Exploiting Generative Models for Performance Predictions of 3D Car Designs

Sneha Saha*, Thiago Rios*, Leandro L. Minku†, Bas vas Stein‡, Patricia Wollstadt*, Xin Yao†§,
Thomas Bäck ‡, Bernhard Sendhoff*, and Stefan Menzel*
*Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany
†CERCIA, School of Computer Science, University of Birmingham, Birmingham, UK
‡Leiden Institute of Advanced Computer Science (LIACS), Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
§Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China
Email: {sneha.saha, thiago.rios, patricia.wollstadt, bernhard.sendhoff, stefan.menzel}@honda-ri.de,
{b.van.stein, t.h.w.baeck}@liacs.leidenuniv.nl, and {l.l.minku, x.yao}@cs.bham.ac.uk

***Abstract*—In automotive digital development, engineers utilize multiple virtual prototyping tools to design and assess the performance of 3D shapes. However, accurate performance simulations are computationally expensive and time-consuming, which may be prohibitive for design optimization tasks. To address this challenge, we envision a 3D design assistance system for design exploration with performance assessment in the automotive domain. Recent advances in deep learning methods for learning geometric data are a promising step towards realizing such systems. Deep learning-based (variational) autoencoder models have been used for learning and compressing 3D data allowing engineers to generate low-dimensional representations of 3D designs. Finding representations in a data-driven fashion results in representations that are agnostic to downstream tasks performed on these representations and are believed to capture relevant design features. In this paper, we evaluate whether such data-driven representations contain relevant information about the input data and whether representations are meaningful in performance prediction tasks for the input data. We use machine learning-based surrogate models to predict the performances of car shapes based on the low-dimensional representation learned by 3D point cloud (variational) autoencoders. Furthermore, we exploit the stochastic nature of the representation learned by variational autoencoders to augment the training data for our surrogate models, since the limited amount of data is usually a challenge for surrogate modeling in engineering. We demonstrate that augmenting training with generated shapes improves prediction accuracy. In sum, we find that geometric deep learning approaches offer powerful tools to support the engineering design process.**

***Index Terms*—representation learning, 3D point clouds, autoencoder, variational autoencoder, regression analysis**

I. Introduction

Computer-aided design (CAD) and engineering (CAE) tools accelerate various tasks in the automotive development process, particularly through virtual prototyping. These virtual models enable engineers to simulate the performance of 3D shapes in multiple domains, *e.g.*, aerodynamic drag and crashworthiness, based on computer simulations, which are more cost-efficient than physical prototyping. However, highly accurate simulations still require excessive computational effort, which limits the performance of engineers in design exploration tasks. To address these challenges in automotive development, we envision a cooperative design system (CDS) that suggests novel 3D car designs along with accurate performance predictions.

State-of-the-art machine learning algorithms offer a promising approach to realize such a CDS. Geometric deep-generative models, like autoencoders (AEs) [1] and variational autoencoders (VAEs) [2] learn low-dimensional latent representations of 3D geometric models in an unsupervised fashion, which allows to efficiently explore the learned design features and generate novel realistic 3D designs [3], [4]. Furthermore, the latent space of autoencoders has also been utilized for many predictive tasks in the chemistry and biology domain [5], allowing the assessment of the chemical compounds before committing to an expensive synthesis. Hence, autoencoders trained on 3D car designs potentially learn enough information in the latent space to be able to predict engineering performance metrics, as targeted in our CDS.

However, generating accurate data-driven prediction models is still challenging. Though benchmark data sets of 3D shapes are available, data sets that are geared towards engineering applications and engineering-specific problems are lacking due to generation costs and confidentiality reasons. Hence, applications of state-of-the-art machine and deep learning techniques are challenging to pursue in the engineering domain. An example of such an application is the prediction of computational fluid dynamics (CFD) from geometric representations, which poses challenges such as highly non-linear data and relationships between performance and geometry. Furthermore, a suitable representation of the geometry has to be found that can be used as input to a predictive model. Here, one approach is to learn these representations in an unsupervised fashion using autoencoders. However, when training autoencoders on geometric data, we neglect information about the geometries'

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

performance, and it is not clear whether such unsupervised learning results in representations suitable for building surrogate models for the prediction of CFD performances.

In the present paper, we propose to exploit the properties of the latent space learned with 3D point cloud (variational) autoencoders to address some of these challenges in the design of 3D car shapes. First, we utilize information-theoretic measures to qualitatively evaluate the learned latent space of (variational) autoencoders with respect to engineering performance metrics, which hints at the quality of the regressions before training a multi-output surrogate model. Second, we evaluate the accuracy of the multi-output surrogate model for predicting performance metrics of 3D car designs from the learned latent representations. Additionally, since the VAE has a regularized latent space, we train the network on a data set of benchmark car shapes and utilize the learned latent distributions to generate additional realistic car shapes to augment the surrogate model's training data. Our main contributions are to build an effective surrogate model for mapping the latent representations to performance metrics and to quantify the advantage of VAEs for shape generations to address data-limitation in regression tasks.

