Logistic Boosting Regression for Label Distribution Learning*

Chao Xing, Xin Geng†, Hui Xue
Key Lab of Computer Network and Information Integration (Ministry of Education)
School of Computer Science and Engineering, Southeast University, Nanjing 211189, China.
{xingchao, xgeng, hxue}@seu.edu.cn

Abstract

Label Distribution Learning (LDL) is a general learning framework which includes both single label and multi-label learning as its special cases. One of the main assumptions made in traditional LDL algorithms is the derivation of the parametric model as the maximum entropy model. While it is a reasonable assumption without additional information, there is no particular evidence supporting it in the problem of LDL. Alternatively, using a general LDL model family to approximate this parametric model can avoid the potential influence of the specific model. In order to learn this general model family, this paper uses a method called Logistic Boosting Regression (LogitBoost) which can be seen as an additive weighted function regression from the statistical viewpoint. For each step, we can fit individual weighted regression function (base learner) to realize the optimization gradually. The base learners are chosen as weighted regression tree and vector tree, which constitute two algorithms named LDLogitBoost and AOSO-LDLogitBoost in this paper. Experiments on facial expression recognition, crowd opinion prediction on movies and apparent age estimation show that LDLogitBoost and AOSO-LDLogitBoost can achieve better performance than traditional LDL algorithms as well as other LogitBoost algorithms.

1. Introduction

Learning with ambiguity is a hot topic in recent machine learning and computer vision research. A learning method is essentially building a mapping from the instances to the label space. We can use a description degree d_x^y as a numerical indicator to measure the relationship of the label y to the instance x and indicate the relative label intensity. As can be seen from Fig.1, there are mainly three ways to label an instance in existing learning paradigms: For the single-label case (a), the label y_2 fully describes the instance, so $d_x^{y_2} = 1$. For the multi-label case (b), each of the two positive labels y_2 and y_4 by default describes 50% of the instance, so $d_x^{y_2} = d_x^{y_4} = 0.5$. Finally, (c) represents a general case of label distribution, which satisfies the constraints $d_x^y \in [0, 1]$ and $\sum_y d_x^y = 1$. d_x^y of the label distribution is neither the label confidence nor label population. It actually represents the description degree that the corresponding label y describes the instance x. In other words, label distribution can not only model the ambiguity of "what describes the instance", but also deal with the more general ambiguity of "how to describe the instance". This example illustrates that label distribution is more general than both the single-label and multi-label cases, and thus can provide more flexibility in the learning process. Inspired by this observation, Geng and Ji [9, 7] proposed a general method named Label Distribution Learning (LDL) which learning a mapping from the instance to its label distribution.

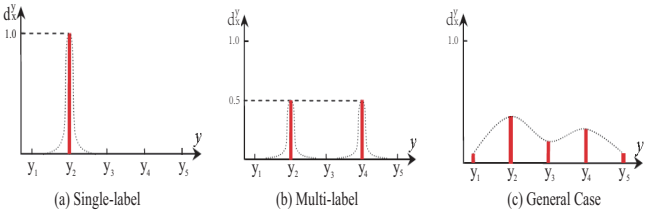

Figure 1. Three ways to label an instance

There are a lot of LDL algorithms which can fit some real applications well. For example, Geng et al. [12] proposed two algorithms named IIS-LDL and CPNN to deal with facial age estimation problems by constructing the age distribution, because the faces at close ages look quite similar. Geng and Ji [9, 7] proposed BFGS-LDL algorithm by using the effective quasi-Newton optimization to further improve IIS-LDL. Geng et al. [14, 13] proposed the AGES algorithm based on the subspace trained on a data structure called aging pattern vector. This algorithm was further extended to Adaptive Label Distribution Learning (ALDL) by Geng et al. [10]. Yang et al. [22] tried to combine the LDL with deep learning methods called Deeply Label Distribu-

*This work was supported by the National Natural Science Foundation of China (61273300, 61232007, 61375057) and the Natural Science Foundation of Jiangsu Province (BK20140022, BK20131298).

†X. Geng is the corresponding author.

tion Learning (DLDL) to deal with the apparent age estimation problem. Zhang et al. [23] proposed a LDL method for crowd counting in public video surveillance. As for expression recognition, LDL can also be adapted to the emotion distribution learning [24], because an expression rarely expresses pure emotion, but often a mixture of different emotions, and each emotion has its own intensity. Geng and Xia [11] extended label distribution to multivariate label distributions and proposed two algorithms to learn from the multivariate label distributions based on the Jeffery divergence. Futher more, LDL can also be combined with SVR named LDSVR [8]. This algorithm was used for pre-release prediction of crowd opinion on movies. Note that the usage of LDL algorithms is not limited to the prediction of label distribution, but includes marketing strategy, the design of computer vision system, interests recommendation, etc.

