3D Object Classification using Shape Distributions and Deep Learning

Melvin Low
Stanford University
mwlow@cs.stanford.edu

Abstract

This paper shows that the Absolute Angle shape distribution (AAD) feature can be used with deep learning to achieve appreciable performance on the 3D object classification problem. The generic shape distribution feature was first introduced in 2002, and represents an object as a distribution sampled from a shape function that measures global geometric properties. AAD and similar distributions have several desirable properties as a feature: they are easy to compute, robust to similarity transformations such as translation and rotation, and are insensitive to small perturbations in the object resulting from noise and tesselation. Current state-of-the-art algorithms for 3D object classification largely require voxelization or multiple 2D renderings of each object. These approaches provide good accuracy but lack the aforementioned benefits of shape distributions. The result of this paper–80% test accuracy on ShapeNetCore with a seven-layer convolutional neural network–suggests that AAD might be considered as a way to augment existing techniques, given its simplicity, efficiency, yet surprising performance.

1. Introduction

The 3D object classification problem is a familiar one in computer vision that has been the focus of much previous research. Like image classification, it has many applications, such as in the medical and manufacturing industries where volumetric images or 3D models are prevalent. To date the problem has not been adequately solved.

A survey of recent attempts to tackle the problem reveals a tendency toward deep learning models. Examples include MVCNN, VoxNet, and 3D ShapeNets. These models are at the forefront, showing high performance and achieving over 90% classification accurary on certain benchmark dataset. Most of them approach the problem by converting each object into voxels or multiple image renderings, and then performing convolutions over the processed data.

Before the resurgence of deep learning during the last couple years, effort was focused on inventing more descriptive features for 3D objects. In 2002, Osada et al. [7] introduced one such feature, the shape distribution. It aimed to describe a mesh through a distribution of values calculated between points sampled from the surface of the mesh. Its original form, *D2*, was a univariate distribution of pairwise distances. Researchers have since then created other ones. The one used in this paper is the *AAD* feature, proposed by Ohbuchi et al. [5] in 2005, that describes a bivariate joint over pairwise distances and angles.

These shape distributions are notable for two reasons. First, they are easy and quick to construct, as they only require sampling from the surface of the mesh. Second, they are robust: minor perturbations in the mesh resulting from noise or tesselation do not greatly affect the feature, and similarity transformations such as rotation and scaling do not affect it at all. In constrast, voxels and image slices, the standard data format used in current deep learning models, are not as efficient to compute and are not invariant to transformations such as rotation.

As shape distributions were introduced a decade ago, they were evaluated with more simple classifiers such as nearest neighbor and support vector machines. Results were ordinary and not exceptional. This paper demonstrates, however, that the full potential of shape distributions for object classification was never realized. More sophisticated classification models such neural networks can boost their performance to suprising levels.

In this paper, several different neural networks were trained to classify objects from the ShapeNetCore dataset, which consists of 41313 models belonging to 55 different classes. The models were not axis-aligned. Each model was converted into a discretized AAD distribution of resolution 128x16, which was then used as input to the neural networks. Several of the networks achieved 80% testing accuracy; the simplest one had seven weight layers and was trained in two hours.

2. Related Work

Neural networks are currently the most common approach toward 3D object classification. As a result, there are many such models. Some of these include: Multi-

view CNN (MVCNN) [8], 3D ShapeNets [10], and VoxNet [4]. Most of these models capitalize on methods that have worked well for 2D image classification. For example, MVCNN, the top performing model at this time, renders each object from multiple different views and runs a neural network on each view, reducing the 3D problem into multiple 2D ones. Both 3D ShapeNets and Voxnet convolve 3D filters over voxel data, which can be seen as a natural extension of 2D filters over image pixel data.

Spatial transformer networks, described by Jaderberg et al. [2], are notable because they provide the capability to spatially transform and manipulate geometries within the network. This is especially relevant to 3D object classification because input models may not be axis-aligned.

Shape distributions have seen sparse mention in recent literature. They were first introduced by Osada et al. [7] [6] around 2001 or 2002. Obuchi et al. [5] introduced enhanced versions three years later, including the AAD distribution used in this paper. A smattering of research has been done on ensembles [9]. As far as I know, this is the first work that uses shape distributions as input to a neural network.

