COB-2023-1847
MECHANICAL COMPONENTS RECOGNITION THROUGH COMPUTER VISION USAGE

Leonel Fernandes Balbino
Carlos Alberto Fiakofski Cadamuro
Giuliana Sardi Venter
Federal University of Paraná
leonel.balbino@ufpr.br
carlos.cadamuro@ufpr.br
giuliana.venter@ufpr.br

Abstract. *The current work is motivated by the investigation of the feasibility of extracting geometric characteristics from 3D models in .obj format so that they can be used in the segmentation of images of real objects. The data originally form a set with 58,000 models grouped into 68 classes. Starting from the original set, extracting 12 projection views of each model, 6 data sets were generated: 3 with white background with resolutions of 64x64, 256x256 and 1024x1024 and another 3 datasets containing the same images as the previous ones, but with the addition of background images. The 6 datasets were used to verify the accuracy gain in training with larger images and to verify the impact of the background variance in segmentation. Initially the classification was made to validate whether the classes are well separable and then the segmentation of only one of the classes was performed. As a result, the ability of the model to separate classes was proven, with an average validation accuracy of 94% in training with 256x256 images. The results obtained in the segmentation indicate that the model cannot segment the objects because the texture of the models is vastly different from the texture of the real objects, and this makes the geometric characteristics have little relevance in the training. This limitation indicates that an improvement in the results must be achieved by improving render quality of the 3D models views.*

Keywords: *convolutional neural networks, semantic segmentation, computer vision*

1. INTRODUCTION

A major challenge in image processing work is the difficulty in obtaining quality databases for specific applications. Based on this limitation, the opportunity arose to investigate the ability to obtain geometric characteristics of models from mechanical design software so that it is possible to use them to identify real mechanical components.

One of the motivations for the authors of the work was the difficulty of finding databases with large-scale mechanical parts for computer vision implementations in manufacturing and supply chain applications, considering that mechanical components have extremely specific geometric characteristics and are not found in libraries already trained with recognition of these features.

Image classification is one of the first tasks that human beings begin to learn shortly after birth. The foundation of this activity is based on the recognition of geometric shapes and patterns around them. In Computing, problems related to image identification can be approached in diverse ways, but it was with the advent of convolutional neural networks that error rates in image classification problems were significantly minimized (Jogin, 2018).

The usage of techniques of automating image classification and segmentation is present in many applications and fields in the industry such as the identification of the correct usage of personal security equipment, identification of diseases in image exams, quality control and self-driving cars.

Convolutional neural networks (CNN) are deep networks whose premise is that the input is an image, which allows the architecture of the next layers to be specific for the treatment of this type of data and its derivatives. An important question related to the type of input is the size that a network can reach when medium-sized images are used. Taking, for example, a conventional neural network built for the classification of images with 3 color channels and 300x300 pixels of resolution, the network will have in its first layer 300*300*3 = 2700000 neurons, which makes it difficult for the network to scale to large images or many inner layers. The architectural advantage of a convolutional neural network is the possibility of extracting features from pieces of the image so that in some way it is feasible to reduce the dimensionality of the input image and consequent smoothing of the exponential behavior of the network growth when working with exceptionally large input data. The components of a convolutional network are basically convolutional layers, activation functions, pooling layers responsible for summarizing features of small regions of the image and fully connected layers. Convolution layers are the main component, even naming this type of neural network architecture. It is always the first layer and can be repeated throughout the process with the purpose of recognizing patterns or sets of properties of the image received as input (Bouti, 2020). Activation functions are responsible for adding nonlinearities to the outputs of the nodes of a layer so that the next layer receives a transformed input (Sharma, 2017). Pooling layers aim to reduce the image resolution by aggregating small sets of pixels summarizing the region to an average of the contained elements or, more often, to their preponderant value in order to maintain prominent features in the region (Scherer, 2010). Completely connected layers, constituents of conventional neural networks, are the densest in terms of parameters. As the name suggests, all nodes of a layer connect with all nodes of previous layers and each of these connections represents a weight or parameters of that network (Basha, 2020).

Finally, an important element of neural network training within the scope of this work is the technique known as transfer learning. It consists of using the weights obtained in training the network with images from other data sets to simplify the training on the set of interest, transferring the learning from one domain to be applied in another.

