A Fast Distributed Classification Algorithm for Large-scale Imbalanced Data

Huihui Wang, Yang Gao*, Yinghuan Shi and Hao Wang
State Key Laboratory for Novel Software Technology, Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing University, Nanjing, China
Email: huihui.wang8917@gmail.com, {gaoy,syh,wanghao}@nju.edu.cn
* Corresponding author:gaoy@nju.edu.cn

***Abstract*—The Alternating Direction Method of Multipliers (ADMM) has been developed recently for distributed classification. Nevertheless, the widely-existing class imbalance problem has not been well investigated. Furthermore, previous imbalanced classification methods lack of efforts in studying the complex imbalance problem in a distributed environment. In this paper, we consider the imbalance problem as *distributed data imbalance* which includes three imbalance issues: (i) *within-node class imbalance*, (ii) *between-node class imbalance*, and (iii) *between-node structure imbalance*. In order to adequately deal with imbalanced data as well as improve time efficiency, a novel distributed Cost-Sensitive classification algorithm via Group-based ADMM (CS-GADMM) is proposed. Briefly, CS-GADMM derives the classification problem as a series of sub-problems with within-node class imbalance. To alleviate the time delay caused by between-node class imbalance, we propose a extension of dual coordinate descent method for the sub-problem optimization. Meanwhile, for between-node structure imbalance, we discreetly study the relationship between local functions, and combine the resulting local variables intra-group to update the global variables for prediction. The experimental results on various imbalanced datasets validate that CS-GADMM could be a efficient algorithm for imbalanced classification.**

I. Introduction

Recently, large-scale classification has been extensively studied in data mining literature [1]. With the explosive growth of the data, distributed algorithms have been proposed for classification [2], [3]. In particular, the Alternating Direction Method of Multipliers (ADMM) is a well-known optimization algorithm that has been used in many applications [4]. By formulating the classification problem as a global consensus problem, ADMM can be used flexibly to solve the problem in a distributed manner. In distributed ADMM, the computing nodes solve small sub-problems by training samples locally in parallel, and aggregate all local variables to update the global variable. Hence, it involves the computation time for optimizing sub-problems, and the communication time between nodes [5], [6]. Although ADMM has a sublinear convergence rate under some mild conditions, distributed ADMM usually converges slowly and suffers from expensive time [7], [8]. Therefore, the development of efficient distributed ADMM algorithms that improve time efficiency is an important problem.

Furthermore, the data coming from different sources usually has unequal class distribution. This situation is known as *classification with imbalanced data* and has gained lots of attention [9], [10]. The common understanding of imbalance in the community is that the number of positive (minority class) samples is significantly less than that of negative (majority class) samples [9]. Nevertheless, the training data encompasses a number of datasets generated and stored in different nodes in a distributed system. In general, there exists the difference in the generation and collection between those datasets.

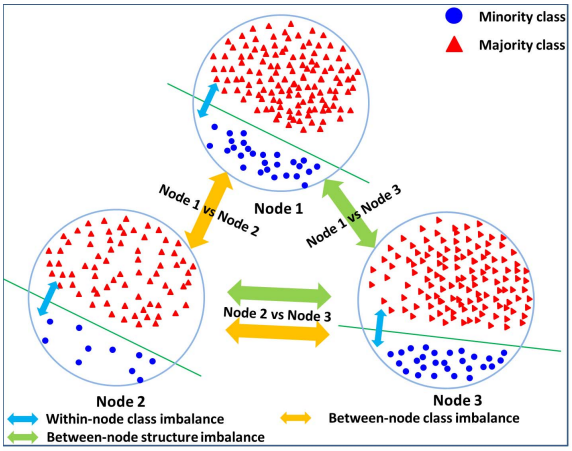

Fig. 1. An illustration of distributed data imbalance. (i) Datasets have the with within-node class imbalance problem on nodes. (ii) There has a between-node class imbalance problem between Node 1 and Node 2. (iii) The between-node structure imbalance issue exists between Node 1 and Node 3. As a whole, distributed data imbalance exists in imbalanced data over a computer network.

In order to highlight the implications of imbalanced learning problems, we consider a more common case about imbalanced data as distributed data imbalance, which includes three imbalance issues: (i) *within-node class imbalance*, where the number of negative samples is much more than that of positive samples on each node, (ii) *between-node class imbalance*, where the number of samples on one node greatly exceeds that on others, and (iii) *between-node structure imbalance*, where the class distribution between datasets may be slightly different. The intuitive illustration is shown in Fig. 1.

In fact, distributed classification with imbalanced data is rather challenging [11]. First, for the first issue, standard imbalanced classification algorithms that train data on a single machine are unable to deal with these huge imbalance datasets.

2374-8486/16 $31.00 © 2016 IEEE
DOI 10.1109/ICDM.2016.73

Second, between-node class imbalance will make different computation time on different nodes, and thus results in the time delay because the training speed is limited by the slowest computing node. Third, for between-node structure imbalance, distributed ADMM need much more iterations to reach a global consensus solution of sub-problems, and thus it is time-consuming. As a consequence, imbalanced classification algorithms are encouraged to be redesigned and adapted for distributed data imbalance so that they can efficiently deal with imbalanced data and improve time efficiency. Unfortunately, this issue has not been well investigated in previous literatures.