The remainder of the paper is organized as follows: In Section II, we review deep-generative models for 3D shapes and surrogate models for predicting performance metrics of 3D designs. Based on the review, we describe our approach in Section III and provide details on the architecture of point-cloud (variational) autoencoders, surrogate models, and the data-augmentation process. In Section IV, we present the experimental settings and information-theoretic analysis of the (variational) autoencoders latent space for regression tasks. In Section V, we define our regression task and discuss the performance of the surrogate model with and without data-augmentation with respect to the performance predictions of 3D designs. Finally, in Section VI, we present a summary and conclusion of our paper.

II. Related Work

A. (Variational) Autoencoders for Unsupervised Learning of 3D Shape Representations

Engineers utilize different geometric representations for processing the design surface of 3D shapes, such as 3D point clouds, voxels, and meshes. Among the explored representations, 3D point clouds are the simplest and most flexible representation, and thus compatible with a wide range of applications [6]. 3D point clouds which are a list of data point coordinates in 3D space are sampled directly from the surface of the CAE models, require less pre-processing effort, and preserve enough geometric details.

Popular deep-generative models for point cloud data are autoencoders (AEs) [1], variational autoencoders (VAEs) [2], and generative adversarial networks (GANs) [7], used to learn and transform high-dimensional 3D data into a compressed low-dimensional representation. The most common AE architectures that process from 3D point clouds are PointNet [8] and PointNet++ [9]. Many networks have adopted PointNet architectures because the architecture addresses the permutation invariance of points in point clouds by handling the input data with point-wise operators followed by a global operator, *e.g.*, *max-pooling*. AEs typically comprise a bottleneck layer after the global operator to learn 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 features, and a decoder, which generates 3D point clouds from representations in the latent space. However, the lack of regularization in the latent space limits the shape-generative capabilities of AEs.

VAEs follow the same topology that of AEs but enforce the regularization of the latent space by adding Kullback-Leibler (KL) divergence as a term in the training loss function of the network. Hence, by improving the distribution of the representations in the latent space, the VAE improves the shape-generative capability compared to AEs [3]. However, the latent representations of AEs and VAEs provide a high-level summary of the input representations [10], and a quantitative analysis of the information stored in latent representations of this model is currently missing.

B. Learning Surrogate Models on Latent Representations

Previous research in the Chemistry domain utilizes autoencoders to learn latent representations for classification and regression tasks [5], [11]. However, in the automotive domain, few studies use analogous representations to predict aerodynamic forces for real-world applications. In 2D laminar flow estimation, Eismann *et al.* [12] utilized a variational autoencoder for learning latent representations of 2D shapes and Gaussian process regression to predict the corresponding drag coefficients. For 3D shapes, Umetani *et al.* [13] utilized a machine learning-based regression model to predict aerodynamic forces of 3D shapes, along with a time-averaged velocity field around the object. However, the proposed model is limited to 3D shapes parameterized using PolyCube maps.

Different from the prior research, we learn design representations from unstructured data. Thus, since the encoder processes data through point-wise operations, the VAE learns primarily local design features. Furthermore, in our application, the data available for training our surrogate model is limited to a subset of the available shapes. Previous works address this data limitation problem by utilizing the generative capability of VAEs to synthesize data to tackle imbalanced data sets [14] and improve classification accuracy [14], [15]. In contrast to classification tasks, the labels in the regression task are in continuous space, so associating labels to each of the generated samples is a challenging task. Therefore, recovering the performance labels from 3D representations in the latent space of the VAE is a more challenging task than in the reviewed cases.

For regression tasks, the multi-layer perceptrons (MLPs) are a popular neural network architecture [16] due to its structural flexibility. An MLP maps input to output data through input, one or more hidden, and output layers. The layers comprise artificial neurons activated with nonlinear functions and fully connected to others in successive layers. An MLP learns the mapping by adjusting the connection weights between neurons by using the back-propagation algorithm [17] for minimizing the error between the MLP predictions and expected output values. In our research, we utilize a multi-output MLP to map the latent representations with their corresponding performance metrics.

III. Methodology

In this section, we first introduce the pre-processing step for sampling point cloud data from surfaces, and we introduce the architectures of our point cloud autoencoder (PC-AE) and variational autoencoder (PC-VAE). Next, we provide details on the information-theoretic measures which we apply to evaluate the information contained in the trained latent representations. Finally, we introduce the architecture for our multi-layer perceptron for the regression task and describe our data augmentation approach using our PC-VAE.

A. *Pre-Processing of the 3D Point Cloud Data*

For our experiments, we sample 3D point clouds from polygonal meshes of the car class of ShapeNetCore [18], which is a large data set of 3D shapes, using the shrink-wrapping algorithm utilized in [19]. The shrink-wrapping samples points uniformly from the external surfaces of 3D objects, which are most relevant for our applications, and generates point clouds organized based on the vertex assignment of the shrink-wrapped mesh. We set the algorithm with six shrinking steps and a single smoothing step (Fig. 1). We sample 3500 shapes from the car class and each shape consists of 24578 points.

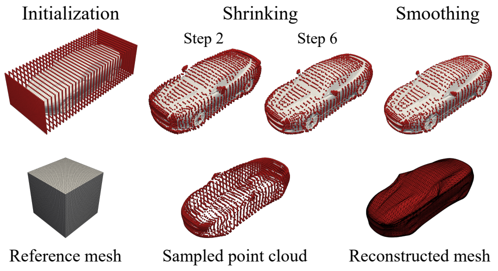


Fig. 1. Shrink-wrapping.