One of the main assumptions made in traditional LDL algorithms is that the description degree d_x^y can be represented by the form of maximum entropy model [2]. In the exponential part of this model, it uses a linear combination of the input space assigned to the instance. Such an approach is limited for the following reasons: firstly, the linear combination of the exponential part has limited description which is not applicable to all LDL problems. Secondly, for particular application, in order to improve the performance with this specific model, we need to improve the loss function. This will make the loss function more complex with the complicate optimization strategies and may lead to over-fitting. Alternatively, replacing this exponential part with a general function to approximate this parametric model can avoid the potential influence of the specific model. This general function is variable such as decision tree or the linear combination. So we can use this general function to represent any specific function which constitutes a LDL model family. Thus in order to find a unified learning framework for these various models of LDL model family with the same optimization strategy, we can assume that the model is the additive. Therefore a unique method called Logistic Boosting Regression (LogitBoost) [6] can be used to learn this general model family all together. On the one hand, LogitBoost is a combination of the boosting method and the logistic regression [4]. It uses the coordinate descent optimization method over the function space based on the second derivative Hessian matrix. On the other hand, LogitBoost can be seen as an additive weighted function regression from a statistical viewpoint. We can fit individual weighted regression function (base learner) to realize the optimization step by step. Although any regression function of the model family can be used for LDL, it has been reported in the literature [20] that traditional LogitBoost turns out to have some drawbacks because of the over approximation such as using a second-order (diagonal) approximation to fit individual regression function. Therefore, Li [17, 16] proposed robust LogitBoost and ABC-LogitBoost which choose the 'base class' for improving single-label classification accuracy adaptively. Sun et al. [20] proposed a base learner named vector tree which can fit an united regression function of each class at the same time. This model used by LogitBoost is called AOSO-LogitBoost. To some extent, ABC-LogitBoost is a special case of AOSO-LogitBoost with a less flexible tree model.

In this paper, we extend the maximum entropy model of traditional LDL to a more general LDL model family. Then we use LogitBoost method to learn this model. In this way, the boosting method can also be applied to LDL which can "boost" the accuracy of any given learning algorithm especially simple learning algorithms and avoid over fitting [19]. Next, two base learners are chosen for LDL model family, namely weighted regression tree [3] and vector tree [20], to constitute two algorithms called LDLogitBoost as well as AOSO-LDLogitBoost. The rest of the paper is organized as follows. Section 2 introduces the LogitBoost for the LDL and proposes two LDL algorithms named LDLogitBoost and AOSO-LDLogitBoost. In Section 3, the experimental results are reported. Finally, several conclusions are drawn in Section 4.

2. Proposed Algorithms

2.1. LDL model family

We begin with the basic settings for the LDL Problems. First of all, all the labels $Y = \{y_1, \ldots, y_j, \ldots, y_C\}$ with a non-zero description degree are actually the "correct" labels to describe the instance $x \in X$ but just with the different importance measured by d_x^y, where X represents the input space $X = \mathbb{R}^q$. C is the number of the labels. Secondly, we can normalize the description degree to make $d_x^y \in [0, 1]$ and $\sum_y d_x^y = 1$ to constitute the label distribution. Measuring the description degree is very important, as one can know how much each label description degree is and how many labels are related to the particular instance. Consequently, the LDL can be formulated that given a training set $S = \{(x_1, D_1), \ldots, (x_i, D_i), \ldots, (x_N, D_N)\}$, where $x_i \in X$ and $D_i = \{d_{x_i}^{y_1}, \ldots, d_{x_i}^{y_j}, \ldots, d_{x_i}^{y_C}\}$ is the label distribution associated with the instance x_i. N is the number of the instances. The goal of LDL is to learn a conditional probability mass function from S:

$$p(y_j|x_i) = \frac{1}{Z} exp(F_j(x_i)) \tag{1}$$

where $Z = \sum_{s=1}^{C} exp(F_s(x_i))$ is the normalization factor. Here $F_j(x)$ is an element of C-dimensional function vector $\mathbf{F}(x) = \{F_1(x), \ldots, F_j(x), \ldots, F_C(x)\}$. It is any function which constitutes a LDL model family. From a viewpoint of LogitBoost, $\mathbf{F}(x)$ of this LDL model family can also be called base (weak) learner. For example, we can assume

that

$$F_j(x_i) = \beta_j^T x_i \tag{2}$$

Substituting Eq. (2) to Eq. (1), we can get traditional LDL model:

$$p(y_j|x_i) = \frac{1}{Z} exp(\beta_j^T x_i) \tag{3}$$

where $Z = \sum_{s=1}^{C} exp(\beta_s^T x_i)$. This model is maximum entropy model which can be learned by traditional LDL algorithms such as IIS-LDL and BFGS-LDL [9, 7]. On the one hand, traditional LDL model may have a potential limitation on the performance of the algorithm. On the other hand, traditional LDL model could not be adapted to various applications. Therefore, this LDL model family can avoid the potential influence of the specific model and different function model can be selected for different applications.

