Split-AE: An Autoencoder-based Disentanglement Framework for 3D Shape-to-shape Feature Transfer

1st Sneha Saha
Honda Research Institute Europe
Offenbach, Germany
sneha.saha@honda-ri.de

2nd Leandro L. Minku
School of Computer Science
University of Birmingham
Birmingham, UK
l.l.minku@cs.bham.ac.uk

3rd Xin Yao
School of Computer Science
University of Birmingham, UK
Department of Computer Science and Engineering, SUSTech, China
x.yao@cs.bham.ac.uk

4th Bernhard Sendhoff
Honda Research Institute Europe
Offenbach, Germany
bernhard.sendhoff@honda-ri.de

5th Stefan Menzel
Honda Research Institute Europe
Offenbach, Germany
stefan.menzel@honda-ri.de

***Abstract*—Recent advancements in machine learning comprise generative models such as autoencoders (AE) for learning and compressing 3D data to generate low-dimensional latent representations of 3D shapes. Learning latent representations that disentangle the underlying factors of variations in 3D shapes is an intuitive way to achieve generalization in generative models. However, it remains an open problem to learn a generative model of 3D shapes such that the latent variables are disentangled and represent different interpretable aspects of 3D shapes. In this paper, we propose *Split-AE*, which is an autoencoder-based architecture for partitioning the latent space into two sets, named as content and style codes. The content code represents global features of 3D shapes to differentiate between semantic categories of shapes, while style code represents distinct visual features to differentiate between shape categories having similar semantic meaning. We present qualitative and quantitative experiments to verify feature disentanglement using our Split-AE. Further, we demonstrate that, given a source shape as an initial shape and a target shape as a style reference, the trained Split-AE combines the content of a source and style of a target shape to generate a novel augmented shape, that possesses the distinct features of the target shape category yet maintains the similarity of the global features with the source shape. We conduct a qualitative study showing that the augmented shapes exhibit a realistic interpretable mixture of content and style features across different shape classes with similar semantic meaning.**

***Index Terms*—Shape analysis, autoencoders, feature disentanglement, point clouds**

I. INTRODUCTION

In automotive digital development, designers use computer-aided design (CAD) and engineering (CAE) tools for various design ideation tasks. Typically, designers manually generate, modify or update existing geometries according to their design vision. However, manually exploring a large design set might be challenging and time-consuming. Therefore, for supporting designers, we envision a human-machine cooperative design system that utilizes existing shape databases and rapidly assesses a novel shape by transferring distinct features from one shape to another. Here, we explore recent developments in machine learning, which allows deep neural networks to learn 3D geometric data for combining the essence of existing shapes from different classes. For example, cars in the automotive domain can be divided into classes, such as convertibles, utility vehicles, passenger cars, etc. (Fig. 1). To generate novel car shapes that combine the geometric features of a convertible and a utility vehicle, we propose an unsupervised approach for learning visual geometric features of 3D shapes from each class. Further, we perform 3D shape-to-shape geometric features transfer to generate novel design variations such that the generated designs reflect shapes from existing shape classes with modified features.

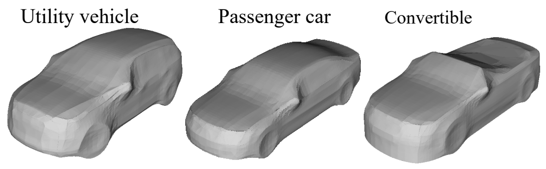


Fig. 1. Car shapes from 3 car classes.

Prior research on 3D shape-to-shape feature transfer represented each 3D shape with two characteristics [1]: *Content* and *style*. Content refers to the global structure of a 3D shape, and style refers to the distinctive parts of the shape that discriminate it from others with the same content. However, we assume for a given 3D shape that the content estimates whether the underlying structure of the 3D shape is coherent as a whole and distinguishes across other semantic categories, while the style refers to the localized regions or distinctive parts of the given 3D shape that allows grouping of shapes into shape classes. As an example, in Fig. 1, sports utility vehicles (SUV) in the utility vehicle class and sedans in the passenger car class have distinctive shape parts such as unique roof structures, while the convertible car has a removable roof. Thus, our research considers style as the distinct shape parts that differentiate between shapes across shape classes. Further, to generate a novel 3D shape by providing a 3D shape as a style reference, we propose to perform a 3D shape-to-shape style transfer. For instance, style transfer from a convertible car to an SUV car should generate a novel realistic car shape with a removable roof structure like the convertible car class while having a similar exterior design similar to other parts of the SUV car.

The difficulty of 3D style transfer has been tackled initially by utilizing point-wise shape deformation approaches [2], [3]. Recent developments in state-of-the-art deep learning algorithms offer a promising approach for unsupervised feature extraction of 3D shapes. Geometric deep learning models, such as autoencoders (AEs) [3], [4] and variational autoencoders (VAEs) [5], learn low-dimensional latent representations of 3D shapes, which allows them to efficiently alter 3D shapes by modifying the latent variables. Segu *et al.* [1] directly address the 3D style transfer problem by learning a style-dependent generative model (3DSNet) for shapes from two classes. Given a pair of 3D shapes as input to the 3DSNet, the network can directly generate new shapes by transferring style features between the input shapes, since the network is trained in parallel for reconstruction and shape translation tasks. Nonetheless, the prior network still lacks a generalized method that addresses style transfer for multiple classes.

In this paper, we propose an autoencoder-based architecture called *Split-AE* that is able to disentangle the content and style of 3D shapes into two separate latent representations, each of which is shared between shapes from multiple classes and across semantic categories. First, we train the Split-AE with 3D shapes to generate two separate content and style latent spaces for the 3D shapes. Next, we utilize quantitative and qualitative measures to indicate whether the content and style features of trained Split-AE represent different underlying factors of variations in 3D shapes, which hints at the position of style features before performing style transfer. Second, given a 3D shape as a style target reference, we utilize the trained Split-AE to generate a novel augmented shape by transferring the style of the target shape to a source shape. Our main contributions are to propose and validate the Split-AE architecture for content and style disentanglement of 3D shapes from multiple classes and semantic categories to perform 3D style transfer.

