RecMVCNN - Simultaneous Classification and Reconstruction

Bastian Wittmann
Technical University of Munich
bastian.wittmann@tum.de

Andrew Desousa
Technical University of Munich
andrew.desousa@tum.de

Florian Donhauser
Technical University of Munich
florian.donhauser@tum.de

Abstract

3D content can be encoded in many different representations, including point cloud representations, occupancy grids, and meshes. To classify the 3D content, different representations usually require specific classification networks tailored to the respective input data structure. By rendering multi-view images from the 3D representation, this issue can be circumvented and the same network, utilizing the rendered multi-view images as input data, can be used for the task of classification for many different input data structures. A prominent approach that utilizes rendered multi-view images as input data for classification is MVCNN[8], which serves as a baseline for our method. In this work, we propose RecMVCNN, a lightweight model for classification and shape reconstruction that extends MVCNN with a reconstruction head. Furthermore, experiments regarding the influence of the reconstruction task on the classification task were conducted.

1. Introduction

The incorporation of the convolution operation into neural networks for computer vision tasks was a breakthrough that demonstrated the possible potential of these then novel techniques. Since then, convolutional layers have gained immense popularity in the 1D, 2D, and 3D domain and are an essential part in many modern neural network architectures. Especially in the 2D domain, convolutional neural networks can be leveraged in order to create powerful feature extractors. This is due to the extensive amount of labeled training data available. The ImageNet[3] dataset, for example, contains more than 14 million labeled images, which enables 2D convolutional networks to counteract the issue of overfitting and allows for great generalization during training.
Even though convolutions can also be carried out in the 3D space to process 3D data, a promising approach to classify 3D geometries relies on multi-view images, rendered based on the target object, as input data. By incorporating 2D convolutional networks to process the rendered images of the 3D geometry, we can also leverage the expressiveness of pre-trained convolutional networks for feature extraction. This in turn leads to architectures performing exceptionally well on the task of classification.
In this work, we therefore try to improve the performance of the multi-view convolutional neural network[8] approach by incorporating a reconstruction proxy loss. Hence, our approach is capable of classifying a 3D geometry by its corresponding multi-view images and can simultaneously reconstruct the 3D geometry in the form of a dense voxel representation. In this regard, we tried to keep the model as lightweight as possible so that it can be trained in a reasonable amount of time on a single GPU. In summary, the main contributions of our work are:

- Incorporating a reconstruction head to the baseline model
- Reducing the parameters of the baseline model
- Testing our method on data from different domains

The code for this project is available at https://github.com/floriandonhauser/ML3D-MVCNN.

2. Related Work

Multi-View Classification. Our work is based on the model described in ”Multi-view Convolutional Neural Networks for 3D Shape Recognition” (MVCNN) by Su *et al.* [8]. The original MVCNN architecture classifies a 3D shape model based on a set of rendered 2D images, which are processed by 2D convolutional networks to extract rich feature maps. A big advantage of this approach is that the model can exploit the expressiveness of powerful pre-trained models like VGG-M[7]. Furthermore, due to the view pooling