Two benchmark datasets commonly used are ModelNet [10] (created for use with the 3D ShapeNets model) and ShapeNet [1]. ShapeNet is currently running a large-scale 3D shape retrieval contest, which may be of interest. The dataset used in this paper is the one provided for this contest.

3. Methods

There were three main components: (1) creating the AAD shape distributions; (2) normalizing and augmenting the data; (3) training and evaluating neural networks.

3.1. AAD Shape Distribution

The AAD shape distribution is described in detail in Ohbuchi et al. [5]. In addition, Osada et al. [7] provides detailed information on sampling methods needed to produce the distributions. A brief overview of these methods is provided in this section, as well as the specific instantiation of their parameters used in this experiment.

In order to construct the AAD feature for a given object, K points are first sampled from its surface. This is done by selecting K faces at random, where each face is weighted by its surface area. Then, a point is randomly sampled from each face. For a triangular face with vertices (A, B, C), the point P can be calculed by generating two random numbers between 0 and 1, r_1 and r_2, and evaluating the following equation [7]:

$$P = (1 - \sqrt{r_1})A + \sqrt{r_1}(1 - r_2)B + \sqrt{r_1}r_2C$$

Then, N pairs of points are randomly selected among the K points generated in the previous step. For each pair of points (a, b), two values are calculated: $v_1 = \|a - b\|$ and $v_2 = \left|n(a)^T n(b)\right|$, where $n(x)$ is the normal vector of the face where the point x resides. The first is thus Euclidean distance. The second is the absolute value of the cosine angle between the faces from which the points were sampled. These two values form a single sample, (v_1, v_2), from a joint distribution between distances and angles.

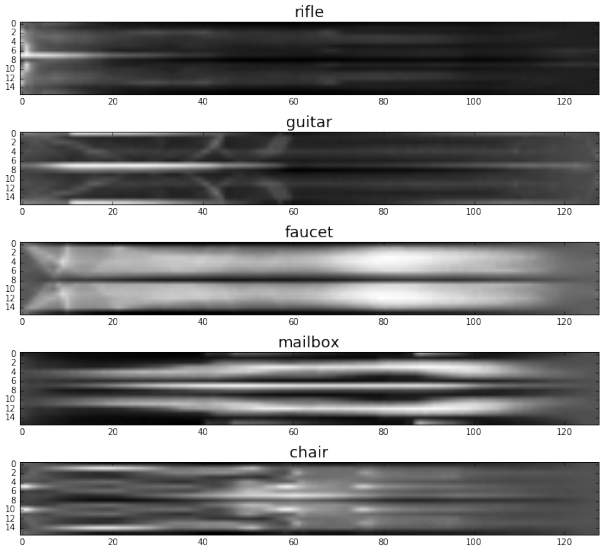


Figure 1: Visualization of AAD shape distributions for 5 objects in ShapeNetCore belonging to different classes.

The final step is to create the AAD distribution itself from the N samples. This is done by finding the minimum and maximium of the samples in each of the two dimensions, discretizing the ranges into (b_1, b_2) bins, and then putting each sample into the bin in which it belongs. The count of each bin is divided by N to ensure that the result sums to 1 and is thus a valid probability distribution.

In this paper, $K = 10^4$, $N = 10^6$, $b_1 = 128$, and $b_2 = 16$. Large (probably unnecessarily large, although this was not thoroughly tested) values were chosen for K and N to emphasize accuracy over performance. For comparison, Ohbuchi et al. [5] has $K = 2(10)^3$, $N = (K)(K-1)/2$, $b_1 = 64$, and $b_2 = 8$. Ohbuchi et al. computes (v_1, v_2) for every pair of the K points sampled from the object.

Regardless of the large parameters, the calculation of the AAD feature for each object took under half a second on a modern cpu. A visualization of the distributions for several objects is shown in Figure 1.

