Learning from Pairwise Preference Data using Gaussian Mixture Model

Mihajlo Grbovic and **Nemanja Djuric** and **Slobodan Vucetic**[1]

Abstract. In this paper we propose a fast online preference learning algorithm capable of utilizing incomplete preference information. It is based on a Gaussian mixture model that learns soft pairwise label preferences via minimization of the proposed soft rank loss measure. Standard supervised learning techniques, such as gradient descent or Expectation Maximization can be used to find the unknown model parameters. Algorithm outputs are soft pairwise label preference predictions that need to be further aggregated to produce a total label ranking prediction, for which several existing algorithms can be used. The main advantages of the proposed learning algorithm are the ability to process a single training instance at a time, low time and space complexity, ease of implementation, and model reuse.

1 INTRODUCTION

Label Ranking is emerging as an important and practically relevant preference learning field. Unlike the standard problems of classification and regression, label ranking learning is a complex learning task, which involves the prediction of strict label order relations, rather than single values. Specifically, in the label ranking scenario, each instance, which is described by a set of features $\mathbf{x}$, is assigned a ranking of labels π, that is a total (e.g. $\pi = (3, 5, 1, 4, 2)$) or partial (e.g. $\pi = (5, 3, 2)$) order over a finite set of class labels $\mathcal{Y}$ (e.g. $\mathcal{Y} = \{1, 2, 3, 4, 5\}$). The label ranking problem consists of learning a model that maps instances $\mathbf{x}$ to a total label order $h : \mathbf{x}_n \rightarrow \pi_n$. It is assumed that a sample from the underlying distribution $D = \{(\mathbf{x}_n, \pi_n), n = 1, ..., N\}$, where $\mathbf{x}_n$ is a d-dimensional feature vector and π_n is a vector containing a total or partial order of a finite set $\mathcal{Y}$ of L class labels, is available for training.

This problem has recently received a lot of attention in the machine learning community and has been extensively studied [6, 4, 9, 3, 16]. A survey of recent label ranking algorithms can be found in [8].

There are many practical applications in which the objective is to learn an exact label preference of an instance in form of a total order. For example, in the case of document categorization, where it is very likely that a document belongs to multiple topics (e.g. sports, entertainment, baseball, etc.), one might not be interested only in predicting which topics are relevant for a specific document, but also to rank the topics by relevance. Additional applications include: meta-learning [18], where, given a new data set, the task is to induce a total rank of available algorithms according to their suitability based on the data set properties; predicting food preferences for new costumers based on the survey results, demographics, and other characteristics of respondents [12]; determining an order of questions in a survey for a specific user based on respondent's attributes. A recent publication [14], suggests clustering of label ranking data, which could be of great practical importance, especially in target marketing.

There are three principal approaches for label ranking. The first decomposes label ranking problem into one or several binary classification problems. Ranking by pairwise comparison (PW) [11], for example, creates $L \cdot (L-1)/2$ classification problems, one for each possible pairwise ranking. Pairwise binary classifier predictions are aggregated into a total label order by voting. The constraint classification (CC) [9], on the other hand, transforms the label ranking problem into a single binary classification problem by augmenting the data set, such that each example $\mathbf{x}$ is mapped into $L \cdot (L-1)/2$, $(d \times L)$-dimensional, examples. This allows for training of a single classifier.

The second approach is to use utility functions, where the goal is to learn mappings $f_k : X \rightarrow R$ for each label $k = 1, ..., L$, which assign a value $f_k(\mathbf{x})$ to each label, such that $f_i(\mathbf{x}) < f_j(\mathbf{x})$ if $\mathbf{x}$ prefers label j over i. For example, the label ranking method proposed in [6] represents each f_k as a linear combination of base ranking functions. The utility functions are learned to minimize the number of ranking errors and the final rank is produced simply by ranking the utility scores. It should be noted that the utility function-based approach is also popular in the related object ranking problem, where techniques based on SVM [10] and AdaBoost [7] have been proposed.

The third approach is represented by a collection of algorithms which use probabilistic approaches for label ranking, such as the ones that rely on the Mallows [13] and the Plackett-Luce (PL) [15] models. A typical representative is the instance-based (IB) label ranking [4, 3]. Given a new instance $\mathbf{x}$, the k-Nearest Neighbor algorithm is used to locate its neighbors in the feature space. Then, the neighbors' label rankings are aggregated to provide prediction. Rank aggregation for prediction is not a trivial task, particularly in presence of partial label ranks. Mallows [4] and Plackett-Luce [3] models that describe probability distribution of rankings have been used to come up with the optimization criterion for rank aggregation. As an alternative, CPS probabilistic model for rank aggregation has been recently proposed [16].

