Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Metric Learning for Ordinal Data

Yuan Shi* and **Wenzhe Li*** and **Fei Sha**
Department of Computer Science
University of Southern California
Los Angeles, CA 90089, USA
{yuanshi, wenzheli, feisha}@usc.edu

Abstract

A large amount of ordinal-valued data exist in many domains, including medical and health science, social science, economics, political science, etc. Unlike image and speech datasets of real-valued data, learning with ordinal variables (i.e., features) presents unique challenges. In particular, the nominal differences between those feature values, which are just ranks, do not necessarily correspond to the real distances between the corresponding categories. Given their wide existence, it is imperative to develop machine learning algorithms that specifically address the need to model and infer with such data. In this paper, we present a novel metric learning algorithm that takes into consideration the nature of ordinal data. Our approach treats ordinal values as latent variables in intervals. Our algorithm then learns what those intervals are as well as distance metrics to measure distances between latent variables in those intervals. We derive the corresponding optimization algorithm and demonstrate how that can be solved effectively. Experimental results show that the proposed approach significantly improves baselines that do not explicitly model ordinal features.

Introduction

Ordinal data arises frequently in behavioral, medical, educational, psychological and social science. For these domains, people usually provide their "measurements" in the forms of subjective and imprecise opinions. For example, college students rate the overall quality of the class as "excellent","good","fair","poor" and "very poor"; consumers rate the product on a 5-point scales; the degree of agreement can be measured by different levels such as "strongly agree", "agree", "neutral", "disagree", etc. We can easily find application domains for such ordinal data where human-generated data plays important role (Fen Xia 2007).

The focus of this paper is to investigate how we can use ordinal data (i.e., features) in machine learning models for prediction and classification. One popular way is to treat them as real-valued continuous variables. However, the difference in numerical scales does not necessarily reflect the true distances between the corresponding categories. For example, the true distance between "strongly agree" and "agree" is usually different from "agree" and "neutral" (Torra et al. 2003; O'Brien 1979), despite nominally they are just one scale apart from each other. Another common approach is to treat ordinal variables as categorical variables, which are then coded with *1-of-K* encoding. However, the major problem of this approach is that it ignores the ordering information which may be crucial for the learning problems.

Ordinal values are qualitative in nature, and a common view of ordinal values is that they are nonstrict monotonic transformations of interval variables (Winship and Mare 1984). That is, one or more values within the same interval are mapped into the same ordinal value. For example, when individuals rank categories, such as "strongly agree", "agree", "neutral", etc, an underlying continuous variable denoting individuals degrees of agreement is mapped into categories that are ordered but are separated by unknown distance (Winship and Mare 1984).

The contribution of this paper is to propose a novel way of modeling and inference with ordinal variables when they are treated as input features. We consider the problem within the context of measuring distances/dissimilarity, which is a fundamental problem in machine learning. In particular, we consider the metric learning framework (Weinberger and Saul 2009; Weinberger and Tesauro 2007; Kedem et al. 2012; Goldberger et al. 2004). Our main idea is to discover the intervals *as well as* to learn a distance metric to measure the distance between ordinal features.

To this end, we extend the popular algorithm large margin nearest neighbors (LMNN) (Weinberger and Saul 2009). The idea is to compute *expected distances* between latent ordinal variables. In order to compute such expectation, we consider different types of distributions for the latent variables, namely, uniform, Beta and Beta rectangular distributions. The optimization can be done efficiently via projected subgradient descent. Our experimental results on real-world datasets show that properly learning ordinal information from the data will improve the classification performance. We also extend our algorithm to kernel regression and obtain positive results (see the Supplementary Materials[1]). Although this paper focuses on metric learning, all the methods proposed here can be easily generalized to other

*Equal contribution.

[1]Can be found at www-scf.usc.edu/~yuanshi/

types of machine learning algorithms which use distances as a measure of dissimilarity.

Related Work

Although there has been a considerable amount of work in the case when the response variables/labels are ordinal (McCullagh 1980; Ananth and Kleinbaum 1997; Paquet, Thomson, and Winther 2012; Cardoso and Pinto da Costa 2007; Frank and Hall 2001; Li and Lin 2006), limited work has considered the case where the input features are ordinal, which are common in many domains. The problem is largely unexplored both in machine learning and statistics communities. In this section, we review two main lines of previous work that explicitly deal with ordinal features.

