Quantifying The Generative Capabilities Of Variational Autoencoders For 3D Car Point Clouds

Sneha Saha*, Stefan Menzel*, Leandro L. Minku†, Xin Yao†‡,
Bernhard Sendhoff*, and Patricia Wollstadt*
*Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany
†CERCIA, School of Computer Science, University of Birmingham, Birmingham, UK
‡Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China
Email: {sneha.saha, stefan.menzel, bernhard.sendhoff, patricia.wollstadt}@honda-ri.de,
{l.l.minku, x.yao}@cs.bham.ac.uk

***Abstract*—During each cycle of automotive development, large amounts of geometric data are generated as results of design studies and simulation tasks. Discovering hidden knowledge from this data and making it available to the development team strengthens the design process by utilizing historic information when creating novel products. To this end, we propose to use powerful geometric deep learning models that learn low-dimensional representation of the design data in an unsupervised fashion. Trained models allow to efficiently explore the design space, as well as to generate novel designs. One popular class of generative models are variational autoencoders, which have however been rarely applied to geometric data. Hence, we use a variational autoencoder for 3D point clouds (PC-VAE) and explore the model's generative capabilities with a focus on the generation of realistic yet novel 3D shapes. We apply the PC-VAE to point clouds sampled from car shapes from a benchmark data set and employ quantitative measures to show that our PC-VAE generates realistic car shapes, wile returning a richer variety of unseen shapes compared to a baseline autoencoder. Finally, we demonstrate how the PC-VAE can be guided towards generating shapes with desired target properties by optimizing the parameters that maximize the output of a trained classifier for said target properties. We conclude that generative models are a powerful tool that may aid designers in automotive product development.**

***Index Terms*—Representation learning, geometric deep learning, point clouds, generative model, novelty**

I. Introduction

Current engineering design development processes are enriched by a manifold of computer-aided design (CAD) and engineering (CAE) tools for various tasks. Human users modify digital designs and evaluate concepts towards their performance under given environment and scenario specifications. Like in many other fields, in the automotive industry a huge amount of digital data emerges from computational models which are processed for quantifying various vehicle characteristics according to required fidelities of detail. As examples, complete car designs are simulated for aerodynamic drag efficiency, or finite element models of structural components are computed for stiffness or crash safety. Besides technical performance, successful and innovative designs naturally build upon further qualities such as aesthetic appeal, which typically relies on a human in the loop to ideate and realize envisioned conceptual design directions.

Through the recent advances in artificial intelligence (AI), a current major trend is to collect and utilize existing digital design and simulation data for the application of data analytics and machine learning. Both help to increase the knowledge on the problem domain, provide qualified system answers and realize a fast exploration of potential design ideas. The finding based on data analytics and machine learning may be communicated to the designer through an interactive vehicle design tool which would favor a cooperative character with well-thought interactions for proper guidance [1] such that the human designers capabilities are enhanced rather then replaced [2]. State-of-the-art AI methods offer a promising approach to the realization of such a cooperative design system (CDS), where in particular generative models may be used to provide guidance or potential alternatives giving the designer the freedom to rethink, discuss and adapt promising product directions.

Essential to such an automotive CDS is a model that provides a representation of 3D geometries that allows to efficiently explore the design space, and is able to generate novel and realistic car shapes. In contrast to a manual design of the representation which would be potentially biased or limited by human heuristics, geometric deep learning offers generative methods such as (variational) autoencoders (V)AE [3], [4] that learn low-dimensional representations of existing 3D shape data in an unsupervised fashion. In particular in the engineering domain, unsupervised approaches are favorable since labeling of data is often prohibitively expensive. Applications of unsupervised models, such as AEs and VAEs, to geometric data are sparse (e.g., [5], [6]), and applications targeted specifically at the engineering application domain are lacking. In the present paper we therefore use a point cloud based variational autoencoder (PC-VAE) that builds on the point cloud autoencoder originally proposed in [6], and extended in [7] and [8], which has been successfully evaluated for engineering applications [7]. We train the proposed model on digital car shapes taken from ShapeNetCore [9], a benchmark

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 766186 (ECOLE).

978-1-7281-2547-3/20/$31.00 ©2020 IEEE

3D shape database for computer graphics research. We further evaluate the generative capability of our PC-VAE by evaluating two central aspects: (1) the realism of the sampled shapes and (2) the capability of the model to generate diverse, novel shapes. In a final step, we demonstrate how the generative process can be guided towards the generation of a specific car type by combining optimization and a multilayer perceptron (MLP) classifier trained on the learned latent space. The last step may be utilized to guide the generative process towards specific objectives, such as the generation of a shape with a desired appeal but also other properties of the generated model.

The remainder of this paper is organized as follows: we present a literature review on generative models from the field of geometric deep learning in Section II. In particular, we focus on VAEs for 3D shape analysis as well as on methods to explore the latent space of such generative models. We then introduce in Section III the architecture of our proposed PC-VAE model and the training process on the car class of ShapeNetCore [9]. We also detail the quantitative measures to evaluate the generative capability of the model for the generation of realistic and diverse shapes. In Section IV, we present the experiments and results of training the PC-VAE on the ShapeNetCore car class data. In particular, we evaluate the model's generative capabilities in comparison to a baseline autoencoder model using quantitative measures, and we describe the training and optimization process used to guide the generative process through the training of an MLP classifier on the learned latent space. Finally, we discuss our results in the light of using a PC-VAE to generate car designs for design suggestions in the context of a CDS in Section V.

