The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Hypergraph Learning with Cost Interval Optimization

Xibin Zhao, Nan Wang, Heyuan Shi, Hai Wan,* Jin Huang, Yue Gao*
Key Laboratory for Information System Security, Ministry of Education
Tsinghua National Laboratory for Information Science and Technology
School of Software, Tsinghua University, China.
{n-wang16,shi}@mails.tsinghua.edu.cn
{zxb,wanhai,huangjin,gaoyue}@tsinghua.edu.cn
* indicates corresponding authors

Abstract

In many classification tasks, the misclassification costs of different categories usually vary significantly. Under such circumstances, it is essential to identify the importance of different categories and thus assign different misclassification losses in many applications, such as medical diagnosis, saliency detection and software defect prediction. However, we note that it is infeasible to determine the accurate cost value without great domain knowledge. In most common cases, we may just have the information that which category is more important than the other categories, *i.e.*, the identification of defect-prone softwares is more important than that of defect-free. To tackle these issues, in this paper, we propose a hypergraph learning method with cost interval optimization, which is able to handle cost interval when data is formulated using the high-order relationships. In this way, data correlations are modeled by a hypergraph structure, which has the merit to exploit the underlying relationships behind the data. With a cost-sensitive hypergraph structure, in order to improve the performance of the classifier without precise cost value, we further introduce cost interval optimization to hypergraph learning. In this process, the optimization on cost interval achieves better performance instead of choosing uncertain fixed cost in the learning process. To evaluate the effectiveness of the proposed method, we have conducted experiments on two groups of dataset, *i.e.*, the NASA Metrics Data Program (NASA) dataset and UCI Machine Learning Repository (UCI) dataset. Experimental results and comparisons with state-of-the-art methods have exhibited better performance of our proposed method.

Introduction

In most real-world applications, different misclassifications may lead to different losses. For example, misdiagnosing a patient with deadly disease as a health case may cause more serious losses than misdiagnosing a patient without disease as an illness case. Another example is in the software defect prediction field, where misclassifying a defect-prone software module may destroy the software system and lead to disaster consequences. However, more and more classification works mainly focus on minimizing the number of errors rather than the total cost.

Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

In recent years, cost-sensitive learning has attracted more attention (Wang et al. 2012; Kai 2002; Liu, Jun, and Ghosh 2009) from both the academia and industry. Although these works concentrate on the cost related issue, these costs for different categories are predefined and usually determined by experts or obtained among a set of candidate values. We note that in most cases, even with the deep domain knowledge, it is still very challenging to determine an precise fixed cost value for different categories. However, compared with the precise cost, cost interval is more easier to obtain. The cost intervals are estimated as ranges of cost and concluded by the opinions from experts, transforming from confidence intervals and the natural bounds (Liu and Zhou 2010). In addition, another challenging problem is that the correlations among data are difficult to describe. Traditional relationship formulations, such as graph structure linking two points at one time, is difficult to explore the high-order correlations among the data. To construct a robust classifier, a more powerful data formulation should be employed to describe the complex relationships.

To tackle these issues, in this paper, we propose a hypergraph learning method with cost interval optimization (CIHL). The main framework of our method is demonstrated in Figure 1. In our method, the hypergraph structure is used to describe the high-order correlations among training data. Comparing with traditional graph whose edge can only links two points, the hyperedge in a hypergraph can link more than two points, which can lead to a degree-free connections among the data. Nowadays, hypergraph has been widely used in many applications due to its superiority on high-order correlations description. With this structure, we conduct learning to estimate the labels for the testing data. More specifically, the cost information is incorporated in the data, and instead of determining a set of fixed costs, we employ cost intervals to optimize the cost information for improving accuracy of the classification results. We have conducted experiments on two types of datasets, *i.e.*, NASA dataset and UCI dataset. Experimental results and comparisons with state-of-the-art methods show the superiority of the proposed CIHL method.

The rest of this paper is organized as follows. Following we briefly review related work. Then we introduce hypergraph learning and its applications. Experiments, comparison and discussions are provided in the following part and

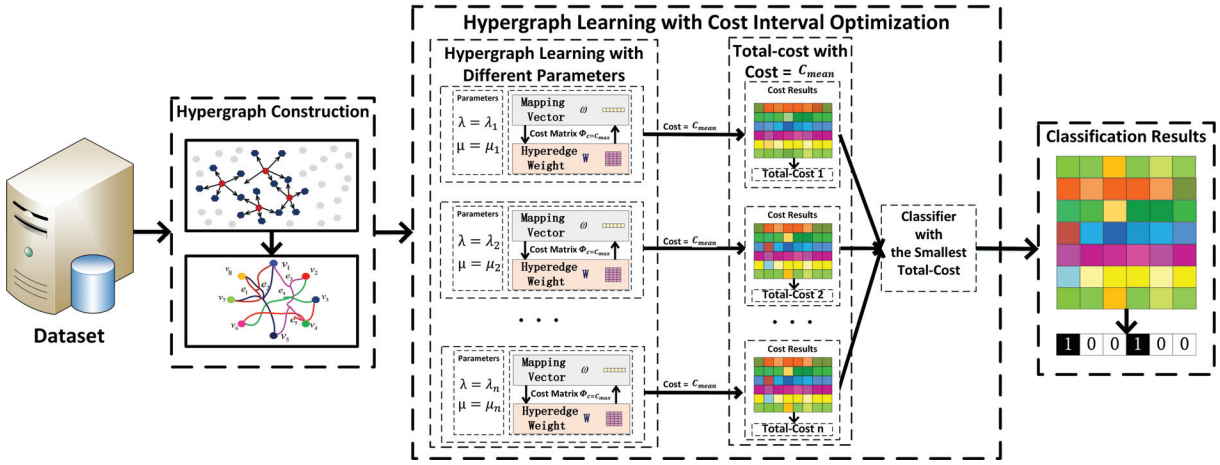


Figure 1: Illustration of the framework of our proposed method.

finally we conclude this paper.

