Graded multilabel classification by pairwise comparison

Ordinale Multilabelklassifikation mittels paarweiser Vergleiche

Bachelor-Thesis von Christian Brinker

Mai 2013

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Departement of Computer Science
Knowledge Engineering Group

Graded multilabel classification by pairwise comparison
Ordinale Multilabelklassifikation mittels paarweiser Vergleiche

Vorgelegte Bachelor-Thesis von Christian Brinker

1. Gutachten: Prof. Dr. Johannes Fürnkranz
2. Gutachten: Eneldo Loza Menzia

Tag der Einreichung:

Erklärung zur Bachelor-Thesis

Hiermit versichere ich, die vorliegende Bachelor-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 23. Mai 2013

(Christian Brinker)

Inhaltsverzeichnis

1 Introduction

1.1 Motivation

The computer technology at first consisted only of some single mainframe systems at some research institutes and universities. Since that time the computer systems have developed to everyday things being at our homes or working places. Thereby the amount of computer systems coming distributed over people changed from one computer in dozens of men to several computer devices per man. During these the computer systems have become necessary to our daily life or working processes. Furthermore in the past ten to twenty years this development let the amount of data in the world nearly explode. The amount of data that was produced in only one day is so tremendous, that if someone wants to work with it, the data has to be filtered immediately during the process of collecting it. To achieve this for each data instance has to be predicted if it is relevant or irrelevant. This is necessary because the capability of storing the data does not compete to its production rate. In long term the amount also grows beneath the capability of a human to work with it on its own. As a sample from daily live the spam filter of an email server can be seen. The enormous amount of spam which is automatically sent each day is even bigger than the capability to read it and the email clients would be full every time. Without spam filters people would spend a lot of their time sorting out undesired mails. The filtering in generally works without a user noticing it. The sensor data arises in fabric production or data about sold products in a supermarket are other examples of such cases of massive data production. Looking at the last one there is the possibility to infer from receipts of customers their shopping behavior and their preferred product combinations. Thereby it would be possible to change the structure of supermarkets to optimize the shopping behavior in an economical way for its owner.
This points out the question, how the information or the classification of such data could be emerged. In terms of the first example this means the decision if some specific email is a spam mail or not. This task is not easily done and the different real world settings need different types of learning processes to solve these classification problems. For example there is the so called supervised learning. The problem of classification in general is to determine for a set of data $\mathbb{X}$, consisting of data instances $\mathbf{x} = x_1, ..., x_n$ with x_i being a measured value in some kind of specific measurement category, called feature, and x_i representing the value in some measurement category a_i, if a specific instance $\mathbf{x}$ is part of the desired class of instances or not. In supervised learning additionally there is knowledge about the desired classification of the instances of a specific data subset, which is used by the different prediction models, in the later called classifiers, to predict the assignments of the classes for the instances not being in the data subset with already known class assignments, to set up a mechanism to infer which ranges of the measurement categories could be used to predict the classification of still unknown data instances. Roughly spoken, which values are interesting for the classification. Afterwards these learned rules were used to classify the data during the working process. An additional dimension of different problems is how the already classified data instances are used to train the classifiers. In batch learning the classifier is inferred from the whole data subset with classified instances at once before he is used to classify the yet unclassified instances. There are several other lear-

ning problems like online learning. Therein the classifier is initially build using only one already classified instance and then is updated with the other data instances one after another. During this thesis only supervised batch learning problems are considered.
Beneath these learning problems there are several different classification problems. These describe the way of classification. The classes to determine cannot only be binary but also be more complex like a discrete value range or a grade of belonging to a class. For example the classification problem is to find out a value in a scale of natural numbers from 0 to 5 with 0 representing the irrelevance to a class and 5 representing the full relevance. The values in-between represent a more or less probable dependency to the class. Compared to the binary classification the other problems allow the existence of partial dependency or the classification of several classes at the same time. As an example for the last one an instance representing a car could be classified as 'big' and 'red'. The first sort of problem uses graded classification and is named ordinal classification. The second one is called multilabel classification and allows several class assignments at once. They will be concreted in the next chapter.
This thesis is about the graded multilabel classification. It consists like the multilabel classification of the classification of several individual labels for one data instance $\mathbf{x}$ with the possibility of each class, here called label, on having a specific grade of dependency to this label. This problem was described for the first time in [3]. This thesis will take a second look at the problem and tries to show alternative attempts of solving it, which will be compared with the solving strategies in [3] in terms of the quality of classification. Therefore their experiment set-up was reconstructed for testing my own solving strategies using pairwise classification and the results will be compared later on in this thesis.
The solving attempts to the graded multilabel classification presented by this thesis are generalizations of the calibrated label ranking. This classification approach to the multilabel classification tries to estimate the labels to be classified by learning binary classifiers doing pairwise comparisons of the labels. This means for each pair of labels one or more classifiers are learned to predict which one of the two labels is more likely to be relevant. The information which other labels are aware is not treated by this classifiers. In the different classification problems such pairwise classifiers showed very good results. Especially the calibrated label ranking as one individual pairwise classification approach was very successfully in the multilabel classification setting. This makes the pairwise approach and especially the approach of the calibrated label ranking very promising to the setting of the graded multilabel classification. The main motivation of this thesis is to examine the suitability of the pairwise approach to the graded multilabel classification. How the calibrated label ranking can be generalized to be used in the setting of the graded multilabel classification and how the pairwise approaches compete to the in [3] proposed solving approaches are the central questions answered by this thesis.
Why is it interesting to formulate different kinds of classification problems, the graded multilabel classification problem in particular, and to find solving strategies for them. In my opinion this question is well answered in [3] which argues by the means of the way data are collected. A graded dependency is far more easy to predict for a human being than an ungraded one. Also the data itself becomes a more detailed grade of information in the way of the declaration of the classification problem itself. Sometimes there is the need to classify several different classes for one data instance, where the correlation and interdependencies of the different classes have to be considered of. In these combined cases the problem has to be graded multilabel classification problem to represent both aspects at the same time.

So collecting data for graded multilabel classification and ways to classify it automatically seems self-evidentially advantageous in times of growing mechanization.

1.2 Goals

The goal of this thesis is to find generalizations of the calibrated label ranking to the graded multilabel classification problem and to evaluate the quality of their predictions. Thereby the suitability of pairwise classifiers in the space of the graded multilabel classification problem should be explored. To solve this the found approaches should be compared with the approaches, which are already available to solve the graded multilabel classification problem.

1.3 Structure

This thesis first provides to the reader the theoretical foundations to understand the graded multilabel classification (see chapter 2). Afterwards the already available approaches to solve the graded multilabel classification are discussed in detail (see chapter 3). Then the main part of the thesis follows when discussing several new approaches to solve the graded multilabel classification problem all generalizing the calibrated label ranking (see chapter 4). At next the quality of prediction of the different approaches is evaluated and compared by an experimental evaluation (see chapter 5). At least the paper is concluded by a summary of its content and the prospects to future research (see chapter 6).

2 Classification problems

The following chapter should enable the reader to understand the theoretical basis of the later discussed solving strategies of the graded multilabel classification problem. To achieve this different classification problems will be discussed and then finally generalized to the graded multilabel classification problem starting with a binary classification problem. Therefore the binary decision of the binary classification problem will be generalized to an ordinal scale of class relevance. Afterwards it will be discussed an orthogonal generalization of the binary problem towards several binary classes, named labels. For each of them it has to be considered which ones have to be classified at the same time. The next discussed problem is label ranking. This formulates the problem of finding an order of preferences of several labels for each instance instead of predicting a subset of all possible labels. The discussion of the label ranking is needed to introduce the calibrated label ranking which is generalized by the solving approaches to graded multilabel classification proposed by this thesis. The graded multilabel classification is the last discussed classification problem. It is a combination of the multilabel and the ordinal classification and is the central discussed problem.
For simplification all classification problems in this chapter are discussed in terms of being a supervised batch learning problem. This assumption is well suited for the comparison of the different problems because the complexity of the learning process of different learning problems is independent of the type of classification problem. The base assumption of the supervised learning is that for some subset $\mathbb{X}_{known}$ of the possible data $\mathbb{X}$ the right classification of each data example $\mathbf{x} \in \mathbb{X}_{known}$ is known at the time of training the classifier. A subset of $\mathbb{X}_{known}$ is used to learn the classifier. This subset is called the training set of the classifier. In batch learning problems the classifier is trained by looking at all instances in the training set at once. Contrary in online learning problems the classifier is initially trained on one instance and afterwards updated by looking at one instance after another.

2.1 Binary classification

The binary classification problem is the simplest thinkable form of a classification problem. For some data set $\mathbb{X}$ with instances

$$\mathbf{x} = \left(x_1 \dots x_n\right) \text{ and } \mathbf{x} \in \mathbb{X}$$

it has to be decided, if these instances $\mathbf{x}$ belong to some class λ or not. $x_1, \dots, x_n$ thereby are the values for the features

$$A = \{a_1, \dots, a_n\}$$

of the data set $\mathbb{X}$. So the classifier is equal to some model

$$H : \mathbb{X} \to \mathbb{M}, \mathbf{x} \mapsto \begin{cases} 1 & \text{if } \mathbf{x} \text{ belongs to } \lambda \\ 0 & \text{otherwise} \end{cases} \quad \text{, with } \mathbb{M} = \{0, 1\}.$$

The assignment of '1' means the instance belongs to class λ. For this kind of problem many different solving approaches, with several allowing a very good classification, exist, if possible

due to the inner appearance of the data set. The rule learners are a good example for one of the different approaches of classifiers. They are a group of classifiers, which aims to build a model for prediction by some set of rules derived from the feature values in the training set. This is done by trying to predict a correlation between the value of a feature or a combination of the values of several features and the class. For example the rules can have the form

If the value of $\mathbf{x}$ *for feature* a_i *is higher than some value t,*
then predict it belongs to class λ.

With the rule set consisting only of this rule the classification model would be

$$H : \mathbb{X} \to \mathbb{M}, \mathbf{x} \mapsto \begin{cases} 1 & \text{if } x_i > t \\ 0 & \text{otherwise} \end{cases}.$$

The specific interdependence between the rules in the rule set and their structure depends on the specific algorithm. They could be pure mathematical or symbolic building a decision tree like the C45-algorithm [12].

2.2 Ordered classification

If the above described binary classification problem is extended by a grade of belonging to the class λ, the ordinal classification problem is created. The target value space $\mathbb{M}$ of the classifier H is changed from a binary range

$$\mathbb{M} = \{0, 1\}$$

to a discrete and ordered finite space

$$\mathbb{M} = \{m_1, \ldots, m_n\}$$

with the inner structure

$$m_1 \prec m_2 \prec \ldots \prec m_n.$$

The values for example can be some subspace of the natural numbers

$$\mathbb{M} = \{0, 1, \ldots, 5\},$$

with '0' having the same meaning as in the binary case and '5' corresponding to '1' from before. All values in between represent some kind of more or less partial classification of λ like 'belongs probably to it' or 'rather not'. An example in the real world could be a review score of the customers in an online shop with products gaining up to 5 points. These data can be collected with features like price or supplier. Note that the semantics of the individual grades in the above example are only one possible interpretation. The lowest grade does not have to have the meaning of the class being not aware and the same holds for the highest grade.
Because this kind of decision to predict a specific grade $m \in \mathbb{M}$ is definitely more complex than a binary one, it is for example more difficult to find an appropriate rule set. The rule set in general would be much bigger. In tests it was shown that the reduction to several binary classification problems can solve the problem in a more efficient way as the direct approach. By

this the complexity of the whole problem is reduced to the combined complexity of the binary problems. In [6] was shown that this is less complex to solve than solving the ordinal problem itself. But it has to be ensured that the additional information about the data gained by the grade is not lost through the act of reduction. Therefore the way of reducing and recombination has to be considered carefully. To take this into account it is tried to build binary classification problems in a way that for each pair of two grades $\left(m_i, m_j\right) \in \mathbb{M}$ with $i < j$ has to be predicted to which one of the two the instance belongs to. Through voting of the classifiers the grade with the highest number of votes is predicted. For instance if five classifier predict the instance belongs to grade m_1 and two predict grade m_2, it is self-evident more likely that the instance belongs to m_1. This sort of procedure is discussed in detail by [6].
In [10] was shown that the decision between two grades seems to be more error prone, if during the reduction to a binary problem instances of higher grades than the higher one of the two compared ones are treated as instances of this grade and analog for the lower grade. The grades in-between can not be associated with one of the two grades because like in [6] written the scale of the grades is not numerical. So there is no kind of closeness of the grades in-between to the surrounding ones. In [6] the error proneness is assumed to be caused by the possible more complex decision boundaries in the feature space when the order in grades does not respond in the feature space. For additional information to this enhancement of the training sets of the pairwise classifiers see [2].

2.3 Multilabel classification

The multilabel classification [13] is another sort of generalization of the binary classification. But the extension used in the multilabel classification is some kind of orthogonal to the one in the ordered classification. Like already written above the classification problem is extended by not only predicting a single class but a subset of all possible binary classes. These classes are named labels. So for an instance $\mathbf{x} \in \mathbb{X}$ has to be found a set of labels

$$P_{\mathbf{x}} \subseteq \mathscr{L}$$

out of the set of all possible Labels $\mathscr{L} = \lambda_1, \ldots, \lambda_n$. The classifier solving this problem is changed to the form

$$H : \mathbb{X} \to \mathscr{P}\left(\mathscr{L}\right), \mathbf{x} \mapsto P_{\mathbf{x}},$$

with $P_{\mathbf{x}}$ being the subset of classified, called labeled, labels λ_i and $\mathscr{P}\left(\mathscr{L}\right)$ being the power set of $\mathscr{L}$. For each instance $\mathbf{x} \in \mathbb{X}$ be $L_{\mathbf{x}}(\lambda)$ a function indicating, if label $\lambda \in \mathscr{L}$ is labeled for this instance or not.

$$L_x : \mathscr{L} \to \{0,1\}, \lambda \mapsto \begin{cases} 1 & \text{,if } \lambda \in P_{\mathbf{x}} \\ 0 & \text{otherwise} \end{cases}$$

Let $\overline{L_{\mathbf{x}}}$ be the complementary function to $L_{\mathbf{x}}$. In practice the set of predicted labels $\hat{P}_{\mathbf{x}}$ may differ from the set of really relevant labels $P_{\mathbf{x}}$.
Like proposed in [3] there are several different strategies to solving the multilabel classiciation problem which use different interpretations of the problem itself. The most proposing approaches use the reduction to binary classification problems. But the several approaches use different

ways to achieve it. The first here discussed one sees each element of $\mathscr{P}(\mathscr{L})$ as single classification problem. For each of them a binary classifier is learned to predict its equality to the searched $P_{\mathbf{x}}$. The drawback of this approach is its complexity. The number of binary classifiers needed grows with $O\left(min\left(2^{|\mathscr{L}|}, n\right)\right)$ with n being the size of the training set. This approach is called label powerset, will not be deepened and is only mentioned for completeness. The second strategy is the so-called binary relevance. Thereby the original problem is reduced to $n = |\mathscr{L}|$ binary classification problems. For each individual label $\lambda_i \in \mathscr{L}$ is learned one binary classifier to predict, if the label belongs to some data instance. Cheng, Dembczyński & Hüllermeier [3] points out that obviously this approach ignores the correlation and interdependencies between the labels which leads to a drawback in the quality of the classification. The third approach tries to predict the preference order of the labels and finds a threshold to separate the higher preferred from the lower preferred ones. Because of the high interest of this approach to the topic of this thesis it is discussed in more details later in the chapter.
An example for the multilabel classification can be a sociological study. Therein the test persons have to answer some questionnaire to sort the persons to several social groups. These social groups can overlap. So a person could belong to more than one group. The set $\mathscr{L}$ of labels would be the set of possible social groups the test person have to be assigned to. An 18-years old boy, who is doing a lot of different sorts of sport, may be classified as 'teenager' and 'passionate to sports'.