II. Prior Art

A. Generative models for shape analysis

Recently, several generative models have been proposed in the 2D domain for creative design inspiration by sketch drawings [10] or for fashion generation [11]. However, realizing similar systems for 3D shape data, especially engineering data, requires different approaches due to the higher inherent complexity of the data, limited number of available source data and various base representations of CAE/D models [12]. By base representation we refer to different options how 3D geometries are potentially represented such that they can be processed in a geometric deep learning pipeline. Here, common representations include voxels, point clouds or polygon meshes, which mainly vary in the level of detail they are able to represent and in memory demand [13]. For our application, we focus on 3D shapes sampled as surface point clouds due to their high flexibility and memory-efficiency, which allows to represent finer geometric detail and scale models to bigger input sizes [8]. Potential candidates for 3D representation learning are autoencoders (AEs) [14], variational autoencoders (VAEs) [3], and generative adversarial networks (GANs) [15].

Popular AE architectures that process point-cloud representations of 3D data are PointNet [16] and PointNet++ [17]. Many networks have adopted the PointNet architecture because of its simplicity and strong representation capability. The architecture overcomes the common difficulty of learning data from point clouds by making models invariant to permutations in the input through pre-processing and modification in the conventional multi-layer convolution network. Achlioptas et al. [6] proposed an AE architecture building on PointNet, where the encoder is similar to PointNet and the decoder comprises fully connected layers. The generative capabilities of the regular AE is limited, therefore, the authors train an additional Gaussian mixture model (GMM) on the latent space of the AE (AE+GMM), which is a computationally expensive further step. In addition, Rios et al. [18] show that transferring features using an AE between two different geometric shapes results in a fuzzy point cloud, which may be a consequence of the irregularity in the latent space of the AE in absence of any regularization technique.

GANs, as potential alternative, are challenging in the training step because they suffer from non-convergence, mode collapse, and vanishing gradients [19]. Consequently, researchers use various forms of regularization to improve the training convergence of a GAN [20], [21], but still tuning the hyperparameters remains challenging.

Recently, variational architectures have been successfully introduced as generative models for point clouds (e.g., [5], [22]), and have been extended using continuous normalizing flow [23]. We here use such a variational architecture, which is an extension of the autoencoder model presented in [6]–[8], and which has been evaluated specifically for the application to engineering problems. We extended the model to a VAE architecture to obtain a model with improved generative capabilities, whose training is less complex and relatively time efficient. Prior research on VAEs for geometric data focused on various tasks, such as 3D shape reconstruction from 2D images [24], 3D shape segmentation [25], part generation and shape generation from segmented objects [26], as well as shape-inference from images and classification [22]. The research closest to our goal of generating novel and diverse shapes is CompoNet [27], a generative neural network for 2D or 3D shapes based on a part-based prior, which relies on a VAE for 2D shape synthesis and the above mentioned AE+GMM for 3D objects. Zamorski et al. [5] proposed an end-to-end solution to generating 3D shapes with an adversarial autoencoder (AAE) for 3D point clouds using a binary representation in the latent space. The AAE differs from a VAE in the loss computed on the latent space, where the AAE uses the adversarial loss similar to a GAN while the VAE uses the Kullback-Leibler (KL) divergence to enforce regularization of the latent space. Zamorski et al. [5] trained the adversarial model with the Earth-Mover distance (EMD) and analyzed the model with various objectives, such as 3D point cloud clustering and object retrieval.

Opposed to prior research on the generative capabilities of PC-VAEs, we here focus on the generative capabilities of generative geometric models, in particular with respect to the *realism* and *diversity* of generated shapes, such as to evaluate the model's potential for applications in engineering design tasks. To this end, we propose a VAE architecture and

evaluate both aspects in a quantitative fashion. Furthermore, we demonstrate how shapes may be generated in a more targeted fashion by optimizing the input of a classifier trained on the learned latent representations.

B. Evaluation of 3D shape generation

When generating novel shapes from a trained model, we aim at generating both realistic and novel shapes. Both aspects are central to the intended application domain of engineering design. In other words, a model should generate plausible shapes, i.e., shapes that in some aspects closely resemble the data set, but also novel or diverse shapes that are sufficiently far away from the training data.

To generate shapes from a trained model, research on generative models typically uses linear interpolation between two shapes as a way of demonstrating that a generative model has not simply memorized the training examples [28]. Also, random sampling is frequently employed to evaluate the explorative capability of the generative models. However, shapes generated from sampling the learned latent space of autoencoders are not always realistic. To quantify the "realism" of a generated shape, Berthelot et al. [29] proposed a mean distance (MD) metric in 2D that compares the minimum cosine distance of the interpolated data-points with the original data-points by finding the nearest neighbor of each interpolation step in the data set. Similar to this approach, Achlioptas et al. [6] proposed a minimum matching distance (MMD) metric to measure the closeness of two point cloud sets using the Chamfer distance (CD) [24], where closeness between sets is assumed to indicate a higher realism.

To limit the generation of samples to realistic shapes it has been proposed to limit the boundaries of the latent space by the sample encoding of the data set shapes [26]. This approach deals with the fact that generative models can produce latent spaces that are not tightly packed and ensures that one samples from the latent manifold only. Even though this approach improves the model reconstruction quality, it may become overly restrictive in the context of a design exploration application such that it prevents the model from suggesting novel samples.

To not only evaluate the realism but also the novelty or *diversity* of generated shapes, Schor et al. [27] rely on a classifier to test the diversity of samples generated from an AE+GMM model. The generative diversity is calculated as the percentage of the generated samples classified as belonging to an unseen test set.

In sum, quantitative approaches for evaluating both the realism and the diversity of 3D shapes generated from geometric deep learning models have been proposed. We here combine both approaches to assess whether the proposed PC-VAE is able to generate shapes that are both realistic and novel. This ability of 3D VAEs to generate realistic and novel shapes has been little explored despite the relative simplicity and higher efficiency of VAEs in comparison to other (3D) generative models. We additionally show that the generation of specific shapes can be enforced by running an optimization on the inputs to the classifier. In the next section, we describe the proposed PC-VAE architecture and optimization procedure in detail before we go on to explore the model's explorative and generative capabilities.