Chen et al., 2020, propose a model to separate silicon wafers, used in chips manufacturing, into for classes: 3 representing different defects and the last one representing the material suitable for manufacturing (Figure 1). Using data augmentation techniques, the authors were able to come up with an architecture that although slightly less accurate than VGG-16 has optimizations to run 6 times faster which represents a significative improvement in the use case.

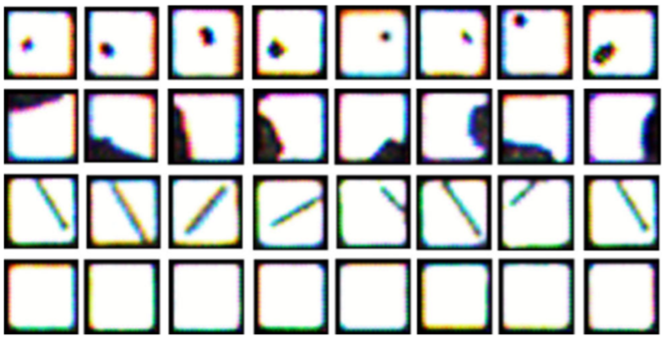

Figure 1. Sample of the four classes in Chen et al.

Another work describing the suitability of CNNs into manufacturing industry is presented in Lin et al., 2019 by conducting an approach to automating detection of 2 kinds of defects in LED components along with the segmentation of defects region into each frame with accuracy of 95% (Figure 2).

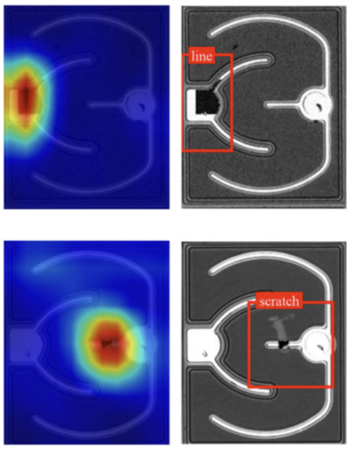

Figure 2. Segmentation of LED components manufacturing defects.

2. METHODS

The data set was built from the work elaborated in Kim et al., 2020 in which the authors generated from the combination of models in 3 databases (GrabCAD, 3D warehouse and TraceParts). This set of 3D models of mechanical components with a total of 163216 parts which, after being filtered to remove duplicate or invalid models, ended up with 58696 parts divided into classes such as bushings, screws, nails, exhaust fans, pumps, etc (Figure 3). For the recognition model to be able to interpret this data set, each 3D model was unfolded into 12 images of orthogonal planes through a Python script using the Pytorch3D library.

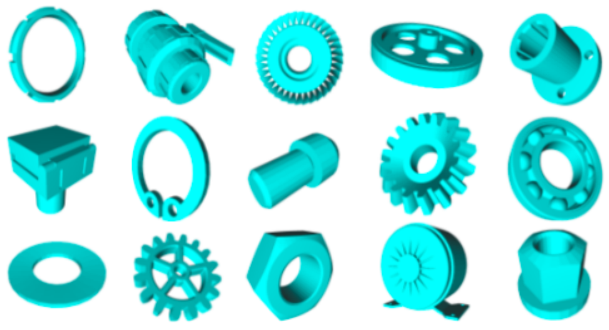

Figure 3. Sample of the models present in the data set.

Firstly, it was observed that for the correct use of Pytorch3D, the library responsible for handling the technical drawings, it was necessary to change the 3D models originally in an .obj format so that they would receive a property that would allow the rotation of the part's points of view so that it was possible to extract the images of the parts at different angles to feed the model. For this change, a Python script was executed that iterates through all the models, checking if the property composed by the mtllib and usemtl directives, which assign a texture to the model, is present and, if not, inserts the property, closes the file and proceeds to the next.

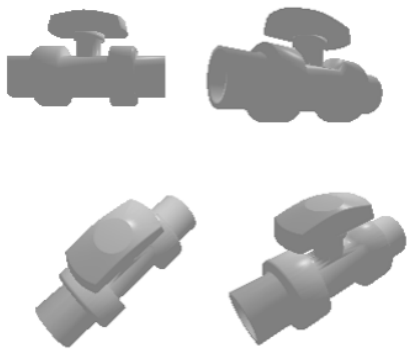

Figure 4. 4 projection views for a valve

Figure 4 brings four examples of views of a valve contained in the models that generated the data set.

