Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 91 (2020) 3–8

www.elsevier.com/locate/procedia

30th CIRP Design 2020 (CIRP Design 2020)

Deep Learning for Automated Product Design

Carmen Krahe[a] *, Antonio Bräunche[a], Alexander Jacob[a], Nicole Stricker[a], Gisela Lanza[a]

[a]*Karlsruhe Institute of Technology, Kaiserstrasse 12, 76131 Karlsruhe, Germany*

* Corresponding author. Tel.: +49-721-608-44011 ; fax: +49-721-608-45005. *E-mail address:* carmen.krahe@kit.edu

Abstract

Product development is a highly complex process that has to be individually adapted depending on the companies involved, the product to be developed and the related designers. Within this process, the approach and the know-how of the designer are very individual and can often only be described with high effort in a rule-based manner. Nevertheless, numerous routine tasks can be identified that offer enormous automation potential. Machine Learning, especially Deep Learning, has proven an immense capability to identify patterns and extract knowledge out of complex data sets. Autoencoder networks are suitable for the conversion of different 3D input data, e.g. Point Clouds, into compact latent representations and vice versa. Point Clouds are a universal representation of 3D objects and can be derived from various 3D data formats. The goal of the approach presented is to use Deep Learning algorithms to identify design patterns specific to a product family out of their underlying latent representation and use the extracted knowledge to automatically generate new latent object representations fulfilling distinct product feature specifications. A deep Autoencoder network with state-of-the-art reconstruction quality is used to encode Point Clouds into latent representations. In this approach, a conditional Generative Adversarial Network operating in latent space for generation of class-, characteristic- and dimension-conditioned objects is introduced. The model is quantitatively evaluated by a comparison of given specifications and the implemented features of generated objects. The presented findings can be used to support designers in the creation process by automatically proposing appropriate objects as well as in the adaption of future product variants to different requirements. This relieves the designer of time-consuming routine tasks and reduces the effort of knowledge-transfer between designers significantly.

Keywords: Artificial Intelligence; Machine Learning; Computer Aided Design; Automation

1. Introduction

Modern product development is shaped by shortening product life cycles, customer individualisation and is, due to the necessary efficiency, mostly based on reference products [1]. Different generations and variants are released throughout a product's lifecycle. This is necessary to increase market attractiveness or comply with revised specifications or standards. Even though the process of designing new products is highly individual and depends on the experience and creativity of engineers and designers [2], a significant automation potential can be identified. For example, different requirements between product generations, revised specifications and standards require manual adaptation of the corresponding CAD model. Automatic creation and adaption of models can support designers by enhancing creativity through suggestions of appropriate object shapes and, moreover, by saving time-to-market through automation of time-consuming routine tasks. To achieve this, highly abstract design patterns have to be identified and applied in the object generation process. Deep Artificial Neural Networks are able to address such complex issues. In contrast to pattern

10.1016/j.procir.2020.01.135

description with case-specific rules, they also allow for transferability to new data, other product families or different engineering domains by transfer learning or re-training. This can be seen as significant advantage in terms of generalization and scaling for efficient product design. In this paper, state-of-the-art research of Conditional Generative Adversarial Networks (GANs) [3], Autoencoder (AE) networks and generative representations of Point Cloud models is applied in the context of product development to present methods capable of supporting engineers and designers in their daily work. The specific contributions are:

- A scalable architecture of Conditional GANs operating in latent space, inspired by recent architectures of latent GANs [4] and Conditional GANs [5], that are able to generate latent representations of new 3D objects of (I) realistic appearance, (II) specified class, (III) specified characteristics and (IV) specified dimensions
- An experimental evaluation of the proposed approach in terms of qualitative features like the realistic appearance of generated objects and quantitative features like deviations between specified and implemented dimensions of an object

These aspects will be presented in the following chapters.