One line of work focuses on measuring distances by exploring the statistical properties without considering the labels. In (Bar 1995; Leon and Carriere 2007; De Leon and Carriere 2005), the authors propose distance measure for mixed nominal, ordinal and continuous data by applying the KL divergence to the general mixed data model, which is an extension of the general location model (Olkin and Tate 1961). All these methods are based on multivariate normal assumptions under some structural constraints on covariance matrices. However, these models are tractable only when dealing with a small number of features. Also it is not clear how to extend them to supervised learning setting. (Podani 1999) proposes another unsupervised method but uses a much simpler and tractable similarity measure, which is built on top of Gower's coefficient measure (Gower 1971). We have used this approach as one of the baselines and also extended it to supervised learning.

Another line of work is to use latent variable models under Bayesian framework. They assume that the ordinal observations are drawn from generative models with latent threshold variables (Poon and Wang 2012; Webb and Forster 2008; Kottas, Mller, and Quintana 2005; Gerd Ronning 1996; Johnson 2006; Kukuk 1998; Tran, Phung, and Venkatesh 2012). However, most of these methods make strong assumptions on the probabilistic models, partially for identifiability issues. Besides, due to the intractability of the posterior distributions, they have to use approximate inference techniques such as Markov Chain Monte Carlo (MCMC), which generally scales poorly to the size of models and data. Finally, those Bayesian approaches cannot be easily extended to learning the distance metrics.

Background and Notations

Assume we have N data points $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)$, where $\boldsymbol{x}_i \in \mathcal{R}^D$ and each dimension is ordinal (i.e. with ordinal scales as 1, 2,$\cdots$), y_i is the corresponding label. We further assume that each input variable x_{id} is associated with a continuous latent variable z_{id} where $d = 1, 2, \cdots, D$. x_{id} and z_{id} satisfy the following *threshold model*: if $x_{id} = t$, then $\boldsymbol{m}_d(t) \leq z_{id} < \boldsymbol{m}_d(t+1)$, where $\boldsymbol{m}_d(t)$ and $\boldsymbol{m}_d(t+1)$ are two thresholds corresponding to the tth ordinal scale at the dth dimension.

Let $\boldsymbol{m}_d$ be a vector containing all thresholds for the dth dimension, then $\boldsymbol{m}_d$ should satisfy following ordinal constraints

$$-\infty < \boldsymbol{m}_d(1) < \boldsymbol{m}_d(2) < \cdots < \boldsymbol{m}_d(k_d + 1) < \infty \quad (1)$$

where k_d is the total number of ordinal scales for the dth dimension. We denote the constraint set as $\mathcal{P}$. In the following, we use vector $\boldsymbol{m}$ to denote the stack of all $\{\boldsymbol{m}_d\}$: $\boldsymbol{m} = [\boldsymbol{m}_1^\top, \boldsymbol{m}_2^\top, \cdots, \boldsymbol{m}_D^\top]^\top$.

Metric learning Many machine learning models are based on some forms of dissimilarity measure between data samples. Typically Euclidean distance, $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$, is used as the default metric[2], which is a special case of the more general Mahalanobis distance

$$\mathcal{D}_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)$$

where $\boldsymbol{M} \in \mathcal{R}^{D \times D}$ and $\boldsymbol{M} \succeq \boldsymbol{0}$.

The choice of $\boldsymbol{M}$ may significantly affect the performance of the underlying machine learning models. For example, in classification setting, an ideal metric would pull data points in the same class closer and push data points in different classes away (Weinberger and Saul 2009). When a set of labeled training samples are available, a better $\boldsymbol{M}$ can often be learned using metric learning algorithms.

Methodology

Distance Metric for Ordinal Features

For ordinal data, when measuring the distance between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we do not directly compute the distance on $\boldsymbol{x}$. We instead compute the distance on the corresponding latent variables $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ which represent the ground truth values for ordinal inputs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$:

$$\tilde{\mathcal{D}}_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{z}_i - \boldsymbol{z}_j)^\top \boldsymbol{M} (\boldsymbol{z}_i - \boldsymbol{z}_j) \quad (2)$$

where the notation $\tilde{\mathcal{D}}$ indicates that the distance is computed on the latent variables.

