Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Top-k Hierarchical Classification

Sechan Oh
Moloco
165 University Avenue
Palo Alto, California 94301

Abstract

This paper studies a top-k hierarchical classification problem. In top-k classification, one is allowed to make k predictions and no penalty is incurred if at least one of k predictions is correct. In hierarchical classification, classes form a structured hierarchy, and misclassification costs depend on the relation between the correct class and the incorrect class in the hierarchy. Despite that the fact that both top-k classification and hierarchical classification have gained increasing interests, the two problems have always been studied separately. In this paper, we define a top-k hierarchical loss function using a real world application. We provide the Bayes-optimal solution that minimizes the expected top-k hierarchical misclassification cost. Via numerical experiments, we show that our solution outperforms two baseline methods that address only one of the two issues.

Introduction

In top-k classification, one is allowed to make k predictions and no penalty is incurred if at least one of k predictions is correct. Top-k classifcation problems have gained increasing interests thanks to the ImageNet Challenge. When the number of classes is large and their distinctions are ambiguous, allowing multiple guesses is a natural remedy (Gupta, Bengio, and Weston 2014; Lapin, Hein, and Schiele 2015). Precision@k also frequently arises in other application areas including information retrieval and search advertising (Broder et al. 2007; Usunier, Buffoni, and Gallinari 2009). The users of such application systems often consider only the first few results shown in the user interface, and thus the performance of the system depends on its predication accuracy at the top.

Hierarchical classification, in which classes form a structured hierarchy, is another common form of classification problems. For example, in news article categorization, 'basketball', 'soccer', and 'baseball' are the sub-classes of 'sports' category. In this case, misclassification costs depend on the relation between the correct class and the incorrect class in the hierarchy. For example, the cost of classifying 'basketball' news to 'soccer' may be less costly than classifying it to 'international politics'. Hierarchical classification takes such hierarchical misclassification cost into account (Cesa-bianchi et al. 2005). Other examples of hierarchical classification problems include music, video, image recognition (Deng et al. 2012), patent categorization (Cai and Hofmann 2004; Eisinger et al. 2013), web content categorization (Dumais and Chen 2000), and gene function prediction problems (Vens et al. 2008). This literature has proposed several versions of hierarchical loss functions, and various classification algorithms (Ramírez-Corona, Sucar, and Morales 2014; Bi and Kwok 2015; Ramírez-Corona, Sucar, and Morales 2016).

Top-k classification and hierarchical classification are not mutually exclusive problems, which implies the existence of many application areas in their intersection. Despite this fact, in the literature the two problems have always been studied separately. This paper connects these two distinctly studied areas. We first introduce a top-k hierarchical loss function, which extends the hierarchical loss function proposed by (Cesa-bianchi et al. 2005) to the top-k setting. Next, we provide a Bayes-optimal solution that minimizes the expected top-k hierarchical misclassification cost. Via numerical experiments, we show that simple extensions of existing hierarchical classification algorithms yield poor performances in the top-k setting.

This research is motivated by a real world application system in a large service enterprise. When engaging on a new service contract, service firms first develop a project proposal, which includes detailed information about job tasks, schedules, dependencies, and job positions to deliver the service (Oh, Rhodes, and Strong 2016). After signing, firms assign the right experts to the project based on the information written in the project proposal. In large service enterprises, labor resource planning is a very complex process, and involves a central planning system (Naveh et al. 2007). One critical requirement for such a system is the use of a standard expertise taxonomy that defines employees' job roles and specialties. Such expertise taxonomies are hierarchical (Wei, Varshney, and Wagman 2015).

Staffing professionals are in charge of submitting a resource request for a project based on the project proposal. An interface system can help automate this task. The interface system receives a raw project proposal and creates a draft resource request. The key part of this conversion would be tagging the appropriate job role and specialty for each job position. For example, if the plan includes a position de-

scribed as "Data Science" in "Big Data" service area, the system may recommend "Research Scientist: Computer Science" as the job role and specialty. The system may show up to k options in the user interface. Staffing professionals will then review the recommended options, and submit a confirmed request after making necessary changes.