The remainder of this paper is organized as follows: In Section II, we review deep generative models for learning 3D shapes and approaches for disentangling the latent representation for 3D style transfer. We then introduce in Section III the architecture of our proposed Split-AE model and the pre-processing steps of the data set. We also detail the experimental setup for generating augmented shapes by style transfers and describe qualitative measures to evaluate the effectiveness of style transfers. In Section IV, we present the experimental settings to train and analyze the content and style features learned by Split-AE. In Section V, we discuss the performance of Split-AE with respect to its ability to generate novel augmented shapes by style transfers. Finally, in Section VI, we present a summary and conclusion of our paper.

II. Prior Art

We first review autoencoders for learning and reconstructing 3D shape representations. Next, we relate the autoencoder network for generating content and style latent representations to pave the way towards latent space-based style transfer for 3D shapes.

A. Autoencoders for content and style disentanglement of 3D shape representations

3D shapes are frequently represented using point clouds, voxels, and polygonal meshes, among which point clouds are the simplest representation for 3D shapes in many tasks [6]. 3D point clouds are a list of data point coordinates in 3D space that are sampled directly from the surface of CAE models and preserve sufficient geometric details for most applications.

Qi *et al.* proposed a deep neural network architecture for learning 3D point clouds, called PointNet [7]. The PointNet architecture addresses the permutation invariance of points in point clouds by handling the input data with point-wise operators followed by a global operator, namely *max-pooling*. Recently developed popular deep generative models for learning point cloud data that extend the PointNet architecture are autoencoders (AEs), variational autoencoders (VAEs), and generative adversarial networks (GANs). An AE architecture comprises a bottleneck layer to represent a compact representation of the input data, the so-called latent space. This layer also divides the network into two parts: an encoder, which maps the input data to the latent variables, and a decoder, which generates 3D point clouds from the variables in the latent space. This latent space provides compressed 1D vector representations of input shapes, which are used for 3D shape manipulation tasks.

3D shapes are highly structured, with different latent variables controlling separate aspects of a 3D shape. Segu *et al.* [1] proposed to divide the latent representation of 3D shapes into two factors: content and style. However, the content space is shared between shapes across two classes, but the style latent space is class-dependent, i.e., for shapes from each class, a separate encoder needs to be trained to generate class-specific style space. In contrast, we tailored our Split-AE to address the content-style disentanglement for 3D shapes across multiple classes and domains, by assuming two separate latent spaces [8]: content and style shared between 3D shapes.

B. Enforcing disentanglement in the latent representation

Research on training autoencoder-based architectures for multiple tasks [9] relies on a combination of geometric loss and penalties that operate on the latent space to ensure that the space contains the information we wish to encode. Earlier analysis on types of penalties [10] showed that the unsupervised disentanglement of the latent space is fundamentally impossible without minimal supervision or inductive biases on models and data. Hence, some form of implicit supervision in terms of labels of the input data is necessary to achieve disentangled latent variables that represent discriminative features of 3D shapes. To enhance the discrimination of features learned by the latent variables, prior research imposes an additional classification task in terms of cross-entropy loss to train their networks [11], [12].

In our research, to divide and enforce the latent variables for learning the content and style of shapes from different domains (having separate semantic meanings) and multiple classes, we utilize two classification tasks. The first task is to enforce the content space to learn the underlying shared global representations of 3D geometries and distinguish shapes across semantic categories or domains. The second task is to enforce the style space to learn in an unsupervised fashion the distinctive local features of 3D shapes from each shape class, such that it leads to an efficient disentanglement of class-essential and class-redundant information [12]. Further, following the idea of this unsupervised approach of shape-class classifications to encode distinctive features in style space helps to extend our architecture to learn shapes from multiple classes, which is not possible with prior approaches [1], [13].

III. Methodology

Given a source shape x_1 and a target shape x_2 as a reference from different classes within the same domain, our goal is to transfer the distinct features or style of x_2 to x_1 ($x_2 \rightarrow x_1$), such that the newly generated shape x_{12} has style features of the target shape and content features similar to the source shape. To implicitly learn the separate content and style features of 3D shapes for performing shape-to-shape style transfer, we describe our approach in the following sections.

In Section III-A, we first introduce the pre-processing steps for sampling point cloud data from surfaces of 3D shapes and describe the shape categories. In Section III-B, we introduce the architecture of our proposed Split-AE and depict the loss functions for training the network. Next, we provide details on the quantitative measures used to evaluate the performance of the network (Section III-C). In Section III-D, we depict the experimental setup to generate novel shapes by style transfer using our network.

A. Pre-Processing of 3D Point Clouds

We choose shapes from $\mathcal{K}$ number of domains with different semantic meanings and from each domain, we consider shapes from different classes, such that each class of shapes has distinct geometric features and also possess common global features between the classes within a domain. We refer to the total number of classes as $\mathcal{X}$ from $\mathcal{K}$ domains. We sample 3D point clouds from polygonal meshes of $\mathcal{X}$ classes from $\mathcal{K}$ domains, using the shrink-wrapping algorithm utilized in [14]. The shrink-wrapping samples N points uniformly from the surface of 3D meshes and generates organized point clouds based on the vertex assignment of the shrink-wrapped mesh. Therefore, each shape is a 3D point cloud consisting of N points.

In this work, we choose shapes from two domains ($\mathcal{K} = 2$), e.g., cars, and airplane domains of the ShapeNetCore data set [15], which is a large data set of 3D shapes. From these two domains, we consider shapes from a total five classes ($\mathcal{X} = 5$) with their respective class labels $\mathcal{C}_i$ $(i = 1, .., \mathcal{X})$ (Fig. 2).

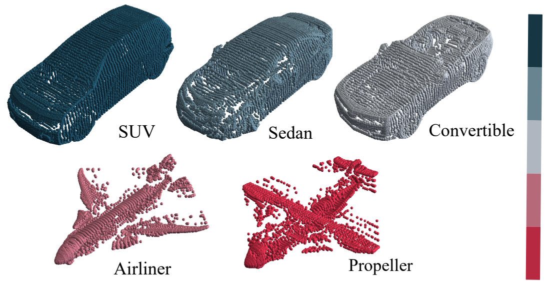


Fig. 2. Visualization of one shape from each of the total 5 classes from the car and airplane domains. The scale shows distinct colors to represent class labels $\mathcal{C}_i$ $(i = 1, .., 5)$. Each shape is color-coded based on their class label.