3.2. Data Augmentation and Normalization

The dataset was augmented by randomly scaling each object four times before the creation of AAD shape distributions. Each dimension of each object was separately scaled by a value chosen uniformly at random from the interval $[0.75, 1.25]$. Random rotation was not performed because

Table 1: Three neural network architectures tested. All convolutional layers are followed by batch normalization, and all activations are Relu. In addition, dropout of 0.50 follows the first two FC layers. *Note: there was an error in the poster report. The 3-layer net is actually model C, a 7-layer net.*

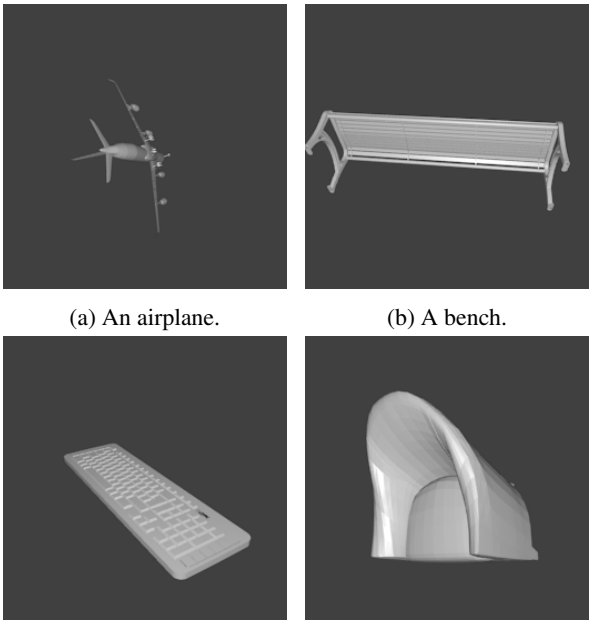
Neural Network Configuration		
A	B	C
input = (128, 16)		
conv3-64	conv3-64	conv3-128
conv3-64	conv3-64	maxpool2x1
maxpool2x1	maxpool2x1	conv3-256
conv3-128	conv3-128	maxpool2x1
conv3-128	conv3-128	conv3-512
maxpool2x1	maxpool2x1	maxpool2x1
conv3-256	conv3-256	conv3-512
conv3-256	conv3-256	maxpool2x2
conv3-256	conv3-256	FC-2048
maxpool2x1	conv3-256	FC-2048
conv3-512	maxpool2x1	FC-55
conv3-512	conv3-512	soft-max
conv3-512	conv3-512	
maxpool2x2	conv3-512	
conv3-512	conv3-512	
conv3-512	maxpool2x2	
conv3-512	conv3-512	
FC-4096	conv3-512	
FC-4096	conv3-512	
FC-55	conv3-512	
soft-max	FC-4096	
	FC-4096	
	FC-55	
	soft-max	

the AAD feature is inherently invariant to rotation.

The training data, consisting of the generated AAD features, was then mean centered and set to have standard deviation of 1 in each bin. Note that the resulting features are no longer valid probability distributions.

3.3. Architecture

Several different neural network were trained on the AAD features. Three of these are shown in Table 1. Models A and B are VGG-16 and VGG-19 with modified pooling layers (to account for the input features being of reduced size). Model C reduces the feature size quickly through alternating convolutional and pooling layers, and then contains dense layers like the others. Batch normalization was used after every convolutional layer, and dropout of 0.50 was added after the first two FC layers in each model. All activations are Relu.



(a) An airplane. (b) A bench.

(c) An keyboard. (d) A chair.

Figure 2: Four models from ShapeNetCore. They are not axis-aligned.

3.4. Training

The networks were implemented in Keras, and training was performed on a NVIDIA 980Ti graphics card with a batch size of 128. The optimizor chosen was Adam [3], which was initialized with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The learning rate α was set initially to 0.001 and reduced by a factor of 10 each time validation loss seemed to stopped decreasing, a total of two times for each model. Finally, regularization on the FC layers was set to 0.005 and cross validation with a split size of 0.10 was used.

4. Dataset

At the time this experiment was being conducted, the ShapeNet team at Stanford University was running a large-scale 3D shape retrieval contest. For the contest, they released a subset of their dataset, ShapeNetCore, which contained about 51300 models belonging to 55 common categories. The dataset was split into training, validation, and test sets with proportions 70%, 10%, and 20%. True labels were given for the training and validation sets but not for the test set. As a result, this experiment discarded the provided test set and replaced it with the validation set. Cross-validation was thus used on the training set in lieu of a separate validation set.