Training our AEs on organized point clouds has two main advantages. First, since the point correspondence between point clouds is known, it allows us to utilize simpler loss functions, here: mean-squared distance, which are computationally less expensive. Second, by enforcing the autoencoder to learn the ordering of the vertices in the shrink-wrapped mesh, it allows us to utilize the autoencoder predicted point cloud to update the vertices of the mesh and quickly generate water-tight models for engineering simulations.

B. *Design Performance Data*

In this work, we define the performances of a 3D design by its volume and aerodynamic drag coefficient. We selected the volume (V) due to its low computational effort and its interpretation, which is in line with the analysis of the latent representations according to [20]. The aerodynamic drag (F_x) requires higher computational effort to compute and is highly nonlinear with respect to the latent representations. Hence, finding suitable surrogate models for aerodynamic data is highly of interest as it would accelerate design optimization in real-world automotive development [21].

To calculate the aerodynamic drag, we perform CFD simulations on the shrink-wrapped meshes using OpenFOAM®[1]. We assume that the cars drive at $U = 110$ km/h in a straight line and that the cars are symmetric with respect to the vertical-longitudinal (xz) plane. We verify the convergence of each model based on the drag force calculated in the last 20 simulations steps, and select the models with standard deviation within 5% of the mean value and magnitude within $[75, 500]$. From the 3500 shapes in the original data set, we obtained the corresponding performance metrics of 600 shapes.

C. *Learning on 3D Point Clouds*

a) Point Cloud Autoencoder (PC-AE): For our experiments, we utilize the PC-AE architecture proposed in [4]. The encoder comprises five 1D convolutional layers, followed by a permutation-invariant max-pooling layer that yields the latent representation z. The decoder comprises three fully connected layers and generates 3D point clouds from samples in the latent space. All layers are activated with rectified linear units (ReLU), apart from the max-pooling and output layers, which are activated with hyperbolic and sigmoid functions, respectively (Fig. 2).

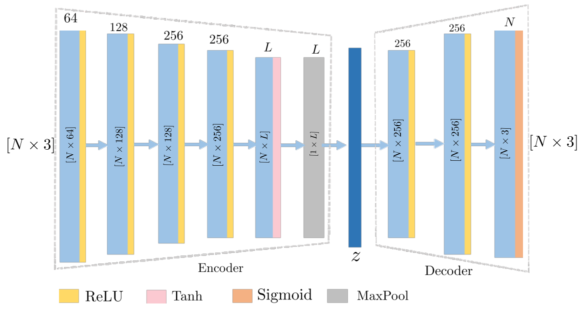


Fig. 2. PC-AE architecture, where N is the number of points in the point-cloud and L is the number of latent variables. The output dimensions of each layer is written vertically inside each layer.

b) Point Cloud Variational Autoencoder (PC-VAE): We train the PC-VAE as proposed in [3]. The architecture is similar to the PC-AE, except that the latent layer is divided into two parts: a mean vector μ and a standard deviation vector σ with a sigmoid activation function (Fig. 3). Thus, the latent representation z is sampled from the encoder's output

[1]https://www.openfoam.com/

distribution with mean μ and standard deviation σ, from which the decoder generates a 3D point cloud. For training the PC-VAE, we minimize a loss function with two terms (Eq. 1): the first term is defined by the difference between the input and reconstructed point cloud, and the second is the KL-divergence loss ($\mathcal{D}_{KL}$) between the learned latent distribution and an assumed prior normal distribution.

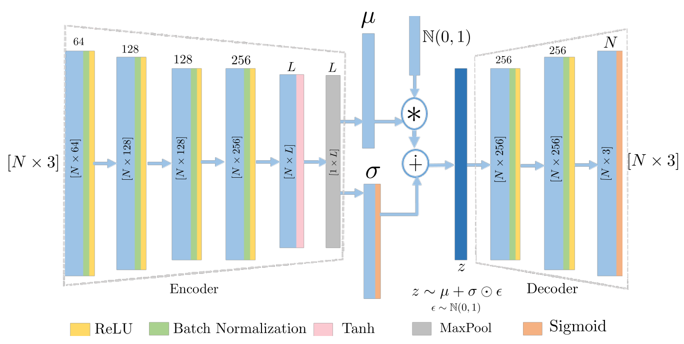


Fig. 3. PC-VAE architecture, where N is the number of points in the point-cloud and L is the number of latent variables. The output dimensions of each layer is written vertically inside each layer.

$$\mathcal{L}_{VAE}\left(x_i, x_o\right) = \alpha\, L_{recon} + \beta\, \mathcal{D}_{KL} \tag{1}$$

In [3], [4], the authors utilize the chamfer distance (CD) to measure the reconstruction loss L_{recon} (Eq. 2), where $S_i, S_o \subset \mathbb{R}^3$ are the input and output point clouds, and x_i, x_o are points in the input and output point clouds respectively.

$$CD = \sum_{x_i \in S_i} \min_{x_o \in S_o} \|x_i - x_o\|_2^2 + \sum_{x_o \in S_o} \min_{x_i \in S_i} \|x_i - x_o\|_2^2 \tag{2}$$

Since the function is invariant with the permutation of the points, the networks trained with CD generate unorganized point clouds. However, as we generated a data set with organized point clouds, we train another PC-AE and PC-VAE with the mean-squared distance (MSD) as a loss function (Eq. 3) to maintain the correspondence between the input and output point clouds and thus generating an organized output point cloud. The MSD between input and output point clouds, defined as

$$MSD = \frac{1}{N} \sum_{j=1}^{N} \|x_{i,j} - x_{o,j}\|_2^2 \tag{3}$$

where $x_{i,j}$ and $x_{o,j}$ are the j-th points of the input and output point clouds respectively.