In the car domain, we selected shapes from three classes: SUV, sedan, and convertible, and two classes from the airplane domain: airliners and propellers. We chose the shapes from the above classes, such that shapes from each class have distinct geometric characteristics that are visually observable, which we refer to as style. However, our assumptions of content and style for shapes from discrete domains are different. For example, we assume that style for shapes from the three classes of the car domain refer to distinctive roof structures, and style for two shape classes in the airplane domain refers to distinctive wing structures (Fig. 2). Therefore, in the next section, we introduce our proposed Split-AE architecture to learn the style of these shapes.

B. Split-AE Architecture

The Split-AE architecture consists of two encoders (E_c and E_s) for mapping input shapes to two latent space branches: the content (Z_c) and style (Z_s) space (Fig. 3). Both of the encoder networks follow the architecture proposed in [3] with five 1D convolutional layers. The first four layers are activated with the rectified linear unit (ReLU) and the last layer with a hyperbolic tangent function. After the fifth convolution layer, a max-pooling operator reduces the dimensionality of the input shape to two L-dimensional vectors: content vector z_c and style vector z_s. Next, these two latent vectors z_c and z_s, are given as input to two multi-layer perceptrons (MLPs): MLP-1 and MLP-2, respectively. In the first content space Z_c branch, the MLP-1 classifies the input shape into one of the $\mathcal{K}$ domain categories. The architecture of the MLP-1 consists of two layers, where the first layer with 10 hidden neurons is activated with ReLU, and the last layer with $\mathcal{K}$ hidden neurons is activated with a soft-max function. The index of the selected domain of the input shape corresponds to the index of the neuron with maximum soft-max activation. In the second style space Z_s branch, the MLP-2 classifies the input shape into one of the $\mathcal{X}$ number of classes. The architecture of the MLP-2 is similar to the MLP-1 in the first branch. But in the output layer of the MLP-2, there are $\mathcal{X}$ number of neurons based on the total number of shape classes, and the index of the selected class corresponds to the index of the neuron with maximum softmax activation. Next, we pad the 2 vectors z_c and z_s to form a $2 * L$ dimensional vector z, which is given as input to the decoder (D). The decoder comprises three fully connected layers, with only the first two layers activated by ReLU.

a) Loss Function: The loss function (L) for training the Split-AE architecture (Fig. 3) consists of 3 terms as follows:

$$\begin{aligned} L &= L_{recon} + \alpha L_{DC} + \beta L_{SC} \\ L_{recon} &= MSD(x_i, x_o) = \frac{1}{N}\sum_{j=1}^{N} \|x_{i,j} - x_{o,j}\|_2^2 \\ L_{DC} &= \sum_{k=1}^{\mathcal{K}} -d_k(x_i) log(\rho_k(x_i)), \quad \mathcal{K} = 2 \\ L_{SC} &= \sum_{l=1}^{\mathcal{X}} p_l MSD(x_i, x_{P_l}), \quad \mathcal{X} = 5 \end{aligned} \tag{1}$$

The first term computes the reconstruction error (L_{recon}) between the input point cloud ($x_{i,j}$) and output reconstructed point cloud ($x_{o,j}$), with N number of points in each point-cloud. Since the network is trained on organized point clouds, i.e., the point-correspondence between the point clouds is known, it allows the utilization of a simple loss function, here: mean-square distance (MSD) [14], [16].

The second term (L_{DC}) in Eq. 1 obtains a partition between shapes from $\mathcal{K}$ different domains in the content space Z_c using cross-entropy loss (Eq. 1), where each input point cloud (x_i) has a ground truth domain label d_k vector. The domain prediction probability vector of the input shape (x_i) is given by ρ_k.

The third term (L_{SC}) in the loss function in (Eq. 1) tries to obtain class-wise partitions of the data into $\mathcal{X}$ number of classes by computing a weighted average of the MSD between the $\mathcal{X}$ chosen prototypes, one from each class and the input shape. Each prototype defines the most representative 3D point cloud shape from each class, which we selected by calculating the mean point cloud of shapes from each class. In Fig. 3, we show the selected prototypes x_{P_i} $(i = 1, .., \mathcal{X})$ for $\mathcal{X}$ classes, considering $\mathcal{X} = 5$. The weight for each prototype is defined by the corresponding probability output p_i $(i = 1, .., 5)$ of the second MLP-classifier (MLP-2). Thus, the third term is minimized when the MLP-2 yields the highest selection probability p_i to the prototype with the lowest MSD value (the highest similarity). The shape-classifier tries to obtain a class-wise partition of the data in the style space Z_s.

All terms of the loss function (Eq. 1) differ by several orders of magnitude. To balance these differences, we introduce two hyper-parameters α and β to scale the classification losses, respectively.

C. Metrics for evaluating the trained Split-AE

For comparing the performance of the trained Split-AE with the baseline models, we considered two quantitative metrics for estimating the reconstruction capability and style class classification accuracy of the three models with respect to the shapes in the unseen test set.

a) Reconstruction error: We consider the Chamfer distance (CD) [4] to calculate the reconstruction error between the input and the generated output point clouds.

b) Style class classification accuracy: We train three random forest-based multi-class classification models to map the latent representations (z_c, z_s and z) of the training set to their respective class labels $\mathcal{C}_i$ $(i = 1, .., 5)$. The goal of the classification is to take any latent representation of the test set samples as input and assign it to any one of the $\mathcal{X}$ style classes and measure the prediction accuracy of the classifier. This enables us to check disentanglement of latent variables based on style classes.

Further, for qualitative evaluation of the features learned by the two latent spaces, we utilize the feature visualization approach in [17] for visual inspections of regions of the input space mapped by the latent variables.

D. Augmented shape generation by style transfer

The trained Split-AE generates a latent vector z_i for each input shape x_i, where each z_i consists of two parts: Content z_{c_i} and style z_{s_i} code. Given two shapes, one source shape x_1 from the $\mathcal{X}_1$ class and another target shape x_2 from the $\mathcal{X}_2$ class within the same domain, we want to evaluate the effectiveness of the style transfer from shape $x_2 \rightarrow x_1$. Both of the source and target shapes are represented with content and style codes, such as, z_{s1}, z_{c1} $(x_1 \sim z_1 = (z_{c1}, z_{s1}))$ and z_{s2}, z_{c2} $(x_2 \sim z_2 = (z_{c2}, z_{s2}))$, respectively.

Our goal of transferring the style code z_{s2} from $x_2 \rightarrow x_1$ results in generating an augmented shape x_{12} $(z_{12} = (z_{c1}, z_{s2}))$, such that x_{12} is perceptually more similar to the target x_2 than the source x_1. Specifically, the augmented shape x_{12} should possess distinct traits of shape x_2 such that it belongs to the target car class $\mathcal{X}_2$.

