extractor approach is the only method clustering both synthetic and real images together.

pre-trained feature extractor as presented in Section 3 and as defined by Eq. 1. In the following, 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. 3b and Fig. 3c, 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

Table 1: We report the L1, SSIM and LIPIPS (Zhang et al., 2018) norm between the reconstructions of the real images (unknown) and the corresponding synthetic (Synth.) training images (realistic or toy) or input images (Real). We report the mean of the norms across the dataset: for SSIM larger ↑ and for the others smaller ↓ is better. E-AE performs best.

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

together into a 2-dimensional representation using t-SNE. In Fig. 4 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 the appendix Fig. 8. Finally, we report in Table 1 a quantitative evaluation between the reconstructions of the real images against their synthetic training counterparts across all dataset images for different norms. We compute the same metrics between the real input images and their reconstruction to measure whether the semantics are being preserved : in all cases E-AE performs best. Additional results can be found in the appendix in Table 8 and reconstructions of synthetic input images in Fig 9. 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.

4.4 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 were optimistic to apply it to images of higher visual complexity, e.g. a vehicle interior. We trained the same model architecture as in the previous section, but with a 64-dimensional latent space, on images from the Tesla vehicle from SVIRO and the Kodiaq vehicle from SVIRO-Illumination dataset, respectively, and evaluated the model on the real TICaM images. Examples of the resulting model's reconstructions are plotted in Fig. 5 (b) and in the appendix Fig. 10. In both cases only blurry human models are being reconstructed, which is similar to the mode collapse in the first row of Fig. 3c. We concluded that more robust features are needed.

4.5 PARTIALLY IMPOSSIBLE RECONSTRUCTION LOSS HELPS GENERALIZATION

As defined by Eq. 2, a partially impossible reconstruction loss (PIRL) for autoencoders has proven to work well for image normalization (Dias Da Cruz et al., 2021). We hypothesized that the same approach could lead to a better generalization to real vehicle interiors. In a first approach, 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. 5 (c) where we trained on the Kodiaq images from the SVIRO-Illumination dataset.

