Title:
Comparative Analysis of Classification Methods for Real-world Personalized STL Models

Authors
Lovro Sever, lovro.sever@fsb.unizg.hr, University of Zagreb
Tomislav Martinec, tomislav.martinec@fsb.unizg.hr, University of Zagreb
Robert Mašović, robert.masovic@fsb.unizg.hr, University of Zagreb
Stanko Škec, stanko.skec@fsb.unizg.hr, University of Zagreb

Keywords:
Convolutional neural networks, STL model, Classification, Real-world dataset

DOI: 10.14733/cadconfP.2025.183-188

Introduction:
Nowadays, in the context of manufacturing personalized and distinctive products, STL files are widely used to facilitate the reliable and accurate transfer of 3D geometries from CAD modelling software to the CNC machines or 3D printers. The challenges of classification of such STL models lie in ensuring the efficiency and accuracy of methods for specific classification problems. Whereas numerous classification methods are available for wider application, 3D model classification largely relies on methods based on convolutional neural networks (CNNs) (e.g., [2, 3, 4, 7]), combined with different representations such as point clouds, voxels, or rendered 2D images from different view angles.

Studies that focus on investigating, developing, and testing neural network architectures for 3D geometry classification commonly rely on standard benchmark datasets like ModelNet10 or ModelNet40 (e.g., [3, 7]). While such datasets are appropriate for training and comparison, due to the large number of models across classes, they do not replicate real-world collections of models that usually appear across industrial projects. The product portfolio of many companies, particularly in the personalized products and prosthetics market, does not consist of geometrically distinctive classes of models that appear in benchmark collections, but rather different variants of a small set of products. These products are typically configured, customized, or personalized based on customer requirements. The practices of geometrical modeling and exporting STL models of product variants in an industry setting can vary significantly, meaning that preprocessing of models is commonly needed to harmonize the dataset prior to classification. Additionally, the size of real-world, that is, actual industry datasets in the context of personalized products, can be relatively small compared to the size of the benchmark collections, with uneven distribution of models across classes, which raises concerns about whether some of the proposed methods could be implemented for such limited datasets.

The goal of this paper is to analyze and compare the performance of different classification methods on a real-world STL model dataset and to identify the main challenges that arise when dealing with limited, unbalanced, and structurally similar data. The product selected for the study is a dental abutment – a connector piece that attaches a dental prosthetic, such as a crown, to an implant anchored in the jawbone. Dental abutments are usually personalized to fit each individual's unique dental structure and aesthetic needs, whereas their general shape depends on the position of the tooth. As a result, different classes of abutments (based on the tooth position) are commonly present, with personalized geometries within each class.

Background:
Understanding the context and the methods of STL model classification is important for positioning this study within the context of classifying real-world industrial datasets. This section shows some examples of how STL model classification can be implemented and represented. In particular, the application of four different classification neural network-based methods is covered: (1) PointNet [3], [9], (2) Volumetric CNN [7, 10], (3) Multi-View CNN [1, 2], and FusionNet [4]. In addition, this section reviews one of the available ways for comparing STL files [5, 6] and methods for translating STL models to other formats, such as voxel [8] and images [2].

One of the simplest ways to interpret STL models is as a set of points. Using this approximation, comparisons between two models can be made using measures for analyzing the differences between sets of points. For example, the Hausdorff distance between two sets of points can be used in shape matching or pattern recognition [6] or 3D object recognition (which can be applied to classification tasks) [5]. For calculating distance measures, the Euclidean distance is used, which is integrated into the algorithm's residue [6]. The same STL model representation can be used for PointNet, a pioneering method that uses unordered point clouds as input into a neural network [9]. This novel deep learning architecture for object classification or segmentation was developed by Qi et al. [9], whereas a similar method was used by Gu et al. [3], whose study focused on research methods that can be used on data sourced from lidar sensors. The proposed network architecture was tested on object-level datasets like ModelNet40, ScanObjectNN, and ShapeNetPart and validated on the scene-level dataset KITTI [3]. The architecture achieved a shape classification accuracy of 86.4% on the real-world ScanObjectNN.