The challenge is that we do not know the true values of $\boldsymbol{z}$. To address this, we make assumptions on the distribution of $\boldsymbol{z}$ and consider the expected distance

$$\begin{aligned}\mathbb{E}[\tilde{\mathcal{D}}_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j)] = (\mathbb{E}[\boldsymbol{z}_i] - \mathbb{E}[\boldsymbol{z}_j])^\top \boldsymbol{M} (\mathbb{E}[\boldsymbol{z}_i] - \mathbb{E}[\boldsymbol{z}_j]) \\ + \mathrm{Tr}(\boldsymbol{M} \mathbb{VAR}[\boldsymbol{z}_i]) + \mathrm{Tr}(\boldsymbol{M} \mathbb{VAR}[\boldsymbol{z}_j]) \quad (3)\end{aligned}$$

where $\mathbb{E}[\boldsymbol{z}]$ is a D dimensional vector for the expectation of $\boldsymbol{z}$, $\mathbb{VAR}[\boldsymbol{z}_i]$ is a diagonal $D \times D$ matrix storing the variances of $\boldsymbol{z}_i$, and Tr denotes the trace of a matrix.

Distribution of $\boldsymbol{z}_{id}$ According to the threshold model, we model $\boldsymbol{z}_{id}$ as a random variable drawn from the interval $[\boldsymbol{m}_d(\boldsymbol{x}_{id}), \boldsymbol{m}_d(\boldsymbol{x}_{id} + 1)]$. In particular, we model $\boldsymbol{z}_{id}$ as

$$\boldsymbol{z}_{id} = \boldsymbol{m}_d(\boldsymbol{x}_{id}) + \delta_d \left(\boldsymbol{m}_d(\boldsymbol{x}_{id} + 1) - \boldsymbol{m}_d(\boldsymbol{x}_{id})\right) \quad (4)$$

where $\delta_d \in [0, 1]$ is a random variable. Then we have

$$\begin{aligned}\mathbb{E}[\boldsymbol{z}_{id}] &= \boldsymbol{m}_d(\boldsymbol{x}_{id}) + \mathbb{E}[\delta_d] \left(\boldsymbol{m}_d(\boldsymbol{x}_{id} + 1) - \boldsymbol{m}_d(\boldsymbol{x}_{id})\right) \\ \mathbb{VAR}[\boldsymbol{z}_{id}] &= \mathbb{VAR}[\delta_d] \left(\boldsymbol{m}_d(\boldsymbol{x}_{id} + 1) - \boldsymbol{m}_d(\boldsymbol{x}_{id})\right)^2\end{aligned}$$

[2] In this paper, we follow the popular terminology in the metric learning literature, calling the *squared distance* as distance.

By recognizing that $\mathbb{E}[\boldsymbol{z}_i]$ is linear in $\boldsymbol{m}$ and $\mathbb{VAR}[\boldsymbol{z}_i]$ is quadratic in $\boldsymbol{m}$, it is easy to write them in more compact forms

$$\mathbb{E}[\boldsymbol{z}_i] = \boldsymbol{A}_i\boldsymbol{m} \quad \mathbb{VAR}[\boldsymbol{z}_i] = (\boldsymbol{B}_i\boldsymbol{m}\boldsymbol{m}^\top\boldsymbol{B}_i^\top) \odot \boldsymbol{I}_D$$

where $\odot$ denotes the Hadamard product of matrices. $\boldsymbol{A}_i, \boldsymbol{B}_i \in \mathcal{R}^{D\times W}$ and W is the dimensionality of $\boldsymbol{m}$. The value of $\boldsymbol{A}_i$ depends on $\mathbb{E}[\delta_d]$ and $\boldsymbol{x}_i$, while the value of $\boldsymbol{B}_i$ depends on $\mathbb{VAR}[\delta_d]$ and $\boldsymbol{x}_i$. Note that we assume $\{\delta_d\}$ are independent with each other.