If staffing professionals find a correct job role and specialty among the recommended options, they can simply click an accept button. If one of the recommended option has the correct job role, but an incorrect specialty, then they need to manually adjust the specialty using a drop-down button. If all recommendations have incorrect job roles, then they need to manually select a proper job role, and then choose a specialty that is dependent on the chosen job role. Here, the time that the staffing professionals spend during this review process defines the misclassification cost of the recommender system. If the k recommended options include the correct correct job role and specialty, no extra cost is incurred. If one of the recommended options has the correct job role, but a wrong specialty, there is some cost (time) to be incurred. If none of the recommended option has the correct job role, about twice as much cost (time) is incurred. Here, the loss function is hierarchical and depends on the top-k performance.

To the best of our knowledge, our work is the first to connect the literature on top-k classification (Usunier, Buffoni, and Gallinari 2009; Boyd et al. 2012; Swersky et al. 2012) and the literature on hierarchical classification. (Cesa-bianchi et al. 2005; Cesa-Bianchi, Gentile, and Zaniboni 2006; Silla Jr and Freitas 2011; Bi and Kwok 2015; Ramírez-Corona, Sucar, and Morales 2016). Our contribution is two-folds. First, we define a new loss function, which extends the hierarchical loss function defined by (Cesa-bianchi et al. 2005) to the top-k setting. Second, we provide the Bayes-optimal solution that minimizes the expected top-k hierarchical misclassification cost. Our solution can be used as a useful first benchmark method for future research.

Top-k Hierarchical Loss Function

Consider the following classification problem. For a given input x, one needs to identify the right class $y = (y_1, y_2, \ldots, y_D)$, which is a vector of dimension D, from a set of classes $\mathcal{Y}$. Each element of the class vector y corresponds to a node in a tree that represents a taxonomy. Thus, a class y indicates a unique path from the root of the three to a leaf node, and every leaf node of the tree has the equal depth of D (with root at depth 0). If this condition is not met, one can create single child at each sub-level of the original leaf node so that all final leaf nodes are located at depth D. The number of classes is the same as the number of leaf nodes in the tree. Figure 1 is an example class set of size 5 and depth 2. We denote the set of the d-th element of each class, i.e., nodes at depth d, by $\mathcal{Y}_d$, and denote the set of children of a node y_d by $\text{chi}(y_d) \subseteq \mathcal{Y}_{d+1}$. Because each node of the tree has a unique path from the root, if two classes y and $\hat{y}$ satisfy $y_d = \hat{y}_d$, then $y_m = \hat{y}_m$ also holds for every $m < d$. We denote the set of classes whose d-th element is y_d by $\text{path}(y_d) \subseteq \mathcal{Y}$. Every input x has a unique correct class, which we denote by $Y(x)$.

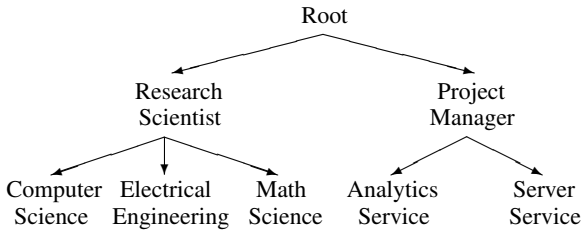

Figure 1: Example class taxonomy

For a given input x, the classifier selects K classes as the candidates for $Y(x)$. Suppose that y^1, y^2, ..., y^K are selected. Then, we define the misclassification loss as

$$\text{err}(y^1, \ldots, y^K, Y(x)) = \min_k \left[\sum_{d=1}^{D} c_d \mathbb{1}\{y_d^k \neq Y_d(x) \wedge y_m^k = Y_m(x), m < d\} \right] \quad (1)$$

where c_d is a decreasing sequence of non-negative numbers. The function $\mathbb{1}\{y_d^k \neq Y_d(x) \wedge y_m^k = Y_m(x), m < d\}$ indicates the event that y^k and $Y(x)$ coincide up to depth $d-1$ and deviate at depth d. In other words, the first deviation at level d incurs a loss of c_d, and no additional loss is incurred at deeper levels. The decreasing property of c_d implies that the penalty is the greatest if the chosen class deviates the correct class at depth 1, and the cost decreases down to zero as the number of the matched elements increases. When k classes are chosen, the misclassification loss is simply the minimum of the individual misclassification losses.