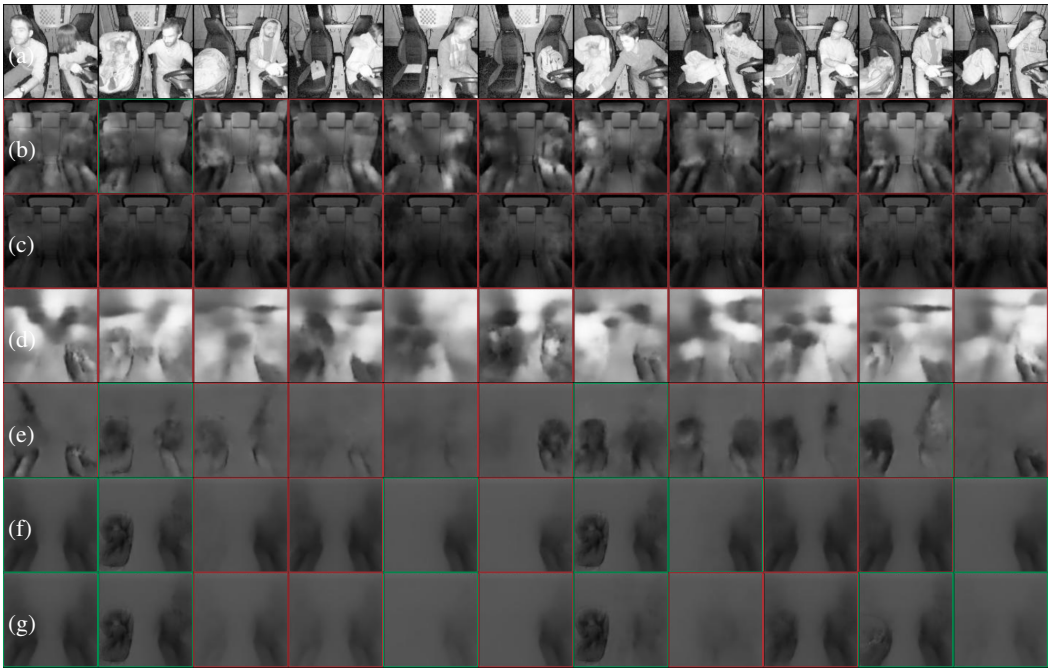


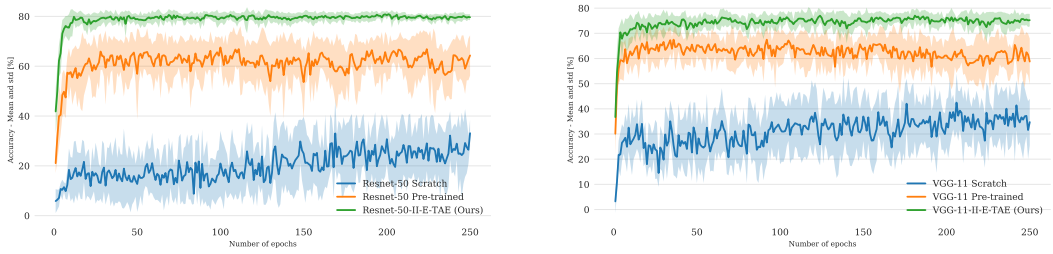
Figure 5: Reconstructions of unseen real data (a) from TICaM: (b) E-AE and (c) I-E-AE trained on Kodiaq SVIRO-Illumination, (d) E-AE, (e) I-E-AE, (f) II-E-AE and (g) II-E-TAE trained on our new dataset. A red (wrong) or green (correct) box highlights whether the classes are preserved.

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 5919 synthetic scenes where 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 HDRI images. Each scene was rendered using 10 different backgrounds. Examples from the dataset are shown in Fig. 7 in the appendix. 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. 2. When applied to real images, see Fig. 5 (e), the model better preserves the semantics 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. 5 (d) the model performs worse.

Finally, we extended this idea further with our novel PIRL loss 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 another image with a person on the left seat, potentially a different person with a different pose. This approach is formulated in Eq. 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. 5 (f). Moreover, this additional randomization improves classification accuracy as discussed in Section 4.7 and in Section 5. A visualization of the different input-target pair combinations can be found in Fig. 2. The dataset can been downloaded from this link (Google Drive - Anonymous user) and it will be made publicly available.

4.6 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: images of the same class should be mapped closely together and images of different classes pushed away. The triplet loss induces a more meaningful L^2-norm in the latent space (Dias Da Cruz et al., 2021) such that a k-nearest neighbor (KNN) classifier can be used in the next section. As the results of Fig. 5 (g) and in the appendix show,



(a) Resnet-50 (b) VGG-11

Figure 6: Comparison of the training performance distribution for each epoch over 250 epochs. II-E-TAE is compared against training the corresponding extractor from scratch or fine-tuning the layers after the features which are used by the extractor in our autoencoder approach.

these final improvements, together with the previous changes, yield the semantically most correct reconstructions. In the appendix we show that due to the triplet loss the nearest neighbour of (g) makes sense and yields a clearer reconstruction. The triplet loss without the PIRL is not sufficient and in Section 5 we show that the II-PIRL loss is the driving force for the improved performance.

4.7 KNN WITH TRIPLET LOSS OUT-PERFORMS FINE TUNED 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 neighbor classifier in the latent space and used our new dataset 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). We choose $k = \sqrt{N} = 115$, where N is the size of the training data together with its flipped version (Jirina et al., 2011). 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. 6 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, 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 2. See the appendix for training from scratch and Densenet-121 results. 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, 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.

5 DISCUSSION AND LIMITATIONS

We want to highlight that most of the contribution to the success of our introduced model variations stems from the novel II variation of the PIRL loss. To this end we trained several types of classifiers in the latent space of different autoencoder model variations and report the results in Table 3. The II variation of the PIRL loss 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 explcitily as a latent space constraints, compared to the implicit use by the II-PIRL.

Table 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 corresponding classification models significantly.

		TICaM		SVIRO	
Model	Variant	Mean	Max	Mean	Max
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
VGG-11	E-TAE	76.7 ± 2.3	81.5	78.6 ± 2.6	82.3
Resnet-50	E-TAE	$\underline{\mathbf{83.8 \pm 1.3}}$	$\underline{\mathbf{86.0}}$	85.8 ± 2.4	89.1
VGG-11	I-E-TAE	79.7 ± 2.1	$\underline{\mathbf{82.2}}$	$\underline{\mathbf{80.9 \pm 4.0}}$	$\underline{\mathbf{85.6}}$
Resnet-50	I-E-TAE	83.5 ± 1.3	85.6	89.2 ± 1.0	90.3
VGG-11	II-E-TAE	$\underline{\mathbf{81.0 \pm 0.6}}$	82.0	79.1 ± 3.9	84.8
Resnet-50	II-E-TAE	$\underline{\mathbf{83.7 \pm 0.5}}$	84.5	$\underline{\mathbf{93.0 \pm 0.8}}$	$\underline{\mathbf{94.1}}$