III. Methodology

In this section, we first introduce the model architecture of our proposed PC-VAE (Section III.A). Next, we provide details on the quantitative measures which are applied to evaluate our model's generative ability (Section III.B). Finally, we propose an approach to guide the generation of more diverse shapes using optimization and machine learning models trained on the learned latent space (Section III.C).

A. Variational autoencoder

Background: The VAE [3], [4] is a generative model that, opposed to a standard AE, aims at learning "disentangled, semantically meaningful, statistically independent and causal factors of variation in data" [30]. The VAE may be seen as a regularized version of the AE that forces the learned latent space towards following an a-priori specified distribution, $p(z)$.

The VAE has the same basic architecture as an AE, with an encoder, $q_\phi(z|x)$, that is parameterized by ϕ and learns to map the input x to a variational distribution. The decoder, $p_\theta(x|z)$, parameterized by θ aims at reconstructing the input x from the latent vector z, sampled from the learned distribution.

The VAE maximizes the evidence lower bound (ELBO) [3] $\mathcal{L}(x;\theta,\phi)$ w.r.t. parameters θ and ϕ,

$$\begin{aligned}\mathcal{L}(x;\theta,\phi) &= \mathbb{E}_{q_\phi(x|z)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)) \\ &\leq \log p(x),\end{aligned} \tag{1}$$

which is a valid lower bound on the log-likelihood of the data. Here, $\mathbb{E}_{q_\phi(x|z)}[\log p_\theta(x|z)]$ is the negative reconstruction loss which enforces the encoder to learn a meaningful latent vector z, such that the decoder can reconstruct the input x from z. $D_{KL}(q_\phi(z|x)||p(z))$ is the Kullback-Leibler (KL) regularization loss which minimizes the KL-divergence between the approximate posterior $q_\phi(z|x)$ and the prior $p(z) \sim (0, I)$.

Network Architecture (PC-VAE): The PC-VAE used in our experiment is implemented based on an architecture presented in [7] and [8], which extends a proposal in [6], [16] (Figure 1) with minor modifications in the activation functions to maintain the limits of the normalization used for the input geometries. The encoder part of the PC-VAE follows [16], who propose to use 1D-convolutional layers together with permutation-invariant global operators (e.g., max-pooling at a deeper layer of the network) in order to make the architecture invariant against permutations in the input point clouds. The encoder-decoder structure used here is similar to the architecture proposed in [6], only the last layer of the decoder is replaced with sigmoid activation functions [7], since the coordinates of all points are normalized to the range $[0.1, 0.9]$. In our encoder, we use five 1D-convolutional layers,

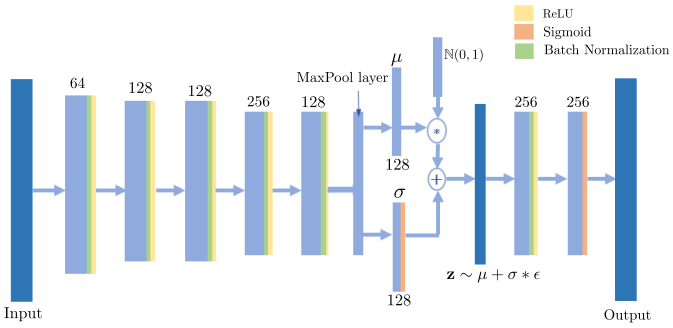

Fig. 1: PC-VAE architecture. Input and output are represented by a 3D point cloud composed of 2048 points.

each followed by a ReLU [31] and a batch normalization layer [32]. The decoder consists of three fully connected layers.

To extend the autoencoder architecture in [6]–[8] to a variational autoencoder, the output of the last convolution layer in the encoder is passed to a max-pool layer that produces a k-dimensional vector that forms the bottleneck for two separate k-dimensional vectors: a mean vector μ and standard deviation vector σ. The mean vector has no activation function, while the deviation vector uses a sigmoid activation function. A vector sampled from the latent distribution is fed into the decoder network for reconstruction of a point cloud and is represented as,

$$z \backsim \mu + \sigma * \epsilon \tag{2}$$

where $\epsilon \backsim N(0, 1)$.

The PC-VAE optimizes the loss function between the input point cloud, S_1, and the reconstructed point cloud, S_2, as

$$\mathcal{L}_{VAE}(S_1, S_2) = \alpha\, d_{CD} + \beta\, \mathcal{D}_{KL}\left(q_\phi\left(z|S_1\right)||p\left(z\right)\right), \tag{3}$$

where the first term on the right-hand side denotes the reconstruction loss, measured by the Chamfer distance (CD) [24],

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2, \tag{4}$$

and the second term is the KL-divergence that quantifies the distance between the learned latent representation and the prior. Both terms of the loss function differ by several orders of magnitude. To bridge this gap, we introduce two parameters α and β to scale the reconstruction loss and KL-divergence, respectively. For training and exploring the generative capabilities of PC-VAE, we split the complete data set $\mathbb{S}$, into a seen training set, $\mathbb{D}_{train}$, and an unseen test set, $\mathbb{D}_{test}$.

B. Evaluating the generative capabilities of the PC-VAE

From the trained PC-VAE we generate new shapes, $\mathbb{G}$, either by interpolating in the latent space or random sampling of the learned distribution in the latent space. We then evaluate both the realism and the diversity of $\mathbb{G}$.

We evaluated the realism of generated shapes qualitatively through visual inspection and quantitatively employing the method introduced in [29], where realism of generated shapes is measured as the mean distance to the data set $\mathbb{S}$. Furthermore, we evaluated the novelty of generated shapes by quantifying their diversity with respect to the training set $\mathbb{D}_{train}$, following the approach in [27]. Hence, we pose two objectives: create shapes close enough to the data set to be considered realistic, but far enough from the training set to be considered novel.

