3D Shape Classification

Comparing Volume CNNs & Image CNNs for Shape Prediction

Leonid Keselman

Introduction

We focuses on ShapeNet, a recently released dataset that includes 3D CAD models of forty different object categories, dubbed ModelNet40. We implement 3D convolutional neural networks operating directly on these volumes, as well as 2D convolutional networks operating on a learned embedded from the 3D model.

Related Work

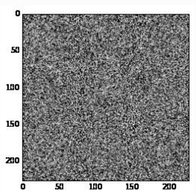
Previously published results suggest that 2D-based CNNs may perform better on ModelNet40. Specifically, the current leader on the dataset uses pre-trained 2D CNNs on multiple rendered views of the models. Similarly, it's been shown that even a single, cylindrical rendering of a mesh, run through a 2D CNN can outperform the initially published 3D CNN results.

Architecture	Convolutions	Pretrain	Top 1 Accuracy
MVCNN [3]	2D	ImageNet	90.1%
VoxNet [2]	3D	None	83.0%
DeepPano [4]	2D		77.6%
3DShapeNets [1]	3D	None	77.0%

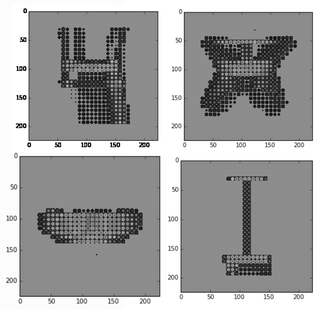
[1] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shapes. CVPR2015.
[2] D. Maturana and S. Scherer. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. IROS2015.
[3] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller. Multi-view Convolutional Neural Networks for 3D Shape Recognition. ICCV2015.
[4] B Shi, S Bai, Z Zhou, X Bai. DeepPano: Deep Panoramic Representation for 3-D Shape Recognition. Signal Processing Letters 2015.

Example Embeddings

Fine-Tuned AlexNet, FC1024 embedding

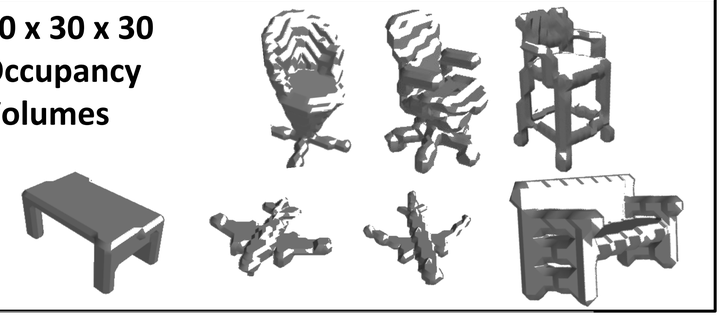


VGG + Upsampling Embeddings



Example Data

30 x 30 x 30 Occupancy Volumes



Methods

Volumetric Convolution

Straightforward mapping of a shallow convolutional neural network to 3D data. Implemented in Torch. Guidance on sizes and dimensions was given by Hao Su and Charles Qi.

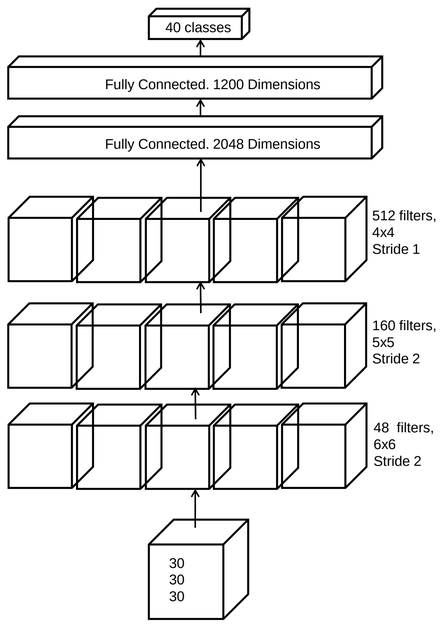
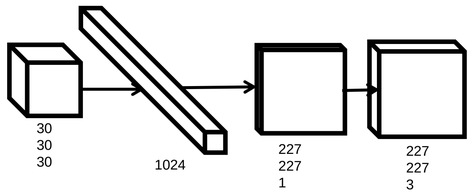