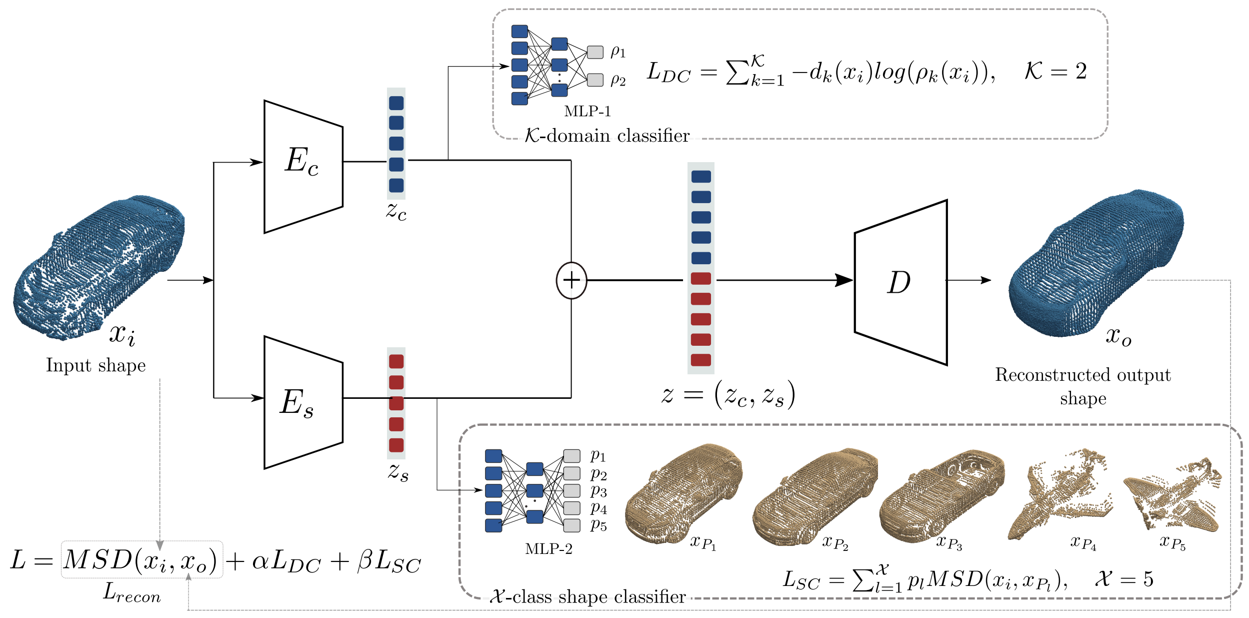
Therefore, to evaluate the success of style transfer between source-target pairs of each domain, we measure the distance similarity of the generated shapes with the source-target pair and also calculate the classification accuracy of how many generated shapes belong to their target shape class using the pre-trained multi-class classifier (Section III-C).

IV. Analysis of Features Learned by Split-AE

In this section, we first present evaluation results, where we compare the Split-AE with baseline models. Next, we present a series of experiments to verify that the learned content and style codes represent various aspects of 3D shapes.

A. Model training

a) Data: We selected 1500 shapes out of 3500 shapes in the car domain consisting of shapes from three car classes and 1100 shapes out of 4045 airliner shapes consisting of shapes from airliner and propeller classes from the ShapeNetCore data set [15]. Each shape consists of N points, here: N = 24578. In total, we considered 2600 point cloud shapes in our data set



$L_{DC} = \sum_{k=1}^{\mathcal{K}} -d_k(x_i) log(\rho_k(x_i)), \quad \mathcal{K} = 2$
MLP-1
$\mathcal{K}$-domain classifier
E_c
z_c
x_i
Input shape
E_s
z_s
$z = (z_c, z_s)$
D
x_o
Reconstructed output shape
MLP-2
$L = MSD(x_i, x_o) + \alpha L_{DC} + \beta L_{SC}$
L_{recon}
$L_{SC} = \sum_{l=1}^{\mathcal{X}} p_l MSD(x_i, x_{P_l}), \quad \mathcal{X} = 5$
$\mathcal{X}$-class shape classifier

Fig. 3. Split-AE architecture and data flow for training the network.

and the coordinates of each point cloud were normalized to the range $[0.1, 0.9]^3$, preserving the aspect ratio of the shapes.

For training our network, we split the considered data set into a 90% training set and 10% test set.

b) Training the Split-AE: We trained the model with a 5D latent vector ($L = 5$) for each content and style code using the Adam optimizer [18] with a learning rate $\eta =$ 5E-04. The training data was organized into batches of 50 shapes and the network was trained for 700 epochs. To optimize the hyper-parameters, α and β, in the loss function in Eq. 1, we set the hyper-parameter values, $\alpha = .03$, and $\beta = .1$, which resulted in an acceptable reconstruction accuracy and provided equal importance between the two classification loss functions.

c) Training baseline models: We considered two baseline models. As a first baseline model, we selected PC-AE [19], as we adapted the encoder-decoder architecture of this network for our Split-AE. The PC-AE network was trained utilizing mean square distance (MSD) as a reconstruction loss function. As a second baseline model, we considered a PC-AE-classifier, where the similar PC-AE network is trained with an additional cross-entropy classification loss function to discriminate all classes equally. We trained both networks with a 10D latent vector using the Adam optimizer with a learning rate of $\eta =$ 5E-04 for 700 epochs.

We implemented the architectures using Python with TensorFlow®for computation on Graphics Processing Units (GPUs). All networks were trained with two CPUs Intel®Xeon®Silver, clocked at 2.10 GHz, and with two GPUs NVidia®GeForce®RTX 8000 with 48GB each. In all cases, the networks were trained on a single GPU.

To reduce the sampling bias on the random split, we repeated the data split and trained all the baseline models and Split-AE 3 times and reported the mean and the standard deviation over 3 runs in Section IV-B. However, for evaluation and transfer of learned style features, we visualized the results from one of the trained Split-AE in Section IV-C and V.