Mean distance (MD) and Minimum matching distance (MMD-CD): To quantify the realism of shapes $\mathbb{G}$ generated through interpolation, we used the mean distance (MD) [29] and minimum matching distance (MMD-CD) [6]. Both measures are similar and quantify realism as the distance between shapes in $\mathbb{G}$ and the data set $\mathbb{S}$. MMD-CD is calculated as the average minimum CD between the shapes in $\mathbb{G}$ and their nearest neighbor in $\mathbb{S}$,

$$\text{MMD-CD}\left(\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_N\}\right) = \frac{1}{N}\sum_{i=1}^{N} \hat{q}_i, \tag{5}$$

where $\hat{x}_i$ is the point cloud reconstructed from the i^{th} interpolation step and

$$\hat{q}_i = \min_{s \in S}\left\{d_{CD}\left(\hat{x}_i, s\right)\right\} \tag{6}$$

is the minimum CD between the reconstructed point cloud $\hat{x}_i$ and any shape s in $\mathbb{S}$. Here, a model showing a lower MMD-CD when performing the interpolations is considered to generate more realistic shapes.

Generative Diversity: To quantify the novelty of generated shapes, $\mathbb{G}$, we used the diversity measure presented in [27]. Similar to the approach outlined there, we trained a binary classifier on the latent representation of the whole data set $\mathbb{S}$, where the information whether a sample belonged to either the training set, $\mathbb{D}_{train}$, or test set, $\mathbb{D}_{test}$, was used as the label. The diversity is then calculated as the percentage of *generated samples*, $\mathbb{G}$, that are classified as belonging to the test set. In other words, the classifier quantifies the percentage of samples generated by the model that are closer to the unseen test set than to the training set. The higher the number of generated samples classified into the unseen test set, the higher the diversity of the model. We here used a multilayer perceptron (MLP) for classification.

Coverage: Another measure of the novelty of $\mathbb{G}$ is the coverage that quantifies the fraction of point clouds in $\mathbb{G}$ that can be matched to the test set, $\mathbb{D}_{test}$. For each point-cloud in $\mathbb{G}$, we find its nearest neighbor in $\mathbb{D}_{test}$ computed with the CD. We calculate the coverage as the percentage of shapes in $\mathbb{D}_{test}$ that are nearest neighbors to $\mathbb{G}$. If this fraction is small, the generated shapes resemble only a small fraction of $\mathbb{D}_{test}$, if the coverage is high, the generated samples represent a large fraction of $\mathbb{D}_{test}$.

C. Enforcing the generation of novel shapes

To use a generative model in the engineering design process, the model has to be able to generate realistic and novel shapes. In the present work, we therefore explored methods to enforce the generation of such realistic and novel shapes. To this end, we first trained a classifier to calculate the diversity of the generated samples, $\mathbb{G}$. As described in Section III.B, the classifier is trained on the latent representations of the whole data set, $\mathbb{S}$, with the information whether a sample belongs to the training set, $\mathbb{D}_{train}$, or test set, $\mathbb{D}_{test}$, as labels. After training the classifier we performed an optimization on the input to the classifier that was aimed at generating novel samples that are classified as belonging to $\mathbb{D}_{test}$, i.e., are considered novel or diverse. Reconstructing the latent representations returned by the optimization process leads then to the generation of novel shapes in a targeted fashion.

For the optimization, we here trained an additional Gaussian process model (GP) on the latent space representation and the corresponding output probabilities of the MLP classifier for each shape. We trained an additional model to make use of gradient-based methods that are typically faster. Hence, we used a differentiable model to map from the latent representation to classification results [33].

We trained the GP on top of the classification results to apply the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm to generate shapes that maximize the GP output, i.e., the probability of classifying a latent sample as belonging to $\mathbb{D}_{test}$. BFGS is an iterative method for solving unconstrained non-linear optimization problems efficiently. For experimental verification of the final shapes proposed by the optimization approach, we applied the MLP classifier to the inputs optimized using the described approach.

In sum, we here propose an approach that is a combination of the exploratory ability of the PC-VAE, and a classifier and optimization applied to the learned latent space. While we make use of the explorative and generative capabilities of the PC-VAE, we utilize an additional model and optimization to identify and guide the generation of shapes towards more novel designs.

IV. Experiments and Results

In this section we first present evaluation results for our proposed architecture, where we compare the PC-VAE with existing models from literature (Section IV.A). Then, we present a sequence of experiments that test the model's generative ability with respect to the realism (Section IV.B) and novelty of generated shapes (Section IV.C). Last, we evaluate the proposed optimization approach (Section III.C) to guide the generation of shapes towards more novel shapes (Section IV.D).

A. Model training

Data: For all experiments, we used point clouds sampled from the car class of the ShapeNetCore data set [9]. As input to the models, we sampled 2048 points using random uniform sampling from each shape's surface, resulting in a matrix of size 2048×3 with spatial coordinates (x, y, z) for each point. We performed all experiments on a single NVIDIA RTX 2080 Ti GPU.

Training the baseline autoencoder (AE): As a baseline for all experiments, we trained a point cloud AE as proposed in [6] and [7]. For training, we split the data set into $60\,\%$ training, $20\,\%$ validation and $20\,\%$ test set. We trained the model with 128 latent dimensions using the Adam optimizer [34] and a learning rate of $5e^{-04}$. We set the batch size to 50 and trained for 600 epochs. We performed a grid search to set the learning rate, batch size and epochs based on the least average reconstruction loss in the validation set.

Training the PC-VAE: The proposed PC-VAE model was trained on the same train-test-split as the AE, with 128 latent dimensions, using the ADAM optimizer and a learning rate of $5e^{-03}$. We performed a grid search approach similar to training the AE, which resulted in a batch size of 80 and a training for 600 epochs.

To optimize the hyperparameters, α and β, in the loss function (eq. 3), we also used a grid search and arrived at parameter values of $\alpha = 250$ and $\beta = 0.001$, which resulted in an acceptable trade-off between the reconstruction accuracy and divergence in the latent space.