Table 3: 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 is due to the novel 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	$\underline{\mathbf{73.2 \pm 3.9}}$	$\underline{\mathbf{68.8 \pm 5.7}}$	66.9 ± 6.7	$\underline{\mathbf{83.7 \pm 1.9}}$	$\underline{\mathbf{79.8 \pm 2.7}}$	$\underline{\mathbf{81.4 \pm 2.2}}$
E-TAE	69.2 ± 3.4	66.4 ± 4.0	$\underline{\mathbf{68.7 \pm 2.2}}$	76.2 ± 2.3	71.2 ± 2.5	75.3 ± 2.5

The II variation of the PIRL loss 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. Under its current form there is no guarantee that, for example, facial landmarks or poses would be presereved. 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: e.g. 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.

Finally, we show that improvements reported in this work are not limited to the application in the vehicle interior. To this end, we trained models using the same design choices on MNIST LeCun et al. (1998) and evaluate the generalization onto real digits De Campos et al. (2009) in Fig. 12 and Table 10 in the appendix: similar improvements by the different design choices can be observed.

6 Conclusion

We introduced an autoencoder model which uses a pre-trained classification model as a feature extractor. Our results showed that the resulting model produces superior reconstructions for synthetic to real generalization. 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. Although only a simple k-nearest neigbor classifier is being used in the latent space, our proposed autoencoder model outperforms consistently and more robustly all classification model counterparts.

REPRODUCIBILITY STATEMENT

Reproducibility of our results is ensured by the code implementation provided in the supplementary material. Moreover, the model weights for all results reported in this work are available for download (anonymously): see the readme file in the supplementary material for download links. This readme file also explains how the code implementation can be used and how the results of the paper can be reproduced. The code implementation contains all the evaluation scripts necessary to get the results reported in this work. The appendix contains additional details about the training and models details as well as the datasets and data pre-processing part. For the latter, we also implemented pre-processing functions for all datasets used in our work together with links to the different datasets to download them. The datasets used in this work are all publicly available. The newly created dataset used in this work is also readily available for download. Lastly, the licenses of all datasets used are detailed in the readme file as well.

CODE OF ETHICS

Our proposed improvements on reducing the performance gap between synthetic and real images could reduce the necessity for human labelling and improve privacy since fewer human subjects are needed for data recordings. It would reduce the financial investment and time investment of companies, institutions and individuals. The question arises whether the usage of a pre-trained extractor introduces biases in the sub-sequent model, whether better disentanglement properties can be achieved and how it is affected by the choice of the latter. The first part of this work investigates model design choices on a simpler dataset without human subjects. The second part investigates the application in the vehicle interior. According to the authors of the TICaM dataset, written consent of the human participants was obtained together with their signature. Regarding the application in the vehicle interior - insights and improvements for the transfer from synthetic to real could be used to cover important edge cases (e.g. accidents) by simulations such that security and safety could be improved. The licenses for all publicly available datasets used in this work is referenced as well.

REFERENCES

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.

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.

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.

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.

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.

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

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.

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.

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.

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.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations (ICLR)*, 2020.

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.

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.

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

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.

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.

Daniel Ho, Kanishka Rao, Zhuo Xu, Eric Jang, Mohi Khansari, and Yunfei Bai. Retinagan: An object-aware approach to sim-to-real transfer. *arXiv preprint arXiv:2011.03148*, 2020.

Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition (SIMBAD)*, 2015.

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.

Marcel Jirina, MJ Jirina, and K Funatsu. Classifiers based on inverted distances. In *New fundamental technologies in data mining*, volume 1, pp. 369–387. InTech, 2011.

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.

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.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning (ICML)*, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), 2020.

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.

Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge, 2017.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

A APPENDIX

A.1 DATASET DETAILS

If not specified otherwise, all images have been centre cropped and resized to 128 pixels.

A.1.1 MPI3D

We used the synthetic realistic and toy images as well as the real images, but we restricted the dataset to use only large objects, since even for humans the small objects cannot always be distinguished reliably. The dataset can be downloaded from Github.

A.1.2 SVIRO

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. Since four classes are available per seat (empty, infant seat, child seat and adult) this results in a total of $4^3 = 64$ classes. The dataset can be downloaded from their website.

