Hierarchical Classification Based on Label Distribution Learning

Changdong Xu and **Xin Geng***
MOE Key Laboratory of Computer Network and Information Integration,
School of Computer Science and Engineering,
Southeast University, Nanjing 210096, China
{changdongxu, xgeng}@seu.edu.cn

Abstract

Hierarchical classification is a challenging problem where the class labels are organized in a predefined hierarchy. One primary challenge in hierarchical classification is the small training set issue of the local module. The local classifiers in the previous hierarchical classification approaches are prone to over-fitting, which becomes a major bottleneck of hierarchical classification. Fortunately, the labels in the local module are correlated, and the siblings of the true label can provide additional supervision information for the instance. This paper proposes a novel method to deal with the small training set issue. The key idea of the method is to represent the correlation among the labels by the label distribution. It generates a label distribution that contains the supervision information of each label for the given instance, and then learns a mapping from the instance to the label distribution. Experimental results on several hierarchical classification datasets show that our method significantly outperforms other state-of-the-art hierarchical classification approaches.

Introduction

In many classification problems, the class labels are organized in a predefined hierarchical structure. For example, the documents are organized with the topic hierarchies in some large-scale text datasets, such as Wikipedia, DMOZ, and Yahoo! Directory; the music is often organized in an audio taxonomy; a gene is arranged by its functions with the tree or the graph structure (Vens et al. 2008). These problems are defined as the hierarchical classification problems (Jr. and Freitas 2011). In this paper, we deal with the hierarchical classification problems where each instance is assigned only one label.

According to a survey on hierarchical classification (Jr. and Freitas 2011), the hierarchical classification approaches can be roughly grouped into three families, i.e., the flat classification approach, the local classifier approach, and the global classifier approach. The flat classification approach is the simplest one to deal with the hierarchical classification problems, which completely discards the hierarchical information. This approach is equivalent to those traditional classification methods. The local classifier approach trains a classifier for each local module. Early research uses the taxonomy to train independent classifiers at each node, and makes a prediction by the top-down manner (Koller and Sahami 1997). Since the local classifier approach often encounters the error propagation issue, some methods have been proposed to solve it. A method changes the training distribution, and utilizes the output of the lower-level classifier to improve the performance of the top-level classifier (Bennett and Nguyen 2009). Then a novel framework is proposed to make the Bayes-optimal prediction by minimizing the designed risks (Bi and Kwok 2015). Correspondingly, an approach uses a surrogate loss to replace the tree-distance loss, and constructs a convex optimization problem which can be solved using binary SVM (Ramaswamy, Tewari, and Agarwal 2015). Another method exploits the correlation between the label and its ancestors in the hierarchy by considering the label of the parent node as an additional attribute. Moreover, the method calculates a score for each path, and makes the final prediction according to the scores (Ramírez-Corona, Sucar, and Morales 2016). Besides, a local hierarchical ensemble framework is proposed to model the relationship between the global prediction and the local prediction as a regression problem (Zhang, Shah, and Kakadiaris 2017). Different from the local classifier approach, the global classifier approach considers the hierarchy as a whole, and learns a single classification model on the training set. Several large-margin methods are adapted to deal with the hierarchical classification problems by defining the different mis-classification penalization in the hierarchy (Cai and Hofmann 2004; Dekel, Keshet, and Singer 2004). Correspondingly, a novel large-margin method is designed to utilize the hierarchical information by enforcing the orthogonality between the parent node and its children (Xiao, Zhou, and Wu 2011). Nevertheless, these methods are quite time-consuming, which can not solve the problems with massive labels. In order to deal with the large-scale hierarchical classification problems, one possible solution is to encourage the model parameters among the nearby labels in the hierarchy to be similar by the recursive regularization (Gopal and Yang 2013). The other possible solution is to transform the hierarchical problem to the cost-sensitive classification problem by defining the mis-classification cost based on the hierarchy (Charuvaka and Rangwala 2015). Recently, a deep neural network is designed to deal with the hierarchical clas-

*Corresponding author.

sification problems by simultaneously optimizing the local and global loss function for the local and global hierarchical information. (Wehrmann, Cerri, and Barros 2018).

One primary challenge in hierarchical classification is the small training set issue. Take the LSHTC-small dataset as an example, there are more than one thousand labels, but the average number of the training samples per label is less than 6. Thus, for the local classifier approaches, there are only a few training samples for the bottom module. Previous local classifier approaches focus attention on the strategy of the prediction, but ignore the significance of the local classifier in the training phase. They consider the labels in the local module as independent, and train local multi-class classifiers. Nevertheless, since the training set for the local module is small, the multi-class classifiers are prone to over-fitting, which becomes a bottleneck of hierarchical classification.