Now we can rewrite the distance function as:

$$\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] = \mathrm{Tr}(\boldsymbol{M}(\boldsymbol{A}_i - \boldsymbol{A}_j)\boldsymbol{m}\boldsymbol{m}^\top(\boldsymbol{A}_i - \boldsymbol{A}_j)^\top) + \mathrm{Tr}\left(\boldsymbol{M}(\boldsymbol{B}_i\boldsymbol{m}\boldsymbol{m}^\top\boldsymbol{B}_i^\top + \boldsymbol{B}_j\boldsymbol{m}\boldsymbol{m}^\top\boldsymbol{B}_j^\top) \odot \boldsymbol{I}\right) \quad (5)$$

which is linear in $\boldsymbol{M}$ and quadratic in $\boldsymbol{m}$.

Distribution of δ_d To model δ_d, we consider three types of distributions

- *Uniform distribution*: $\mathbb{E}(\delta_d) = 0.5$ and $\mathbb{VAR}(\delta_d) = 1/12$.
- *Beta distribution*, which has two parameters α and β controlling the shape of distribution: $\mathbb{E}(\delta_d) = \alpha/(\alpha+\beta)$ and $\mathbb{VAR}(\delta_d) = \alpha\beta/((\alpha+\beta)^2(\alpha+\beta+1))$.
- *Beta rectangular distribution*, which is a finite mixture of Beta distribution and uniform distribution. In addition to parameter α, β, it has a third parameter θ. The form of mean and variance are more complicated than the Beta distribution: $\mathbb{E}(\delta_d) = \theta\alpha/(\alpha+\beta) + (1-\theta)/2$ and $\mathbb{VAR}(\delta_d) = (\theta\alpha(\alpha+1))/(k(k+1)) + (1-\theta)/3 - ((k+\theta(\alpha-\beta))^2)/(4k^2)$ where $k = \alpha+\beta$.

For simplicity, we stack the parameters for $\{\delta_d\}$ into a vector $\boldsymbol{\theta}$, and denote its domain as Θ.

Margin Based Metric Learning

In the following, we show how an optimal metric can be learned in classification tasks where discrete labels are given, i.e. $y \in \{1, 2, \cdots, C\}$.

We follow the Large Margin Nearest Neighbor (LMNN) framework (Weinberger and Saul 2009) which learns a Mahalanobis metric to reduce the classification error of k-nearest neighbor (kNN). For any point $\boldsymbol{x}_i$ in the training set, LMNN differentiates two sets of neighboring data points: *target neighbors* - points whose labels are the same as $\boldsymbol{x}_i$, and *imposters* - points whose labels are different from $\boldsymbol{x}_i$.

LMNN optimizes the following objective to find an optimal metric $\boldsymbol{M}$

$$\min_{\boldsymbol{M}\succeq\boldsymbol{0},\boldsymbol{\xi}\geq\boldsymbol{0}} \sum_i \sum_{j\in S_i^+} \mathcal{D}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \lambda \sum_{ijk} \xi_{ijk}$$
$$\text{s.t.} \quad \mathcal{D}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mathcal{D}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k) + 1 \leq \xi_{ijk}$$
$$\forall j \in S_i^+, k \in S_i^- \quad (6)$$

where S_i^+ and S_i^- denote the set of target neighbors and imposters of $\boldsymbol{x}_i$, respectively. λ is the tradeoff parameter which balances two forces, pulling the target neighbors towards $\boldsymbol{x}_i$ and pushing the imposters away so that the distance to an imposter should be greater than the distance to the target neighbor by a margin of 1 using the slack variable ξ_{ijk}. Because the problem is linear in its parameters, we set the margin equal to 1 as its exact value only impacts the scale of $\boldsymbol{M}$.

To learn a metric for ordinal data, we replace the distance $\mathcal{D}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with the expected distance $\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ and arrive at the following optimization problem

$$\min_{\boldsymbol{M}\succeq\boldsymbol{0},\boldsymbol{\xi}\geq\boldsymbol{0},\boldsymbol{m}\in\mathcal{P},\boldsymbol{\theta}\in\Theta} \sum_i \sum_{j\in S_i^+} \mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] + \lambda \sum_{ijk} \xi_{ijk}$$
$$\text{s.t.} \quad \mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] - \mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)] + \gamma \leq \xi_{ijk}, \forall j \in S_i^+, k \in S_i^- \quad (7)$$

Besides the substituted distance function, there are two important changes in (7) compared to (6). First, we have additional constraints $\boldsymbol{m} \in \mathcal{P}$ and $\boldsymbol{\theta} \in \Theta$. Second, since (7) is nonlinear in its parameters, we cannot arbitrarily set the margin value γ to 1. Instead, we treat γ as a hyper-parameter of our model.

