IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-3, Issue-10 E-ISSN: 2347-2693

Comparative Analysis of Cluster based Boosting

Nilam Kolhe[1*], Harshada Kulkarni[2], Ishita Kedia[3] and Shivani Gaikwad[4]

[1*,2,3,4]*Department, Of Computer Engineering,* K.K.Wagh College Of Engineering & Research, Savitribai Phule Pune University, Maharashtra, India

www.ijcseonline.org

Received: Sep/17/2015 Revised: Oct/01/2015 Accepted: Oct /16/2015 Published: Oct /31/ 2015

Abstract -Clustering focuses on grouping similar objects in one cluster and dissimilar objects into another cluster. In clustering, this concept of boosting applies to the area of predictive data mining to generate multiple clusters. There is an existing cluster based boosting(CBB) system which focus on real data sets applied to it as input. It uses K-means algorithm that evolved in limited number of clusters with over fitting and it also holds two limitations: 1.Subsequent functions ignoring troublesome areas 2.Complex subsequent functions. To overcome these drawbacks hierarchical clustering is proposed and thus enhances the accuracy of desired output of CBB approach compared to popular boosting algorithm. The comparative analysis may show the improvement in performance of the system. The users may obtain refined clusters with more accuracy as desired output.

Keywords— : Boosting, Clustering, Hierarchical clustering, Classifier combining, Machine Learning, Supervised learning, Computer graphics, Artificial intelligence.

I. INTRODUCTION

This project idea is based in data-mining domain. In this project over-fitting problems in clusters is focused nd using boosting technique refined clusters are achieve in dataset, containing noisy labels.

Clustering is the process in which relevant data is put together with the system learning technique. These clusters sometimes may contain irrelevant data due to noise in training dataset. This noise can be wrong labels in dataset or mismatched features of members as far as other members are concern. This motivates us to work on refinement of clusters using boosting technique. Also with boosting [9]and clustering system should achieve noiseless and relevant clusters.

Here we further discuss our cluster-based boosting solution. The main strategy for CBB is to incorporate clusters created on the training data directly into the boosting process using these clusters and the initial function to learn the subsequent functions[1]. First, the clusters created provide additional structure for the subsequent functions since these clusters include both correct and incorrect instances from previous functions. This structure helps to mitigate the filtering problem in subsequent functions. Next, these clusters are designed to break up the training data into different areas since each cluster encapsulates only instances with a high degree of similarity. These separate areas help to mitigate overfitting in subsequent functions[3]. One explanation for why boosting has problems is the way it learns subsequent functions. These functions are trained focusing on all the incorrect instances in the training data where the initial function did not predict the correct label. This additional training forces subsequent functions to accommodate highly dissimilar training data. This can result in subsequent functions with an increased complexity and likelihood of overfitting. At the same time, the training process for these subsequent functions tends to ignore problematic training data on which the initial function predicted the correct label. This can result in important information withheld from subsequent functions such as the labels for correct instances that are highly similar to the incorrect instances. To address the limitations of boosting, we hereby propose a novel cluster-based boosting approach that incorporates clusters into the boosting process to improve how boosting learns these subsequent functions. Our CBB approach partitions the training data into clusters that contain highly similar member data to break up and localize the problematic training data. CBB then uses these clusters integrated into boosting to improve the subsequent functions as opposed to previous work that has used clusters only for preprocessing. First, CBB evaluates each cluster separately to identify whether the problematic training data should be used to learn subsequent functions. This allows for more selective boosting to accommodate different types of problematic training data. Next, CBB learns subsequent functions separately on each cluster using only the member data in that cluster. This allows for less complex subsequent functions and helps to mitigate overfitting from being propagated into boosting. Last, CBB learns subsequent functions starting with all the cluster members—not just those deemed incorrect by the initial function. This allows for more inclusive boosting that can accommodate problematic training data deemed correct.

The rest of this paper is organized as follows. Section 2 provides the literature survey on boosting and related work on using clustering and boosting.

Section 3 provides a more in-depth discussion on the proposed CBB system . Section 4 concludes and discusses future work.

II. LETURATURE SURVAY

There exists a large body of previous work that demonstrates the effectiveness of boosting. Theoretical results have shown that boosting is resistant to overfitting the common SL problem where the algorithm overspecializes on nuances in the training data to the degree that predictive accuracy on new data instances is reduced [1].

Furthermore, empirical results on a wide variety of existing data sets have shown that boosting generally achieves higher predictive accuracy than using a single function from the same SL system [2]. In addition to benchmark data sets, boosting has also been used effectively on a wide range of applications [3].