There are different criteria that can be used to measure the distance or similarity between two distributions. For example, if the Kullback-Leibler divergence (K-L) is used as the distance measure between the true label distributions and the predicted label distributions, then the loss of this model is determined by:

$$L = \sum_{i=1}^{N} loss(x_i) \tag{4}$$

$$loss(x) = \sum_{j=1}^{C} d_x^{y_j} \log \frac{d_x^{y_j}}{p(y_j|x)} \tag{5}$$

Minimizing Eq. (5) is equivalent to minimizing the negative log-likelihood loss:

$$loss(x) = -\sum_{j=1}^{C} d_x^{y_j} \log p(y_j|x) \tag{6}$$

Substituting Eq. (1) to Eq. (6), we can get:

$$loss(x; \mathbf{F}) = -\sum_{j=1}^{C} d_x^{y_j} \log \frac{1}{Z} exp(F_j(x_i)) \tag{7}$$

Eq. (7) is the target function for the optimization. However, different models or loss functions may lead to different optimization strategies. Thus in order to find a unified learning framework for these various models or loss functions with the same optimization strategy, we can use Logistic Boosting Regression (LogitBoost) [6] to avoid this problem, which assume $\mathbf{F}$ is a flexible additive function:

$$\mathbf{F}^{(T)}(x) = \sum_{t=1}^{T} v_t \mathbf{f}^{(t)}(x, a_t) \tag{8}$$

where v_t is the shrinkage and T is the total steps of the additive model. $\mathbf{f}^{(t)}(x, a_t)$ are the t-th C-dimensional base(weak) function learners which correspond with the function $\mathbf{F}(x)$. Owing to its iterative nature of Eq. (8), in each step of the optimization, the best $\mathbf{f}^{(t)}(x)$ is added only based on $\mathbf{F}^{(t-1)}(x) = \sum_{s=1}^{(t-1)} v_s \mathbf{f}^{(s)}(x)$. Formally:

$$\mathbf{f}^{(t)}(x) = \arg\min_{f} \sum_{i=1}^{N} loss(x_i; \mathbf{F}^{(t-1)} + \mathbf{f}^{(t)}) \tag{9}$$

The function of $\mathbf{f}^{(t)}(x)$ can be obtained by using a strategy similar to the greedy stage wise algorithm [6], which is a well-know algorithm for LogitBoost. This algorithm suggests to a second-order (diagonal) approximation, which needs to calculate the first two derivatives of the log-likelihood loss function Eq. (6) with respective to the function $F_j(x)$. We can derive:

$$g_j(x) = \frac{\partial loss(x)}{\partial F_j} = -(d_x^{y_j} - p(y_j|x)) \tag{10}$$

$$h_{jj}(x) = \frac{\partial^2 loss(x)}{\partial F_j^2} = p(y_j|x)(1 - p(y_j|x)) \tag{11}$$

$$h_{js}(x) = \frac{\partial^2 loss(x)}{\partial F_j \partial F_s} = -p(y_j|x)p(y_s|x) \tag{12}$$

In this way, we can fit an individual regression function:

$$\begin{aligned} \sum_{i=1}^{N} f_j(x_i) &\leftarrow -\frac{\sum_{i=1}^{N} g_j(x_i)}{\sum_{i=1}^{N} h_{jj}(x_i)} \\ &\leftarrow -\frac{\sum_{i=1}^{N} \left(\frac{g_j(x_i)}{h_{jj}(x_i)} h_{jj}(x_i) \right)}{\sum_{i=1}^{N} h_{jj}(x_i)} \\ &\leftarrow -E_w(\frac{g_j(x)}{h_{jj}(x)}) \end{aligned} \tag{13}$$

where $E_w(\cdot)$ indicates a weighted expectation of x with the weight w. Equivalently, the Newton update $\mathbf{f}(x)$ solves the weighted least-squares approximation about $\mathbf{F}(\mathbf{x})$ to the log-likelihood:

$$\min_{f_j(x)} E_{w_j(x)} \left(f_j(x) - (-\frac{g_j(x)}{h_{jj}(x)}) \right)^2 \tag{14}$$

Thus it can be summarized as: At the each step, we can fit a weighted regression function on the training instances x_i for the j-th label class, with the target:

$$z_{i,j} = -\frac{g_j(x_i)}{h_{jj}(x_i)} \tag{15}$$

and the weight:

$$w_{i,j} = h_{jj}(x_i) \tag{16}$$

For identifiability, $\sum_{j=1}^{C} F_j(x) = 0$, i.e., the sum-to-zero constraint, is usually adopted in [6, 20, 16, 17]. Therefore, in order to satisfy the sum-to-zero constraint, we need to change the updated form:

$$F_j^{(t)}(x) \leftarrow F_j^{(t-1)}(x) + v_t \frac{C-1}{C}\left(f_j^{(t)}(x) - \frac{1}{C}\sum_{s=1}^{C} f_s^{(t)}(x)\right) \quad (17)$$

2.2. LDLogitBoost

We can choose any function of LDL model family for this individual regression function. The weighted regression tree is very common in LogitBoost [3]. It can be concluded: At the each step, we can fit individual weighted regression function on the training instances x_i for the j-th label class, with the target $z_{i,j}$ and the weight $w_{i,j}$. This method can be called LDLogitBoost and its pseudo code is given in Algorithm 1.

Algorithm 1 LDLogitBoost

```
F_j(x) = 0, p_{i,j} = 1/C, j = 0 to C and i = 1 to N
for t = 1 to T do
    for j = 1 to C do
        for i = 1 to N do
            Compute w_{i,j} by Eq. (16)
            Compute z_{i,j} by Eq. (15)
        end for
        Fit a weighted regression tree f_j^(t)(x) on the training instances x_i with targets z_{i,j} and weights w_{i,j}.
        Update F_j^(t)(x) by Eq. (17)
    end for
    Calculate p(y_j|x_i) by Eq. (1)
end for
```

2.3. AOSO-LDLogitBoost

Although any regression function of the model family can be used for LDL, it has been reported in the literature [20] that vector tree model may lead a better performance than other regression functions. Each node in the vector tree model is the vector where the unique classifier output is guaranteed by adding a sum-to-zero constraint. Fig. 2 gives us a vector tree model for a 3-class problem. A class pair is selected for each tree node adaptively.