Different from the previous local classifier approaches, we believe that the labels in the local module are correlated, and the degree of the correlation is variant in different local modules. For example, the labels with common ancestors are correlated, and the correlation among the labels will be enhanced with the increase of common ancestors. If we can make use of the label correlation in the local module, the influence of the small training set issue will be relieved greatly because the instances with correlated labels can also contribute to the training of the current class. In order to achieve this, we adopt a recently proposed machine learning paradigm called Label Distribution Learning (LDL) (Geng 2016). The label distribution covers a certain number of labels, representing the degree to which the corresponding label describes the instance. The description degrees of all the labels sum up to 1. LDL has been successfully applied to many real-world problems, such as facial age estimation (Geng, Yin, and Zhou 2013), head-pose estimation (Geng and Xia 2014), pre-release prediction of crowd opinion on movies (Geng and Hou 2015), crowd counting in public video surveillance (Zhang, Wang, and Geng 2015), ordinal zero-shot learning (Huo and Geng 2017), facial beauty prediction (Ren and Geng 2017), deep learning (Gao et al. 2017), etc. In this paper, we use the label distribution to explicitly represent the correlation between the true label and its siblings. It means that not only the true label but also its siblings can describe the instance. Higher description degree indicates that the corresponding label is more correlated to the instance. Different from the multi-class classifier in the local module, where the siblings of the true label offer no supervision information, in the proposed method, the supervision information of the siblings can be extended to the true label. By LDL, the proposed approach allows an instance to be correlated with multiple labels with different importance, which can take better advantage of the label correlation.

The main contribution of this paper is to find a new solution to solve the small training set issue of the local module in hierarchical classification. Compared with the previous local classifier approaches which are committed to the strategy of the prediction, we focus on the label correlation in the training phase. To the best of our knowledge, this is the first attempt to explicitly represent the label correlation with the label distribution in the local module. We conduct the experiments on several hierarchical classification datasets, which demonstrate the effectiveness of our proposed method.

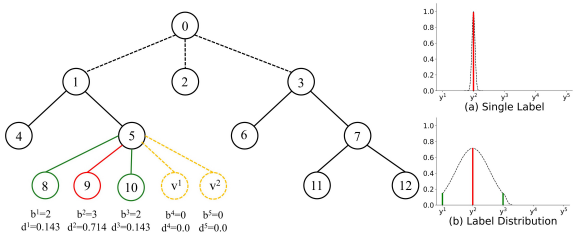

Figure 1: An example of the label distribution representation in the local module M_5. The label space is $\{y^1 = 8, y^2 = 9, y^3 = 10, y^4 = v^1, y^5 = v^2\}$. For an instance $\boldsymbol{x}$, the true label is y^2. The graph (a) shows the single label annotation for $\boldsymbol{x}$, and the graph (b) shows the label distribution annotation for $\boldsymbol{x}$.

The rest of this paper is organized as follows. Firstly, we introduce the definition of label distribution learning, the transformation method from the single label to the label distribution for a given instance, the algorithm of label distribution learning, and the strategy of the prediction. Secondly, we report the results of hierarchical classification experiments. Finally, we draw a conclusion of our paper.

Proposed Method

We propose a local classifier approach to deal with the small training set issue of the local module. For each local module, we use a label distribution instead of a single label to represent a certain instance. The label distribution covers the whole possible labels, and assigns a real number to each label. It means that the true label as well as its siblings can provide the supervision information to the instance.

In this paper, we assume that the labels are organized with the tree structure. Let the hierarchy be a tree defined as T. The nodes in T are indexed from 0 (for the root), $1, 2, ..., |T|$. We also use T to express the set of all nodes. Let $L \subset T$ be the set of the leaf nodes, and $L^C = T \setminus L$ be the set of the internal nodes in T. For a node t, we denote its parent by $pa(t)$, its set of ancestors by $anc(t)$, its set of siblings by $sib(t)$, its set of children by $child(t)$, and its set of path nodes by $path(t) = \{a|a \in anc(t)\}\setminus\{0\}\cup\{t\}$, respectively.

Training

Formally, for an internal node $t \in L^C$, t and $child(t)$ form a local module which is called M_t for short. We define the label space for M_t as $Y_t = child(t) = \{y^1, y^2, ..., y^K\}$, where y^j represents the j-th label, and K is the number of the labels. Given the training samples $S_t = \{(\boldsymbol{x}_i, y_i)|1 \leq i \leq n_t\}$ for M_t, where $\boldsymbol{x}_i$ is the i-th instance, y_i is the true label for $\boldsymbol{x}_i$, and n_t is the number of the training samples, we need to train the probability estimator $p(k|t, \boldsymbol{x})$, where $k \in child(t)$.

For M_t, where $t \neq 0$, when an instance $\boldsymbol{x}$ does not belong to M_t, we hope M_t can still output some probability for $\boldsymbol{x}$. Therefore, we introduce a virtual node set $V_t = \{v^1, v^2, ..., v^m\}$ into M_t, where m is the path length

of node t. The instance $\boldsymbol{x}$ is marked as $v \in V_t$ when $\boldsymbol{x}$ falls outside M_t. Then we extend the children set of t as $child(t) = child(t) \cup V_t$. The reason that we use a virtual node set instead of one virtual node is that, as shown below, the training set of the virtual node set is sampled from different levels in the hierarchy. We think the samples in different levels are variant, which means that the samples in different levels belong to different classes. Thus, one virtual node can not represent all the samples, and we decide to add more than one virtual node.