2.4 Label ranking

Contrary to the multilabel classification problem in the label ranking problem [8] is not tried to find a subset $P_{\mathbf{x}} \subseteq \mathscr{L}$ but a ranking of the elements of in $\mathscr{L}$. In other words a ranking $\lambda^{(1)} \prec_{\mathbf{x}} \ldots \prec_{\mathbf{x}} \lambda^{(n)}$ with $n = |\mathscr{L}|$, $\lambda^{(1)}$ being the less preferred label for instance $\mathbf{x} \in \mathbb{X}$ and $\lambda^{(n)}$ being the most preferred one is predicted. The ranking represents the probability of a label belonging to an instance $\mathbf{x}$ in relation to the other labels by its position in the ranking. The ranking can obviously be different for each single instance $\mathbf{x} \in \mathbb{X}$. The probability also is no discrete value but only a preference in relation to the other labels and can be directly inferred by its position in the ranking.
The order is gained through learning a scoring function

$$f : \mathbb{X} \times \mathscr{L} \to \mathbb{R},$$

with $\mathbb{R}$ being the set of real numbers. Important is that the result of $f(\cdot, \mathbf{x})$ grows monotone, or the better a strongly monotone, from $\lambda^{(1)}$ to $\lambda^{(1)}$ for an explicit instance $\mathbf{x}$. If the value of f grows accordingly to the probability of the label belonging to an instance, f induces obviously a ranking of the elements in $\mathscr{L}$. In case of f being only monotone the ranking cannot be considered definite because the value of two labels could be equal. The positions of these labels then are interchangeable. This can be necessary to represent the data correct but is out of a ranking point of view not so well suited. The ranking for values

$$f\left(\mathbf{x}, \lambda^{(1)}\right) < f\left(\mathbf{x}, \lambda^{(2)}\right) < \ldots < f\left(\mathbf{x}, \lambda^{(i)}\right) = f\left(\mathbf{x}, \lambda^{(j)}\right) < \ldots f\left(\mathbf{x}, \lambda^{(n-1)}\right) < f\left(\mathbf{x}, \lambda^{(n)}\right)$$

would be

$$\lambda^{(1)} \prec_{\mathbf{x}} \lambda^{(2)} \prec_{\mathbf{x}} \ldots \prec_{\mathbf{x}} \lambda^{(i)} =_{\mathbf{x}} \lambda^{(j)} \prec_{\mathbf{x}} \ldots \prec_{\mathbf{x}} \lambda^{(n-1)} \prec_{\mathbf{x}} \lambda^{(n)}.$$

With a strongly monotone growing f the inferred ranking is definite.

$$f\left(\mathbf{x}, \lambda^{(1)}\right) < f\left(\mathbf{x}, \lambda^{(2)}\right) < \ldots < f\left(\mathbf{x}, \lambda^{(i)}\right) < f\left(\mathbf{x}, \lambda^{(j)}\right) < \ldots f\left(\mathbf{x}, \lambda^{(n-1)}\right) < f\left(\mathbf{x}, \lambda^{(n)}\right)$$

would be

$$\lambda^{(1)} \prec_{\mathbf{x}} \lambda^{(2)} \prec_{\mathbf{x}} \ldots \prec_{\mathbf{x}} \lambda^{(i)} \prec_{\mathbf{x}} \lambda^{(j)} \prec_{\mathbf{x}} \ldots \prec_{\mathbf{x}} \lambda^{(n-1)} \prec_{\mathbf{x}} \lambda^{(n)}.$$

An example of this problem can be the prediction of the winning probabilities of an action of an artificial intelligence playing poker at a special point in a game. The scoring function of the artificial intelligence thereby models the winning probability of the different possible actions to achieve a ranking of these.

2.5 Calibrated label ranking

The calibrated label ranking is the above mentioned third approach to solve the multilabel classification problem. In calibrated label ranking the original multilabel classification problem is reinterpreted as a label ranking problem. The problem is changed from

$$c : \mathbb{X} \rightarrow \mathscr{P}(\mathscr{L})$$

to find a ranking of $\mathscr{L}$ with

$$f : \mathbb{X} x \mathscr{L} \rightarrow \mathbb{R},$$

and some threshold

$$t \in \mathbb{R}$$

to partition the gained ranking into two parts. The sets of labels of the two parts of the ranking are the relevant Labels $P_{\mathbf{x}}$ and the set of not relevant ones $N_{\mathbf{x}}$. So in calibrated label ranking the two sets have the form

$$P_{\mathbf{x}} = \{\lambda | f(\lambda) > t\}$$

and

$$N_{\mathbf{x}} = \mathscr{L} \setminus P_{\mathbf{x}}.$$

So $L_{\mathbf{x}}$ is concreted to

$$L_x(\lambda) = \begin{cases} 1 & \text{if } f(\lambda) > t \\ 0 & \text{otherwise} \end{cases}$$

Obviously the difficulty of this approach is the search of an appropriate threshold t to reduce the difference between the set of predicted labels $\hat{P}_{\mathbf{x}}$ and the correct labels $P_{\mathbf{x}}$. Through experiments was shown that the approach of calibrated label ranking proposed in [7] produces the best results for pairwise classification. This approach tries to predict the ranking by learning

pairwise comparisons between the labels. That means for each pair of labels $\left(\lambda_i, \lambda_j\right) \in \mathscr{L} \times \mathscr{L}$ with $i \neq j$ is learned a binary classifier $H_{(\lambda_i,\lambda_j)}$ to predict which is more probable.

$$H_{(\lambda_i,\lambda_j)} : \mathbb{X} \to \{0,1\}, \mathbf{x} \mapsto \begin{cases} 1 & \text{if } \lambda_i \in P_\mathbf{x} \wedge \lambda_j \notin P_\mathbf{x} \\ 0 & \text{if } \lambda_i \notin P_\mathbf{x} \wedge \lambda_j \in P_\mathbf{x} \end{cases}$$

It is trained with the instances $\mathbf{x} \in \mathbb{X}_{train}$ labeled either with λ_i or λ_j but not if both are labeled. So only the two cases handled in the projection function can occur. To infer the ranking the scoring function is

$$f\left(\mathbf{x}, \lambda_i\right) = \sum_{\substack{\lambda_j \\ \lambda_i \neq \lambda_j}}^{\mathscr{L}} H_{(\lambda_i,\lambda_j)}.$$

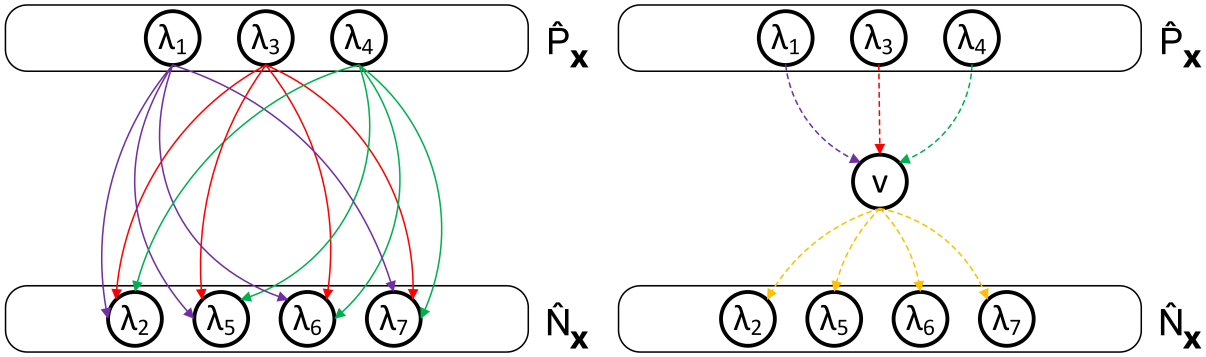
So the score is gained through summing up the votes of the individual binary classifiers which is called voting.
Obviously the prediction of the classifier $H_{(\lambda_i,\lambda_j)}$ is completely complementary to the prediction of the classifier $H_{(\lambda_j,\lambda_i)}$. So the complementary classifier has not to be learned and its prediction can be inferred.
To achieve a better understanding, which classifiers are learned during the training phase, in figure 2.1a the pairwise classifiers $H_{(\lambda_j,\lambda_i)}$ are shown as arcs. In this figure the prediction of the classifiers for an explicit instance $\mathbf{x}$ during the classification step is shown. The prediction is represented by the corresponding arc pointing from the preferred label to the unpreferred one.
To determine the threshold t to partition the scores of the scoring function f, an additional virtual label v is inserted to the set of labels $\mathscr{L}$. The new set of labels used for prediction is

$$\mathscr{L}_v = \mathscr{L} \cup \{v\}$$

During the phase of learning the classifiers $H_{(\lambda_i,\lambda_j)}$ for each classifier $H_{(\lambda_i,v)}$ for each instance $\mathbf{x}$ v is assumed labeled, if λ_i is absent and vice versa. So v is assumed higher ranked than the labels in $N_\mathbf{x}$ but lower ranked than the ones in $P_\mathbf{x}$ (see Figure 2.1b). So the set of learned preferences is changed from holding preferences from $\hat{P}$ to $\hat{N}$ to holding additional ones from $\hat{P}$ to v and from v to $\hat{N}$ (see Figure 2.1c). The resulting ranking consists of the virtual label separating the labels in $\hat{N}$ from the ones in $\hat{P}$ (see Figure 2.1d). So the threshold t is the score of the virtual label v

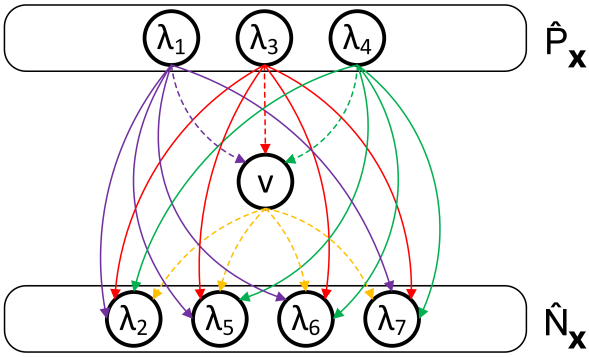
$$t = f\left(\mathbf{x}, v\right).$$

The calibrated label ranking is by this thesis generalized to the graded multilabel classification problem.

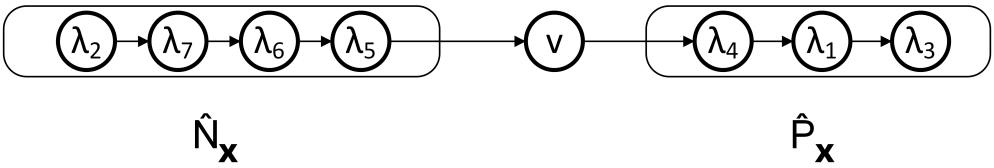


(a) The pairwise comparisons between the labels of a multilabel classification problem

(b) The additional pairwise comparisons due to the virtual label added by the calibrated label ranking to the multilabel classification problem



(c) All pairwise comparisons done in the calibrated label ranking



(d) The predicted ranking of the labels of a multilabel classification problem

Figure 2.1: An example of the prediction of calibrated label ranking

2.6 Graded multilabel classification

The graded multilabel classification problem is the last discussed and central classification problem in this thesis. This problem was introduced for the first time by Cheng, Dembczyński & Hüllermeier [3], which is until now the only publication to this topic.
Graded multilabel classification is a combination of the multilabel classification with the ordered classification. This means each label λ in the set of relevant labels $P_{\mathbf{x}}$ of instance $\mathbf{x} \in \mathbb{X}$ is not longer only relevant or not, but can have a discrete grade of relevance like in ordered classification. For simplification these grades of the different labels are mentioned to be similar for each label. For all labels $\lambda \in \mathscr{L}$ be $\mathbb{M} = \{m_1, m_2, \ldots, m_n\}$ the finite set of ordered grades.
The classification function of the graded multilabel classification problem is

$$H : \mathbb{X} \times \mathscr{L} \to \mathbb{M}, (\mathbf{x}, \lambda) \mapsto m.$$

The function $L_{\mathbf{x}}$ is generalized from

$$L_{\mathbf{x}}\colon \mathscr{L} \to \{0,1\}, \lambda \mapsto \begin{cases} 1 & \text{if } \lambda \in P_{\mathbf{x}} \\ 0 & \text{otherwise} \end{cases}$$

to

$$L_{\mathbf{x}}\colon \mathscr{L} \to \mathbb{M}, \lambda \mapsto \left\{ m_i \,\middle|\, \lambda \in P_{\mathbf{x}}^{m_i} \right\}$$

with $P_{\mathbf{x}}^{m_i}$ being the set of labels with grade m_i. So the set of relevant labels $P_{\mathbf{x}}$ from multilabel classification is generalized to the set

$$P_{\mathbf{x}} = \left\{ P_{\mathbf{x}}^{m_1}, P_{\mathbf{x}}^{m_2}, \ldots, P_{\mathbf{x}}^{m_n} \right\}.$$

The set of grades $\mathbb{M} = \{m_1, m_2, \ldots, m_n\}$ has the inner structure

$$m_1 \prec m_2 \prec \ldots \prec m_n$$

like in ordered classification. This order of $\mathbb{M}$ leads to the effect, that, if a label is labeled with grade m_i, it is assumed also be labeled with all lower grades m_j with $j < i$. The grade m_1 describes the absence of a label and m_n its full presence. All labels $m_2, m_3, \ldots, m_{n-1}$ are grades of particular presence between m_1 and m_n. In contrary to [3] the setting $m_1 = 0$ and $m_n = 1$ is not assumed to prevent from the false impression that the inner structure of $\mathbb{M}$ is some numerical scale. Because like in the ordered classification the 'distance' between two successive grades is not determined. Note that this is one of two possible ways to interpret the ordinal structure. The other possibility is to only assume one specific grade to be classified. For example some data instance $\mathbf{x}$ is classified with $\{(\lambda, m_2)\}$ then in the first interpretation it is also assumed labeled with m_1 but in the second interpretation it is only labeled with grade m_i Because [3] used the first interpretation and this work tries to find approaches which can be compared to the ones in [3], this work uses also the first interpretation.
[3] proposed as biggest advantage of the graded multilabel classification problem the possibility of finer granular data acquisition than in the normal multilabel classification setting. These granularity in graded preference of a label is assumed as more comfortable to a person while judging. Through this the grade of information of the predicted classifications of a graded multilabel classifier is higher than the one of a binary multilabel classifier because of its higher detailed results. But this higher detail is only available with the expense of the problems complexity. For instance the dimension of the target space T grows from $|T_{binary}| = 2^{|\mathscr{L}|}$ to $|T_{graded}| = |\mathbb{M}|^{|\mathscr{L}|}$.

3 Solving approaches to graded multilabel classification from literature

The last chapter discussed several different classification problems. This chapter introduces the existing solving strategies for the graded multilabel classification problem from [3]. Firstly the vertical and horizontal reduction of the graded multilabel classification problem are discussed. These meta-techniques to reduce the complexity of the problem are used by both approaches of [3]. Then a look at the combination of both techniques is taken, which is the first solving approach. Afterwards the second solving approach, named IBLR-ML, is discussed.

3.1 Vertical reduction

Caused by the high complexity of the graded multilabel classification problem the reduction to less complex classification problems is reasonable to reduce error rate and computation time. One approach to achieve this is the in [3] proposed vertical reduction. Here the graded multilabel classification is reduced analog to the binary relevance in multilabel classification. In greater detail in binary relevance for each label $\lambda_1, \ldots, \lambda_n$ of the multilabel classification problem is learned one binary classifier to predict the labels relevance. So the multilabel classification problem is reduced to n individual binary classification problem. In terms of the vertical reduction the graded multilabel classification problem is reduced to n ordered classification problems.

$$H : \mathbb{X} \times \mathscr{L} \rightarrow \mathbb{M} \text{ is transformed to } [H]_{\lambda_1}, \ldots, [H]_{\lambda_n}$$

with

$$[H]_{\lambda_i} : \mathbb{X} \rightarrow \mathbb{M} \text{ for } i = 1, \ldots, n.$$

The notation $[\cdot]_{rp}$ denotes the function belongs to the reduced problem rp and exists only in it. The training sets of the individual ordinal classifiers $[H]_{\lambda_i}$ consist of all instances from the training set of the graded multilabel classification problem H but their classification is reduced to the grade $m \in \mathbb{M}$ of the label λ_i in the original classification. Like in the binary relevance the information about correlation and interdependencies between the individual labels is lost and can not be used to optimize the process of classification.