Related Work

Most of traditional classification works assume that the misclassification of different categories are with the same cost. However, in practice, the misclassification cost of different categories may vary significantly. Most classification problems are naturally cost-sensitive, such as fraud detection, medical diagnosis, and object detection. Therefore, instead of minimizing the amount of misclassifications, it is meaningful to decrease the total cost. In real-word applications, there are many kinds of costs (Turney 2002), such as the cost of misclassification errors, the cost of intervention, and so on. The misclassification cost can be divided into two categories, *i.e.*, the constant error cost and the conditional error cost. For the conditional error cost, different types of errors are based on different conditions, *e.g.*, individual case and feature value. As for the constant cost, the cost for all the cases in the same category could be a constant value, and this is also the focus of our paper.

In past few years, in order to minimize the total cost, these kinds of learning methods attract more and more attention (Domingos 1999; Liu, Miao, and Zhang 2014; Bertoni, Frasca, and Valentini 2011). Cost-sensitive learning can be divided into two types, *i.e.*, algorithm dependent method and rescaling method. The first type modifies the specific cost insensitive algorithm directly. For example, Masnadi *et al.* (Masnadi-Shirazi and Vasconcelos 2010) produced a cost-sensitive hinge loss function, and a cost-sensitive SVM was derived to minimize the associated risk. Sheng *et al.* (Sheng and Ling 2005) proposed a hybrid cost-sensitive decision tree, which combined the advantage of cost-sensitive decision tree and cost-sensitive naive Bayes. The second type of cost-sensitive method is rescaling. Some research on data distribution demonstrates that natural distribution limits the performance of the classifier. Thus, rescaling has become one of the most general approaches in cost-sensitive learning. Its working principle is to adjust the weighting, threshold or rebalance the distribution of training data in the preprocessing step, so that the classifier pay more attention to the important classes and improve the detection rate of these classes. Usually, rescaling can be further divided into three types, *i.e.*, changing the decision thresholds, assigning different weights to testing data and resampling. In the type of changing the decision thresholds, Domingos *et al.* (Domingos 1999) proposed MetaCost which treated the classifier as a black box and required no domain knowledge. It rescaled the data base on minimizing the total cost. Wu *et al.* (Wu and Chang 2003) proposed a method to adjust the class boundary by transforming the kernel function or modifying the kernel matrix. Chan *et al.* (Chan and Stolfo 1998) presented a multi-classifier meta-learning method, which can reduce the loss by immediate transactions. In the type of assigning different weights to testing data, the cost of different classes can be determined by the domain knowledge or suggestions from the experts. Liu *et al.* (Liu, Miao, and Zhang 2014) proposed a two-stage cost-sensitive learning method for software defect prediction, and this algorithm utilized the cost information for feature selection and classification. Moreover, Kai (Kai 2002) proposed an instance-weighting method to construct cost-sensitive trees. Bertoni *et al.* (Bertoni, Frasca, and Valentini 2011) proposed a hopfield-based cost-sensitive neural network algorithm (COSNet), which assigned different weights to different nodes and adopted a cost-sensitive methodology to manage the unbalance between positive and negative labels. The last kind of rescaling is resampling. In this method, many scholars adopt over-sampling the minority class examples, under-sampling the majority class examples or combining the two sampling methods to assign different importance to different classes. For example, Liu *et al.*(Liu, Jun, and Ghosh 2009) used an uncertain sampling approach to reduce the total cost of misclassifications. Wang *et al.*(Wang et al. 2012) proposed a adaptive over-sampling technique based on the data density, which adaptively adjusted the number of the minority samples according to its level of learning difficulty.

Although there are many works taking cost-sensitive into

account, most of existing methods use uncertain fixed cost value as the true cost. The true cost is the most effective cost to guide the classifier to minimize the total cost. Uncertain cost value may limit the development of the cost-sensitive learning. Comparing with accurate cost value, cost intervals are more available. Therefore, some researchers focus on constructing a learning method with cost interval, such as the excellent work in (Liu and Zhou 2010), Liu *et al.* proposed a method working with cost interval. In the learning procedure, this method achieved the optimization by minimizing the worst and mean case risks in meanwhile. Specifically, this method included two steps: (1) minimizing the total cost with the maximum cost of the cost interval and learning a variation of SVMs; and (2) minimizing the mean risk by parameter value selection on the variation of SVMs. Moreover, Zhou *et al.*(Zhou and Zhou 2016) proposed cisLDM, which aimed to optimize the margin distribution on testing data when minimizing the worst-case and the mean total cost simultaneously using the cost interval.

Introduction of Hypergraph Learning

Hypergraph has been widely used in many fields. Here, we briefly introduce hypergraph learning and its applications. Nowadays, there are many learning methods based on graph structure (Yang et al. 2015; Kapoor et al. 2005; Zhang et al. 2016). However, the pairwise connections limit the ability of the classifier to mine the deep relationships among data. Different from traditional graph structure, hypergraph can explore the high-order correlations among the data by the flexibility of hyperedges.

Generally, a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ contains three components, *i.e.*, a vertex set $\mathcal{V}$, an hyperedge set $\mathcal{E}$, and weights of hyperedges $\mathbf{W}$. The hypergraph can be represented as a $|\mathcal{V}| \times |\mathcal{E}|$ incidence matric $\mathbf{H}$, and the entry of $\mathbf{H}$ is defined as

$$h(v,e) = \begin{cases} 1 & \texttt{if } v \in e \\ 0 & \texttt{if } v \notin e \end{cases}, \tag{1}$$

which represents the relationships between the vertices and the hyperedges.