B. Validation of the model

To validate the implementation of our proposed architecture, we compared the reconstruction capability and style-class classification performance of the Split-AE with baseline models on the test set using the quantitative metrics from Section III-C.

a) Reconstruction capability of models: We calculated the reconstruction losses between the reconstructed output point clouds and their corresponding input point clouds in the training and test sets using Chamfer distance (CD) in Table I. For a 95% confidence interval, we observed that Split-AE achieved lower reconstruction errors (low CD), which are better than the errors of the PC-AE and PC-AE-classifier. The reconstruction quality of Split-AE is closer to the PC-AE-classifier, which indicates that training a network with a classification loss improves the quality of the reconstructed shapes. However, the run-time of Split-AE is much higher compared to other models, and this is due to the extended architecture and multitask loss functions for training Split-AE.

TABLE I
RECONSTRUCTION PERFORMANCE OF THE MODELS.

	Split-AE	PC-AE	PC-AE-Classifier
CD_{train}	**(7.54±.77)E-05**	(9.70±.12)E-05	(8.69±.41)E-05
CD_{test}	**(8.00±.31)E-05**	(10.0±.20)E-05	(8.94±.50)E-05
Run-time	4hrs 50mins 20secs	**1hr 40mins 20secs**	2hrs 54mins 32 secs

b) Style class classification accuracy based on latent representation: In this experiment, we measured the classification accuracy of the trained multi-class classifiers (Section

III-C) to predict the class labels of the test samples based on their latent representations (Table II). Both the baseline models learn content and style features of 3D shapes in a common latent space (Z). Therefore, we calculated style class classification accuracy of the two baseline models based on the latent space Z opposed to Split-AE, where we calculated classification accuracy based on content (Z_c), style (Z_s) and combined space (Z). Higher accuracy relates to the model's ability to generate distinctly separable latent representations of the test samples based on style classes.

TABLE II
STYLE CLASS CLASSIFICATION ACCURACY OF THE MODELS.

	Split-AE	PC-AE	PC-AE-classifier
Content space (Z_c)	0.953±.004	-	-
Style space (Z_s)	0.972±.006	-	-
Combined space (Z)	**0.979±.003**	0.935±.005	**0.977± .004**

In Table II, we observed that the 5D style space in Split-AE shows higher accuracy compared to the 5D content space. These results stress the effectiveness of our disentangling approach, as the style space holds higher class-specific information in comparison to the content space, even though the style space in Split-AE learns the class labels based on shape similarity. Further, in the combined latent space of Split-AE, classification accuracy improved and became similar to the PC-AE-classifier, signifying models trained with classification loss generate better disentangled latent representations based on style classes. In the next set of experiments, we provide an intuitive interpretation of the features learned by the content and style variables of the Split-AE for representing the underlying factors of variations in 3D shapes.

C. Disentangled shape features learned by Split-AE

a) Content and style variables: For qualitative analysis of the learned features and better understanding of the complex model architectures, we utilized the feature visualization approach [17] for projecting learned network features into human-comprehensible space, i.e., in the 3D input space. To verify the relation between latent variables and the point distributions of a 3D shape in the input space, we projected the activation values of the last convolutional layer of both encoders (E_c and E_s) onto an input point cloud. In Fig. 4, we show the visualization of features learned by content ($z_{c_i}, i = 1..., 5$) and style ($z_{s_i}, i = 1, .., 5$) variables of the trained Split-AE on an SUV car shape.

By analyzing the visualizations across multiple input shapes, we observed that each feature maps to similar regions in the input space rather than similar geometric characteristics of the input shapes [17]. Also, the regions of activation in the input space are different for style z_{s_i} and content variables z_{c_i}, where z_{c_i} mapped to wider regions in the input space compared to z_{s_i}, which focused on distinct smaller regions in the input space with higher activation values. Therefore,

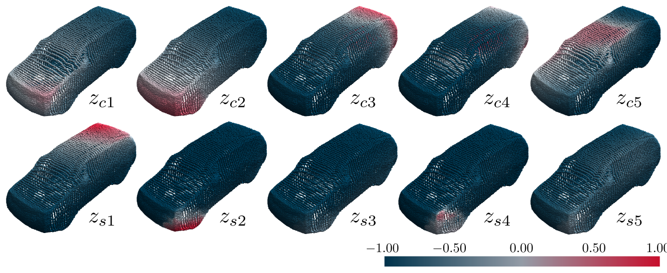


Fig. 4. Feature visualization method applied to a point cloud representation of an SUV car shape obtained using Split-AE with 5D content and 5D style spaces. The scale shows the activation values of the visualized features.

we concluded that style latent variables map to distinctive localized regions in the input space to ensure separability between classes.

b) Linear combination of features: As a second test to verify that content variables hinder learning distinct traits specific to the shapes of each car class, we considered a mean representation x_μ generated by estimating a mean content $z_{c\mu}$ and style code $z_{s\mu}$ of the car shapes from the training set, such as $x_\mu \sim z_\mu = (z_{c\mu}, z_{s\mu})$. Next, we considered three distinct shapes (x_1, x_2 and x_3) from 3 car classes and combined the content codes of these shapes with the mean style code $z_{s\mu}$ of the shape x_μ in Fig. 5. We expected that the 3 generated shapes would potentially represent shapes similar to the mean shape x_μ and cannot be classified into their 3 respective car classes, since the distinct content codes learn an overall global representation of 3D shapes.

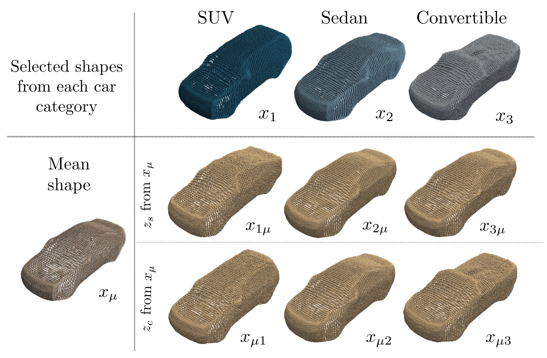


Fig. 5. First row: $x_{1\mu}, x_{2\mu}$ and $x_{3\mu}$ shapes generated by combining content code of x_1, x_2 and x_3 with the style code of x_μ, respectively. Second row: $x_{\mu 1}, x_{\mu 2}$ and $x_{\mu 3}$ shapes generated by combining the content code of x_μ with style code of x_1, x_2 and x_3, respectively.

