Large-scale Distributed Optimization for Improving Accuracy at the Top

Stephen Boyd, Corinna Cortes, Chong Jiang, Mehryar Mohri, Ana Radovanović, Joelle Skaf
Google, Inc., New York

Abstract

In this paper, we present a large-scale distributed implementation of the accuracy at the top algorithm, which is a new notion of classification accuracy based on the top τ-quantile values of a scoring function. Our implementation approach is based on the Alternating Direction Method of Multipliers (ADMM) consensus framework, written in Pregel (a unified framework for performing large-scale graph computations, [6]) and meant for solving large scale convex optimization problems in a distributed fashion.

1 Introduction

The learning problem we consider is that of maximizing the *accuracy at the top*, which consists of achieving the ordering of all items so that items whose scores are among the top τ-quantile are as accurate as possible. Thus, ideally, all non-preferred items are ranked below the quantile. This problem is crucial for many information retrieval systems such as search engines or recommendation systems, since most users of these systems browse or consider only the top selected items.

As discussed in [4], different criteria have been introduced in the past to measure the quality of getting the 'top' items, including the precision at k (Precision@k), the normalized discounted cumulative gain (NDCG) and other variants of DCG, or the mean reciprocal rank (MRR) when the rank of the most relevant document is critical. In this regard, several machine learning algorithms have been recently designed to optimize these criteria and other related ones [5, 11, 10, 16, 7, 13, 12]. A general algorithm inspired by the structured prediction technique SVMStruct [17] was incorporated in an algorithm by [14] which can be used to optimize a convex upper bound on the number of errors among the top k items. The algorithm seeks to solve a convex problem with exponentially many constraints via several rounds of optimization with a smaller number of constraints, augmenting the set of constraints at each round with the most violating one. Another algorithm, also based on structured prediction ideas, is proposed in an unpublished manuscript of [15] and covers several criteria, including Precision@k and NDCG. A regression-based solution is suggested by [9] for DCG in the case of large sample sizes. Some other methods have also been proposed to optimize a smooth version of a non-convex cost function in this context [8]. [1] discusses an optimization solution for an algorithm seeking to minimize the position of the top irrelevant item.

In [4], we propose an algorithm, called AATP, that optimizes accuracy in some top *fraction* of scores returned by a real-valued hypothesis. The desired objective is to learn a linear scoring function that is as accurate as possible for the items whose scores are above the top τ-quantile. Our algorithm optimizes a convex surrogate of the corresponding loss in the case of linear scoring functions. We show that the solution of this problem can be obtained *exactly* by solving several, independent, convex optimization problems in parallel. More specifically, each optimization problem corresponds to a single quadratic program (QP) that minimizes the convex loss function subject to the assumption that the particular item k's score corresponds to the τth quantile of the resulting scoring function

(QP k):

$$\min_{\mathbf{w}} \sum_{i=1}^{m} \left(f_i(\mathbf{w}, q^*(\mathbf{w})) + f_i'(\mathbf{w}, q^*(\mathbf{w})) \right) \tag{1}$$
$$\text{s.t.} \quad q^*(\mathbf{w}) = \mathbf{w} \cdot \mathbf{z}_k,$$

where $\mathbf{z}_k \in \{\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{x}_1', \ldots, \mathbf{x}_m'\}$, and

$$f_i(\mathbf{w}, q^*(\mathbf{w})) = (\mathbf{w} \cdot \mathbf{x}_i - q^*(\mathbf{w}) + 1)_+ + C/(2m)\|\mathbf{w}\|^2,$$

and

$$f_i'(\mathbf{w}, q^*(\mathbf{w})) = (q^*(\mathbf{w}) - \mathbf{w} \cdot \mathbf{x}_i + 1)_+ + C/(2m)\|\mathbf{w}\|^2.$$

Then, if $\mathbf{w}_k$ denotes the solution of the problem QP k, the resulting solution of this sequence of problems equals to $\mathbf{w}_k$ for which kth score corresponds to the τ-quantile of the scoring function with the minimum achievable loss value. An interested reader is referred to [4] for more details.

Solving the set of independent QPs suggests distributed implementation executed in parallel. In addition, the separability of the objective function and very large training sets in real life applications led to the implementation approach that is based on *Alternating Direction Method of Multipliers* (ADMM) (see [2] for more details). The method is particularly suited for solving large scale convex optimization problems that arise in (but are not limited to) areas of statistics and machine learning. Most commonly, these problems appear in application domains where data sets are extremely large, high-dimensional, and stored in a distributed manner. In the rest of this write-up, we present a brief overview of our implementation approach and overall methodology, which can easily be generalized to solving any convex (constrained and non-constrained) optimization problem.