Instance-based label ranking algorithms are simple and intuitive. Furthermore, they have been shown to outperform the competitors in various label ranking scenarios. However, their success comes at a large cost associated with both memory and time. First, they require that the entire training data set is stored in memory, which can be costly or even impossible in the resource-constrained applications. Storing the original data can also raise privacy issues as the data might contain sensitive user information. Second, the prediction involves costly nearest neighbor search and aggregation of neighbors' label rankings. The aggregation is slow as it requires using op-

[1] Temple University, Department of Computer and Information Sciences, Philadelphia, USA, email: {mihajlo.grbovic, nemanja.djuric, slobodan.vucetic }@temple.edu

timization techniques at prediction time, such as iterative Minorization Maximization(IB-PL) or exhaustive search (IB Mallows).

In this paper we propose an online, time- and memory-efficient algorithm for learning label preferences based on the Gaussian Mixture Model (GMM), which could be attractive because of an intuitively clear learning process and ease of implementation. The model preserves privacy as it consists of mixtures defined by prototypes which are not the actual data points. Every prototype is associated with preference judgments for each pair of labels. Unlike many competitors, our algorithm is not limited to a specific type of label ranking and could support various ranking structures (bipartite, multipartite, etc). For an unlabeled instance, GMM predicts the soft label preferences by averaging prototypes' pairwise preferences according to distances.

These soft label preferences in form of a preference matrix need to be aggregated further, into a total order of labels. This is a well known problem in preference learning and is an especially popular research area in the object ranking scenario, where numerous methods have been proposed [5, 1, 2].

2 PRELIMINARIES

In the label ranking scenario, a single instance, described by a d-dimensional vector $\mathbf{x}$, is associated with a total order of assigned class labels, represented as a permutation π of the set $\{1, ..., L\}$, where L is the number of available class labels. We define π such that $\pi(i)$ is the class label at i-th position in the order, and $\pi^{-1}(j)$ is a position of the y_j class label in the order. The permutation can also describe incomplete ranking $\{\pi(1), ..., \pi(k)\} \subset \{1, ..., L\}, k < L$.

In our approach, instead of the total order, we use a zero-diagonal preference matrix $\mathbf{Y}$. When a preference between labels y_i and y_j exists, we set $\mathbf{Y}(i,j) > \mathbf{Y}(j,i)$ if y_i is preferred over y_j and $\mathbf{Y}(i,j) < \mathbf{Y}(j,i)$ otherwise, $\mathbf{Y}(i,j) + \mathbf{Y}(j,i) = 1$, for $i, j \in \{1, ..., L\}$. A value of $\mathbf{Y}(i,j)$ which is close to 1 is interpreted as a strong preference that y_i should be ranked above y_j. A typical approach is to assign $\mathbf{Y}(i,j) = 1$ and $\mathbf{Y}(j,i) = 0$ if $y_i \succ_{\mathbf{x}} y_j$. Similarly, uncertain (soft) preferences can be modeled by using values lower than 1. For example, indifferences (ties) are represented by setting $\mathbf{Y}(i,j) = \mathbf{Y}(j,i) = 0.5$. In case of non-existing, incomparable or missing preferences, both $\mathbf{Y}(i,j) = 0$ and $\mathbf{Y}(j,i) = 0$.

This representation allows us to work with complete and partial label orders, as well as with pairwise preferences with uncertainties and indifferences. Finally, bipartite and multi-partite label rankings could be handled as well.

Evaluation metrics. Let us assume that N historical observations are collected in a form of a data set $D = \{(\mathbf{x}_n, \mathbf{Y}_n), n = 1, ..., N\}$. The objective in all scenarios is to train a ranking function $h : \mathbf{x}_n \rightarrow \hat{\pi}_n$ from data set D that outputs a total label order.

