4th National Conference On Emerging Trends In Computer Science & Engineering (NCETCSE-2018)

Construction of Regression Trees by Using SAIRT and GBRT

Ashwini M[1], Kavyashree C M[2], Reshma M S[3],Sahana C M[4]

Niveditha N M[5], [1,2,3,4] UG Students, [5] Assistant professor

Department of Computer Science and Engineering

BGS Institute of Technology, BG Nagar

Abstract-We introduce a new algorithm for incremental construction of binary regression trees is presented, This algorithm is also called as an SAIRT ,adapts the induced model, when facing data streams involving unknown dynamics like gradual and abrupt function drift, this regression trees which also handles both symbolic and numeric attributes on this condition current regression methods need a careful configuration depending on the dynamics of the problem, the proposed algorithm may adapt to particular complete changes of the target underlying regression function, because it expands or prunes subtree and automatically adjust its internal parameters to improve a local performance measure in each node. The another parallel boosting trees is Gradient Boosted Regression Trees (GBRT) are the current state-of –the-art learning paradigm for machine learned web search ranking a domain notorious for very large data sets, In this paper We propose a never method for parallelizing the training of GBRT, Our techniques parallelizes the construction of the individual regression trees and operators using the master processor uses these to build one layer of regression trees.Since this approach is based on data partitioning, and requires a small amount of communication.

Keywords—Ranking, Boosted, Boosted regression trees, Gradient Boosted Regression tree, Sub tree

I. INTRODUCTION

In a huge and potentially infinite volumes of data are often continuously generated by real-time system like management communications networks. Online transaction in the financial market or real industry. Scientific and engineering experiments, surveillance systems and other dynamic environments, producing data streams in order to achieve better productivity learning algorithms are able to construct models from these data containing the hidden underlying relations between the independent feature and the dependent one. Traditional learning algorithms can be executed in batch mode in order to obtain models representing relations of data. When Traditional learning algorithm are gradually changes, When dealing with dynamic data streams are derived from complex algorithms, the underlying relations between variables, may change over time, these changes will not always occur with the same speed. Sometimes they are faster, and others are slower, when changes occur will also occur partially or totally.

This can be illustrated making use of a simple univariate function as the underlying relation to be learned. Suppose that the initial function in the data stream is a straight line. After sometime the function moves up by incrementing its interception in some small this process is repeated several times until the function reaches the final position. A well known technique to learn models in these conditions is to forget past examples and take into account new data, updating the model accordingly, with regards to the forgetting mechanism; learning algorithm may use local or global windows, with fixed or adaptive size. Windows contain the information about recent data from data stream. Since our proposal is an incremental regression tree, each branch from root of the tree to a leaf Represents a region of the function. The purpose of the presented method called SAIRT (self Adaptive Induction of regression tree is deal with tasks involving unknown dynamics. and also its deals with a learning algorithms. which leaf maintains a local window.

In this parallel boosted regression tress for web search ranking. It's a world wide web from an initial experiment at CERN to a global phenomenon. In this time, web search engines have come to play an important role in how users access the internet and have seen tremendous advances, a crucial part of a search engine is the ranking function, which order to retrieves documents according to decreasing relevance to the query. Recently web search ranking has been recognized as a supervised machine learning problem. Where each query-document pair is represented by a high-dimensional feature vector and its label indicates the documents degree to relevance to query. In GBRT is angularly defines the current states of the art. Due to the increasing amount of available data and the ubiquity of multicourse and clouds, there is increasing interest in parallelizing machine learning algorithms. In this paper, we take the opposite approach and parallelize the construction of individual weak learners.

In our approach, the algorithm works step by step by constructing one layer of the regression tree at a time. One layer of the regression tree at a time, one processor is designated the master processor, and the others are the workers, the data which are portioning to the weaker, at each steo the workers compress their portion of the data into small histograms and send this histograms to the master, its then communicates this layer to the workers, which allows them to compute histograms for the construction of subsequent lays. The constructing steps when a predefined depth is reached. This master-workers approach with bounded communication has several advantages,

1. It can be generalized to multicourse, shared memory machines, clusters and clouds with relatively little efforts,

2. The data is portioned among users.

II. RELATED WORK

In a traditional task of stationary regression learning (i.e., approximation of a function from a static database) several algorithms have been provided, construction of regression models from a stream of data in which different unknown dynamics may occur have received less attention from the community.

