Stacked Local Predictors for Large-Scale Point Cloud Classification

Benjamin Eckart, Alonzo Kelly
The Robotics Institute
Carnegie Mellon University
eckart@cmu.edu, alonzo@ri.cmu.edu

1. Introduction

Many modern 3D range sensors, such as a Velodyne or Kinect, generate on the order of one million data points per second. For purposes of real-time semantic scene understanding, care must be made for efficient algorithms that not only run adequately fast, but also make the best use of the large amounts of data available. In this paper, we propose a novel point cloud classification scheme that 1) can be trained in a Map-Reduce framework and 2) allows real-time inference on commodity hardware. This method of training and classification makes the algorithm suitable for training on large amounts of labeled point cloud data and fast classification on new data during run-time. The algorithm works by segmenting feature space using random projections (LSH or Locality Sensitive Hashing) and training local classifiers. A separate contextual classifier is then run on neighbors in Euclidean space as a meta-learning procedure (stacking). The end result is a fast algorithm that outperforms the current state-of-the-art on a million point benchmark dataset.

2. Goals

One of the most fundamental and important tasks for a mobile robot is to be able to adequately assimilate and respond to the data coming from its sensors. In the case of 3D range sensing, modern-day sensors generate massive quantities of data that are often thrown away due to lack of computational resources. We would like to be able to effectively harness the power of "big data" in this setting to not only train simpler models [2], but to also generate fast inference procedures, capable of running in real-time using onboard hardware.

Our goal is to devise a method providing the descriptive power of non-linear classifiers with the efficiency of linear classifiers. Furthermore, we need to make sure that underrepresented classes do not get ignored and the assigned labels are spatially coherent. We would like to do all of this in a way that capitalizes on the large quantities of data we have available.

3. Proposed Method

The proposed algorithm has two learning phases. The first phase learns traditional per-point classification in a "locally linear" way, and the second phase learns spatial context through stacking to enforce coherency among nearby points.

3.1. Phase 1: Local Learning

Using spin images, the first phase splits up the data into meaningful feature clusters to train local classifiers on each cluster. The general idea is illustrated in Figure 1. The graphic on the left shows how a linear classifier may do a poor job on data with a non-linear decision boundary. In this case, each feature vector has two dimensions and the classification problem is binary. The graph on the right shows the same data, but with several partitions over one of the features. A different linear classifier is learned per feature partition, resulting in no misclassified examples.

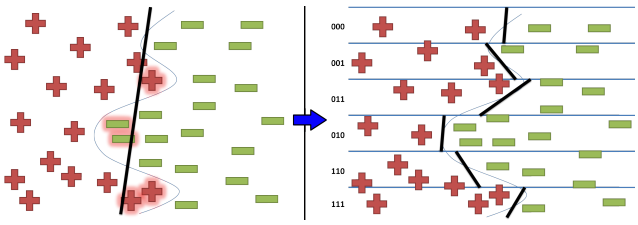

Figure 1. The idea behind "locally linear" learning. Non-linear decision boundaries can be more closely approximated by clustering the feature space and learning linear classifiers locally on each feature cluster.

Not only will non-linear decision boundaries be more closely approximated by several locally linear decision boundaries, but learning locally also has computational advantages in that much of the work can be split up in parallel. If the feature clustering can be done in an efficient way, then each local learner can be trained simultaneously.

Unlike iterative algorithms such a k-means clustering, LSH allows us to cluster the feature space in constant time

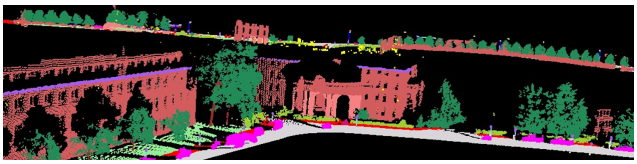

Figure 2. Figure from [5]. This point cloud is colored coded by label and was taken from the Oakland neighborhood of Pittsburgh, PA, USA.

per point. This means that we do not need to see all of our data to make decisions about which cluster a particular data point belongs. This convenient property allows us to efficiently segment our training data into feature clusters and simultaneously train local learners on each cluster. Each cluster has significantly lower class entropy after a given segmentation, meaning that LSH effectively acts as a single step decision tree split. The fact that the after-split entropy is much lower means that each individual learner has an easier job teasing out the parameters that discriminate among classes. Furthermore, rare classes often get segmented to the same code, boosting their relative frequency within that particular local classifier.