Optimization We substitute the slack variable ξ_{ijk} with the hinge loss function

$$\xi_{ijk} = \left[\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] - \mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)] + \gamma\right]_+ \quad (8)$$

where $[x]_+ = \max(0, x)$. We then follow an alternating optimization strategy to optimize $\boldsymbol{M}$, $\boldsymbol{m}$ and $\boldsymbol{\theta}$. Each time we fix two sets of parameters, and optimize the rest:

- Fix $\boldsymbol{m}$ and $\boldsymbol{\theta}$, optimize $\boldsymbol{M}$. Note that the objective becomes convex in $\boldsymbol{M}$, and $\boldsymbol{M}$ can be optimized using projected subgradient descent method. At each descent step, we first obtain a candidate solution $\hat{\boldsymbol{M}}$, and then project it into the positive semidefinite cone.
- Fix $\boldsymbol{M}$ and $\boldsymbol{\theta}$, optimize $\boldsymbol{m}$. Since $\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ is quadratic in $\boldsymbol{m}$, ξ_{ijk} is the difference between two convex functions. We therefore use the Convex Concave Procedure (CCCP) (Yuille, Rangarajan, and Yuille 2002) to optimize $\boldsymbol{m}$. The procedure iteratively solves the following sequence of convex programs

$$\min_{\boldsymbol{m}} \sum_{ij} \mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] + \lambda \sum_{ijk} (\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] - \boldsymbol{m}^\top \nabla_{\boldsymbol{m}} \left(\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)]\right) + \gamma)_+ \quad (9)$$

  where $\boldsymbol{m}^\top \nabla_{\boldsymbol{m}} \left(\mathbb{E}[\tilde{\mathcal{D}}^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)]\right)$ is a linearization for the concave part in the original objective function (8). (9) can be optimized via projected subgradient descent. In each descent step, we first obtain a candidate solution $\hat{\boldsymbol{m}}$ and then do the projection by solving the quadratic programming below

$$\min_{\boldsymbol{m}} \|\boldsymbol{m} - \hat{\boldsymbol{m}}\|_2^2 \quad \text{s.t.} \quad \boldsymbol{m} \in \mathcal{P} \quad (10)$$

  where $\mathcal{P}$ denotes a set of linear ordinal constraints.

- Fix $\boldsymbol{m}$ and $\boldsymbol{M}$, optimize $\boldsymbol{\theta}$, which is done by coordinate descent. In each descent step, we optimize the parameters corresponding to a single δ_d[3].

Our alternating optimization strategy only converges to local optimal solutions whose quality depend on the initialization. In our implementation, $\boldsymbol{m}$ is initialized such that the mean of $\boldsymbol{z}$ is the same as the ordinal scale value of $\boldsymbol{x}$, $\boldsymbol{M}$ is initialized with identity matrix, and $\boldsymbol{\theta}$ is initialized such that the mean of δ_d is 0.5 and its variance is closest to 1/12 (i.e. close to uniform distribution).

Note that the computational complexity of our method is arguably higher than LMNN (Weinberger and Saul 2009) which learns a size $D \times D$ matrix $\boldsymbol{M}$. The additional parameters our method learn are linear in D: $\boldsymbol{m}$ is $\mathcal{O}(D \cdot N_{max\_scale})$ (N_{max_scale} denotes the maximum number of ordinal scales that an ordinal feature could take, which is often a relatively small number in practice) and $\boldsymbol{\theta}$ is $\mathcal{O}(D)$. When $\boldsymbol{m}$ and $\boldsymbol{\theta}$ are fixed, the computation for learning $\boldsymbol{M}$ is the same as LMNN. When $\boldsymbol{M}$ is fixed, learning $\boldsymbol{m}$ and $\boldsymbol{\theta}$ is also straightforward with QP and gradient descent. The alternating steps converge quickly (often fewer than 10 alternations, and one does not need full convergence in each alternation). Thus, empirically the overall complexity is not substantially higher than LMNN and may be further improved.