Furthermore, since the VAE learns a stochastic mapping between the input space and the latent space z, we consider the latent representations of the VAE as $z = \mu$ (in Fig. 3) to get a fixed set of latent variables for our experimental analysis in the following sections.

D. *Information-Theoretic Analysis of the Latent Representations*

We analyze the relationships between the latent representations learned by our PC-(V)AEs and the design performance metrics by calculating the Pearson correlation coefficient (PCC) and mutual information (MI). Both measures quantify the relationship between a pair of variables X and Y. However, the $PCC(X, Y)$ assumes a linear relation between the variables, while the MI measures the amount of information that X holds about Y as the KL divergence between the joint distribution $P_{(X,Y)}$ and the product between the marginal distributions P_X, P_Y. Hence, MI describes also nonlinear relations in the data, which is not captured by PCC. To estimate the MI, we used the estimator for continuous input variables by Kraskov *et al.* [22]. We use both PCC and MI to quantify relationships between variables to evaluate whether using the PCC is sufficient, which may miss non-linear relationships, while it is computationally less expensive to estimate from data.

Note, that MI estimators have known bias when applying them to finite data [22]. A common approach to handle this bias is to apply statistical testing to the MI estimate, to evaluate whether the estimated MI is truly different from zero. We here use permutation testing under the Null-hypothesis of zero MI. The Null distribution is generated by repeatedly estimating the MI from shuffled data such as to generate a Null-distribution, against which the original estimate is compared.

E. *Surrogate Model*

For training a surrogate model from the latent representations to their desired properties, we train a 2-output multi-layer perceptron (MLP) from latent representations z to fit two outputs variables: normalized volume (V) and normalized drag coefficient (F_x). The multi-output MLP is trained for a regression task, i.e., given a latent representation z of an unseen 3D shape instance $S \subset \mathbb{R}^3$, the learned multi-output regression function $f(\cdot)$ predicts a two-dimensional vector y as output, predicting V and F_x.

Our MLP comprises a single hidden layer with 10 hidden neurons, which are activated with ReLU. The output layer comprises two neurons that yield the predicted values of volume (V) and drag coefficient (F_x), and is activated with a sigmoid function. We divide the data set into partitions of 85$\%$ and 15$\%$ for training and testing, respectively, and applied a 5-fold cross-validation strategy on the training set to tune the number of hidden neurons, and the learning rate of the MLP. We optimize the weights of the network by minimizing the mean-squared error for a maximum number of 1000 epochs utilizing the Adam optimizer [23]. We set the optimization algorithm with a learning rate of $\eta = .001$ and feed the data to the network in batches of 50 samples.

After selecting the hyper-parameters of the MLP, we re-train the MLP on the whole training set and evaluate the performance of the surrogate model on the test set using R-square (R^2), root mean square error (RMSE), and mean absolute error (MAE). R^2 score measures the squared correlation between the true and predicted values of the surrogate model, whereas RMSE and MAE measure the prediction error of the model. Thus, the higher the R^2 score and lower the RMSE and MAE, the better the surrogate model.

F. Data Augmentation

Due to the data cleaning, the available performance measures data (Section III-B) are sparse in certain regions of the latent space. This imbalance in the data set potentially leads the MLP to overfit the data corresponding to the most abundant target values or miss input-output relations of less observable samples. Thus, performance prediction of designs in the sparse regions are less accurate. To address this issue, we utilize the trained VAE to extrapolate new designs based on the distributions (μ, σ) learned in the sparse regions.

The advantage of using the VAE is that the stochastic latent space allows us to quickly generate new designs with shared geometric features (Fig. 4), which is challenging to perform by directly manipulating the ShapeNetCore models. The sampled latent vectors are fed to the trained decoder of the PC-VAE-MSD. Note, we used the PC-VAE trained on MSD for this shape generation task, as we aim to generate water-tight meshes from the output point clouds, which is a requirement for CFD simulations.

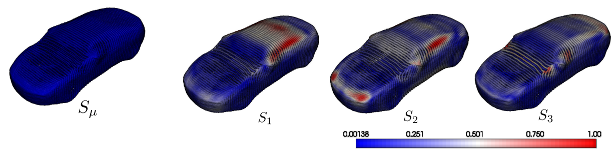


Fig. 4. Reconstruction of the sampled shapes from a latent distribution. S_μ is the shape reconstructed by decoding the μ vector of the distribution. The colors of shapes (S_1, S_2, S_3) indicate difference of the shape from S_μ.

