Multi-label Classification: Inconsistency, Ambiguity and Class Balanced $K$NN Classification

Hua Wang, Chris Ding and Heng Huang

Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019
huawang2007@mavs.uta.edu, {chqding.heng}@uta.edu

Abstract

Many existing researches employ one-vs-others approach to decompose a multi-label classification problem into a set of 2-class classification problems, one for each class. This approach is valid in traditional single-label classification. However, it incurs training inconsistency in multi-label classification, because a multi-label data point could belong to more than one class. In this work, we further develop classical $K$-Nearest Neighbor classifier and propose a novel Class Balanced $K$-Nearest Neighbor (B$k$NN) approach for multi-label classification by emphasizing balanced usage of data from all the classes. In addition, we also propose a Class Balanced Linear Discriminant Analysis approach to address high-dimensional multi-label input data. Promising experimental results on three broadly used multi-label data sets demonstrate the effectiveness of our approach.

Introduction

Multi-label classification frequently arises in many real life applications, such as document categorization, image annotation, etc. Different from traditional single-label classification where each object belongs to only one class, multi-label classification deals with problems where an object may belong to more than one class. Formally, for a classification task, given $K$ predefined classes and $n$ data points, each data point $x_i \in \mathbb{R}^p$ is associated with a subset of class labels represented by a binary label indicator vector $y_i \in \{0,1\}^K$, such that $y_i(k) = 1$ if $x_i$ belongs to the $k$-th class, and 0 otherwise. In single label classification, it holds that $\sum_{k=1}^{K} y_i(k) = 1$; while for multi-label classification, $\sum_{k=1}^{K} y_i(k) \geq 1$. Given a training data set $\{(x_i, y_i)\}_{i=1}^{n}$ with $l$ ($l < n$) data points, our goal is to predict labels $\{y_i\}_{i=l+1}^{n}$ for the rest $n-l$ testing data points $\{x_i\}_{i=l+1}^{n}$.

Inconsistency of One-vs-Others Method in Multi-label Multi-class Classification

Some of the most powerful classification algorithms, such as Support Vector Machine (SVM), are built upon 2-class classifiers. One of the most popular way to construct a multi-class classifier from these algorithms is to employ "one-vs-others" approach (Ding and Dubchak 2001), in which a multi-class problem is decomposed into a set of 2-class classification problems, one for each class. In each 2-class subproblem, data points belonging the given class are considered as positive samples, while those belonging to other classes are considered as negative samples. This treatment is reasonable in single-label classification, because the classes here are assumed mutually exclusive and a data point belongs to only one class.

As one contribution in this paper, we point out that the above one-vs-others approach has a fundamental deficiency when it is applied to multi-label classification, although this one-vs-others approach has been broadly used in many existing multi-label classification algorithms.

The deficiency occurs at the training (or model parameter estimation) stage. The traditional training process of the one-vs-others approach for multi-label classification is as following. In a 2-class classifier for a given class, the training data points belonging to the class are used as positive samples, while those not belonging to the class are used as negative samples. This traditional training data division, however, is inconsistent in multi-label classification. For example, let us consider a 4-class multi-label classification problem, in which we have four target classes, which are denoted as $C_1$, $C_2$, $C_3$ and $C_4$, respectively. The 2-class subproblem $C_1$-vs-others is $\{C_1\}$-vs-$\{C_2 \cup C_3 \cup C_4\}$. Suppose data point $x_1$ has two labels $C_1$ and $C_2$. It should simultaneously belong to both class $\{C_1\}$ and class $\{C_2 \cup C_3 \cup C_4\}$. More precisely, $x_1$ belongs to class $\{C_1\}$ with probability 1, and belongs to class $\{C_2 \cup C_3 \cup C_4\}$ with probability 1/3 assuming that every class has same number of training samples. Therefore, $x_1$ can not be completely considered as a positive training sample for class $\{C_1\}$. Suppose another data point $x_2$ has three labels $C_1$, $C_2$ and $C_3$. When training the same $\{C_1\}$-vs-$\{C_1 \cup C_2 \cup C_3\}$ 2-class classifier, $x_2$ should belong to class $C_1$ with probability 1, and belong to class $\{C_2 \cup C_3 \cup C_1\}$ with probability 2/3. Therefore, the training process in traditional one-vs-others approach for multiple-label classification is inconsistent.

A remedy to the training inconsistency problem when using one-vs-others approach in multi-label classification is to weight training errors individually. For example, we can assign a weight of 1 to the training error of misclassifying $x_1$ to class $\{C_2 \cup C_3 \cup C_1\}$, and assign 1/3 to the training error of misclassifying $x_1$ to class $\{C_1\}$.
Another multi-class classifier construction approach from 2-class classifiers is “one-vs-one” approach (Ding and Dubchak 2001), which decomposes a multi-class problem into $K (K - 1) / 2$ two-class subproblems, one for each class pair. This approach can be modified to avoid the above training inconsistency problem as following. When training the $\{C_k\} - \{C_l\}$ 2-class classifier, we divide training data points into three categories: (1) data points with both labels $C_k$ and $C_l$ are excluded, (2) data point not belonging to either class are excluded, (3) the rest data points are uniquely used by either $C_k$ or $C_l$.