Experiments

We compare our methods to several baselines in a number of classification tasks. Besides the classification performance, we also study the learned intervals and low-dimensional embedding derived from our methods.

We compare our methods to several baselines. The first set of baselines consists of different encoding schemes for ordinal data (Kurczynski 1970; Boriah, Chandola, and Kumar 2008; Fitkov-Norris, Vahid, and Hand 2012), and the remaining ones are methods specially designed for measuring similarities on ordinal data (Podani 1999). All these baseline algorithms are summarized below:

Real-Eucl, Real-LMNN These methods treat ordinal scales directly as continuous values. Eucl and LMNN indicate the corresponding metric: Euclidean metric and the metric learned by LMNN, respectively.

Binary-Eucl, Binary-LMNN These methods treat ordinal variables as categorical variables, and encode them using *1-of-K* encoding scheme. For example, ordinal scales $1, 2, 3$ are encoded into binary vectors $[1, 0, 0]$, $[0, 1, 0]$ and $[0, 0, 1]$.

Binary-Ord-Eucl, Binary-Ord-LMNN These methods use a binary encoding scheme which is capable of encoding ordinal information. For example, ordinal scales $1, 2, 3$ are encoded into $[1, 0, 0]$, $[1, 1, 0]$ and $[1, 1, 1]$.

Ex-Gower The method uses a modified similarity measure proposed in (Podani 1999), as an extension of Gower's coefficient measure (Gower 1971). It calculates the similarity[4] between two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by

$$\mathcal{G}_{ij} = \frac{\sum_d w_d(i,j) s_d(i,j)}{\sum_d w_d(i,j)} \tag{11}$$

where $s_d(i,j)$ measures the similarity of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on the dth feature. $s_d(i,j)$ is set to 1 if $\boldsymbol{x}_{id} = \boldsymbol{x}_{jd}$. When $\boldsymbol{x}_{id}$ and $\boldsymbol{x}_{jd}$ are different, $s_d(i,j)$ is calculated by

$$s_d(i,j) = 1 - \frac{|\boldsymbol{x}_{id} - \boldsymbol{x}_{jd}| - (T_d(i)-1)/2 - (T_d(j)-1)/2}{k_d - 1 - (T_d(k_d)-1)/2 - (T_d(1)-1)/2}$$

where $T_d(i)$ is the number of data points which have the same ordinal scale for the dth feature as $\boldsymbol{x}_i$, and k_d is the total number of ordinal scales for the dth feature. Intuitively, the numerator measures the number of steps required to move a point with the same value as $\boldsymbol{x}_{id}$ into the position of another point with the same value as $\boldsymbol{x}_{jd}$, and denominator is for normalization purpose. $w_d(i,j)$ is the weight for the dth feature which is typically set to 1.

Ex-Gower* We extend Ex-Gower to learn the weights w_d based on metric learning. Specifically, we learn these weights by optimizing LMNN objectives, using the following distance

$$\mathcal{D}_{\mathcal{G}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = C - \sum_d w_d s_d(i,j) \tag{12}$$

where $0 \leq w_d \leq 1$. C is a constant to maintain the non-negativity of distances.

Ord-LMNN-Uni, Ord-LMNN-Beta, Ord-LMNN-RecBeta Our methods that jointly learn the metric and thresholds. Uni, Beta, and RecBeta indicate the corresponding distribution of δ_d.

Thresh-Eucl The thresholds are learned using our proposed methods, but with the metric $\boldsymbol{M}$ fixed as identity matrix. We report the best results among different distributions of δ_d. This baseline is designed to study the effect of learning the thresholds alone.

Parameters of these methods are tuned on the validation data. Tunable parameters include the tradeoff parameter in LMNN, as well as the tradeoff parameter and margin parameter in our methods.

Datasets and Evaluation

We use 9 real-world datasets from UCI machine learning repository (Lichman 2013). For the first 5 datasets (Car, Nursery, Cancer, Hayes-Roth, Balance), all features are ordinal, which are directly applicable for our methods. For the rest 4 datasets (Vehicle, Vowel, Segmentation and Magic), the original real-valued features are first discretized[5] and then used by all methods. The number of samples in these datasets vary from a few hundreds to thousands. The largest

[3] Since the number of parameters are small, we use a grid search to find the optimal parameters

[4] We treat the negative similarity values as distances

[5] We discretize each dimension into 8 ordinal scales using proper thresholding so that different ordinal scales have almost the same frequency.