The degree of each vertex in the hypergraph is defined by $d(v) = \sum_{e\in\mathcal{E}} \omega(e)\, h(v,e)$, and for each hyperedge, its degree is defined by $\delta(e) = \sum_{v\in\mathcal{V}} h(v,e)$.

Then, two diagonal matrices $\mathbf{D}_v$ and $\mathbf{D}_e$ can be used to denote the degrees of vertices and hyperedges. Moreover, the diagonal matrix $\mathbf{W}$ denotes the weights of the hyperedges.

Zhou *et al.* (Zhou, Huang, and Schölkopf 2006) proposed a regularizer on hypergraph, which was defined as $\arg\min_{\mathbf{F}} \{\lambda R_{emp}(\mathbf{F}) + \Omega(\mathbf{F})\}$. In this framework, the trade-off parameter $\lambda > 0$ balances the weights between $R_{emp}(\mathbf{F})$ and $\Omega(\mathbf{F})$. $\mathbf{F}$ is the to-be-learned relevance matrix, $\Omega(\mathbf{F})$ is a hypergraph structure regularizer, and $R_{emp}(\mathbf{F})$ is the empirical loss based on the labeled data.

Usually, the empirical loss $R_{emp}(\mathbf{F})$ is defined as $R_{emp}(\mathbf{F}) = \|\mathbf{F} - \mathbf{Y}\|_{\mathrm{F}}^2$, where $\mathbf{Y}$ is the label matrix of samples. As for $\Omega(\mathbf{F})$, the regularizer on hypergraph is defined by

$$\Omega(\mathbf{F}) = \frac{1}{2} \sum_{e\in\mathcal{E}} \sum_{u,v\in\mathcal{V}} \sum_{k=1}^{C} \frac{w(e)\,\mathbf{H}(u,e)\,\mathbf{H}(v,e)}{\delta(e)} \left(\frac{\mathbf{F}(u,k)}{\sqrt{d(u)}} - \frac{\mathbf{F}(v,k)}{\sqrt{d(v)}} \right)^2, \tag{2}$$

which aims to smooth the relationships among the data, based on the principle that the more connections between two vertices exist, the stronger similarity between their labels will be.

Here, let $\mathbf{\Theta} = \mathbf{D}_v^{-\frac{1}{2}} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^T \mathbf{D}_v^{-\frac{1}{2}}$, and $\mathbf{\Delta} = \mathbf{I} - \mathbf{\Theta}$, then the normalized cost function can be rewritten as $\Omega(\mathbf{F}) = \mathrm{tr}\left(\mathbf{F}^T \mathbf{\Delta} \mathbf{F}\right)$, where $\mathbf{\Delta}$ is called hypergraph Laplacian.

Then, the learning function can be re-written as $\arg\min_{\mathbf{F}} \left\{ \mathrm{tr}\left(\mathbf{F}^T \mathbf{\Delta} \mathbf{F}\right) + \lambda \|\mathbf{F} - \mathbf{Y}\|_{\mathrm{F}}^2 \right\}$.

Due to the superiority of hypergraph structure, it has been used in many areas, such as image retrieval (Gao et al. 2012), feature selection (Zhu et al. 2017), image classification (Gao et al. 2014) and object classification (Su et al. 2017). Huang *et al.*(Huang et al. 2010) proposed a transductive learning method for image retrieval. In this method, the images were regarded as vertices in the hypergraph structure, and then the image retrieval problem can be transformed into the problem of hypergraph ranking. Regarding the multi-label classification task, Sun *et al.*(Sun, Ji, and Ye 2008) introduced a hypergraph spectral learning formulation to mine the relationships among different classes. Chen *et al.* (Chen et al. 2015) used the hypergraph to formulate textual, visual and emotion information jointly for sentiment prediction.

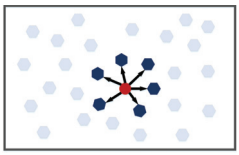
Hypergraph Learning with Cost Interval Optimization

Data Modeling for High-order Correlation

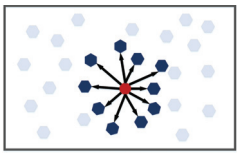
Hypergraph plays an important role in modeling high-order correlations among the data. In this work, we regard each sample as a vertex in a hypergraph. Given a set of testing and training samples $\{\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_n\}$, our task is to explore the relationships among the samples and map the features of $\{\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_n\}$ to labels. In the following sections, in order to distinguish different variables, we denote matrices as boldface upper-case letters, vectors as boldface lower-case letters and scalars as normal italic letters.

In hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, the number of vertices N_v is the same as the number of both the training and testing data. Vertex i in the hypergraph is represented by a feature vector $\mathbf{s}_i$ ($\mathbf{s}_i \in R^p$), and then we let $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{N_v}\}$ denote features for all vertices. In the process of building hyperedges, we select one vertex in $\{\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_n\}$ as the centroid, and select K nearest neighbors according to the distance between the centroid with other vertices, and then the centroid vertex and the selected neighbors are connected by the hyperedges. The process of hyperedge construction is demonstrated in Figure 2. After all vertices have been selected as the centroid vertex, we build N_e hyperedges.

In addition, we denote the weights of all hyperedges as a diagonal matrix $\mathbf{W}$, and for each hyperedge, $\mathbf{W}_{ii}$ is initialized as 1. Here, we utilize a probabilistic incidence matrix $\mathbf{H}$ to represent the connections on the hypergraph. As for



(a) A hyperedge example with six nearest neighbors.



(b) A hyperedge example with ten nearest neighbors.