For each internal node (filled), the pair is for computing split gain; For terminal nodes (unfilled), it is for node vector update. That is, at each step t, we can construct a vector tree model $\{\mathbf{a_m}, R_m\}_{m=1}^{M}$, where $\mathbf{a_m} = \{a_{m1}, \ldots, a_{mj}, \ldots, a_{mC}\}$ is a vector value with the sum of 0 and M is the number of terminal nodes. It updates the values of $\mathbf{F}(\mathbf{x})$ all together by first computing a rectangular

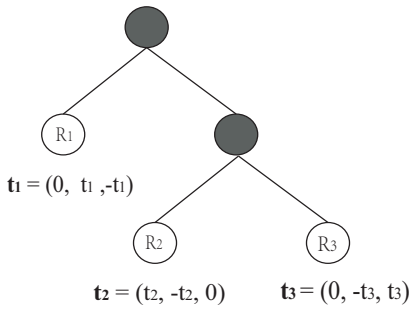

Figure 2. vector tree model

partition of the feature space $\{R_m^t\}_{m=1}^{M}$ and corresponding node vector value $\{\mathbf{a_m^t}\}_{m=1}^{M}$ at the same time, then incrementing F_i by $\mathbf{a_m^t}$ where m is the index of the region R_m^t such that $x_i \in R_m^t$:

$$F_j^{(t)}(x) \leftarrow F_j^{(t-1)}(x) + v_t \sum_{m=1}^{M} a_{m,j}^t I(x_i \in R_m^t) \quad (18)$$

where $I(\cdot)$ is the indication function and v_t is the shrinkage factor. So the loss function of the node becomes:

$$NodeLoss(a; R_m) = \sum_{x_i \in R_m} loss(x_i; \mathbf{F}) \quad (19)$$

As we can see, Eq. (19) is equivalent to Eq. (4) if the vector tree only has the root node. Minimizing the Eq. (19) allows us to obtain a set of nodes $\{a_m, R_m\}_{m=1}^{M}$ using the following procedures:

In the first step, to obtain the values $\mathbf{a_m}$ from a given R_m, we can simply take the minimization of Eq. (19):

$$\mathbf{a_m} = \arg\min_{\mathbf{a}} NodeLoss(\mathbf{a}; R_m) \quad (20)$$

where $\mathbf{a_m}$ is the vector value of the related terminal node as

$$a_{m,k} = \begin{cases} +(-\bar{g}/\bar{h}) & \text{if k=r} \\ -(-\bar{g}/\bar{h}) & \text{if k=s} \\ 0 & \text{otherwise} \end{cases} \quad (21)$$

with

$$\bar{g} = -\sum_{x_i \in R_m} (g_r(x_i) - g_s(x_i)); \quad (22)$$

$$\bar{h} = \sum_{x_i \in R_m} (h_{rr}(x_i) + h_{ss}(x_i) + 2h_{rs}(x_i)); \quad (23)$$

as well as the class pair r and s can be selected by:

$$r = \arg\max_j \{-g_j\} \quad (24)$$

$$s = \arg\max_j \{\frac{g_r - g_j}{h_{rr} + h_{jj} - 2h_{rj}}\} \quad (25)$$

In the second step, to obtain the partition $\{R_m\}_{m=1}^{M}$, we recursively perform binary splitting until there are M-terminal nodes. Suppose an internal node with n training instances, we fix on some feature and re-index all the n instances to their sorted feature values. Now we need to find the index n' with $1 < n' < n$ that maximizes the node gain defined as loss reduction after a division between the n'-th and $(n'+1)$-th instances. At the same time, we can calculate the NodeLoss as:

$$NodeLoss(a^*; R_m) = \frac{-\bar{g}^2}{2\bar{h}} \tag{26}$$

which contributes to the biggest node gain approximately as:

$$NodeGain(n') = \frac{-\bar{g}_L^2}{2\bar{h}_L} + \frac{-\bar{g}_R^2}{2\bar{h}_R} - \frac{-\bar{g}^2}{2\bar{h}} \tag{27}$$

The selection of a class pair (r, s) for a vector tree is adaptive. And the vector tree is a binary tree with the vector value. By using this model as the base (weak) learner, we can generate a new algorithm called AOSO-LDLogitBoost and its pseudo code is given in Algorithm 2, where "AOSO" means "Adaptive one vs one".

Algorithm 2 AOSO-LDLogitBoost

1: $F_j = 0, p_{i,j} = \frac{1}{C}$, $j = 0$ to C and $i = 1$ to N
2: **for** $t = 1$ to T **do**
3: Obtain $\{R_{t,m}\}_{m=1}^{M}$ by recursive region partition. Node split gain is computed as Eq. (27), where the class pair (r,s) is selected using Eq. (24) and Eq. (25).
4: Compute $\{a_{t,m}\}_{m=1}^{M}$ by Eq. (21), where the class pair (r,s) is selected using Eq. (24) and Eq. (25).
5: Update $F_j^{(t)}(x)$ by Eq. (18)
6: Calculate $p(y_j|x_i)$ by Eq. (1)
7: **end for**

3. Experiment

To demonstrate the effectiveness of the proposed LDLogitBoost and AOSO-LDLogitBoost algorithms, we perform experiments on three different databases: s-BU_3DFE (scores-Binghamton University 3D Facial Expression) [24], COPM (Crowd Opinion Prediction on Movies) [8] and ChaLearn Age Estimation Competition Data Set (CAECD) [5]. The first two data sets are popular label distribution data sets, and the last one is a single label data set. The additional prior knowledge of mean value and the standard deviation of the label distribution is given in the last data set.