2. State of the Art

Deep Learning is one of the most promising fields of research of Artificial Intelligence (AI) in the past years [6]. Deep Learning algorithms can tackle various tasks, for example classification, clustering or regression and are used in many fields, for example Image Recognition, Natural Language Processing and Robotics [6], partly reaching superhuman performances [7]. Several approaches in manufacturing and design exist, which cover different areas of the engineering domain, e.g. visual part recognition for robot control [8], clustering for modular product architecture [9] and generating 2D-car design sketches [10].With the accessibility of large 3D databases [11,12] the interest in Deep Learning applications on 3D objects increased significantly. Current approaches to AI-supported processing of CAD can be differentiated in two groups according to the degree of automation. On the one hand, certain manually defined features of the model are encoded and then processed by using machine learning methods [13]. On the other hand, features can be learned automatically from the input data. In this area, mainly four different approaches can be differentiated: multi-view images [13,14], voxel models [15], Point Clouds [16,17] or graphs [18].

With the multi-view approach introduced by [13] 3D objects are represented by images generated from different perspectives. Those images can then be used for classification tasks with the help of Convolutional Neural Networks (CNNs). The voxel approach is based on approximating the objects through a cube-shaped grid, as in [15]. In analogy to pixels, the generated voxel structures can then be classified by CNNs with three-dimensional convolution kernels. In addition, there are also approaches that combine 2D projections and voxels representations for classification, as in [19]. To represent 3D models by point-clouds, points are distributed randomly in the volume or on the surface of the model. The coordinates of those points can then be used as input for Deep Learning algorithms. For example in [17], Point Clouds are used for classification and segmentation tasks.

More recent approaches, as presented in [4,20–26], focus on generative and discriminative representations for geometry, mainly based on view-based image projections, voxel grids or Point Clouds. They make use of a combination of AEs and generative models, such as GANs [3], to create new 3D objects. An AE is a Deep Learning architecture consisting of an encoder and a decoder that is trained to first encode the input into a low dimensional representation, and then to reconstruct the original input from this compressed latent representation again. In contrast, a GAN is based on an adversarial game between a generator and a discriminator. The generator is trained to synthesize samples that look very similar to real data. The discriminator, on the other hand, is trained to distinguish between artificially generated and real data. For example [4] introduces an approach based on Point Clouds, where, in addition to pure classification tasks, generative models are also used to synthesize new 3D objects in form of Point Clouds. An AE is trained to learn a data representation, on the basis of which realistic examples from complex underlying distributions are generated with the help of GANs. However, the output of these approaches is created randomly and in no way specified. Conditional GANs enable feature specification to control the output of generative models. The idea of these Conditional GANs was first presented in [5], where images are conditioned on their class labels. In [27], an architecture is developed in which the image generation process is factorized firstly in structure and then in style generation. The style is ultimately conditioned by the underlying structure represented by surface normal maps. Not only does this cause the generator to produce real images, but also images that match the given surface normal map. In [28], an approach with a conditional GAN based on Point Clouds was already presented to output a given 3D object with a conditioned rotation angle. However, the application of conditional GANs to Point Clouds has not yet been sufficiently researched [22]. Hence, the contribution of this paper is an approach to condition the generation of new 3D models in the form of Point Clouds, based on the methodology introduced by [4], via a specific 3D object specifications. A proof of concept is carried out on the basis of an exemplary dataset in an experimental evaluation. The long-term goal is to transfer the developed approach to the area of product generation in order to relieve the designer of time-consuming work when designing a product variant with new specifications by automatically generating the corresponding 3D model.

3. Methodology

The approach developed is based on the generative model presented in [4] consisting of an AE and a GAN. The AE converts the 3D objects into a latent vector representation. Since training and use of the GAN is fully based on these latent vectors, the GAN is called l-GAN (latent GAN) and is independent of the input data format. In this paper, Point Clouds are used as 3D data representation. However, on condition that a suitable AE can be trained, the approach can be applied to any 3D data format. In the following, the used Conditional latent space GAN as well as the 3D object specification for conditioning the generated output are described in more detail.

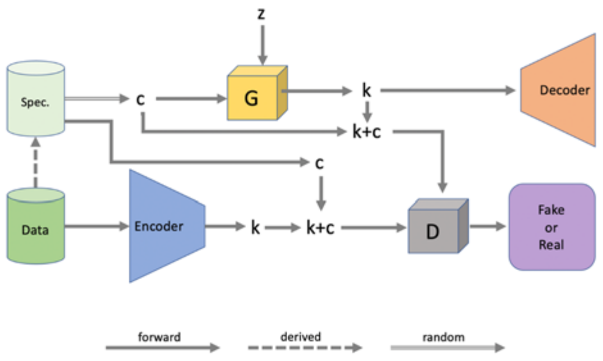

Fig. 1. Specifications are derived from training Point Clouds. Point Clouds are encoded by the encoder to a latent representation k and merged with their respective specifications c to a real discriminator input k+c. The generator receives random noise input z and random specifications c to generate a fake latent object representation k, which is merged with its specifications c to a fake discriminator input k+c. Fake and real inputs are classified by the discriminator alternately. The fake latent representations k can be decoded by the decoder to receive new object Point Clouds with considered specifications.

