A scalable parallel boosting scheme for bulk synchronous parallel environments

Sándor Kazi and Gábor Nagy

Under the pressure of the several V-s (volume, velocity, variety, etc.) of BigData [1] a common approach is to use distributed computation frameworks on top of distributed filesystems. The open source flag-carrier of this approach is Hadoop which has transformed from an open-source project to a widely applied business solution in the last few years. Machine learning possibilities on top of Hadoop is basically identified by the somewhat limited capabilities of Mahout, a machine learning library for Hadoop, and the programming capabilities of a user. However MapReduce has disadvantages when it comes to iterative task execution [2]. Therefore a large set of machine learning algorithms call for a different approach. A possible way to implement distributed iterative machine learning algorithms on top of the Hadoop infrastructure (overcoming the limitations mentioned before) is to use the BSP computation model (designed by Leslie Valiant [3] in the early '90s and revisited by Google in 2010 [4]). This model is supported by two Hadoop packages now: Apache Giraph (mainly for graph processing) and Apache Hama. If any dimension of the data (including its velocity, etc.) is too much for one node to handle these distributed frameworks can provide scalable parallel implementation possibilites.

We hereby present a scheme to create distributed versions of a boosting algorithm using the BSP model. The idea of boosting comes from the task of training a set of weak learners to form a strong learner, a popular representative of this meta modeler group is Gradient Boosting which uses a gradient based method to calculate the new labels for each weak learner to use for training. One of the most (or the most) common weak learners used with gradient boosting are Decision Trees, the abbreviation GBDT refers to this construction [5, 6]. To introduce the scheme we use GBDT in our demonstrations, although the approach can be similarly applied for some other boosting algorithms.

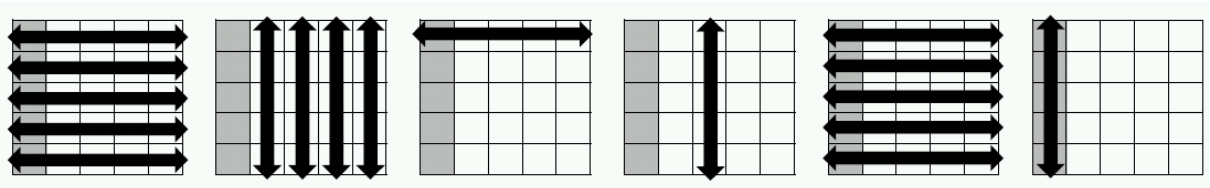

Figure 4: A schematic representation of the parallel GBDT training process.

The data is presumed to be distributed among multiple nodes, several factors are considered to ensure efficent execution of the GBDT training. Some of the nodes should have one of the two special roles, the role setting can be mapped to the data distribution setting. A schematic representation of the parallel GBDT training process is presented on figure 4: each row/column represents nodes having parts of the same data row/column. Residuals are distributed in the first step, statistics are gathered in each column in the second, the best split for each column is broadcasted and the global best split is selected afterwards. Then the leaf is updated using the new split description at the nodes having that column. If the tree is not completed yet we restart from statistics gathering (but for the new leafs only) otherwise the model is updated with the current tree (with appropriate multipliers calculated for each leaf). Every new iteration of GBDT starts with a new residual calculation step. Most of these can be executed in parallel and can be efficiently balanced to minimize waiting times by (even dynamic) distribution of the data.

The efficiency of the parallel setting can depend on the properties of the dataset, the distribution setting and the loss function. Communication costs in this scheme can be and should be minimized, this can be done in several ways. Some of these methods can guarantee the model to be the same as it would be in a non-distributed approach, some others (like the histogram setting of Ben-Haim and Tom-Tov [7]) are approximations but operate without reasonable degradation in efficiency [8].

Acknowledgements

This work is supported by the grant: FUTURICT, TÁMOP-4.2.2.C-111KONV, "Financial Systems" subproject.

References

[1] D. Laney (2001-02-06). *The Importance of 'Big Data': A Definition*. Gartner.

[2] K. Lee, Y-J. Lee, H. Choi, Y. D. Chung, and B. Moon. Parallel data processing with mapreduce: a survey. *SIGMOD Rec.*, 40(4):11–20, January 2012.

[3] L. G. Valiant, "A bridging model for parallel computation," *Commun. ACM*, vol. 33, no. 8, pp. 103–111

[4] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: a system for large-scale graph processing," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, ser. SIGMOD '10. New York, NY, USA: ACM, 2010, pp. 135–146.

[5] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics and Data Analysis*, vol. 38, pp. 367–378, 1999.

[6] J. H. Friedman, "Greedy function approximation: A gradient boosting machine,"*The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 10 2001.

[7] Y. Ben-Haim, E. Tom-Tov, "A Streaming Parallel Decision Tree Algorithm," *The Journal of Machine Learning Research*, vol. 11, 3/1/2010, pp. 849–872

[8] S. Tyree, K. Q. Weinberger, K. Agrawal, and J. Paykin, "Parallel boosted regression trees for web search ranking," in *Proceedings of the 20th International Conference on World Wide Web*, ser. WWW '11. New York, NY, USA: ACM, 2011, pp. 387–396.