3.2 Horizontal reduction

Similar to a reduction to ordered classification problems the graded multilabel classification can be reduced to several binary multilabel classification problems. This was proposed in [3] as horizontal reduction. Inspired by the fuzzy set theory [15] the target space of the classifier H is divided along each grade $m_i \in \mathbb{M}$ into levels. In other words for each grade $m_i \in \mathbb{M}$ is learned one classifier to predict, if a label should be classified or not. The classification function is replaced by

$$H : \mathbb{X} x \mathscr{L} \rightarrow \mathbb{M} \text{ is transformed to } [H]_{m_0}, \ldots, [H]_{m_n}$$

with

$$[H]_{m_i} : \mathbb{X} \to \mathscr{P}(\mathscr{L}) \text{ for } i = 1, \ldots, n.$$

Analog to the vertical reduction the individual multilabel classifiers are trained with the whole set of training data. In difference to the vertical reduction in the horizontal reduction the classification of the individual training data instances is reduced to the information if the grade of a distinct label is higher than the grade depending to the level or not. If the grade is higher or equal than the level grade the instance is treated as a positive example and as a negative one otherwise. Like mentioned in [3] the classifiers $[H]_{m_i}$ are not independent of each other. If some label is relevant to some grade m_i, it is also relevant to all grades $m_j \prec m_i$. In other words the for each prediction of the multilabel classifiers must hold

$$\left[\hat{P}_{\mathbf{x}}\right]_{m_j} \subseteq \left[\hat{P}_{\mathbf{x}}\right]_{m_i} \text{ if } j < i.$$

So the size of the sets grows monotone with increasing grade. This obviously leads to the non-trivial problem of assuring this monotony during the aggregation of the predictions of the $n = |\mathbb{M}|$ different classifiers to the overall prediction of the classifier ensemble. In general this can not be done. To solve this problem [3] proposed to choose for each label the highest predicted grade. In my opinion this strategy is prone for classification errors. If some classifier $[H]_{m_i}$ classifies a label λ as false positive, in other words $\lambda \in \left[\hat{P}_{\mathbf{x}}\right]_{m_i}$ but $\lambda \notin \left[P_{\mathbf{x}}\right]_{m_i}$, and all other classifiers predict correct, the label would be predicted to have grade m_i. So one single classifier can produce a wrong prediction for the complete ensemble of classifiers. The intensity of the error in terms of the order of grades only corresponds to the grade of the wrong classifying classifier itself. For example, if a classifier, which predicts, if a label is relevant with grade m_5, predicts this label to be relevant with grade m_5, even if the correct grade will be m_1 and all other classifiers do a correct prediction, this would be the prediction of the whole ensemble with an error of four grades distance. In contrary if $m_i \prec m_{correct}$ and $[H]_{m_i}$ would predict the label to not be relevant with grade m_i this error would be ignored. So the approach of [3] is vulnerable to predict too high grades and making high errors. But also being robust to predict too low grades.

3.3 Complete reduction

The combination of both above mentioned reduction approaches is the first solving approach discussed in [3]. In the further text I will call it complete reduction because it reduces the graded multilabel problem to a set of binary classification problems consisting of one binary classifier for each possible output combination of label and grade. [3] showed that there are methods for both vertical and horizontal reduction which can be seen as analog to the other reduction technique and can be applied to the resulting ordered or multilabel classification problems. For the first sort of problem this is the approach of [5] and for the second one it is the binary reduction, which is already discussed in the section about the multilabel classification in the last chapter.
If both reductions are applied to the graded multilabel classification problem, it is gained a set

of $|\mathbb{M}x\mathscr{L}|$ binary classification problems. One for each grade-to-label combination $(\lambda_i, m_j) \in \mathbb{M}x\mathscr{L}$. The single classifiers are

$$[H]_{(\lambda_i, m_j)} : \mathbb{X} \to \{0, 1\}$$

The prediction of the classifier $[H]_{(\lambda_i, m_j)}$ assumes that for an instance $\mathbf{x}$ the label λ_i is relevant with a grade $m_k \geq m_j$ or with other words

$$L_{\mathbf{x}}(\lambda_i) \geq m_j.$$

[3] proposed that the order of applying the horizontal and vertical reduction is interchangeable as long as the above mentioned monotonicity condition holds, because it only differs in the type of aggregating the binary classifiers. Any differences between the two ways of aggregation is due to violations of this condition.

3.4 IBLR-ML

The second approach to solve the graded multilabel classification proposed by [3] uses the horizontal reduction to achieve a set of $k = |\mathscr{L}|$ independent multilabel classification problems. These problems are individually solved by the IBLR-ML method. This method was first published in [4]. It combines the logistic regression with instance-based learning to take advantage of the interdependencies and correlation between the different labels of a multilabel classification problem. The prediction of the method is based on the assumed posterior probability $\pi_0^{(i)}$ of an label λ_i being relevant for some instance $\mathbf{x}_0$. The classifier for this setting is

$$[H]_{m_j}(\mathbf{x}_0) = \left\{ \lambda_i \in \mathscr{L} | \pi_0^{(i)} > \frac{1}{2} \right\}.$$

The posterior probabilities $\pi_0^{(i)}$ are derived from the logistic regression equation

$$log\left(\frac{\pi_0^{(i)}}{1-\pi_0^{(i)}}\right) = \omega_0^{(i)} + \sum_{j=1}^{|\mathscr{L}|} \gamma_j^{(i)} \cdot \omega_{+j}^{(i)}(\mathbf{x}_0)$$

with

$$\omega_{+j}^{(i)}(\mathbf{x}_0) = \sum_{\mathbf{x} \in \mathscr{N}_k(\mathbf{x}_0)} \kappa(\mathbf{x}_0, \mathbf{x}) \cdot y_j(\mathbf{x}),$$

$\kappa(\mathbf{x}_0, \mathbf{x})$ being an kernel function like the in [3] used KNN-kernel

$$\kappa(\mathbf{x}_0, \mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathscr{N}_k(\mathbf{x}_0) \\ 0 & \text{otherwise} \end{cases},$$

$\mathscr{N}_k(\mathbf{x}_0)$ being the set of k nearest neighbours of $\mathbf{x}_0$,

$$y_j(\mathbf{x}) = \begin{cases} 1 & \text{if } \lambda_j \in P_{\mathbf{x}} \\ -1 & \text{otherwise} \end{cases}$$

and $\gamma_j^{(i)}$ being an coefficient representing the correlation between label λ_i and λ_j.
[3] suggested another setup to solve the graded multilabel classification. It uses the vertical reduction instead of the horizontal one and exchanges the binary logistic regression by an ordinal logistic regression.

3.5 Comparison of Reduction techniques

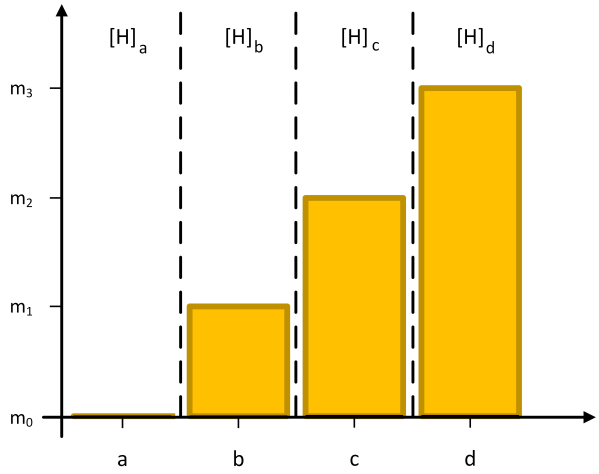
For a better understanding how the different reduction techniques reduce the graded multilabel classification problem in the following an example of how an instance $\mathbf{x}$ is used to learn the individual classifiers from the reduced problems.
Assume the following graded multilabel classification problem

$$\begin{aligned} \mathbb{M} &= \{m_0, m_1, m_2, m_3\} \\ \mathscr{L} &= \{a, b, c, d\} \end{aligned} .$$

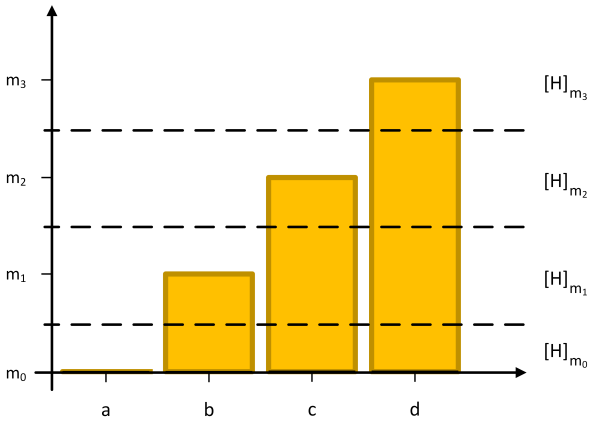
and an instance $\mathbf{x}$ with the given real classification

$$L_{\mathbf{x}} = \{(a, m_0), (b, m_1), (c, m_2), (d, m_3)\}.$$

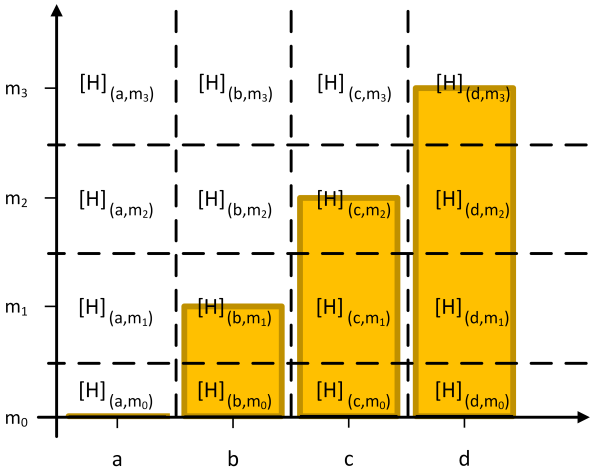
In the figures 3.1a), 3.1b) and 3.1c) the classification of the instance is shown. The x-axes shows the label range and the y-axes the grade range. The colored area shows that a label is relevant with a certain grade. The dotted lines separate the areas used by the individual classifiers to predict the correct classification. The reduction of the vertical reduction is shown in figure 3.1a). There a single classifier has only information about the grade of one label. In figure 3.1b) the classifiers of the horizontal reduction are shown. The individual classifiers have only knowledge which labels have some distinct grade and which ones not. The classifiers of the complete reduction can be seen in Figure 3.1c). Obviously there are more classifiers needed than in the horizontal or vertical reduction. The instance is used as positive instance for classification if the inner of the dotted rectangle in the figure is (partially) filled with color.



(a) The reduction of an instances classification in vertical reduction



(b) The reduction of an instances classification in the horizontal reduction



(c) The reduction of an instances classification in the complete reduction

Figure 3.1: An example of the reduction of the classification of an instance in vertical, horizontal and complete reduction

4 Graded multilabel classification by pairwise comparisons

In the last chapter the solving approaches to the graded multilabel classification of [3] were discussed. This chapter will propose two new approaches to it generalizing the calibrated label ranking to the graded multilabel classification. This is the central topic of this thesis. At first the calibrated label ranking is discussed in more detail. Afterwards the concrete implementation is printed as pseudo code. Then the three approaches to the graded multilabel classification are created step by step by generalizing from the calibrated label ranking to the graded case. During this extension the several steps are discussed and reasoned. Afterwards the concrete implementations of the approaches to graded multilabel classification are shown as pseudo code. Lastly the complexity of each approach is discussed.

4.1 Calibrated label ranking for multilabel classification

Like mentioned above the calibrated label ranking tries to predict a ranking of the labels from the set of labels $\mathscr{L}$ of a multilabel classification problem. Afterwards a threshold t is searched to separate this ranking. To make this possible the approach of [6] induces a virtual label v into $\mathscr{L}$. By also ranking virtual label its position in the final ranking represents the separating threshold. Like stated above this label is preferred less than the relevant labels but more than the not relevant ones. To achieve a ranking 'learning by pairwise comparison' ([6]) is used, which is a totally different binarization strategy as the one used in the reduction techniques from [3]. Like mentioned when introduced the calibrated label ranking, to achieve the ranking the labels are pairwise compared. In more detail for each pair of labels $\left(\lambda_i, \lambda_j\right) \in \mathscr{L} \times \mathscr{L}$ is learned a binary classifier

$$[H]_{(\lambda_i,\lambda_j)} : \mathbb{X} \to \{0,1\}, \mathbf{x} \mapsto \begin{cases} 1 & \text{if } \lambda_i \in P_\mathbf{x} \wedge \lambda_j \notin P_\mathbf{x} \\ 0 & \text{if } \lambda_i \notin P_\mathbf{x} \wedge \lambda_j \in P_\mathbf{x} \end{cases}$$

to predict if the label λ_i is more preferred than the label λ_j for an instance $\mathbf{x} \in \mathbb{X}$. Note that like mentioned above the classifiers are only learned with instances of one of the two cases handled in the projection function p. Finding a appropriate scoring function f is no trivial task because some relations may not be part of the training set. [6] proposed a solution. The scoring function f is achieved by summing up the results of the different classifiers $[H]_{(\lambda_i,\lambda_j)}$

$$f\left(\mathbf{x}, \lambda_i\right) = \sum_{i \neq j} [H]_{(\lambda_i,\lambda_j)}$$

Or in other words the score is gained by full unweighted 0/1-voting. This score is equal to the number of labels λ_j which are less preferred than λ_i. [6] showed, that this solution is good in practice and also proposed a theoretic reasoning to it. To induce the virtual label v into the ranking it has also to be compared to each label $\lambda \in \mathscr{L}$. So the virtual label has to be induced into the label set before the classifiers $[H]_{(\lambda_i,\lambda_j)}$ are learned

$$\mathscr{L}_v = \mathscr{L} \cup \{v\}$$

By this the virtual label is pairwise compared to each label through binary classifiers.
The upcoming question is how to transform the original problem to the reduced problem. The instances $\mathbf{x} \in \mathbb{X}$ can be reused but the set $P_\mathbf{x}$ denoting the set of relevant labels has to be projected in some kind to the set $\{0,1\}$. Because for the comparison of the labels λ_i and λ_j the instances with $\lambda_i, \lambda_j \in P_\mathbf{x}$ or $\lambda_i, \lambda_j \notin P_\mathbf{x}$ have no additional information to separate the labels, these instances have not to be mentioned for training the classifier $[H]_{(\lambda_i,\lambda_j)}$. So the data set used to train the classifier can be reduced to

$$\left[\mathbb{X}_{train}\right]_{(\lambda_i,\lambda_j)} = \left\{ \mathbf{x} \in \mathbb{X}_{train} \middle| \left(\lambda_i \in P_\mathbf{x} \wedge \lambda_j \notin P_\mathbf{x}\right) \vee \left(\lambda_i \notin P_\mathbf{x} \wedge \lambda_j \in P_\mathbf{x}\right) \right\}.$$

Through this the binary classifiers only use instances with information about the decision. So the complexity of training each classifier is reduced and thereby the overall complexity in terms of training time of the approach is also minimized. Additional the quality of each binary classifier is improved because simpler decision boundaries in the feature space have to be found.
The missing detail to find an appropriate projection is that the virtual label v is not labeled for any instance $\mathbf{x} \in \mathbb{X}_{train}$. So the set $P_\mathbf{x}$ has to be enriched by the virtual label to some set $P_\mathbf{x} \cup \{v\}$ to train the classifiers $[H]_{(\lambda,v)}$. To rank the virtual label between the relevant and not relevant ones, $P_\mathbf{x}$ only is enriched in absence of λ. So the enriched sets for the classifiers are

$$P_{\mathbf{x},v} = \begin{cases} P_\mathbf{x} & \text{if } \lambda \in P_\mathbf{x} \\ P_\mathbf{x} \cup \{v\} & \text{otherwise} \end{cases}.$$

So the above mentioned projection from the set of relevant labels $P_\mathbf{x}$ to the binary range is some function

$$\left[p\right]_{(\lambda_i,\lambda_j)} : \mathscr{P}\left(\mathscr{L}_v\right) \to \{0,1\}, P_{\mathbf{x},v} \mapsto \begin{cases} 1 & \text{if } \lambda_i \in P_{\mathbf{x},v} \wedge \lambda_j \notin P_{\mathbf{x},v} \\ 0 & \text{if } \lambda_i \notin P_{\mathbf{x},v} \wedge \lambda_j \in P_{\mathbf{x},v} \end{cases}.$$

Note that all instances with no difference to $P_{\mathbf{x},v}$ towards the pairwise comparison have to be filtered out before the instances are projected. This is no problem towards the quality of the prediction of the classifiers because there is no additional information to distinguish between the awareness of the two labels. If P was the set of all $P_\mathbf{x}$ and $[P]_{(\lambda_i,\lambda_j)}$ the set of all $P_\mathbf{x} \in P$ would be transformed by $\left[p\right]_{(\lambda_i,\lambda_j)}$.
If an instance $\mathbf{x} \in \mathbb{X}$ has to be classified by the ensemble H of binary classifiers, each individual classifier makes its independent prediction. Afterwards the score is gained by summing up the individual binary predictions and the score of the virtual label is used as threshold

$$t = f\left(\mathbf{x}, v\right).$$

Finally the labels

$$\hat{P}_\mathbf{x} = \{\lambda \in \mathscr{L} | f\left(\mathbf{x}, \lambda\right) > t\}$$

are predicted as relevant.