Figure 2: Two hyperedge examples.

the entry in the matrix $\mathbf{H}$, the connectivity between vertex v_i and hyperedge e_j is displayed as the (i,j)-th entry in $\mathbf{H}$, and matrix $\mathbf{H}$ is defined as

$$\mathbf{H}(i,p) = \begin{cases} \exp\left(-\frac{d(v_i,v_{centroid})^2}{\alpha \bar{d}^2}\right) & \text{if } v_i \in e_p \\ 0 & \text{if } v_i \notin e_p \end{cases}, \quad (3)$$

In equation (3), $d(v_i, v_{centroid})$ is the distance between vertex v_i and vertex $v_{centroid}$. $v_{centroid}$ is the centroid vertex for the hyperedge e_j, and $\bar{d}$ is the average distance between each pair of samples in the hypergraph.

According to matrix $\mathbf{H}$, the degree of vertex v_i is generated by $d(v_i) = \sum_{e\in\mathcal{E}} \omega(e)\, h(v_i, e)$, and the degree of hyperedge e_j is generated by $\delta(e_j) = \sum_{v\in\mathcal{V}} h(v, e_j)$. Then, the degrees of all vertices and hyperedges are represented by diagonal matrices $\mathbf{D}_v$ and $\mathbf{D}_e$.

In order to minimize total cost, the misclassification costs of different classes are introduced into hypergraph framework. As for the regularization framework of cost-sensitive hypergraph, there are mainly three components, *i.e.*, the empirical loss with cost information $\mathcal{R}_{emp}(\boldsymbol{\omega})$, the hypergraph Laplacian regularizer $\Omega(\boldsymbol{\omega})$, and the optimal cost-sensitive hypergraph regularization $\Psi(\mathbf{W})$.

As for the cost-sensitive empirical loss $\mathcal{R}_{emp}(\boldsymbol{\omega})$, it is defined as

$$\mathcal{R}_{emp}(\boldsymbol{\omega}) = \|\boldsymbol{\Phi}(\mathbf{S}\boldsymbol{\omega} - \mathbf{y})\|_2^2 = \sum_{i=1}^{N_v} \left(\boldsymbol{\Phi}_{i,i}(\mathbf{s}_i\boldsymbol{\omega} - \mathbf{y}_i)\right)^2, \quad (4)$$

here, $\boldsymbol{\omega}$ is the to-be learned mapping vector, which translates the features of the data to correlation labels. $\mathbf{s}_i$ is the feature vector for the i-th sample. $\mathbf{S}\boldsymbol{\omega}$ is the classification results and the diagonal matrix $\boldsymbol{\Phi}$ represents the misclassification cost information. The cost of the i-th sample is defined as $\boldsymbol{\Phi}_{ii}$.

In the learning process, some hyperedges can represent the data correlations well and the others may be not. In order to re-weight the hyperedges and improve the ability of hypergraph in classification, the optimal cost-sensitive hypergraph regularization is employed, which is defined as $\Psi(\mathbf{W}) = \|\mathbf{W}\|_{\mathrm{F}}^2$. In this equation, $\|\mathbf{W}\|_{\mathrm{F}}$ represents the Frobenius norm of $\mathbf{W}$ and the diagonal matrix $\mathbf{W}$ is the weights of all hyperedges.

As for the regularizer on the hypergraph, it is similar to traditional hypergraph, which is defined as

$$\begin{aligned}\Omega(\boldsymbol{\omega}) &= \tfrac{1}{2}\sum_{e\in\mathcal{E}}\sum_{v_i,v_j\in\mathcal{V}} \frac{\mathbf{W}(e)\mathbf{H}(v_i,e)\mathbf{H}(v_j,e)}{\delta(e)}\left(\frac{\boldsymbol{\omega}\mathbf{s}_i}{\sqrt{d(v_i)}} - \frac{\boldsymbol{\omega}\mathbf{s}_j}{\sqrt{d(v_j)}}\right)^2 \\ &= (\mathbf{S}\boldsymbol{\omega})^\top \Delta(\mathbf{S}\boldsymbol{\omega}).\end{aligned} \quad (5)$$

Then, the cost-sensitive regularization framework can be summarized into

$$\begin{aligned}\arg\min_{\boldsymbol{\omega},\mathbf{W}} \quad & \left\{\|\boldsymbol{\Phi}(\mathbf{S}\boldsymbol{\omega}) - \mathbf{y}\|_2^2 + \mu(\mathbf{S}\boldsymbol{\omega})^\top \Delta(\mathbf{S}\boldsymbol{\omega}) + \lambda\|\mathbf{W}\|_{\mathrm{F}}^2\right\} \\ \text{s.t.} \quad & \sum_{j=1}^{N_e} \mathbf{W}_{j,j} = 1,\ \forall\, \mathbf{W}_{j,j} \geq 0,\end{aligned} \quad (6)$$

in which the parameters μ and λ are tradeoffs among the three components.

Cost Interval Optimization on Hypergraph Structure

Since the accurate cost value is difficult to obtain, we utilize the cost interval to optimize the hypergraph structure and present our proposed CIHL method. Here, we take the binary classification as the example. In binary classification, we fix the cost of the less important class as 1, and then we just consider the cost interval of the more important class. In the above section, cost-sensitive hypergraph learning assumes the true cost value is known in advance, but generally the true cost is in the cost interval $[C_{max}, C_{min}]$, so the cost-sensitive hypergraph cannot work directly. Here, we select surrogate cost to guide the learning on hypergraph structure with cost interval optimization.