3.1. Conditional latent space GAN

To actively control the output of the l-GANs generator, the approach of [4] is extended to a Conditional l-GAN. Therefore, additional information is provided as an input to specify features of objects to be generated. In [4], the model input is a noise vector z of size k ×1, drawn by a spherical Gaussian. Here k is the size of the AE's latent representation of the input, also called bottleneck. In contrast to [4], not only does the extended model feed the generator with a noise vector of size k×1, but also with feature specifications of size c×1 (see Fig. 1). This results in an total input vector with size (k+c)×1. Based on this input vector, the generator is trained to generate latent representations of objects of size k×1. The c specifications are appended to the generator output to get a discriminator input of size (k+c)×1. In the same way latent representations of real training objects and their specifications are merged. Real training samples and fake samples of the generator are alternately fed into the discriminator. Based on the given conditions the discriminator learns to consider a comparison of the provided specification parameters with the actual parameters of the object as authenticity criterion. Therefore, the generator is guided not only to generate realistic objects, but also to create objects that correspond to the given parameters in order to maintain its chance to deceive the discriminator.

3.2. 3D object specification

To specify different requirements of a 3D object, various features can be defined in the Conditional l-GAN. One-hot-encoding is applied to integrate discrete features like object classes or characteristic specifications. That means that for c feature types, a vector of size c is appended to the model input (see also 3.1). If an object is to represent a certain characteristic, the vector entry is set to 1 at this position, otherwise to 0. Dimension characteristics such as height and width, on the other hand, are represented in the vector by their specific value. To one-hot-encode class and characteristic features, the training data is labeled with respect to the considered feature type.

In addition to product-specific features, this paper also considers product dimension related features of 3D objects. For this purpose, the dimensions of an object have to be determined. Assuming that the 3D objects, represented by Point Clouds, are aligned in a standardized way, this can be achieved by measuring the difference between the maximum and minimum value of the points on the respective coordinate axis. The evaluation of the generated objects with regard to the required dimensioning is carried out using the amount of the difference between the dimensions D of the generated object and the required dimensions S. Consequently, the error is always greater than or equal to 0 and is determined relative to the given dimension specification:

$$E = \frac{|D-S|}{S} \qquad (1)$$

For instance, if the required dimension is 0.5 units and the observed error is 10 %, the measured dimension of the generated object will be either 0.45 or 0.55 units. To predefine specifications that do not result in objects outside of the solution space, e.g. objects with unrealistic dimensions, the values used should correspond to the considered data set. For this either a distribution for the characteristics of the specification values is estimated and the values are generated according to this distribution, or the values are drawn randomly from the existing specification values of the considered database. In the experiments represented in section 4, the latter is applied.

4. Experimental Evaluation

The approach presented is evaluated through different experiments on Point Cloud object representations derived from ShapeNet [11]. In the following experiments, the l-GAN is conditioned by three different types of specifications:

- **Classes:** A labeled training data set of classes chair, table and sofa is collected.
- **Features:** A labeled training data subset of the class table is created, distinguishing between rectangular and round tables with one, two, three or four table legs respectively.
- **Dimensions:** Depth, width and height of all training data objects are determined and used for the specification of new objects.

In the first experiment the l-GAN is only conditioned on given dimensions, e.g. height and width, for the considered object class (see 4.3). This is expanded to a further experiment by adding certain object feature specifications, which is done exemplarily for the class table, e.g. the number of table legs (see 4.4). The last experiment considers dimension as well as object class specifications (see 4.5). For the executed experiments, two different AEs have been trained: one based on objects of only the table class (single class AE) and one based on objects of the classes chair, table and sofa (multi class AE).