On the other hand, the field of concept learning on data streams has been an effervescent area in the recent past, even when changes are present. The works by Domingo's and Hulten and Nunez˜ et al. are notables when dealing with dynamic data streams for a classification. The following paragraphs try to Resume the current state of the art in the field of regression induction from dynamic data streams by remarking on the most important characteristics (from our point of view) of algorithms

That faces these problems:

1) Memory management

2) Adaptation of internal parameters.

Another interesting aspect is the management of the internal parameters for dealing with real applications. This allows the algorithm to learn from different conditions without user action. In this paper, present a sample of previous work on parallel machine learning most related to our work. The related work falls into three categories:

1) Parallel decision trees

2) Parallelization of boosting

3) Parallelization of web search ranking using other approaches such as bagging.

Parallel decision tree algorithms have been studied for many years, and can be grouped into two main categories:

1) task-parallelism

2) data-parallelism

Algorithms in the first category has divide the tree into sub-trees, which are constructed on deferent workers, e.g. after the first node is split, the two remaining sub-trees are constructed on separate workers. There are two downsides of this approach. First, each worker should either have a full copy of the data or large amount of data has to be communicated to workers after each split. Therefore, for large data sets, especially if the entire data set doesn't fit in each worker's memory, this scheme would likely provide slowdown rather than speedup. Second is and a small trees are unlikely to get much speedup, since they cannot utilize all the available workers.

This algorithm is most similar to BenHaim and YomTov's work on parallel approximate construction of decision trees for classification. Our histogram methods were largely inspired by their publication. However, our approach diers in several ways. First, we are using a regression trees instead of classification requiring us to interpolate relevance scores within histogram bins instead of computing one histogram per label. Further, our method explicitly parallelizes gradient boosted regression trees with a fixed small depth. This pGBRT algorithm obtains more speed-ups on larger data sets, as the parallel scan of the data to construct histograms takes a larger fraction of the overall running time.

Finally, multiple approaches have been applied bagging to web-search ranking. Recent work by Pavlov and Brunch [24] uses bagged boosted regression trees. Bagging is inherently parallel but requires additional computation time as it is averaged over many independent runs (Pavlov and Brunk used a total of M = 300,000 trees of depth d = 12 in the Yahoo Learning to Rank Challenge). Usually, the choice between bagging and boosting is based on desired learning paradigm rather than computational resources.

III. DESCRIPTION

3.1. SAIRT learning algorithm

```
PROCEDURE SAIRT (tree, example)
Node = Root (tree)
Visited nodes = {node}
WHILE! Is Leaf (node) AND Incoherent (node) DO node = Next Child (node, example)
Visited nodes = visited nodes ∪ {node} ENDWHILE
IF Is Leaf (node) THEN - node is called leaf –
Store (example, leaf)
Update Statistics ({leaf})
Node = Try Expansion (leaf)
IF Is Leaf (node) THEN AdaptLocalWindow ({leaf})
ENDIF
ELSE–node is not coherent–
Drop (example, node)
AdaptLocalWindow (Degraded Leaves (node))
Leaf = Prune (node)
Update Statistics ({leaf})
Try Expansion (leaf)
ENDIF
Update Statistics (visited nodes)
```

SAIRT is an algorithm for incremental induction of bi-nary regression trees, which also supports adaptability to gradual, and abrupt function drift, the handling of symbolic and numeric attributes, and robustness to noise and virtual drift in data. Depending on the dynamics of the problem this method expands or prunes sub trees and adjusts its internal parameters to improve the local performance measure in each node. SAIRT develops a partial memory management; that is to say, it selectively forgets examples and stores the remaining ones in local windows present in the leaves of the tree. This method is designed to be used under unknown dynamics, and thus it is able to automatically adapt its parameters for each problem.

3.2. Gradient Boosted Regression Trees

```
Input: data set $D = \{(x_i, y_i)\}^n_{i=1}$, Parameters: $\alpha$, m, d
Initialization: $r_i = y_i, \forall i$
$C(\cdot) = 0$
for t = 1 to m do
$g_t \leftarrow O(\{(x_i, r_i\})$
$C(\cdot) \leftarrow C(\cdot) + \alpha g_t(\cdot)$
for i = 1 to n do
$r_i \leftarrow - \frac{\partial}{\partial h_t(x_i)}$
end for
end for
return C
```

In traditional GBRT algorithm, spends the majority of its computation time evaluating split points during the creation of regression tree. we speed up and parallelize this process by summarizing label and feature value distributions using histograms. Here we describe how a single split, evaluated on the data that reaches the particular node, is computed using this histograms.