To this end, a distributed Cost-Sensitive classification algorithm via Group-based Alternating Direction Method of Multipliers (CS-GADMM) is proposed for imbalanced data. In particular, CS-GADMM decomposes the problem as a series of sub-problems with within-node class imbalance can be solved by cost-sensitive learning. Then, we further propose a extension of the dual coordinate descent method to alleviate the time delay caused by between-node class imbalance. Furthermore, inspired by the recent work in [12], the relationship between local functions is exploited to alleviate the between-node structure imbalance problem. In contrast to distributed ADMM, the local variables of local functions intra-group are combined to update the corresponding local variables and the global variable. The global consensus can be regarded as seeking consensus between the newly updated local variables inter-group and global variable. Hence, CS-GADMM can accelerate the convergence speed and improve time efficiency. The experiments demonstrate that CS-GADMM is effectiveness for imbalanced classification. The main contributions of our work can be summarized into the following three folds:

- A more common imbalance problem, distributed data imbalance is investigated and presented in this paper. In particular, we discuss between-node class imbalance and between-node structure imbalance, and propose corresponding strategies to alleviate these issues, respectively.
- We propose an efficient CS-GADMM algorithm to deal with large-scale imbalanced data, which cautiously leverages the exploitation of distributed data imbalance to improve the classification performance and time efficiency.
- The experiments results on various imbalanced datasets show that CS-GADMM can obtain promising results, and is a highly efficient algorithm for practical use.

The remainder of this paper is organized as follows. In Section 2, we provide and discuss the related work. Section 3 presents our proposed CS-GADMM in detail. The experimental results are reported in Section 4. Finally, Section 5 draws the conclusion and outlines the future work.

II. Related Work

Our work addresses distributed classification with large-scale imbalanced data, which is mainly related to the following research: (i) cost-sensitive classification, and (ii) distributed classification via ADMM.

A. Cost-sensitive Classification

Numerous imbalanced classification methods have been proposed to deal with imbalanced data, which can be roughly summarized into two groups: data-level methods [9], [13], [14], which resample the original dataset to obtain a balanced dataset, and algorithm-level methods [15], [16] that modify the existing classifier to improve the classification performance. In these methods, cost-sensitive learning [17], [18] is a widely-used method for solving the class imbalance and unequal misclassification problem. In recent years, there are a few attempts for studying large-scale imbalanced classification using cost-sensitive learning over a distributed network. A parallel Cost-Sensitive Random Forest algorithm (RF-BigDataCS) was proposed to deal with imbalanced big data [10]. Also, Del Río *et al.* proposed a Cost-Sensitive Fuzzy Rule Based Classification System (Chi-FRBCS-BigDataCS) [19], in which the penalized cost-sensitive certainty factor was applied for imbalanced classification. The above algorithms usually suffer from expensive time cost under the MapReduce framework. Moreover, these algorithms only focus on the within-node class imbalance issue, without involving between-node class imbalance and between-node structure imbalance.

B. Distributed Classification via ADMM

ADMM has attracted much attention since it is particularly suitable for distributed computation [20]. Several distributed algorithms via ADMM have been proposed for cost-insensitive classification [3], [6]. In particularly, Zhang *et al.* proposed distributed classification algorithms via ADMM to solve linear SVM problems in parallel [3]. Also, distributed ADMM has been extended to deal with structured SVM [21]. Distributed ADMM usually converge slowly, and suffer from expensive time in practice [7], [8]. To accelerate the convergence speed, Wang *et al.* recently proposed a group-based ADMM (GADMM) method for distributed classification [12]. In GADMM, a group layer was utilized to divide nodes into several groups, and then group variables are coordinated to update the global variable. The advances on GADMM motivate the present work. However, those methods lack of efforts in studying distributed data imbalance and applying for imbalanced classification. In this paper, we study and present distributed cost-sensitive algorithm via GADMM to deal with imbalanced data.

There are other issues related to the communication time and computation time in distributed learning and optimization, e.g., synchronization vs asynchronization and star network vs arbitrary network [8]. These aforementioned issues are beyond the scope of our work and can be investigated in our future work. In this paper, we focus on the synchronous distributed ADMM algorithms for imbalanced data in a star network.

III. Our Algorithm

The core idea of CS-GADMM is that we cautiously study distributed data imbalance and propose efficient strategies to improve the classification performance and time efficiency. In the following, the details will be presented.