4.1.1 Implementation of calibrated label ranking

Like stated in [6] the calibrated label ranking can be divided into two different phases. The first is the training phase, wherein the reduction to the binary problems and the training of the corresponding binary classifiers is done. This phase is exemplary implemented in pseudo-code in algorithm 4.1.1. The second phase is the classification of an unknown instance $\mathbf{x} \in \mathbb{X}$. The phase is implemented in algorithm 4.1.2.

Algorithm 4.1.1 Training_CLR

Input: $\mathscr{L}, \mathbb{X}_{train}, P$
Output: H

1. $\mathscr{L}_v \leftarrow \mathscr{L} \cup \{v\}$
2. $H \leftarrow \{\}$
3. **for** $i \leftarrow 1$ **to** $\left|\mathscr{L}_v\right| - 1$ **do**
4. $\quad \lambda_i \leftarrow \mathscr{L}_v[i]$
5. $\quad$ **for** $i \leftarrow i+1$ **to** $\left|\mathscr{L}_v\right|$ **do**
6. $\quad\quad \lambda_j \leftarrow \mathscr{L}_v[j]$ // learn pairwise comparison of λ_i, λ_j
7. $\quad\quad [\mathbb{X}]_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
8. $\quad\quad [P]_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
9. $\quad\quad$ **for all** $\mathbf{x} \in \mathbb{X}_{train}$ **do** // filter training instances and learn a pairwise classifier
10. $\quad\quad\quad P_{\mathbf{x}} \leftarrow P[\mathbf{x}]$
11. $\quad\quad\quad$ **if** $\left(\lambda_i = v \wedge \lambda_j \notin P_{\mathbf{x}}\right)$ **or** $\left(\lambda_j = v \wedge \lambda_i \notin P_{\mathbf{x}}\right)$ **then** // if one label is a virtual label and the other label is not aware, then assume the virtual label being aware
12. $\quad\quad\quad\quad P_{\mathbf{x},v} \leftarrow P_{\mathbf{x}} \cup \{v\}$
13. $\quad\quad\quad$ **else**
14. $\quad\quad\quad\quad P_{\mathbf{x},v} \leftarrow P_{\mathbf{x}}$
15. $\quad\quad\quad$ **end if**
16. $\quad\quad\quad$ **if** $\lambda_i \in P_{\mathbf{x},v} \wedge \lambda_j \notin P_{\mathbf{x},v}$ **or** $\lambda_i \notin P_{\mathbf{x},v} \wedge \lambda_j \in P_{\mathbf{x},v}$ **then** // filter out instances not distinguishing between λ_i and λ_j
17. $\quad\quad\quad\quad [\mathbf{x}]_{(\lambda_i,\lambda_j)} \leftarrow \mathbf{x}$
18. $\quad\quad\quad\quad [\mathbb{X}]_{(\lambda_i,\lambda_j)} \leftarrow [\mathbb{X}]_{(\lambda_i,\lambda_j)} \cup \left\{[\mathbf{x}]_{(\lambda_i,\lambda_j)}\right\}$ // extend training set of pairwise classifier with instance **x**
19. $\quad\quad\quad\quad \left[P_{\mathbf{x}}\right]_{(\lambda_i,\lambda_j)} \leftarrow [p]_{(\lambda_i,\lambda_j)}\left(P_{\mathbf{x},v}\right)$ // detect which label is more preferred in this instance
20. $\quad\quad\quad\quad [P]_{(\lambda_i,\lambda_j)} \leftarrow [P]_{(\lambda_i,\lambda_j)} \cup \left[P_{\mathbf{x}}\right]_{(\lambda_i,\lambda_j)}$ // label instance
21. $\quad\quad\quad$ **end if**
22. $\quad\quad$ **end for**
23. $\quad\quad$ **if** $[\mathbb{X}]_{(\lambda_i,\lambda_j)} \neq \{\}$ **then** // filter out classifiers with an empty training set
24. $\quad\quad\quad$ train $[H]_{(\lambda_i,\lambda_j)}$ on $[\mathbb{X}]_{(\lambda_i,\lambda_j)}, [P]_{(\lambda_i,\lambda_j)}$ // train a pairwise classifier
25. $\quad\quad\quad H \leftarrow H \cup \left\{[H]_{(\lambda_i,\lambda_j)}\right\}$
26. $\quad\quad$ **end if**
27. $\quad$ **end for**
28. **end for**
29. **return** H

Algorithm 4.1.2 Classification_CLR

Input: $\mathbf{x}, \mathscr{L}, H$
Output: $\hat{P}_{\mathbf{x}}, \hat{N}_{\mathbf{x}}$

1. $f \leftarrow []$
2. **for all** $[H]_{(\lambda_i,\lambda_j)} \in H$ **do** // sum up the votes of the classifiers
3. **if** $[H]_{(\lambda_i,\lambda_j)}(\mathbf{x}) = 1$ **then**
4. $f\left[\lambda_j\right] \leftarrow f\left[\lambda_j\right] + 1$
5. **else**
6. $f\left[\lambda_i\right] \leftarrow f\left[\lambda_i\right] + 1$
7. **end if**
8. **end for**
9. $\hat{P}_{\mathbf{x}}, \hat{N}_{\mathbf{x}} \leftarrow \{\}$
10. **for all** $\lambda \in \mathscr{L}$ **do** // look which labels are predicted
11. **if** $f[\lambda] \leq f[v]$ **then** // compare position in ranking with the position of the virtual label
12. $\hat{N}_{\mathbf{x}} \leftarrow \hat{N}_{\mathbf{x}} \cup \{\lambda\}$ // the virtual label is predicted not to be relevant
13. **else**
14. $\hat{P}_{\mathbf{x}} \leftarrow \hat{P}_{\mathbf{x}} \cup \{\lambda\}$ // the virtual label is predicted to be relevant
15. **end if**
16. **end for**
17. **return** $\hat{P}_{\mathbf{x}}, \hat{N}_{\mathbf{x}}$

4.1.2 Complexity of calibrated label ranking

In [6] the complexity of the training phase is stated as being in $O(d \cdot l \cdot n)$ with d being the average number of labels in $P_{\mathbf{x}}$, l being the number of labels in $\mathscr{L}$ and n being the number of examples $\mathbf{x} \in \mathbb{X}_{train}$.
Obviously the complexity of the classification is $O\left(l^2\right)$. [6] proposed to use the QWeighted algorithm [11] to reduce the complexity in general to $O\left(d \cdot l \cdot \log l\right)$ per instance to classify.

4.2 The direct approach to generalize calibrated label ranking

How can the above mentioned approach be generalized to solve the graded multilabel classification problem? Like described in [3] the classical multilabel classification problem can be mentioned to be some special simplified case of the graded multilabel classification problem with $|\mathbb{M}| = 2$ and one threshold is needed to separate these two grades in the ranking produced by the calibrated label ranking. With that in mind obviously there are $n-1$ thresholds needed to separate $n = |M|$ different grades in a ranking. So the most obvious approach seems to be learning some ranking of the labels like before but instead of enriching the set of labels $\mathscr{L}$ by only one virtual label v it has to be found some rule to place one virtual label for each pair of following grades m_i and m_{i+1} correct into the order. So there are pairwise classifiers learned for each pair from $\mathscr{L} \times \mathscr{L}$ like before. Obviously the projection function $\left[p\right]_{(\lambda_i,\lambda_j)}$ has to be changed to take the grades of the labels into account. This is easily done by generalizing the binary decision $\left(\lambda_i \in P_{\mathbf{x}} \wedge \lambda_j \notin P_{\mathbf{x}}\right) \vee \left(\lambda_i \notin P_{\mathbf{x}} \wedge \lambda_j \in P_{\mathbf{x}}\right)$ to the comparison of the individual

grades $\left(L_{\mathbf{x}}\left(\lambda_i\right) \prec L_{\mathbf{x}}\left(\lambda_j\right)\right) \vee \left(L_{\mathbf{x}}\left(\lambda_j\right) \prec L_{\mathbf{x}}\left(\lambda_i\right)\right)$. So the relations learned by the classifiers are stating that one label is more relevant than the other. In difference to before this is not only done for one pair of grades but all possible pairs of grades at once.
Next thing to get a ranking of the labels is finding an appropriate scoring function f. Because the projection into the binary problems is that simple and the behavior of the classifiers is not changed, the same scoring function as in the calibrated label ranking

$$f\left(\mathbf{x}, \lambda_i\right) = \sum_{\substack{\lambda_j \\ \lambda_i \neq \lambda_j}} \left[H\right]_{\left(\lambda_i, \lambda_j\right)}$$

can be used.
At least a way to induce the virtual labels into the ranking has to be found. The simplest way is to enrich the set of labels $\mathscr{L}$ analog to the calibrated label ranking with the whole set of virtual labels

$$V = \left\{v_0, \ldots, v_n\right\} \text{ with } n = |\mathbb{M}|$$

before the binary classifiers are learned. So the set of labels which elements are compared pairwise is

$$\mathscr{L}_V = \mathscr{L} \cup V.$$

Similar to the calibrated label ranking there are no instances $\mathbf{x} \in \mathbb{X}$ which are labeled with any virtual label. So the projection function has to be changed to have regard to the graded virtual labels. To hold the monotony constraint the virtual label partitioning higher grades always have to be preferred above the virtual labels partitioning lower grades. So their order can directly inferred by their index. Also a virtual label v separating two labels λ and λ' with λ being relevant with grade m_i, λ' being relevant with grade m_j and $m_i \prec m_j$, v has to be preferred more than λ_i but less than λ_j. So the projection function to project the graded classification of the instance into the binary range has the form

$$\left[p\right]_{\left(\lambda_i, \lambda_j\right)} : L'_{\mathbf{x}} \mapsto \begin{cases} 1 & \text{if } L'_{\mathbf{x}}\left(\lambda_j\right) \prec_v L'_{\mathbf{x}}\left(\lambda_i\right) \\ 0 & \text{if } L'_{\mathbf{x}}\left(\lambda_i\right) \prec_v L'_{\mathbf{x}}\left(\lambda_j\right) \end{cases}$$

with

$$L'_{\mathbf{x}} : \lambda \mapsto \begin{cases} L_{\mathbf{x}}(\lambda) & \text{if } \lambda \notin V \\ \lambda & \text{otherwise} \end{cases}$$

and $\prec_v$ being an order of the grades separated by the virtual labels $v_i \in V$ so that holds

$$m_0 \prec_v v_0 \prec_v m_1 \prec_v v_1 \prec_v m_2 \prec_v \ldots \prec_v v_{n-1} \prec_v m_n$$

with $n = |\mathbb{M}|$ and

$$m_0 \prec m_1 \prec m_2 \prec \ldots \prec m_n.$$

If L is the set of all $L_{\mathbf{x}}$ for all $\mathbf{x} \in \mathbb{X}_{train}$. Note that the classifiers $H_{\left(v, v'\right)}$ with $v \neq v'$ comparing to virtual labels do not have to be trained because due to the fact that the projection all instances are labeled as being relevant with the higher graded virtual label. So these classifiers can be

replaced by some constant vote towards the higher virtual label.
So like in calibrated label ranking, additionally to the pairwise comparisons between the normal labels $\lambda \in \mathscr{L}$ (see figure 4.1a) the normal labels have to be compared pairwise with each virtual label (see figure 4.1b). In figure 4.1c the complete set of pairwise comparisons can be seen. Note that compared with the calibrated label ranking the amount of pairwise comparisons grows for each additional virtual label. So the complexity of the solving approach also depends on the number of grades.
To achieve a prediction the ranking induced by the scoring function is separated into the sets $\hat{P}_{\mathbf{x}}^{m_i}$. This is shown for an example in figure 4.1d.
The problem of this approach is that for a comparison of some labels λ_i, λ_j the size of the difference in terms of their grades in some instance $\mathbf{x}$ does not harm their grade of information to a binary classifier $[H]_{(\lambda_i,\lambda_j)}$. This is caused by the projection into the binary space done by the projection function $\left[p\right]_{(\lambda_i,\lambda_j)}$. So instances with a high difference between the grades of the two labels would be as meaningful to the classifier as instances with a low difference. Due to this the correlation between the labels is reduced to some binary relationship $\lambda_i \prec_{\mathbf{x}} \lambda_j$. Roughly spoken the expressiveness of the individual classifiers and thereby the expressiveness of the whole approach is reduced in terms of the reduced correlation between the labels. But the major interest in formulating the graded multilabel classification problem was the additional information gained through the grades and the graded correlation. In addition this approach is more prone for problems caused by noise during the acquisition of the data. For example if two instances $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{X}_{train}$ for two labels λ_i, λ_j in fact have the same grade and through some kind of failure during the acquisition of $\mathbb{X}$ for instance $\mathbf{x}_2$, λ_i is labeled with a higher grade, the decision boundary of the classifier $[H]_{(\lambda_i,\lambda_j)}$ may be changed compared to the real decision boundary. Because the binary classifiers cannot determine how big the difference in grade is, the wrong labeled instance has a maximal influence to the predictions of this classifier. Because the grade of a label holds the information about the relevance of the label, an approach to solve the graded multilabel classification problem should have some robustness to bigger errors in terms of the grade. But this simple approach does not have this robustness. Besides the 'small' number of classifiers produces only a small range in the target set of the scoring function to induce the virtual labels into. For example the setting of a distinct graded multilabel problem has a very big set of grades but a very small number of labels. So the direct solving approach has to induce a high number of virtual labels in some very small ranged ranking, which promises to cause a high failure rate.

4.2.1 Implementation of the solving approach

Like the calibrated label ranking the direct approach can be partitioned into two phases. The training phase is implemented using pseudo-code in algorithm 4.2.1 and the classification phase in algorithm 4.2.2.