The class labels in ShapeNet are very long. One such la-

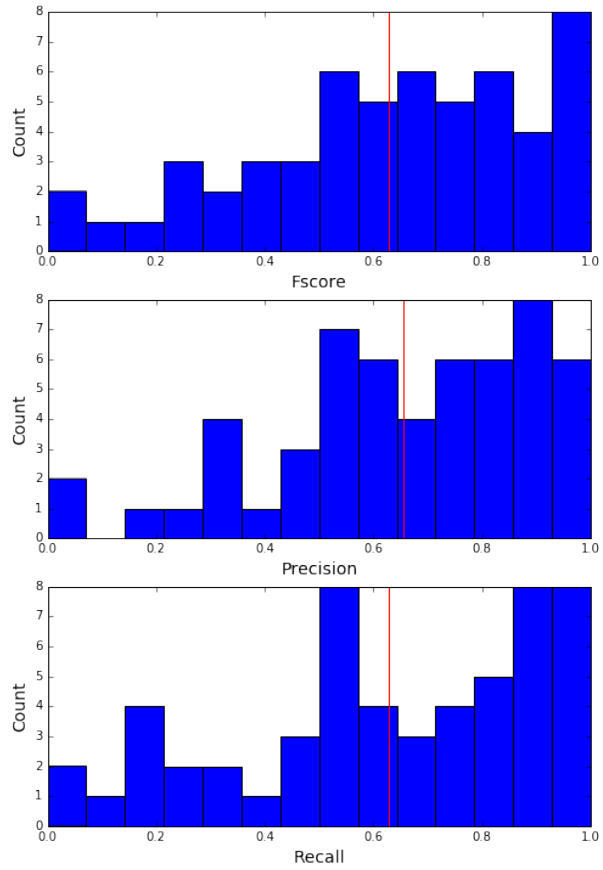


Figure 3: Histograms of fscore, precision, and recall over the test set. The red line in each figure denotes the mean.

bel is: "bag,traveling bag,travelling bag,grip,suitcase." For brevity and also because of some formatting issues, I have manually reduced each of these labels as best as I could. For example, I have changed the label mentioned above to simply "bag."

Four models from the dataset are shown in Figure 2.

5. Results and Analysis

All three models were able to achieve at least 80% accuracy on the test set and 0.65 mean accuracy precision (mAP). Training loss was consistently lower than validation loss with all models despite strong regularization, suggesting that there was some overfitting. As a result, it is possible that model A, the simplest model, could have been simplified further without sacrificing performance. The results that follow are from model A; the other two models achieved similar results.

5.1. Microstatistics

Precision, recall, and fscore for each of the 55 classes are given in Table 2, and Figure 3 displays the histograms

Table 2: Results by class, sorted by fscore.

Class Label	Precision	Recall	Fscore	Support
bicycle	1	1	1	6
car	0.997	1	0.998	353
plane	0.950	1	0.974	405
guitar	0.974	0.95	0.962	80
laptop	0.92	1	0.958	46
rifle	0.952	0.940	0.946	237
bus	0.927	0.946	0.936	94
motorbike	0.891	0.970	0.929	34
chair	0.903	0.907	0.905	678
skateboard	1	0.8	0.888	15
knife	0.883	0.883	0.883	43
table	0.815	0.916	0.863	843
vessel	0.801	0.917	0.855	194
train	0.888	0.820	0.853	39
sofa	0.814	0.861	0.837	317
bathtub	0.891	0.776	0.830	85
bowl	0.761	0.888	0.820	18
handgun	0.729	0.870	0.794	31
faucet	0.769	0.8	0.784	75
lamp	0.765	0.745	0.755	232
phone	0.709	0.790	0.747	105
mug	0.727	0.727	0.727	22
display	0.745	0.697	0.720	109
bottle	0.815	0.62	0.704	50
can	0.666	0.727	0.695	11
helmet	0.565	0.812	0.666	16
pillow	0.833	0.555	0.666	9
bed	0.857	0.521	0.648	23
washer	0.6	0.705	0.648	17
bench	0.824	0.519	0.637	181
keyboard	0.5	0.857	0.631	7
cabinet	0.564	0.700	0.625	157
projectile	0.625	0.555	0.588	9
microwave	0.562	0.6	0.580	15
jar	0.581	0.542	0.561	59
speaker	0.652	0.468	0.545	160
earphone	0.5	0.571	0.533	7
piano	0.611	0.458	0.523	24
trash bin	0.515	0.5	0.507	34
birdhouse	0.6	0.428	0.5	7
bookshelf	0.469	0.511	0.489	45
dishwasher	0.384	0.555	0.454	9
printer	0.466	0.437	0.451	16
tower	0.32	0.615	0.421	13
clock	0.479	0.353	0.407	65
mailbox	0.6	0.3	0.4	10
file cabinet	0.666	0.2	0.307	30
stove	0.333	0.272	0.3	22
bag	0.333	0.25	0.285	8
cap	0.5	0.2	0.285	5
basket	0.333	0.181	0.235	11
remote control	0.25	0.166	0.2	6
pot	0.171	0.1	0.126	60
camera	0	0	0	11
microphone	0	0	0	7