A. CS-GADMM Framework for Imbalanced Classification

We now apply CS-GADMM to distributed classification with imbalanced data. Given the data $\{(\mathbf{x}_i, y_i) \in \mathbb{R}^n \times \{-1,+1\}\}_{i=1}^{l}$, we use cost-sensitive support vector machine (CSSVM) as the classification model. Let $D^- = \{\mathbf{x}_j | y_j = -1\}$ and $D^+ = \{\mathbf{x}_j | y_j = 1\}$, CSSVM solves the primal problem as:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C(c_p \sum_{\mathbf{x}_j \in D_i^+} \xi_j + c_n \sum_{\mathbf{x}_j \in D_i^-} \xi_j), \tag{1}$$

where $C>0$ is the penalty parameter, ξ_j is the misclassification loss of sample $\mathbf{x}_j$. c_p and c_n are the misclassification cost parameters for positive and negative classes. Assume all samples are stored in N nodes, the objective is reformed as:

$$\begin{aligned} \min_{\mathbf{w}_i,\ldots,\mathbf{w}_n,\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{z}\|^2 + C\sum_{i=1}^{N}(c_p \sum_{\mathbf{x}_j \in D_i^+} \xi_j + c_n \sum_{\mathbf{x}_j \in D_i^-} \xi_j), \\ s.t. \quad & \mathbf{w}_i = \mathbf{z}, i = 1, 2, \ldots, N, \end{aligned} \tag{2}$$

where D_i is the dataset in node i, the constraint is used to ensure all local variables $\mathbf{w}_i$s reach a consensus with the global variable $\mathbf{z}$. Basically, the augmented Lagrangian for Eq. (2) can be mathematically formulated as:

$$\begin{aligned} L_\rho(\mathbf{w}, \mathbf{z}, \boldsymbol{\lambda}) = & \frac{1}{2}\|\mathbf{z}\|^2 + C\sum_{i=1}^{N}(c_p \sum_{\mathbf{x}_j \in D_i^+} \xi_j + c_n \sum_{\mathbf{x}_j \in D_i^-} \xi_j) \\ & + \sum_{i=1}^{N}(\boldsymbol{\lambda}_i^T(\mathbf{w}_i - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{z}\|^2), \end{aligned} \tag{3}$$

where $\rho>0$ is the penalty parameter, $\boldsymbol{\lambda}_i$ is the dual variable. The augmented Lagrangian can be rewritten in a simple form:

$$L_\rho(\mathbf{w}, \mathbf{z}, \mathbf{u}) = g(\mathbf{z}) + \sum_{i=1}^{N}(f_i(\mathbf{w}_i) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{z} + \mathbf{u}_i\|^2), \tag{4}$$

where $f_i(\mathbf{w}_i) = C(c_p \sum_{\mathbf{x}_j \in D_i^+} \xi_j + c_n \sum_{\mathbf{x}_j \in D_i^-} \xi_j)$, is the total loss of dataset D_i. $g(\mathbf{z}) = \frac{1}{2}\|\mathbf{z}\|^2$ and $\mathbf{u} = \frac{1}{\rho}\boldsymbol{\lambda}$. Although distributed cost-sensitive classification via ADMM (CS-ADMM) in Algorithm 1 can effectively solve the within-node class imbalance issue, it is time-consuming for seeking a global consensus solution of sub-problems. Furthermore, the other imbalance issues will aggravate the above situation because of both the slow convergence speed and time delay.

Algorithm 1 CS-ADMM

Input: $\mathbf{z}^0$, $\mathbf{u}_i^0$ and $\rho > 0$
Output: optimization variables $\mathbf{w}_i^{k+1}$, $\mathbf{z}^{k+1}$

1: **for** $k = 0, 1, \ldots,$ **do**
2: $\quad \mathbf{w}_i^{k+1} = \arg\min_{\mathbf{w}_i} f_i(\mathbf{w}_i) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{z}^k + \mathbf{u}_i^k\|^2$
3: $\quad \mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\sum_{i=1}^{N}\|\mathbf{w}_i^{k+1} - \mathbf{z} + \mathbf{u}_i^k\|^2$
4: $\quad \mathbf{u}_i^{k+1} = \mathbf{u}_i^k + \mathbf{w}_i^{k+1} - \mathbf{z}^{k+1}$
5: **end for**

Regarding between-node structure imbalance, we consider that local functions which train the similar datatsets have similar local variables. Inspired by the recent work in [12], we explicitly study the relationship between local functions (described in Subsection III-B) by grouping, and use the weighted average of local variables intra-group to approximately update local variables and the global variable. Hence, the major difference compared with Algorithm 1 is that CS-GADMM involves two local variable updates per iteration rather than one. The updates of local variable $\mathbf{w}_i$, using

$$\mathbf{w}_i^{k+\frac{1}{2}} = \arg\min_{\mathbf{w}_i} f_i(\mathbf{w}_i) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{z}^k + \mathbf{u}_i^k\|^2; \tag{5}$$

$$\mathbf{w}_i^{k+1} = \sum_{F_i \in G_j} \beta_i * \mathbf{w}_i^{k+\frac{1}{2}}, \tag{6}$$

where $\beta_i > 0$ is the weight parameter, and $F_i \in G_j$ means that local function F_i belongs to group G_j. Furthermore, for between-node class imbalance, we extend the dual coordinate descent method to alleviate the time delay (described in Subsection III-C).