2 Implementation Approach

The implementation of the AATP algorithm is based on the ADMM consensus framework, written in Pregel (a unified framework for performing large-scale graph computations, [6]) and meant for solving convex optimization problems in a distributed fashion. The framework consists of a data model and a library of solvers that together permit the representation and solution of virtually any convex optimization problem. In general, solving a specific problem comes down to distributing the objective(s) and constraint(s) over nodes in a connected graph, in particular:

- Picking a topology for a connected graph;
- Encapsulating the data for the function at each node in a proto;
- Specifying the proximal step solver for each node.

ADMM is a simple, provably converging, algorithm that uses a decomposition-coordination procedure, in which solutions to small local subproblems are coordinated to find a solution to a large global problem.

In the context of the AATP, each QP in (1) corresponds to a single connected component in a graph (see Figure 1 for clarity). Each connected component is a directed graph, where a single node, say i, optimizes $f_i(\mathbf{w}, q^*(\mathbf{w}))$ or $f_i'(\mathbf{w}, q^*(\mathbf{w}))$, depending on the term in the objective function it corresponds to. Graph nodes exchange messages in each superstep of the consensus algorithm, and the problem variables are updated in a strictly set order. Once the ADMM consensus converges to the optimal solution, all nodes in the connected component store the optimal values of the primal and dual variables.

The used framework allows us to train on large data sets with millions of nodes and thousands of features. Connected components are distributed across workers in a cloud. By careful optimization of the number of nodes per worker and sparse representation of the input data, we increase the training capacity. The restrictions of the current implementation stem mainly from the Pregel's limitations. However, with the continuous improvements of the Pregel framework, we expect enhancements in capabilities of our implementation as well.

There are several parameters that impact the speed of convergence of the ADMM consensus and are tunable. One is the graph topology: the larger the connectivity of components, the faster is

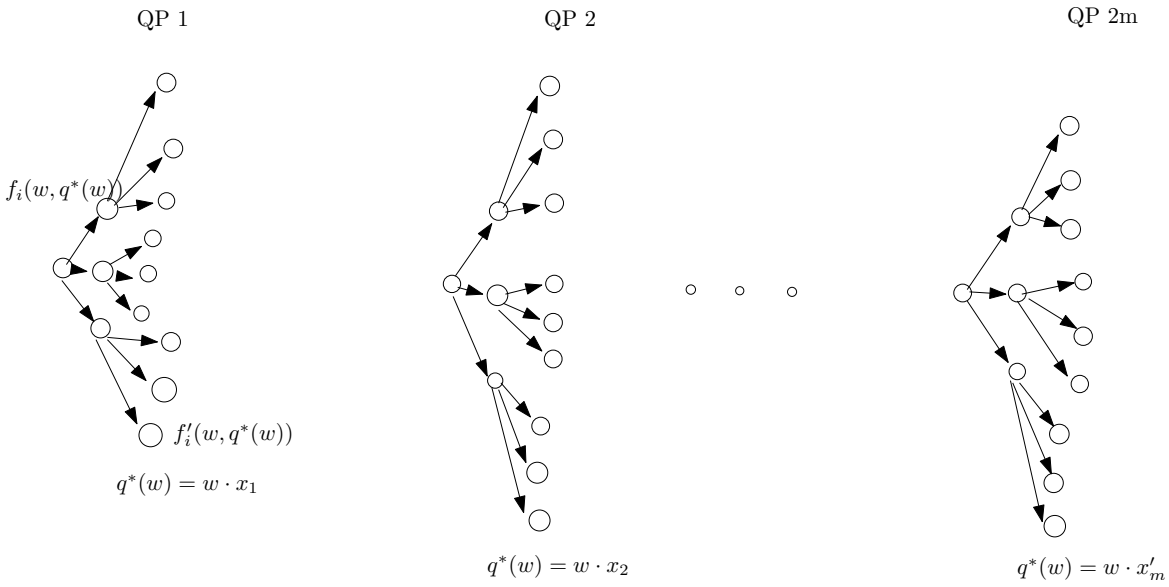

Figure 1: Consensus graph with its connected components organized in tree topology.