In the Label Ranking scenario, to measure the degree of correspondence between true and predicted rankings for n-th example, π_n and $\hat{\pi}_n$ respectively, it is common to use the Kendall's tau distance $d_n = \mid \{(y_i, y_j) : \pi_n^{-1}(y_i) > \pi_n^{-1}(y_j) \wedge \hat{\pi}_n^{-1}(y_j) > \hat{\pi}_n^{-1}(y_i)\} \mid$. To evaluate a label ranking model, the label ranking loss on the data set D is defined as the average normalized Kendall's tau distance,

$$loss_{LR} = \frac{1}{N} \sum_{n=1}^{N} \frac{2 \cdot d_n}{L \cdot (L-1)}. \tag{1}$$

Note that the measure simply counts the number of discordant label pairs and reports the average over all considered pairwise rankings. Given the general preference matrix representation, assuming binary matrix predictions $\hat{\mathbf{Y}}_n$, we can rewrite (1) as

$$loss_P = \frac{1}{N} \sum_{n=1}^{N} \frac{\parallel \mathbf{Y}_n - \hat{\mathbf{Y}}_n \parallel_F^2}{L \cdot (L-1)}, \tag{2}$$

where $\| \cdot \|_F$ is Frobenius matrix norm. Indeed, for each example n, the square of the Frobenius norm sums up to double the number of discordant label pairs.

For models with soft label preference predictions $\hat{\mathbf{Y}}_n$, e.g., $\hat{\mathbf{Y}}_n(i,j) = 0.7, \hat{\mathbf{Y}}_n(j,i) = 0.3$, loss (2) can be interpreted as a soft version of (1).

We can solve the preference learning task in two stages. In the learning stage, function $f : \mathbf{x}_n \rightarrow \mathbf{Y}_n$ is learned via minimizing (2). In the aggregation stage, given the model predictions in a form of $\mathbf{Y}_n$, the total order prediction π_n is computed using a preference aggregation mapping $g : \mathbf{Y}_n \rightarrow \pi_n$. In the next section we show the details of the proposed Gaussian Mixture Model algorithm to be used in the learning stage. Existing algorithms such as [5, 1, 2], can be used in the aggregation stage.

3 GAUSSIAN MIXTURE MODEL FOR LABEL RANKING

The GMM model for label ranking is completely defined by a set of K mixtures, i.e., prototypes $\{(\mathbf{m}_k, \mathbf{Q}_k), k = 1, ..., K\}$, where $\mathbf{m}_k$ is a d-dimensional vector in input space and $\mathbf{Q}_k$ is the corresponding preference matrix.

First, we introduce the probability $\mathbb{P}(k \mid \mathbf{x})$ of assigning observation $\mathbf{x}$ to k-th prototype that is dependent on their (Euclidean) distance. Let us assume that the probability density $\mathbb{P}(\mathbf{x})$ of $\mathbf{x}$ can be described by a mixture model,

$$\mathbb{P}(\mathbf{x}) = \sum_{k=1}^{K} \mathbb{P}(\mathbf{x} \mid k) \cdot \mathbb{P}(k), \tag{3}$$

where K is the number of prototypes, $\mathbb{P}(k)$ is the prior probability that a data point is generated by k-th prototype, and $\mathbb{P}(\mathbf{x} \mid k)$ is the conditional probability that k-th prototype generates particular data point $\mathbf{x}$. Let us represent the conditional density function $\mathbb{P}(\mathbf{x} \mid k)$ with the normalized exponential form $\mathbb{P}(\mathbf{x} \mid k) = \theta(k) \cdot \exp(f(\mathbf{x}, \mathbf{m}_k))$ and consider a Gaussian mixture with $\theta(k) = (2\pi\sigma_p^2)^{-1/2}$ and $f(\mathbf{x}, \mathbf{m}_k) = -\|\mathbf{x} - \mathbf{m}_k\|^2 / 2\sigma_p^2$. We assume that all prototypes have the same standard deviation σ_p and the same prior, $\mathbb{P}(k) = 1/K$. Given this, using the Bayes' rule we can write the assignment probability as

$$\mathbb{P}(k \mid \mathbf{x}) = \frac{\exp(-\|\mathbf{x} - \mathbf{m}_k\|^2 / 2\sigma_p^2)}{\sum_{u=1}^{K} \exp(-\|\mathbf{x} - \mathbf{m}_u\|^2 / 2\sigma_p^2)}. \tag{4}$$

To derive a cost function, we propose the following mixture model for the posterior probability $\mathbb{P}(\mathbf{Y} \mid \mathbf{x})$,

$$\mathbb{P}(\mathbf{Y} \mid \mathbf{x}) = \sum_{k=1}^{K} \mathbb{P}(k \mid \mathbf{x}) \cdot \mathbb{P}(\mathbf{Y} \mid k). \tag{5}$$