IV. Analysis of the Representations for Regression Tasks

Considering the discussed methods, we present a setup for training the PC-AE and PC-VAE architectures and analyze the different learned latent representations using the PCC and MI.

A. Models Training

1) Data: The data set for our experiments comprises 3500 shapes selected from the car class of the ShapeNetCore repository [18], sampled with shrink-wrapping (Section III-A). To train the networks, we randomly selected 90% percent of the shapes in the data set. The coordinates of each point cloud in the training data were normalized to the range $[0.1, 0.9]^3$, preserving the aspect ratio of the shapes.

2) Training the PC-AE: We trained the PC-AE with a 20D latent vector using the Adam optimizer [23] with a learning rate $\eta = 5\text{E-}04$. The training data was organized into batches of 50 shapes and the network was trained for 700 epochs. We considered two variations of the PC-AE models: The first (PC-AE-CD) was trained utilizing the Chamfer Distance (CD) as reconstruction loss, while the second (PC-AE-MSD) was trained using MSD as loss function.

3) Training the PC-VAE: We trained PC-VAE with a 20D latent vector, using the Adam optimizer and a learning rate of $\eta = 5\text{E-}03$. The hyper-parameters α and β to balance the reconstruction and KL divergence loss for the PC-VAE-CD (in Eq. 1) were tuned to $\alpha = 250$ and $\beta = 0.001$.

Similarly, we also considered a second PC-VAE model where we replaced the CD with the MSD in the loss function (PC-VAE-MSD). Our motivation is to allow the PC-VAE to generate water-tight meshes (Section III-A). Hence, we optimized the hyper-parameters α and β for the PC-VAE-MSD using a grid search. We selected the values of $\alpha = 1000$ and $\beta = 0.001$ for the hyper-parameters as they yield an acceptable trade-off between reconstruction accuracy and divergence in the latent space.

All the 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 the cases the networks were trained on a single GPU.

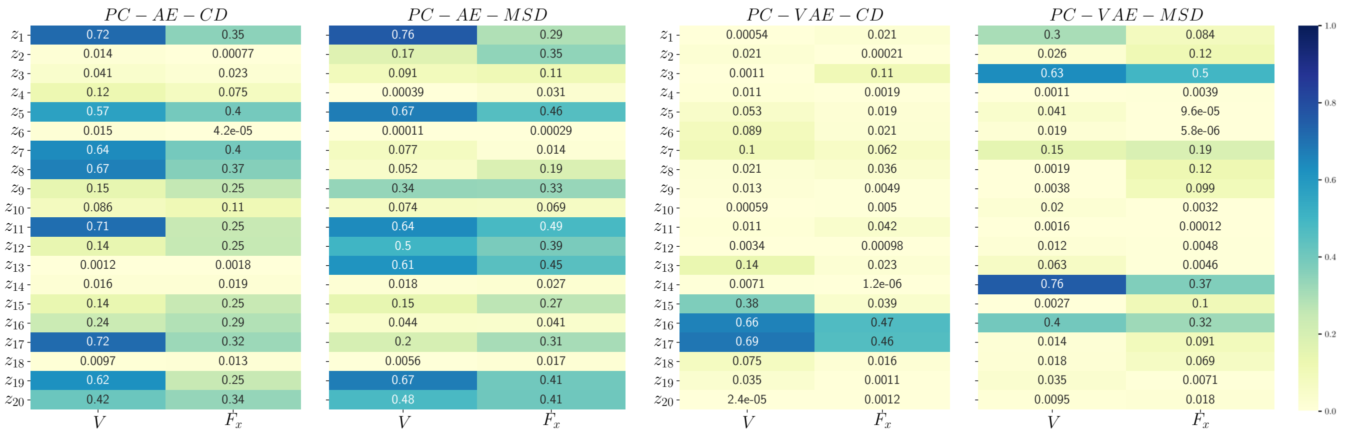
B. Information-Theoretic Analysis of the Latent Representations

We calculated the Pearson correlation coefficient (PCC) and mutual information (MI) between the latent representations learned by each model (PC-AE-CD, PC-AE-MSD, PC-VAE-CD, and PC-VAE-MSD) and two performance metrics: normalized volume (V) and normalized drag coefficient (F_x) (Figs. 5 and 6). In this analysis, we considered only the sub-set of the data with converged CFD calculations, which comprised 600 shapes.

The absolute values of PCC and MI obtained for the PC-AE models were higher than observed for the PC-VAE models. Higher PCC magnitude indicates a lower complexity for surrogate models to learn the data since the relations between input and output have a higher degree of linearity. The MI indicates that the latent representations of the PC-AE hold more information about the design performances than the PC-VAE variants. Hence, regardless of the data linearity, the PC-AE latent space potentially leads to more accurate surrogate models, since it allows the surrogate model to leverage more information for learning the performance.