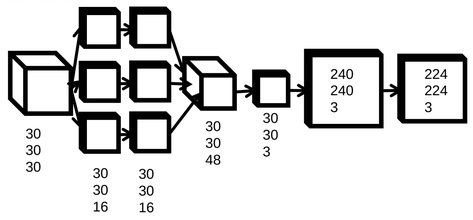
Image Convolution + Embedding

Fine-tuning an embedding layer into an ImageNet-trained classifier. Implemented in Caffe. To utilized an ImageNet trained classifier, we must use an embedding from 2D to 3D. Two architectures were tested and shown to be sucessful for 2D to 3D embedding.

Fully connected embedding. The 30x30x30 volume is first mapped into a 1024 dimensional embedding layer, before being mapped to a 227x227 sized monochromatic image, and them duplicated across all three color channels.



Deconvolution-based embedding. First the input volume is convolved with kernels of sized 1x1x30, 1x30x1, 30x1x1, down each cardinal axis. Then they pass through a bank of convolutions to learn a non-linear transform. They are then concatenated, collapsed into 3 channels, upsampled, and trimmed with an appropriately sized kernel. All convolutions are 3x3.



Tested CNN Architectures & Results

Volumetric Convolution

- **Vanilla Model:** No dropout, using SGD with momentum, 8 epochs. **83.8%**
- **Vanilla Model:** No dropout, using Adam, 8 epochs, **82.2%** (faster convergence)
- **Dropout Model: D**ropout varying from 0.8 to 0.5 retention, using SGD, 8 epochs, **84.6%**
- **Pool + 3x3 Model:** Replace first convolution layer with 3x3 convolutions followed by 3x3 max pooling, dropout at FC layers, SGD. 8 epochs, **85.7%**
- **Pool + 3x3 Model:** Use Nestrov SGD instead, **86.7%**

Image Convolution + Embedding

- **Fully Connected Embedding + CaffeNet: 66.5%**
- **Fully Connected + NetworkInNetwork: 63.0%**
- **Fully Connected + GoogleNet: 57.0%**
- **Fully Connected + VGG:** Lost the protofile!

- **Fully Connected + 3 sets of 1x1 convolution + Single Channel, reducing the number of parameters and** making it a 900 dimensional hidden state**: 67.3%**
- **Fully Connected Embedding + 1x1 Conv + Single Channel + Locking entire original network:** By locking the additional convolutional layers, I was able to get an even better result, at **75.6%**
- **Fully Connected Embedding + 1x1 Conv + Single Channel + Locking entire original network:** Reducing the data to only use a single view dropped it all the way down to **63%**
- **Fully Connected Embedding + 1x1 Conv + Single Channel + train from scratc**h: Training the same network as two above, but without initializing to
- ImageNet learned weights gives only **64%** performance**.**

- **Deconvolutional Embedding + VGG-16: 78.0%**
- **Deconvolutional Embedding + VGG-16 + fine-tune: 83.5%**

- **ResNet50 + deconvolution embedding: 50%**
- **GoogleNet + deconv embedding: 31%**
- **VGG-16 + deconv + xavier init for deconv: 0.025%.**

Challenges and Next Steps

- **Better Embedding Architectures:** Currently we've tested a straightforward set of embedding architectures, including full-connected layers and deconvolutional layers. However, deeper networks or other possible topologies may operate better.
- **Fine-tune additional layers:** Currently these results come from a single pass training operation where the image convolutional network is held locked while the top classifier labels and embedding network are trained.
- **Different data formats**: These are simple occupancy voxel grids, perhaps using something more sophisticated, like a signed distance field, would yield better results.
- **Completing the loop and using real 3D data:** Comparing 3D data from a single viewpoint (like a depth camera) or learning a 3D embedding from 2D data, to do shape classification from images, based only on model annotations.