Based on this model, example $\mathbf{x}$ is assigned to the prototypes probabilistically and its preference matrix is a weighted average of the prototype preference matrices. The mixture model assumes the conditional independence between $\mathbf{x}$ and $\mathbf{Y}$ given k, $\mathbb{P}(\mathbf{Y} \mid \mathbf{x}, k) = \mathbb{P}(\mathbf{Y} \mid k)$. For $\mathbb{P}(k \mid \mathbf{x})$ we assume the Gaussian distribution from

(4). For the probability of generating a preference matrix $\mathbf{Y}$ by prototype k, $\mathbb{P}(\mathbf{Y} \mid k)$, we also assume Gaussian error model with mean $(\mathbf{Y} - \mathbf{Q}_k)$ and standard deviation σ_y. The resulting cost function $l(\lambda)$ can be written as the negative log-likelihood,

$$l(\lambda) = -\frac{1}{N}\sum_{n=1}^{N}\ln\sum_{k=1}^{K}\mathbb{P}(k \mid \mathbf{x}) \cdot \mathcal{N}(\mathbf{Y} - \mathbf{Q}_k, \sigma_y^2), \qquad (6)$$

where $\lambda = \{\mathbf{m}_k, \mathbf{Q}_k, k = 1, ..., P, \sigma_p, \sigma_y\}$ are the model parameters. For the compactness of notation, let us define $g_{nk} = \mathbb{P}(k \mid \mathbf{x}_n)$ and $e_{nk} = \mathcal{N}(\mathbf{Y}_n - \mathbf{Q}_k, \sigma_y^2)$.

It is important to observe that, after proper normalization, (6) reduces to (2) if examples are assigned to prototypes deterministically. Therefore, it can be interpreted as its soft version. If prototype matrices $\mathbf{Q}_k$ consisted of only 0 and 1 entries (hard label preferences) (6) further reduces to (1).

The objective is to estimate the unknown model parameters, namely prototype positions $\mathbf{m}_k$ and their preference matrices $\mathbf{Q}_k, k = 1, ..., K$. This is done by minimizing the cost function $l(\lambda)$ with respect to the parameters. This can be achieved in several different ways. If online learning capability is a requirement, one can use the stochastic gradient descent method and obtain the learning rules by calculating derivatives $\partial l(\lambda)/\partial \mathbf{m}_k$ and $\partial l(\lambda)/\partial \mathbf{Q}_k$ for $k = 1, ..., K$. This results in following rules for n-th training example,

$$\begin{aligned} \mathbf{m}_k^{n+1} &= \mathbf{m}_k^n - \alpha(n)\tfrac{(L_n - e_{nk})\cdot g_{nk}}{L_n}\tfrac{(\mathbf{x}_n - \mathbf{m}_k)}{\sigma_p^2} \\ \mathbf{Q}_k^{n+1} &= \mathbf{Q}_k^n - \alpha(n)\tfrac{e_{nk}\cdot g_{nk}}{L_n}\tfrac{(\mathbf{Y}_n - \mathbf{Q}_k)}{\sigma_y^2}, \end{aligned} \qquad (7)$$

where $L_n = \sum_{k=1}^{P}(g_{nk} \cdot e_{nk})$ and $\alpha(n)$ is the learning rate.

The resulting model has complexity $\mathcal{O}(NKL)$. Otherwise, the problem can straightforwardly be mapped into Expectation-Maximization (EM) framework following the procedure from [17].

Initialization is done by selecting the first P training points as the initial prototypes. If any $\mathbf{Q}_k$ prototype preference matrix obtained in such manner contains empty elements, they are replaced with 0.5 entries, as the corresponding labels will initially be treated equally.

GMM model generalizes the training data to produce a representation in terms of prototype vectors and effectively utilizes distances to prototypes as a similarity measure to calculate the predicted label rank. When compared to IB-based algorithms, GMM is a more global model, that aggregates over more data, thus also alleviating influence of noise. Therefore, it is expected to outperform IB-based algorithms, whose performance is highly dependent on the quality of the training data and the presence of outliers, since no abstraction is made during the training phase. We could try to aggregate over more data by considering a large number of neighbors in IB algorithms, however, by doing so we start ignoring distances between the query instance and its neighbors as a similarity measure. This remains to be seen after a proper experimental evaluation.