A check regarding the positioning of the models in relation to the origin of the orientation system was necessary to verify that the images of the views were representative of the parts, but given the high number of models, the simple visual inspection would have little statistical power. Tests were conducted to sample these models and verify their positioning to statistically guarantee that, when the images were generated, they would all have relevant content for the classification model. To increase data variability, it is possible to generate as many views as necessary considering the three-dimensional nature of the source dataset. In this case, 12 views of each modeled part were selected, taking a set of approximately 58,000 models to something around 700,000 images. The next step is to separate the images into training and test sets with approximately 75% of the models belonging to the training data. For this, a random selection was made and the names of the models saved to be used as indexes for the data. Using the methodology described for the generation of datasets, 3 datasets were generated having the same views for all models: one of images considered small with 64x64 pixels, a second with medium images having 256x256 and the last one with large images having 1024x1024 pixels as dimensions. Still in the image manipulation stage, 10 images with the theme of a mechanical workshop and scrap metal were selected (Figure 5) to serve as a background for the evaluation of the results, contrasting the models trained with images of white background and models trained with images with a complex background. For this, the white backgrounds of the views of the models were removed so that transparent background images were obtained and then the views were pasted onto randomly selected backgrounds as seen in Figure 6.

Figure 5. Examples of background images.

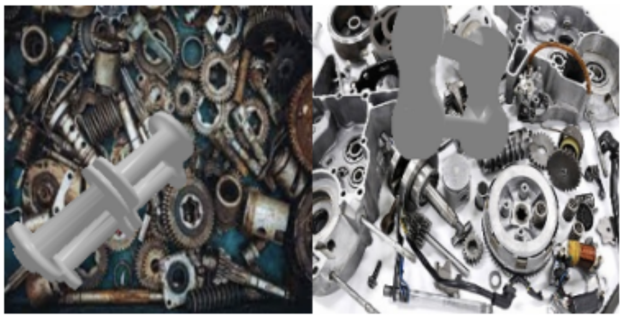

Figure 6. Examples of images of models placed into the backgrounds.

Next, the classification model with transfer learning to be used for data validation was selected. It was decided to be a resnet152 pre-trained with the Imagenet dataset to have its weights to be used in the mechanical components' images training. At the end of training, the model was evaluated by predicting the validation data, already classified, and comparing the classification obtained by the model and the real classification, already known, using accuracy as an evaluation method.

After validating the ability of the data to be well separated into its 68 classes, a segmentation model was also trained with transfer learning, seeking to segment only one of the classes for reasons of simplicity at first. In this case, the chosen class was chain links. Unlike the classification model that observes an image and outputs a class for the image, the segmentation model in this work delivers a matrix with the same dimensions as the input image with the probability that each pixel contains the object of interest.

The system used to prepare the images and run the training consists of an Intel Core i9 13900k processor with 32 threads, 64 GB of DDR5 RAM memory and a GeForce RTX 3070 video card with 8 GB DDR6. The system specification can be useful for verifying the relevance of video processors in computer vision tasks briefly discussed in the final section of this work.

3. RESULTS

After processing the data to generate the datasets, the results of this work were obtained considering two perspectives: the ability of the dataset and the classification model to separate the 68 classes contained in the dataset and the ability of a segmentation model to point out where the objects are in the image.

Starting with the classification models, tests were performed with the images on white background with ResNet152 since it was the most powerful architecture among those available in the Pytorch library that provides an ecosystem for training neural networks in python. With images of 64x64 pixels, the average accuracy obtained was 76% with one training run and 90% for training until convergence, which took about 11 hours. As expected, training the same architecture, but with medium-sized images, led to a more accurate result, converging in something close to 13 hours of training for 94%. Finally, in the case of large-size images, training proved to be unfeasible given that the low availability of video memory in the system for images of this size meant that the estimated time until convergence was around 40 hours without a guarantee of significant improvement in the performance obtained with the images of 256x256 pixels. Considering that the dataset is composed of 68 classes with an important level of imbalance, a more relevant measure of the model's predictive capacity is the accuracy per class. In the tables, the values of the classes with greater and lesser accuracy obtained in the set of medium-sized images and the accuracy histogram.