We need to collect the training samples for V_t. In order to make the training samples of V_t cover all the samples as much as possible, we propose a stratified sampling method. Specifically, we build a node set $G_t = \{g | g \in sib(a), a \in path(t)\}$, and then take the training set of V_t from the samples belonging to G_t. In Figure 1, for instance, the local module of node 5 is M_5. The label space of M_5 is $\{y^1 = 8, y^2 = 9, y^3 = 10, y^4 = v^1, y^5 = v^2\}$, where v^1 and v^2 are the virtual nodes. The training set of v^1 will be collected from the samples of the node set $\{2, 3\}$, since node 2 and node 3 are the siblings of node 1. The training set of v^2 will be collected from the samples of node 4 which is the sibling of node 5.

Due to the small training set issue of the local module, the traditional local classifier used in the previous studies suffers over-fitting easily. One remarkable observation is that the labels in the local module are correlated, which means that the siblings of the true label can provide additional supervision information to the instance. The above observation coincides with the mechanism of label distribution learning, and we can use the label distribution to represent the description degree of each label for the given instance.

Given the training set $S_t = \{(\boldsymbol{x}_i, y_i) | 1 \leq i \leq n_t\}$ for M_t, where $t \neq 0$, we define the label distribution for $\boldsymbol{x}_i$ as $\boldsymbol{d}_i = (d_i^1, d_i^2, ..., d_i^K)$, where d_i^j is the j-th element of $\boldsymbol{d}_i$ corresponding to y^j, and K is the number of the labels. The label distribution $\boldsymbol{d}_i$ satisfies $d_i^j \in [0, 1]$ and $\sum_{j=1}^{K} d_i^j = 1$. The value of d_i^j expresses the degree to which the label y^j describes $\boldsymbol{x}_i$, and thus is called the description degree of y^j to $\boldsymbol{x}_i$. The description degree corresponding to the true label has the highest value.

Since the ground truth label distribution is not available in the training set of the local module, we need to transform the true label to the label distribution. There are many methods to make the transformation, and in this paper, we achieve it by exploiting the knowledge of the common nodes with the true path. More concretely, let $\boldsymbol{b} = (b^1, b^2, ..., b^K)$ be the Number of the Common Nodes with the True Path (NCNTP), where b^j is the NCNTP for y^j. It is obvious that the correlation among the labels is stronger with larger NCNTP. For instance, in Figure 1, given an instance $\boldsymbol{x}$, if the true label of $\boldsymbol{x}$ is leaf node 9, then the true path is $\{1, 5, 9\}$. The correlation among $\{8, 9, 10\}$ is stronger than $\{4, 5\}$, and the NCNTP of $\{8, 9, 10\}$ is larger than $\{4, 5\}$. So, we can use the NCNTP to represent the correlation among the labels as

$$b^j = \begin{cases} |path(y) \cap path(y^j)| & y^j \notin V_t \\ 0 & y^j \in V_t \end{cases}, \tag{1}$$

Algorithm 1 Training

Input:
S: the training set $\{(\boldsymbol{x}_i, y_i) | 1 \leq i \leq n\}$;
T: the tree hierarchy;
α: the degree parameter;
λ: the penalty parameter.
Output:
$\boldsymbol{\phi}$: the set of the local model for each local module.

```
Set φ to be empty;
Set the queue Q to be empty;
Put node 0 to Q;
while Q is not empty do
    Get the first element t in Q, and delete it from Q;
    if t ∈ L^C then
        Put the children of node t to Q;
        if t ≠ 0 then
            Extend the children set of node t as child(t) =
            child(t) ∪ V_t, where V_t is the virtual node set;
        end if
        Collect the training set S_t = {(x_i, y_i) | 1 ≤ i ≤ n_t}
        for M_t, where y_i ∈ Y_t = child(t);
        if t ≠ 0 then
            Transform S_t to S_t' = {(x_i, d_i) | 1 ≤ i ≤ n_t} by
            Eq. (3);
            Train a label distribution model on S_t' by Eq. (7);
        else
            Train a multi-class classifier on S_t;
        end if
        Add the local model to φ;
    end if
end while
return the set of local model φ;
```

where y is the true label, and V_t is the virtual node set. Although the node in V_t is the sibling of y, it is not correlated with y, thus we set the NCNTP of the node in V_t to 0. Then, $\boldsymbol{b}$ is normalized as

$$\boldsymbol{r} = \frac{1}{Z}\boldsymbol{b}, \tag{2}$$

where $Z = \sum_{j=1}^{K} b^j$ is a normalization factor. We can calculate the label distribution by

$$\boldsymbol{d} = (1 - \alpha)\boldsymbol{h} + \alpha \boldsymbol{r}, \tag{3}$$

where $\boldsymbol{h} = (h^1, h^2, ..., h^K)$, and $h^j \in \{0, 1\}$ indicates whether the instance $\boldsymbol{x}$ has the label y^j. Since $\boldsymbol{x}$ has only one label, $\boldsymbol{h}$ satisfies $\sum_{j=1}^{K} h^j = 1$. The parameter $\alpha \in [0, 1]$ controls the degree of the correlation among the labels. When α is set to 0, the model degenerates into the standard multi-class classifier. When α is set to 1, we get a coarse label distribution which may not match the problem. Thus, we use α as a trade-off to get a suitable label distribution.