The used AE is setup as proposed in [4], encoding Point Clouds of size 2048×3 to latent representations of size 128×1 and back to the original format again. The original architecture of l-GAN in [4] was adapted according to section 3.1. With a latent vector z of size k×1, k=128, the generator input is increased by the number of conditions c to k+c. According to [4], the noise vector was sampled by a spherical Gaussian of 128 dimensions with a mean of 0 and a standard deviation of 0.2 units. The generator consists of the input layer, a fully connected hidden layer of size k with ReLu activation function, and the output layer (see also Fig. 1). The input layer of the discriminator is adapted in the same manner to increase its input size to k+c. It is built out of two fully connected hidden layers of size 256 and 512, respectively, and the output layer. All Conditional l-GANs used in the experiments are trained with a learning rate of 0.0001 and a batch size of 50 for 1500 epochs.

4.1. Training data

A subset of the ShapeNet database [11] was used to validate the developed approach in various experiments. In [4] a large number of ShapeNet object classes have already been converted to Point Clouds, which are referred to in this paper. In particular, for the experimental evaluation, Point Cloud objects of the classes table, chair and sofa are used. For feature specification, table objects are partly divided up into labeled subclasses, distinguishing between frequently occurring standard classes: rectangular tables with four, two, or one leg, round tables with four, three or one leg. Additionally, to investigate the AE dimension error, a subclass with special, unique tables is introduced. Examples from all subclasses are shown in Fig. 2.

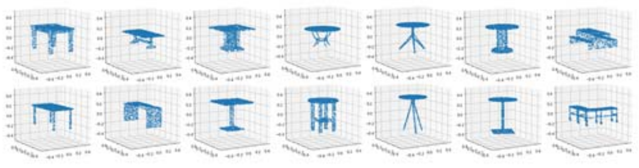
Fig. 2. Table subclasses each represented by two examples from left to right: rectangular 4 legs, rectangular 2 legs, rectangular 1 leg, round 4 legs, round 3 legs, round 1 leg, special.

4.2. Dimension error of the Autoencoder

For the following experiments, it is of particular interest to observe the differences between the required specifications of depth, width and height and the actual dimensions of a generated object. The goal is to minimize these differences while producing new, realistic objects with the Conditional l-GAN. When creating object Point Clouds, two sources of error for different dimensions can be identified. The architecture of the Conditional l-GAN is the first and most important factor to be investigated. However, if high-quality latent object representations with valid dimensions are generated by the Conditional l-GAN, there still remains the possibility of generating relevant dimension errors while decoding the latent vector to a Point Cloud. For a better interpretation of Conditional l-GAN results, the dimension error generated during the encoding-decoding process of the AE with real training data is determined. For this purpose, a data set consisting of 8000 training and 500 test data Point Clouds of tables is created. A single class AE is trained on the training data objects. By comparing the dimensions of 1000 randomly selected training objects and their encoded-decoded representations an average error of 3.31 % across all dimensions is observed. The encoding-decoding process for all 500 test objects outputs an average total dimension error of 3.63 %. Obviously, the AE achieves a solid generalization ability to process unseen data with an increase of the dimension error of only 0.32 %. To investigate the differences of the AE's performance between various object types, the AE dimension error is determined for the table subclasses as depicted in Fig. 2. The results are shown in table 1. Apparently, the AE's performance varies dependent on the considered table subclass. In the case of special tables, high dimension errors of more than 8 % can be observed, whereas the dimension error decreases significantly for more frequently occurring standard tables, mostly even below the average total error.

Table 1. AE's dimension errors of specific table subclasses.

Table subclass	AE dimension error
Rectangular, 4 legs	1.96 %
Rectangular, 2 legs	4.13 %
Rectangular, 1 leg	3.27 %
Round, 4 legs	2.26 %
Round, 3 legs	2.78 %
Round, 1 leg	2.04 %
Special	8.17 %

To investigate the dimension error for experiments in which objects of several classes are considered, a multi class AE has to be trained. For this purpose a training set of 9000 Point Clouds and a test set of 300 Point Clouds from the three classes tables, chairs, and sofas are used. Each class consists of 3000 training and 100 test Point Clouds. After 500 training epochs, an average error of 3.45 % is achieved for training objects and an average error of 3.65 % for test objects. This illustrates the ability of the multi class AE to generalize across different classes, with only slightly worse performance than a single class AE.