4.2.2 Complexity of the solving approach

Obviously the complexity of the training phase of this approach is related to the complexity of the training complexity of the calibrated label ranking. The only difference in complexity is the enhancement of the label set with the set of virtual labels. So the complexity is $O\left(d \cdot l_v \cdot n\right)$ with

Algorithm 4.2.1 Training_Direct

Input: $\mathscr{L}, \mathbb{M}, \mathbb{X}_{train}, L$
Output: H

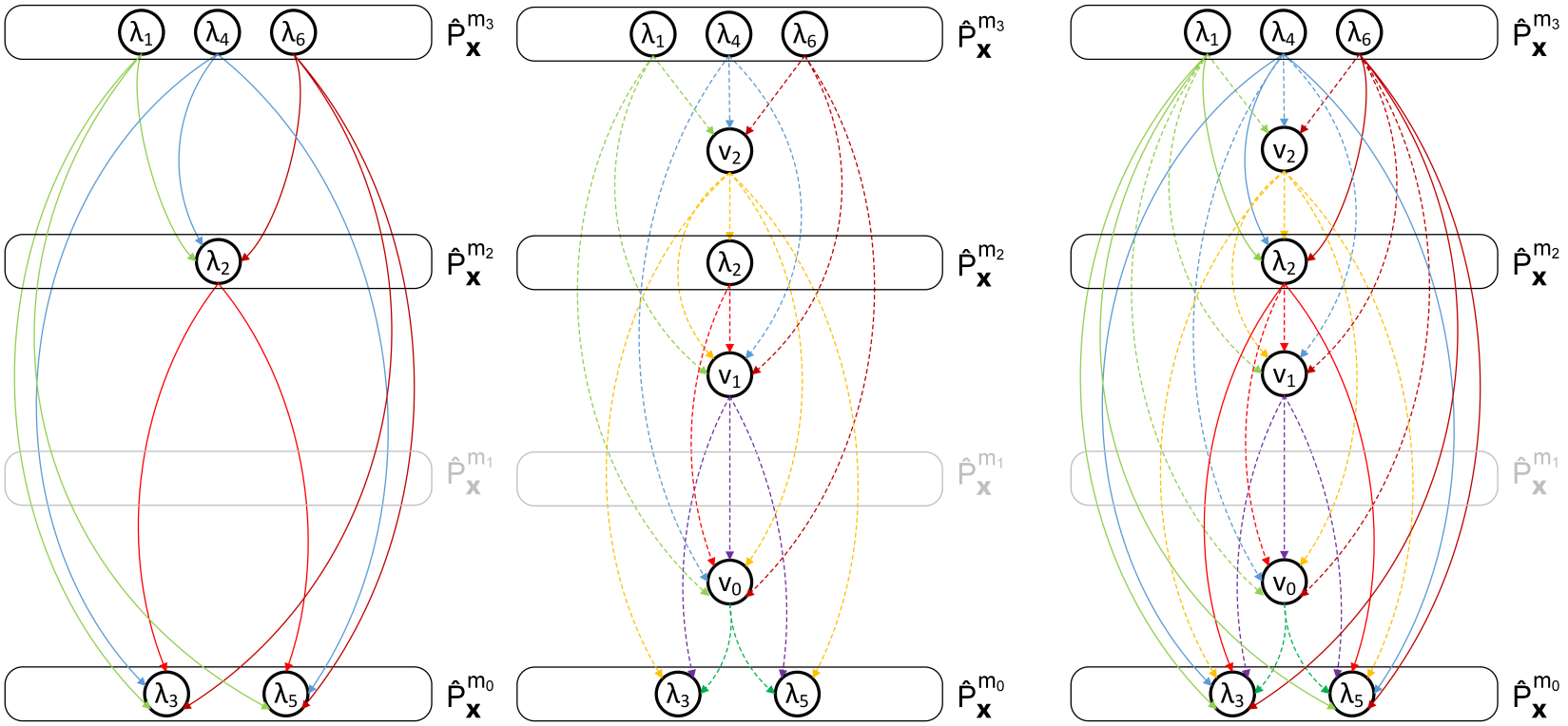
1. $n \leftarrow |\mathbb{M}| - 2$
2. $\mathscr{L}_v \leftarrow \mathscr{L} \cup \{v_0, \ldots, v_n\}$ // extend label set with virtual labels
3. $H \leftarrow \{\}$
4. **for** $i \leftarrow 1$ **to** $\left|\mathscr{L}_v\right| - 1$ **do**
5. $\quad \lambda_i \leftarrow \mathscr{L}_v[i]$
6. $\quad$ **for** $j \leftarrow i + 1$ **to** $\left|\mathscr{L}_v\right|$ **do**
7. $\quad\quad \lambda_j \leftarrow \mathscr{L}_v\left[j\right]$ // learn pairwise comparison of λ_i, λ_j
8. $\quad\quad [\mathbb{X}]_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
9. $\quad\quad [P]_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
10. $\quad\quad$ **for all** $\mathbf{x} \in \mathbb{X}_{train}$ **do** // filter training instances and learn a pairwise classifier
11. $\quad\quad\quad L_{\mathbf{x}} \leftarrow L[\mathbf{x}]$
12. $\quad\quad\quad L'_{\mathbf{x}} \leftarrow [\,]$
13. $\quad\quad\quad$ **for all** $\lambda \in \mathscr{L}_v$ **do** // build L'_v as extension of L to treat virtual labels
14. $\quad\quad\quad\quad$ **if** $\lambda \in V$ **then**
15. $\quad\quad\quad\quad\quad L'_{\mathbf{x}}[\lambda] \leftarrow \lambda$
16. $\quad\quad\quad\quad$ **else**
17. $\quad\quad\quad\quad\quad L'_{\mathbf{x}}[\lambda] \leftarrow L_{\mathbf{x}}[\lambda]$
18. $\quad\quad\quad\quad$ **end if**
19. $\quad\quad\quad$ **end for**
20. $\quad\quad\quad$ **if** $L'_{\mathbf{x}}(\lambda_i) \neq L'_{\mathbf{x}}(\lambda_j)$ **then** // filter out instances not distinguishing between λ_i and λ_j
21. $\quad\quad\quad\quad [\mathbf{x}]_{(\lambda_i,\lambda_j)} \leftarrow \mathbf{x}$
22. $\quad\quad\quad\quad [\mathbb{X}]_{(\lambda_i,\lambda_j)} \leftarrow [\mathbb{X}]_{(\lambda_i,\lambda_j)} \cup \left\{[\mathbf{x}]_{(\lambda_i,\lambda_j)}\right\}$// extend training set of pairwise classifier with instance $\mathbf{x}$
23. $\quad\quad\quad\quad \left[P_{\mathbf{x}}\right]_{(\lambda_i,\lambda_j)} \leftarrow [p]_{(\lambda_i,\lambda_j)}\left(L'_{\mathbf{x}}\right)$ // detect which label is more preferred in this instance
24. $\quad\quad\quad\quad [P]_{(\lambda_i,\lambda_j)}(\mathbf{x}) \leftarrow \left[P_{\mathbf{x}}\right]_{(\lambda_i,\lambda_j)}$// label instance
25. $\quad\quad\quad$ **end if**
26. $\quad\quad$ **end for**
27. $\quad\quad$ **if** $[\mathbb{X}]_{(\lambda_i,\lambda_j)} \neq \{\}$ **then** // filter out classifiers with an empty training set
28. $\quad\quad\quad$ train $[H]_{(\lambda_i,\lambda_j)}$ on $[\mathbb{X}]_{(\lambda_i,\lambda_j)}, [P]_{(\lambda_i,\lambda_j)}$ // train a pairwise classifier
29. $\quad\quad\quad H \leftarrow H \cup \left\{[H]_{(\lambda_i,\lambda_j)}\right\}$
30. $\quad\quad$ **end if**
31. $\quad$ **end for**
32. **end for**
33. **return** H

Algorithm 4.2.2 Classification_Direct

Input: $\mathbf{x}, \mathscr{L}, H$
Output: $\hat{P}_{\mathbf{x}}$

1. $f \leftarrow []$
2. **for all** $[H]_{(\lambda_i,\lambda_j)} \in H$ **do** // sum up the votes of the classifiers
3. **if** $[H]_{(\lambda_i,\lambda_j)}(\mathbf{x}) = 1$ **then**
4. $f\left[\lambda_j\right] \leftarrow f\left[\lambda_j\right] + 1$
5. **else**
6. $f\left[\lambda_i\right] \leftarrow f\left[\lambda_i\right] + 1$
7. **end if**
8. **end for**
9. $\hat{L_{\mathbf{x}}} = []$
10. **for all** $\lambda \in \mathscr{L}$ **do** // initialize prediction with lowest grade for all labels assumed
11. $\hat{L_{\mathbf{x}}}[\lambda] \leftarrow m_0$
12. **end for**
13. **for** $i = 0$ **to** $|V| - 1$ **do** // compare scores of the labels with the scores of all virtual labels
14. $v \leftarrow V[i]$
15. **for all** $\lambda \in \mathscr{L}$ **do**
16. **if** $f[\lambda] > f[v]$ **then** // if the score of the label is higher than the one of the virtual label, the labels grade is predicted being higher than the virtual label
17. $\hat{L_{\mathbf{x}}}[\lambda] \leftarrow m_{i+1}$
18. **end if**
19. **end for**
20. **end for**
21. **return** $\hat{L_{\mathbf{x}}}$

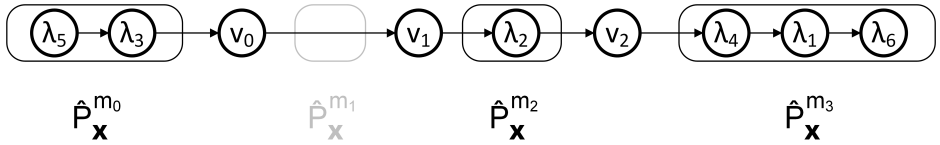
$l_v = l + |\mathbb{V}|$ being the, due to the enrichment of $\mathscr{L}$, increased number of pairwise classifiers. In greater detail to compare the normal labels $O(e \cdot l)$ pairwise classifiers have to be trained with e being the mean number of labels with the same grade per instance. Additionally the classifiers to compare the normal labels have to be trained. These are always learned. So their number is $O(v \cdot l)$ with $v = |\mathbb{V}|$. Like stated above no classifiers have to be learned to compare the virtual labels. So no more classifiers have to be added. Because for each classifier has to be decided which instances are used to train the classifier the classification complexity is $O(e \cdot l \cdot n + v \cdot l \cdot n) = O((e + v) \cdot l \cdot n)$ which is some smaller upper boundary to $O\left(d \cdot l_v \cdot n\right)$ with $d = l_v$.
The complexity of the classification step grows similar to the training step to $O\left(l_v^2\right)$.



(a) The pairwise comparisons between the labels of a graded multilabel classification problem

(b) The additional pairwise comparisons with the virtual label added by the direct approach to the graded multilabel classification problem

(c) All pairwise comparisons done in the direct approach



(d) An example of the partitioning of a ranking in the direct approach

Figure 4.1: An example of the prediction of the direct approach to generalize the calibrated label ranking

4.3 Generalizing the calibrated label ranking using horizontal reduction and dependent level cuts

Because the direct generalization approach seems to have some drawbacks, a different generalization of the calibrated label ranking is introduced in the following. To take the correlation of the grades into account the single binary classifiers not only have to produce some general preference which label is more preferred but also have to take into accord the grades of the compared labels. Or roughly spoken that some pair of labels λ_i, λ_j the label λ_i is more relevant by looking at a distinct grade m.
One approach to do this is using the horizontal reduction to produce $n = |M| - 1$ multilabel classification problems. Then solve these individual problems by calibrated label ranking and aggregate the rankings. This was already mentioned by [3] as not trivial because the individual rankings may differ. So the position of a label may be different in the different rankings. In the worst case the order of the virtual labels in the global ranking could be wrong which would be a violation of the monotonicity condition. [3] proposed the sum up the score functions $\left[f\right]_{m_{k-1}|m_k}$ of the horizontal level cuts $m_{k-1}\left|m_k\right.$ to gain a global scoring function

$$f : (\mathbf{x}, \lambda) \mapsto \sum_{k=1}^{|\mathbb{M}|-1} \left[f\right]_{m_{k-1}|m_k} (\mathbf{x}, \lambda).$$

But this setting would not fit well in the case of virtual labels because they are only part of the ranking of their individual label cuts. So the global score of a virtual label v_i is equal to the score of the level cut

$$f\left(\mathbf{x}, v_i\right) = \left[f(\mathbf{x}, v_i)\right]_{m_i|m_{i+1}}.$$

The global scores of the virtual labels also cannot be gained through using the scores of the 'lower' virtual labels as substitute when summing up

$$f : \mathbb{X}, V \to \mathbb{R}, \left(\mathbf{x}, v_j\right) \mapsto \sum_{k}^{|\mathbb{M}|-1} \sum_{\substack{v_i \\ i \leq j}}^{\mathbb{V}} \left[f\left(\mathbf{x}, v_i\right)\right]_{m_{k-1}|m_k}$$

But this does not solve the problem. The score of the calibration label of some level cut $m_{k-1}\left|m_k\right.$ for an instance $\mathbf{x}$ is smaller than the score of all labels being relevant with a higher grade because the comparing classifier has to vote for the relevant label. If the votes of the lower virtual labels are summed up with the votes of the higher virtual labels to get a score for them this distance in votes is accumulated each level cut. So a virtual label representing a higher level cut is less scored by the scoring function than a normal label which has to be placed lower in the global ranking than the virtual label.
The solution to this problem is to reinterpret the meaning of the individual level cuts. Instead of learning classifiers to predict if some label has a distinct grade or not, the classifiers predict the level cut between two following grades. More precisely which of the two compared labels has a higher probability to be relevant with a distinct grade or a higher one. So there are also learned pairwise classifiers for $n = |\mathbb{M}| - 1$ different level cuts, but there is one projection function $\left[p\right]^{m_i}_{(\lambda_i,\lambda_j)}$ per level cut. This function is generalized from the projection function of the

calibrated label ranking to separate the labels according to the individual level cut.
Because the scores of the virtual labels with a lower grade cannot easily be summed up to gain the score of a virtual label with higher grade, the label set has to be enriched similar to the direct generalization approach. So the label set for learning the pairwise classifiers is like in the direct approach

$$\mathscr{L}_V = \mathscr{L} \cup V.$$

In difference to the direct approach the virtual labels are not all treated as calibration labels but normal labels with higher or lower grade out of the point of view of an pairwise classifier separating a level cut. Only the virtual label representing this level cut is projected like in the direct generalization. For the level cut $m_{k-1} \left| m_k \right.$ the calibration label would be the virtual label v_{k-1}. This is done because the classifier only predicts which of the compared labels is more relevant with a higher grade than the virtual label representing this level cut. So all other virtual labels do not harm the classifiers of this level cut except they are one of the compared labels. But then they can be used as labels with a higher or lower grade than the virtual label of this level cut in terms of the order $\prec_v$.
This way the virtual labels representing the other level cuts are scored by the classifiers of this level cut. So the projection for the level cut from $m_{k-1} \left| m_k \right.$ would be

$$\left[p\right]_{(\lambda_i,\lambda j)}^{m_{k-1}|m_k} : L'_{\mathbf{x}} \mapsto \begin{cases} 1 & \text{if } \lambda_i = v_{k-1} \wedge L'_{\mathbf{x}}\left(\lambda_j\right) \prec_v m_k \\ 1 & \text{if } \lambda_j = v_{k-1} \wedge m_{k-1} \prec_v L'_{\mathbf{x}}\left(\lambda_i\right) \\ 1 & \text{if } \lambda_i, \lambda_j \neq v_{k-1} \wedge m_{k-1} \prec_v L'_{\mathbf{x}}\left(\lambda_i\right) \wedge L'_{\mathbf{x}}\left(\lambda_j\right) \prec_v m_k \\ 0 & \text{if } \lambda_j = v_{k-1} \wedge L'_{\mathbf{x}}\left(\lambda_i\right) \prec_v m_k \\ 0 & \text{if } \lambda_i = v_{k-1} \wedge m_{k-1} \prec_v L'_{\mathbf{x}}\left(\lambda_j\right) \\ 0 & \text{if } \lambda_i, \lambda j \neq v_{k-1} \wedge m_{k-1} \prec_v L'_{\mathbf{x}}\left(\lambda_j\right) \wedge L'_{\mathbf{x}}\left(\lambda_i\right) \prec_v m_k \end{cases}.$$