In Figure 1, take node 5 as an example, the local module M_5 consists of node 5 and its children. The label space in M_5 is $Y_5 = \{y^1 = 8, y^2 = 9, y^3 = 10, y^4 = v^1, y^5 = v^2\}$, where v^1 and v^2 are the virtual nodes. If the true label is y^2 for $\boldsymbol{x}$, then $\boldsymbol{b} = (2, 3, 2, 0, 0)$, $\boldsymbol{r} = (\frac{2}{7}, \frac{3}{7}, \frac{2}{7}, 0, 0)$. Since y^4 and y^5 are not correlated with y^2, we set b^4 and b^5 to

0. If we set $\alpha = 0.5$, we can get the label distribution for $\boldsymbol{x}$ as $\boldsymbol{d} = (0.143, 0.714, 0.143, 0, 0)$. Originally, only the true label y^2 provides the supervision information to $\boldsymbol{x}$. After the transformation, the true label y^2 as well as its siblings $\{y^1, y^3\}$ can provide the supervision information to $\boldsymbol{x}$. Moreover, the true label y^2 has the highest description degree. Although y^4 and y^5 are the siblings of y^2, they are virtual nodes and not correlated with y^2. Thus, their description degrees are set to 0.

After the transformation, the training set becomes $S_t^{'} = \{(\boldsymbol{x}_i, \boldsymbol{d}_i)|1 \leq i \leq n_t\}$, where $\boldsymbol{d}_i$ is the label distribution for $\boldsymbol{x}_i$, and d_i^j is the j-th element of $\boldsymbol{d}_i$. The goal is to find the parameter $\boldsymbol{\theta}$ in a conditional mass function $p(\boldsymbol{y}|pa(y_i), \boldsymbol{x}_i; \boldsymbol{\theta})$ that can generate a label distribution similar to $\boldsymbol{d}_i$. There are many criteria to measure the similarity between two distributions, e.g., the discrete Jeffrey's divergence between two distributions $\boldsymbol{P}$ and $\boldsymbol{Q}$ is defined by

$$D_J(\boldsymbol{P}||\boldsymbol{Q}) = \sum_i (P_i - Q_i) \ln \frac{P_i}{Q_i}, \tag{4}$$

where P_i and Q_i are the i-th element of $\boldsymbol{P}$ and $\boldsymbol{Q}$, respectively. Similar to the work of (Geng, Yin, and Zhou 2013), we assume the function to be a maximum entropy model, i.e.,

$$p(y^j|pa(y_i), \boldsymbol{x}_i; \boldsymbol{\theta}) = \frac{1}{\Gamma_i} \exp(\sum_r \theta_{j,r} x_i^r), \tag{5}$$

where $\Gamma_i = \sum_k \exp(\sum_r \theta_{k,r} x_i^r)$ is the normalization factor, x_i^r is the r-th feature of $\boldsymbol{x}_i$, and $\theta_{j,r}$ is an element in $\boldsymbol{\theta}$ corresponding to the label y^j and the r-th feature.

Then the best parameter $\boldsymbol{\theta}^*$ is determined by

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_i D_J(\boldsymbol{d}_i||p(\boldsymbol{y}|pa(y_i), \boldsymbol{x}_i; \boldsymbol{\theta})) + \frac{\lambda}{2}||\boldsymbol{\theta}||^2. \tag{6}$$

The first term is Jeffrey's divergence, where $\boldsymbol{d}_i$ is the ground truth label distribution for $\boldsymbol{x}_i$, and $p(\boldsymbol{y}|pa(y_i), \boldsymbol{x}_i; \boldsymbol{\theta})$ is the predicted label distribution. The second term is a regularization term, and λ is a parameter that controls the trade-off between the training loss and the complexity of the model.

Substituting Eq. (5) to Eq. (6) yields the target function

$$\begin{aligned} T(\boldsymbol{\theta}) = & \sum_i \ln\Gamma_i - \sum_i \sum_j (d_i^j \sum_r \theta_{j,r} x_i^r) + \\ & \sum_i \sum_j \frac{1}{\Gamma_i} \exp(\sum_r \theta_{j,r} x_i^r)(\sum_r \theta_{j,r} x_i^r \\ & - \ln\Gamma_i - \ln d_i^j) + \frac{\lambda}{2}||\boldsymbol{\theta}||^2. \end{aligned} \tag{7}$$

The problem can be solved by the limited-memory quasi-Newton method L-BFGS (Liu and Nocedal 1989). After obtaining $\boldsymbol{\theta}^*$, given an instance, we can get its predicted label distribution. The whole training process is summarized in Algorithm 1.

Prediction

For the local classifier approach, there are many ways to make the prediction, such as the top-down strategy and the strategy based on Bayesian decision theory (Bi and Kwok 2015). Our method predicts the labels by maximizing the path score which takes the influence of the path length into account. The whole prediction process is summarized in Algorithm 2.