3.3 Parallel cart algorithm Parameter: maximum depth tree ← unlabeled node

while tree depth < maximum depth do for each feature do Instantiate an empty histogram at each leaf for each worker do Initiate non-blocking receive for worker's histograms end for while non-completed receives do wait for some receive to complete merge received histograms at leaves end while update best splits for each leaf end for send next layer of leaves to each worker

end while

return tree

3.4 Parallel CART worker

```
Input: data set $D=\{x_i,r_i\}_i^n=1$
Parameter: maximum depth
Tree←unlabeled node
While tree depth < maximum
depth do Navigate training data D
to leaf nodes v For each feature f
do merge($h_{vi}$,($[x_i]_f$,1,$r_i$))
end for
initiate non-blocking send for histograms from all
leaves end for
receive next layer of leaves from
master end while
```

In our algorithm we have used master processor and P workers we assume the data divided into P disjoint subset and stored in different physical locations in each worker can access one of these locations. Then the master processor creates the regression trees in layer by layer at each iteration a new layer is constructed. By using histograms the worker compresses its share of the data and it sends to the master processor. Then the master merges the histograms and uses them. We select the best splits for each leaf node, their constructing new layer. then this new layer sends to the worker. Then worker constructs the histograms for the new layer. So the communication consists entire of the worker sending histograms to the master and master sending the new layer of the tree since the depth of the regression tree is small.

IV. EXPERIMENTAL RESULTS

In this section presents the results of the proposed algorithm when facing different kinds of problems, from classic (stationary) datasets to data streams where the underlying knowledge changes. To compare results, we have chosen several well-known algorithms based on different techniques on ma-chine learning and data mining: M5' [32] is a regression tree-based learning algorithm that extends M5 and is used when facing problems without changes in the function to induce; IBk is a nearest neighbour algorithm used for both classification and prediction tasks; Linear-Regression tries to obtain a linear model to approximate the underlying function; finally, MLP is a multilayer perception-based algorithm. We used a suite for data mining called Weka [32] to obtain results with the algorithms described above. In addition, the algorithm FIRT-DD, a regression tree-based algorithm able to ex-tract knowledge from time-changing data streams, was also used in the comparisons of this section. Otherwise stated, default parameters for all methods are used on all the experiments. For each experiment, we will collect measurements of error rate (specifically, Root Mean Squared Error, RMSE), memory consumption and time devoted to processing one example. With the aim of knowing when SAIRT reacts to changes in functions when there are not (false alarms), an additional metric, called estimated rate of function change (erfc), is collected for each experiment. It is calculated with the arrival. In this section, we describe the empirical evaluation of our algorithm using two

Fig 1:ERR and NDCG for yahoo set 1(top) and yahoo set 2 (bottom) on parallel (pGBRT) and exact (GBRT) implementation with various tree depths d.The NDCG plot for set 1 (top right) shows nicely that pGBRT with a tree depth d+1 leads to results similar to the exact algorithm with depth d.

Table 1: Statistics of the yahoo competition and Microsoft learning to rank data set

	Yahoo LTRC		MSLR MQ2008 Folds				
TRAIN	Set 1	Set 2	F1	F2	F3	F4	F5
# Features	700	700	136	136	136	136	136
# Documents	473,134	34,815	723,412	716,683	719,111	718,768	722,602
# Queries	19,944	1266	6000	6000	6000	6000	6000
Avg # Doc per Query	22.723	26.5	119.569	118.447	118.852	118.795	119.434