While the PointNet method uses the point cloud as direct input into the CNN [9], the volumetric CNN uses a voxel matrix as input [10]. Decomposing STL models into a set of voxels allows for the description of the continuous model with minimal error [8]. Models generated in this way require large storage if one wants to replicate complex surface models with small error values. For example, models represented using 400x400x400 voxels require 64 MB of storage space [8]. Using an octree representation of voxel models, memory usage can be significantly reduced [7]. Examples of volumetric CNNs application are reported in Wang et al. [10] and Muzahid et al. [7], who tested their volumetric CNN architecture on the ModelNet benchmark dataset, achieving accuracy of up to 91.1%. Additionally, the accuracy improved by combining volumetric CNN and NormalNet architecture to 93.1%.

In addition to analyzing 3D models as point clouds or voxel representations, classification can also be performed using the Multi-View CNN [1, 2]. In this method, each view of the model is represented as a 2D image and is analyzed separately using a neural network. Results from all model views are combined and used for object detection or classification. Using this method, validation accuracy can reach up to 95% [2]. A similar network architecture was used for classifying CAD models, where 2D images are rendered from CAD models. When using a method similar to Multi-view, the validation accuracy of the MVCNN++ method can be as high as 95.45% [1].

The last reviewed network architecture is the FusionNet. To improve classification performance, a combination of voxel and 2D images can be used as input data [4]. ModelNet10 and ModelNet40 datasets were used for training and validation, whereas the obtained results demonstrate the importance of using multiple representations of a model to improve classification performance [4].

Most studies use standard benchmark datasets for testing network architectures, achieving validation accuracy of up to 95% [1]. From the analyzed papers, it is observed that there is a lack of testing different classification methods on real-world datasets, such as personalized products, where there is a shortage of models for training networks and smaller difference between classes.

Methodology:
To compare different classification methods for real-world STL files, a dataset of 1560 individual dental abutment models was used. The dataset was provided by a company that designs and manufactures dental prosthetics and other related products. The models from the dataset belong to seven classes, designated as tooth positions from 1 to 7, which correspond to the different positions of the corresponding teeth. This dataset is specific as the models, which nominally belong to the same class (designed for the same tooth position), can exhibit significantly different geometric characteristics. In

contrast, there are many cases where the difference between the two classes is not obvious when performing a visual inspection of geometry.

Examples of abutment models corresponding to each class are shown in Figure 1. The models on the top represent the characteristic geometries that are distinctive for a specific position. The models in the middle show the average geometries which are most common within a class. Finally, the bottom row shows examples of indistinctive models, with geometry that cannot be classified in a straightforward manner.

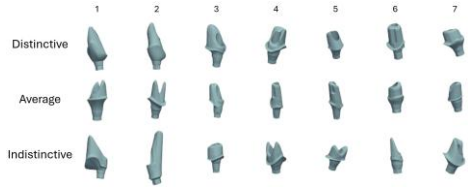

Fig. 1: Dataset sample.

Besides the differences in geometry, the collected STL files were generated by different individuals using several different oral 3D scanners and dental and general-purpose CAD tools, which resulted in inconsistent mesh resolution, origin positions, and model orientations. Therefore, before applying the different classification methods, the models first had to be realigned (preprocessed). To do so, the center point of the model bounding box was translated to the coordinate 0, 0, 0 (X, Y, Z), as shown in Figure 2, left. After the translation, each model was additionally rotated around the Z-axis to the position in which the Hausdorff distance between the reference model (manually selected example) and the aligned model was minimal. The next figure shows two STL files before (left) and after the alignment process (right).

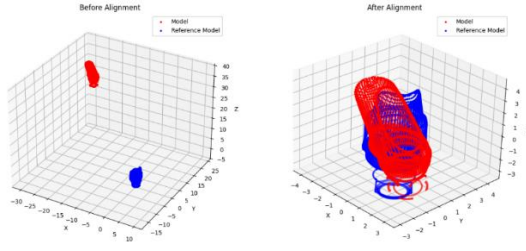

Fig. 2: Example of model realignment.

After the alignment, five different classification methods were applied to the dataset: (1) Hausdorff distance, (2) PointNet, (3) Volumetric CNN, (4) Multi-View CNN, and (5) FusionNet. As shown in Figure 3, three basic types of input data were used across these five methods: point clouds (1 and 2), voxels (3 and 5) and images (4 and 5).

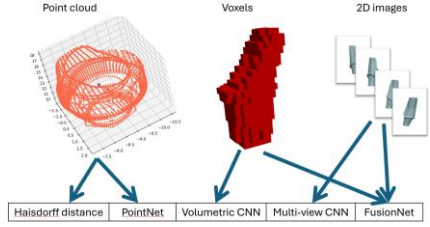

Fig. 3: Types of input data used in the classification methods.