For an internal node $t \in L^C$, t and $child(t)$ form a local module M_t. If $t \neq 0$, we extend $child(t)$ as $child(t) = child(t) \cup V_t$, where V_t is the virtual node set. The label space of M_t is $Y_t = child(t) = \{y^1, y^2, ..., y^K\}$, where y^j represents the j-th label, and K is the number of the labels. Given a test instance $\boldsymbol{x}$, we need to calculate the probability $p(y^j|t, \boldsymbol{x})$, where $y^j \in Y_t$. For node 0, we can get the probability of its children by the trained local classifier. For an internal node $t \neq 0$, we can get the predicted label distribu-

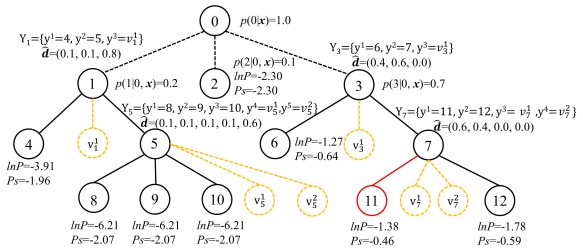

Figure 2: An example of the prediction for a test instance $\boldsymbol{x}$, where Y_t is the label space for the local module M_t, V_t is the virtual node set for M_t, $\hat{\boldsymbol{d}}$ is the predicted label distribution, Ps and lnP are the path score and the logarithmic posterior probability for the leaf node. The leaf node 11 with the maximum path score is chosen to be the predicted label.

Algorithm 2 Prediction

Input:
$\boldsymbol{x}$: the test instance;
$\boldsymbol{\phi}$: the set of the local model for each local module.
Output:
$\hat{y}$: the predicted label for $\boldsymbol{x}$.

Set the queue Q to be empty;
Put node 0 to Q;
while Q is not empty **do**
  Get the first element t in Q, and delete it from Q;
  if $t \in L^C$ **then**
    Put the children of node t to Q;
    Get the local model for t from $\boldsymbol{\phi}$;
    if $t \neq 0$ **then**
      Calculate the label distribution as the probability of its children;
    else
      Calculate the probability of its children;
    end if
  else
    Calculate the path score of t by Eq. (9);
  end if
end while
Predict the label by Eq. (10);
return the predicted label $\hat{y}$;

tion $\hat{\boldsymbol{d}} = (\hat{d^1}, \hat{d^2}, ..., \hat{d^K})$. It is reasonable that we use $\hat{\boldsymbol{d}}$ to represent the probability of the labels. By this way, we use $\hat{d^j}$ corresponding to y^j to represent $p(y^j|t, \boldsymbol{x})$.

We calculate the logarithmic posterior probability of the leaf node $l \in L$ for the test instance $\boldsymbol{x}$ as

$$\ln(p(l|\boldsymbol{x})) = \sum_{t \in path(l)} \ln(p(t|pa(t), \boldsymbol{x})). \tag{8}$$

The path length has an important influence on the prediction. For example, the posterior probability of the leaf node with long path may be small, even though the probability of each node in the path is large. In order to eliminate the influence of the path length, we define the path score as

$$Ps(l|\boldsymbol{x}) = \frac{\ln(p(l|\boldsymbol{x}))}{|path(l)|}. \tag{9}$$

We finally choose the leaf node with the maximum path score as the predicted label, i.e.,

$$\hat{y} = \arg\max_{l \in L} Ps(l|\boldsymbol{x}). \tag{10}$$

In Figure 2, for a test instance $\boldsymbol{x}$, the local module M_0 outputs the probability for $\boldsymbol{x}$, and other modules output the label distribution for $\boldsymbol{x}$. We use the label distribution to represent the probability in the absence of the doubt. The path score for each leaf node is calculated by Eq. (9). Considering the influence of the path length, we finally choose node 11 with the maximum path score as the predicted label, rather than node 6 with the maximum posterior probability.

Experiments

Datasets

We conduct our experiments on several hierarchical classification datasets, including one image dataset and four document datatsets, all of which have one label per example. The basic statistics of the datasets are listed in Table 1.

- **CLEF** (Dimitrovski et al. 2011) is an image dataset which consists of medical X-ray images.
- **IPC** [1] is a document dataset which is a collection of patents arranged with the International Patent Classification Hierarchy.
- **LSHTC-small, DMOZ-2010, and DMOZ-2012** [2] (Partalas et al. 2015) are a number of document datasets released from the LSHTC (Large-Scale Hierarchical Text Classification) challenges 2010 and 2012.

Algorithms for Comparison

We compare our proposed method with other state-of-the-art hierarchical classification algorithms, including the flat classification approach, the local classifier approach, and the global classifier approach.

- **Flat-SVM** is the one-versus-rest SVM which discards the hierarchical information. It is considered as the conventional flat classifier.
- **TD** (Dumais and Chen 2000) is a local classifier approach. In the training phase, the approach trains a series of local classifiers. In the test phase, the approach makes the prediction by the top-down strategy.
- **MAS** (Bi and Kwok 2014) is a local classifier approach. In the training phase, the approach trains a series of local classifiers that can output the probabilities. In the test phase, the approach makes the prediction by minimizing the designed loss.
- **HierCost** (Charuvaka and Rangwala 2015) is a global classifier approach. This work uses a cost-sensitive classification method to deal with the hierarchical classification problem by defining the mis-classification cost based on the hierarchy.