Since the true cost is hard to determine, we need to find the surrogate cost c^* to guide the optimization, and the surrogate classifier h^* is expected to as effective as the classifier h^t, which is constructed with true cost. However, c^t and h^t are not available. So it is a difficult work to find the surrogate cost c^* and classifier h^*. It is noted that the true cost is between the C_{max} and C_{min}. Then the most direct way is to try all the value in the cost interval and minimize the value of all total cost, which is inefficient. In order to find a surrogate cost to match the condition, we formulate the problem as

$$\begin{aligned}\min_{h,c_*} \quad & L(h, c^*) \\ \text{s.t.} \quad & p(L(h,c) < \theta) > 1 - \varphi,\ \forall\, c \in [c_{min}, c_{max}], \\ & c_{min} \leq c^* \leq c_{max},\end{aligned} \quad (7)$$

where $L(h,c)$ represents the empirical risk, and is defined as $L(h,c) = \sum_{i=1}^{N_v} l(c, \rho_i, y)$. ρ_i is the label of testing data i, which can be calculated by $\rho_i = s_i\omega$. ω is calculated by the cost-sensitive hypergraph h with equation (6). The loss of misclassifying sample s_i is defined as $l(c,\rho_i,y) = cI(\rho_i \neq y \wedge y = +) + I(\rho_i \neq y \wedge y = -)$, $+$ is the label of the more important class and $-$ is the label of the less important class. The equation (7) guarantees that all of the possible total costs are small enough with a high probability.

But according to the infinite constraints, it is difficult to get the optimal solutions. To overcome this problem, according to (Liu and Zhou 2010), we utilize the worst-case risk to promise all of the constraints can be obeyed. This is because if the surrogate classifier h^* is the worst-case classifier and it meet the constraints in the (7), and then it should satisfy the following equation: $h_* = \arg\min_h \sup_c L(h,c)$, and

$p(\sup_c L(h_*, c) < \theta) > 1 - \varphi$, so for any c, $p(L(h_*, c) < \theta) > 1 - \varphi$. It can be proved (Liu and Zhou 2010) that the worst-case risk can be achieved when the surrogate cost c_* is equal to C_{max}.

Considering that the true cost may be far from C_{max}, for example, $c^t = C_{min}$, so just optimizing the surrogate cost C_{max} is unable to make the true cost small enough. As explained in (Liu and Zhou 2010), the mean cost $C_{mean} = 0.5(C_{max} + C_{min})$ is the smallest maximal distortion of the true risk, so it is another best choice to guide the learning process.

After selecting the surrogate costs C_{max} and C_{mean}, we can use these to guide the optimization of cost-sensitive hypergraph simultaneously. This procedure can be summarized in two steps. In the first step, we utilize C_{max} as the surrogate cost to optimize the objective function and learn a set of cost-sensitive hypergraph structures with different parameter values. In the second step, we utilize the C_{mean} as the surrogate cost to select the smallest total cost on the valid dataset, and then the corresponding hypergraph structure is selected as the unique solution.

Algorithm 1 The workflow of our proposed CIHL method.

The training data $S_{train} = \{s_{train_1}, s_{train_2}, \cdots, s_{train_n}\}$.
The testing data $S_{test} = \{s_{test_1}, s_{test_2}, \cdots, s_{test_n}\}$.
The valid data $S_{valid} = \{s_{valid_1}, s_{valid_2}, \cdots, s_{valid_n}\}$.
Cost interval: $\{C_{min}, C_{max}\}$
A set of parameters: $\{\lambda, \mu\}$
Output:

1: For each λ, μ, construct a cost-sensitive hypergraph-based classifier with C_{max}.
2: Obtain a set of mapping vectors $\omega(k)$ by minimizing the worst case total cost.
3: Calculate the label ρ^k on valid data by $\rho_i^k = s_{valid_i} \times \omega(k)$ with each mapping vector.
4: Use C_{mean} to evaluate $h(k)$ by calculating the $L(\rho^k, C_{mean})$.
5: **return** the classifier $h(k) = \arg\min_{h_k} L(h_k, C_{mean})$

Experiment

In this section, we introduce the testing datasets, compared methods, experimental results and discussions.

The Testing Datasets

In our experiments, we employ the widely used eight data from NASA Metrics Data Program (NASA) dataset (Menzies, Greenwald, and Frank 2007), including CM1, KC3, MC2, MW1, PC1, PC3, PC4, PC5 and seven data from binary UCI Machine Learning Repository (UCI)(Lichman 2013), including haberman, heartstatlog, sonar, SPET, SPECTF, wdbc, wpbc to evaluate the performance of our method.

Compared Methods

To evaluate the performance of our method, we compare CIHL with several state-of-the-art methods in the latest three years, which are introduced as follow:

1. Non-negative Sparse Graph Based Label Propagation (NSGLP) (Zhang, Jing, and Wang 2017). In this graph-based learning method, the Laplacian score sampling strategy was used to label the defect-free samples and solve the imbalance problem between the data. Then, it employed a nonnegative sparse algorithm to calculate the weight of the relationship graph and guide the learning process. At last, it utilized the label propagation method to predict the label of testing samples.
2. Cost-sensitive Discriminative Dictionary Learning (CDDL) (Jing et al. 2014). This method utilized multiple dictionaries, sparse representation coefficients and misclassification cost to construct a cost-sensitive discriminative dictionary learning (CDDL) method.
3. Our proposed method with three fixed cost values, *i.e.*, C_{max}, C_{mean}, C_{min}, which is named as $CSHL_{MAX}$, $CSHL_{MEAN}$, $CSHL_{MIN}$.

In order to prove the effectiveness of hypergraph structure, we compare our method with two learning methods with cost interval optimization.

1. CISVM (Liu and Zhou 2010). In this method, the author constructed a SVM with cost intervals optimization.
2. cisLDM (Zhou and Zhou 2016). cisLDM optimized the margin distribution on labeled and unlabeled data with cost interval information.

Table 1: Cost Intervals in Our Experiments.