A.1.3 SVIRO-ILLUMINATION

For the classification evaluation we used all the training and test images from all the different vehicles. We used all the variations per scenes, i.e. not just a single variation per illumination variation. The dataset can be downloaded from their website.

A.1.4 OUR NEWLY RELEASED DATASET

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 high dynamic range images (HDRI). The latter were downloaded from https://hdrihaven.com/. Human models, child seats and infant seats were randomly placed as if they were located inside a vehicle, but no vehicle is visible. 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 created randomly 172 adults using http://www.makehumancommunity.org/ and we used 6 child seats and 7 infant seats which were textured using randomly one out of five textures. Since the dataset is synthetic, there are no consent and privacy concerns. We will release the dataset under the license CC BY-NC-SA 4.0. At the moment, it can be downloaded from this link (Google Drive - Anonymous user). Examples are visualized in Fig. 7. We noticed that a larger number of different human models increases the transferability to real images.

A.1.5 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: 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 $s03$, $s13$, $s04$, $s14$ 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. The dataset can be downloaded from their website.

Figure 7: Examples of sceneries with different backgrounds from the newly generated dataset.

A.2 TRAINING DETAILS

All our experiments were conducted using PyTorch 1.8. Pre-defined and models pre-trained on Imagenet were taken from torchvision 0.9.0.

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 $1e-4$ and weight decay of $1e-5$. We used a batch size of 64 and the only augmentation performed was a random horizontal flip. All models were trained for 100 epochs on MPI3D and 250 epochs for the other datasets.

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 and copied the single grayscale image channel twice along the channel dimension.

For the autoencoder training, we used the structural similarity index measure (SSIM) Bergmann et al. (2018) or the binary cross entropy (BCE) to measure the error between reconstruction and target image. We used PyTorch MS-SSIM Gongfan (2019) to compute the SSIM. In Eq. 5 we chose $\alpha = 1$ and $\beta = 1$. 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. Further, we used the ReLU activation function. In case of a triplet loss, we used the swap parameter of Pytorch to make the negative mining more challenging Balntas et al. (2016). 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 partially impossible reconstruction loss was used, the target images for the positive and negative samples are chosen to be partially impossible as well.

A.2.1 MODEL DETAILS

The autoencoder model architecture details are provided in Table 4, 6 and 7. Regarding the pre-trained models, we used the output of the following layers to retrieve the extracted features. The notations is according to the torchvision model definitions:

```
VGG-11
(16): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(17): ReLU(inplace=True)

Resnet-50
(5): Bottleneck(
  (conv1): Conv2d(1024, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (relu): ReLU(inplace=True)
)

Densenet-121
(denselayer24): _DenseLayer(
  (norm1): BatchNorm2d(992, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (relu1): ReLU(inplace=True)
  (conv1): Conv2d(992, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (relu2): ReLU(inplace=True)
  (conv2): Conv2d(128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
)
```

Table 4: Model architecture for AE, VAE and β-VAE on MPI3D

Encoder	Decoder
Input: 3 x 64 x 64	Input: 10
Conv, 4x4, 32, padding 1, stride 2 ReLU	FC, 256, bias True ReLU
Conv, 4x4, 32, padding 1, stride 2 ReLU	FC, 1024, bias True ReLU
Conv, 4x4, 64, padding 1, stride 2 ReLU	ConvTranspose, 4x4, 64, padding 1, stride 2 ReLU
Conv, 4x4, 64, padding 1, stride 2 ReLU	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
FC, 256, bias True ReLU	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
FC, 10, bias True (twice in case of VAE)	ConvTranspose, 4x4, 3, padding 1, stride 2 Sigmoid

Table 5: Model architecture for FactorVAE on MPI3D. The model is exactly the same as the VAE model and uses the following discriminator.