To validate the implementation and function of our proposed PC-VAE model, we compared its generative performance to that of existing models as reported in the literature, specifically a regular AE [6] and a 3D-Adversarial AE (3dAAE-G) [5]. As no prior work was based on 3D car shapes, we re-trained our model on the chair class of the ShapeNet data set, split into $85\,\%$ training, $5\,\%$ validation, and $10\,\%$ test-split to match the settings to the reference models from literature. We calculated the MMD-CD between the reconstructed point clouds and their corresponding ground truth in the test data set of the chair object class (Table I). The resulting MMD-CD values indicate that our model is comparable to existing models regarding the ability to encode and reconstruct the data set, so that in a next step, we can evaluate the model's generative capabilities with respect to realism and novelty of generated shapes. Note that this evaluation aims at judging the model's ability to *reconstruct inputs* and not to generate novel shapes. Hence, we here aim at a comparable performance only.

TABLE I: Comparison of the reconstruction capability between our PC-VAE and reference models.

Methods	**MMD-CD**
PC-VAE (ours)	.0008
AE [6]	.0011
3dAAE-G [5][1]	.0008

[1] The model was trained using the Earth-Mover distance (EMD) to calculate reconstruction loss.

B. Evaluating the realism of generated shapes

We hypothesized that the PC-VAE due to the applied regularization learns a smoother latent space compared to the

AE model, which should lead to the generation of more realistic car shapes when reconstructing random or interpolated samples from the latent space of the PC-VAE compared to the AE. To test this hypothesis, we compared reconstructions from a 10-step linear interpolation between 50 randomly selected pairs of car geometries from the data set $\mathbb{S}$ using both our PC-VAE and the AE (Figure 2). Each interpolation pair consisted of an initial shape A and a target shape C, as well as intermediate shapes B, which were reconstructed from the interpolation in the latent space.

We first evaluated the generated shapes qualitatively by visual inspection. We show one example (Figure 2), where we observed a major difference in the roof region. Here, the intermediate interpolations performed in the AE latent space lead to a slanted roof as well as dents in the roof region of the car shape. On the other hand, the interpolation with the PC-VAE showed a more gradual change in the roof region and mostly maintained the curvature of the roof, in total leading to more realistic interpolated car shapes.

To also quantitatively assess the difference of the generative capability between AE and PC-VAE, we calculated the closeness (MMD-CD) between the samples generated for each interpolation and the samples in the complete data set $\mathbb{S}$. Figure 3 shows the MMD-CD over all 50 interpolations for both, AE and PC-VAE. Overall, the PC-VAE showed a lower MMD-CD compared to the AE, indicating that the interpolation led to shapes that were more similar to the shapes in data set and hence produced more realistic cars.

Lastly, we tested the difference in MMD-CD for the interpolations of the 50 car pairs for statistical significance using a non-parametric one-sided Wilcoxon signed-rank test. We performed a one-sided test of the null hypothesis that the MMD-CD calculated from the PC-VAE interpolations (MMD_{PC-VAE}) was greater than or equal to the MMD-CD calculated from the AE interpolations (MMD_{AE}). We found that the MMD-CD from the PC-VAE was significantly smaller, $MMD_{PC-VAE} < MMD_{AE}$, for the tested sample ($z = -4.1449$, $p < 0.001 * **$). Hence, the shapes generated from interpolation in the PC-VAE latent space were more realistic car shapes as indicated by a lower distance to the data set $\mathbb{S}$.

C. Evaluating the diversity/novelty of generated shapes

Next, to not only test whether our proposed model PC-VAE generated realistic, but also novel shapes, we evaluated the diversity of generated shapes [27]. We compared the performance of our PC-VAE model against the generative capabilities of the baseline AE, where we trained a Gaussian-Mixture Model (AE+GMM) on the learned latent space to improve generative capabilities, following the approach proposed in [6]. We divided the car class data set into three different combinations of training- and test-splits as proposed in [27] and re-trained the model for each split. In each run, we generated a set $\mathbb{G}$ of 5000 shapes by random sampling from the distributions over the latent space.

Overall generative diversity of the models: For each train-test-split we calculated the diversity of the generated set, $\mathbb{G}$, as described in Section III.B. We classified samples in $\mathbb{G}$ into training, $\mathbb{D}_{train}$, and unseen test set, $\mathbb{D}_{test}$, using an MLP classifier with two hidden layers $[100, 60]$ and a *tanh* activation function for the hidden layers. The MLP architecture was optimized through a grid search. We repeated the experiment ten times. The averaged percentage of generated samples classified as belonging to the test set are shown in Table II. For all train-test-splits, the PC-VAE generated shapes with a higher diversity than the baseline model.

TABLE II: Comparing generative diversity of the PC-VAE and the AE+GMM (baseline). Best generative diversity for each train-test-split shown in bold.

Train-test-split	AE+GMM	PC-VAE Encoder
50/50	40.3 ± 0.80	$\mathbf{58.19 \pm 0.19}$
70/30	21.7 ± 0.5	$\mathbf{26.96 \pm 0.8}$
90/10	4.0 ± 0.3	$\mathbf{6.5 \pm 0.08}$

For an additional, qualitative evaluation of the generative diversity of the models, we randomly selected three generated samples from each model and searched for the three nearest neighbors of each shape in the training set, $\mathbb{D}_{train}$ (Figure 4). Both models generated realistic shapes, however the PC-VAE generated shapes that were more dissimilar to its nearest neighbors in $\mathbb{D}_{train}$ compared to the baseline model.

Ability to generate new types of car shapes: In the next experiment, we tested whether our proposed architecture was able to generate a novel *type* of car. We re-trained the PC-VAE and the baseline AE on the car class after manually excluding all pickup truck designs from the data. The pickup truck shapes present in the data set are a combination of convertibles and coupe-like car designs (see Figure 5A for examples). We used the separated pickup truck shapes as the test set, $\mathbb{D}_{test}$, (630 shapes in total).