For a better understanding of the above formula the cases will now be discussed in detail. Note that the first three cases differ from the other three cases only in interchanging the labels λ_i and λ_j. So only three cases have to be discussed. First of all the first three cases denote that λ_i is labeled with a higher grade than λ_j and the other three cases denote the complementary. The first case is that λ_i is the calibration label and λ_j is labeled with a lower grade for $\mathbf{x}$ than the calibration label. The second case is that λ_j is the calibration label. and λ_i is labeled higher than the calibration label. So the first two cases are handling comparisons of a label with the calibration label of the level cut. Remember only one of the virtual labels is the calibration label of a level cut. The third case handles the comparison of two labels without one being the calibration label. It is that case that λ_i has a higher grade than the calibration label and λ_j has a lower grade than the calibration label. This way the individual horizontal level cuts are not independent but depend on the virtual labels, which are not the calibration label of this level cut. Finally a score function to get a global ranking is needed. [3] stated that the aggregation of the scores of the individual level cuts cannot be easily done, because neither the individual rankings of the level cuts cannot be assumed identical nor the scales of the scoring functions have to be comparable. But in this special setting this can be assumed to be possible under the assumption of the pairwise classifiers working correctly. So the rankings of the level cuts are identical and each score function produces some score for all labels in $\mathscr{L}_V$ the score functions have the same range of values from $min\left[f\right]_{m_{k-1}|m_k} = 0$ to $max\left[f\right]_{m_{k-1}|m_k} = |\mathscr{L}_V| - 1$. So the values cannot be simply used to gain the global ranking. But using the circumstance that the

rankings are identical, the values can obviously summed up to gain the global ranking.
This can easily be shown by the following example. If the virtual label v_0 separates the labels with grade m_0 from the ones with a higher label, its score in the corresponding level cut is the sum of the labels with grade m_0. Because of the assumption that all classifiers work correctly the classifiers $H_{(\lambda,v)}$ with $\lambda \neq v_0$ and $L_{\mathbf{x}}(\lambda) = m_0$ vote for the virtual label v_0. Also the classifiers $H_{(\lambda,\lambda')}$ with $\lambda, \lambda' \neq v_0$, $L_{\mathbf{x}}(\lambda) = m_0$ and $L_{\mathbf{x}}(\lambda') \geq m_0$ will vote for λ'. Additionally each classifier of this level cut which compares with the virtual label v_0 with a label λ' with a higher grade prefers the other label. So the score of the virtual label is higher than the score of the labels with grade m_0 but lower than the score of a label with a higher grade than m_0

$$f(\mathbf{x}, \lambda) < f(\mathbf{x}, v_0) < f(\mathbf{x}, \lambda') \text{ with } \forall \lambda \in \mathscr{L} L_{\mathbf{x}}(\lambda) = m_0, \forall \lambda' \in \mathscr{L} L_{\mathbf{x}}(\lambda') \geq m_1.$$

The votes of the classifiers comparing the labels with the grade m_0 can be determined irrelevant because the number of votes a single label can get is smaller than the number of votes the virtual label gets. Looking at the next level cut from grade m_1 to m_2 neither the labels with grade m_0 nor the ones with grade m_1 or the virtual label v_0 get a vote of a classifier of this level cut comparing them with a higher grade normal or virtual label. So the virtual label v_1 gets a higher score than these labels. By summing up the scores of the two level cuts the labels with grade m_1 can be isolated because their score is higher than the one of the virtual label v_0 but lower than the one of the virtual label v_1. So the ranking seems to be separated in a well suited manner.
The identical ordered rankings of the different level cuts can be assumed because for the set of instances $\left[\mathbb{X}_{pos}\right]_{(\lambda i,\lambda_j)}^{m_{k-1}|m_k}$ from $\mathbb{X}_{train}$ with $\left[p\right]_{(\lambda_i,\lambda j)}^{m_{k-1}|m_k} = 1$ used to train the classifiers $[H]_{(\lambda_i,\lambda j)}^{m_{k-1}|m_k}$ of the level cut $m_{k-1}|m_k$, due to the form of the projection function, always holds

$$\left[\mathbb{X}_{pos}\right]_{(\lambda i,\lambda_j)}^{m_{k-1}|m_k} \subseteq \left[\mathbb{X}_{pos}\right]_{(\lambda i,\lambda_j)}^{m_{l-1}|m_{l+1}} \text{ with } k < l.$$

So the assumption of the correct working binary classifiers assures an identical ordering over the different level cuts.

4.3.1 Discussion

Unfortunately there are effects aware that prevent this theoretical theory from being full applicable. On the one hand the binary classifiers do not always work correct. On the other hand there are also some other problems to this theoretical foundation of this way to gain the global score.
By looking at the data the virtual labels are not relevant to any instance in the data set. So all virtual labels, which are lower in the order of V than the calibration label of a distinct level cut, are always preferred less than the compared labels. This is aware even if the label has a lower grade than the virtual label, because the classifier to compare these two labels only gets to see instances with the normal label having a higher grade than the virtual label. Each other instances are stated irrelevant because the grade of both the normal and the virtual label are beneath the calibration label of the level cut. As a result of this the predicted score of the normal labels is tendential too high. So the predicted grade is probably also too high.

A second effect on the predicted ranking is that looking at a distinct level cut and a binary classifier comparing a virtual label higher in the order $\prec_v$ than the calibration label and a normal label, this classifier will always prefer the virtual label above the normal because of the same reason as the lower virtual labels were preferred less. The only instances with a difference in grade toward the level cut between the two labels are ones with the normal label having a lower grade than the virtual label. So the normal labels are scored lower than needed and so a too low grade may be predicted.
The consequence of the two effects is that the approach tends to predict grades too near to the middle of the range of grades when a grade from the edges of the range has to be predicted. Also the two effects more and more cancel out each other the more the real grade of a label is near two the middle of the range of grades. So in conclusion the predicted grades are tendential from the middle of the range of $\mathbb{M}$. This effect also grows with the size of $\mathbb{M}$. Obviously the effects only take place, if there are instances aware in the training set, which produce these effects. Otherwise the depending classifiers are discarded.
To assure a better understanding of the two negative effects an example is given. Assume the setting $\mathbf{x} \in \mathbb{X}_{train}$ and $\mathbf{x}$ shows up during the classification phase and has to be classified by the approach. So the instance should be classified correctly by the classifier. The setting of the problem is

$$\begin{aligned}\mathbb{M} &= \{m_0, m_1, m_2, m_3\} \\ \mathscr{L} &= \{a, b, c, d\} \\ \mathbb{V} &= \{v_0, v_1, v_2\} \\ L_{\mathbf{x}} &= \{(a, m_0), (b, m_1), (c, m_2), (d, m_3)\}\end{aligned} .$$

For a better understanding the real classification $L_{\mathbf{x}}$ of $\mathbf{x}$ is visualized in Figure 4.2.

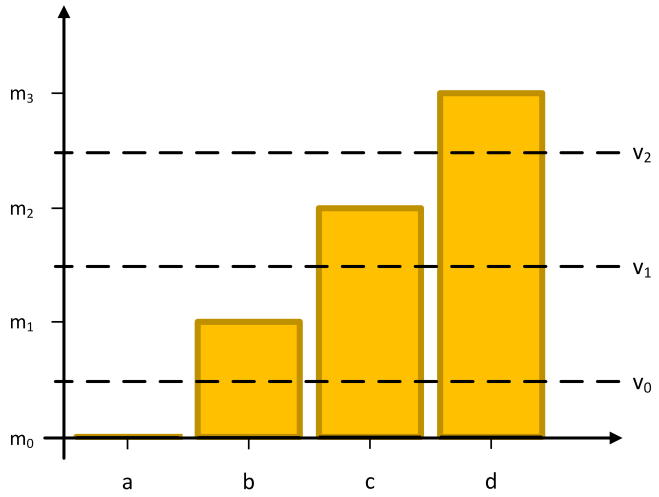


Figure 4.2: An example of the real classification $L_{\mathbf{x}}$ of an instance $\mathbf{x} \in \mathbb{X}$. The grades are printed on the y-axes and the labels on the y-axes. The virtual labels partitioning the grades are shown as dashed lines.

The classifiers of the level cut $m_0 \,|\, m_1$ would vote in the following manner

$$
\begin{aligned}
[H]^{m_0|m_1}_{(a,\lambda)} \text{ with } \lambda \in \mathscr{L} \setminus \{a\} &\rightarrow \text{vote for } \lambda \\
[H]^{m_0|m_1}_{(\lambda,\lambda')} \text{ with } \lambda, \lambda' \in \mathscr{L} \setminus \{a\} \wedge \lambda \neq \lambda' &\rightarrow \text{vote either for } \lambda \text{ or } \lambda', \text{ but the distinct vote is} \\
&\quad \text{unpredictable, because there is no difference in} \\
&\quad \text{grade towards the level cut aware} \\
[H]^{m_0|m_1}_{(\lambda,v)} \text{ with } \lambda \in \mathscr{L} \setminus \{a\} \wedge v \in V \setminus \{v_0\} &\rightarrow \text{vote for } v, \text{ because the second effect takes} \\
&\quad \text{place} \\
[H]^{m_0|m_1}_{(a,v_0)} &\rightarrow \text{votes for } v_0 \\
[H]^{m_0|m_1}_{(v_0,\lambda)} \text{ with } \lambda \in \mathscr{L} \setminus \{a\} &\rightarrow \text{vote for } \lambda \\
[H]^{m_0|m_1}_{(v_0,v)} \text{ with } v \in V \setminus \{v_0\} &\rightarrow \text{vote for } v
\end{aligned}
$$

So the resulting scores of the level cut are

$$[f]^{m_0|m_1} = \{(a,0),(b,2(+2)),(c,2(+2)),(d,2(+2)),(v_0,1),(v_1,2(+3)),(v_2,2(+3))\}$$

where $(+i)$ means that the votes could be extended either by one of the two negative effects or a classifier with doing an unpredictable vote. Note that the unpredictable votes above are counted at both labels each.
Looking at the next level cut $m_1 \,|\, m_2$ its classifiers would make the following votes

$$
\begin{aligned}
[H]^{m_1|m_2}_{(\lambda,\lambda')} \text{ with } \lambda \in \{a,b\} \wedge \lambda' \in \{c,d\} &\rightarrow \text{vote for } \lambda' \\
[H]^{m_1|m_2}_{(\lambda,\lambda')} \text{ with } \lambda, \lambda' \in \mathscr{L} \setminus \{a,b\} \wedge \lambda \neq \lambda' &\rightarrow \text{vote either for } \lambda \text{ or } \lambda', \text{ but the distinct vote is} \\
&\quad \text{unpredictable, because there is no difference in} \\
&\quad \text{grade towards the level cut aware} \\
[H]^{m_1|m_2}_{(\lambda,\lambda')} \text{ with } \lambda, \lambda' \in \mathscr{L} \setminus \{c,d\} \wedge \lambda \neq \lambda' &\rightarrow \text{vote either for } \lambda \text{ or } \lambda', \text{ but the distinct vote is} \\
&\quad \text{unpredictable, because there is no difference in} \\
&\quad \text{grade towards the level cut aware} \\
[H]^{m_1|m_2}_{(\lambda,v_0)} \text{ with } \lambda \in \{a,b\} &\rightarrow \text{vote for } \lambda, \text{ because the first effect takes place} \\
[H]^{m_1|m_2}_{(\lambda,v_0)} \text{ with } \lambda \in \{c,d\} &\rightarrow \text{vote for } \lambda \\
[H]^{m_1|m_2}_{(\lambda,v_1)} \text{ with } \lambda \in \{a,b\} &\rightarrow \text{votes for } v_1 \\
[H]^{m_1|m_2}_{(\lambda,v_1)} \text{ with } \lambda \in \{c,d\} &\rightarrow \text{votes for } \lambda \\
[H]^{m_1|m_{12}}_{(\lambda,v)} \text{ with } \lambda \in \{a,b\} \wedge v \in V &\rightarrow \text{vote for } v \\
[H]^{m_1|m_{12}}_{(\lambda,v)} \text{ with } \lambda \in \{c,d\} \wedge v \in V &\rightarrow \text{vote for } v, \text{ because the second effect takes} \\
&\quad \text{place} \\
[H]^{m_1|m_2}_{(v,v_2)} \text{ with } v \in V \setminus \{v_2\} &\rightarrow \text{vote for } v_2 \\
[H]^{m_1|m_2}_{(v_0,v_1)} &\rightarrow \text{vote for } v_1
\end{aligned}
$$

The scores of this level cut are

$$[f]^{m_1|m_2} = \{(a,0(+2)),(b,0(+2)),(c,4(+1)),(d,4(+1)),(v_0,0),(v_1,3),(v_2,4(+2))\}.$$

With the scores from the level cut $m_2 \,|\, m_3$

$$[f]^{m_2|m_3} = \{(a,0(+4)),(b,0(+4)),(c,0(+4)),(d,6),(v_0,0),(v_1,0),(v_2,5)\}$$

the global scores of the labels are

$$f = \{(a,0(+6)),(b,2(+8)),(c,6(+7)),(d,12(+3)),(v_0,1),(v_1,5(+3)),(v_2,11(+5))\}.$$

If the unpredictable votes are assumed to be normally distributed over the depending labels, the following scores are mean scores of the labels

$$f_{mean} = \{(a,0(+4.5)),(b,2(+5.5)),(c,6(+4.5)),(d,12(+1.5)),(v_0,1),(v_1,5(+3)),(v_2,11(+5))\}.$$

So the predicted ranking is

$$v_0 \prec_{\mathbf{x}} a \prec_{\mathbf{x}} b \prec_{\mathbf{x}} v_1 \prec_{\mathbf{x}} c \prec_{\mathbf{x}} d \prec_{\mathbf{x}} v_2$$

and the prediction of the classifier is

$$\hat{L}_{\mathbf{x}} = \{(a,m_1),(b,m_1),(c,m_2),(d,m_2)\}$$

Note that the order of the normal labels is predicted correctly, but the position of the virtual labels is not estimated correctly. So the failure caused by the two effects is a failure prone calibration of the virtual labels with a tendency to grow the distance between these in the predicted ranking. Obviously the positions of two neighboring normal labels can be interchanged, if the unpredictable votes tend to one of the labels.
Interestingly one way to reduce the effects on the score of the higher ordered virtual labels is to change to discard all classifiers $[H]^{m_{k-1}|m_k}_{(v_i,v_{k-1})}$ with $i < k-1$. This arbitrary seeming change was found by accident and showed quite good results in practice. The predicted grades of the normal labels converged to the real grades and so the overall error done was minimized, leaving the problem of the normal labels with a low real grade being predicted with a too high grade. But this will be discussed later when the experimental results are discussed.
The consequences to the global score with normally distributed unpredictable votes done by this filtering of the classifiers would be

$$f_{mean} = \{(a,0(+4.5)),(b,2(+5.5)),(c,6(+4.5)),(d,12(+1.5)),(v_0,1),(v_1,4(+3)),(v_2,9(+5))\}.$$

So the changed ranking is

$$v_0 \prec_{\mathbf{x}} a \prec_{\mathbf{x}} v_1 \prec_{\mathbf{x}} b \prec_{\mathbf{x}} c \prec_{\mathbf{x}} d \prec_{\mathbf{x}} v_2.$$

Note that the missclassification of the labels b and d relies only on 0.5 votes done by an unpredictable vote. So their missclassification is dependent on the decision of these classifiers.

4.3.2 Implementation of the solving approach

The implementation of the training phase of this approach can be found in algorithm 4.3.1. Note that the filtering of the classifiers $[H]^{m_{k-1}|m_k}_{(v_i,v_{k-1})}$ with $i < k-1$ mentioned above is integrated in the condition in line 12 of the reduction method algorithm 4.3.2.
Interestingly the changes to the direct approach do not harm the classification phase and so the implementation is algorithm 4.2.2.

Algorithm 4.3.1 Training_DHR

Input: $\mathscr{L}, \mathbb{M}, \mathbb{X}_{train}, L$
Output: H

1. $n \leftarrow |\mathbb{M}| - 2$
2. $V = \{v_0, \ldots, v_n\}$
3. $\mathscr{L}_v \leftarrow \mathscr{L} \cup V$ // extend label set with virtual labels
4. $H \leftarrow \{\}$
5. **for** $k \leftarrow 1$ **to** $n+1$ **do** // learn pairwise classifiers for each level cut
6. $\quad m_{k-1} \leftarrow \mathbb{M}[k-1]$
7. $\quad m_k \leftarrow \mathbb{M}[k]$
8. $\quad v \leftarrow V[k]$
9. $\quad$ **for** $i \leftarrow 1$ **to** $\left|\mathscr{L}_v\right| - 1$ **do**
10. $\quad\quad \lambda_i \leftarrow \mathscr{L}_v[i]$
11. $\quad\quad$ **for** $j \leftarrow i$ **to** $\left|\mathscr{L}_v\right|$ **do** // lean pairwise comparison of λ_i, λ_j
12. $\quad\quad\quad \lambda_j \leftarrow \mathscr{L}_v\left[j\right]$
13. $\quad\quad\quad$ **for all** $\mathbf{x} \in \mathbb{X}_{train}$ **do** // filter training instances and learn a pairwise classifier
14. $\quad\quad\quad\quad [\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}, [P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow$Reduction_DHR$\left(\mathbf{x}, \lambda_i, \lambda_j, m_{k-1}, m_k, v, L\right)$
15. $\quad\quad\quad$ **end for**
16. $\quad\quad\quad$ **if** $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \neq \{\}$ **then** // filter out classifiers with an empty training set
17. $\quad\quad\quad\quad$ train $[H]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}$ on $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}, [P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}$ // train a pairwise classifier
18. $\quad\quad\quad\quad H \leftarrow H \cup \left\{[H]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}\right\}$
19. $\quad\quad\quad$ **end if**
20. $\quad\quad$ **end for**
21. $\quad$ **end for**
22. **end for**
23. **return** H

4.3.3 Complexity of the solving approach

The complexity of the training of this approach is obviously higher than the one of the direct approach because $|V|$ different horizontal reductions have to be done. So the complexity of training the complete ensemble of classifiers raises to $O\left(l_v^2 \cdot n \cdot |V|\right)$.
The additional classifiers have to be taken into account when computing the complexity of the classification phase. So the complexity is $O\left(l_v^2 \cdot |V|^2\right)$.