We use cross-validation to select the optimal parameters on the datasets. Specifically, the penalty parameters of all the algorithms are chosen with a range from 10^{-3} to 10^3. The kernel of Flat-SVM is linear. The local classifier of TD is SVM whose kernel is linear. We use the logistic regression which can output the probability of the class as the local classifier of MAS. The prediction strategy of MAS is to maximize the posterior probability. We set the cost type of HierCost to exponentiated tree distance and imbalance in the same way as (Charuvaka and Rangwala 2015). For our method, α is decided in a range from 0 to 1.

Evaluation Metrics

We use four metrics to measure the performance of all the approaches, including the standard classification metrics and the hierarchical classification metrics.

- **Micro-$\mathbf{F}_1$ and Macro-$\mathbf{F}_1$** (Gopal and Yang 2013) are standard classification metrics in the conventional classification problems.
- **HF**(Jr. and Freitas 2011) is Hierarchical F-measure, which is commonly used in the hierarchical classification problems. It is an extension of the standard F-measure.
- **TE** (Dekel, Keshet, and Singer 2004) is Tree-induced Error to measure the average distance in the tree between the true and the predicted labels.

For TE, the smaller values are better; while for the other metrics, the larger values are better.

Results

We set up two groups of experiments. The first experiment is designed to compare the effectiveness of the algorithms, and the second experiment explores the trend of the performance variation with the reduction of the training samples. The true test labels of DMOZ-2010 and DMOZ-2012 are not available, and the results can only be evaluated through an online evaluation system where some metrics are not supported. For DMOZ-2010, the HF results are not available, and for DMOZ-2012, the TE results are not available.

Table 2 shows the results of the first experiment. The results of HierCost on DMOZ-2010 and DMOZ-2012 are reported from (Charuvaka and Rangwala 2015). The results of Flat-SVM on DMOZ-2012 are not reported due to the

[1] http://www.wipo.int/classifications/ipc/en/support/

[2] http://lshtc.iit.demokritos.gr/

Dataset	Train	Test	Nodes	Leaves	Depth	Features
CLEF	10,000	1,006	97	63	3	80
IPC	46,324	28,926	553	451	3	310,586
LSHTC-small	6,323	1,858	2,388	1,139	5	51,033
DMOZ-2010	128,710	34,880	17,222	12,294	5	381,580
DMOZ-2012	383,408	103,435	13,963	11,947	5	348,548

Table 1: Dataset Statistics

		Micro-F_1(↑)	Macro-F_1(↑)	HF(↑)	TE(↓)
CLEF	Flat-SVM	79.52	**54.41**	83.10	1.01
	TD	73.75	35.78	77.63	1.34
	MAS	75.34	39.05	78.69	1.28
	HierCost	**80.12**	53.07	**83.93**	**0.96**
	Ours	77.24	46.77	80.96	1.14
IPC	Flat-SVM	**54.04**	46.13	65.67	2.04
	TD	50.59	43.63	63.55	2.17
	MAS	52.53	45.20	65.03	2.08
	HierCost	53.89	**47.69**	65.75	2.04
	Ours	53.22	46.46	**65.97**	**2.03**
LSHTC-small	Flat-SVM	49.14	36.21	62.85	3.55
	TD	46.23	33.43	63.36	3.52
	MAS	46.34	32.47	63.85	3.46
	HierCost	49.46	36.02	63.90	3.45
	Ours	**50.51**	**37.02**	**67.74**	**3.07**
DMOZ-2010	Flat-SVM	44.64	31.34	N/A	3.55
	TD	42.39	29.67	N/A	3.46
	MAS	43.25	30.13	N/A	3.43
	HierCost	**45.87**	32.41	N/A	3.32
	Ours	45.24	**33.05**	N/A	**3.15**
DMOZ-2012	Flat-SVM	N/A	N/A	N/A	N/A
	TD	54.31	**33.47**	75.11	N/A
	MAS	54.58	28.67	75.17	N/A
	HierCost	53.36	28.47	73.79	N/A
	Ours	**55.30**	32.41	**75.97**	N/A

Table 2: Predictive performance of each comparing algorithm on datasets.

memory limitation. In general, our model is comparable, or performs better in terms of the standard classification metrics (i.e., Micro-F_1 and Macro-F_1), and outperforms other methods significantly in terms of the hierarchical classification metrics (i.e., HF and TE). As showed in (Jr. and Freitas 2011), the standard classification metrics like Micro-F_1 and Macro-F_1 are not ideal, because the errors at different levels of the class hierarchy should not be penalized in the same way. Thus, we prefer to believe that the hierarchical classification metrics can evaluate the models better, and the above results can prove the advantage of our method. In detail, on the CLEF and IPC datasets, the performance of our method is not as good as Flat-SVM and HierCost in terms of some metrics. The main reason might be that the training samples are sufficient, and the problem is relatively simple. Our method is designed to solve the small training set issue of the local module, in the case of having sufficient training samples, the performance of our method may not be outstanding. On the other datasets, which consist of massive labels but small training set, our method is superior to other methods in most cases. These results coincide with the motivation of our method.