Table 1. 10 classes with lower accuracy score.

Category	Correct count	Total count	Accuracy
Spacers	9	260	0.034
Nozzle	181	352	0.514
Clamps	191	369	0.518
Turbine	106	201	0.527
Hook	165	275	0.600
Collars	76	119	0.639
Impeller	225	344	0.654
Spring washers	91	131	0.695
Hinge	84	120	0.700
Valve	158	215	0.735

Table 2. 10 classes with higher accuracy score.

Category	Correct count	Total count	Accuracy
Plain guiding	106	107	0.9906
Washer bolt	2139	2159	0.9907
Studs	9623	9702	0.9918
Screws and bolts with hexagonal head	16643	16743	0.9940
Conventional rivets	9006	9043	0.9959
Keys and keyways, splines	11708	11716	0.9993
Square	168	168	1.0000
Chain drivers	239	239	1.0000
Castle nuts	535	535	1.0000
Bearing accessories	250	250	1.0000

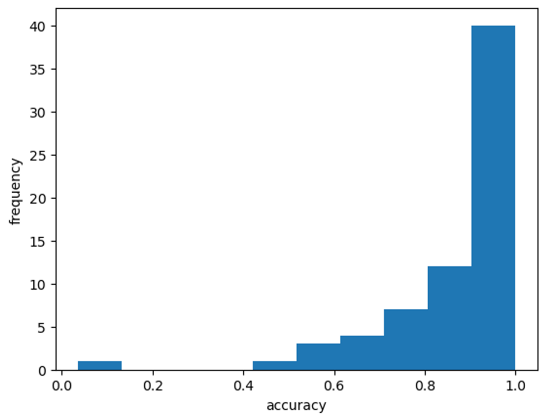

Figure 7. Histogram of accuracy for the 68 classes.

A relevant factor during model training does not seem to be directly related to accuracy: the number of training images. However, even though the linearity between the two variables is not observed, it is observed that all classes listed as those with lower accuracy have a reduced number of samples. Observing the results obtained in the classification stage (tables 1 and 2), it is clear that the model is sufficient to separate the vast majority of classes well considering that of the 68, only 8 have an accuracy of less than 70%. Thus, this can be considered a high-quality dataset for computer vision tasks (Figure 7).

The images presented in the results section are divided into 2 types: histogram of probability thresholds and image segmentation against real images. The threshold histogram shows the probability distribution of each pixel having part of the component of interest. Ideally, the histogram of the thresholds should contain a large volume close to 0 and another significant volume above 80% indicating that the model "is sure" where the component is and where the component is not (Figure 8).

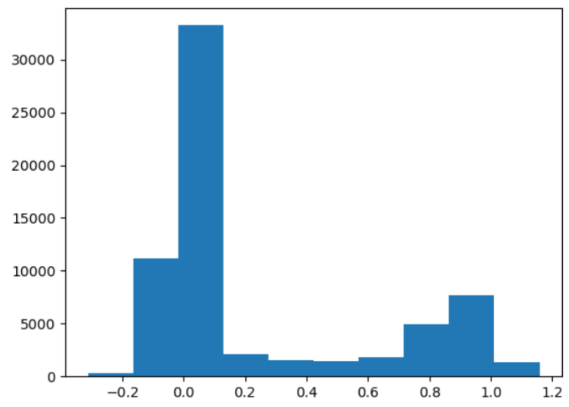

Figure 8. Example of probabilities histogram from segmentation

Histograms with distributions that are too flat or too concentrated around 0 indicate an inability of the model to draw the boundaries that define the component in the image. The frames of the opposing images represent purple for no and yellow for yes depending on the threshold value accepted as being “sure enough” of the model regarding the probability of the presence of the object. For example, a segmentation image with a threshold of 0.2 marks as purple all pixels that were classified with less than 20% probability and marks as yellow those pixels with a probability greater than or equal to 20% of containing an object of interest. Thus, the higher the threshold, the fewer pixels marked as yellow, but those marked as yellow have a greater probability of containing the object to be segmented. This relationship can be well exemplified in the comparison made in Figure 16.