Recall the motivating example of the expertise classification problem. In that example, misclassified job role and specialty leads staffing professionals spend extra time for manual selection, and the extra time can be seen as the misclassification cost. No loss is incurred if one of the chosen job role and specialty is fully correct. Partial loss is incurred if one of the selection has the correct job role, but an incorrect specialty. The top-k hierarchical loss function (2) well captures this cost structure. This loss function extends the hierarchical loss function of (Cesa-bianchi et al. 2005) to the top-k setting. The loss function is fairly general, and reduces to a more conventional loss function if the hierarchical costs c_d meet a certain condition. For example, if $c_d = 1$ for every d, our error function is the standard top-k 0/1 loss function of (Lapin, Hein, and Schiele 2015).

Bayes-Optimal Classifier

Suppose that we are given a probability mass function $p(y|x) = Prob(Y(x) = y)$, which measures the probability that $y \in \mathcal{Y}$ is the correct class for the given input x. Recall from the tree structure that if two classes y and $\hat{y}$ satisfy $y_d = \hat{y}_d$, then $y_m = \hat{y}_m$ also holds for every $m < d$. Thus, we have $Prob(Y_d(x) = y_d) = Prob(Y_m(x) = y_m$ for every $m \leq d)$. Now recursively we define

$$\begin{aligned} p_D(y_D|x) &= Prob(Y_D(x) = y_D) = p(y|x), \\ p_d(y_d|x) &= Prob(Y_d(x) = y_d) \end{aligned}$$

Table 1: Example class hierarchy and matching probability

y_1	y_2	$p(y\|x)$
Research Scientist	Computer Science	0.25
Research Scientist	Electrical Engineering	0.15
Research Scientist	Math Science	0.25
Project Manager	Analytics Service	0.3
Project Manager	Server Service	0.05

$$= \sum_{j \in \text{chi}(y_d)} p_{d+1}(j|x) \text{ for } d < D, \quad (2)$$

which indicates the probability that the d-th element $Y(x)$ is y_d.

Given the probability mass function, the optimal classifier selects K classes that minimizes the following expected loss function:

$$\min_{y^1,\ldots,y^K \in \mathcal{Y}} E_{Y(x)}[\text{err}(y^1, \ldots, y^K, Y(x))]. \quad (3)$$

We denote the optimal set of classes by $\mathcal{Y}^*(x)$.

Single Selection Case ($K = 1$)

Before discussing the general case, we first consider the single selection case, i.e., $K = 1$.

Theorem 1. *When $K = 1$, selecting the y with the maximum reward*

$$r(y|x) = \sum_{d=1}^{D-1} (c_d - c_{d+1}) p_d(y_d|x) + c_D P_D(y_D|x)$$

is optimal.

If $c_d = 1$ for every d, i.e., when the loss function is the conventional top-k 0/1 loss function, the Bayes-optimal classifier simply chooses the K classes with the highest matching probability $p(y|x)$. The loss cannot be avoided unless a chosen class fully coincide with the right class at every depth. In contrast, when the loss is reduced if parts of the class, i.e., some elements of the class, coincide with those of the right class, the Bayes-optimal classifier may not simply select the k classes with the highest matching probability.

To elaborate the intuition behind the reward function in Theorem 1, we provide an example. Table 1 shows an example class tree and a probability mass function for a given input x. In this example, 'Project Manager: Analytics Service' has the highest matching probability of 0.3. All three classes whose depth 1 element is 'Research Scientist' have smaller matching probabilities than this class. However, the probability that 'Research Scientist' is the correct job role is 0.65, which is much larger than the probability that 'Project Manager' is the correct job role. Hence, when c_2 is much small compared to c_1, i.e., when there is significant time saving for a partially matched class, the Bayes-optimal classifier will choose 'Research Scientist: Computer Science' or 'Research Scientist: Math Science'.

The reward function $r(y|x)$ in Theorem 1 reflects such partial matching rewards. When a certain class y is chosen, not only its own matching probability, but also the matching probabilities of other classes that share common elements with y contribute to the total cost. More specifically, if the d-th element of the chosen class is y_d, the expected error is reduced by the multiplication of $c_d - c_{d+1}$ and the sum of the matching probabilities of all classes that has the same d-th element, i.e., $(c_d - c_{d+1}) p_d(y_d|x)$.