3.1. Facial Expression Recognition

Most existing facial expression recognition methods assume the availability of a single emotion for each expression in training set. However, in practical applications, an expression rarely expresses pure emotion, but often a mixture of different emotions, and each emotion has its own intensity. Facial expression recognition is a LDL problem which can be seen from Fig. 3. The emotion distribution is represented by a curve. There are six values at the horizontal axis labeled by the six basic emotions: happiness (HAP), sadness (SAD), surprise (SUR), anger (ANG), disgust (DIS) and fear (FER). The values at the vertical axis represent the description degrees of each emotion. We need to learn a mapping from the facial expression image to its emotion distribution.

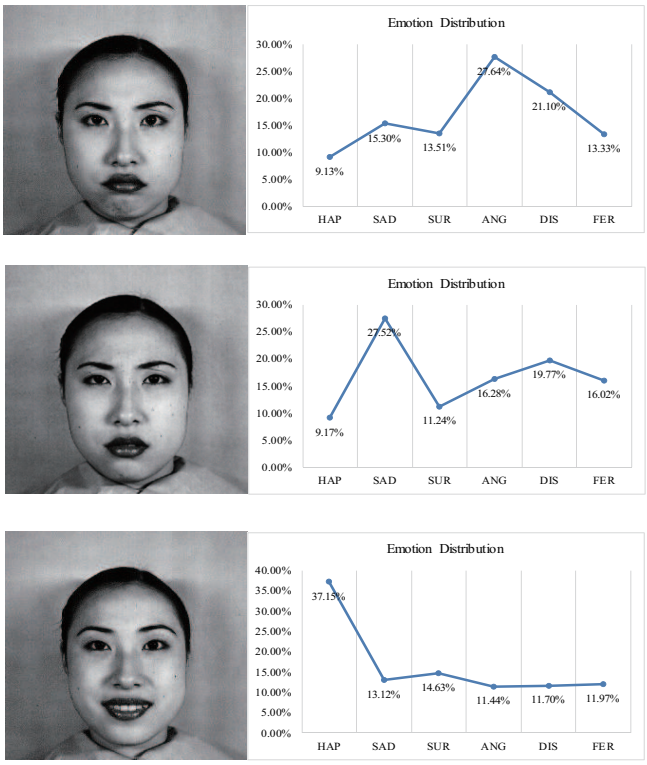

Figure 3. LDL of Expression Recognition

Here we use s-BU_3DFE data set which has 2500 instances. 23 students are asked to score the s-BU_3DFE database on 6 basic emotions (i.e., happiness, sadness, surprise, fear, anger and disgust) with a 5 level scale (5 represents the highest emotion intensity, while 1 represents the lowest emotion intensity). The average score of each emotion is used to represent the specific emotion intensity. The images are cropped manually so that the eyes are at the same positions, and then the cropped images are resized to 110*140 pixels. Features are extracted by the method of Local Binary Patterns (LBP) [1].

A nature choice of evaluation measures is the average similarity or difference between the real label distributions P_j and the predicted distributions Q_j. There are many measures for similarity/distance between two distributions,

Name	Formula
Kullback-Leibler(K-L)	$Dis_1 = \sum_{j=1}^{C} P_j ln \frac{P_j}{Q_j}$
Euclidean	$Dis_2 = \sqrt{\sum_{j=1}^{C}(P_j - Q_j)^2}$
Sørensen	$Dis_3 = \frac{\sum_{j=1}^{C}\lvert P_j - Q_j\rvert}{\sum_{j=1}^{C}(P_j + Q_j)}$
SquaredX^2	$Dis_4 = \sum_{j=1}^{C} \frac{(P_j - Q_j)^2}{P_j + Q_j}$
Fidelity	$Sim_1 = \sum_{j=1}^{C} \min(P_j, Q_j)$
Intersection	$Sim_2 = \sum_{j=1}^{C} \sqrt{P_j Q_j}$

Table 1. Evaluation Measure for LDL Algorithms

which are summarized in Table 1. For the four distance measures, "↓" indicates "the smaller is the better" and for the two similarity measures, "↑" indicates "the larger the better".

LDLogitBoost and AOSO-LDLogitBoost are compared with seven existing LDL methods. For each compared method, several parameter configurations are tested and the best performance is reported by 10-fold cross validation. AOSO-LDLogitBoost uses maximum 5000 steps with $v = 0.05$, while LDLogitBoost uses maximum 2000 steps with $v_t = 0.05$(when $t > 500$, we changed $v_t = 0.01$). For the traditional LDL methods, K in AA-KNN is set to 6. Linear kernel is used in PT-SVM. The number of hidden-layer neurons for AA-BP is set to 60. The insensitivity parameter ε is set to 0.1 of LDSVR. At the same time, the maximum steps in BFGS-LDL is 300 and IIS-LDL is 5000.