We note that one-vs-others approach is particularly appropriate to classify new/testing data points in multi-label classification, because it naturally assigns multiple labels to a new/testing data point. However, one-vs-others approach is not very appropriate for single-label classification, because the $K$ binary classifiers $\{\{C_k\} - \{C_l\}\}$ for $1 \leq k \leq K$ often yield more than one positive decision, which leads to decision ambiguity (Ding and Dubchak 2001). One-vs-one approach uses majority voting in single-label classification. However, in multi-label case, a threshold is required to determine how many labels a new data point should acquire.

An entirely new direction to avoid the training inconsistency problem is to use non-parametric classification methods, such as $K$-Nearest Neighbor ($K$NN) classifier, either directly on the original input space or on a reduced feature space, because there is no training phase involved. The $K$NN approach, however, suffers from unbalanced data distribution problem, which become more severe for multi-label problem. When we assign multiple labels to a test data point, a simple 1NN or 3NN do not have enough information for this multi-label assignment (in contrast to single-label classification). To address this problem, we propose a novel Class Balanced $K$-Nearest Neighbor (CBKNN) approach for multi-label classification by emphasizing balanced usage of data from all the classes.

### Class Balanced Linear Discriminant Analysis

Same as single-label classification, high-dimensional input data could make multi-label classification computationally infeasible due to “curse-of-dimensionality” (Fukunaga 1990). Therefore, dimensionality reduction to prune irrelevant features and reduce dimensionality is necessary prior to classification. In this work, we further develop the classical LDA to reduce the dimensionality of multi-label data. We first point out the difficulties in computing the scatter matrices when using traditional single-label definitions in multi-label classification, and then propose our class-wise multi-label scatter matrices to deal with the problem. Meanwhile the powerful classification capability inherited from classical LDA is preserved.

In traditional single-label multi-class classification, the scatter matrices $S_b$ and $S_w$ are well defined in standard LDA algorithm as per the geometrical dispersion of data points. These definitions, however, become obscure when applied to multi-label classification. Because a data point with multiple labels belong to different classes at the same time, how much it should contribute to the between-class and within-class scatter remains unclear.

Therefore, instead of computing the scatter matrices from data points perspective as in standard LDA, we propose to formulate them by class-wise, i.e., $S_b = \sum_{k=1}^{K} S_b(k)$, $S_w = \sum_{k=1}^{K} S_w(k)$, and $S_l = \sum_{k=1}^{K} S_l(k)$. In this way, the structural variances of the training data are represented more lucid and the construction of the scatter matrices turns out easier. Especially, the ambiguity, how much a data point with multiple labels should contribute to the scatter matrices, is avoided. The multi-label between-class scatter matrix is defined as:

$$S_b = \sum_{k=1}^{K} S_b^{(k)}, S_b^{(k)} = \left( \sum_{i=1}^{l} y_{ik} \right) (m_k - \bar{m})(m_k - \bar{m})^T,$$

where $m_k$ is the mean of class $k$ and $\bar{m}$ is the multi-label global mean, which are defined as follows:

$$m_k = \frac{\sum_{i=1}^{l} y_{ik} x_{ik}}{\sum_{i=1}^{l} y_{ik}}, \quad \bar{m} = \frac{\sum_{k=1}^{K} \sum_{i=1}^{l} y_{ik} x_{ik}}{\sum_{k=1}^{K} \sum_{i=1}^{l} y_{ik}}.$$

Note that, the multi-label global mean $\bar{m}$ defined in Eq. (3) is different from the global mean in single-label sense as in standard LDA. The latter is defined as $\frac{1}{T} \sum_{i=1}^{l} x_i$.

Equipped with the class-wise scatter matrices defined in Eqs. (1–2), we can compute the transformation matrix $U \in \mathbb{R}^{p \times r}$ following the standard LDA algorithm (Fukunaga 1990), where $r$ is the reduced dimensionality. The projected data points are hence computed by $q_i = U^T x_i$.

### Class Balanced $K$-Nearest Neighbor Classifier

Given input data, either original features $x_i$ or reduced features $q_i$, we may use statistical learning method to conduct classification. Considering the training inconsistency problem of one-vs-others method in multi-label classification, we consider to use KNN classifier due to its non-parametric property. However, the following difficulties prevent us from directly using the classical KNN classifier.

The first one is the unbalanced data distribution. The second one is the thresholding problem, which is also caused by the nature of multi-label data. For example, for a 4-class multi-label classification problem same as before, we use 3NN and assume a testing data point $x$ has the following nearest neighbors, $x_1$ with labels $C_3$ and $C_4$, $x_2$ with labels $C_2$ and $C_3$, $x_3$ with labels $C_1$ and $C_4$. Therefore, the most frequently appearing labels in the neighbors of the testing data point $x$ are sorted as following: $C_4$ for 3 times, $C_3$ for 2 times, $C_1$ for 1 times and $C_2$ for 1 times. Apparently, $C_1$ will be assigned to $x$ if this is a single-label problem. However, in multi-label case, a threshold is required to make classification, whereas an optimal one is not known a priori.