Discriminator
Input: 10
FC, 1000, bias True LeakyReLU(0.2)
FC, 1000, bias True LeakyReLU(0.2)
FC, 1000, bias True LeakyReLU(0.2)
FC, 1000, bias True LeakyReLU(0.2)
FC, 1000, bias True LeakyReLU(0.2)
FC, 2, bias True

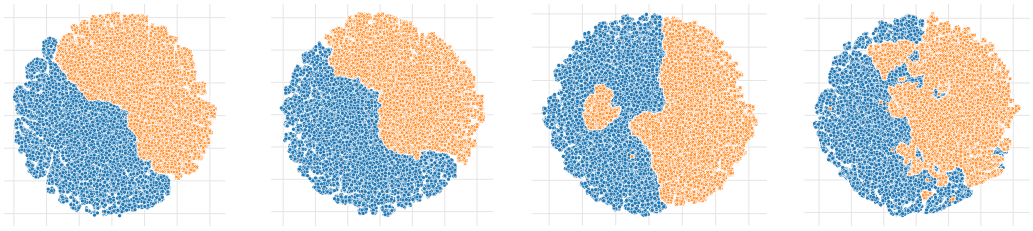
Table 6: Model architecture for E-AE on MPI3D. The extractor is fixed during training.

Extractor + Summarizer + Encoder	Decoder
Input: 3 x 224 x 224	Input: 10
VGG-11 extractor after 7th Conv layer + ReLU Avgpool, 2x2, stride 2, padding 0	FC, 256, bias True ReLU
Conv, 4x4, 256, padding 0, stride 1 ReLU	FC, 1024, bias True ReLU
FC, 256, bias True ReLU	ConvTranspose, 4x4, 64, padding 1, stride 2 ReLU
FC, 10, bias True	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
	ConvTranspose, 4x4, 3, padding 1, stride 2 Sigmoid

Table 7: Model architecture for E-AE on SVIRO, SVIRO-Illumination and TICaM. C is the channel dimension which is 1 for all datasets. The extractor is fixed during training.

Extractor + Summarizer + Encoder	Decoder
Input: C x 224 x 224	Input: 64
VGG-11 extractor after 7th Conv layer + ReLU Avgpool, 2x2, stride 2, padding 0	FC, 256, bias True ReLU
Conv, 4x4, 256, padding 0, stride 1 ReLU	FC, 4096, bias True ReLU
FC, 256, bias True ReLU	ConvTranspose, 4x4, 64, padding 1, stride 2 ReLU
FC, 64, bias True	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
	ConvTranspose, 4x4, 32, padding 1, stride 2 ReLU
	ConvTranspose, 4x4, C, padding 1, stride 2 Sigmoid

A.3 ADDITIONAL RESULTS

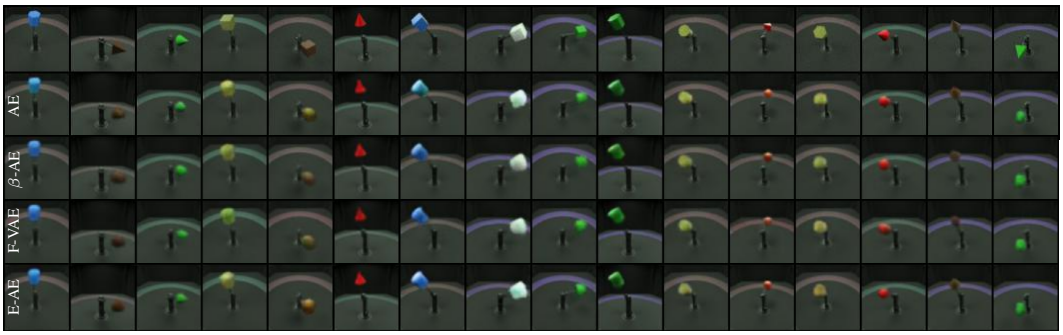


(a) AE (b) β-VAE (c) FactorVAE (d) E-AE

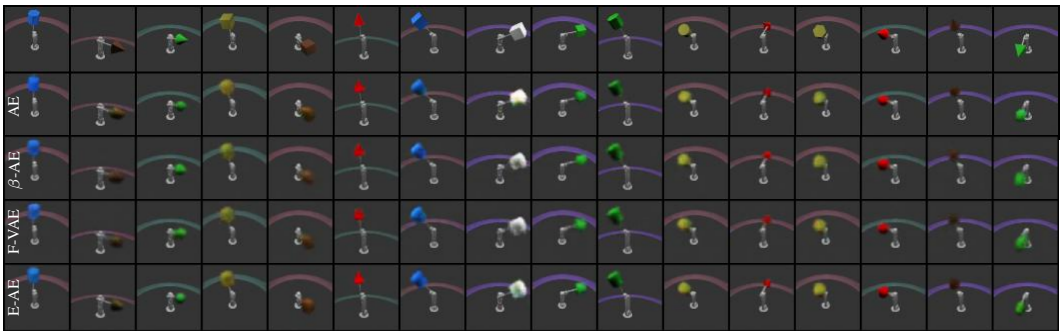
Figure 8: t-SNE projection of the 10 dimensional latent space representation of the toy training (blue circle) together with the real (orange cross) images. Autoencoder (AE), β Variational Autoencoder (β-VAE), FactorVAE and Extractor Autoencoder (E-AE). When trained on toy images, our extractor approach performs still best although the synthetic-real distributions are not as overlapped as if trained on realistic images.