Multiple Selection Case ($K > 1$)

Next, we consider the multiple selection case, i.e., $K > 1$. Note that in the objective function (3) the order between minimization and expectation cannot be reversed. Thus, choosing the K labels with the highest rewards $r(y|x)$ as define in Theorem 1 may not provide the optimal set of classes. Recall the example in Table 1. Suppose that c_2 is much smaller than c_1, and thus 'Research Scientist: Computer Science' and 'Research Scientist: Mathematical Science' have the same maximum rewards $r(y|x)$. If one selects both of these two classes, the probability of having the correct job role is only 0.65, which is the same as in the case of choosing only one. The probability of full matching is 0.5. In contrast, if one selects only one of the two, and selects 'Project Manager: Analytics Service' as the second choice, the probability of having the correct job role is now 1, and the probability of full matching also increased to 0.55. The observation highlights the importance of ensuring the diversity of selected classes.

In this subsection, we derive a dynamic programming algorithm that constructs the correct reward function for the multiple selection case. To do so, we first transform the loss function (2) such that minimization can be omitted. Note that in the original definition, $\text{err}(y^1, y^2, \ldots, y^K, Y) = c_d$ implies that $y^k_{d-1} = Y_{d-1}$ for some k, i.e., at least one class has the correct depth $d-1$ element and that $y^k_d \neq Y_{d-1}$ for all k, i.e., no class has the correct depth d element. Thus, the following holds:

$$\begin{aligned} & \text{err}(y^1, y^2, \ldots, y^K, Y) \\ = & \sum_{d=1}^{D} c_d \mathbb{1}\{\exists k.\, y^k_d \neq Y_d \wedge \forall k.\, y^k_{d-1} = Y_{d-1}\} \\ = & \sum_{d=1}^{D} c_d \left(\mathbb{1}\{\exists k.\, y^k_{d-1} = Y_{d-1}\} - \mathbb{1}\{\exists k.\, y^k_d = Y_d\} \right). \end{aligned}$$

This transformation enables us to simplify the expected loss function as:

$$\begin{aligned} & E_{Y(x)}[\text{err}(y^1, y^2, \ldots, y^K, Y(x))] \\ = & \; c_1 - \sum_{d=1}^{D-1} (c_d - c_{d+1}) \text{Prob}(\exists k.\, y^k_d = Y_d(x)) \\ & - c_D \text{Prob}(\exists k.\, y^k_D = Y_D(x)). \end{aligned} \quad (4)$$

In this loss function, a reward of $(c_d + c_{d+1}) Prob(\exists k.\, y^k_d = Y_d(x))$ is attained (a cost of that amount is reduced) for each depth d. This reward is the same as the reward defined in Theorem 1 except that $p_d(y_d|x)$ is replaced with $\text{Prob}(\exists k.\, y^k_d = Y_d(x))$. It is important to note that

Prob($\exists k.\ y_d^k = Y_d(x)$) $\neq \sum_{k=1}^{K} p_d(y_d^k|x)$ unless $y_d^k \neq y_d^l$ for every $k \neq l$. An inclusion of a class whose depth d element y_d to the choice set reduces the expected loss by $(c_d - c_{d+1})p_d(y_d|x)$, but the inclusion of another class which shares the same d-th element does not provide any further benefit. Hence, unlike in the single selection case, the reward of selecting a certain class depends on other classes that are included in the choice set.

In order to obtain the Bayes-optimal classifier, we first prove a simple theorem that shows the selection priority between two classes that deviate only at the last depth.

Theorem 2. *Suppose that $y \in \mathcal{Y}^*(x)$. Then, $\hat{y} \in \mathcal{Y}^*(x)$ holds for every $\hat{y}$ such that $y_d = \hat{y}_d$ for every $d < D$ and $p(\hat{y}|x) > p(y|x)$.*

The theorem implies that if two classes deviate only at the leaf node, the class with a higher matching probability should always be chosen earlier than the other. Thus, when assigning the reward of a node y_{D-1} to its children, one can safely assign the full reward $(c_{D-1} - c_D)p_{D-1}(y_{D-1}|x)$ to the single child y_D that has the largest probability $p_D(y_D|x)$, and assign no rewards to other children.

The following algorithm extends this result, and assigns the matching rewards at each depth via backward recursion. Initially, every class gets its depth D matching reward of $c_D p(y|x)$. Then, recursively at each depth $d < D$, for each node $y_d \in \mathcal{Y}_d$ the matching reward of $(c_d - c_{d+1})p_d(y_d|x)$ is fully assigned to a single child that has the highest reward-to-go $r_D(y|x)$, and other children get no additional rewards.