There are many different strategies for computing clusters. Here, we use the centroid-based k-Means because of its proven effectiveness and popularity. However, there is no best clustering algorithm on all data sets. However, on real-world data sets with numerous, irrelevant features, the clusters may be distorted to the point that they become no longer useful for selective boosting, and thus reducing the effectiveness of CBB. We intend to investigate both applying feature selection before clustering and using semi-supervised during CBB to address this possible limitation [4].

The initial function failed to predict the label correctly for certain instances, not because the initial function learned was incorrect, but because these instances were labeled wrong to begin with. However, boosting does not realize that the labels were wrong and, thus, holds the initial function responsible. As a result, boosting focuses subsequent functions on learning how to "correctly" predict these instances assuming that the wrong labels provided are correct [5].

III. PROPOSED SYSTEM

The existing CBB system focused on real data sets applied to the it. It used K-means algorithm which needed number of clusters to be specified thereby replacing it with X means algorithm. The proposed system used hierarchical clustering for cluster formation having advantages:

- It does not need centroids to be specified.
- Improves accuracy of clusters by forming hierarchical tree depending on the level of resolution.

The hierarchical clustering is used along with boosting for enhancing the accuracy of CBB system and removing label noise, overfitting and functions complexity. We will provide a comparative analysis of clustering algorithms based on which we shows hierarchical clustering is more efficient[13].

The main strategy for CBB is to incorporate clusters created on the training data directly into the boosting process using these clusters and the initial function to learn the subsequent functions.[7] First, the clusters created provide additional structure for the subsequent functions since these clusters include both correct and incorrect instances from previous functions. This structure helps to mitigate the filtering problem in subsequent functions. Next, these clusters are designed to break up the training data into different areas since each cluster encapsulates only instances with a high degree of similarity. These separate areas help to mitigate overfitting in subsequent functions.

(A)Cluster Creation:

Our CBB solution is based on unsupervised clustering that tries to decompose or partition the training data into clusters where the member instances in a cluster are similar to each other and as different as possible from members in other clusters. There are many different strategies for creating clusters on the training data. Probably the most popular strategy for clustering is k-Means (centroid-based) that assigns training data to the cluster to minimize the distance between each member and the cluster center. The CBB solution uses k-Means in this paper to establish that clustering can improve boosting in a general way.

The goal for k-Means clustering is to assign each instance to the cluster that minimizes the following objective function :

$$\sum_{c=1}^{k} \sum_{\chi_i \ni \pi_c} \|\chi_i - m_c\|^2 \quad ....(1)$$

where χ_i is the instance, π_c is the cluster, m_c is the cluster centroid, and norm squared is the distance between the member instance and the cluster center. (For k-Means, distance and similarity are inversely proportional with zero distance corresponding to perfect similarity.) This objective function is difficult to solve precisely and k-Means clustering usually employs an iterative method where cluster assignments are updated until the distance between the members is minimized.

The principal limitation for k-Means clustering is that the number of clusters used (the k) must be specified beforehand. Our CBB solution addresses the limitation in k-Means clustering by using a modified version called X-

Means that learns the appropriate number of clusters automatically. X-Means starts with the set of clusters from a small k and then dynamically increases k as long as it lowers the Bayesian Information Criterion (BIC) in the new set of clusters:

$$BIC(\pi_c)= |\chi| \ln\sigma^2 + k \ln|\chi| \text{ ....(2)}$$

where χ is all the training data in cluster π_c and σ^2 is the same as the inner summation in objective function. The value of k when BIC is minimal is thus considered the optimal number of clusters for the data set. Note that the BIC metric rewards sets of clusters containing similar members, while penalizing clusters that are too small. In this way, the BIC encourages cluster compactness while discouraging clusters too small to encapsulate meaningful areas.

(B)Learning Subsequent Functions:
CBB uses a modified boosting process that learns subsequent functions selectively on the clusters. CBB consider four different cluster types based on two independent factors: cluster membership (heterogeneous and homogenous) and previous function accuracy (prospering and struggling).these cluster types are given in a descending order of difficulty for the functions.

(a)Heterogeneous struggling : The cluster contains members with different labels and previous functions struggle to predict the correct labels. Since such a cluster generally contains troublesome training data and previous functions have been struggling, CBB uses boosting with a high learning rate (high-eta boosting) on this type—learning subsequent functions focusing on incorrect members until accuracy improves.

(b)Heterogeneous prospering : The cluster contains members with different labels, but previous functions are still able to predict the correct label for most of the members. Since such data is difficult , boosting can still make improvements by refining the final decision boundary, CBB uses boosting with a low learning rate on this type—learning fewer subsequent functions focusing on incorrect members. Homogenous struggling - A cluster contains members with the single label, but the previous functions struggle to predict the correct labels. This type can happen when the previous functions sacrifice these members focusing instead on learning other areas of the training data to achieve the highest accuracy. Since this type is easy for a function to predict (simply by predicting the majority label), CBB learns a single, subsequent function on all members without boosting on incorrect members.

