Illumination Normalization by Partially Impossible Encoder-Decoder Cost Function

Supplementary Material

Steve Dias Da Cruz [1,2,3]
steve.dias-da-cruz@iee.lu

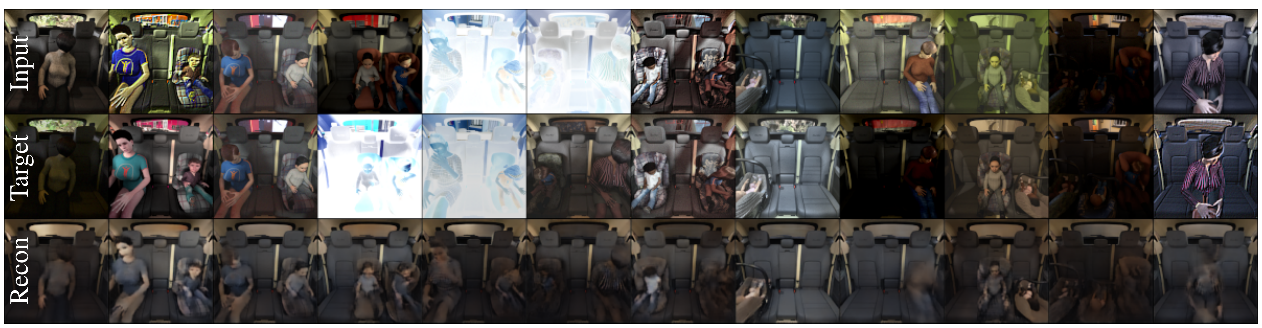
Bertram Taetz [3]
bertram.taetz@dfki.de

Thomas Stifter [1]
thomas.stifter@iee.lu

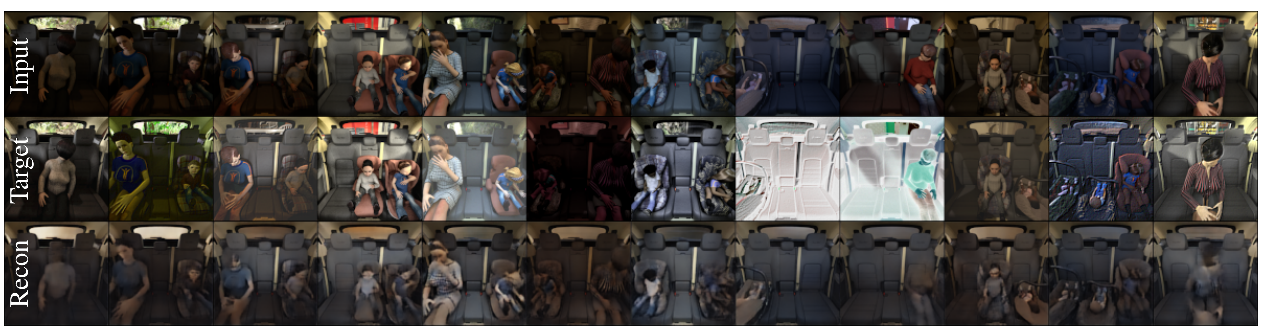
Didier Stricker [2,3]
didier.stricker@dfki.de

[1] IEE S.A. [2] University of Kaiserslautern [3] German Research Center for Artificial Intelligence

1. Augmentation based impossible reconstruction



(a) Input and target augmented



(b) Target only augmented

Figure S1: One can augment the input and target images differently or augment the target images only, instead of using identical sceneries under different lightning conditions for the input-target pairs. Our proposed cost functions still provides valid reconstructions, though the objects are blurrier than for the application presented in the main paper. 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.