From the point of view of image segmentation quality, 2 cases were taken: one class among those listed with the highest accuracy with white background and the same images after pre-processing to add a background image. In this case, 256x256 pixels of components of the chain drives class of the dataset are also taken. The selected segmentation model was the deeplabv3_resnet101 provided by Pytorch, which outputs the probability that each pixel contains part of the component used for training.

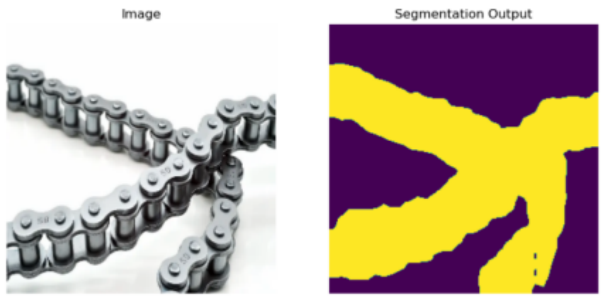

Figure 9. Segmentation of a transmission chain in white background by a model trained with images of white background.

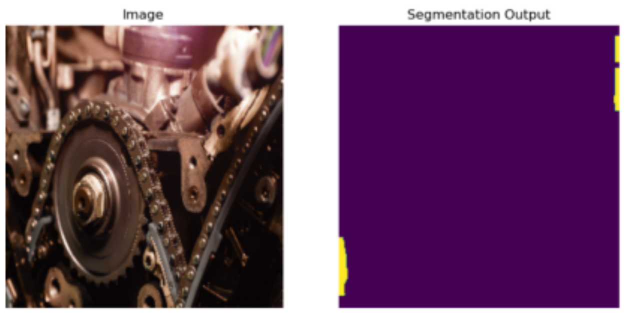

Figure 10. Segmentation of a transmission chain in complex background by a model trained with images of white background.

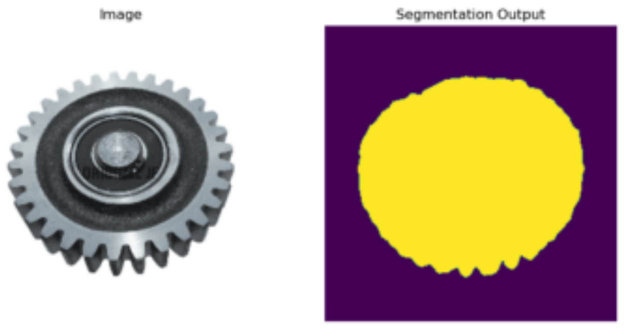

Figure 11. Segmentation of a gear in white background by a model trained with images of white background.

At first, Figure 9 appears to have a good ability to segment the object of interest in white background although Figure 10 shows chains in complex background are not segmented. The comparison between Figure 9 and Figure 11 shows that the model trained to recognize transmission chains also segments a gear. The most probable cause for this unwanted behavior is already a well-known problem in computer vision. Since only images on a white background are used in training, the model assigns a lot of weight to the contour variation, that is, the model trained on images with a white background learns to differentiate anything that is not a white background. To overcome this limitation, a first approach is to add diverse backgrounds to the training images so that the model incorporates the context in which these images appear. This strategy increases the ability to identify elements on complex backgrounds, while decreasing the ability to identify objects on a white background, since now the weight of simple differentiation of white or not is much less significant.

The observation of Figures 12 and 13 indicates that in fact the model loses ability to segment white background images when trained with images of complex background, but the expected gain in the ability to segment the current in complex background images is not observed.

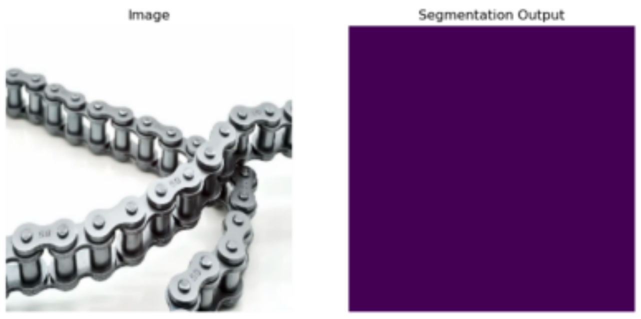

Figure 12. Segmentation of a transmission chain in white background by a model trained with images of complex background.