Table 2 reports the experimental results of LDLogitBoost, AOSO-LDLogitBoost and other LDL methods. The best performance on each measure is highlighted by boldface. The two-tailed T-tests with 0.05 significance level are performed to see whether the differences are statistically significant. As can be seen, AOSO-LDLogitBoost performs best on all criteria followed by LDLogitBoost. This implies that vector tree model is more competitive than weighted regression tree as well as maximum entropy model on this data set.

3.2. Crowd Opinion Prediction on Movies

The prediction of crowd opinion on movies is an interesting problem. Thousands of new movies are produced and shown in movie theaters each year, among which some are successful, many are not. On the one hand, the increasing cost and competition boosts the investment risk of the movie producers. On the other hand, the prevalent immodest advertisement and promotion makes it hard for movie audiences to choose a movie worth watching. Therefore, both sides demand a reliable prediction of what people will think about a particular movie before it is actually released or even during its planning phase.

Fig. 4 gives a typical example of such case. The movies $Twilight$ (a) and $I, Frankenstein$ (b) both have the same average rating $5.2/10$. But the top two popular ratings in the rating distribution of $Twilight$ are the lowest rating 1 (15.7%) and the highest rating 10 (15.3%), respectively, while those of $I, Frankenstein$ concentrate at the medium ratings 6 (21.4%) and 5 (20.1%). As a result, the budget/gross ratio of $Twilight$ is \$37M/\$191M and that of $I, Frankenstein$ is \$65M/\$19M. Obviously, the former movie is more worthy to invest and watch. This example illustrates that the average rating is not a good indicator of the crowd opinion. Therefore, the prediction of the rating distribution is more useful which is not limited to a gross prediction, but includes marketing strategy, advertising design, movie recommendation, etc.

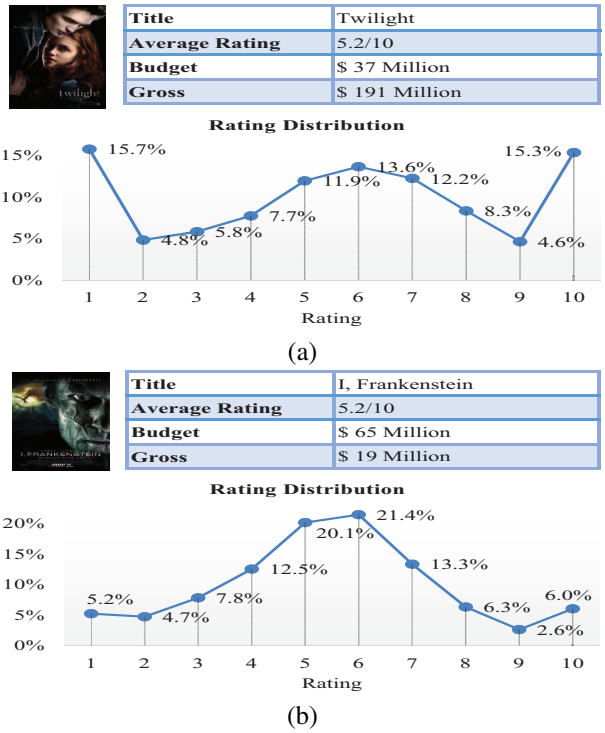

Figure 4. LDL of Crowd Opinions on movies $Twilight$ (a) and $I, Frankenstein$ (b)

Here we use the database named COPM (Crowd Opinion Prediction on Movies) which has 7755 movies and 54,242,292 ratings from 478,656 different users. The ratings come from Netflix, which are on a scale from 1 to 5 integral stars. Each movie has, on average, 6,994 ratings. The rating distribution is calculated for each movie as an indicator for the crowd opinion on that movie. The prerelease metadata is crawled from IMDB according to the unique movie IDs. There are both numeric and categorical attributes in this data set. Finally, all the attributes are normalized to the same scale through the min-max normalization.

To solve this problem, LDLogitBoost and AOSO-

Method	K-L ↓	Euclidean ↓	Sørensen ↓	SquaredX^2 ↓	Fidelity ↑	Intersectioin ↑
AOSO-LDLogitBoost	**.0491±.0023**	**.1263±.0033**	**.1200±.0031**	**.0467±.0020**	**.9881±.0005**	**.8800±.0031**
LDLogitBoost	.0515±.0027	.1297±.0039	.1236±.0036	.0493±.0024	.9874±.0006	.8764±.0036
BFGS-LDL	.0542±.0027	.1341±.0035	.1289±.0033	.0521±.0024	.9867±.0006	.8711±.0033
IIS-LDL	.0653±.0025	.1501±.0036	.1445±.0032	.0619±.0023	.9842±.0006	.8555±.0032
LDSVR	.0667±.0021	.1512±.0031	.1450±.0028	.0628±.0018	.9839±.0004	.8550±.0028
AA-KNN	.0743±.0031	.1549±.0036	.1464±.0042	.0697±.0036	.9821±.0004	.8536±.0008
AA-BP	.0808±.0063	.1648±.0076	.1595±.0065	.0760±.0061	.9804±.0006	.8405±.0017
PT-Bayes	.0830±.0037	.1659±.0044	.1606±.0049	.0766±.0039	.9803±.0004	.8394±.0010
PT-SVM	.0877±.0029	.1701±.0032	.1638±.0047	.0799±.0044	.9794±.0004	.8362±.0007

Table 2. Results on Facial Expression Recognition. AOSO-LDLogitBoost is better.