B. Relationship Measurement between Local Functions

In CS-GADMM, the local function of a particular subproblem i with respect to $\mathbf{w}_i$ can be reformed as:

$$F_i(\mathbf{w}_i) = f_i(\mathbf{w}_i) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{z} + \mathbf{u}_i\|^2,$$

where $\mathbf{z}$ is the global variable, $\mathbf{u}_i^k$ can be denote as $\sum_{j=1}^{k}(\mathbf{w}_i^j - \mathbf{z}^j) + \mathbf{u}_i^0$ by its update process. In [22], a δ-relationship between local functions from theoretical perspective is defined as:

$$F_i(\mathbf{w}) = \mathbf{w}^T\mathbf{A}_i\mathbf{w} + \mathbf{b}_i\mathbf{w} + c_i, \mathbf{A}_i \in \mathbb{R}^{d\times d}, \mathbf{b}_i \in \mathbb{R}^d, c_i \in \mathbb{R}$$

are δ-related, if for any $i, j \in \{1, 2, \ldots, n\}$, it holds that

$$\|\mathbf{A}_i - \mathbf{A}_j\| \le \delta, \|\mathbf{b}_i - \mathbf{b}_j\| \le \delta, |c_i - c_j| \le \delta.$$

Inspired by the observation, the relationship between local functions can be measured by the similarity between datasets as $f_i(\mathbf{w}_i)$ is the total loss related to the datasets D_i. In contrast to [5] which combines all local variables to approximate local functions without in consideration of the difference between local functions, we cautiously study the similarity between local functions by grouping methods. We consider that local functions intra-group are more similar than those inter-group, and use local variables intra-group, no inter-group, to approximate the local variables in the same group.

The traditional similarity measurement between samples on different nodes involves the expensive communication and computation time cost, and is impracticable. In recent years, the statistical data similarity has been used to recommend the appropriate classification algorithm for different datasets [23], [24]. Also, Arjevani *et al.* mention that the statistical data similarity can be used to measure the similarity between local functions [22]. Therefore, in this paper, we utilize the means of positive samples and negative samples, and local variables used in [12] to measure the similarity between datasets using the Euclidean distance. Moreover, we utilize the L method to group local functions into several groups, which can efficiently determine a suitable number of groups from any hierarchical clustering [25].

C. Efficient Methods for Sub-problem Optimization

For the sub-problem optimization, the soft-margin CSSVM is adopted to deal with imbalanced dataset, and the primal problem can be rewritten in a more clear way:

$$\begin{aligned} \min_{\mathbf{w}_i} \ & F_i(\mathbf{w}_i) = C(\sum_{\mathbf{x}_j \in D_i^+} c_p \xi_j + \sum_{\mathbf{x}_j \in D_i^-} c_n \xi_j) + \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{v}_i\|^2, \\ & s.t.\ y_j \mathbf{w}_i^T \mathbf{x}_j \geq 1 - \xi_j, \xi_j \geq 0, j = 1, 2, \ldots, n, \end{aligned} \tag{7}$$

where $\mathbf{v}_i = (\mathbf{z} - \mathbf{u}_i)$. Basically, the Lagrange for the primal problem is mathematically formulated as:

$$\begin{aligned} L_\rho(\mathbf{w}_i, \xi) = & \frac{\rho}{2}\|\mathbf{w}_i - \mathbf{v}_i\|^2 + C(\sum_{\mathbf{x}_j \in D_i^+} c_p \xi_j + \sum_{\mathbf{x}_j \in D_i^-} c_n \xi_j) \\ & + \sum_{j=1}^{l_i} \alpha_j (1 - \xi_j - y_j \mathbf{w}_i \mathbf{x}_j) - \sum_{j=1}^{l_i} \beta_j \xi_j, \end{aligned} \tag{8}$$

where α_j and β_j are the Lagrange multipliers, l_i is the number of samples in dataset D_i. Its dual problem can be written as:

$$\begin{aligned} & \min_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) = \frac{1}{2\rho}\boldsymbol{\alpha}^T \mathbf{Q} \boldsymbol{\alpha} - \mathbf{b}^T \boldsymbol{\alpha}, \\ & s.t. \quad 0 \leq \alpha_j \leq C_j, \quad j = 1, \ldots, l_i \end{aligned} \tag{9}$$

where $Q_{jt} = y_j y_t \mathbf{x}_j^T \mathbf{x}_t$. If $\mathbf{x}_j$ is a positive sample, $C_j = c_p * C$, otherwise $C_j = c_n * C$. $\mathbf{b} = [1 - y_1 \mathbf{v}_i^T \mathbf{x}_1, \ldots, 1 - y_{l_i} \mathbf{v}_i^T \mathbf{x}_{l_i}]^T$.

The dual coordinate descent method (DCD) [26], [27] can be used to solve the dual problem in Eq. (9). To alleviate the between-node class imbalance problem, we discreetly propose a min-batch extension of DCD (MDCD) for the sub-problem optimization. In MDCD, we count the numbers of samples on different nodes and broadcast the minimum number l' to all nodes. Then, a sub-dataset of size l', $D_{l'}$ is randomly chosen in each node for the sub-problem optimization. The partial derivative, $\bigtriangledown_j f(\boldsymbol{\alpha}^k)$ with respect to α_j is,

$$\bigtriangledown_j f(\boldsymbol{\alpha}^k) = \frac{1}{\rho}\sum_{t=1}^{l_i} \alpha_t^k Q_{jt} - b_j = y_j \mathbf{w}_j^k \mathbf{x}_j - 1. \tag{10}$$

Then, $\mathbf{w}_i$ can be updated as follow:

$$\mathbf{w}_i^{k+1} = \mathbf{w}_i^k + (\alpha_j^{k+1} - \alpha_j^k) y_j \mathbf{x}_j. \tag{11}$$

Note that if only one sample in D_i is randomly chosen at each iteration, we can obtain a stochastic DCD method which is related to the tradeoff between the communication and computation time and can be studied in our future work.