2. Triplet loss illustration

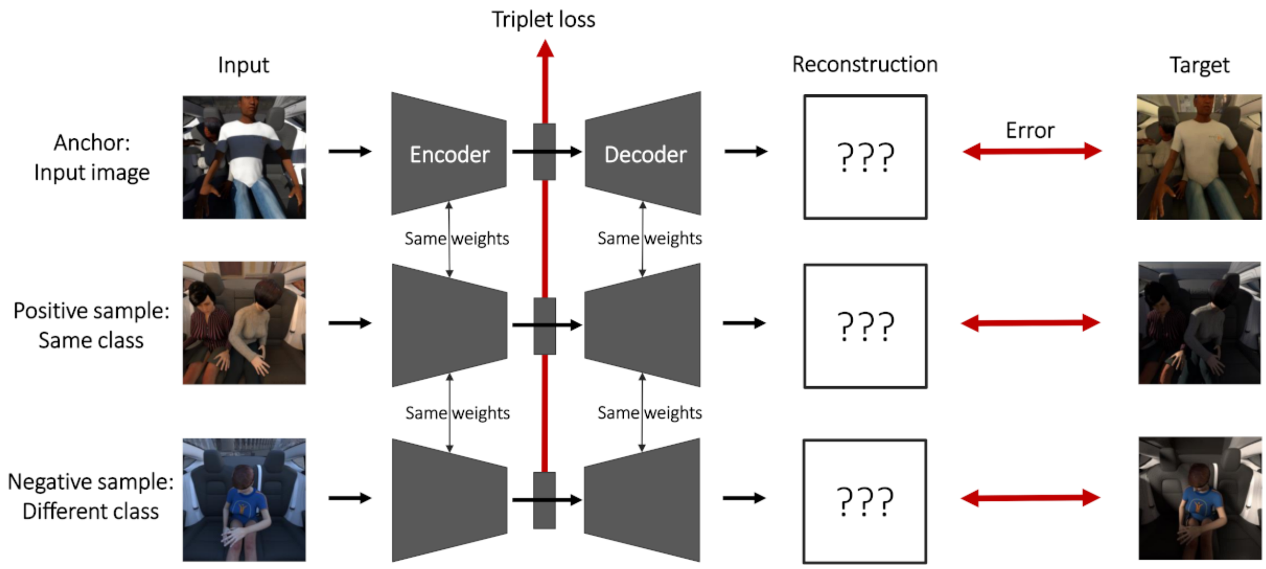


Figure S2: Illustration of the triplet loss when applied to SVIRO-Illumination. We chose the positive sample to be of the same class as the anchor image (but from a different scenery) and the negative sample to differ only on one seat (*i.e*. change only the class on a single seat w.r.t. the anchor image). Notice the difference in illumination of the target image w.r.t. input image in order to apply our proposed partially impossible cost function.