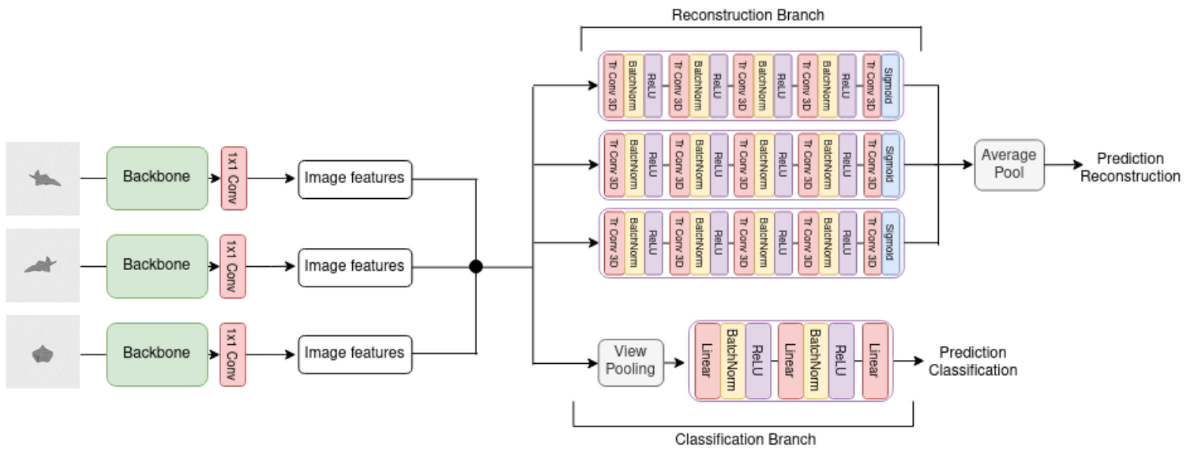

Figure 1. Architecture of our approach.

operator, the model can take an arbitrary number of images from different camera angles as input. In the original paper, options between 1 and 24 views were tested.

Later work by Qi *et al*. [6] reviewed the importance of the rendered 2D images by comparing the rendered meshes to renderings of spheres. They found that the resolution of the voxel representation, rendered with spheres, has an influence on the accuracy of a multi view approach. When the resolution was lowered the achieved accuracy also decreased.

Multi-View Reconstruction. Multiple 2D images can also be used for the reconstruction of a 3D shape. There are multiple approaches to this challenge, one being the usage of Recurrent Neural Networks (RNN), e.g. the model 3D-R2N2 developed by Choy *et al*. [2]. The RNN allows the model to use an arbitrary amount of input images and each additional image is supposed to refine the predicted 3D volumetric occupancy grid.

Another approach is used by the Pix2Vox model from Xie *et al*. [12], which utilizes an encoder-decoder architecture to generate coarse 3D volumes for each input image. The encoder contains a pre-trained VGG-16 model. Afterwards, coarse volumes are fused with the help of a multi-scale context-aware fusion module to produce a fused volume, which is then refined to generate the final output. Instead of using the different views one after another like in an RNN based approach, this model can create the different coarse volumes in parallel. It also allows for an arbitrary amount of views to be used.

Instead of predicting 3D volumetric occupancy grids, a model can also be trained to predict meshes which can enable higher resolution outputs. The work Mesh R-CNN by Gkioxari *et al*. [4] is a model that outputs coarse voxel representations only as an intermediate step and afterwards converts them into meshes which are also further refined. In contrast to our work, their experiments focused on single view reconstructions of meshes.

3. Methodology

Our architecture RecMVCNN is based on the original idea of MVCNN and takes multiple 2D views of a 3D shape as input. The same pre-trained CNN backbone is used on each input image followed by a 1*1 convolution to reduce the number of channels to 64. These image features are then used for two tasks, the classification, as performed with the original MVCNN, but additionally also a reconstruction of the 3D object for a 32^3 voxel grid. The classification is performed using view pooling followed by 3 linear layers. The reconstruction is performed using the reconstruction branch, which creates a 3D reconstruction for each image. These outputs are then combined using average pooling to get the final predicted 3D shape. The detailed structure of our method can be observed in Fig.1.

To test the model, we use two versions of the ShapeNet dataset[1], both with corresponding voxel representations [11]. First, we try a version which contains 24 RGB views of rendered 3D objects with a resolution of 137*137[10]. We use a subset of ShapeNet consisting of 13 major categories and 39,406 examples and a train-validation-test split of about 78%, 11%, and 11%. The dataset can be downloaded from the Stanford University Computational Vision & Geometry Lab's website. It contains pictures for 13 different classes and the virtual camera is rotated horizontally around each object to create the 24 views. On its own, this data can be used to classify the objects into the 13 different categories as is done with the original MVCNN architecture. To make the reconstruction head work, we use a version of the same ShapeNet dataset which contains voxelized models of the objects with a resolution of 32^3 which has been published by the lab as well.

Method	Backbone	#Views	Cls. Weight	Rec. Weight	Classification Acc.	Reconstruction IoU
ShapeNet Mesh Renderings Test Set						
RecMVCNN	MobileNetV3s	3	1	0	94.52	-
RecMVCNN	ResNet-18	3	1	0	94.79	-
RecMVCNN	ResNet-18	3	0.5	0.5	94.77	39.20
RecMVCNN	ResNet-18	3	0.2	0.8	94.88	40.04
RecMVCNN	ResNet-18	3	0.05	0.95	94.88	40.89
RecMVCNN	ResNet-18	3	0	1	-	41.17
ShapeNet Point Cloud Renderings Test Set						
RecMVCNN	ResNet-18	3	1	0	64.30	-
RecMVCNN	ResNet-18	3	0	1	-	14.62

Table 1. Quantitative results. We compared different configurations for the task of classification and reconstruction.