Table 1: kNN classification error on the datasets with ordinal features

Method	Car	Nursery	Cancer	Hayes-Roth	Balance
Real-Eucl	11.4±0.7	8.6±0.1	**4.3**±0.2	38.5±3.1	15.2±1.1
Real-LMNN	5.0±0.3	2.4±0.1	**4.3**±0.3	23.1±1.6	17.6±0.9
Binary-Eucl	24.0±1.4	24.0±0.2	6.0±0.3	50.0±2.9	32.7±1.9
Binary-LMNN	4.1±0.3	2.3±0.1	4.7±0.4	16.0±1.0	17.8±1.2
Binary-Ord-Eucl	12.3±0.4	8.7±0.1	4.4±0.3	45.5±3.3	16.7±0.8
Binary-Ord-LMNN	4.1±0.2	1.9±0.1	**4.3**±0.4	**15.4**±1.2	13.4±0.8
Ex-Gower	12.1±0.7	8.8±0.1	6.1±0.8	37.2±1.9	32.8±1.3
Ex-Gower*	7.9±0.4	2.6±0.1	5.5±0.5	33.3±1.4	35.1±1.2
Thresh-Eucl	4.5±0.4	2.3±0.1	4.5±0.4	22.3±1.9	14.5±0.7
Ord-LMNN-Uni	3.8±0.3	1.8±0.1	**4.3**±0.4	20.5±1.3	6.8±0.5
Ord-LMNN-Beta	3.7±0.3	**1.6**±0.1	**4.3**±0.4	18.6±1.0	**6.1**±0.5
Ord-LMNN-RecBeta	**3.4**±0.3	**1.6**±0.1	**4.3**±0.4	18.6±1.0	6.4±0.5

ones are Nursery (12,960 samples) and Magic (19,020 samples).

For each dataset, we randomly split it into the training (60% of samples), validation (20%) and test (20%) sets. The validation set is used to tune the parameters, and the performance is measured on the test set. All methods are evaluated using k-nearest neighbor classifier with k set to 3. For each dataset, we do 20 random splits, and the average results (mean and standard error) are reported.

Classification results

Table 1 reports the classification results on datasets with ordinal features, and Table 2 reports the results on datasets with discretized features[6]. Methods with the best error rates are highlighted. From the results, we observe that no matter what the underlying representation (Real, Binary, Binary-Ord) is, metric learning often improves the classification performance. Binary-LMNN and Binary-Ord-LMNN perform better than Real-LMNN, due to the fact that the binary encoding scheme increases the parameter space thus enhances the power of learned metrics. Binary-Ord-LMNN further improves over Binary-LMNN, which demonstrates the benefit of exploring ordinal information in the data. Among all methods, our proposed ones consistently perform the best or second best, with clear advantages on the Car, Nursery and Balance datasets. The training time of our methods is comparable to that of LMNN[7]. Note that by setting the distribution of δ_d other than uniform distribution, our methods can obtain slightly smaller error rates. This suggests that a relatively complicated distribution like the Beta-rectangular distribution may better capture the structure in ordinal data. The better performance of our methods over Thresh-Eucl reveals the importance of joint learning both the intervals and distance metric.

[6]As a reference point, the error rates of LMNN on original real-valued features are: 23.9±0.7 (Vehicle), 7.1±0.5 (Vowel), 3.4±0.2 (Segmentation), and 21.1±0.1 (Magic).

[7]On the Nursery dataset, Ord-LMNN-Uni takes 18 mins per hyperparameter setting while LMNN takes 5 mins. On the Car dataset, Ord-LMNN-Uni takes 2 mins and LMNN takes 1 min per hyperparameter setting (1.4GHz CPU).

Learned intervals

Once the thresholds are learned, interpreting the ordinal intervals is valuable and can reveal interesting properties in the data, for example, the distances between different ordinal scales, as well as the impact of particular scales. In Figure 1, we plot the learned intervals on selected datasets.