[1,5], [1,10], [1,15], [6,10], [5,15], [5,20], [11,15], [10,20], [10,25], [15,25], [15,30], [16,20], [20,30]

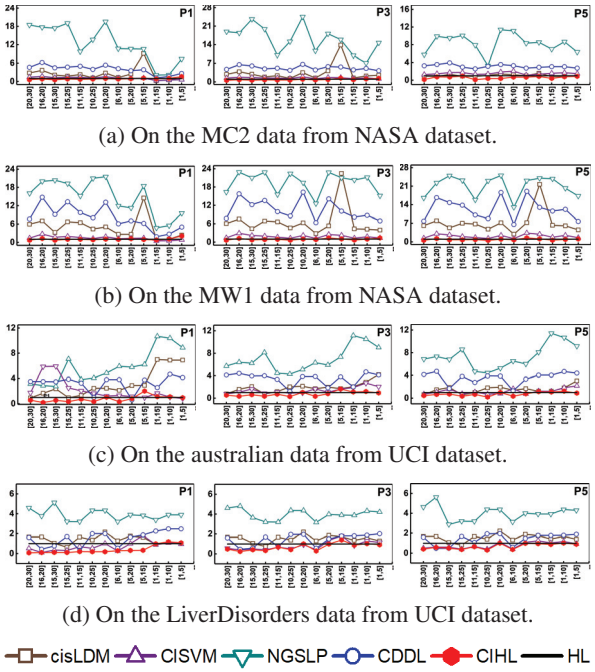
Experimental Settings

In experiments, we randomly divide the data into three parts, *i.e.*, $1/3$ data for training, $1/3$ data for testing, and $1/3$ data for validation. The data partition process repeats 30 times and the average performances are used for comparison. Both of the two parameters, *i.e.*, λ, μ, are selected for each cost interval from the set of $0.01, 0.1, 1, 10, 100$. α in equation (3) is set as 0.05 in our experiments.

Since the true cost is unknown, we need to evaluate the performance of all compared methods in different situations. So quadrisection points of an interval are considered in turn as the true cost: $C_{min} + (C_{max} - C_{min}) \times 0, 0.25, 0.5, 0.75, 1$. We call them test points P1-P5. The cost intervals with different widths used in our experiments are shown in Table 1.

Experimental Results

Experimental results are shown in Table 2 and Table 3. In our experiment, we have evaluated each method on 15 data from two datasets. For each data, we have used 13 cost intervals and each with 5 test points, and then we have conducted 975 tests. In Table 2 and Table 3, for result of two compared methods at each test point, the sum of win, tie and loss is



(a) On the MC2 data from NASA dataset.

(b) On the MW1 data from NASA dataset.

(c) On the australian data from UCI dataset.

(d) On the LiverDisorders data from UCI dataset.

Figure 3: The experimental results with different cost intervals. X-axis shows cost intervals, Y-axis is the ratio of the total cost of each method against that of hypergraph-based classifier without cost

equal to 195, and the methods with more number of bold results are more competitive.

Due to the page limit, in Figure 3, we show the detailed results between our method and other state-of-the-art methods on four data from NASA and UCI dataset. In Figure 4, we provide the comparison between our method and cost-sensitive hypergraph classifier optimization with different fixed costs on the other four data from NASA and UCI dataset, respectively.

From the results, we can notice that our method always performs better than all compared methods. More specifically, we can have the following observations.

1. At P_1, the true cost is C_{min}. In this test point, $CSHL_{MIN}$ uses the true cost to guide the learning process, and its results are better than $CSHL_{MAX}$ and $CSHL_{MEAN}$. Although we use the C_{max} to optimize the learning process in the first step of our method, we can find our method is also better than $CSHL_{MIN}$. The similar results can be found when compared with other state-of-the-art methods.
2. At P_2, the true cost is $0.5(C_{min} + C_{mean})$. In this test point, no methods use the true cost, and our method beats all compared methods in the two comparisons. We can find that the performances of $CSHL_{MIN}$ and $CSHL_{MEAN}$ are better than $CSHL_{MAX}$. The reason is that the true cost is between C_{min} and C_{mean}, and the cost of $CSHL_{MAX}$ is little far from the true cost. Apparently, the uncertain cost value will affect the performance of classifier.

Table 2: $Win/tie/loss$ counts of method-in-row compare with method-in-colum after t-test at 95 percent significant level on NASA and UCI dataset.

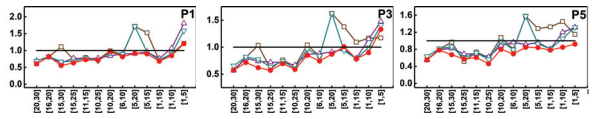
		$CSHL_{MIN}$	$CSHL_{MEAN}$	$CSHL_{MAX}$
p_1	$CIHL$	**98/13/84**	**110/9/76**	118/9/68
	$CSHL_{MAX}$	36/47/112	73/40/82	
	$CSHL_{MEAN}$	65/23/107		
p_2	$CIHL$	**135/24/36**	**132/10/53**	**125/38/32**
	$CSHL_{MAX}$	66/42/87	73/40/82	
	$CSHL_{MEAN}$	**111/23/61**		
p_3	$CIHL$	**127/33/35**	**132/11/52**	**128/39/28**
	$CSHL_{MAX}$	**121/47/27**	57/35/103	
	$CSHL_{MEAN}$	**126/7/62**		
p_4	$CIHL$	**132/14/49**	**129/29/37**	**126/22/47**
	$CSHL_{MAX}$	**126/34/35**	77/40/78	
	$CSHL_{MEAN}$	**122/23/50**		
p_5	$CIHL$	**111/9/75**	**115/5/71**	**123/19/53**
	$CSHL_{MAX}$	**102/47/46**	61/93/41	
	$CSHL_{MEAN}$	**125/23/47**		

3. At P_3, the true cost is C_{mean}. The performance of $CSHL_{MEAN}$ is better than $CSHL_{MIN}$ and $CSHL_{MAX}$, and in this test point, our method is sightly better than $CSHL_{MEAN}$ and significantly better than other methods.
4. At P_4, the true cost is $0.5(C_{mean} + C_{max})$, in this point, we can find $CSHL_{MIN}$ is significantly worse than other methods, while $CSHL_{MEAN}$ and $CSHL_{MAX}$ are comparable. Our proposed method CIHL is significantly better than other state-of-the-art methods.
5. At P_5, the true cost is C_{max}. The performance of $CSHL_{MEAN}$, $CSHL_{MAX}$, CIHL are all significantly better than $CSHL_{MIN}$. Additionally, CIHL shows better performance compared with $CSHL_{MAX}$. The results prove the effectiveness of cost interval optimization.