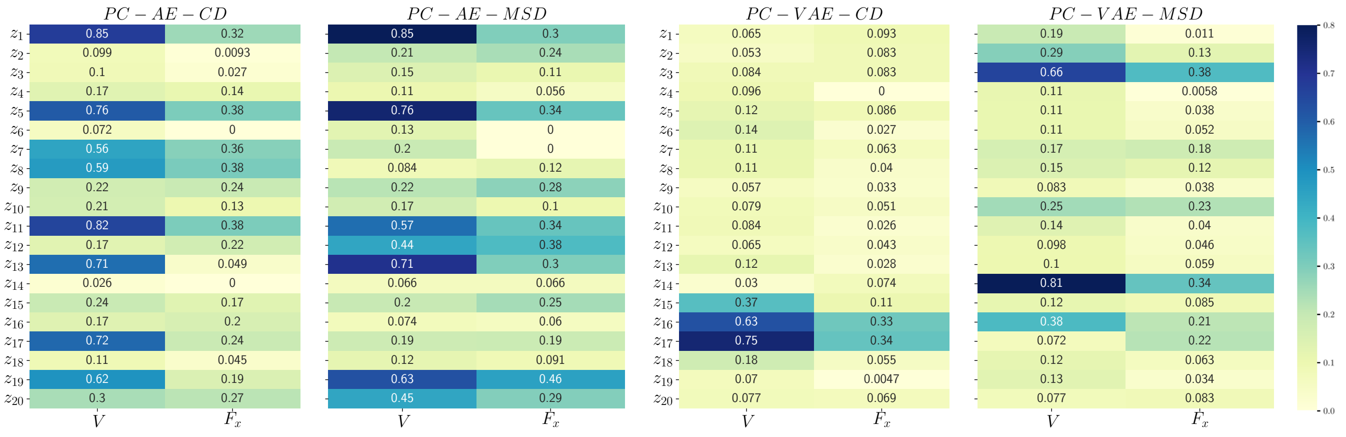
We also noticed that the values of PCC and MI were more evenly distributed in the PC-AE latent spaces than obtained with the PC-VAE. This difference indicates that the PC-AEs enable different design degrees of freedom than the PC-VAEs and are in line with the results reported in [19]. Furthermore, comparing the obtained MI between models trained with CD and MSD (Fig. 6, columns (b) and (d)), we observed that models trained with MSD yielded higher values. Our interpretation is that the MSD allows the autoencoders to implicitly learn global shape information by enforcing the organization of the points and, thus, improving the correlation of the latent representations to more complex performance data.

To test statistical significance of MI estimations in Fig. 6, we shuffled the order of values for each latent variable, z_i, to obtain permuted values z'_i. We then calculated a surrogate



(a) (b) (c) (d)

Fig. 5. Pearson Correlation between the latent representations of the autoencoders (PC-AE-CD, PC-AE-MSD) and variational autoencoders (PC-VAE-CD and PC-VAE-MSD) and normalized performance metrics calculated from the designs: volume (V) and drag (F_x).



(a) (b) (c) (d)

Fig. 6. Mutual Information between the latent representations of the autoencoders (PC-AE-CD, PC-AE-MSD) and variational autoencoders (PC-VAE-CD and PC-VAE-MSD) and normalized performance metrics calculated from the designs: volume (V) and drag coefficient (F_x).

MI value $I'(z'_i, y)$. This process was repeated 200 times for each combination of latent variable z_i and performance value. The p-value for each $I(z_i, y)$ was obtained as the fraction of surrogate values larger than the original estimate, $I'(z'_i, y) > I(z_i, y)$. We considered an estimate statistically significant if the p-value was lower than the critical α-level of 0.05. We found that all MI estimates were statistically significant at the specified α-level. Hence, we also find for MI estimation in the PC-VAE latent space that a small set of variables holds information about the performance, compared to more latent variables with high information in the PC-AE latent space.

Hence, we concluded that our baseline MI values (Fig. 6) are representative and our observation that the MSD-based training increases the information contained in the latent space still holds. Furthermore, by visually inspecting the shape reconstructions (Fig. 7), we observed that the MSD improved the quality of the mesh reconstruction of the PC-VAE, which confirms the advantages we expected to achieve by modifying the loss functions.

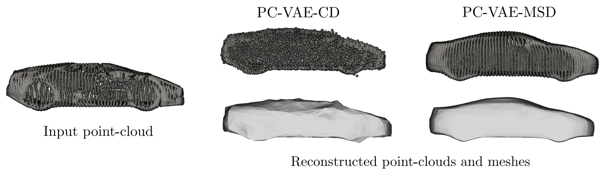


Fig. 7. Reconstruction of the input point cloud using our trained PC-VAE-CD and PC-VAE-MSD models.

V. Regression Analysis

To train a surrogate model, we considered 600 shapes along with their performance metrics (as described in Section III-B).

A. Surrogate Model Training

We trained two MLPs to predict the performance metrics: The first based on the latent representations of PC-AE-MSD

(MLP-AE) and the second based on the latent representations of PC-VAE-MSD (MLP-VAE). We considered a two-output multi-layer perceptron to map the latent representations with the performance metrics: F_x and V. As a baseline, we trained a model that predicted the performance as the mean performance in the training set for both, the PC-AE-MSD (Baseline AE) and the PC-VAE-MSD (Baseline VAE). We divide the dataset of 600 shapes with their performance metrics into 85% and 15% training and test set.