We generated 1800 samples (three times the size of $\mathbb{D}_{test}$), $\mathbb{G}$, from the trained PC-VAE and baseline AE+GMM. We then retrained the MLP classifier on the new $\mathbb{D}_{train}$ and $\mathbb{D}_{test}$ to calculate the *diversity* of the generated shapes. We furthermore calculated the *coverage* and MMD-CD. Results are shown as averages over three repetitions in Table III. All three measures showed that the PC-VAE was able to generate more truck-like and hence more novel shapes. Examples of generated shapes that were classified as belonging to $\mathbb{D}_{test}$ i.e., pickup trucks are shown in Figure 5(B,C). Note that the trucks generated by the PC-VAE show a slightly higher quality.

We want to emphasize that this experiment poses a significant challenge for the generative model since we are testing for the generation of a shape that is significantly different from the training set, $\mathbb{D}_{train}$. This difficulty is also reflected in the small number of generated shapes that are classified as truck-like (3% of 1800 generated samples, Table III). Nevertheless, both models surprisingly showed the ability to generate such novel shapes, not seen in the training set, where the PC-VAE generated shapes of higher quality.

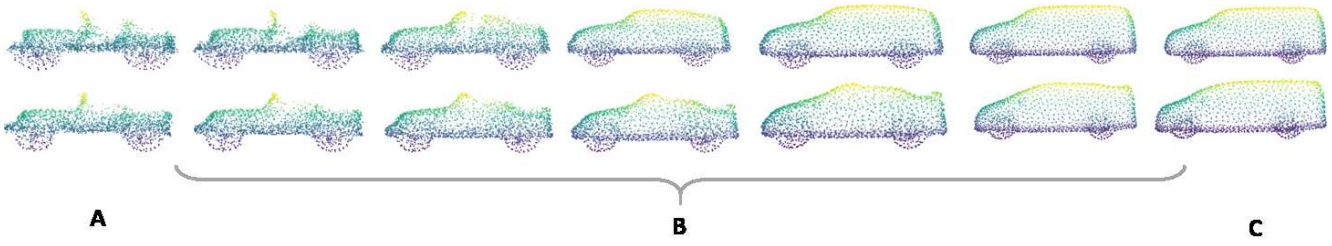

Fig. 2: **A** Reconstruction of the initial shape. **B** Interpolation between shapes A and C in 10 steps—reconstruction of the interpolation at steps 2, 4, 6, 8, and 10. **C** Reconstruction of the target shape. **Top row** Reconstruction of the interpolation using the proposed PC-VAE. **Bottom row** Reconstruction of the interpolation using the AE.

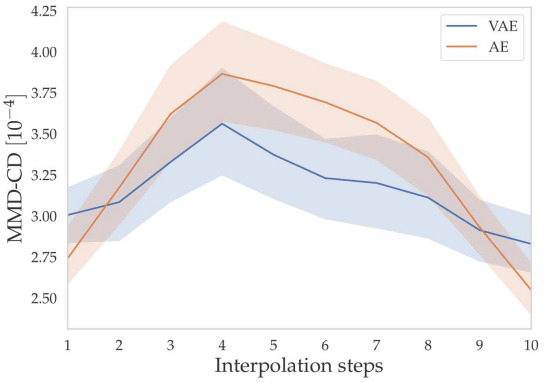

Fig. 3: MMD-CD measure for 50 randomly selected car pairs (each with 10 interpolation steps) using AE and PC-VAE (shaded areas indicate the standard deviation).

TABLE III: Comparing generative diversity, coverage and minimum matching distance (MMD-CD) for a subclass of car shapes on test split (best performance for each measures shown in bold.)

Models	Generative diversity	Coverage	MMD-CD
AE+GMM	0.3 $\pm$ .13	7.3	.00048
PC-VAE Encoder	**3.1 $\pm$ 2.2**	**9.7**	**.00047**

D. Optimization to guide the generation of novel shapes

In the last experiment, we evaluated our proposed approach to guide the model towards generating more novel shapes by performing an optimization of the inputs to the MLP classifier trained to calculate the diversity of the generated set, $\mathbb{G}$. We ran an optimization on the classifier trained in the previous experiment, which optimized the input to the classifier such that it was classified as belonging to $\mathbb{D}_{test}$. To be able to perform a gradient-based optimization, we trained a Gaussian Process (GP) model on all samples in $\mathbb{D}_{train}$ and $\mathbb{D}_{test}$ and their corresponding output probabilities of the MLP classifier. We then optimized the inputs to the classifier using the BFGS-algorithm towards inputs that maximized the GP output, i.e., the probability of belonging to the test set $\mathbb{D}_{test}$.

We verified our approach by selecting an initial sub-set of generated shapes, $\mathbb{G}' \subset \mathbb{G}$, that comprised only samples that were classified as belonging to the training set, $\mathbb{D}_{train}$ (we choose 1744 samples as initial shapes out of 1800 generated samples). We then used the BFGS algorithm to change the latent representations of these shapes such that the output of the MLP was maximized, i.e., shapes were classified as belonging to $\mathbb{D}_{test}$ consisting of the pickup designs. Figure 6 shows examples of initial and optimized shapes. To evaluate whether the optimization resulted in pickup-like shapes in $\mathbb{D}_{test}$, we use the previously trained classifier on the optimized shapes. We found that 1307 out of 1744 initial shapes were classified as pickup truck designs after optimization. Hence, in 75 % of the investigated cases, we could guide the model towards generating a novel design.

V. Conclusion and Outlook

Building interactive systems to support engineers in a design task is a long-standing topic in engineering research [1], [35]. In particular, guiding the design process has been identified as a central aspect of building such a system. We here evaluated the generative capabilities of a variational 3D model for providing such a guidance and found that our proposed PC-VAE was able to generate novel, yet realistic car shapes after training. We furthermore demonstrated a universal approach to guiding the model towards generating novel shapes with certain properties. We here enforced the generation of a new type of car shape, but other objectives, e.g., maximum aerodynamic performance or mechanical properties, are conceivable. We believe that generative, deep learning models provide a promising approach to building design systems that provide assistance in a collaborative fashion such that—instead of replacing the human designer—the designer's abilities are amplified when creating novel shapes.