4.3. Conditional l-GAN: Fulfilling dimension specifications

A Conditional l-GAN for dimension specification input is trained on all table objects using the pre-trained single class AE described in the previous section. The dimension specifications used as generator input are derived from randomly selected training Point Clouds. To benchmark the observed results an unconditioned l-GAN is instantiated in order to evaluate the corresponding dimension errors. The Conditional l-GAN reduces the dimension error from over 25 % at the beginning of training to 2.34 % after 1500 training epochs. In contrast, the unconditional l-GAN error converges to 27.50 %, which is close to the standard deviation of all dimensions of training data Point Clouds. Fig. 3 shows the course of the corresponding dimension errors over the training process of both GANs. While the dimension error decreases during training, a significant qualitative improvement of the generated objects can also be observed. Especially during the first 100 epochs of training, the visual quality of the generated objects is constantly improving. However, since a certain quality of visual appearance is achieved after 100 epochs, only slight qualitative differences in the generator outputs can be observed between the 100th and 1500th epoch of training. Fig. 4 shows exemplary generator outputs from different training epochs.

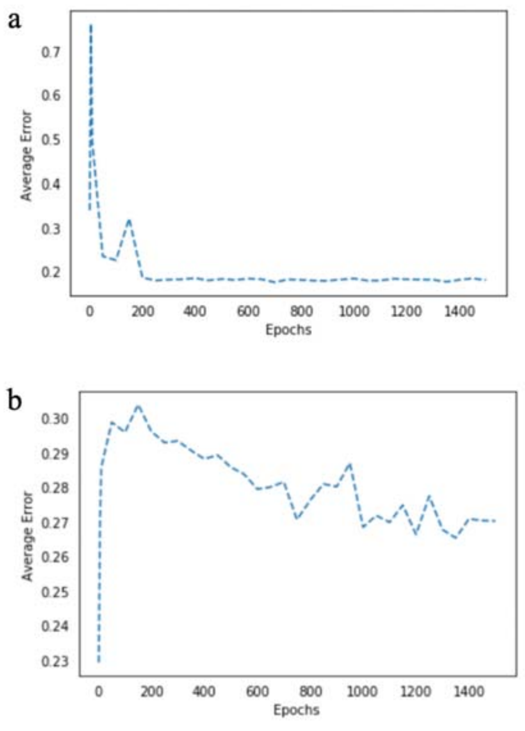

Fig. 3. Comparison of dimension error of Conditional l-GAN (a) and l-GAN(b) over 1500 training epochs.

The significant reduction of the dimension error of the Conditional l-GAN compared to the unconditional l-GAN demonstrates the effectiveness of the investigated approach. It is noticeable that the minimum dimension error of the Conditional l-GAN is less than the average dimension error generated by the AE. Obviously, the Conditional l-GAN produces more standard tables for which the error of the AE is lower. Although it is not possible to determine the proportion of the dimension error caused by the Conditional l-GAN or the AE from the total dimension error, it implies a very low dimension error caused by Conditional l-GAN.

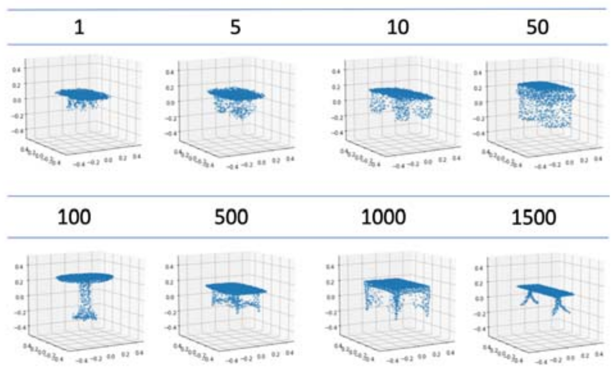

Fig. 4. Conditional l-GAN generator output in different epochs.

4.4. Conditional l-GAN: Fulfilling dimension specifications and object feature specification

In the next step, the Conditional l-GAN is not only conditioned with dimensions, but also with different table features. Thus, it is required to generate specific tables with given dimensions. For this, the specifications regarding features and dimensions are randomly derived from the respective subclasses of tables, which are shown in Fig 2. After 1500 training epochs, the Conditional l-GAN achieves a minimum dimension error of 3.99 % and a manually determined feature error of 0 %. Thus, every feature specification is implemented according to the generated object. Since dimension features are randomly derived from original training objects of the respective subclass, the visual similarity between the corresponding original sample object and the generated object is investigated. As can be seen in Fig. 5, the Conditional l-GAN obviously creates new objects with different styles and shapes.