In Fig. 5, selected shapes from each car class are represented by content and style codes z_{ci} and z_{si} ($x_i \sim z_i = (z_{ci}, z_i), i = 1, 2, 3$). We generated augmented shapes ($x_{1\mu}, x_{2\mu}$ and $x_{3\mu}$) by combining distinct content codes (z_{c1}, z_{c2} and z_{c3}) with the mean style code z_{s_μ} such as $x_{1\mu} \sim z_{1\mu} = (z_{c1}, z_{s\mu})$. We observed that the generated shapes $x_{1\mu}, x_{2\mu}$ and $x_{3\mu}$ are similar to each other without having distinctive shape parts (roof structures similar in all 3 shapes). Therefore, we conclude that content variables do not learn distinct features to differentiate shapes across shape categories.

Alternatively, combining the mean content code $z_{c\mu}$ of the shape x_μ with distinct style codes of shapes from 3 car classes should generate shapes that can be classified into their respective car classes, since style learns localized traits distinct to each car class. In Fig. 5, by visual inspection of shapes $(x_{\mu 1}, x_{\mu 2}$ and $x_{\mu 3})$, we observed each of these shape represents shapes from 3 different classes. The generated augmented shape $x_{\mu 1}$ represents an SUV car design like x_1. Likewise, the generated shape $x_{\mu 2}$ resembles a sedan shape (x_2) and $x_{\mu 3}$ resembles a convertible shape (x_3).

Thus, we concluded that the style code learns localized distinctive shape parts that differentiate a shape from others with the same content and can describe shapes in more detail. While the content code learns a global representation of the 3D shape without specific details. In the next set of experiments, we utilized Split-AE as a shape generative model for new augmented shape generation by style transfer.

V. Style transfer between paired shapes

In this section, we utilized Split-AE to generate new augmented shapes by providing different 3D shapes for style reference (Section III-D).

a) Augmented shape generation and validation: We performed 3D shape-to-shape style transfer between shapes within a domain. We hypothesized that the augmented shape should possess the distinct shape part of the target reference shape class from where it adapted the style code. In Fig. 6 we show examples of augmented shapes generated by style transfer between source and target shapes in the car and airplane domains.

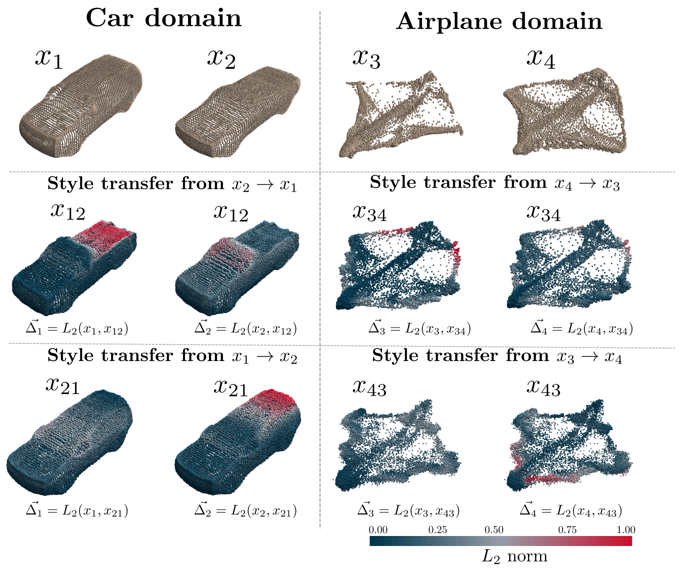

Car domain
Airplane domain
x_1
x_2
x_3
x_4
Style transfer from $x_2 \to x_1$
Style transfer from $x_4 \to x_3$
x_{12}
x_{12}
x_{34}
x_{34}
$\vec{\Delta}_1 = L_2(x_1, x_{12})$
$\vec{\Delta}_2 = L_2(x_2, x_{12})$
$\vec{\Delta}_3 = L_2(x_3, x_{34})$
$\vec{\Delta}_4 = L_2(x_4, x_{34})$
Style transfer from $x_1 \to x_2$
Style transfer from $x_3 \to x_4$
x_{21}
x_{21}
x_{43}
x_{43}
$\vec{\Delta}_1 = L_2(x_1, x_{21})$
$\vec{\Delta}_2 = L_2(x_2, x_{21})$
$\vec{\Delta}_3 = L_2(x_3, x_{43})$
$\vec{\Delta}_4 = L_2(x_4, x_{43})$
0.00
0.25
0.50
0.75
1.00
L_2 norm

Fig. 6. In car and airplane domains, we illustrate style transfer results for pairs $x_1 \leftrightarrow x_2$ and $x_3 \leftrightarrow x_4$, respectively. Each point of the generated point cloud shapes $(x_{12}, x_{21}, x_{34}$ and $x_{43})$ is color coded based on the Euclidean (L_2) distance vector $(\vec{\Delta}_i, i = 1, .., 4)$ written below each generated shape.

In the car domain (Fig. 6), we selected an SUV (x_1) and a convertible (x_2) as a source-target pair. The augmented shapes x_{12} and x_{21} are generated by style transfer from $x_2 \to x_1$ and $x_1 \to x_2$, respectively. We observed that the roof structures of the two generated shapes $(x_{12}$ and $x_{21})$ change according to the reference shape class from where it adapted the style codes, i.e., for x_{12} the roof structure is like x_2 and for x_{21} like x_1.

Further, for qualitative analysis of the structural similarity between the generated shape x_{12} to x_1 and x_2, we calculated two normalized Euclidean (L_2) distance vectors $(\vec{\Delta}_1$ and $\vec{\Delta}_2)$ between each point in the generated point cloud shape x_{12} with x_1 and x_2. In Fig. 6, we visualized two samples of the generated shapes x_{12} and projected $\vec{\Delta}_1$ and $\vec{\Delta}_2$ as color maps onto the two 3D point cloud representations of x_{12}, respectively. We observed that x_{12} color coded with $\vec{\Delta}_1$ shows the largest differences (high normalized L_2 values) in the roof region with respect to x_1 and higher similarity (low normalized L_2 values) in the frontal and lower region with x_1 from where it adapted the content code. Analogously, for the generated shape x_{12} color coded with $\vec{\Delta}_2$, we observed that the roof structure of x_{12} is similar (low normalized L_2 values) to the target shape x_2. Alternatively, the augmented shape x_{21} generated by style transfer from $x_1 \to x_2$ shows roof structure like x_1 and the bottom region of the car shape similar to x_2. These results signify that the style transfer between source-target pairs generate new shapes with modified shape parts.