We quantitatively showed that our model performed comparable to or better than other models proposed in literature while being faster to train, e.g., compared to the generative approach proposed by Achlioptas [6]. We showed that our model was able to generate shapes that were both realistic, yet novel. We demonstrated in two experimental set-ups that our proposed model generated a higher percentage of diverse unseen shapes compared to the baseline AE model. In particular, we showed that the model was able to extrapolate to some extend such

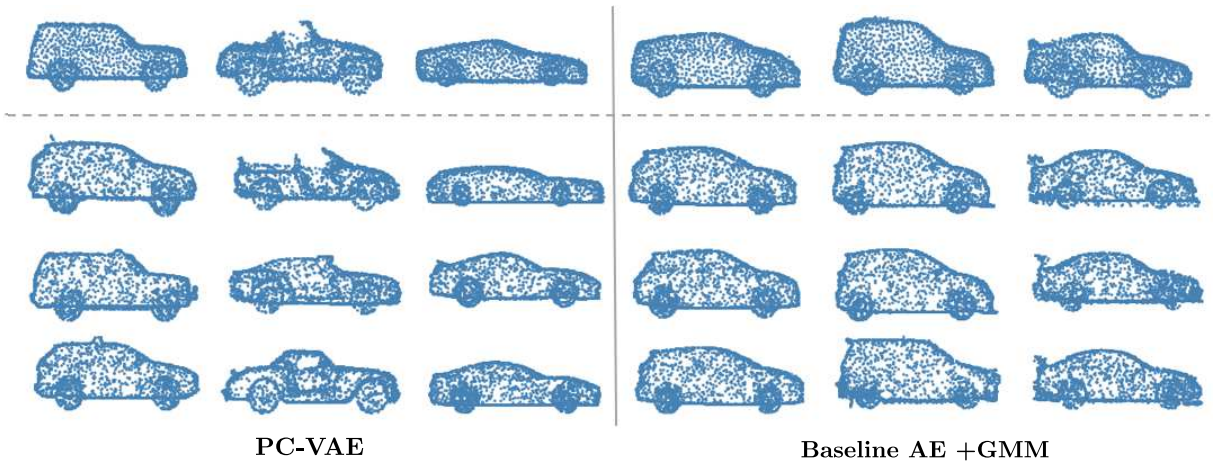

Fig. 4: Qualitative comparison of generative diversity of PC-VAE and baseline AE+GMM. **Row 1:** Three randomly selected shapes from the generated samples. **Row 2-4:** Nearest neighbors of the generated shapes in the training set (measured by CD).

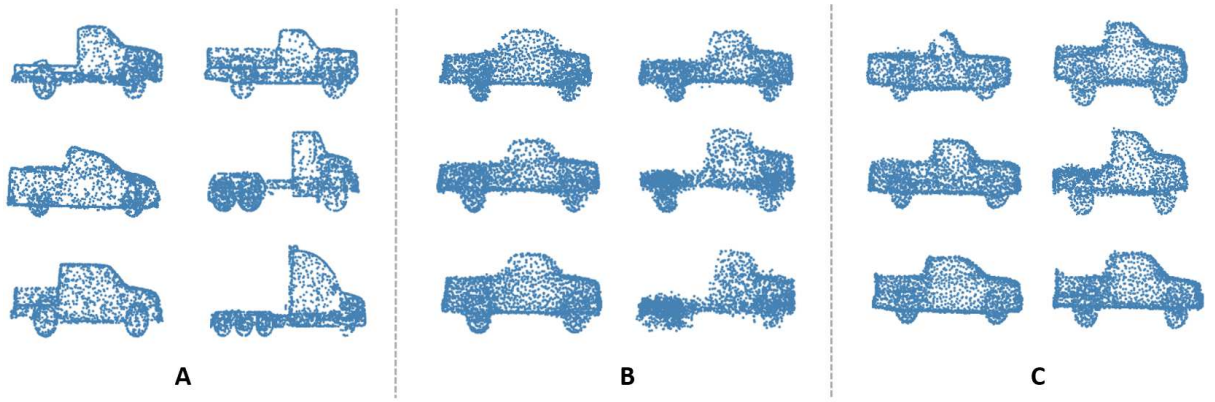

Fig. 5: **A** Randomly sampled geometries from the test set consisting of pickup truck shapes only. **B** Shapes generated by the AE+GMM baseline model that were classified as belonging to the test set. **C** Shapes generated by the PC-VAE classified as belonging to the test set.

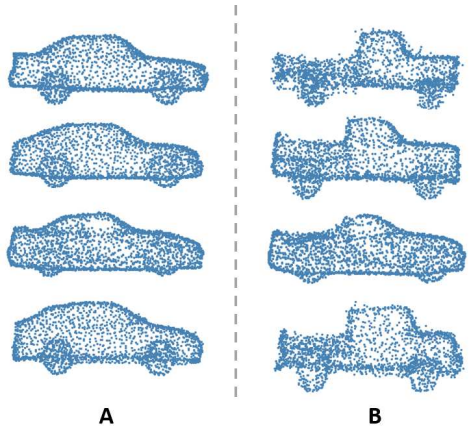

Fig. 6: **A** Initial shapes chosen for optimization in the latent space of the PC-VAE. **B** Optimized shapes.

that it was able to generate shapes that belonged to a novel *type* of car. Hence, the generative ability of the model was not confined to the training set only.

Lastly, we demonstrated how machine learning models trained on the latent space and a subsequent optimization of the input to these models could be used to guide the generative capabilities of the proposed model towards generating more diverse designs. We want to highlight that such an approach is versatile with respect to the optimization's objective. In other words, instead of guiding the model towards generating novel shapes, we may guide it towards generating shapes with a different target property such as aerodynamic performance of the generated car shape.