3. SVIRO-Illumination - Additional examples

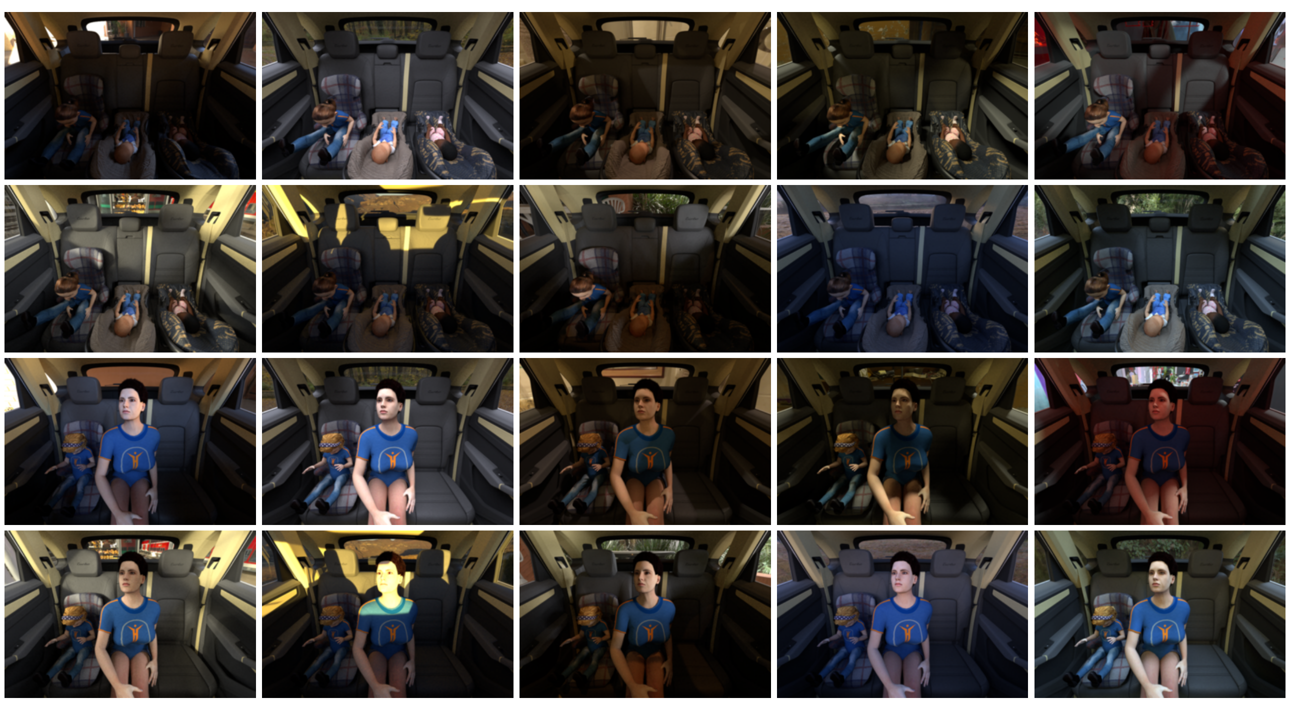


Figure S3: Additional example images for the Cayenne vehicle from SVIRO-Illumination.

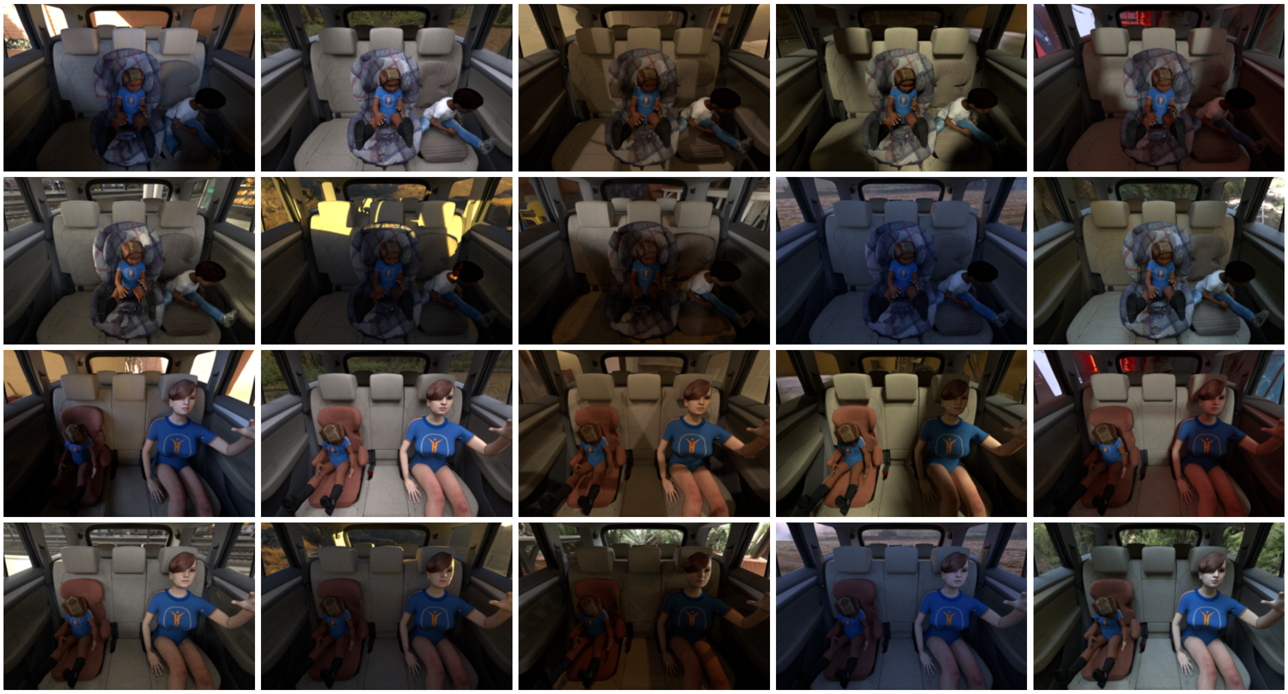


Figure S4: Additional example images for the Kodiaq vehicle from SVIRO-Illumination.

Figure S5: Additional example images for the Kona vehicle from SVIRO-Illumination.

4. Encoder-decoder model architecture

Table S1: The encoder-decoder model architecture used in the main paper. Left: Encoder model, Middle: Latent space model and Right: Decoder model. C specifies the number of channels of the input image and LatentDimension is the dimension of the latent space to use (2 and 16 for the paper). The encoder is based on the VGG-11 model, but we use only half the amount of filters per channel. The decoder is almost the reverse of the encoder model for which the number of filters needed to be adapted to match the output shape. We use a sigmoid activation for the output to ease the reconstruction. The latent space model uses as input the output of the encoder model. The decoder model uses as input the output of the latent space model.