Algorithm K-H

Input: $x, \mathcal{Y}, c_d, p_d(y_d|x)$
Output: $r_1(y|x)$ for $y \in \mathcal{Y}$
Initialize: $r_D(y|x) \leftarrow c_D p(y|x)$ for each $y \in \mathcal{Y}$
for $d = D-1$ **to** 1 **do**
  for each $y_d \in \mathcal{Y}_d$ **do**
    $\delta_d(y_d) \leftarrow (c_d - c_{d+1})p_d(y_d|x)$
    $y^*(y_d) \leftarrow \text{argmax}_{y \in \text{path}(y_d)} r_{d+1}(y|x)$
    for each $y \in \text{path}(y_d)$ **do**
      $r_d(y|x) \leftarrow r_{d+1}(y|x) + \delta_d(y_d)$ **if** $y = y^*(y_d)$
      $r_d(y|x) \leftarrow r_{d+1}(y|x)$ **otherwise**
    end for
  end for
end for

Theorem 3. *Selecting K classes with the largest reward $r_1(y|x)$ computed by Algorithm K-H is optimal, and the corresponding optimal expected loss is $c_1 - \sum_{y \in \mathcal{Y}^*} r_1(y|x)$.*

Numerical Experiments

In this section, we perform numerical experiments with real-world data sets. First, we use the twelve functional genomics data sets by (Clare and King 2003; Vens et al. 2008) (https://dtai.cs.kuleuven.be/clus/hmcdatasets/). These data sets contain features of yeast genes and their functional classes, which form a tree-structured hierarchy. Second, we use the data collected from the real-world application system that we discussed in the introduction.

In all experiments, we first estimate the conditional probability $p(y|x)$ using the hierarchical structure of classes. More specifically, we first construct $p_1(y_1|x)$ as a multinomial logit model with lasso penalty using all training and validation data sets. For $d > 1$, we construct $p_d(y_d|x, y_{d-1})$ for each y_{d-1} in the same way as we construct $p_1(y_1|x)$ except that we now use only the data samples whose depth $d-1$ element is y_{d-1}. Then, the probability $p_d(y_d|x)$ is given as $p_1(y_1|x)\prod_{l=2}^{d} p_l(y_l|x, y_{l-1})$. Finally, we consctuct the Bayes-optimal classifier via Algorithm K-H.

We compare the misclassification cost of our Bayes-optimal classifier (Algorithm K-H) with those of two benchmark methods. Each benchmark method addresses either top-k classification or hierarchical classification, but not both. The first benchmark method computes the single choice reward function defined in Theorem 1, and simply selects the k classes with the highest rewards. This method takes the hierarchical loss function into account, but ignores its impact on top-k performance. We call this method H-only method. The second benchmark method simply chooses the k classes with the highest matching probability $p(y|x)$. When c_d is the same for all d, i.e., when the loss function is not hierarchical, this method is Bayes-optimal for top-k selection. Hence, we call this method K-only method.

Table 2 reports the results from the gene function data. While setting the cost parameters at $c = [40, 30, 20, 10/(D-3), \ldots, 10/(D-3)]$, we computed the misclassification costs on the testing data set for several values of K, i.e., number of classes to choose.

When $K = 1$, the Bayes-optimal method and the H-only method always select the same single class. Hence, the performances of the two methods are the same in this case. In contrast, the K-only method, which selects classes according to their simple matching probabilities, performs worse than the Bayes-optimal method even when only one class can be selected. The only exception appears for the 'hom' data set, in which K-only method slightly outperforms the Bayes-optimal method. Note that when $K = 1$, our loss function reduces to the standard hieararchical loss function of (Cesa-bianchi et al. 2005). Under this loss function, the H-only method, which takes the hieararchical loss function into account, peforms the same as our algorithm. In contrast, the K-only method underforms our algorithm. Hierarchical classification is an instance of cost sensitive learning. In cost sensitive learning, misclassification loss defers depending on the correct class and the wrong class. As the closeness of the two classes within the tree increases (as they shore more ancestors), the cost of misclassification decreases. The Bayes-optimal classifier incorporates this cost structure into account, whereas the K-only method does not.