(c)Homogenous prospering : The cluster contains members with predominately a single label and the previous functions already predict the correct label for most of the members. CBB does not learn any subsequent functions on this type to prevent those functions from learning the label noise.

(d)Homogenous struggling : The cluster contains members with predominately a single label, but the previous functions struggle to predict the correct labels. This type can happen when the previous functions sacrifice these members focusing instead on learning other areas of the training data to achieve the highest accuracy. Since this type is easy for a function to predict CBB learns a single, subsequent function on all members without boosting on incorrect members.

The four cluster types are computed using two separate metrics. First, the localized estimate (LE) metric is used to decide whether a cluster is struggling or prospering:

$$LE(\pi_c) = \{\text{prospering if } acc(F,\pi_c) \geq 1- \delta_1, \text{ struggling otherwise}\} \text{ ....(3)}$$

where acc (F, π_c)is the accuracy of the previous functions evaluated only on the cluster members and d1 is a tunable parameter on the range $0.1\leq \delta_1 \leq 0.3$. This range is sensible because 1. a smaller δ_1 (<0.1) would render the typing too strict such that almost all clusters would fall into the struggling category and 2. a larger δ_1 (>0.3) would probably allow us too many borderline struggling clusters to be considered prospering. Second, the minority label estimate decides whether a cluster type is homogenous or heterogeneous by using following function:

$$MLE(\pi_c) = \{\text{homogeneous if } minority(\pi_c) < \delta_2, \text{ homogeneous otherwise}\} \text{ ....(4)}$$

where minority(π_c) is the minority label percentage on the cluster members and δ_2 is a tunable parameter on the range $0.2\leq \delta_2\leq 0.4$.This parameter uses a range to accommodate data sets with varying label distributions. Data sets with a larger skew towards the majority label need a correspondingly smaller threshold. Last, CBB computes the weighted vote for a function using:

$$vote(f_t) = \eta \ln(1-\varepsilon_t/\varepsilon_t) \text{ ....(4)}$$

where (f_t) is the function, η is the learning rate used to control the update of the weights for the incorrect instances, and ε_t is the weighted error on the member data.Tthis vote is also used as the basis for updating instance weights in the boosting probability distribution.

(C)Cluster-Based Boosting Approach :

```
$D = Training\ Data$
$S = Supervised\ Learning\ System$
$F = Function\ Set = \emptyset$
$cluster(D, k)$ // cluster using Eq. (1)
$boost(D, S, \eta)$ // learn multiple functions using eta learning rate
$learn(D, S)$ // learn single function on training data
$type(\pi_c) = HES, HEP, HOS, HOP$ // using Eq. (3) and Eq. (4)
function CBB returns F
(1)  for $k = 2$ to $m$
(2)    $\pi^k \leftarrow cluster(D, k)$ // cluster using Eq. (1)
(3)    $s^k \leftarrow 0$
(4)    foreach $\pi_c^k$ in $\pi^k$
(5)      $s^k \leftarrow s^k + BIC(\pi_c^k)$ // using Eq. (2)
(6)    end foreach
(7)  end for
(8)  $\pi^r \leftarrow argmin_\pi(s)$ // use set with lowest BIC
(9)  $F \leftarrow learn(D, S)$ // single function on all data
(10) foreach $\pi_c^r$ in $\pi^r$
(11)   if $type(\pi_c^r) == HES$
(12)     $F \leftarrow F \cup boost(\pi_c^r, S, 1)$
(13)   end if
(14)   elseif $type(\pi_c^r) == HEP$
(15)     $F \leftarrow F \cup boost(\pi_c^r, S, 0.5)$
(16)   end elseif
(17)   elseif $type(\pi_c^r) == HOS$
(18)     $F \leftarrow F \cup learn(\pi_c^r, S)$
(19)   end elseif // do nothing for HOP
(20) end foreach
```

fig.1 CBB pseudo code

We now discuss our approach for the CBB solution with pseudo code provided in Fig. 1. First lines 1-7, the training data is broken into sets of clusters with varying k where each set of clusters minimizes the objective function. During this process, CBB computes the BIC for the set of clusters at line 5. Second ,CBB chooses the set of clusters with the lowest BIC at line 8. Third, CBB learns the initial function using all the training data using line 9. After clustering, lines 10-20 performs selective boosting based on the cluster type. The cluster type is computed using the localized estimate metric from provided in eq.2 and the minority label metric from provided in eq.3. If the cluster is Heterogeneous Struggling , high-eta boosting has a learning rate on the high end for AdaBoost (η=1) [4]. Otherwise, if the cluster is Heterogeneous Prospering, low-eta boosting has a learning rate on the low end for AdaBoost (η=0.5) . Otherwise, if the cluster is Homogeneous Struggling , a single function is learned without boosting . No functions are learned if the cluster is Homogeneous Prospering to avoid learning label noise. After selective boosting, the set of functions is assigned the weighted vote based on vote function provided in fig.4 and used to predict the labels for a new instance. There are two different ways that these subsequent functions can be used: restricted and unrestricted. Both of course would count the initial function in the voting. Restricted only counts the subsequent functions learned on the cluster to which the new instance would be assigned and disregards votes from other clusters. Unrestricted counts the votes from subsequent functions learned from all the clusters. We use restricted CBB in the rest of this paper because it is more consistent with the proposed selective boosting on each cluster.