To consider the stochastic nature of the MLPs initialization, we evaluated the performances on the training and test set over 30 runs and reported the mean and standard deviation of the errors. Table I shows the prediction performances of MLPs and baseline regression models on the training and held-out test set. We observed that both models, MLP-AE and MLP-VAE showed better prediction performances compared to the baseline models. The R^2 score, RMSE, and MAE between MLP-AE and MLP-VAE are comparable for the training set. However, in the test set, the MLP-AE predicts the output performances more accurately than the MLP-VAE, which indicates that the latent space learned by the PC-AE is more suitable for surrogate modeling than the space learned by the PC-VAE.

We visually analyzed the prediction results of the two performance metrics using two scatter plots of true predicted values for the test samples in Fig. 8. We observed that both models predicted the volume with better performance than the aerodynamic drag, for which the scatter plot showed higher dispersion of the data with respect to the ideal regression model.

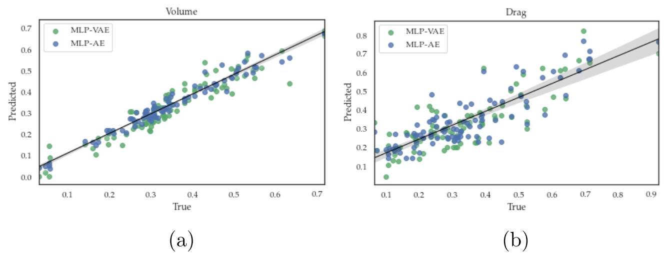


Fig. 8. True *vs.* predicted values of drag and volume measures on the test set.

In the second verification, we analyzed the reconstructions of 4 samples from the test set, with their performance predicted values (Fig. 9). We observed that the prediction of the MLP-AE on 4 samples is more accurate compared to MLP-VAE. However, for the shape with the lowest drag (second column), both models failed to predict the lower drag value. We assumed this could be due to limited data of that particular car class in the training set.

Analyzing the distribution of the normalized volume and drag of the training samples, we observed that most of the samples yield normalized performance values in the interval $[0.2, 0.6]$. To improve the distribution in the training data, we explored the shape generative capability of the PC-VAE-MSD to augment the data set and improve the quality of the surrogate models.

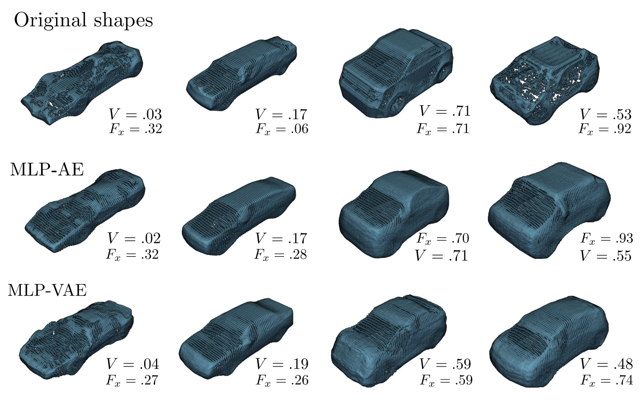


Fig. 9. Comparison between the original shapes in the test set with true performance values and reconstruction of the samples using PC-AE-MSD and PC-VAE-MSD with performance values predicted using trained MLP regressors on their latent space.

B. Exploiting the Generative Capabilities of VAE for Data Augmentation

To augment the dataset, we sampled additional 300 latent vectors based on the input shapes with normalized volume and drag measures below 0.2 and higher than 0.6. For each vector, we reconstructed the point clouds using the decoder of the PC-VAE-MSD (Section III-F), converted the 3D point clouds into meshes, and calculated the volume and drag coefficients of these shapes using OpenFOAM®.

Starting with the prediction performance on the test set in Table I, we incrementally added a part of the augmented dataset generated by PC-VAE-MSD to the existing training set. The ratio r of the amount of generated samples added to the training data of the regression model is increased from 0 to 1 by 0.2.

We trained the MLPs with the augmented data set and tested on the previously held-out test set for 30 runs and reported the mean and standard deviation of the RMSEs on the test set. We evaluated the generalization performance of the surrogate models on the test set by adding augmented data. When ratio $r = 0$, the regression model is trained only on the original training data, so the RMSE values in Table II are similar to the ones in Table I.

We observed in Table II that augmenting the data set improved the generalization performance of both, the MLP-VAE and MLP-AE since the RMSE values decreased. We concluded that from ratios 0.2 to 0.6, the added samples increased the diversity of the data, which in general improves the quality of the surrogate models. But for higher $r = 0.8$ or $r = 1$ ratios, the added samples shared a high degree of similarity with the existing data, thus adding redundancies and decreasing the performance of the MLPs. Further, we confirmed our observation by testing the difference in the RMSE values of each ratio with the RMSE values at $r = 0$

TABLE I

PREDICTION ERROR ESTIMATION USING R-SQUARED VALUE (R^2), ROOT MEAN SQUARE (RMSE) AND MEAN ABSOLUTE ERROR (MAE) ON TRAINING AND TEST SET. BEST MODEL FOR TRAIN-TEST SPLIT SHOWN IN BOLD.