The second version of the dataset contains renderings of point clouds. Compared to the mesh renderings, they offer less details and no color information. We created the dataset ourselves with Open3D's[13] rendering API. We generate the renderings by rendering three evenly spaced views of an object rotated around its y-axis. Point cloud models were taken from an existing repository[9]. For configuring the resolution of the spherical points as well as their sizes, we visually inspected the renderings and decided upon parameters that best represented the models in ShapeNet.
First, we tried out the performance using only the classification task. For the backbone, we tested different pretrained CNNs and compared their performance. We decided to stick with ResNet-18[5] for later experiments as it performed the best.
Next, we included the task of reconstructing the voxelized 3D shapes. To combine both loss functions, we calculate each loss individually and weight them. The experiments with only classification are equivalent to weights 1 for classification and 0 for reconstruction. We further conducted experiments aimed to show the influence of those weights.

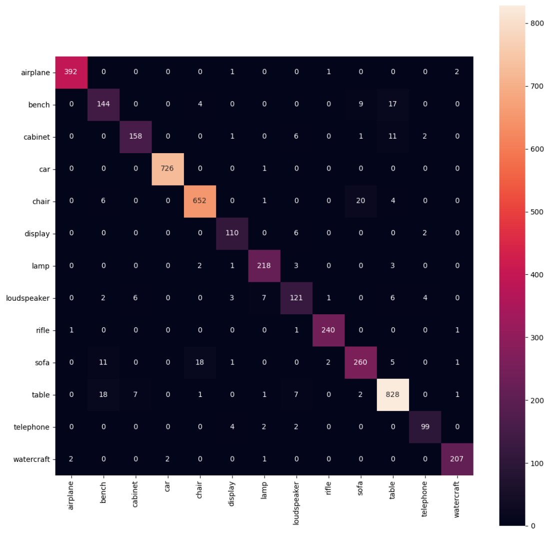

Figure 2. Confusion matrix for RecMVCNN with a ResNet-18 backbone and a 0.2/0.8 loss weighting.

4. Quantitative Results

To evaluate the performance across the different versions of our network, we use classification accuracy and reconstruction IoU. We performed a manual grid search to find the best learning rate and use early stopping to diminish the influence of overfitting. The best learning rates were determined to be $5 * 10^{-5}$ for ResNet-18 and $5 * 10^{-4}$ for MobileNetV3s. Nearly all configurations that we evaluated on the ShapeNet mesh renderings test set resulted in close to 95% classification accuracy. However, using a MobileNetV3s backbone results in slightly worse results.
Analyzing the incorrect classification predictions from our model evaluations with a confusion matrix depicted in Fig.2 shows that RecMVCNN struggles the most in distinguishing between objects which have similar parts. For example, airplanes, which are unique with respect to the other classes, would be easy for our models to distinguish. On the other hand, benches would be misclassified more often and would often times be confused for sofas and chairs due to each of the classes having common parts like seats and legs. For a classification only evaluation of RecMVCNN, 25 out of 26 misclassifications on bench examples were labeled as other classes of objects containing legs and seats such as benches and chairs.
For analyzing the reconstructions, we can utilize the IoU scores that were achieved on the test set. One interesting thing worth noting is how the model does better for certain classes than others. For the top 100 IoU examples evaluated on RecMVCNN with a 0.2/0.8 classification reconstruction weighting, 31 of them were cars and 29 of them were telephones, which is a disproportionately high amount of examples when compared to the overall distribution of

the classes. On the other hand, sofas, benches, and chairs had 2, 0, and 0 respective examples in the top 100 IoU evaluations. In general, the more weight was given to the reconstruction loss, the better the performance on the IoU score. We also evaluated our model on point cloud and incomplete point cloud inputs. Incomplete point clouds were simply the original point clouds cut by a plane. In general, these evaluations did significantly worse, likely due to the loss of information in the point cloud renderings. For classification, we were able to achieve 64.30% classification accuracy. For reconstruction, we could only achieve an IoU of 14.62, which is less than half of the quality that we were getting on the original renderings.