Table 8: We report the L1, SSIM and LIPIPS (Zhang et al., 2018) norm between the reconstructions of the real images (unknown) and the corresponding synthetic training images (realistic or toy). We report the mean of the norms across the entire reduced dataset: for SSIM larger ↑ and for the others smaller ↓ is better. E-AE performs best. Some models used SSIM, others BCE during training.

Trained on	Model	Variant	L1 ↓	SSIM ↑	LPIPS ↓
Toy	AE	BCE	768	0.559	0.412
Toy	AE	SSIM	932	0.558	0.347
Toy	E-AE (ours)	BCE	291	**0.896**	**0.095**
Toy	E-AE (ours)	SSIM	**177**	**0.899**	**0.103**
Toy	VAE	BCE	659	0.497	0.338
Toy	β-VAE	BCE, $\beta = 4$	710	0.527	0.311
Toy	β-VAE	BCE, $\beta = 8$	406	0.709	0.258
Toy	FactorVAE	BCE, $\gamma = 10$	521	0.660	0.262
Toy	FactorVAE	BCE, $\gamma = 30$	447	0.710	0.344
Toy	FactorVAE	BCE, $\gamma = 50$	430	0.712	0.221
Realistic	AE	BCE	373	0.841	0.211
Realistic	AE	SSIM	568	0.832	0.195
Realistic	E-AE (ours)	BCE	**220**	**0.917**	**0.071**
Realistic	E-AE (ours)	SSIM	**251**	**0.921**	**0.081**
Realistic	VAE	BCE	482	0.740	0.197
Realistic	β-VAE	BCE, $\beta = 4$	372	0.810	0.176
Realistic	β-VAE	BCE, $\beta = 8$	384	0.794	0.189
Realistic	FactorVAE	BCE, $\gamma = 10$	**218**	0.880	0.151
Realistic	FactorVAE	BCE, $\gamma = 30$	**244**	0.862	0.161
Realistic	FactorVAE	BCE, $\gamma = 50$	391	0.779	0.164



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



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

Figure 9: Reconstruction of realistic and toy training data for different autoencoders: Autoencoder (AE), β Variational Autoencoder (β-VAE), FactorVAE (F-VAE) and Extractor Autoencoder (E-AE).

Table 9: For each experiment, the best performance (in percentage) on real vehicle interior images (TICaM) across all epochs is taken and then the mean and maximum of those values across all 10 runs is reported. For the same backbone model extractor, our approach outperforms the vanilla classification models significantly. The model weights achieving the maximum performance per run are also evaluated on SVIRO where they perform better as well.

	Dataset	TICaM		SVIRO	
	Dataset size	13356		11959	
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	76.7 ± 2.3	81.5	78.6 ± 2.6	82.3
Resnet-50	E-TAE	$\mathbf{83.8 \pm 1.3}$	**86.0**	85.8 ± 2.4	89.1
Densenet-121	E-TAE	78.5 ± 2.4	81.8	86.7 ± 1.3	88.2
VGG-11	I-E-TAE	79.7 ± 2.1	**82.2**	$\mathbf{80.9 \pm 4.0}$	**85.6**
Resnet-50	I-E-TAE	83.5 ± 1.3	85.6	89.2 ± 1.0	90.3
Densenet-121	I-E-TAE	77.2 ± 1.7	79.3	$\mathbf{90.4 \pm 1.3}$	92.1
VGG-11	II-E-TAE	$\mathbf{81.0 \pm 0.6}$	82.0	79.1 ± 3.9	84.8
Resnet-50	II-E-TAE	$\mathbf{83.7 \pm 0.5}$	84.5	$\mathbf{93.0 \pm 0.8}$	**94.1**
Densenet-121	II-E-TAE	$\mathbf{79.3 \pm 1.3}$	**81.5**	89.9 ± 1.8	**92.3**