D. Update Processes of CS-GADMM

Here, the weighted average method is used for the variable update in consideration of the imbalance ratio and different sizes of datasets. Therefore, the weight parameter, β_i of local function F_i can be defined as follow:

$$\beta_i = \frac{B_i(X) * N_i(X)}{\sum_{F_i \in G_j} B_i(X) * N_i(X)},$$

where $B_i(X)$ and $N_i(X)$ are the imbalance ratio (*IR*) and size ratio of dataset D_i in group G_j, respectively. At each iteration, CS-GADMM involves two local variable updates. After $\mathbf{w}_i^{k+\frac{1}{2}}$ is obtained by the sub-problem optimization, we aggregates all local variables of local functions intra-group to update the corresponding local variables again,

$$\mathbf{w}_i^{k+1} = \sum_{F_i \in G_j} \beta_i * \mathbf{w}_i^{k+\frac{1}{2}}.$$

Then, the update of $\mathbf{z}$ is mathematically formulated as:

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2}\sum_{i=1}^{N} \|\mathbf{w}_i^{k+1} - \mathbf{z} + \mathbf{u}_i^k\|^2, \tag{12}$$

Since the objective in (12) with respect to $\mathbf{z}$ is differentiable, it can be obtained by a simple average of $\mathbf{w}_i^{k+1}$s and $\mathbf{u}_i^{k+1}$s,

$$\mathbf{z}^{k+1} = \frac{\rho}{1 + N\rho}\sum_{i=1}^{N} (\mathbf{w}_i^{k+1} + \mathbf{u}_i^k). \tag{13}$$

Finally, the dual variable $\mathbf{u}_i^{k+1}$ is updated as follow:

$$\mathbf{u}_i^{k+1} = \mathbf{u}_i^k + \mathbf{w}_i^{k+1} - \mathbf{z}^{k+1}. \tag{14}$$

The primal residual $\mathbf{r}^{k+1} = \mathbf{w}^{k+1} - \mathbf{z}^{k+1}$, and dual residual $\mathbf{s}^{k+1} = \rho(\mathbf{z}^{k+1} - \mathbf{z}^k)$ can measure how well the iterations satisfy the optimality conditions [6]. Thus, the stopping criteria,

$$\|\mathbf{r}^k\|_2 \leq \epsilon^{\text{pri}} \quad \|\mathbf{s}^k\|_2 \leq \epsilon^{\text{dual}}, \tag{15}$$

is defined in [12], where ϵ^{pri} and ϵ^{dual} are feasible tolerances. The steps of CS-GADMM are summarized in Algorithm 2.

Algorithm 2 Distributed classification via CS-GADMM

Input: datasets: $\mathbf{D} = \{D_1, D_2, \ldots, D_N\}$, parameters ($C_j$, ρ, α, ϵ^{pri}, ϵ^{dual}, $\mathbf{r}^0$, $\mathbf{s}^0$, $\mathbf{z}^0$ and $\mathbf{u}_i^0$)
Output: the global variable $\mathbf{z}^{k+1}$

$\beta_i = \frac{B_i(X) * N_i(X)}{\sum_{F_i \in G_j} B_i(X) * N_i(X)}$
$k \leftarrow 0$
while $\|\mathbf{r}^k\|_2 > \epsilon^{\text{pri}}$ or $\|\mathbf{s}^k\|_2 > \epsilon^{\text{dual}}$ **do**
  $\mathbf{w}_i^{k+\frac{1}{2}}$ can be obtained by MDCD in parallel
  if $k == 0$ **then**
    local functions are grouped into several groups
  end if
  $\mathbf{w}_i^{k+1} = \sum_{f_i \in G_j} \beta_i * \mathbf{w}_i^{k+\frac{1}{2}}$
  $\mathbf{z}^{k+1} = \frac{\rho}{1+N\rho}\sum_{i=1}^{N}(\mathbf{w}_i^{k+1} + \mathbf{u}_i^k)$
  $\mathbf{u}_i^{k+1} \leftarrow \mathbf{u}_i^k + \mathbf{w}_i^{k+1} - \mathbf{z}^{k+1}$
  $\mathbf{r}^{k+1} \leftarrow \mathbf{w}^{k+1} - \mathbf{z}^{k+1}, \quad \mathbf{s}^{k+1} = -\rho(\mathbf{z}^{k+1} - \mathbf{z}^k)$
  $k \leftarrow k + 1$
end while

IV. Experimental Results

A. Datasets and Experimental Settings

Public imbalanced datasets are adopted for performance evaluation, with information shown in Table I. The first part of Table I shows three binary class datasets[1]; the second part shows these datasets derived from multi-class dataset KDD

[1]http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/binary.html.