Finally, the following three performance metrics were used for the purpose of comparing the five selected classification methods: (1) average classification time, (2) number of epochs, and (3) accuracy (inspired by the metrics used in studies [2], [3], [4]).

- **Average classification time** metric reveals how much time was needed for the classification of a single STL model (average time per model).
- **Accuracy** metric represents the percentage of properly classified STL models. Accuracy was calculated based on validation sample (models that were not used for training, which represent 20% of the overall dataset.
- **Number of epochs** metric reveals the number of training cycles required for the model's accuracy to converge for methods based on trained neural network.

Results:

The dataset was classified using the five methods, which were then compared using the three metrics. The performance of each method in terms of the average classification time in seconds, the number of epochs before the method converged (where applicable) and the classification accuracy is presented in Tab 1.

Classification using Hausdorff distance exhibits low accuracy (32.8%) but does not require training. The other methods require approximately two hours of training time. Namely, the training involved tuning of large number of parameters, such as batch size, number of network layers, division of training and testing sample, etc., and was performed using the following configuration: Intel Core I7 12700K, 32 GB of RAM, and NVIDIA RTX A2000 GPU. PointNet shows slightly better performance; however, the accuracy is still relatively low (38.8%). The best performance was achieved when using the FusionNet method with an accuracy of 62.6% and an average classification time of 0.73 s. When using only the Volumetric CNN method, the accuracy is slightly worse, but the average classification time is fairly lower.

	Hausdorff distance	PointNet	Volumetric CNN	Multi-View	FusionNet
Average classification time [s]	1.09	0.70	0.50	0.60	0.73
Number of epochs	-	11	21	16	11
Accuracy [%]	32.8	38.2	54.8	55.79	62.6

Tab. 1 Metrics values.

Results presented in Table 1 reveal information about the performance of each method, but do not provide insights into the geometric characteristics of incorrectly classified models. Therefore, an additional figure is shown, where one specific model example is presented for each class (tooth position), which was not correctly classified by none of the method (Fig. 4).

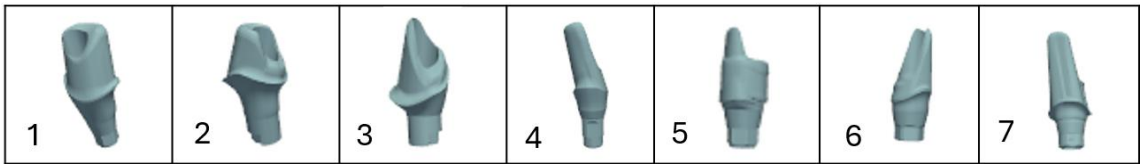

Fig. 4 Examples of models that were incorrectly classified by all methods and their corresponding correct classes.

Most models that were not correctly classified fit into the vague category of models introduced in Figure 1, meaning that they are challenging to classify based on visual inspection as well. For example, when

comparing the model that was not successfully classified as class 1 (the leftmost in Table 2) with models shown in Figure 1, it can be seen that this model is geometrically more similar to the distinctive model of class 6 rather than any model from class 1. Similar observations can be made for other examples. The most persistent classification mistakes by all tested methods were made in-between classes 6 and 7. Not only is their geometry visually similar, but they exhibit similarities with models from classes 1 and 2. After analyzing all incorrectly classified models, such as the examples from Table 2, it can be concluded that there were generally no problems with the classification of models that contain distinctive, class-specific features (e.g., the top row in Figure 1). Most of the misclassification cases were related to models containing vague features (bottom row in Figure 1), with some examples of "average" models as well (middle row in Figure 1).

Discussion and conclusion:

The study compares different methods for STL model classification on a dataset of 1560 STL models of dental abutments. This abutment dataset is specific compared to the typically used datasets in CAD model classification studies, as it represents a collection of models characterized by a low quantity and inconsistent distribution of models per class. As such, it mirrors the problem often found in industry – the lack of models, the high similarity expressed between different classes of models, and the limited computational resources. Therefore, the main goal of this study was to identify the challenges faced in addressing the classification of such datasets using well-established classification methods.

Through this comparative study, the usage of existing classification methods doesn't yield results with high accuracy. Better results could potentially be achieved by considering additional methods and additional preprocessing steps. The use of a classification method that combines voxels and 2D images gives better results (as demonstrated by the FusionNet method), and one possible way to improve accuracy is by using higher-resolution voxel models and images. This improvement presents challenges related to the large amount of data, which requires significant computational resources (especially memory) that are typically unavailable in industrial environments. For the same purpose, one must consider the use of additional input parameters and the detection of specific features related to the dataset, which can aid in classification.