In the first experiment, our method has the trend to perform well on the dataset with relatively small training set. To further demonstrate this, we design the second experiment that gradually decrease the number of the training samples. Specifically, we select randomly a part of all the training samples as the training set, while keeping the test set unchanged. The process is repeated for five times, and the average performance of all the algorithms is recorded. For CLEF, we choose $\{1/10, 1/20, 1/40\}$ of all the samples as the training set; for IPC, we choose $\{1/5, 1/10, 1/20\}$ of all the samples as the training set; for LSHTC-small, we choose $\{3/4, 2/4, 1/4\}$ of all the samples as the training set, respectively. Since some metrics are not available, we do not conduct the second experiment on DMOZ-2010 and DMOZ-

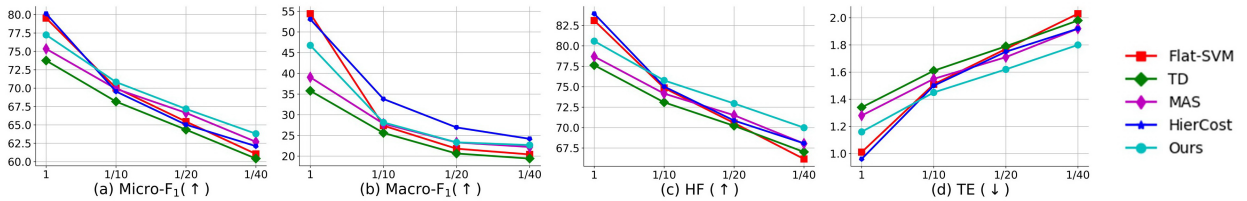

Figure 3: Results of the data-sparsity experiment on the CLEF dataset. The horizontal axis is gradually decrease number of the training samples, vertical axis is the results of some metrics.

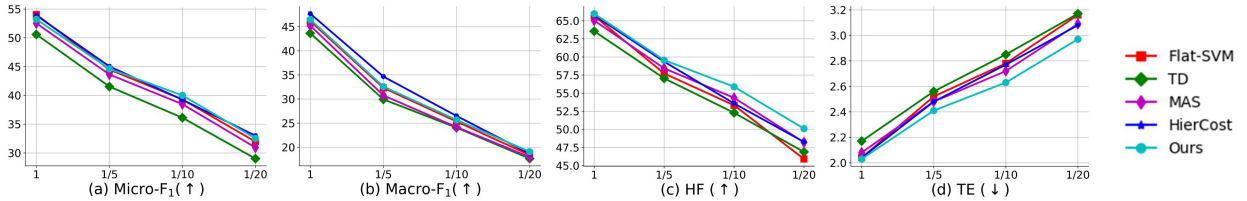

Figure 4: Results of the data-sparsity experiment on the IPC dataset. The horizontal axis is gradually decrease number of the training samples, vertical axis is the results of some metrics.

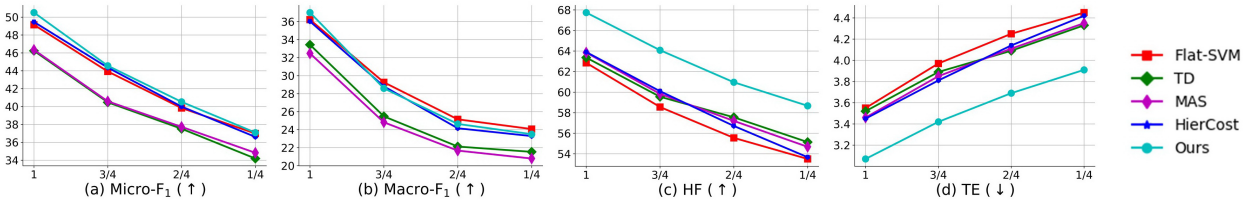

Figure 5: Results of the data-sparsity experiment on the LSHTC-small dataset. The horizontal axis is gradually decrease number of the training samples, vertical axis is the results of some metrics.

2012. The results of the second experiment are shown in Figure 3, 4, and 5. In general, our method performs the best in most cases compared with other hierarchical classification methods, but the advantages of our method in terms of the standard classification metrics (i.e., Micro-F_1 and Macro-F_1) are not outstanding. With the decrease of the training samples, the model is more likely to make a wrong prediction. When the mis-classification occurs, our model is prone to predict a label which is correlated with the true label. However, the standard classification metrics ignore the hierarchical information completely, and they consider the cost of the mis-classification equally, no matter the predicted label is correlated with the true label or not. Thus, the results of our method in terms of the standard classification metrics may no longer maintain their advantages. In detail, on the CLEF dataset, our method performs not as good as HierCost and Flat-SVM at beginning, but with the decrease number of the training samples, our method outperforms other methods gradually. Flat-SVM performs well with the full training samples, but as the training set becomes smaller, its performance in terms of the HF and TE metrics deteriorates quickly. The main reason might be that Flat-SVM discards the hierarchical information completely, and it is not friendly to the hierarchical classification metrics when the training set is small. The performances in terms of the HF and TE metrics between HierCost and MAS become similar when the training samples are reduced. TD performs the worst in most cases. The trend on the IPC dataset is similar to the CLEF dataset. On the LSHTC-small dataset, the performances among Flat-SVM, HierCost and our method are similar in terms of the Micro-F_1 and Macro-F_1 metrics, but in terms of the HF and TE metrics, our method outperforms other methods significantly. These results prove that our method can deal with the small training set issue well.