Finally we shows the result in the form of tables comparing hierarchical and k-means algorithms based on number of accurate clusters probablity.

IV. CONCLUSION

This system helps to identify the drawbacks in boosting on supervised learning algorithm. Cluster based boosting technique helps to overcome the drawback of weak supervised learning[2]. Clusters formed using proposed technique is having most relevant members and it avoids the over-fitting problem. This approach help to form clusters for noisy datasets. These data sets may contain wrong labels and vague featured members. This proposed approach work on them and form fine grained clusters. This proposed approach work on refinement of the initial function and subsequent functions that selects the member to form clusters.

This CBB approach attempts to address two specific limitations for current boosting both resulting from boosting focusing on incorrect training data. First is filtering for subsequent functions when the training data contains troublesome areas and/or label noise and second is overfitting in subsequent functions that are forced to learn on all the incorrect instances .These limitations are addressed by reducing filteration of subsequent functions, using the appropriate amount of boosting for each cluster. And over-fitting in subsequent functions since they required to learn only the similar member data (correct and incorrect) in a single cluster. We will demonstrate the effectiveness of CBB through extensive empirical results data sets with two different kinds of clustering algorithms. First, we show that CBB achieves superior predictive accuracy using k-means[1], the most popular clustering algorithm. Second, we show that CBB achieves superior predictive accuracy to Hierarchical clustering[11], another algorithm that uses dynamic cluster creation in clustering. The CBB parameters for the localized estimate metric and the minority label estimate metric need to be fine-tuned on each dataset. Fine-tuning these parameters adds complexity to the final solution leading to scalability issues on big data. We intend to investigate how to automatically set these parameters based on the dataset structure and properties.

V. REFERENCES

[1]L. Dee Miller and Leen-Kiat Soh,"Cluster based Boosting", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, Volume-27, Isuue-6,Page No (1-12),June **2015**

[2]C. Zhang and Y. Ma,"Ensemble Machine Learning" New

York, NY, USA: Springer,Page No (76),July 2012

[3]A. Vezhnevets and O. Barinova,"Avoiding boosting overfitting by removing confusing samples", Springer Berlin Heidelberg,Volume-4701,Page No (430–441),**2007**

[4]D.-S. Kim, Y.-M. Baek, and W.-Y. Kim,"Reducing overfitting of adaboost by clustering-based pruning of hard examples", The Korean Society of Broadcast Engineers, Volume-18, Issue-4**,** Page No (643-646),Jul **2007**

[5]M. Okabe and S. Yamada,"Clustering by learning constraints priorities," in Proc. Int. Conf. Data Mining,Page No (1050–1055),**2012**.

[6]A. Ganatra and Y. Kosta,"Comprehensive evolution and evaluation of boosting", Int. J. Comput. Theory Eng.,Volume-2, Page No (931–936),**2010**.

[7]D. Frossyniotis, A. Likas, and A. Stafylopatis,"A clustering method based on boosting", Pattern Recog. Lett.,Volume-25,Page No (641–654), **2004**.

[8]L. Reyzin and R. Schapire,"How boosting the margin can also boost classifier complexity", in Proc. Int. Conf. Mach. Learn.,Page No (753–760),**2006**

[9]R. Schapire and Y. Freund,"Boosting: Foundations and Algorithms", Cambridge, MA, USA: MIT Press, **2012**.

[10] J. Chou, C. Chiu, M. Farfoura, and I. Al-Taharwa, "Optimizing the prediction accuracy of concrete compressive strength based on acomparison of data-mining techniques," J. Comp. Civil Eng.,Volume-25,Page No (242–253), **2011**

[11]David Eppstein,"Fast Hierarchical Clustering and Other Applications of Dynamic Closest Pairs",ACM New York, NY, USA, Volume-5,2005

[12]Y. Freund,"An adaptive version of the boost by majority algorithm,"Mach. Learn.,Volume-43,Page No (293–318), **2001**

[13]Preeti Baser and Dr. Jatinderkumar R. Saini,"A Comparative Analysis of Various Clustering Techniques used for Very Large Datasets",Volume-3,Page No (1-3),Issue-4,March **2013**