Likewise, in the airplane domain (Fig. 6), style transfers between an airliner (x_3) and a propeller (x_4) shape pair generated shapes x_{34} and x_{43}. Different from the style transfer in the car domain, we observed that each of the augmented shape $(x_{34}$ or $x_{43})$ in the airplane domain changes the wing structures according to the reference shape from where it adapted the style code, i.e., the wings of the generated shape x_{34} changes according to the propeller shape x_4. Thus, Split-AE correctly identifies distinctive styles in different domains.

Additionally, to confirm our observations in Fig. 6 that each style transfer generates a novel augmented shape which has mixed shape parts from both source-target shapes, we analyzed the performance of style transfers for additional augmented shapes from each domain. We generated 500 new augmented car shapes and 500 airplane shapes by randomly selecting 500 source-target pairs from the car and airplane shapes in the data set, respectively. For qualitative visual inspection, we show 25 out of 500 augmented car shapes in Fig. 7, where the target shape alters in each column and the source shape changes row-wise. Thus, the horizontal axis indicates traversing of style and the vertical axis shows the change of content, i.e., shapes in each row have fixed content with changing style codes. To verify the distinct shape parts similarity of the augmented shapes with the target shape class, we used the trained multi-class classifier to predict the class label of each augmented shape from its latent representation z and color-coded each shape based on the predicted class label (Section III-C).

In the first row of Fig. 7, the augmented shapes $(x_{1j}, j = 1, .., 5)$ are generated by combining the content code of an SUV shape x_1 with style codes of shapes from different car classes. We observed significant differences in the roof structure of the augmented shapes $(x_{1j}, j = 1, .., 5)$, however,

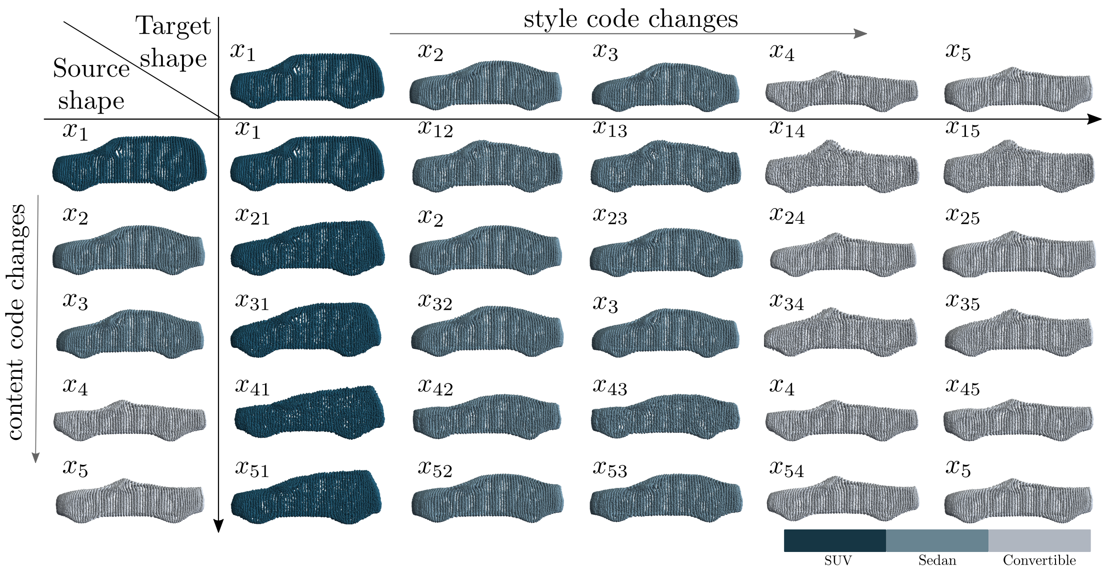

Target shape
Source shape
style code changes
content code changes
x_1 x_2 x_3 x_4 x_5
x_1 x_{12} x_{13} x_{14} x_{15}
x_{21} x_2 x_{23} x_{24} x_{25}
x_{31} x_{32} x_3 x_{34} x_{35}
x_{41} x_{42} x_{43} x_4 x_{45}
x_{51} x_{52} x_{53} x_{54} x_5
SUV
Sedan
Convertible

Fig. 7. Visualization of reconstructions of 25 augmented car shapes generated by style transfer from target to source shapes. The content code is fixed in each row while the style code varies. Similarly, the style code is fixed in each column while the content code varies. Each of the generated shapes is color-coded based on the predicted class label.

preserving the similarity in the lower-body and frontal design with the source shape x_1. Likewise, the augmented shapes ($x_{j1}, j = 1..5$) in the first column hold distinct shape parts (roof structure) of the SUV class, but the frontal and lower-body design changes in each row according to each row-based source shape. Also, for each augmented shape in Fig. 6, the prediction of the class label using the trained multi-class classifier shows that the style of each target shape dominated the class allocation of each augmented shape. Thus, column-wise, each generated shape has a similar prediction label (same color) that matches the target shape label in that column.

In addition, we predicted the class labels of 500 shapes using the trained multi-class classifier and measured the accuracy, i.e., the predicted labels of the augmented shapes match the class labels of their respective target shape class. Style transfer in both domains shows a high accuracy of 0.91 for 500 generated car shapes and 0.82 for 500 airplane shapes, indicating the success of our style transfer approach. Therefore, Split-AE provides the flexibility to combine features between two shapes with its disentangled latent spaces. Therefore, we concluded, Split-AE excels in part-based modification by replacing distinct shape parts, while accurately maintaining the underlying structure for generating novel realistic 3D shapes. This helps to generate novel shapes which are modified versions of the existing designs.

b) Latent space exploration: As a second experiment to evaluate the goodness of learned disentangled variables, we performed latent space-based interpolation in the content and style space in Fig. 8, where we performed uniform distance interpolations of the content and style codes from a source shape (SUV) to a target shape (convertible). Since our network was trained to generate an organized point cloud like that of the input shapes, we measured the similarity of each interpolated shape with the target shape to illustrate the change in each 3D shape by interpolation of the latent variables.