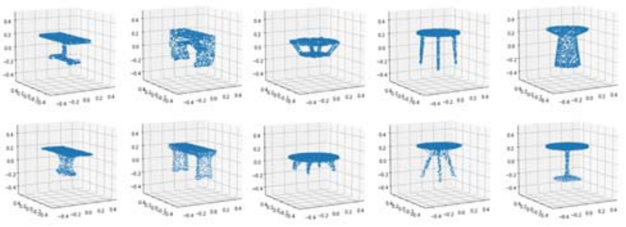

Fig. 5. Depiction of original training objects (upper row) and generated objects (lower row) with respect to the original and generated dimension and feature specification.

4.5. Conditional l-GAN: Fulfilling dimension specifications and class conditions

Within the framework of this experiment, the Conditional l-GAN is trained for generating class and dimension conditioned objects. For this, the pre-trained multi class AE with the categories table, chair and sofa is used. During 1500 epochs of

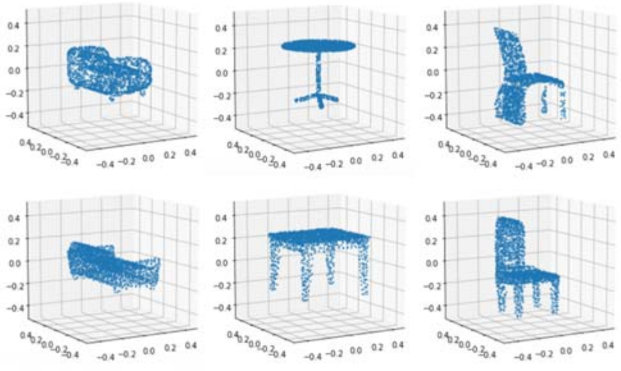

Fig. 6. Depiction of original training objects (upper row) and generated objects (lower row) with respect to the original dimension and class specification.

training, the Conditional l-GAN reduces the dimension error to a minimum value of 4.04 %. In 100 % of the test cases the objects are generated according to the required class. By a visual comparison it can be shown that actually new objects are generated by the Conditional l-GAN, as presented in Fig. 6.

For all experiments a virtual machine with the following specifications is used: NVIDIA K80-GPU, four virtual CPU's and a memory of 61 GiB RAM. The training of the AEs takes about 6 to 7 hours, the training of the GANs about 0.5 hours.

5. Conclusion and Outlook

In this paper, an approach for the conditioned generation of 3D objects in the form of Point Clouds was developed. For this purpose, the architecture introduced by [4], consisting of an AE, which first converts the 3D objects into a latent representation, and a GAN, which generates new objects in the form of latent vectors, was extended by the specification of certain object properties. The functionality of the approach is validated in a series of experiments. With regard to the area of product development, the presented approach offers enormous potential. On the one hand new products within a product family can be generated according to a given product specification. In this way, implicit knowledge in the form of requirements (e.g. with regard to materials, production technology) is transferred to new product generations. On the other hand, when designing new products, various suggestions can be generated for the designer as to how new products could be designed based on an existing product family. Further investigations of how to transform more complex 3D data formats (e.g. polygon meshes) into latent representations can enable the use of latest GAN technologies even within CAD environments. However, to enable industrial use it is necessary to improve the implementation accuracy of the dimension specifications which are considered in this paper. As shown in section 4, the main dimension error source is given by the AE. Thus, further research could build on introducing 3D AEs generating less dimension errors during the encoding and decoding process. Since the applied architectures of both the AE and the GAN are quite simple, it may be possible to obtain improved performance through more complex networks.

Acknowledgements

This paper was also funded by the Federal Ministry of Education and Research (BMBF) project AIAx, Machine Learning-driven Engineering – CAx goes AIAx (01IS18048B).

References

References

[1] Albers, A., Behrendt, M., Klingler, S., Reiß, N., Bursac, N., 2017. Agile product engineering through continuous validation in PGE – Product Generation Engineering. Des. Sci. 3, 8.