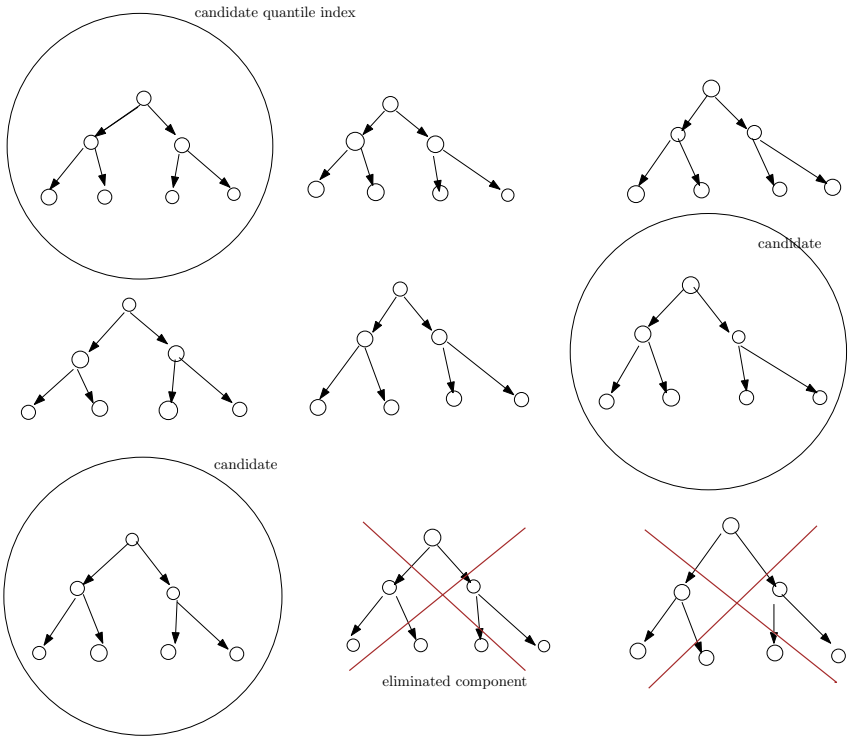

Figure 2: Iterative sampling of candidate components for optimization runs.

the convergence. Another parameter is the positive factor in the augmented Lagrangian term which plays a significant role in updating the variables. Finally, having some prior knowledge of the primal variables is exploited in the form of the warm start of the optimization run.

In order to further increase the potential size of the training set, we implement an *iterative optimization procedure*, that randomly selects a *small* subset of candidate graph components and, using the obtained results, eliminates connected components that are unlikely candidates for the optimal

scoring function. Thus, instead of running a full-blown optimization where the number of connected components equals to the number of the input instances, we run a small number of optimizations at a time, and learn unlikely candidates. After each optimization run, we update the best quantile candidate, and using its weights, warm-start the next iteration (see Figure 2).

References

[1] S. Agarwal. The infinite push: A new support vector ranking algorithm that directly optimizes accuracy at the absolute top of the list. In *Proceedings of the SIAM International Conference on Data Mining*, 2011.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] S. Boyd, C. Cortes, M. Mohri, and A. Radovanović. *Accuracy at the Top*. In *NIPS 2012*, to appear.

[5] J. S. Breese, D. Heckerman, and C. M. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1998.

[6] G. Malewicz, M. Austern, and A. Bik, J. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: A System for Large-Scale Graph Processing. In *SIGMOD '10*, Indianapolis, Indiana, USA, 2010.

[7] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM.

[8] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In *NIPS*, pages 193–200, 2006.

[9] D. Cossock and T. Zhang. Statistical analysis of Bayes optimal subset ranking. *IEEE Transactions on Information Theory*, 54(11):5140–5154, 2008.

[10] K. Crammer and Y. Singer. PRanking with ranking. In *Neural Information Processing Systems (NIPS 2001)*. MIT Press, 2001.

[11] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4, December 2003.

[12] R. Herbrich, K. Obermayer, and T. Graepel. *Advances in Large Margin Classifiers*, chapter Large Margin Rank Boundaries for Ordinal Regression. MIT Press, 2000.

[13] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM.

[14] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, pages 377–384, 2005.

[15] Q. V. Le, A. Smola, O. Chapelle, and C. H. Teo. Optimization of ranking measures. Unpublished, 2009.

[16] C. Rudin, C. Cortes, M. Mohri, and R. E. Schapire. Margin-based ranking meets boosting in the middle. In *COLT*, pages 63–78, 2005.

[17] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.