When multiple selection is allowed, the Bayes-optimal method significantly outperforms the two benchmark methods. The performance gap is apparent even when ten classes are allowed to be chosen. An interesting observation is that H-only method substantially under-performs the K-only method except for the single selection case. The H-only method takes into account the benefit of having a partially matched class in the selection, but ignores the joint impacts when multiple classes are chosen. Hence, if there is a non-

Table 2: Misclassification costs on gene function data

data set	K	K-H	H-only	K-only
cellcyle	1	**90.01**	**90.01**	90.97
cellcyle	5	**52.47**	79.11	61.64
cellcyle	10	**39.43**	71.99	48.19
church	1	**89.46**	**89.46**	90.75
church	5	**52.66**	81.56	63.39
church	10	**38.02**	74.90	44.58
derisi	1	**89.27**	**89.29**	92.78
derisi	5	**54.46**	82.34	60.99
derisi	10	**38.89**	73.82	45.12
eisen	1	**89.21**	**89.21**	90.06
eisen	5	**49.55**	81.74	59.99
eisen	10	**38.79**	76.09	42.02
expr	1	**88.53**	**88.53**	93.28
expr	5	**51.18**	80.35	65.88
expr	10	**36.44**	71.51	47.02
gasch1	1	**89.30**	**89.30**	92.26
gasch1	5	**51.32**	84.52	63.46
gasch1	10	**37.24**	76.66	49.59
gasch2	1	**89.65**	**89.65**	90.94
gasch2	5	**52.25**	80.84	63.85
gasch2	10	**38.84**	74.25	51.89
hom	1	89.13	89.13	**89.01**
hom	5	**52.46**	78.38	63.36
hom	10	**38.84**	72.70	47.72
pheno	1	**89.39**	**89.39**	90.93
pheno	5	**53.42**	83.47	66.66
pheno	10	**37.54**	74.32	49.64
seq	1	**88.93**	**88.93**	91.63
seq	5	**52.64**	71.62	63.39
seq	10	**37.13**	63.84	45.62
spo	1	**89.38**	**89.38**	89.46
spo	5	**53.71**	85.08	65.38
spo	10	**40.28**	78.45	47.35
struc	1	**89.76**	**89.76**	92.73
struc	5	**51.69**	85.78	64.41
struc	10	**37.63**	77.95	48.79

leaf node that has a very high matching probability, the H-only method is likely to choose multiple classes whose paths include this node. In this case, the chances of partial matching decreases. We investigate this issue further in the next experiment.

We remark that there are several hiearachical classification algorithms proposed in the literature. See for example (Cesa-bianchi et al. 2005; Bi and Kwok 2015; Ramírez-Corona, Sucar, and Morales 2016). Similar to our H-only method, one can use other hieararchical classification algorithms and choose the k classes that have the highest matching probability as the solution set. Athough such methods can slightly outperform the H-only benchmark method, they are unlikely to perform well in the top-k setting because they ignore the joint impact of classes on the misclassification loss.

The second experiment is on the data set collected from the expertise matching application. For each position, one is given a free form text description of the role and a formal service area name. The system needs to identify the correct job role and specialty for each position. The expertise taxonomy is a tree of depth 2. The interface system first computes the similarity score between the job position description and the title and description of each job role and specialty combination. The similarity score takes values from 0 to 1. The interface system also computes demand based score of each job role and specialty combination. More specifically, the system computes the relative frequency that a certain job role and specialty combination has been used by other projects that fall in the same service area, and use the relative frequency as the second score. The interface system uses these two scores for each class to identify the correct one.

We collected a data set of 539 job positions. For each position, we are given the two scores described above for all job role and specialty combinations, and the correct answer confirmed by staffing professionals via the system. As above, we use the multinomial logistic model to estimate the conditional matching probabilities $p_d(y_d|x)$, and construct the Bayse-optimal classifier using Algorithm K-H. While fixing $c_1 = 100$, we computed the misclassification costs of the three algorithms on the test data set while changing the values of K, i.e., the number of classes to select, and c_2, i.e., the cost of partially matched class. The results are reported in Table 3.