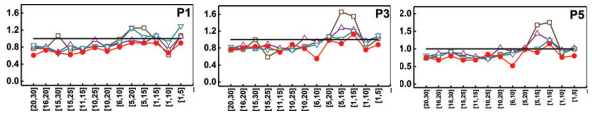
Discussion

Compared with state-of-the-art methods, *i.e.*, NGSLP, CDDL, cisLDM, CISVM, our method CIHL achieves the best performances. The results can be contributed to two advantages of CIHL. First, we utilize cost intervals to optimize our learning method. During the optimization process, we aim to minimize the total cost instead of decreasing the amount of misclassifications. So when compared with the methods without cost-sensitive learning, *i.e.*, NGSLP, the superiority of our method is obvious. We can also notice that all learning methods with cost interval optimization, including CISVM, cisLDM, and our method, outperform NGSLP and CDDL.

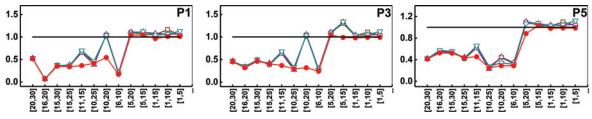
Second, CIHL constructs a hypergarph structure to formulate the correlations among the data. As shown in the experimental results, our proposed method CIHL has better performances than graph-based method, *i.e.*, NGSLP. The reason for these phenomenons can be concluded as follows. In our method, we use the hypergraph structure, which can adequately describe the high-order correlations under the dataset. In the hypergraph structure, each hyperedge can connect more than two vertices, so it can model the complex relationships between the features and the labels. Af-



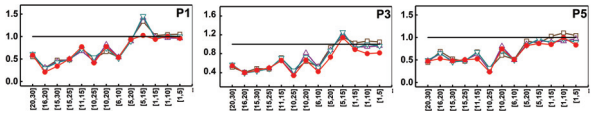
(a) On the CM1 data from NASA dataset.



(b) On the KC3 data from NASA dataset.



(c) On the SPECT data from UCI dataset.



(d) On the wpbc data from UCI dataset.

Figure 4: The experimental results with different cost intervals. X-axis shows cost intervals, Y-axis is the ratio of the total cost of each method against that of hypergraph-based classifier without cost

ter the hypergraph is constructed, we can use the structure to explore the relationships among testing data and utilize the hypergraph manifold to determined the label in a semi-supervised way. As for traditional graph structure, the pairwise connection is inability to explore the complex information adequately. The similar results also can be found in the comparison with CISVM and cisLDM, which also use the cost interval in the learning process. In these two methods, even though they used the cost interval to guide the learning process, our method can still achieve much better performance due to the advantage of hypergraph structure.

Moreover, as shown in the Figure 4, we notice that our method also performs the best compared with cost-sensitive hypergraph learning with fixed cost values. In this comparison, we find that the methods learning with true cost value perform better than other methods. This phenomenon shows that the uncertain cost value will limit the performance of the classifier. Additionally, without precise cost value, our method is better than or comparable to the classifier learning with true cost, and the contrast results indicate that our method is effective and more practical. The better results can be dedicated to that we utilize C_{max} as the surrogate cost to guide the learning process to satisfy the constraints, and then use the smallest maximal distortion C_{mean} to select the classifier.

Table 3: $Win/tie/loss$ counts of method-in-row compare with method-in-colum after t-test at 95 percent significant level on NASA and UCI dataset.

		$CDDL$	$NGSLP$	$CISVM$	$cisLDM$
p_1	$CIHL$	**140/49/5**	**137/52/6**	**95/52/48**	**128/51/16**
	$cisLDM$	89/50/56	91/52/10	23/50/122	
	$CISVM$	**132/62/1**	133/19/43		
	$NGSLP$	56/52/87			
p_2	$CIHL$	**144/50/1**	**133/51/11**	**90/51/54**	**130/51/14**
	$cisLDM$	86/41/68	89/49/57	**123/50/22**	
	$CISVM$	**140/54/1**	**124/61/10**		
	$NGSLP$	60/55/80			
p_3	$CIHL$	**123/46/261**	**115/39/41**	**123/51/21**	**116/50/29**
	$cisLDM$	97/50/48	**101/52/33**	89/30/76	
	$CISVM$	**127/38/30**	**120/32/43**		
	$NGSLP$	**100/52/43**			
p_4	$CIHL$	**144/51/0**	**131/39/25**	**101/40/54**	**141/41/13**
	$cisLDM$	88/50/60	86/71/38	**123/39/33**	
	$CISVM$	**122/43/30**	**120/52/23**		
	$NGSLP$	71/36/88			
p_5	$CIHL$	**142/49/4**	**129/51/15**	**98/41/56**	**123/39/33**
	$cisLDM$	72/41/82	86/51/58	**101/51/23**	
	$CISVM$	**124/32/35**	**113/32/50**		
	$NGSLP$	62/52/81			

Conclusion

In real-world classification tasks, the costs for different classification errors vary a lot while the precise cost of different categories is still challenging to obtain, which requires rich domain knowledge. In this paper, we propose a hypergraph learning method with cost interval optimization. In this method, the hypergraph structure is used to formulate the high-order correlations among the data, and the cost-sensitive learning with cost interval optimization is conducted to estimate the cost-oriented data labels. We have conducted experiments on two groups of datasets and experimental results have demonstrated better performance compared with the state-of-the-art methods.