The present results are a promising step towards using the generative capabilities of 3D-VAEs in cooperative design tools in the automotive domain. In particular, the smooth latent space learned by the PC-VAE allows for an efficient exploration of novel areas of the design space, while providing realistic designs usable in further design steps. Hence, the PC-VAE may be used as part of a CDS that enables an intuitive communication of high-quality design suggestions to

the designer to provide plausible guidance but also as potential alternatives giving the designer the freedom to rethink, discuss and adapt promising product directions. By adding the proposed optimization procedure, the design process may be guided by specific objectives set by the designer.

We conclude that variational models are a feasible and effective approach to making suggestions in a design process and may be used as a component in a CDS. Models are able to generate realistic and novel shapes beyond samples observed in the training set. In the future, we intend to use PC-VAE along with the proposed optimization approach to form a complete cooperative framework to generate realistic and diverse designs.

REFERENCES

[1] D. G. Ullman, "Toward the ideal mechanical engineering design support system," *Research in Engineering Design - Theory, Applications, and Concurrent Engineering*, vol. 13, no. 2, pp. 55–64, 2002.

[2] J. Heer, "Agency plus automation : Designing artificial intelligence into interactive systems," *Proceedings of the National Academy of Sciences*, vol. 116, no. 6, pp. 1844–1850, 2019.

[3] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR*, 2014, pp. 1–14.

[4] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic Backpropagation and Approximate Inference," in *31st International Conference on International Conference on Machine Learning*, 2014, pp. 1278 – 1286.

[5] M. Zamorski, M. Zięba, P. Klukowski, R. Nowak, K. Kurach, W. Stokowiec, and T. Trzciński, "Adversarial autoencoders for compact representations of 3D point clouds," *Computer Vision and Image Understanding*, vol. 193, 2020.

[6] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," *35th International Conference on Machine Learning, ICML*, vol. 1, pp. 67–85, 2018.

[7] T. Rios, B. Sendhoff, S. Menzel, T. Back, 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*, 2019, pp. 791–798.

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

[9] 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

[10] D. Ha and D. Eck, "A Neural Representation of Sketch Drawings," in *6th International Conference on Learning Representations, ICLR*, 2018.

[11] O. Sbai, M. Elhoseiny, A. Bordes, Y. LeCun, and C. Couprie, "DesIGN: Design inspiration from generative networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11131 LNCS, pp. 37–44, 2019.

[12] S. Jayanti, Y. Kalyanaraman, N. Iyer, and K. Ramani, "Developing an engineering shape benchmark for CAD models," *CAD Computer Aided Design*, vol. 38, no. 9, pp. 939–953, 2006.

[13] T. Friedrich, N. Aulig, and S. Menzel, "On the Potential and Challenges of Neural Style Transfer for Three-Dimensional Shape Data," in *Rodrigues H. et al. (eds) EngOpt 2018 Proceedings of the 6th International Conference on Engineering Optimization*, Springer International Publishing, Ed., 2019, pp. 581–592.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," in *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, 2013.

[15] I. J. Goodfellow, J. Pouget-abadie, M. Mirza, B. Xu, and D. Warde-farley, "Generative Adversarial Nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[16] 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.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[18] T. Rios, B. V. Stein, S. Menzel, B. Thomas, B. Sendhoff, and P. Wollstadt, "Feature Visualization for 3D Point Cloud Autoencoders," *International Joint Conference on Neural Networks (IJCNN)*, 2020.

[19] M. Y. Liu and O. Tuzel, "Coupled generative adversarial networks," *Advances in Neural Information Processing Systems*, pp. 469–477, 2016.

[20] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *5th International Conference on Learning Representations, ICLR*, 2017.

[21] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann, "Stabilizing training of generative adversarial networks through regularization," in *Advances in Neural Information Processing Systems*, 2017.

[22] M. Gadelha, R. Wang, and S. Maji, "Multiresolution tree networks for 3D point cloud processing," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11211 LNCS, pp. 105–122, 2018.

[23] G. Yang, X. Huang, Z. Hao, M. Y. Liu, S. Belongie, and B. Hariharan, "Pointflow: 3D point cloud generation with continuous normalizing flows," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[24] H. Fan, H. Su, and L. Guibas, "A point set generation network for 3D object reconstruction from a single image," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2463–2471, 2017.

[25] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3D Shape segmentation with projective convolutional networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

[26] C. Nash and C. K. Williams, "The shape variational autoencoder: A deep generative model of part-segmented 3D objects," *Computer Graphics Forum*, vol. 36, no. 5, pp. 1–12, 2017.

[27] N. Schor, O. Katzir, H. Zhang, and D. Cohen-Or, "CompoNet: Learning to generate the unseen by part synthesis and composition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 8758–8767.

[28] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *4th International Conference on Learning Representations, ICLR*, pp. 1–16, 2016.

[29] D. Berthelot, I. Goodfellow, C. Raffel, and A. Roy, "Understanding and improving interpolation in autoencoders via an adversarial regularizer," *7th International Conference on Learning Representations, ICLR*, pp. 1–20, 2019.

[30] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends in Machine Learning*, vol. 12, no. 4, 2019.

[31] V. Nair and G. E. Hinton, "Rectified linear units improve Restricted Boltzmann machines," in *27th International Conference on Machine Learning*, 2010.

[32] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[33] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, "Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules," *ACS Central Science*, vol. 4, no. 2, pp. 268–276, 2018.

[34] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, 2015.

[35] S. Saha, T. Rios, S. Menzel, B. Sendhoff, T. Bäck, X. Yao, Z. Xu, and P. Wollstadt, "Learning Time-Series Data of Industrial Design Optimization using Recurrent Neural Networks," in *2019 International Conference on Data Mining Workshops (ICDMW)*, nov 2019, pp. 785–792.