Cup 1999[2]. All datasets are randomly split into training/testing datasets with a ratio of $8/2$. Moreover, $\mathbf{w}_i^{k+1}$ can be replaced with its relaxation form to facilitate the global consensus,

$$\mathbf{w}_i^{k+1} = \alpha \mathbf{w}_i^{k+1} + (1-\alpha)\mathbf{z}_i^k.$$

In all algorithms, we set the parameters (ρ, c_p, c_n, α) to (1, *IR*, 1, 1.6), which are empirically well chosen in [3]. C is chosen from $2^i (i = -3, \ldots, 8)$ by inner five-fold cross validation.

TABLE I
EXPERIMENTAL DATASETS. THE NUMBER OF SAMPLES AND NUMBER OF ATTRIBUTES ARE DENOTED AS (#EX.) AND (#ATTS.), WITH IR AND C.

Datasets	#Ex.	#Atts.	#IR.	#C.
real-sim	72,309	20,958	2.25	1
a9a	48,842	123	3.18	4
ijcnn1	141,691	22	9.45	0.5
dosvsnormal	4,856,151	41	3.99	64
normalvsprb	1,013,883	41	23.67	32
dosvsprb	3,924,472	41	94.58	32

In this paper, geometric mean (*GM*), β-F-measure (*FM*), number of iterations (N_{iter}) and training time (T_{time}) are used as evaluation metrics, where β is set to *IR* [10]. All algorithms are synchronous and implemented on a star distributed system. 18 machines are used under the framework of Message Passing Interface (MPI), each of them with an Intel(R) Xeon(R) CPU E5-2650 (2.6Ghz/30M Cache) processor and 64 GB RAM.

B. Comparison with Other Distributed Algorithms

We firstly discuss the within-node class imbalance and between-node structure imbalance issues. To validate the efficiency of CS-GADMM for imbalanced classification, we compare the following versions of distributed ADMM algorithms:

- **ADMM-DCD:** The dual coordinate descent method (DCD) is utilized to solve sub-problems in distributed classification via ADMM without cost-sensitive learning in [3].
- **ADMM-TRON:** In distributed ADMM, the trust region Newton method (TRON) is used to optimize the sub-problem.
- **GADMM-DCD:** A group layer is added in the framework of GADMM, and sub-problems can be optimized by DCD.
- **CS-ADMM-DCD:** This is a distributed cost-sensitive classification algorithm. We implement it based on ADMM, and use DCD for the sub-problem optimization.
- **CS-GADMM-DCD:** The relationship between local functions is studied and DCD is also used in CS-GADMM.

The experiments are conducted repeatedly ten times and the averaged results are reported. Table II shows that CS-GADMM-DCD performs better performance than others, especially on datasets *a9a* and *ijcnn1*. In the experiments, those datasets derived from KDD Cup 1999 can be well classified by linear classifiers, and thus the performance improvement is not very obvious. As a whole, CS-GADMM can efficiently solve the within-node class imbalance problem, and improve the classification performance of imbalanced data.

For between-node structure imbalance, to validate the superiority in time efficiency of CS-GADMM, as well as to compare with CS-ADMM, we use N_{iter} and T_{time} as evaluation metrics. Also, we compare the relative differences in geometric mean and F-measure between current test performance (*FM* and *GM*) and best performance (*FM** and *GM**), using

$$FM_D\% = \frac{FM^* - FM}{FM^*} * 100\%, \text{ and } GM_D\% = \frac{GM^* - GM}{GM^*} * 100\%.$$

[2]http://archive.ics.uci.edu/ml.

TABLE II
THE EXPERIMENTAL RESULTS OF THESE COMPARATIVE ALGORITHMS.

Dataset	ADMM-DCD	ADMM-TRON	GADMM-DCD	CS-GADMM-DCD
FM (%): f-measure				
real-sim	95.76(0.13)	95.92(0.19)	94.95(0.28)	**95.94**(0.24)
a9a	59.44(0.95)	59.72(1.43)	59.23(1.13)	**80.05**(1.65)
ijcnn1	35.94(0.61)	33.47(0.30)	27.77(0.72)	**89.85**(0.68)
dosvsnormal	99.90(0.00)	99.85(0.00)	99.86(0.00)	**99.92**(0.00)
normalvsprb	95.09(0.15)	93.22(0.16)	89.45(0.15)	**96.60**(0.13)
dosvsprb	98.28(0.06)	98.61(0.07)	87.52(0.09)	**99.84**(0.03)
GM (%): geometric mean				
real-sim	96.91(0.10)	97.02(0.15)	96.38(0.20)	**97.08**(0.17)
a9a	73.47(0.74)	73.67(1.01)	73.38(0.77)	**81.67**(0.70)
ijcnn1	59.33(0.49)	57.44(0.23)	52.17(0.56)	**89.13**(0.33)
dosvsnormal	99.89(0.00)	99.88(0.00)	99.89(0.00)	**99.91**(0.00)
normalvsprb	96.49(0.08)	96.54(0.08)	94.53(0.01)	**97.93**(0.07)
dosvsprb	99.13(0.03)	99.30(0.04)	93.55(0.05)	**99.90**(0.02)

TABLE III
THE EXPERIMENTAL RESULTS OF THE TWO COMPARATIVE ALGORITHMS.