	Training set			Test set		
Models	R^2	**RMSE**	**MAE**	R^2	**RMSE**	**MAE**
Baseline AE	-.005 ± .004	.160 ± .002	.120 ± .001	-.007 ± .000	.119 ± .000	.156. ± .0000
MLP - AE	**.875 ± .002**	**.044 ± .002**	**.035 ± .004**	**.834 ± .003**	**.048 ± .0006**	**.068 ± .0015**
Baseline VAE	.101 ± .280	.140 ± .030	.110 ± .020	-.008 ± .000	.110 ± .000	.157 ± .0000
MLP -VAE	.861 ± .010	.048 ± .005	.036 ± .002	.750 ± .001	.055 ± .0023	.072 ± .0023

TABLE II

RMSE ON TEST SET. RMSE MARKED IN BOLD ARE STATISTICALLY SIGNIFICANT COMPARED TO THE BASELINE RMSE (WHEN $r = 0$).

Ratio of Augmented Data	**MLP-AE**	**MLP-VAE**
0	0.0480 ± .0006	0.0550 ± .0023
0.2	**0.0454 ± .0011**	**0.0488 ± .0014**
0.4	**0.0447 ± .0011**	**0.0492 ± .0015**
0.6	**0.0452 ± .0013**	**0.0491 ± .0011**
0.8	0.0474 ± .0012	0.0557 ± .0018
1	0.0492 ± .0017	0.0570 ± .0020

using the Mann-Whitney test. The results of the statistically significant analysis are in bold (Table II) (p-values$<$ 0.01). Nevertheless, the experiment shows that the data augmentation using the PC-VAE has the potential to improve the quality of surrogate models, which was our objective.

VI. CONCLUSION

Deep-generative models provide a novel data-driven representation for design exploration in automotive engineering. However, these models still require engineers to perform computationally expensive simulations to assess the performance of the generated designs. In this paper, we address this challenge by learning surrogate models that map the representations learned by point cloud (variational) autoencoders to engineering performance data. Hence, by coupling the surrogate models to the autoencoders, we enable engineers to explore designs taking into account the estimated performance on different domains.

We addressed two challenges for learning surrogate models based on the latent representations learned by the point cloud (variational) autoencoders. First, the autoencoder training data neglects any explicit design performance information. Hence, we measured the mutual information between the latent variables learned by four different architectures and two performance metrics (shape volume and aerodynamic drag) to evaluate the capability of these models to learn performance-related information. We concluded that enforcing the autoencoders to learn organized point clouds increases the amount of information available in the latent space, potentially by providing an implicit description of design global design features.

Second, we utilize the regularized latent space of a variational point cloud autoencoder for fast design prototyping and data augmentation. Since imbalanced training data decreases the quality of surrogate models, we utilize the distributions learned by the variational autoencoder to sample and generate new realistic designs in sparsely sampled regions of the data set, which is faster and more intuitive than direct manipulation of the 3D designs. We show that, within a threshold, our proposed method improved the accuracy and generalization capability of the surrogate models for predicting aerodynamic performances of car shapes.

REFERENCES

[1] 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.

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

[3] 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.

[4] 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, pp. 791–798.

[5] 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.

[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] 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.

[8] 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.

[9] 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.

[10] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel, “Variational lossy autoencoder,” in *5th International Conference on Learning Representations (ICLR)*, 2017, pp. 1–17.

[11] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[12] S. Eismann, S. Bartzsch, and S. Ermon, “Shape optimization in laminar flow with a label-guided variational autoencoder,” *ArXiv*, vol. abs/1712.0, no. 3, pp. 0–4, 2017.

[13] N. Umetani and B. Bickel, “Learning Three-Dimensional Flow for Interactive Aerodynamic Design regression prediction for new shape,” *ACM Transactions on Graphics*, vol. 37, no. 4, 2018.

[14] Z. Wan, Y. Zhang, and H. He, "Variational autoencoder based synthetic data generation for imbalanced learning," *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017 - Proceedings*, vol. 2018-January, pp. 1–7, 2018.

[15] Z. Islam, M. Abdel-Aty, Q. Cai, and J. Yuan, "Crash data augmentation using variational autoencoder," *Accident Analysis and Prevention*, vol. 151, no. July 2021, 2021.

[16] F. Murtagh, "Multilayer perceptrons for classification and regression," *Neurocomputing*, vol. 2, no. 5, pp. 183–197, 1991.

[17] Y. Hirose, K. Yamashita, and S. Hijiya, "Back-propagation algorithm which varies the number of hidden units," *Neural Networks*, vol. 4, no. 1, pp. 61–66, 1991.

[18] 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

[19] T. Rios, B. van Stein, T. Back, B. Sendhoff, and S. Menzel, "Multi-Task Shape Optimization Using a 3D Point Cloud Autoencoder as Unified Representation," *IEEE Transactions on Evolutionary Computation*, no. December 2020, pp. 1–12, 2021.

[20] T. Rios, B. van Stein, S. Menzel, T. Back, 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.

[21] C. Smith, J. Doherty, and Y. Jin, "Royal Aeronautical Society – Applied Aerodynamics Conference 2014," 2014, pp. 1–13.

[22] A. Kraskov, H. St, and P. Grassberger, "Estimating Mutual Information," *Physical Review*, vol. 69, no. 6, 2004.

[23] 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.