of these values. A normalized confusion matrix can additionally be found in Figure 4, at the end of this report.

The classifier performed extremely well on several classes but very poorly on others. This difference in performance might be explained by the fact that the AAD distribution intuitively captures holistic depth and volume information, which are fairly distinctive for some classes and not so much for others. For example, bicycles, cars, planes, and guitars–classes with the highest fscore–all have easily distinguishable shapes. On the other hand, baskets and pots and trash bins–classes with low fscores–all have the same general shape: an open receptable that is (most likely) curved around the perimeter. Corroborating this theory, one common mistake the classifer made was to label remote controls as phones. Both of these classes are box-like objects with buttons, so it would make sense that the AAD distribution feature could not do a good job distiguishing between them.

5.2. Macrostatistics

80.72% accuracy on the test set was achieved with 0.655 mAP. For comparison, initial results from the ongoing ShapeNet retrieval contest show the best two performers, Bai_GIFT and SU_MVCNN, attaining 0.730 and 0.662 mAP on a similar dataset (they evaluated their algorithms on the competition test set, which was discarded in this project–see Section 4). One interesting thing to note is that both of these competition algorithms performed better on axis-aligned models, getting mAP of 0.740 and 0.817 respectively. However, since AAD shape distributions are rotation invariant, the model presented in this paper would still achieve the same performance, 0.655 mAP, on axis-aligned models–which should theoretically be an easier task.

5.3. Discussion and Future Work

The performance of the AAD shape distribution suggests that it might be a good choice for a feature when the goal is to classify objects that have distinctive shapes and volumes, such as planes and cars. On the other hand, for classes that have more subtle differences, such as different types of cars, the AAD shape distribution won't do as well.

Regardless, because of the ease of computing the AAD shape distribution, the feature might be considered as an simple way to boost the performance of existing models, although this needs to be tested and is a good candidate for future research.

It is entirely possible that as much performance has been squeezed out of the 128x16 resolution AAD features as possible. This is suggested by the fact that models with a lot of layers could not improve upon the result attained by the simple 7-layer one. Future work may involve increasing the resolution of the AAD feature to see if it addresses this issue.

6. Conclusion

This paper introduces a new model for classifying 3D objects using the AAD shape distribution. It shows that, using a 7-layer net, 80% classification accuracy can be attained on ShapeNetCore, a benchmark dataset. While this result is not state-of-the-art, the simplicity of the model as well as other nice properties of AAD–its invariance to rotation, ease of calculation–make for an interesting discussion.

References

[1] 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., AND YU, F. ShapeNet: An Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[2] JADERBERG, M., SIMONYAN, K., ZISSERMAN, A., AND KAVUKCUOGLU, K. Spatial transformer networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2017–2025.

[3] KINGMA, D., AND BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[4] MATURANA, D., AND SCHERER, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on* (2015), IEEE, pp. 922–928.

[5] OHBUCHI, R., MINAMITANI, T., AND TAKEI, T. Shape-similarity search of 3d models by using enhanced shape functions. *International Journal of Computer Applications in Technology 23*, 2-4 (2005), 70–85.

[6] OSADA, R., FUNKHOUSER, T., CHAZELLE, B., AND DOBKIN, D. Matching 3d models with shape distributions. In *Shape Modeling and Applications, SMI 2001 International Conference on.* (2001), IEEE, pp. 154–166.

[7] OSADA, R., FUNKHOUSER, T., CHAZELLE, B., AND DOBKIN, D. Shape distributions. *ACM Transactions on Graphics (TOG) 21*, 4 (2002), 807–832.

[8] SU, H., MAJI, S., KALOGERAKIS, E., AND LEARNED-MILLER, E. G. Multi-view convolutional

neural networks for 3d shape recognition. In *Proc. ICCV* (2015).