Algorithms	*real-sim*	*a9a*	*ijcnn1*	*dosvsnormal*	*normalvsprb*	*dosvsprb*
N_{iter}: the number of iterations						
CS-ADMM-DCD	49.02	647.20	298.60	96.20	336.80	215.40
CS-GADMM-DCD	**14.40**	**420.80**	**175.40**	**73.80**	**171.80**	**190.80**
T_{time}(s): training time starting from the data loading						
CS-ADMM-DCD	2.29	2.79	5.25	79.17	36.57	194.75
CS-GADMM-DCD	**1.31**	**2.60**	**3.41**	**53.76**	**18.22**	**151.30**
FM_D (%): the relative difference in f-measure						
CS-ADMM-DCD	**0.0034**	**0.0222**	0.0111	0.0001	**0.0002**	0.0027
CS-GADMM-DCD	0.0122	0.0256	**0.0103**	0.0001	0.0014	**0.0024**
GM_D (%): the relative difference in geometric mean						
CS-ADMM-DCD	**0.0025**	0.0138	0.0074	0.0001	**0.0001**	**0.0014**
CS-GADMM-DCD	0.0075	**0.0109**	**0.0069**	0.0001	0.0007	0.0036

In Table III, we can observe that CS-GADMM-DCD converges faster than CS-ADMM-DCD, which significantly reduces the number of iterations and saves the training time with the competitive classification performance. The main reason is that by exploiting the relationship between local functions, local variables of local functions intra-group are aggregated to approximately update the global variable and the corresponding local variables again. In contrast to CS-ADMM, fewer different local variables need to be iteratively updated until reach a consensus, and thus the convergence speed can be accelerated in CS-GADMM. The experimental results in Table II and III demonstrate that CS-GADMM can effectively deal with large-scale imbalanced data.

C. Evaluation of Imbalanced Classification with Distributed Data Imbalance

In this section, we further discuss the distributed data imbalance issue in imbalanced data. To evaluate the performance of our proposed CS-GADMM with MDCD as well as to compare with other distributed algorithms, we randomly split the datasets into N disjoint datasets whose sizes are observably

different, with the max ratio 1 : 16. The experimental results of our proposed CS-GADMM with MDCD comparing with the baseline distributed algorithms are shown in Table IV.

TABLE IV
THE EXPERIMENTAL RESULTS OF THESE ALGORITHMS.

Algorithms	*real-sim*	*dosvsnormal*	*normalvsprb*	*dosvsprb*
FM (%): F-measure				
CS-ADMM-DCD	**96.61**(0.22)	**99.93**(0.00)	**96.59**(0.16)	99.96(0.00)
CS-ADMM-MDCD	96.58(0.22)	99.93(0.00)	99.58(0.16)	99.94(0.01)
CS-GADMM-DCD	95.61(0.23)	99.92(0.00)	96.58(0.16)	99.95(0.02)
CS-GADMM-MDCD	95.60(0.23)	99.91(0.00)	96.55(0.15)	**99.78**(0.01)
GM (%): geometric mean				
CS-ADMM-DCD	**97.26**(0.18)	99.92(0.00)	98.16(0.01)	99.97(0.02)
CS-ADMM-MDCD	**97.26**(0.18)	**99.93**(0.00)	98.15(0.01)	**99.97**(0.01)
CS-GADMM-DCD	96.71(0.16)	99.92(0.00)	98.14(0.01)	99.96(0.00)
CS-GADMM-MDCD	96.69(0.16)	99.91(0.00)	98.13(0.01)	**99.88**(0.02)
T_{time}: training time starting from the data loading				
CS-ADMM-DCD	2.50	296.26	291.31	389.76
CS-ADMM-MDCD	2.19	139.63	105.70	**153.35**
CS-GADMM-DCD	1.36	217.07	91.14	367.18
CS-GADMM-MDCD	**1.24**	**87.26**	**23.96**	156.24

Table IV shows that although DCD can obtain slightly better classification performance than that of MDCD, MDCD converges much faster than DCD, and thus it can significantly reduce the training time. The main reason is that the computation time on each node is different since all samples need to be trained in DCD by between-node class imbalance. Meanwhile, the training speed is limited by the slowest node, and thus the coordinations between nodes such as communications and synchronization involve expensive time cost. While MDCD chooses the same number of samples for training at each iteration, and alleviates the time delay problem. Hence, CS-GADMM can improve time efficiency, and is suitable to deal with imbalanced data with distributed data imbalance.

V. CONCLUSION

In this paper, we propose an efficient distributed cost-sensitive classification algorithm via Group-based ADMM (CS-GADMM) for imbalanced data. In CS-GADMM, the problem can be solved by the coordination of sub-problems in parallel. For distributed data imbalance, we exploit the relationships between local functions and propose a MDCD method to improve time efficiency. The experiments on benchmark datasets show that CS-GADMM converges faster against CS-ADMM with the competitive classification performance.