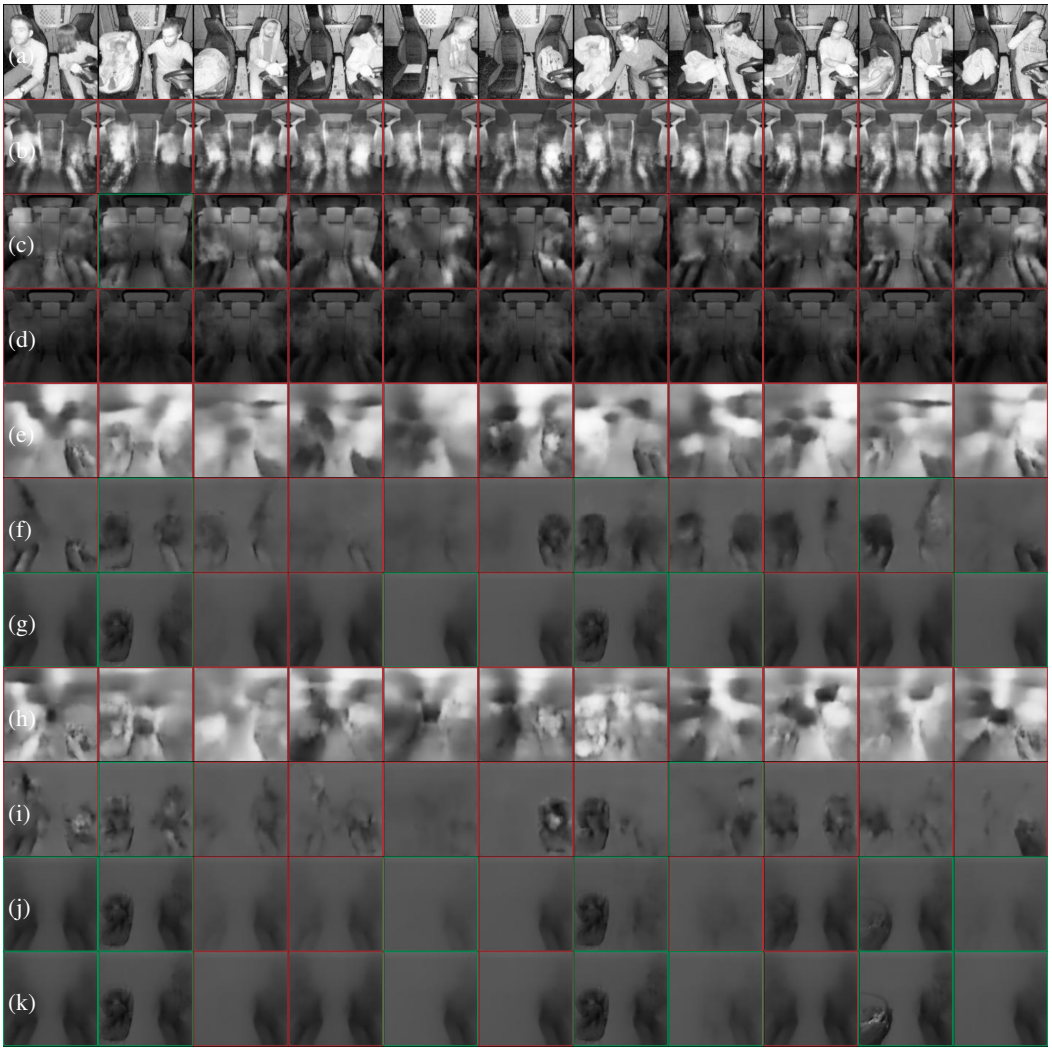
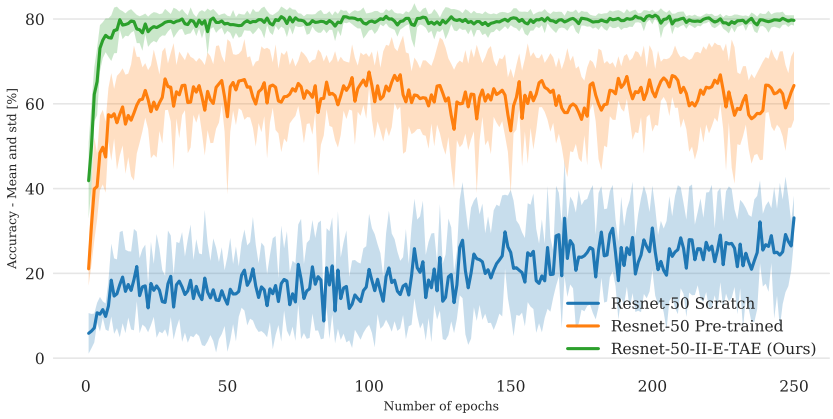