5. Qualitative Results

The reconstruction results for many objects represent the objects semantics fairly well. However, it does not work equally well for all objects. Thin cylindrical parts for objects like lamps often failed to be reconstructed as a well-defined voxel grid representation and are usually just ignored. This can also be observed in Fig.5. The wings of the plane or the legs of the chair are reconstructed only very poorly.

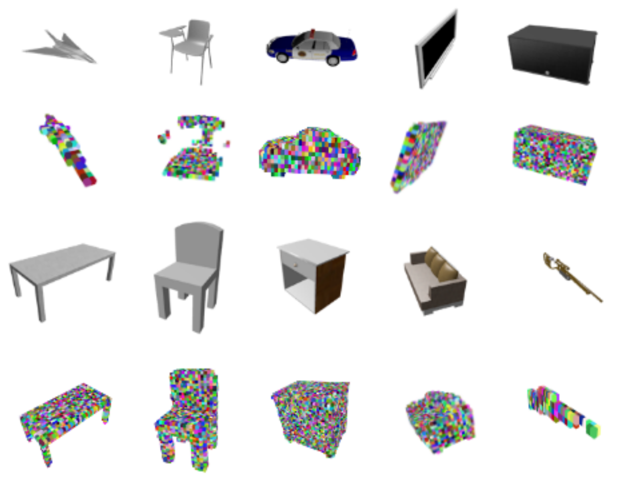
Figure 3. Examples of reconstructions from various classes. The first and third row represent single views of mesh models, while the other rows represent voxel grid reconstructions from RecMVCNN.

6. Ablation Studies

Backbone Networks. We experimented with multiple different pre-trained backbone options, which we received from the torchvision library. Our focus, in this regard, lied on a trade-off between performance and the number of trainable parameters. Experiments on a small part of the dataset showed that ResNet-18 and MobileNetV3s best fulfilled our requirements. Since ResNet-18 slightly outperformed MobilenNetV3s, we conducted further experiments with ResNet-18.

Concatenate Classification Results. To improve the performance of our model for the reconstruction task, we concatenated the classification scores to the input feature list of the reconstruction branch. This additional information appears to be useful for the reconstruction task and increased the IoU score by 0.4 %.

Regularization. We experimented with dropout as well as batchnorm layers for the regularization of the classification branch. Experiments showed that batchnorm layers worked slightly better.

7. Conclusion and Future Work

RecMVCNN is a lightweight neural network architecture that can be used for classification and reconstruction of 3D models from multi-view rendering. We demonstrated that the renderings can be based on multiple input types and used renderings of meshes, point clouds, and partial point clouds with missing regions. The additional reconstruction loss does not behave as a beneficial proxy loss but sharing the backbone during training does not reduce the classification accuracy either.

For future works, the performance of the model for object reconstruction tasks can be analyzed in more detail. With more GPU resources available, more hyperparameters could be tried out to test more combinations of weighting the losses, backbones, or learning rates. Training time on our available RTX3070 GPU was about 3-4h and limited the amount of experiments. Furthermore, an additional mesh reconstruction head based on Mesh R-CNN[4] could be added to potentially improve the reconstructions. Also, analyzing the performance on a more challenging dataset with e.g. more categories to classify would be interesting.

References

[1] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. 2

[2] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. *CoRR*, abs/1604.00449, 2016. 2

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[4] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. *CoRR*, abs/1906.02739, 2019. 2, 4

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 3

[6] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. *CoRR*, abs/1604.03265, 2016. 2

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. 1

[8] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. *CoRR*, abs/1505.00880, 2015. 1

[9] An Tao. Point cloud datasets. `https://github.com/AnTao97/PointCloudDatasets`, 2020. 3

[10] Stanford Computational Vision and Geometry Lab. Shapenet rendering. `http://cvgl.stanford.edu/data2/ShapeNetRendering.tgz`. 2

[11] Stanford Computational Vision and Geometry Lab. Shapenet voxel. `http://cvgl.stanford.edu/data2/ShapeNetVox32.tgz`. 2

[12] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, Shengping Zhang, and Xiaojun Tong. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. *CoRR*, abs/1901.11153, 2019. 2

[13] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 3