Logarithmic-time multi-class prediction via metric embedding of classifiers

Srinivas C Turaga, H Sebastian Seung

February 13, 2011

Abstract

Large-scale multiclass classification becomes computationally challenging as the number of classes grows. The classic method of training "one-vs-all" binary classifiers has computational complexity $O(k)$ with k classes. This can be improved to $O(\log k)$, yet existing methods are problematic. We show how re-casting the problem as nearest neighbor search between patterns and classes allows us to solve multi-class classification in a rather simple way, while still achieving $O(\log k)$ complexity both at training time and testing time. The same trick can be applied to existing "one-vs-all" linear predictors leading to exponential improvements in their computational complexity as well.

Many important problems in computer science can be cast as large-scale multi-class prediction. Object recognition in the real world involves categorizing an object into not just 1 of 256 categories, but millions of categories, and newer benchmark datasets already incorporate over 1000 categories. Face recognition can involve picking the best match from a database of hundreds of millions of individuals. Document search involves picking the best matching document from well over millions of other documents, for a given query. With the rise in large-scale prediction problems, it is important to have learning algorithms that scale well with the number of classes.

In this work, we tackle the computational complexity of multi-class prediction by introducing a class of "metric" multiclass predictors that have $O(\log k)$ complexity at test time, where k is the number of possible classes an instance can belong to. These predictors can also be trained using stochastic gradient methods in time $O(\log k)$ per epoch.[1]

Our work is motivated by a curious fact. The popular "one-vs-all" multiclass predictor is believed to take $O(k)$ time to produce a prediction at test time [5], after training which also takes $O(k)$. In contrast, with efficient data-structures [1], the simple nearest neighbor classifier can produce a prediction in time $O(\log N)$ with N training samples. At first glance, the runtime of the nearest neighbor classifier appears to have no dependence on k. But since there must be atleast one training sample per class, the number of training samples must exceed k, $N \geq k$. Thus the nearest neighbor classifier which really produces multi-class predictions in time $O(\log k)$, is faster than the one-vs-all classifier.

Brute-force nearest-neighbor computation can take $O(N)$, yet modern nearest neighbor methods achieve $O(\log N)$ run-time by exploiting the metric structure of the distance function used, most commonly the euclidean and Mahalanobis distances. The triangle inequality $d(\mathbf{x}_a, \mathbf{x}_c) \leq d(\mathbf{x}_a, \mathbf{x}_b) + d(\mathbf{x}_b, \mathbf{x}_c)$ that metric distance functions are required to satisfy enables the construction of search trees that speed up the computation.

Inspired by the efficiency of nearest-neighbor search in metric spaces, we can design a multiclass classifier that uses the same triangle inequality to prune the search space when looking for the best class. By embedding classifiers into the same metric space as the patterns, one can compute and use distances between the classes themselves to prune the search space.

[1] The training time also depends on N, the number of training samples.

The multiclass predictor is $f(\mathbf{x}) = \arg\min_i d(\mathbf{x}, \mathbf{w_i})$, where $\mathbf{x}$ is the pattern to be classified and $\mathbf{w}_i$ is metric space representation of class i. And we can use standard nearest neighbor methods to find the exact nearest neighbor in $O(\log k)$. We can use stochastic gradient descent to train this predictor at a cost of $O(\log k)$ per sample, also by exploiting the same data structures and by minimizing a structured multiclass loss function $l(\mathbf{x}, y; \{\mathbf{w_i}\}) = [1 + d(\mathbf{x}, \mathbf{w}_y) - \min_{j \neq y} d(\mathbf{x}, \mathbf{w}_j)]_+$, where y is the true class of pattern $\mathbf{x}$, and the minimization is over all false classes in order to find the most offensive false class.

The same tricks can be used to speed up the classic "one-vs-all" SVM, which is a leading performer on benchmarks such as the ImageNet 2010 challenge with 1000 categories[2]. This model $f(\mathbf{x}) = \arg\max_i \mathbf{x}^T \mathbf{w_i}$, can be trained and tested efficiently by constructing correlation-based search trees. While the triangle inequality of distances no longer applies, a similar inequality can be derived for the inner product in euclidean spaces making efficient search possible. Thus contrary to popular belief [5], the "one-vs-all" SVM costs only $O(\log k)$ at test time. Somewhat counter-intuitively, training this predictor using the structured multiclass loss costs only $O(\log k)$ while the naive classic method using an inferior loss function costs $O(k)$.

This method can certainly be generalized easily to multilabel prediction, but perhaps conditional probability prediction as well, making it competitive with all our competition [5, 2, 3, 4].

References

[1] A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd international conference on Machine learning*, pages 97–104. ACM, 2006.

[2] Alina Beygelzimer, John Langford, Yuri Lifshits, Gregory Sorkin, and Alex Strehl. Conditional probability tree estimation analysis and algorithms. 2009.

[3] Alina Beygelzimer, John Langford, and Pradeep Ravikumar. Error-correcting tournaments. 02 2009.

[4] T.G. Dietterich and G. Bakiri. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[5] Daniel Hsu, Sham M. Kakade, John Langford, and Tong Zhang. Multi-label prediction via compressed sensing. 02 2009.

Category: Learning algorithms
Presented by: Srini Turaga
Preference: Oral

[2] http://www.image-net.org/challenges/LSVRC/2010/