Table 3: Misclassification costs on expertise matching data

K	c_2	K-H	H-only	K - only
3	100	**27.06**	**27.06**	**27.06**
3	80	**23.66**	24.62	24.91
3	60	**22.04**	23.45	23.20
3	40	**19.52**	24.92	21.77
3	20	**17.18**	26.21	19.73
3	0	**14.61**	26.40	17.87
5	100	**19.10**	**19.10**	**19.10**
5	80	**17.26**	21.13	17.72
5	60	**15.29**	20.25	16.26
5	40	**14.49**	22.08	14.71
5	20	**12.46**	21.20	12.95
5	0	**9.67**	20.54	11.44
7	100	**15.30**	**15.30**	**15.30**
7	80	**13.78**	17.76	14.29
7	60	**12.35**	16.76	12.84
7	40	**10.61**	18.98	11.56
7	20	**8.78**	18.38	9.77
7	0	**7.53**	17.73	8.65

Note that when $c_2 = 100$, the cost of a partially matched answer is the same as the cost of a fully incorrect answer. In this case, the top-k hierarchical loss function reduces to the standard top-k loss function, and all the three algorithms select classes according to their simply matching probabilities. Hence, their performances are the same when $c_2 = 100$.

As c_2 decreases, i.e., as the benefit of having a partially

matched class increases, the performance gap between the Bayse-optimal classifier and the two benchmark methods increases. As in the previous experiment, K-only method, which selects the classes according to the simple matching probability outperforms the H-only method, which ignores the joint impact of the classes that share common ancestors. When $c_2 = 0$, identifying the correct job role completely removes the misclassification cost, and thus the diversity of the job roles in the chosen set becomes critically important. When $c_2 = 0$, the reward of all classes that share the same job role will become identical under the H-only method. Hence, this method will select all classes whose job role has the highest matching probability up to the extent possible. Thus, this method substantially under-performs the other two methods when $c_2 = 0$.

To further the insights on the class selection logic behind the three methods, in Table 4 we also report the percentage of cases in which each classifier recommends the fully correct class or a partially correct class (i.e., correct job role and incorrect specialty).

Table 4: Partial and full matching percentages on expertise matching data

		Job role match only			Full match		
K	c_2	K-H	H-only	K-only	K-H	H-only	K-only
3	100	8.69	8.69	8.69	72.94	72.94	72.94
3	80	8.56	7.28	8.69	74.63	73.93	72.94
3	60	8.85	7.49	8.69	74.42	73.56	72.94
3	40	10.35	6.56	8.69	72.98	71.14	72.94
3	20	12.30	6.58	8.69	72.98	68.52	72.94
3	0	14.23	5.64	8.69	71.16	67.90	72.94
5	100	7.71	7.71	7.71	80.90	80.90	80.90
5	80	7.74	6.52	7.71	81.19	77.56	80.90
5	60	9.01	6.79	7.71	81.11	77.04	80.90
5	40	9.74	5.35	7.71	79.67	74.71	80.90
5	20	9.86	4.98	7.71	79.65	74.81	80.90
5	0	13.50	5.12	7.71	76.83	74.36	80.90
7	100	6.42	6.42	6.42	84.70	84.70	84.70
7	80	8.41	5.09	6.42	84.54	81.22	84.70
7	60	8.19	4.81	6.42	84.38	81.31	84.70
7	40	8.67	4.31	6.42	84.19	78.44	84.70
7	20	8.55	3.75	6.42	84.38	78.62	84.70
7	0	14.03	4.10	6.42	78.44	78.52	84.70

Under the Bayes-optimal classifier, the full matching percentage generally decreases as c_2 decrease at a mild rate. In contrast, the percentage of cases in which the classifier selects a correct job role, but no correct specialty increases rapidly as c_2 decreases. In other words, the Bayes-optimal classifier sacrifices the full matching probability for the partial matching probability. When c_2 is small, making sure that there is at least one answer whose job role is correct is very important. Thus, the Bayse-optimal solution is likely to include the classes that have different job roles to increase this chance.

Under the H-only classifier both the full matching percentage and the partial matching percentage decrease as c_2 decreases. As discussed above, this benchmark method ignores the fact that adding a new class whose job role is already included in the solution set does not increase the probability of having a correct job role. If a certain job role has a high matching probability, its reward is problematically added to all classes that has this job role, and they may all appear in the selection set of the H-only classifier. The result highlights the importance of assigning a matching reward of a certain node only to a single child as addressed in Algorithm K-H.

Note that the full matching percentage indicates the performance of the three methods under the standard top-k loss function (no hierarchical loss strcuture). The result shows that under the standard top-k loss function the Bayes-optimal solution underperforms the K-only method. The K-only method selects classes based only on the full matching probability, and thus regardless of the misclassification cost c_2, the matching percentages remain the same.