Most industrial datasets require preprocessing, such as translation and rotation of models, repairing of meshes, or remeshing models. The influence of preprocessing on classification performance, such as accuracy, is not known for specific cases and needs to be investigated. An additional problem with the dataset is that the original STL models are generated in different CAD tools by various users, and these models differ in aspects such as resolution or mesh generation logic. This issue can influence results, especially for methods that use point clouds. The influence of all these input factors must be investigated. The last aspect to be considered is the capability of an expert to correctly classify STL models. Future studies could thus answer the following question: Can an expert, who creates these types of models, correctly classify a specific dataset, and what is the actual classification accuracy that we can expect?

Despite the identified challenges, this study has several limitations. The methods considered were tested on a single real-world dataset, meaning that the results could vary for other cases. Moreover, due to the scope of the paper, only five classification methods have been analyzed and tested. Further testing of new methods could result in significantly better performance and must be considered in future work. Also, as mentioned earlier, the resolution of the voxel and image model representations was relatively low, due to the limitations imposed by the computational resources. Future studies should investigate the resolution's effect on classification performance. The last limitation is closely related to the challenge of dealing with a real-world dataset, which in this case consisted of a relatively small number of models that are not uniformly distributed across classes. For example, one of the classes only had 50 models for network training and testing, while the others had up to 200 models. Hence, the effect of increasing the number of models and harmonizing their distribution across classes should also be considered in further studies.

Lovro Sever, https://orcid.org/0009-0000-0493-3439
Tomislav Martinec, https://orcid.org/0000-0002-6487-4749

Robert Mašović, https://orcid.org/0000-0002-0091-0480
Stanko Škec, https://orcid.org/0000-0001-7549-8972

Acknowledgement:
This research was funded by the project NPOO.C3.2.R3-I1.04.0121: Generative Design for Mass Personalization of Dental Implantoprosthetic Abutments (GENKON).

References:

[1] Angrish, A.; Bharadwaj, A.; Starly, B.: MVCNN++: Computer-Aided Design Model Shape Classification and Retrieval Using Multi-View Convolutional Neural Networks, Journal of Computing and Information Science in Engineering, 21(1), 2021. https://doi.org/10.1115/1.4047486

[2] Ghadekar, P.; Deshmukh, M.; Deshmukh, S.; Jangid, D.; Dewalkar, D.; Dighole, R.: 3D Image Classification Based on Multi View CNN Using 2D Images, Proceedings - 2023 3rd International Conference on Innovative Sustainable Computational Technologies, 2023. https://doi.org/10.1109/CISCT57197.2023.10351341

[3] Gu, L. et al.: PointeNet: A Lightweight Framework for Effective and Efficient Point Cloud Analysis, [Online], Dec. 2023. http://arxiv.org/abs/2312.12743

[4] Hegde, V.; Zadeh, R.: FusionNet: 3D Object Classification Using Multiple Data Representations, [Online], Jul. 2016. http://arxiv.org/abs/1607.05695

[5] Lin, X.; Zhu, K.; Wang, Q. G.: Three-Dimensional CAD Model Matching with Anisotropic Diffusion Maps, IEEE Transactions on Industrial Informatics, 14(1), 2018, 265-274. https://doi.org/10.1109/TII.2017.2696042

[6] Marošević, T. M.: The Hausdorff distance between some sets of points, [Online], 2018. http://www.mathos.hr/mc

[7] Muzahid, A. A. M.; Muzahid, A. A. M.; Wan, W.; Wan, W.; Hou, L.: A New Volumetric CNN for 3D Object Classification Based on Joint Multiscale Feature and Subvolume Supervised Learning Approaches, Computational Intelligence and Neuroscience, 2020, 2020. https://doi.org/10.1155/2020/5851465

[8] Patil, S.; Ravi, B.: Voxel-based Representation, Display and Thickness Analysis of Intricate Shapes, 2005.

[9] Qi, C. R.; Su, H.; Mo, K.; Guibas, L. J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation.

[10] Wang, C.; Cheng, M.; Sohel, F.; Bennamoun, M.; Li, J.: NormalNet: A voxel-based CNN for 3D object classification and retrieval, Neurocomputing, 323, 2019, 139-147. https://doi.org/10.1016/j.neucom.2018.09.075