LDLogitBoost are compared with six existing LDL methods. For each compared method, several parameter configurations are tested and the best performance is reported by 10-fold cross validation. AOSO-LDLogitBoost uses maximum 500 steps with $v = 0.05$, while LDLogitBoost uses maximum 200 steps with $v_t = 0.3$. For the LDL methods, K in AA-KNN is set to 10 and the number of hidden-layer neurons for CPNN is set to 80. The insensitivity parameter ε is set to 1 of LDSVR. The number of hidden-layer neurons for AA-BP is set to 60. At the same time, the maximum steps in BFGS-LLD and IIS-LLD is 50.

Table 3 reports the experimental results of LDLogitBoost, AOSO-LDLogitBoost and other LDL methods. As can be seen, AOSO-LDLogitBoost performs best on all of other algorithms. But LDLogitBoost has the same performs with LDSVR. The reason might be two-fold. Firstly, LDSVR takes advantage of the large margin regression by a support vector machine. Secondly, the application of the kernel trick makes it possible for LDSVR to solve this problem in a higher-dimensional.

3.3. ChaLearn Age Estimation

In the age estimation competition organized by ChaLearn [5], apparent ages of images are provided. Uncertainty of each apparent age is induced because each image is labeled by multiple individuals. Such uncertainty makes this age estimation task be different from common chronological age estimation tasks. The age of each image in the data set is labeled by multiple individuals rather than its chronological age. For each image, its mean age μ and the standard deviation σ are given. As can be seen from Fig.5, the horizontal axis shows the apparent ages and the vertical axis represents the description degree d_x^y of each age y assigned with the facial image x. In this data set, totally 3,615 facial images (2,479 in the training data set and 1,136 in the validation data set), with their apparent ages and standard deviations, are provided. There are three steps of preprocessing of images. The facial region of each image is detected by the DPM model described in [18]. Then the detected face is fed to a public available facial point detector software [21] to detect five facial key points including the left/right eye centers, nose tip and left/right mouth corners. Finally, based on these facial points, we employed facial alignment for these facial images and resized to 256*256 pixels. The features are extracted by the method of Biologically Inspired Features (BIF) [15].

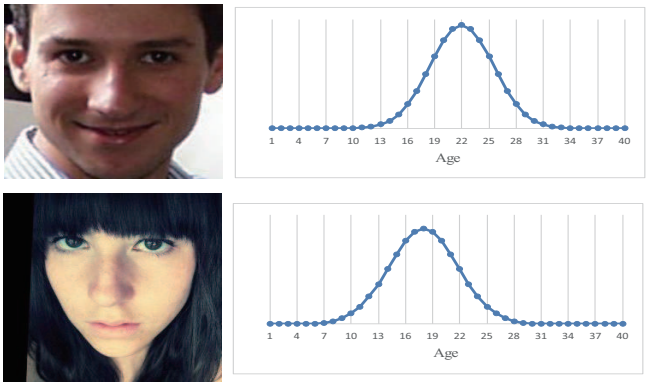

Figure 5. LDL of Apparent Age Estimation

In this competition, on the one hand, traditional single-label method can be used to learn a mapping from the apparent image to the given mean age μ. Thus, we can use traditional LogitBoost [6] and AOSO-LogitBoost [20] algorithms. On the other hand, we can use the mean age μ and the standard deviation σ to generate an age distribution assigned with each image. AOSO-LDLogitBoost, LDLogitBoost and BFGS-LDL can be used to learn a mapping from the apparent age image to its age distribution.

The performance of the age estimation is evaluated by Mean Absolute Error (MAE) and the formula provided by competition organization (named ϵ-Error), which is:

$$\epsilon = 1 - \exp(-\frac{(t-\mu)^2}{2\sigma^2}) \tag{28}$$

where t is the predicted age, μ is the mean apparent age and σ is the standard deviation. Each prediction is evaluated using the above formula, getting an error value between 0

Method	K-L ↓	Euclidean ↓	Sørensen ↓	SquaredX^2 ↓	Fidelity ↑	Intersectioin ↑
AOSO-LDLogitBoost	**.0855±.0037**	**.1547±.0030**	**.1521±.0032**	**.0837±.0034**	**.9778±.0009**	**.8478±.0031**
LDLogitBoost	.0900±.0038	.1585±.0032	.1552±.0031	.0875±.0034	.9767±.0009	.8448±.0031
LDSVR	.0918±.0047	.1583±.0035	.1559±.0035	.0884±.0042	.9765±.0011	.8440±.0035
BFGS-LDL	.0989±.0040	.1666±.0036	.1639±.0034	.0960±.0037	.9744±.0011	.8360±.0034
AA-KNN	.1274±.0069	.1917±.0045	.1899±.0047	.1246±.0062	.9664±.0018	.8101±.0047
AA-BP	.1276±.0038	.2026±.0038	.1990±.0039	.1285±.0040	.9653±.0011	.8010±.0039
IIS-LDL	.1288±.0070	.1866±.0041	.1828±.0044	.1195±.0054	.9676±.0014	.8172±.0044
CPNN	.1826±.0274	.2209±.0148	.2153±.0150	.1625±.0206	.9551±.0061	.7847±.0150

Table 3. Results on COPM. AOSO-LDLogitBoost is better.