Encoder based on VGG-11
Input: C x 224 x 224
Conv 3x3, 32 BatchNorm ReLU
MaxPooling 2x2
Conv 3x3, 64 BatchNorm ReLU
MaxPooling 2x2
Conv 3x3, 128 BatchNorm ReLU Conv 3x3, 128 BatchNorm ReLU
MaxPooling 2x2
Conv 3x3, 256 BatchNorm ReLU Conv 3x3, 256 BatchNorm ReLU
MaxPooling 2x2
Conv 3x3, 256 BatchNorm ReLU Conv 3x3, 256 BatchNorm ReLU
MaxPooling 2x2

Latent space
Input: 256 x 7 x 7
Flatten
FC 784 ReLU
FC LatentDimension
FC 784 ReLU FC 12544 ReLU
Reshape

Decoder based on VGG-11
Input: 256 x 7 x 7
Nearest neighbour up-sampling x2
Conv 3x3, 256 BatchNorm ReLU Conv 3x3, 256 BatchNorm ReLU
Nearest neighbour up-sampling x2
Conv 3x3, 256 BatchNorm ReLU Conv 3x3, 128 BatchNorm ReLU
Nearest neighbour up-sampling x2
Conv 3x3, 128 BatchNorm ReLU Conv 3x3, 64 BatchNorm ReLU
Nearest neighbour up-sampling x2
Conv 3x3, 32 BatchNorm ReLU
Nearest neighbour up-sampling x2
Conv 3x3, C
Sigmoid