TEST	Set 1	Set 2	F1	F2	F3	F4	F5
# Documents	165,660	103,174	241,521	241,988	239,093	242,331	235,259
# Queries	6983	3798	6000	6000	6000	6000	6000
Avg # Doc per Query	22.723	26.165	119.761	119.994	118.547	120.167	116.6295

publicly available web search Ranking data compilations. We see impressive speedups on both shared memory and distributed memory machines. In ad-dition, we found that, while the individual regression trees are weaker using our approximate parallel algorithm (as expected), with appropriate parameter settings, the final regressor did not lose much accuracy. In some cases, our parallel implementation generates a regressor that is just as good as the sequential implementation, while in others, it was slightly less accurate. We conducted experiments on a parallel shared memory machine and a distributed memory cluster. The shared memory machine is an AMD Opteron 1U-A1403 48-core SMP machine with four sockets contain-ing AMD Opteron 6168 Magny Cours processors. The distributed memory cluster consists of 8-core, Nehalem based computing nodes running at 2.73GHz. They each have 24GB of RAM. For our experiments, we used 6 of these nodes (with a total of 48 cores). For our empirical evaluation, we use the two data sets from Yahoo! Inc.'s Learning to Rank Challenge 2010 [9], and the five folds of Microsoft's LETOR [21] dataset.

Each of these sets come with predefined training invalidation and test sets. Table 1 summarizes the statistics of these data sets. The above Figure shows the ERR and NDCG of the parallel implementation "pGBRT" and of the exact algorithm "GBRT" as a function of the number of boosting iterations on the Yahoo Set 1 and 2 under varying tree depths. For the parallel implementation, we used b = 25 bins for Set 2 and b = 50 for the much larger Set 1. The step-size was set to α = 0.06 in both cases. As expected, the histogram approximation reduces the ac-curacy of the weak learners. Consequently, with equal depth and iterations, pGBRT has lower ERR and NDCG than the exact GBRT. However, we can compensate for this effect by either running additional iterations or increasing the depth of the regression trees. In fact, it is remarkable that on Set 1 (Figure 1) the NDCG curves of pGBRT with d = 6 and d = 5 align almost perfectly with the curves of GBRT with d = 5 and d = 4, respectively. For Set 2 the lines are mostly shifted by approximately 200 iterations. The additional computation required by either of these approaches (increasing d or m) is more than compensated for by the better performance of the

histogram method since it does not require feature sorting. (For small $d \leq 10$ – while the computation is dominated by while the computation is dominated by computation – the running time increases roughly linearly with increasing d. On the Yahoo Set 1, training pGBRT with m = 6000 trees on 16 CPUs and depth d=5 was only a factor 1.34 slower than d = 4 and a depth of d = 6 slowed the training time down by a factor of 1.75.)

V. CONCLUSION

As far as we know, there is no other algorithm able to construct and adapt regression trees to data streams whose underlying function changes over time as well as having other dynamics present without an a-priori parameterization. In order to face a wide spectrum of tasks, systems and users, we think that learning algorithms should be as simple to use as possible and that the models generated should be understandable. This has been a principal motivation in the design and evaluation of the SAIRT algorithm. The way in which the knowledge is represented helps to understand both the patterns discovered in each moment and the problem itself. We have presented a parallel algorithm for training gradient boosted regression trees. To our knowledge, this is the first work that explicitly parallelizes the construction of regression trees for the purpose of gradient boosting. We have shown that our approach provides impressive speedups on several large-scale web-search data sets without any significant sacrifice in accuracy.

Our method applies to multicore shared-memory systems as well as to distributed setups in clusters and clouds. Since each processor only needs enough physical memory for its partition, and the communication is strictly bounded, this allows the training of machine-learned rankers on web-scale data sets even with standard off-the-shelf computer hardware. We are planning to extend this work in several directions. First, we think we can further increase the efficiency and performance by eliminating the master and merging histograms pair wise among workers. In addition to freeing the master processor for useful work, this approach would further overlap computation and communication. Second, we are planning to run experiments with more workers on clouds to gauge the of this approach on non-dedicated machines. Third, we intend to investigate more aggressive speed vs. accuracy trade in the computation of the splits based on stochastic approximations of the histograms. Given the current trend towards multi core processors, parallel computing and larger data sets, we expect our algorithm to increase in both relevance and utility in the foreable future.

REFERENCES

[1] Raul Fdalfo-Merino, and Marlon Nunez “ self adaptive induction of regression tree”

[2] Stephen Tyree and Kunal Agrawal “parallel boosted regression trees for web search Ranking”.

[3] D.Pavlov and C.Brunk. Bagboo: Bagging the gradient boosting. Talk at Workshop on Web search Ranking at the 27th International Conference on Machine Learning,2010.

[4] M.Nunez, R.Fidalgo, and R.Morales, “On-line learning of decision tree in problems with unknown dynamics,” in proceeding of the 4th Mexicon International Conference on Artificial Intelligence, 2005.