Algorithm 4.3.2 Reduction_DHR

Input: $\mathbf{x}, \lambda_i, \lambda j, m_{k-1}, m_k, v, L$

Output: $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}, [P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}$

1. $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
2. $[P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow \{\}$
3. $L_{\mathbf{x}} \leftarrow L\,[\mathbf{x}]$
4. $L'_{\mathbf{x}} \leftarrow []$
5. **for all** $\lambda \in \mathscr{L}_v$ **do** // build L'_v as extension of L to treat virtual labels
6. **if** $\lambda \in V$ **then**
7. $L'_{\mathbf{x}}[\lambda] \leftarrow \lambda$
8. **else**
9. $L'_{\mathbf{x}}[\lambda] \leftarrow L_{\mathbf{x}}[\lambda]$
10. **end if**
11. **end for**
12. **if** $\left(L'_{\mathbf{x}}(\lambda_i) \prec_v v \wedge v \prec_v L'_{\mathbf{x}}(\lambda_j)\right) \vee \left(L'_{\mathbf{x}}(\lambda_j) \prec_v v \wedge v \prec_v L'_{\mathbf{x}}(\lambda_i)\right) \vee \left(\lambda_i = v \wedge v \prec_v L'_{\mathbf{x}}(\lambda_j)\right) \vee \left(\lambda_j = v \wedge v \prec_v L'_{\mathbf{x}}(\lambda_i)\right)$ **then** // filter out instances not distinguishing between λ_i and λ_j and apply the tweak to filter out the classifiers comparing the calibration label with lower virtual labels by filtering out all instances for them
13. $[\mathbf{x}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow \mathbf{x}$
14. $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow [\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \cup \left\{[\mathbf{x}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}\right\}$// extend training set of pairwise classifier with instance $\mathbf{x}$
15. $\left[P_{\mathbf{x}}\right]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)} \leftarrow [p]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}\left(L'_{\mathbf{x}}\right)$// detect which label is more preferred in this instance
16. $[P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}(\mathbf{x}) \leftarrow \left[P_{\mathbf{x}}\right]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}$ // label instance
17. **end if**
18. **return** $[\mathbb{X}]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}, [P]^{m_{k-1}|m_k}_{(\lambda_i,\lambda_j)}$

4.4 An approach using horizontal reduction and independent level cuts

The problem of the last approach was that the enhancement of the label set used for the different horizontal problems had some drawback on the combined global ranking of the labels. The most obvious way to solve this is to use complete independent horizontal problems and combine the predictions afterwards. So for each level cut $m_i|m_{i+1}$ is learned some independent calibrated label ranking. To achieve this the grade assignment to the labels has to be projected to the individual multilabel classification problems. Thereby a different projection function has to be used for each level cut $m_{l-1}\left|m_{l+1}\right.$. The binary decision, if some label is relevant or not, therefore has to be equal to the statement, that a label has a lower grade than a distinct grad or a higher one. Obviously this threshold is the virtual label used as calibration label in the calibrated label ranking problem the original problem was reduced to. So the projection functions have the form

$$[H]_{m_{l-1}|m_{l+1}} : L_{\mathbf{x}} \mapsto \left\{\lambda \in \mathscr{L} \left| m_{k-1} \prec L_{\mathbf{x}}(\lambda)\right.\right\}.$$

Because, like stated above, the scores of the individual level cuts cannot easily be recombined, some aggregation function has to be found to gain a graded classification. Out of the problems proposed when discussing the last solving approach to the graded multilabel classification the summation of the scores is not preferable. So like in the approaches of [3] the highest of a level cut depending classifier predicted grade is used as global prediction with the above stated drawbacks. In this special settings they should not be as intense as in the approaches from [3], because the wrong prediction of a single classifier can only produce a minimal increment or decrement to the score of a label in one of the horizontal problems. Even if the false classification is done in the highest level cut, it only turns into a wrong result, if the score of the label is equal to the score of the virtual label. But in calibrated label ranking this is not possible, if only one classifier does a wrong classification. So at least two binary classifiers have to make a missclassification. As consequence the calibrated label ranking seems to be more robust to missclassification than the binary relevance and the IBLR-ML approach.

4.4.1 Implementation of the solving approach

Because this approach devides the overall problem into serveral calibrated label ranking problems, the training phase implementation in algorithm 4.4.1 uses the training phase implementation algorithm 4.1.1 of the calibrated label ranking.
Also the implementation of the classification phase of this approach algorithm 4.4.2 relies on the implementation of the classifying phase algorithm 4.1.2 of the calibrated label ranking.

4.4.2 Complexity of the solving approach

Because this solving approach does not enhance the label set with the whole set of virtual labels but only one virtual label each, the overall training complexity of this approach is $|V|$ times the complexity of a calibrated label ranking. In other words it is $O\left(l^2 \cdot n \cdot |V|\right)$.
The complexity of the classification step grows in the same way to $O\left(l^2 \cdot |V|\right)$.

Algorithm 4.4.1 Reduction_IHR

Input: $\mathscr{L}, \mathbb{X}_{train}, \mathbb{M}, L$
Output: H

1. $H \leftarrow \{\}$
2. **for** $i \leftarrow 1$ **to** $|\mathbb{M}| - 1$ **do** // learn independent calibrated label ranking problems for each level cut
3. $\quad m_i \leftarrow \mathbb{M}[i]$
4. $\quad m_{i+1} \leftarrow \mathbb{M}[i+1]$
5. $\quad P \leftarrow \{\}$
6. $\quad$ **for all** $\mathbf{x} \in \mathbb{X}_{train}$ **do**
7. $\quad\quad L_{\mathbf{x}} \leftarrow L[\mathbf{x}]$
8. $\quad\quad P_{\mathbf{x}} \leftarrow [p]_{m_i|m_{i+1}}(L_{\mathbf{x}})$ // detect which labels are preferred with a grade higher than the level cut
9. $\quad\quad P(\mathbf{x}) \leftarrow P_{\mathbf{x}}$ // label instances
10. $\quad$ **end for**
11. $\quad [H]_{m_i|m_{i+1}} \leftarrow$ Training_CLR$(\mathscr{L}, \mathbb{X}_{train}, P)$ // train multiple pairwise classifiers using calibrated label ranking
12. $\quad H \leftarrow H \cup \left\{[H]_{m_i|m_{i+1}}\right\}$
13. **end for**
14. **return** H

Algorithm 4.4.2 Classify_IHR

Input: $\mathbf{x}, \mathbb{H}, \mathscr{L}$
Output: $\hat{L}_{\mathbf{x}}$

1. $P \leftarrow \{\}$
2. **for all** $[H]_{m_i|m_{i+1}} \in H$ **do**
3. $\quad \left[\hat{L}_{\mathbf{x}}\right]_{m_i|m_{i+1}} \leftarrow$ Classification_CLR$\left(\mathbf{x}, \mathscr{L}, [H]_{m_i|m_{i+1}}\right)$ // use predictions of the calibrated label ranking
4. **end for**
5. $\hat{L}_{\mathbf{x}} \leftarrow []$
6. **for all** $\lambda \in \mathscr{L}$ **do**
7. $\quad \hat{L}_{\mathbf{x}}[\lambda] \leftarrow \max\left[\hat{L}_{\mathbf{x}}\right]_{m_i|m_{i+1}}[\lambda]$ // use the highest voted grade as prediction of the classifier ensemble for grade λ
8. **end for**
9. **return** $\hat{L}_{\mathbf{x}}$

5 Evaluation

After the different approaches to the graded multilabel classification problem are stated in the last chapters the quality of the predicition of the different classification problems is the topic of this chapter. First several losses are proposed to gain some measurements to the different classifiers. Afterwards the experiment to compare the classifying approaches and its result is discussed.

5.1 Losses

To evaluate the quality of a distinct classification approach it is necessary to use appropriate measurements determining the classification errors. This measurements are called loss functions. For the graded multilabel classification problem [3] generalized several of the losses of the multilabel classification. Because I used these measures in the evaluation of my classification approaches, I will discuss the measures and the kind of information they propose in short. In supervised learning the dataset of instances $\mathbf{x} \in \mathbb{X}_{known}$ with known classification is separated into two subsets. The first already above mentioned one is the training set which is used to train the classifier. The second one is the test set. For each instance $\mathbf{x} \in \mathbb{X}_{test}$ the classifier has to predict a classification. Afterwards the losses are calculated for each prediction. The losses of each individual instance are averaged to determine the mean loss of the classifier on the test set. For simplicity the loss functions in this chapter are shown for one single instance $\mathbf{x} \in \mathbb{X}_{test}$

5.1.1 Hamming loss

The hamming loss measures how much labels $\lambda \in \mathscr{L}$ are predicted wrong. This means how many false positive and false negative predictions were done. For some instance $\mathbf{x} \in \mathbb{X}$ of a multilabel classification problem this means

$$E_H\left(H\left(\mathbf{x}\right), L_{\mathbf{x}}\right) = \frac{1}{|\mathscr{L}|}\left|H\left(\mathbf{x}\Delta L_{\mathbf{x}}\right)\right|$$

with Δ being the symmetric difference between two sets.
In the graded multilabel classification this loss can be generalized measuring the loss in terms of horizontal or vertical cuts of the label space. [3] showed that both functions are equal to each other. So for simplification the hamming loss is stated the vertical hamming loss

$$E_H^{\star}\left(H\left(\mathbf{x}, \cdot\right), L_{\mathbf{x}}\right) = \frac{\sum_{i=1}^{|\mathscr{L}|} AE\left(H\left(\mathbf{x}, \lambda_i\right), L_{\mathbf{x}}\left(\lambda_i\right)\right)}{\left(|M| - 1\right) \cdot |\mathscr{L}|}$$

with

$$AE : \mathbb{M} \times \mathbb{M} \to \mathbb{N}, \left(m_i, m_j\right) \mapsto |i - j|.$$

So the hamming loss in graded multilabel classification denotes the mean deviation of the predicted label grades to the real ones.

5.1.2 Subset zero-one loss

As a second loss [3] proposed a more strict function. The subset zero-one loss measures the percentage of labels with the wrong grade predicted. In contrary to the hamming loss this loss function does not determine how big the difference in grade between predicted and real grade is. Only the existence of a difference is measured

$$E_{0/1}^{\star}\left(H\left(\mathbf{x},\cdot\right),L_{\mathbf{x}}\right)=\frac{1}{\left|\mathscr{L}\right|}\sum_{i=1}^{\left|\mathscr{L}\right|}\begin{cases}0 & H\left(\mathbf{x},\lambda_i\right)=L_{\mathbf{x}}\left(\lambda_i\right)\\ 1 & H\left(\mathbf{x},\lambda_i\right)\neq L_{\mathbf{x}}\left(\lambda_i\right)\end{cases}.$$

5.1.3 C-index

The C-index is the generalization of the rank loss. The rank loss measures the pairwise classification error of the pairs $\left(\lambda,\lambda'\right)\in P_{\mathbf{x}}\times N_{\mathbf{x}}$. In particular the classification error of a ranking established by some scoring function f is measured. This scoring function can be the scoring function of the calibrated label ranking. The error grows for pairs of label with one or more labels being predicted in the wrong order, because a not relevant label should always have a lower score than a relevant one using a with growing relevance of a label monotone growing scoring function. The rank loss for multilabel classification is

$$E_R\left(f,L_{\mathbf{x}}\right)=\frac{\sum_{\left(\lambda,\lambda'\right)\in P_{\mathbf{x}}\times N_{\mathbf{x}}}S\left(f\left(\mathbf{x},\lambda\right),f\left(\mathbf{x},\lambda'\right)\right)}{\left|P_{\mathbf{x}}\times N_{\mathbf{x}}\right|}$$

with

$$S:\mathbb{R}\times\mathbb{R}\rightarrow\{.\frac{1}{2},1\},\left(x,y\right)\mapsto\begin{cases}0 & \text{if } x<y\\ \frac{1}{2} & \text{if } x=y\\ 1 & \text{otherwise}\end{cases}.$$

To fit the graded case [3] extended this error to measure the pairwise ranking error between a pair of labels out of two different sets

$$P_{\mathbf{x}}^{m_i}=\left\{\lambda\in\mathscr{L}|L_{\mathbf{x}}(\lambda)=m_i\wedge m_i\in\mathbb{M}\right\}$$

of labels with the same grade. So the C-index measures the wrong order of labels with different grade in the ranking predicted with the scoring function

$$E_R^{\star}\left(f,L_{\mathbf{x}}\right)=\frac{\sum_{i<j}\sum_{\left(\lambda,\lambda'\right)\in P_{\mathbf{x}}^{m_i}\times P_{\mathbf{x}}^{m_j}}S\left(f\left(\mathbf{x},\lambda\right),f\left(\mathbf{x},\lambda'\right)\right)}{\sum_{i<j}\left|P_{\mathbf{x}}^{m_i}\times P_{\mathbf{x}}^{m_i}\right|}.$$

If the classification approach does not use a ranking or scoring function, [3] proposed to use the predicted grade of a label as its score. This produces an appropriate ranking above the labels and is used by me when calculating the C-index of the binary relevance approach and the approach using complete independent level cuts in the evaluation of the solving approaches.

5.1.4 One error rank loss

The last loss function stated by [3] is the generalization of the one error. This loss measures if the top ranked label of a ranking is relevant or not.

$$E_{1E}\left(f, L_{\mathbf{x}}\right) = \begin{cases} 0 & \text{if } \left(\arg\max_{\lambda \in \mathscr{L}} f\left(\mathbf{x}, \lambda\right)\right) \in P_{\mathbf{x}} \\ 1 & \text{otherwise} \end{cases}$$

In case of the calibrated label ranking this is an interesting loss function to measure the general suitability of the ranking used to predict multilabel classification problem.
In [3] this loss is generalized to measure if the highest ranked label has the highest possible grade.

$$E^{\star}_{1E}\left(f, L_{\mathbf{x}}\right) = AE\left(max_{m \in \mathbb{M}} m, L_{\mathbf{x}}\left(\arg\max_{\lambda \in \mathscr{L}} f\left(\mathbf{x}, \lambda\right)\right)\right)$$

The drawback of this function is, if in an instance out of the test set has no label with a relevance of the highest possible grade, the one error cannot be zero, even if the classification of the instance is completely correct. In terms of the multilabel classification this case is an instance with no relevant label. This instance would never be assumed correct classified by the one error. This special case is in general not very common in a data set of a multilabel classification problem, but in the data set of a graded multilabel classification problem an instance without a label with maximal grad seems more common. To solve this problem I propose a changed version of the one error comparing the real grade of the highest ranked label with the highest grade of all labels of an instance.

$$E^{\star}_{1E}\left(f, L_{\mathbf{x}}\right) = AE\left(\max L_{\mathbf{x}}\left(\lambda'\right), L_{\mathbf{x}}\left(\arg\max_{\lambda \in \mathscr{L}} f\left(\mathbf{x}, \lambda\right)\right)\right)$$

5.1.5 Calibration loss

When Examining generalizations of the calibrated label ranking a loss function measuring the correct positions of the virtual labels in the predicted ranking becomes necessary because the calibrated label ranking itself has some tendency to underestimate the score of the virtual label in relation to the other labels. So when comparing different sorts of generalizations of the calibrated label ranking the information given by this loss seems crucial.
This concrete loss function measures the quality of the partitioning of predicted global ranking. Therefore the ranking is repartitioned by the real grade partitioning of the instance. In other words the number of labels with a distinct grade is counted and the same number of labels in the ranking is assumed to have this distinct grade. Afterwards the hamming loss of this new 'prediction' $E^{\star}_{H}\left(H'\left(\mathbf{x},\right), L_{\mathbf{x}}\right)$ is calculated. The remaining error is due to ordering failures of the labels done by the scoring function but no errors due to the wrong positioning of the virtual labels is left. So by comparing this new hamming loss with the one calculated based on the original prediction measures $E^{\star}_{H}\left(H\left(\mathbf{x},\right), L_{\mathbf{x}}\right)$ the error done by wrong positioning the virtual labels in the ranking are calculated.

$$E_{C} = \left|E^{\star}_{H}\left(H'\left(\mathbf{x},\right), L_{\mathbf{x}}\right) - E^{\star}_{H}\left(H\left(\mathbf{x},\right), L_{\mathbf{x}}\right)\right|$$

5.2 Experiment

In the following the experiment is done to compare the different solving approaches is discussed. To achieve a good comparison to the results from [3] their experiment was reimplemented by me. The BeLa-E data set used to perform the original experiment was provided to me by Professor Hüllermeier. At first the generation of different classification problems from the data set is discussed. Afterwards the concrete implementation setting of the different classifiers is explained. At least the results of the loss functions from above are presented and discussed.