Conclusion

In this paper, we have studied the top-k hierarchical classification problem. We have introduced a new loss function, which extends the hierarchical loss function of (Cesabianchi et al. 2005) to the top-k setting. We have shown that under this top-k hierarchical loss function, selecting the class with the highest matching probability is not necessarily optimal even when only one class can be chosen. The algorithm takes into account the fact that the expected reduction in the misclassification cost due to the addition of a particular element should only be computed once. We have provided an algorithm that computes the rewards of adding each class to the choice set via a backward induction. The outcome of this algorithm is the Bayes-optimal classifier that minimizes the expected top-k hierarchical loss function. Via numerical experiments, we show that the Bayse-optimal classifier significantly reduce the misclassification loss compared to the benchmark classifiers that addresses only either top-k classification or hierarchical classification. The relative performance gap is significant when the cost of partial matched class is substantially smaller than the full mismatch cost, and when the number of classes to select is large.

Our research leaves several open issues. In the paper, we have focused on the Bayse-optimal classifier, and the probability estimation is based on standard learning methods. Both the top-k classification literature and the hierarchical classification have proposed non-Bayesian learning methods for each problem. As our numerical experiments show, the simple extension of the method developed for each problem does not address the two issues simultaneously. Because of the complexity of the top-k hierarchical loss function, developing a specialized learning algorithm for this loss function can be a challenging problem. Yet, a specialized learning method may further reduce the misclassification loss. The proposed loss function and the classification algorithm is also limited to trees. Directed acyclic graph is another common form of hierarchical taxonomies. Extension of the proposed solution to directed acyclic graphs will be important future research.

References

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

Boyd, S.; Cortes, C.; Mohri, M.; and Radovanovic, A. 2012. Accuracy at the top. In *Advances in neural information processing systems*.

Broder, A.; Fontoura, M.; Josifovski, V.; and Riedel, L. 2007. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*.

Cai, L., and Hofmann, T. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*.

Cesa-bianchi, N.; Gentile, C.; Tironi, A.; and Zaniboni, L. 2005. Incremental algorithms for hierarchical classification. In *Advances in Neural Information Processing Systems*.

Cesa-Bianchi, N.; Gentile, C.; and Zaniboni, L. 2006. Hierarchical classification: combining bayes with svm. In *Proceedings of the 23rd international conference on Machine learning*.

Clare, A., and King, R. D. 2003. Predicting gene function in saccharomyces cerevisiae. *Bioinformatics* 19.

Deng, J.; Krause, J.; Berg, A. C.; and Fei-Fei, L. 2012. Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*.

Dumais, S., and Chen, H. 2000. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*.

Eisinger, D.; Tsatsaronis, G.; Bundschus, M.; Wieneke, U.; and Schroeder, M. 2013. Automated patent categorization and guided patent search using ipc as inspired by mesh and pubmed. *Journal of biomedical semantics* 4(1).

Gupta, M. R.; Bengio, S.; and Weston, J. 2014. Training highly multiclass classifiers. *Journal of Machine Learning Research* 15(1).

Lapin, M.; Hein, M.; and Schiele, B. 2015. Top-k multiclass svm. In *Advances in Neural Information Processing Systems*.

Naveh, Y.; Richter, Y.; Altshuler, Y.; Gresh, D. L.; and Connors, D. P. 2007. Workforce optimization: Identification and assignment of professional workers using constraint programming. *IBM Journal of Research and Development* 51.

Oh, S.; Rhodes, J.; and Strong, R. 2016. Impact of cost uncertainty on pricing decisions under risk aversion. *European Journal of Operational Research* 253(1).

Ramírez-Corona, M.; Sucar, L. E.; and Morales, E. F. 2014. Multi-label classification for tree and directed acyclic graphs hierarchies. In *European Workshop on Probabilistic Graphical Models*, 409–425.

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

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

Swersky, K.; Tarlow, D.; Adams, R.; Zemel, R.; and Frey, B. 2012. Probabilistic n-choose-k models for classification and ranking. In *Advances in Neural Information Processing Systems*.

Usunier, N.; Buffoni, D.; and Gallinari, P. 2009. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th annual international conference on machine learning*.

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

Wei, D.; Varshney, K. R.; and Wagman, M. 2015. Optigrow: People analytics for job transfers. In *Big Data (BigData Congress), 2015 IEEE International Congress on*.