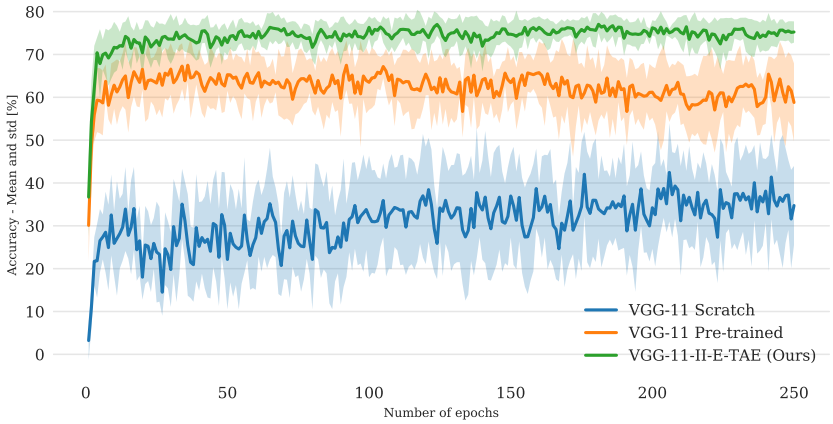
Figure 10: Reconstruction results of unseen real data (a) from the TICaM dataset: (b) E-AE Trained on Tesla SVIRO, (c) E-AE Trained on Kodiaq SVIRO-Illumination , (d) I-E-AE Trained on Kodiaq SVIRO-Illumination , (e) E-AE, (f) I-E-AE, (g) II-E-AE, (h) E-TAE, (i) I-E-TAE, (j) II-E-TAE and (k) Nearest neighbour of (j). Examples (e)-(k) are all trained on our new dataset. A red (wrong) or green (correct) box highlights whether the semantics are preserved by the reconstruction.

Table 10: Different model architecture variations trained on MNIST. Then different classifiers were trained on the latent space representation of the training data and evaluated on real images of digits. Models were trained for 20 epochs using a latent dimension of 64 and MSE reconstruction loss. See Fig. 12 for the corresponding reconstruction results and input images.

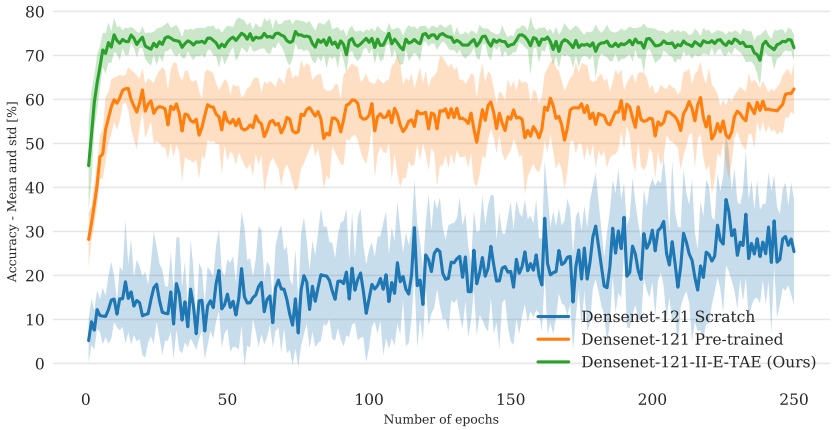
Model	KNN	RForest	SVM
AE	15.7	12.5	11.6
TAE	11.1	11.6	8.4
II-AE	27.8	20.2	23.6
II-TAE	21.8	17.9	23.9
E-AE	27.3	23.1	26.5
E-TAE	26.1	19.1	23.3
II-E-AE	65.	61.9	65.6
II-E-TAE	64.1	63.7	63.7



(a) Resnet-50



(b) VGG-11



(c) Densenet-121

Figure 11: Comparison of the training performance distribution for each epoch over 250 epochs. II-E-TAE is compared against training the corresponding extractor from scratch or fine-tuning the layers after the features which are used by the extractor in our autoencoder approach.

Figure 12: Reconstruction of real input images of digits by models trained on MNIST. Similar to the vehicle interior, the II-PIRL loss provides the best class preserving reconstructions. The latter is supported by the quantiative results in Table 10.