5. 2 and 16 dimensional latent space representations with PCA and T-SNE

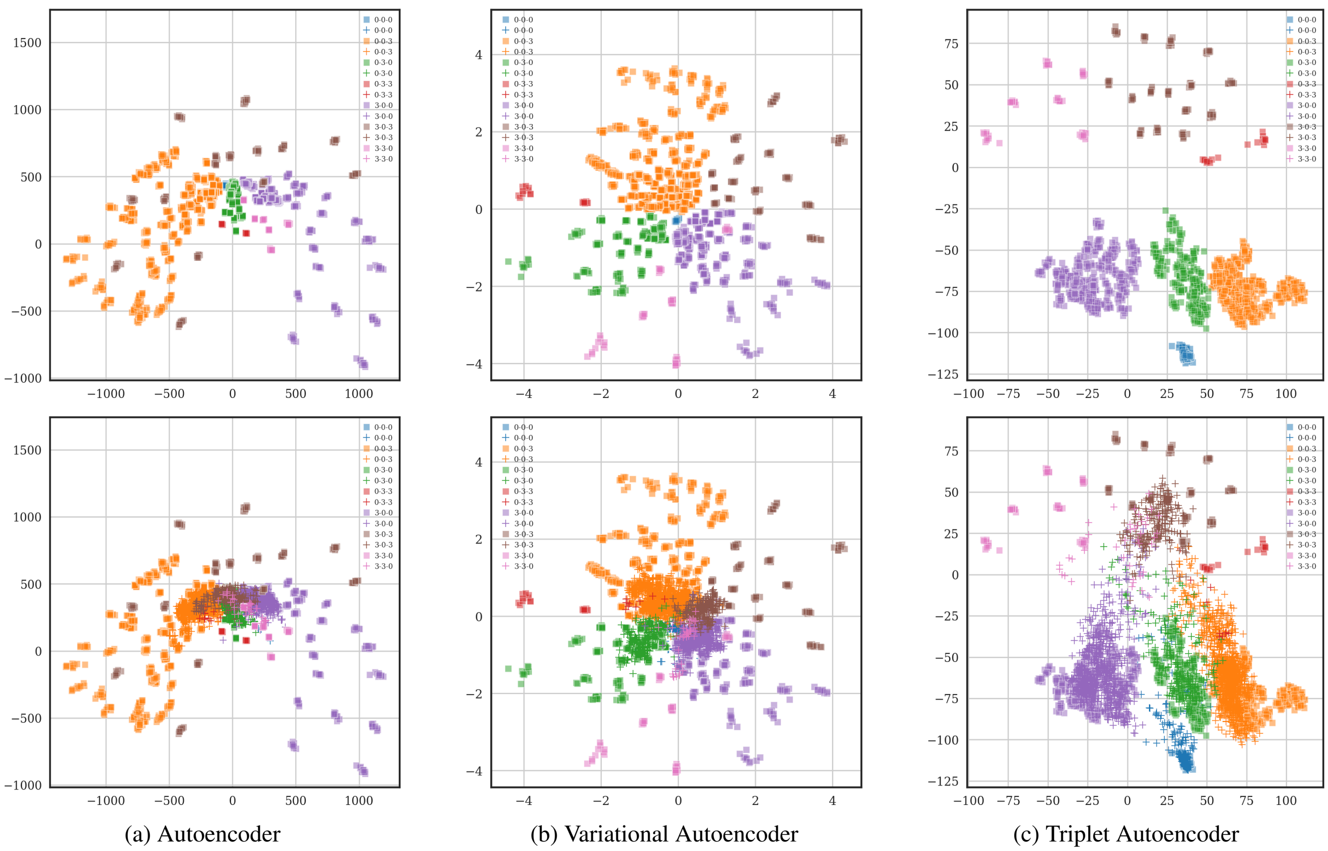
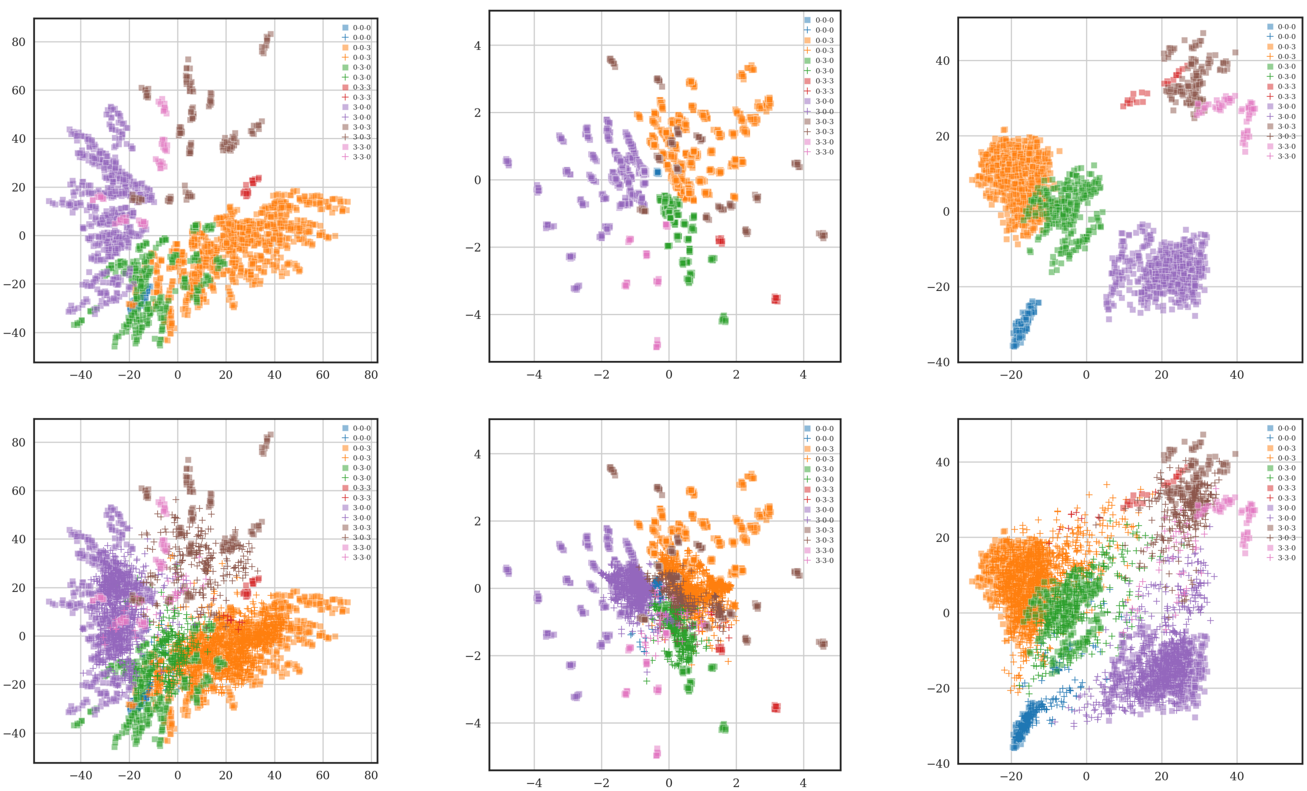


Figure S6: Additional plot for the latent space representation of the main paper. For ease of visualization, we plot the training distribution only ■ (first row) and the training distribution ■ together with the test distribution + (second row). The triplet autoencoder produces a better test distribution which could potentially be used for outlier detection.