Conclusion

In this paper, we propose a novel hierarchical classification method based on label distribution learning. The motivation of our method is to solve the small training set issue of the local module by utilizing the correlation between the true label and its siblings. Toward this, we propose to generate the label distribution that contains supervision information of each label for the instance in the local module. After transforming the true label to the label distribution, we conduct a mapping from the instance to its label distribution which is optimized by L-BFGS algorithm. The results on several hierarchical classification datasets prove that our method performs significantly better than other hierarchical classification approaches, and can deal with the small training set issue well.

Acknowledgments

This research was supported by the National Key Research & Development Plan of China (No. 2017YFB1002801), the National Science Foundation of China (61622203), the Jiangsu Natural Science Funds for Distinguished Young Scholar (BK20140022), the Collaborative Innovation Center of Novel Software Technology and Industrialization, and the Collaborative Innovation Center of Wireless Communications Technology.

References

Bennett, P. N., and Nguyen, N. 2009. Refined experts: improving classification in large taxonomies. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*, 11–18.

Bi, W., and Kwok, J. T. 2014. Mandatory leaf node prediction in hierarchical multilabel classification. *IEEE Transactions on Neural Networks and Learning Systems* 25(12):2275–2287.

Bi, W., and Kwok, J. T. 2015. Bayes-optimal hierarchical multilabel classification. *IEEE Transactions on Knowledge and Data Engineering* 27(11):2907–2918.

Cai, L., and Hofmann, T. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of ACM International Conference on Information and Knowledge Management*, 78–87.

Charuvaka, A., and Rangwala, H. 2015. Hiercost: Improving large scale hierarchical classification with cost sensitive learning. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 675–690.

Dekel, O.; Keshet, J.; and Singer, Y. 2004. Large margin hierarchical classification. In *Proceedings of International Conference on Machine Learning*, 27–35.

Dimitrovski, I.; Kocev, D.; Loskovska, S.; and Dzeroski, S. 2011. Hierarchical annotation of medical images. *Pattern Recognition* 44(10-11):2436–2449.

Dumais, S. T., and Chen, H. 2000. Hierarchical classification of web content. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*, 256–263.

Gao, B.; Xing, C.; Xie, C.; Wu, J.; and Geng, X. 2017. Deep label distribution learning with label ambiguity. *IEEE Transactions on Image Processing* 26(6):2825–2838.

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

Geng, X., and Xia, Y. 2014. Head pose estimation based on multivariate label distribution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1837–1842.

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

Geng, X. 2016. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering* 28(7):1734–1748.

Gopal, S., and Yang, Y. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 257–265.

Huo, Z., and Geng, X. 2017. Ordinal zero-shot learning. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1916–1922.

Jr., C. N. S., and Freitas, A. A. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22(1-2):31–72.

Koller, D., and Sahami, M. 1997. Hierarchically classifying documents using very few words. In *Proceedings of International Conference on Machine Learning*, 170–178.

Liu, D. C., and Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Math. Program.* 45(1-3):503–528.

Partalas, I.; Kosmopoulos, A.; Baskiotis, N.; Artières, T.; Paliouras, G.; Gaussier, É.; Androutsopoulos, I.; Amini, M.; and Gallinari, P. 2015. LSHTC: A benchmark for large-scale text classification. *CoRR* abs/1503.08581.

Ramaswamy, H. G.; Tewari, A.; and Agarwal, S. 2015. Convex calibrated surrogates for hierarchical classification. In *Proceedings of International Conference on Machine Learning*, 1852–1860.

Ramírez-Corona, M.; Sucar, L. E.; and Morales, E. F. 2016. Hierarchical multilabel classification based on path evaluation. *International Journal of Approximate Reasoning* 68:179–193.

Ren, Y., and Geng, X. 2017. Sense beauty by label distribution learning. In *Proceedings of International Joint Conference on Artificial Intelligence*, 2648–2654.

Vens, C.; Struyf, J.; Schietgat, L.; Dzeroski, S.; and Blockeel, H. 2008. Decision trees for hierarchical multi-label classification. *Machine Learning* 73(2):185–214.

Wehrmann, J.; Cerri, R.; and Barros, R. C. 2018. Hierarchical multi-label classification networks. In *Proceedings of International Conference on Machine Learning*, 5225–5234.

Xiao, L.; Zhou, D.; and Wu, M. 2011. Hierarchical classification via orthogonal transfer. In *Proceedings of International Conference on Machine Learning*, 801–808.

Zhang, L.; Shah, S. K.; and Kakadiaris, I. A. 2017. Hierarchical multi-label classification using fully associative ensemble learning. *Pattern Recognition* 70:89–103.

Zhang, Z.; Wang, M.; and Geng, X. 2015. Crowd counting in public video surveillance by label distribution learning. *Neurocomputing* 166:151–163.