A disadvantage of the GMM model is that it requires aggregation of the predicted label preference matrix to produce a total order of labels. Luckily, most of the existing algorithms have low complexity, e.g. $\mathcal{O}(L \log L)$ for QuickSort [1]. In our future work we plan to evaluate the pros and cons of different aggregation methods.

4 CONCLUSION AND FUTURE WORK

We introduced an idea of a Gaussian Mixture Model algorithm for Label Ranking. The main advantages of the new method are: (1) it is capable of operating in an online manner, (2) it is memory-efficient since it operates on a predefined budget, (3) it preserves privacy, (4) it could potentially reuse the model when new labels are introduced. There are several avenues which need to be pursued further: (1) experimental evaluation of the proposed method on benchmark data (2) determining the optimal number of prototypes K using statistical learning theory, (3) low rank approximation of pairwise preference matrices to reduce memory requirements, (4) evaluating different preference matrix aggregation algorithms, (5) applying the algorithm to clustering of label ranking data. In the future work we plan to address these issues.

REFERENCES

[1] Nir Ailon and Mehryar Mohri, 'An Efficient Reduction of Ranking to Classification', in *COLT*, pp. 87–98. Omnipress, (2008).

[2] Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin, 'Robust Reductions from Ranking to Classification', in *COLT*, volume 4539 of *Lecture Notes in Computer Science*, pp. 604–619, (2007).

[3] Weiwei Cheng, Krzysztof Dembczyński, and Eyke Hüllermeier, 'Label ranking methods based on the Plackett-Luce model', in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 215–222, (2010).

[4] Weiwei Cheng, Jens Hühn, and Eyke Hüllermeier, 'Decision tree and instance-based learning for label ranking', in *Proceedings of the 26th International Conference on Machine Learning (ICML-09)*, pp. 161–168, (2009).

[5] William W. Cohen, Robert E. Schapire, and Yoram Singer, 'Learning to order things', *Journal of Artificial Intelligence Research*, **10**, 243–270, (1999).

[6] Ofer Dekel, Christopher Manning, and Yoram Singer, 'Log-Linear Models for Label Ranking', in *Advances in Neural Information Processing Systems*, volume 16, MIT Press, (2003).

[7] Yoav Freund, Raj D. Iyer, Robert E. Schapire, and Yoram Singer, 'An Efficient Boosting Algorithm for Combining Preferences', *Journal of Machine Learning Research*, **4**, 933–969, (2003).

[8] Thomas Gärtner and Shankar Vembu, 'Label Ranking Algorithms: A Survey', in *Preference Learning*, ed., Eyke Hüllermeier Johannes Fürnkranz, Springer–Verlag, (2010). (to appear).

[9] Sariel Har-Peled, Dan Roth, and Dav Zimak, 'Constraint classification for multiclass classification and ranking', in *Proceedings of the 16th Annual Conference on Neural Information Processing Systems, NIPS-02*, pp. 785–792. MIT Press, (2003).

[10] R. Herbrich, T. Graepel, and K. Obermayer, 'Large Margin Rank Boundaries for Ordinal Regression', in *Advances in Large Margin Classifiers*, pp. 115–132. MIT Press, (2000).

[11] Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker, 'Label ranking by learning pairwise preferences', *Artificial Intelligence*, **172**, 1897–1916, (2008).

[12] Toshihiro Kamishima and Shotaro Akaho, 'Efficient Clustering for Orders', in *ICDM Workshops*, pp. 274–278. IEEE Computer Society, (2006).

[13] C. L. Mallows, 'Non-null ranking models', *Biometrika*, **44**, 114–130, (1967).

[14] Grbovic Mihajlo, Nemanja Djuric, and Slobodan Vucetic, 'Supervised clustering of label ranking data', *SIAM International Conference on Data Mining*, (2012).

[15] R. L. Plackett, 'The analysis of permutations', *Applied Statistics*, **24**(2), 193–202, (1975).

[16] Tao Qin, Xiubo Geng, and Tie-Yan Liu, 'A New Probabilistic Model for Rank Aggregation', in *Advances in Neural Information Processing Systems 23*, 1948–1956, MIT Press, (2010).

[17] Weigend A. S., Mangeas M., and Srivastava A. N., 'Nonlinear Gated Experts for Time Series: Discovering Regimes and Avoiding Overfitting', *International Journal of Neural Systems*, **6**, 373–399, (1995).

[18] Ricardo Vilalta and Youssef Drissi, 'A perspective view and survey of meta-learning', *Artificial Intelligence Review*, **18**, 7795, (2002).