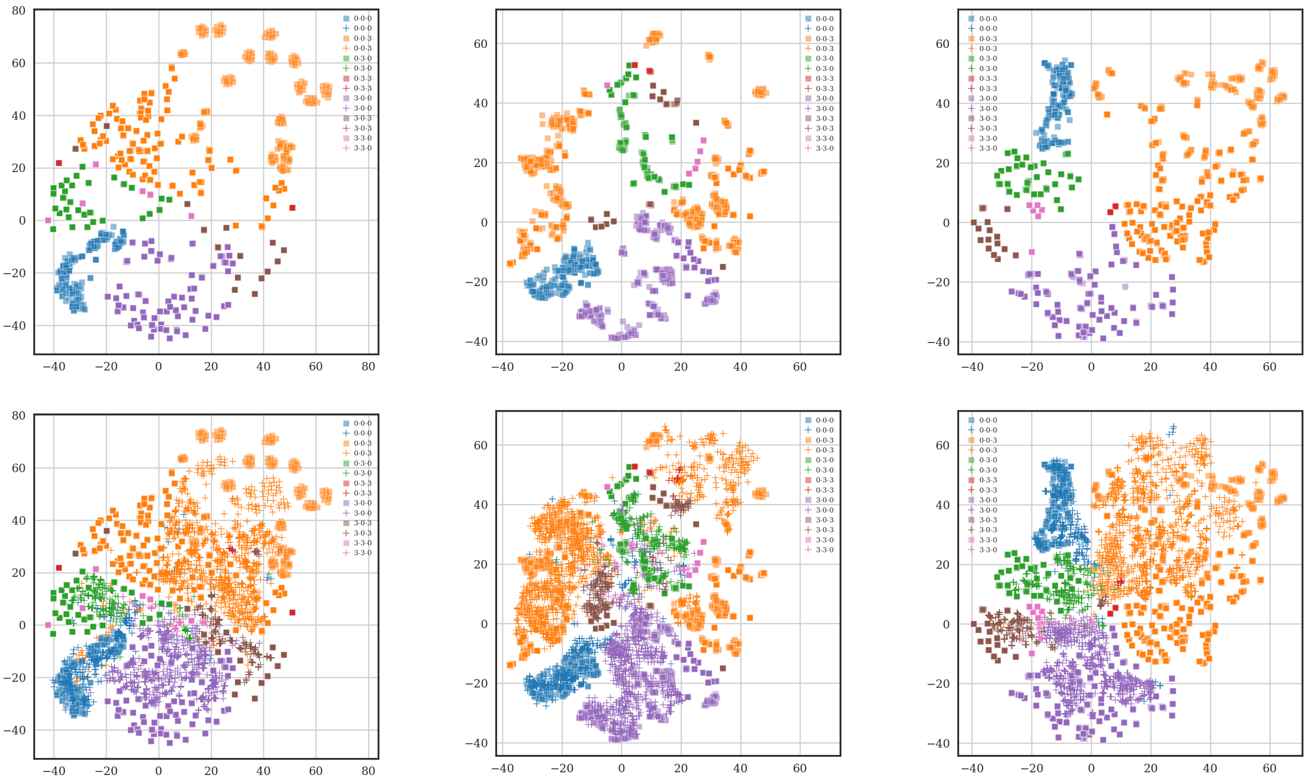


(a) Autoencoder

(b) Variational Autoencoder

(c) Triplet Autoencoder

Figure S7: Different autoencoders were trained with a latent dimension of 16. We report the first two principal components of a principal component analysis (PCA). Both, the training and test distribution were computed and the PCA was calculated on both distributions together. We plot the training distribution only ■ (first row - test points are made invisible) and the training distribution ■ together with the test distribution **+** (second row). The latter choice was only made to ease visualization and highlight the test samples easier. The first two principal components of the triplet autoencoder provide a good separation, especially considering that a PCA is a linear mapping.



(a) Autoencoder (b) Variational Autoencoder (c) Triplet Autoencoder

Figure S8: Different autoencoders were trained with a latent dimension of 16. We report the two-dimensional T-SNE projection. Both, the training and test distribution were computed and the T-SNE was learned on both distributions together. We plot the training distribution only ■ (first row - test points are made invisible) and the training distribution ■ together with the test distribution **+** (second row). The latter choice was only made to ease visualization and highlight the test samples easier. The triplet autoencoder T-SNE projection is the most consistent ones with almost no wrong test sample projections.