(correct) and 1 (far from age). Not predicted images are evaluated with 1.

Table 4 reports the experimental results of LDLogitBoost, AOSO-LDLogitBoost, BFGS-LDL and LogitBoost as well as AOSO-LogitBoost. AOSO-LDLogitBoost and AOSO-LogitBoost use maximum 5000 steps with $v = 0.05$, while LDLogitBoost uses maximum 800 steps with $v_t = 0.1$. For the LDL method, BFGS uses maximum 200 steps. As can be seen, AOSO-LDLogitBoost and LDLogitBoost perform better than all of other algorithms. It implies LDLogitBoost and AOSO-LDLogitBoost can make full use of the standard deviation among the labels which can lead to a better performances than traditional LogitBoost and AOSO-LogitBoost algorithms.

Method	MAE ↓	ϵ-Error ↓
AOSO-LDLogitBoost	**7.2949**	**0.5483**
LDLogitBoost	7.3449	0.5507
BFGS-LDL	7.4243	0.5518
AOSO-LogitBoost	8.0361	0.5758
LogitBoost	9.0458	0.6044

Table 4. Results on Charlearn Age Estimation. AOSO-LDLogitBoost is better.

4. Conclusion

In this paper, we propose a LDL model family which extends the traditional maximum entropy model of LDL as this special case. Logistic Boosting Regression can be used to learn this general model which uses the coordinate descent optimization method over the function space based on the second derivative Hessian matrix. The base (weak) learners can be chosen as weighted regression tree and vector tree, which constitute two algorithms named LDLogitBoost and AOSO-LDLogitBoost, respectively. On the one hand, experiments on expression recognition and crowd opinion prediction on movies show that LDLogitBoost and AOSO-LDLogitBoost are more effective than traditional LDL algorithms. On the other hand, experiment on apparent age estimation shows that LDLogitBoost and AOSO-LDLogitBoost can lead to a better performances than traditional LogitBoost and AOSO-LogitBoost algorithms by making full use of the prior knowledge such as the standard deviation among the labels.

References

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[2] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[3] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[4] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. *Machine Learning*, 48(1–3):253–285, 2002.

[5] S. Escalera, J. Fabian, P. Pardo, X. Baró, J. Gonzàlez, H. J. Escalante, and I. Guyon. ChaLearn 2015 apparent age and cultural event recognition: Datasets and results. In *IEEE Conference on Computer Vision, ChaLearn Looking at People workshop*, 2015.

[6] J. Friedman, T. Hastie, R. Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[7] X. Geng. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering*, 2016, in press.

[8] X. Geng and P. Hou. Pre-release prediction of crowd opinion on movies by label distribution learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3511–3517, 2015.

[9] X. Geng and R. Ji. Label distribution learning. In *Proceedings of the 13th IEEE International Conference on Data Mining Workshops*, pages 377–383, 2013.

[10] X. Geng, Q. Wang, and Y. Xia. Facial age estimation by adaptive label distribution learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2234–2240, 2007.

[11] X. Geng and Y. Xia. Head pose estimation based on multivariate label distribution. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1837–1842, 2014.

[12] X. Geng, C. Yin, and Z.-H. Zhou. Facial age estimation by learning from label distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(10):2401–2412, 2013.

[13] X. Geng, Z.-H. Zhou, and K. Smith-Miles. Automatic age estimation based on facial aging patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2234–2240, 2007.

[14] X. Geng, Z.-H. Zhou, Y. Zhang, G. Li, and H. Dai. Learning from facial aging patterns for automatic age estimation. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 307–316. ACM, 2006.

[15] G. Guo, G. Mu, Y. Fu, and T. S. Huang. Human age estimation using bio-inspired features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 112–119. IEEE, 2009.

[16] P. Li. Abc-boost: Adaptive base class boost for multi-class classification. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 625–632. ACM, 2009.

[17] P. Li. Robust logitboost and adaptive base class (abc) logitboost. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

[18] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool. Face detection without bells and whistles. In *European Conference on Computer Vision*, pages 720–735. 2014.

[19] R. E. Schapire. A brief introduction to boosting. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 99, pages 1401–1406, 1999.

[20] P. Sun, M. D. Reid, and J. Zhou. An improved multiclass logitboost using adaptive-one-vs-one. *Machine Learning*, 97(3):295–326, 2014.

[21] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3476–3483. IEEE, 2013.

[22] X. Yang, B.-B. Gao, C. Xing, Z.-W. Huo, X.-S. Wei, Y. Zhou, J. Wu, and X. Geng. Deep label distribution learning for apparent age estimation. In *Proceedings of ChaLearn LAP 2015 workshop at ICCV*, 2015.

[23] Z. Zhang, M. Wang, and X. Geng. Crowd counting in public video surveillance by label distribution learning. *Neurocomputing*, 166:151163, 2015.

[24] Y. Zhou, H. Xue, and X. Geng. Emotion distribution recognition from facial expressions. In *Proceedings of the 23rd ACM International Conference on Multimedia*, 2015.