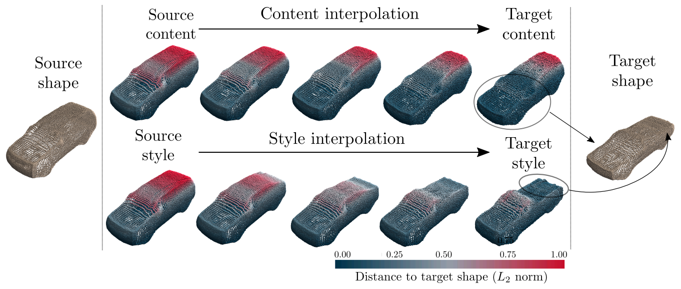


Fig. 8. Interpolation in the content and style space learned by Split-AE from a source to a target shape. In the first row, we generate shapes by uniformly interpolating between source content and target content, keeping the source style. In the second row, we generate shapes by uniformly interpolating between source style and target style, keeping the source content constant.

In Fig. 8, we observed the shapes produced with the interpolated contents in the content space keeping the source-style code fixed, show the frontal and lower region of the exterior design of the car shapes gradually becoming similar to the target shape (low normalized L_2 distances). Alternatively, shapes produced by interpolating style from the source to the target shape keeping the content of the source shape fixed, show a gradual change in the roof designs to make the roof design of the interpolated shapes like the convertible (low normalized L_2 distances). This shows the credibility of two separate content and style spaces for design exploration.

VI. Conclusion

Motivated by the challenges of learning disentangled latent representations using a generative model for representing different interpretable visual features of 3D shapes, we propose in this paper a novel method involving minimum supervision to disentangle 3D representations into content and style features. We demonstrate with quantitative and qualitative measures that our method is effective in disentangling content and style space that represent various aspects of 3D shapes. We also present examples of 3D shape-to-shape feature transfer for shapes across multiple classes within a domain. In contrast to previous work, our approach learns content and style codes for shapes across multiple classes and domains, which allows generating augmented shapes using style transfer in different domains. These also help to achieve our main goal to have machine learning assist in design transformation tasks for the designers in automotive design development.

The main technical limitation of our work was that we considered style features resembling localized regions of 3D shapes. However, our analysis does not consider more global style features, which will help to understand the style of cars from different brands. Moreover, we regard our work as a first step towards a generic model for features disentanglement of shapes across domains. Also, in this research, we utilized car and airplane shapes for our evaluation and therefore did not use the model for style transfer across domains. Thus, as future work, we aim at identifying style also as global features common to car shapes from a brand and investigate 3D style transfer between cross-domain shapes.

References

[1] M. Segu, M. Grinvald, R. Siegwart, and F. Tombari, “3DSNet: Unsupervised Shape-to-Shape 3D Style Transfer,” *arXiv preprint arXiv:2011.13388*, 2020. [Online]. Available: http://arxiv.org/abs/2011.13388

[2] R. Hanocka, N. Fish, Z. Wang, R. Giryes, S. Fleishman, and D. Cohen-Or, “AligNet: Partial-shape agnostic alignment via unsupervised learning,” *ACM Transactions on Graphics*, vol. 38, no. 1, 2018.

[3] T. Rios, B. Sendhoff, S. Menzel, T. Bäck, and B. Van Stein, “On the Efficiency of a Point Cloud Autoencoder as a Geometric Representation for Shape Optimization,” in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 791–798.

[4] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, “Learning representations and generative models for 3d point clouds,” in *35th International Conference on Machine Learning (ICML)*, vol. 80, 2018, pp. 40–49.

[5] S. Saha, S. Menzel, L. L. Minku, X. Yao, B. Sendhoff, and P. Wollstadt, “Quantifying The Generative Capabilities Of Variational Autoencoders For 3D Car Point Clouds,” in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 1469–1477.

[6] T. Friedrich, N. Aulig, and S. Menzel, “On the Potential and Challenges of Neural Style Transfer for Three-Dimensional Shape Data,” in *EngOpt 2018 Proceedings of the 6th International Conference on Engineering Optimization*. Springer International Publishing, 2019, pp. 581–592.

[7] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017, pp. 652–660.

[8] M. Y. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 701–709, 2017.

[9] R. Kuga, A. Kanezaki, M. Samejima, Y. Sugano, and Y. Matsushita, “Multi-task Learning Using Multi-modal Encoder-Decoder Networks with Shared Skip Connections,” *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, vol. 2018-Janua, pp. 403–411, 2017.

[10] F. Locatello, S. Bauer, M. Lucie, G. Rätsch, S. Gelly, B. Schölkopf, and O. Bachem, “Challenging common assumptions in the unsupervised learning of disentangled representations,” *36th International Conference on Machine Learning, ICML 2019*, vol. June, pp. 7247–7283, 2019.

[11] Q. Zhu and R. Zhang, “A Classification Supervised Auto-Encoder Based on Predefined Evenly-Distributed Class Centroids,” pp. 1–16, 2019. [Online]. Available: http://arxiv.org/abs/1902.00220

[12] R. Das and S. Chaudhuri, “On the Separability of Classes with the Cross-Entropy Loss Function,” 2019. [Online]. Available: http://arxiv.org/abs/1909.06930

[13] K. Yin, Z. Chen, H. Huang, D. Cohen-Or, and H. Zhang, “LOGAN: Unpaired shape transform in latent overcomplete space,” *ACM Transactions on Graphics*, vol. 38, no. 6, 2019.

[14] T. Rios, B. van Stein, T. Bäck, B. Sendhoff, and S. Menzel, “Multi-Task Shape Optimization Using a 3D Point Cloud Autoencoder as Unified Representation,” *IEEE Transactions on Evolutionary Computation*, pp. 1–12, 2021.

[15] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, “ShapeNet: An Information-Rich 3D Model Repository,” Stanford University - Princeton University - Toyota Technological Institute at Chicago, Tech. Rep., 2015. [Online]. Available: http://arxiv.org/abs/1512.03012

[16] S. Saha, T. Rios, L. L. Minku, B. Stein, P. Wollstadt, X. Yao, T. Bäck, B. Sendhoff, and S. Menzel, “Exploiting Generative Models for Performance Predictions of 3D Car Designs,” in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2021, pp. 1–9.

[17] T. Rios, B. van Stein, S. Menzel, T. Bäck, B. Sendhoff, and P. Wollstadt, “Feature Visualization for 3D Point Cloud Autoencoders,” in *Proceedings of the International Joint Conference on Neural Networks*, 2020, pp. 1–9.

[18] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations (ICLR)*, 2015, pp. 1–15.

[19] T. Rios, P. Wollstadt, B. V. Stein, T. Bäck, Z. Xu, B. Sendhoff, and S. Menzel, “Scalability of Learning Tasks on 3D CAE Models Using Point Cloud Autoencoders,” in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 1367–1374.