5.2.1 Data generation

Because there were no data sets of graded multilabel classification known to [3] when implementing their solving approaches, they used a data set from the social psychology called BeLa-E from [1]. This data set consists of 1930 instances each representing a graduate student. Each instance has 50 attributes. The first attribute is the age and the second one the sex of the student. The remaining 48 attributes modeling the importance of a distinct property of their future jobs. Each of these remaining attributes has a grade from '1'(unimportant) to '5'(very important). The data was collected by interviewing the students.
From this data I generated 50 data sets. From the 48 similar graded attributes were a subset of n attributes as labels randomly chosen. The remaining $50 - n$ attributes were used as features of the instances. This is done like in [3] for $n = 5$ and $n = 10$ labels.
In difference to their setup no binary data for a multiabel classification was generated because the focus of the experiment was changed from showing the benefit of formulating graded multilabel classification problem to comparing different solving strategies.

5.2.2 Implementation setup

The different solving approaches, except the IBLR-ML, were implemented by me as part of the LPCforSOS framework[1] (see [14]) of the Knowledge Engineering Group of the Technical University of Darmstadt[2], which is an extension of the Weka framework[3] [9]. As binary base classifier the J48 classifier of the Weka framework is used, which is an implementation of the C45-algorithm [12]. The complete reduction approach was reimplemented by me using horizontal reduction and binary relevance like done in [3]. When calculating the rank losses for the complete reduction approach and the approach using independent level cuts the predicted grade is used as score. On each of the generated data sets each classifier is trained using 10-fold cross validation. Afterwards the results of all 50 data sets with the same number of labels are averaged. In general it is no good practice to average over the results of different data sets, but in this case the individual data sets are generated from the same original data set and like stated by [3] they should be evenly distributed.

[1] http://www.lpcforsos.sf.net
[2] http://www.ke.tu-darmstadt.de
[3] http://www.cs.waikato.ac.nz/ml/weka

5.2.3 Results

The averaged results of the different solving approaches to the graded multilabel classification on the generated BeLaE data set problem are shown in table 5.1. The upper half shows the results of the test sets with $|\mathscr{L}| = 5$ labels and the lower half the results from problems with $|\mathscr{L}| = 10$. For each sort of problems all loss functions discussed above were calculated and the mean value and the standard deviation of each is shown in the table. The values are presented as percentage values and rounded after two decimal places. In the columns the results of the different approaches are listed beginning by the direct approach to generate the calibrated label ranking (see section 4.2, followed by the complete reduction (see section 3.3, the generalization approach using dependent level cuts (see section 4.3 and the one using independent level cuts (see section 4.4. The rank of the approaches to the individual losses is printed in brackets after the mean values and the best result is made bold. Note that like mentioned above in section 5.1.5 the calibration loss can not be calculated for the generalization approach using independent level cuts and the complete reduction.
Comparing the different classifying approaches the complete reduction has the worst results in the losses calculated for it. This seems to be obvious remembering that the interdependencies and the correlation between the different labels and grades are ignored by this approach. So the quality of its prediction relies only on the probability of the base classifier to isolate the different label-grade combinations in the target space of the graded multilabel classification problem. This correlates to the results from [3], where the complete reduction also showed worse results than the horizontal reduction with using the IBLR-ML classifier.
The direct generalization approach however seems to have no big problems in finding the correct ranking of the labels $\lambda \in \mathscr{L}$ when looking at the remaining hamming loss and the small C-index, but the combination of the high ratio of the calibration loss to the Hamming loss and the high subset zero one loss show that the relative high hamming loss done by the approach is caused by its inability to find the correct positions of the virtual labels in the ranking. An explanation to this effect could be its binarization of the grade difference between two labels when learning the pairwise classifiers leading to the problem of not being able to differentiate between the virtual labels during prediction.
The most unforeseen results are the one of the generalizing approach using dependent level cuts mentioning the two negative effects to the calibration of the virtual labels. The fact that it has a small C-index corresponds to the theoretical assumption that the negative effects does not harm the ranking of the normal labels. The relative small hamming and subset zero one loss show that the filtering of the classifiers seems to compensate the negative effects on the positioning of the virtual labels. This assumption is also supported by the smallest one error of all approaches. So like mentioned during the discussion of the approach at least the upper half of the ranking seems to be correct calibrated. Interestingly the calibration loss to hamming loss ratio is more than halved by the doubling of the number of labels to classify. Remembering the idea of the negative effects grow with the size of number of labels to classify, this is not obvious. So either the filtering is the more able to compensate the effects with growing number of labels or it overcompensates the effects and it exists a size of $\mathscr{L}$ upon which the compensation is higher than the effect itself and the quality of the classification of the approach gets worse.
At least the results of the generalization approach using independent level cuts should be discussed. Like expected the hamming loss and the subset zero one loss of this approach are the

best of all compared approaches. This seems to be reasonable, because the individual rankings of the level cuts are not harmed by any negative effects like in the approach with dependent level cuts, but the difference in grade between two grades is modeled by the approach due to the horizontal reduction. The higher one error and c-index compared to the other pairwise classifying approaches at first sight seems to be somewhat suspicious. But remembering the choice of the predicted grade relies on the prediction of the highest predicted grade of the several level cut problems. So the classifier might have a tendency to overestimate the grade of the labels and the one error is some kind of sensitive to such a behavior. The higher C-index may be is resulted in the same effects.
As conclusion to this stands the impression, the more the additional information gained through the combination of the grades and labels is integrated into the approaches the better is the quality of the prediction of a classifier is in the setting of the classifier. With other words the differentiation of the distances between two labels during the training of the classifiers resulted in minimizing the hamming and the subset zero one loss and the integration of the differences in the reaggregation of the reduced classification problems helped to minimize the rank losses C-index, subset zero-one loss and calibration loss. This seems to support the results from [3], which stated that the reduction of a graded multilabel problem to conventional multilabel problem before the classifier leads to worse results than the training and classification in the graded space and then reduce the predictions into the conventional multilabel problems space.

			direct	complete reduction	dependent cuts	independent cuts
$\|\mathscr{L}\| = 5$	Hamming loss	mean	33.97 (4)	28.07 (3)	16.62 (2)	**15.77 (1)**
		deviation	4.53	1.68	0.9	1.22
	One Error loss	mean	8.5 (2)	12.89 (4)	**7.19 (1)**	11.07 (3)
		deviation	1.85	1.83	1.49	1.73
	C-index	mean	20.38 (2)	32.63 (4)	**18.16 (1)**	23.88 (3)
		deviation	3.35	2.85	3.0	3.39
	subset zero-one loss	mean	73.44 (4)	61.27 (3)	57.28 (2)	**51.9 (1)**
		deviation	6.01	3.11	2.06	2.78
	calibration loss	mean	22.97 (2)	-	**7.0 (1)**	-
		deviation	4.39	-	1.08	-
	remaining Hamming loss	mean	11.0 (2)	-	**9.62 (1)**	-
		deviation	0.14	-	0.18	-
$\|\mathscr{L}\| = 10$	Hamming loss	mean	35.44 (4)	27.77 (3)	16.89 (2)	**15.13 (1)**
		deviation	3.06	0.93	0.58	0.77
	One Error loss	mean	8.19 (2)	17.03 (4)	**7.77 (1)**	11.56 (3)
		deviation	1.3	2.47	0.98	1.52
	C-index	mean	18.57 (2)	32.85 (4)	**17.58 (1)**	22.79 (3)
		deviation	1.74	1.77	1.65	2.0
	subset zero-one loss	mean	75.11 (4)	61.17 (3)	58.58 (2)	**50.45 (1)**
		deviation	3.63	1.77	1.56	1.74
	calibration loss	mean	22.74 (2)	-	**4.86 (1)**	-
		deviation	2.76	-	0.73	-
	remaining Hamming loss	mean	12.70 (2)	-	**12.03 (1)**	-
		deviation	0.30	-	0.15	-

Table 5.1: Table of the experiments results of the different solving approaches to graded multilabel classification with $|\mathscr{L}| = 5$ and $|\mathscr{L}| = 10$ on the BeLaE dataset

6 Conclusions

In the following chapter first the content of this theses is recapped and summarized. Afterwards the future prospects of the approaches proposed in this thesis and their relevance to further research is discussed.

6.1 Summary

During this thesis the basic foundations of the graded multilabel classification were recapped. Thereby different classification problems like the binary, the ordinal, the multilabel classification were discussed. The multilabel classification was compared with the problem of label ranking to introduce the calibrated label ranking, an approach to solve a multilabel classification problem using a reinterpretation as label ranking problem. At least the graded multilabel classifiaction was introduced and discussed.
Afterwards the already available solving approaches from literature were recapped and discussed. These are the horizontal reduction which cuts the original graded multilabel classification problem into several multilabel classification problems to solve the overall problem, the vertical reduction which reduces the original problem to several ordinal classification problems to solve the overall problem and the complete reduction which uses the ideas from both horizontal and vertical reduction to reduce the problem to binary binary classification problems. All three approaches solve the reduced problems and aggregate the results to find a solution to the original problem. Also to concrete implementations of these techniques were discussed. The IBLR-ML which uses the horizontal reduction and solves each with the IBLR-ML algorithm and an implementation of the complete reduction using horizontal reduction and binary reduction to achieve a vertical reduction of the horizontal subproblems.
Next the already aware approaches were compared with the three new, by this thesis introduced approaches using pairwise classifiers. These classifiers and their individual advantages and drawbacks were discussed in detail. First the calibrated label ranking was discussed in more detail to allow a generalization of it to the graded multilabel classification because all three discussed solving approaches introduced in this thesis are generalizations of the calibrated label ranking. Afterwards the direct approach was introduced which reduces the grade difference between two labels of some data instance to a binary relation and tries to learn pairwise classifiers to predict which one is probable be aware with a higher grade. Additionally a set of virtual labels is introduced to separate the labels with different grades. Next the approach using dependent level cuts is discussed. This approach tries to learn pairwise classifiers for each individual level cut and by this tries to use the hole grade distance information to find a better calibration of the virtual labels in the ranking of the labels. The pairwise classifiers are learned for all label comparisons with regard to the individual level cut this means if the label has a higher grade than the virtual label representing a level cut or not. During the discussion of this approach two negative effects to its accuracy were shown. Additionally an strategy to reduce the negative effects was presented which showed good results in the evaluation of the approaches. Lastly the approach using independent level cuts was presented. This approach tries to solve one calibrated label ranking problem per level cut and aggregate the results similar to the horizontal reduction. Concluding the new approaches and one of the already available approaches were

tested on a data set and the results were presented and discussed. Therefore several losses for the graded multilabel classification were discussed and a new loss measuring the calibration of the virtual labels was introduced. To compare the approaches the experiments using the BeLaE data set from [3] were rebuild and as baseline the BR-10NN was reimplemented. The result of the experiments were shown and discussed. Thereby the general suitability of the pairwise approach to the graded multilabel classification was observed. The results also showed that the more the grade information and the correlation between the labels is used to predict the better the results of an approach are. For the case of a reduction to other classification problems the way of aggregating the individual results was shown as being crucial to gain good predictions.

6.2 Future prospects

Like mentioned in the motivation of this thesis, finding good solutions to the graded multilabel classification is necessary. The new approaches presented by this thesis and their results seem to promise the general suitability of pairwise approaches in the context of the graded multilabel classification and the varying of the results of the different approaches support the assumption that the results of the pairwise approaches even can be better. So a further research in different other approaches of pairwise classifiers is promising. The most promising research field here in my opinion is the search of other possible approaches to aggregate the predictions of the individual horizontal problems recapping the consequences of the two different way used until now. But also the comparison with the second already available approach with the approaches presented by this work seems to be worth mentioned to facilitate the evaluation of the pairwise approach in the setting of the graded multilabel classification.
One non-satisfying fact was that the experimental comparison of the different approaches had to rely on their results on classification problems generated from only one data set. This, like already mentioned in the introduction of this thesis, relies on the fact, that at the moment no other graded multilabel classification data set is known to me. But especially the advances caused by the filtering during the approach using dependent horizontal cuts may only rely on some special circumstances like the inner structure of the data set. So in my opinion the foundation of all further research to this topic should be the acquisition of additional and appropriate data sets to enable consolidated findings. In order to this problem I collected raw data for some new data set. This will consist of the redactional ratings of movies made by a German TV-guide. Each movie is awarded with zero to three points in the categories 'humor', 'intellectual value', 'action', 'excitement' and 'eroticism'. But unfortunately I was not able to finish the data set in time to use it for evaluating the approaches in this introduced by this thesis. I hope to be able to present the results of the approaches on this data set in some future publication.

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] Andrea E. Abele-Brehm and Mahena Stief. Die Prognose des Berufserfolgs von Hochschulabsolventinnen und -absolventen. *Zeitschrift für Arbeits- und Organisationspsychologie*, 48(1):4–16, 2004.

[2] Jaime S. Cardoso and Joaquim F. Pinto da Costa. Learning to classify ordinal data: The data replication method. *J. Mach. Learn. Res.*, 8:1393–1429, December 2007.

[3] Weiwei Cheng, Krzysztof Dembczyński, and Eyke Hüllermeier. Graded multilabel classification: The ordinal case. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 223–230, Haifa, Israel, June 2010. Omnipress.

[4] Weiwei Cheng and Eyke Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.

[5] Eibe Frank and Mark Hall. M.: A simple approach to ordinal classification. In *In: Proc 12th Europ Conf on Machine Learning*, pages 145–156. Springer, 2001.

[6] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 65–82. Springer-Verlag, 2010.

[7] Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73(2):133–153, June 2008.

[8] T. Gärtner and Shankar Vembu. Label ranking algorithms: A survey. In Eyke Hüllermeier Johannes Fürnkranz, editor, *Preference Learning*. Springer–Verlag, 2010. (to appear).

[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1):10–18, November 2009.

[10] Ge Hyun Nam. Ordered pairwise classification. Master's thesis, TU Darmstadt, Knowledge Engineering Group, 2007. Diplom.

[11] Sang-Hyeun Park and Johannes Fürnkranz. Efficient pairwise classification. In J. N. Kok, J. Koronacki, Ramon López de Mántaras, S. Matwin, Dunja Mladenić, and A. Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning (ECML 2007, Warsaw, Poland)*, pages 658–665. Springer-Verlag, 2007.

[12] Ross J. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[13] Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *Int J Data Warehousing and Mining*, 2007:1–13, 2007.

[14] Alexander Vitanyi. LPCforSOS framework. Technical report, Departement of Computer Science, TU Darmstadt, Germany, 2010.

[15] L.A. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.