In the future, we will study the asynchronization issue to improve time efficiency of distributed learning. Also, we will propose stochastic methods for the sub-problem optimization.

The code is available at https://github.com/huihuiw/icdm16.

VI. ACKNOWLEDGEMENT

The work is supported by the National Science Foundation of China (Nos. 61432008, 61305068, 61673203).

REFERENCES

[1] G.-X. Yuan, C.-H. Ho, and C.-J. Lin, "Recent advances of large-scale linear classification," *Proceedings of the IEEE*, vol. 100, no. 9, pp. 2584–2603, 2012.

[2] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, pp. 1663–1707, 2010.

[3] C. Zhang, H. Lee, and K. G. Shin, "Efficient distributed linear classification algorithms via the alternating direction method of multipliers," in *AISTATS*, 2012, pp. 1398–1406.

[4] Y. Zhuang, W.-S. Chin, Y.-C. Juan, and C.-J. Lin, "Distributed newton methods for regularized logistic regression," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2015, pp. 690–703.

[5] D. Mahajan, N. Agrawal, S. S. Keerthi, S. Sellamanickam, and L. Bottou, "An efficient distributed learning algorithm based on effective local functional approximations," *Journal of Machine Learning Research*, vol. 16, pp. 1–36, 2015.

[6] 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*, vol. 3, no. 1, pp. 1–122, 2011.

[7] C.-P. Lee and D. Roth, "Distributed box-constrained quadratic optimization for dual linear SVM," in *ICML*, 2015.

[8] T. Yang, "Trading computation for communication: Distributed stochastic dual coordinate ascent," in *NIPS*, 2013, pp. 629–637.

[9] S. Barua, M. M. Islam, X. Yao, and K. Murase, "Mwmote–majority weighted minority oversampling technique for imbalanced data set learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 26, no. 2, pp. 405–425, 2014.

[10] S. del Río, V. López, J. M. Benítez, and F. Herrera, "On the use of mapreduce for imbalanced big data using random forest," *Information Sciences*, vol. 285, pp. 112–137, 2014.

[11] S. Li, Z. Wang, G. Zhou, and S. Y. M. Lee, "Semi-supervised learning for imbalanced sentiment classification," in *IJCAI*, 2011.

[12] H. Wang, Y. Gao, Y. Shi, and R. Wang, "Group-based alternating direction method of multipliers for distributed linear classification," *IEEE Transactions on Cybernetics*, 2016. [Online]. Available: http://dx.doi.org/10.1109/TCYB.2016.2570808

[13] Y. Qian, Y. Liang, M. Li, G. Feng, and X. Shi, "A resampling ensemble algorithm for classification of imbalance problems," *Neurocomputing*, vol. 143, pp. 57–67, 2014.

[14] H. He, E. Garcia *et al.*, "Learning from imbalanced data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, no. 9, pp. 1263–1284, 2009.

[15] R. Batuwita and V. Palade, "Fsvm-cil: fuzzy support vector machines for class imbalance learning," *Fuzzy Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 558–571, 2010.

[16] B. Krawczyk and M. Woźniak, "Diversity measures for one-class classifier ensembles," *Neurocomputing*, vol. 126, pp. 36–44, 2014.

[17] M. Kim, "Large margin cost-sensitive learning of conditional random fields," *Pattern Recognition*, vol. 43, no. 10, pp. 3683–3692, 2010.

[18] Z. E. Xu, M. J. Kusner, K. Q. Weinberger, and M. Chen, "Cost-sensitive tree of classifiers." in *ICML*, 2013, pp. 133–141.

[19] V. López, S. del Río, J. M. Benítez, and F. Herrera, "Cost-sensitive linguistic fuzzy rule based classification systems under the mapreduce framework for imbalanced big data," *Fuzzy Sets and Systems*, vol. 258, pp. 5–38, 2015.

[20] T. Suzuki, "Dual averaging and proximal gradient descent for online alternating direction multiplier method," in *ICML*, 2013, pp. 392–400.

[21] C.-P. Lee, K.-W. Chang, S. Upadhyay, and D. Roth, "Distributed training of structured SVM," *arXiv preprint arXiv:1506.02620*, 2015.

[22] Y. Arjevani and O. Shamir, "Communication complexity of distributed convex learning and optimization," in *NIPS*, 2015, pp. 1747–1755.

[23] Q. Song, G. Wang, and C. Wang, "Automatic recommendation of classification algorithms based on data set characteristics," *Pattern recognition*, vol. 45, no. 7, pp. 2672–2689, 2012.

[24] G. Wang, Q. Song, H. Sun, X. Zhang, B. Xu, and Y. Zhou, "A feature subset selection algorithm automatic recommendation method," *Journal of Artificial Intelligence Research*, 2013.

[25] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms," in *ICTAI*, 2004.

[26] M. Jaggi, V. Smith, M. Takác, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-efficient distributed dual coordinate ascent," in *NIPS*, 2014, pp. 3068–3076.

[27] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan, "A dual coordinate descent method for large-scale linear svm," in *ICML*, 2008, pp. 408–415.