[9] WOHLKINGER, W., AND VINCZE, M. Ensemble of shape functions for 3d object classification. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on* (2011), IEEE, pp. 2987–2992.

[10] WU, Z., SONG, S., KHOSLA, A., YU, F., ZHANG, L., TANG, X., AND XIAO, J. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 1912–1920.

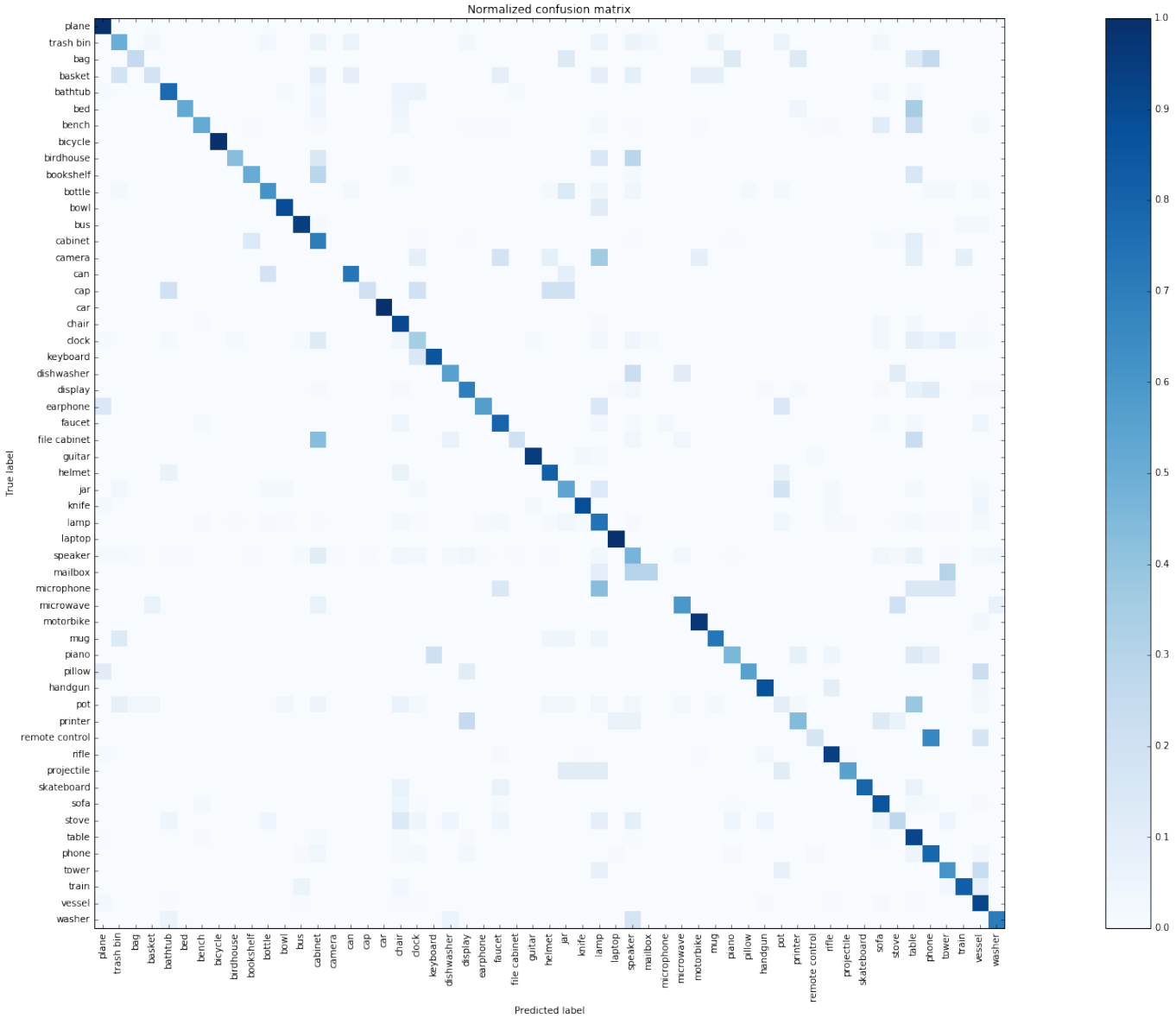


Figure 4: Normalized conflusion matrix.