Although the performance of the CIHL shows its advantage in classification, there are still several limitations. One important limitation is that how to incorporate domain knowledge in the cost related formulation, which is very essential on modelling the data correlations, has not been investigated.

Acknowledgments. This work was supported in part by the National Key R and D Program of China under Grant 2017YFC011300, in part by the National Natural Science Funds of China under Grant 61671267 and Grant 61527812, in part by National Science and Technology Major Project (No. 2016ZX01038101), MIIT IT funds (Research and application of TCN key technologies) of China, and The National Key Technology R and D Program (No. 2015BAG14B01-02).

References

Bertoni, A.; Frasca, M.; and Valentini, G. 2011. Cosnet: A cost sensitive neural network for semi-supervised learning in graphs. In *Proceedings of Machine Learning and Knowledge Discovery in Databases*, 219–234.

Chan, P. K., and Stolfo, S. J. 1998. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, 164–168.

Chen, F.; Gao, Y.; Cao, D.; and Ji, R. 2015. Multimodal hypergraph learning for microblog sentiment prediction. In *2015 IEEE International Conference on Multimedia and Expo*, 1–6.

Domingos, P. M. 1999. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining*, 155–164.

Gao, Y.; Wang, M.; Tao, D.; Ji, R.; and Dai, Q. 2012. 3-d object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing* 21(9):4290–4303.

Gao, Y.; Ji, R.; Cui, P.; Dai, Q.; and Hua, G. 2014. Hyperspectral image classification through bilayer graph-based learning. *IEEE Transactions on Image Processing* 23(7):2769–2778.

Huang, Y.; Liu, Q.; Zhang, S.; and Metaxas, D. N. 2010. Image retrieval via probabilistic hypergraph ranking. In *CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, 3376–3383.

Jing, X.; Ying, S.; Zhang, Z.; Wu, S.; and Liu, J. 2014. Dictionary learning based software defect prediction. In *Proceedings of the 36th International Conference on Software Engineering*, 414–423.

Kai, M. T. 2002. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering* 14(3):659–665.

Kapoor, A.; Qi, Y. A.; Ahn, H.; and Picard, R. W. 2005. Hyperparameter and kernel learning for graph based semi-supervised classification. In *Proceedings of Advances in Neural Information Processing Systems*, 627–634.

Lichman, M. 2013. UCI machine learning repository.

Liu, X., and Zhou, Z. 2010. Learning with cost intervals. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 403–412.

Liu, A.; Jun, G.; and Ghosh, J. 2009. A self-training approach to cost sensitive uncertainty sampling. *Machine Learning* 76(2-3):257–270.

Liu, M.; Miao, L.; and Zhang, D. 2014. Two-stage cost-sensitive learning for software defect prediction. *IEEE Transactions on Reliability* 63(2):676–686.

Masnadi-Shirazi, H., and Vasconcelos, N. 2010. Risk minimization, probability elicitation, and cost-sensitive svms. In *Proceedings of International Conference on Machine Learning*, 759–766.

Menzies, T.; Greenwald, J.; and Frank, A. 2007. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering* 33(1):2–13.

Sheng, S., and Ling, C. X. 2005. Hybrid cost-sensitive decision tree. In *Proceedings of The European Conference on Machine Learning & Principles and Practice of Knowledge Discovery in Databases*, 274–284.

Su, L.; Gao, Y.; Zhao, X.; Wan, H.; Gu, M.; and Sun, J. 2017. Vertex-weighted hypergraph learning for multi-view object classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2779–2785.

Sun, L.; Ji, S.; and Ye, J. 2008. Hypergraph spectral learning for multi-label classification. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 668–676.

Turney, P. D. 2002. Types of cost in inductive concept learning. *Computing Research Repository* cs.LG/0212034.

Wang, S.; Li, Z.; Chao, W.; and Cao, Q. 2012. Applying adaptive over-sampling technique based on data density and cost-sensitive SVM to imbalanced learning. In *Proceedings of The 2012 International Joint Conference on Neural Networks*, 1–8.

Wu, G., and Chang, E. Y. 2003. Class-boundary alignment for imbalanced dataset learning. *Proceedings of International Conference on Machine Learning* 49–56.

Yang, P.; Zhao, P.; Zheng, V. W.; and Li, X. 2015. An aggressive graph-based selective sampling algorithm for classification. In *Proceedings of 2015 IEEE International Conference on Data Mining*, 509–518.

Zhang, Y.; Zhang, X.; Yuan, X.; and Liu, C. 2016. Large-scale graph-based semi-supervised learning via tree laplacian solver. In *Proceedings of The Association for the Advancement of Artificial Intelligence*, 2344–2350.

Zhang, Z.; Jing, X.; and Wang, T. 2017. Label propagation based semi-supervised learning for software defect prediction. *Automated Software Engineering* 24(1):47–69.

Zhou, Y., and Zhou, Z. 2016. Large margin distribution learning with cost interval and unlabeled data. *IEEE Transactions on Knowledge and Data Engineering* 28(7):1749–1763.

Zhou, D.; Huang, J.; and Schölkopf, B. 2006. Learning with hypergraphs: Clustering, classification, and embedding. In *Proceedings of Neural Information Processing Systems*, 1601–1608.

Zhu, X.; Zhu, Y.; Zhang, S.; Hu, R.; and He, W. 2017. Adaptive hypergraph learning for unsupervised feature selection. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 3581–3587.