3.2. Phase 2: Contextual Learning

One problem of solely using Phase 1 is one of spatial consistency. Since per point features encode local properties of surfaces only, the larger context of scene is lost. To recover this context, we can perform a single round of "stacking" [3], where the candidate function is a nearest neighbor function in a global Euclidean x-y-z space on the point cloud. In other words, we can use the Learners themselves to generate labels and then look at local label distributions in x-y-z space for each point. A final classifier can then be learned on this data, resulting in a learner that knows spatial context. The intuition is that nearby class labels are very informative to the point currently being classified, and furthermore, the class labeling should be smooth and consistent with its neighbors.

4. Results

In this work, we use the point cloud dataset first introduced in [4]. It consists of roughly 1.2 million data points, with many different labels like "car, tree, telephone wire, curb, etc." A picture of part of the dataset can be seen in Figure 2 with the points color-coded by class. The scene is from the neighborhood of Oakland in Pittsburgh, PA, USA.

We did several tests over the data using different combinations of Phase 1, Phase 1+2, and the classical methods. The results are summarized in Table 1. The first classifier, a majority classifier, simply classifies all points as ground.

Table 1. Classification accuracies the various learning methods.

DATA SET	NAIVE
MAJORITY CLASSIFIER	0.748
LINEAR SVM (LIBLINEAR) [2]	0.897
LOCALLY HASHED MAJORITY (LSH-10BIT)	0.827
LOCALLY HASHED 5-NN (LSH-9BIT)	0.870
LOCALLY HASHED SVM (LSH-4BIT+LIBLINEAR)	0.903
STACKED LOCALLY HASHED 5-NN (LSH-9BIT)	0.912
STACKED LOCALLY HASHED SVM (LSH-8BIT)	0.929
HIGH ORDER ASSOCIATIVE MARKOV NETWORK [5]	0.871
PAIRWISE ASSOCIATE MARKOV NETWORK (3-NN) [5]	0.884
PAIRWISE ASSOCIATE MARKOV NETWORK (5-NN) [5]	0.889

It is the baseline. The next classifier is linear SVM classifier optimized to be efficient using large datasets and trained over the entire data [1] (liblinear). We use this classifier as our base local learner, unless otherwise noted. The next two rows detail the local learner accuracies. The first is a locally hashed majority classifier, which does slightly better than the non-locally hashed majority classifier. This result is an indication that LSH is doing well separating data into clusters of different classes. We see that the 4-bit code with liblinear outperforms the globally trained liblinear. The next two results in the table are using both training phases, applying stacking as a final classifier to add contextual information. Here, the 8-bit stacked linear SVM classifier significantly outperforms every other method at .929 overall class accuracy. Finally, the last three results come from related work, using Markov networks to perform similarly fast contextual real-time inference. Their results are comparable with the locally trained learners, but are outperformed by the methods with stacking.

References

[1] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: a library for large linear classification. *J. Mach. Learn. Res.*, 9:18711874, June 2008. 2

[2] A. Halevy, P. Norvig, and F. Pereira. The unreasonable effectiveness of data. *Intelligent Systems, IEEE*, 24(2):8 –12, Apr. 2009. 1

[3] Z. Kou and W. Cohen. Stacked graphical models for efficient inference in markov random fields. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, 2007. 2

[4] D. Munoz, J. Bagnell, N. Vandapel, and M. Hebert. Contextual classification with functional Max-Margin markov networks. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 975 –982, June 2009. 2

[5] D. Munoz, N. Vandapel, and M. Hebert. Onboard contextual classification of 3-D point clouds with learned high-order markov random fields. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2009 – 2016, May 2009. 2