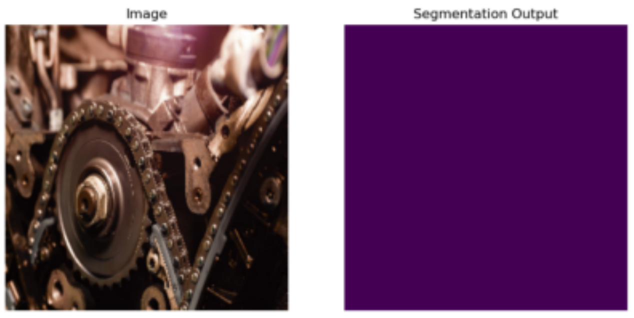

Figure 13. Segmentation of a transmission chain in complex background by a model trained with images of complex background.

Considering that the segmentation model was trained with the complex background images exemplified in Figure 6 with high accuracy, the suspicion of the difficulty in segmenting real images is the lack of texture in the training images.

Another aspect to be considered for segmentation evaluated when observing Figures 12 and 13 where the model simply does not identify a highlighted chain in the image. The cause for this nonconformity is the difference in texture between the 3D models and the real images. This indicates that only the geometric characteristics of the drawings are not enough for the model to learn to identify real objects.

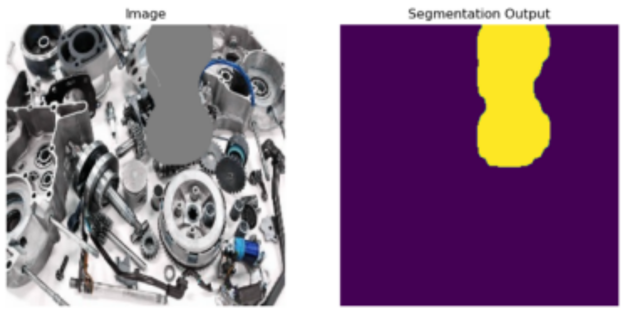

Figure 14. Segmentation of a chain driver in complex background by a model trained with images of complex background.

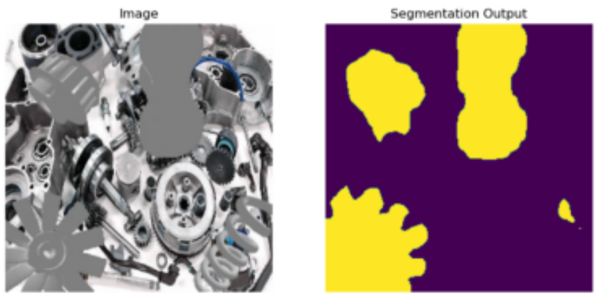

Figure 15. Segmentation of a chain driver, a turbine, a helix and a spring in complex background by a model trained with images of complex background.

A final analysis regarding the object segmentation capability of the model can be initiated by observing Figure 14, which indicates that the segmentation model accurately identifies the outline of a chain link in a complex background, but the observation of Figure 15 shows that the segmentation model trained to recognize only chain links, also segments all other objects. This indicates that the problem identified in the non-segmentation of real chains in Figures 12 and 13 is caused by the texture difference between the dataset models and the real images. This texture difference causes the model to interpret the texture as more relevant than the geometry.

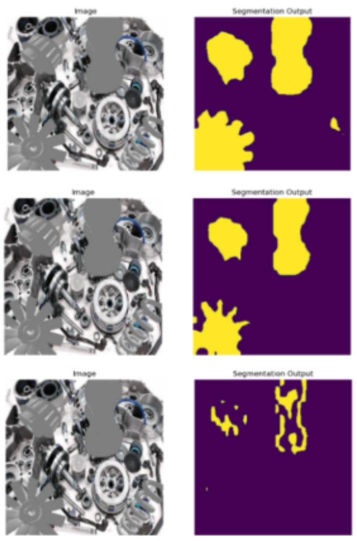

Figure 16. Segmentation of objects present in Figure 11 with probabilities limits of 0.2, 0.7 and 0.99 from top to bottom.