In order to exploit its non-parametric property, in this section, we further develop classical KNN method, and propose a Class Balanced $K$-Nearest Neighbor (CBKNN) approach for multi-label classification. The aforementioned
two problems in classical KNN classifier are solved by our B\(K\)NN approach.

Algorithm. Step 1. Given a test data point \(x_i(i + 1 < i ≤ n)\), we pick up \(b\) nearest neighboring data points in each class. This leads to at most \(K \times b\) data points, denoted as \(\Gamma_i\). Let \(\{x_{ij}^{(k)}\}_{j=1}^b\) be the data points in \(\Gamma_i\) for the \(k\)th class, given a similarity matrix \(W ∈ \mathbb{R}^{n \times n}\) with \(W_{ij}\) indicating the similarity between \(x_i\) and \(x_j\), we define:

\[
s_i^{(k)} = \sum_{j=1}^b W(x_{ij}^{(k)}, x_i), \quad s_i = \frac{\sum_{k=1}^K s_i^{(k)}}{K},
\]

where \(W(x_i, x_j) = \exp (-\|x_i - x_j\|^2/2\sigma)\) in this work.

Step 2. We compute the Balanced K-Nearest Neighbor Decision Score of a test data point \(x_i\) for the \(k\)-th class as:

\[
f_i^{(k)} = \frac{s_i^{(k)} - s_i}{s_i}.
\]

Step 3. In order to decide the class membership of \(x_i\), we learn a threshold from training data. We first compute \(f_i^{(k)}\) for all the training data points for the \(k\)th class. A threshold \(h_k\), let \(p_k(h_k)\) and \(r_k(h_k)\) denote the corresponding “precision” and “recall” on the training data using \(\{f_i^{(k)}\}_{i=1}^l\), we select our Adaptive Decision Boundary as:

\[
h_k^{\text{adm}} = \text{arg max}_{h_k} \left( \frac{2}{p_k(h_k) + \frac{1}{r_k(h_k)}} \right),
\]

where \(\alpha\) is an application dependent parameter to determine how much the “weighted F1 score” should be biased to precision, and empirically selected as 0.5 in this work. Finally, the labels \(y_i\) for a test point \(x_i\) is determined by

\[
y_i^{(k)} = \text{sign}\left( f_i^{(k)} - h_k^{\text{adm}} \right).
\]

Obviously, although different class could have very different number of labeled data points, for a given test data point, we pick up the most representative training data points from each class with equal number, i.e., \(b\) for each class, such that the label of a test data point is determined via the information from all the classes in a balanced manner.

Note that, \(b\) in our approach is a free parameter like \(K\) in \(K\)NN, which is normally fine tuned by cross validation. Empirically, small \(b\) gives good classification results.

**Empirical Studies**

We use standard 5-fold cross validation to evaluate the proposed B\(K\)NN approach in multi-label classification, and compare the experimental results with the following most recent multi-label classification methods: (1) Semi-supervised learning by Sylvester Equation (SMSE) (Chen et al. 2008) method, (2) Multi-Label Least Square (MLLS) (Ji et al. 2008) method and (3) Multi-label Correlated Green’s Function (MCGF) (Wang, Huang, and Ding 2009) method. We follow the implementation details as in the original works. The input data are first projected to a lower \(r\) dimensional subspace before being fed into the respective classification approaches. In our evaluations, we set \(r = K - 1\).

<table>
<thead>
<tr>
<th>Approaches</th>
<th>TRECVID</th>
<th>Yahoo</th>
<th>Music</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMSE</td>
<td>10.7%</td>
<td>13.5%</td>
<td>21.3%</td>
</tr>
<tr>
<td>MLLS</td>
<td>23.7%</td>
<td>27.6%</td>
<td>31.2%</td>
</tr>
<tr>
<td>MCGF</td>
<td>24.9%</td>
<td>24.3%</td>
<td>30.3%</td>
</tr>
<tr>
<td>B(K)NN ((b = 5))</td>
<td>25.6%</td>
<td>26.8%</td>
<td>36.2%</td>
</tr>
</tbody>
</table>

We apply the compared approaches on the following three broadly used multi-label datasets from different applications: TRECVID 2005 dataset (Smeaton, Over, and Kraaij 2006) (image), Music emotion dataset (Trohidis et al. 2008) (music), and Yahoo dataset (“Science” topic) (Ueda and Saito 2002) (document). For performance evaluation, we adopt “Average Precision” as recommended by TRECVID (Smeaton, Over, and Kraaij 2006), which computes the precision for each class and average them over all the classes.

We adopt “Average Precision” as our performance metric as recommended by TRECVID (Smeaton, Over, and Kraaij 2006), which computes the precision for each class and average them over all the classes.

Table 1 presents the overall classification performance comparisons measured by average precision on the three data sets. The results show that the proposed B\(K\)NN approach outperforms all the other compared approaches, which quantitatively demonstrate the advantage of our approach.

**Acknowledgments**

This research is supported by NSF CCF-0830780, NSF CCF-0939187, NSF CCF-0917274, NSF DMS-0915228.

**References**