[2] Albers, A., Haug, F., Fahl, J., Hirschter, T., Reinemann, J., Rapp, S., 2018 - 2018. Customer-Oriented Product Development: Supporting the Development of the Complete Vehicle through the Systematic Use of Engineering Generations, in: 2018 IEEE ISSE, pp. 1–8.

[3] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative Adversarial Networks.

[4] Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L., 2018. Learning Representations and Generative Models for 3D Point Clouds. 35th International Conference on Machine Learning (ICML).

[5] Mirza, M., Osindero, S., 2014. Conditional Generative Adversarial Nets.

[6] Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

[7] Schmidhuber, J., 2015. Deep Learning in Neural Networks: An Overview. Neural Networks 61, 85–117.

[8] Aivaliotis, P., Zampetis, A., Michalos, G., Makris, S., 2017. A Machine Learning Approach for Visual Recognition of Complex Parts in Robotic Manipulation. Procedia Manufacturing 11, 423–430.

[9] Pandremenos, J., Chryssolouris, G., 2011. A neural network approach for the development of modular product architectures. International Journal of Computer Integrated Manufacturing 24 (10), 879–887.

[10] Radhakrishnan, S., Bharadwaj, V., Manjunath, V., Srinath, R., 2018. Creative Intelligence – Automating Car Design Studio with Generative Adversarial Networks (GAN), in: Holzinger, A., Kieseberg, P., Tjoa, A.M., Weippl, E. (Eds.), Machine Learning and Knowledge Extraction, vol. 11015. Springer International Publishing, Cham, pp. 160–175.

[11] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F., 2015. ShapeNet: An Information-Rich 3D Model Repository.

[12] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3D ShapeNets: A deep representation for volumetric shapes, in: 2015 IEEE CVPR, pp. 1912–1920.

[13] Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E., 2015 - 2015. Multi-view Convolutional Neural Networks for 3D Shape Recognition, in: 2015 IEEE ICCV, pp. 945–953.

[14] Kanezaki, A., Matsushita, Y., Nishida, Y., 2016. RotationNet: Joint Object Categorization and Pose Estimation Using Multiviews from Unsupervised Viewpoints, 24 pp.

[15] Sedaghat, N., Zolfaghari, M., Amiri, E., Brox, T. Orientation-boosted Voxel Nets for 3D Object Recognition, in: Procedings of the British Machine Vision Conference 2017.

[16] Charles, R.Q., Su, H., Kaichun, M., Guibas, L.J., 2017 - 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, in: 2017 CVPR., pp. 77–85.

[17] Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space.

[18] Simonovsky, M., Komodakis, N., 2017. Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs.

[19] Hegde, V., Zadeh, R., 2016. FusionNet: 3D Object Classification Using Multiple Data Representations.

[20] Chen, Y.-C., Tan, D.S., Cheng, W.-H., Hua, K.-L., 2019. 3D Object Completion via Class-Conditional Generative Adversarial Network, in: Kompatsiaris, I., Huet, B., Mezaris, V., Gurrin, C., Cheng, W.-H., Vrochidis, S. (Eds.), MultiMedia Modeling, vol. 11296. Springer International Publishing, Cham, pp. 54–66.

[21] Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A., 2016. Learning a Predictable and Generative Vector Representation for Objects.

[22] Li, C.-L., Zaheer, M., Zhang, Y., Poczos, B., Salakhutdinov, R., 2018. Point Cloud GAN.

[23] Simonovsky, M., 2019. Deep Learning on Attributed Graphs: A Journey from Graphs to Their Embeddings and Back.

[24] Wang, Y., Xie, Z., Xu, K., Dou, Y., Lei, Y., 2016. An efficient and effective convolutional auto-encoder extreme learning machine network for 3d feature learning. Neurocomputing 174, 988–998.

[25] Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J., 2016. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling, in: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29. Curran Associates, Inc, pp. 82–90.

[26] Yang, Y., Feng, C., Shen, Y., Tian, D., 2017. FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation.

[27] Wang, X., Gupta, A., 2016. Generative Image Modeling using Style and Structure Adversarial Networks.

[28] Öngün, C., Temizel, A., 2018. Paired 3D Model Generation with Conditional Generative Adversarial Networks.