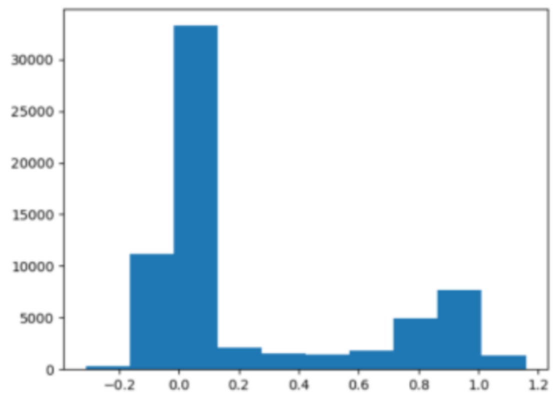

Figure 17. Probabilities histogram for segmentations in Figure 16.

The analysis of Figures 16 and 17 corroborates the conclusion that the model does not separate objects by geometry but based on texture. As the threshold increases, the segmentation tends to be more accurate in identifying the chain link, but only at a probability value of 0.99.

4. CONCLUSIONS

The obtained results indicate that the data set is indeed valid for computer vision applications because from the geometric point of view, the classes are well separable. The evaluation of the segmentation results indicates that the object images generated from the 3D models are not sufficient for the identification of real objects because they have vastly different textures. With that in mind, the next step in the evolution of this work so that the segmentation occurs as expected is to render the 3D models so that the textures approximate the textures of real objects.

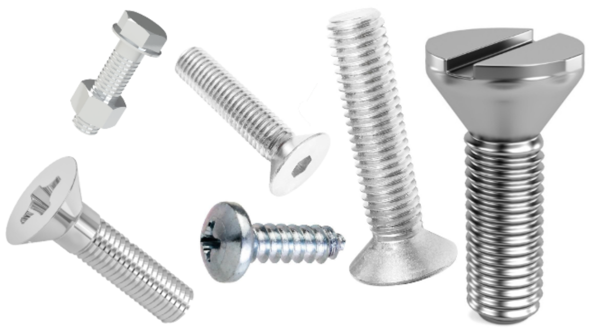

Figure 18. 3 pictures of screws and 3 rendered screws.

Figure 18 exemplifies the level of detail to be achieved by rendering 3D models. The volume of data makes manual rendering of objects in the dataset unfeasible, so it is necessary to find a library or interface with some modeling software that enables the automation of rendering with the models contained in the dataset.

5. ACKNOWLEDGEMENTS

We would like to thank everyone involved in the development and publishing of Kim et al. (2020) dataset.
Carlos A. F. Cadamuro acknowledge the support of CAPES - Coordination for the Improvement of Higher Education Personnel for their scholarship.
To the Brazilian National Council of Scientific and Technological Development - (CNPq), project 408080/2022-1.

6. REFERENCES

Basha, Sh Shabbeer et al. Impact of fully connected layers on performance of convolutional neural networks for image classification. Neurocomputing, v. 378, p. 112-119, 2020.
Bouti, Amal et al. A robust system for road sign detection and classification using LeNet architecture based on convolutional neural network. Soft Computing, v. 24, n. 9, p. 6721-6733, 2020.
CHEN, Xiaoyan et al. A light-weighted CNN model for wafer structural defect detection. IEEE access, v. 8, p. 24006-24018, 2020.

Jogin, Manjunath et al. Feature extraction using convolution neural networks (CNN) and deep learning. In: 2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT). IEEE, 2018. p. 2319-2323.

Kim, Sangpil et al. A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16. Springer International Publishing, 2020. p. 175-191.

LIN, Hui et al. Automated defect inspection of LED chip using deep convolutional neural network. Journal of Intelligent Manufacturing, v. 30, p. 2525-2534, 2019.

Scherer, Dominik; Müller, Andreas; Behnke, Sven. Evaluation of pooling operations in convolutional architectures for object recognition. In: Artificial Neural Networks–ICANN 2010: 20th International Conference, Thessaloniki, Greece, September 15-18, 2010, Proceedings, Part III 20. Springer Berlin Heidelberg, 2010. p. 92-101.

Sharma, Sagar; Sharma, Simone; Athaiya, Anidhya. Activation functions in neural networks. Towards Data Sci, v. 6, n. 12, p. 310-316, 2017.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.