The Car dataset is for evaluating cars based on several ordinal features. Figure 1 (a) shows the intervals on feature `person` (number of persons to carry), with original ordinal scales as 1 (2 persons), 2 (4 persons), 3 (more than 4 persons). We notice that the right-most curve is nearly flat, which reveals that the distance between ordinal scale 2 and 3 is smaller than the one between ordinal scale 1 and 2. In other words, "capacity exceeds 4" is perceived similarly to "capcity of 4", which is consistent with our experience as car buyers generally pay more attention to whether a car carries 2 persons or 4 persons. Figure 1 (b) shows the intervals on feature `door` (number of doors), where the distances between different ordinal scales are almost the same.

The Nursery dataset is for ranking applications for nursery schools. Figure 1 (c) plots the learned intervals on feature `has-nursery` (level of the child's nursery), with original ordinal scales as 1 (proper), 2 (less proper), 3 (improper), 4 (critical) and 5 (very critical). We can see that the left-most curve is nearly flat, revealing that a child with proper nursery is perceived similarly to the one with less proper nursery. Figure 1 (d) plots the intervals on feature `health`, with original ordinal scales as 1 (recommended), 2 (priority), and 3 (not-recommended). The figure shows that the health condition "not-recommended" has a significant impact, which is reasonable as a child with poor health condition probably does not fit the nursery school.

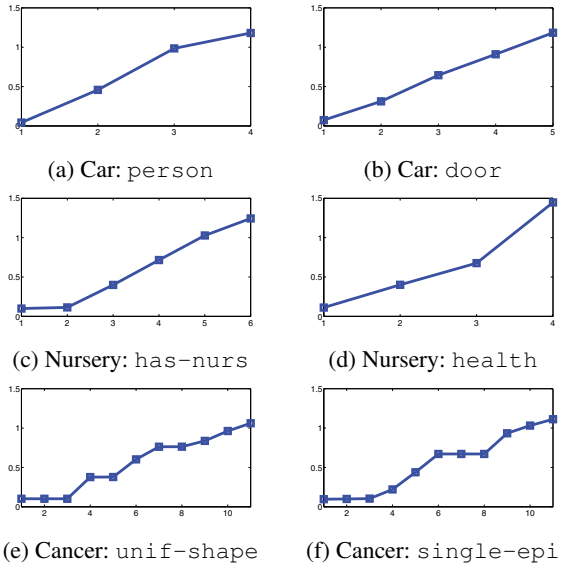
The Cancer dataset is for breast cancer diagnosis. Figure 1 (e) and (f) show the learned intervals on two features `unif-shape` (uniformity of cell shape) and `single-epi` (single epithelial cell size). Both features take ordinal scales from 1 to 10. The learned intervals clearly reveal the uneven distances between different ordinal scales.

Visualization of the learned distances

Once the distance metric is learned, we can use it to compute pairwise distances between samples. We further obtain 2-dimensional embeddings from the distance matrix

Table 2: kNN classification error on the datasets with discretized features

Method	Vehicle	Vowel	Segmentation	Magic
Real-Eucl	28.3±1.6	11.1±0.7	4.5±0.3	**17.0**± 0.1
Real-LMNN	27.8±0.6	11.3±0.5	4.4±0.2	17.1±0.2
Binary-Eucl	29.4±1.7	22.2±1.3	6.5±0.4	23.0±0.2
Binary-LMNN	**25.1**±0.7	12.8±0.4	3.6±0.2	19.1±0.1
Binary-Ord-Eucl	28.4±0.6	11.8±0.3	4.2±0.2	17.5±0.2
Binary-Ord-LMNN	25.2±0.8	10.9±0.5	**3.3**±0.2	17.1±0.1
Ex-Gower	30.8±0.7	22.6±0.5	7.4±0.2	20.2±0.2
Ex-Gower*	35.1±0.8	23.3±0.7	5.0±0.2	17.9±0.2
Thresh-Eucl	27.7±0.8	11.3±0.3	3.8±0.2	**17.0**±0.2
Ord-LMNN-Uni	25.4±0.8	10.8±0.5	3.4±0.2	**17.0**±0.2
Ord-LMNN-Beta	25.4±0.8	**10.6**±0.5	**3.3**±0.2	**17.0**±0.3
Ord-LMNN-RecBeta	25.5±0.8	**10.6**±0.5	**3.3**±0.2	**17.0**±0.3

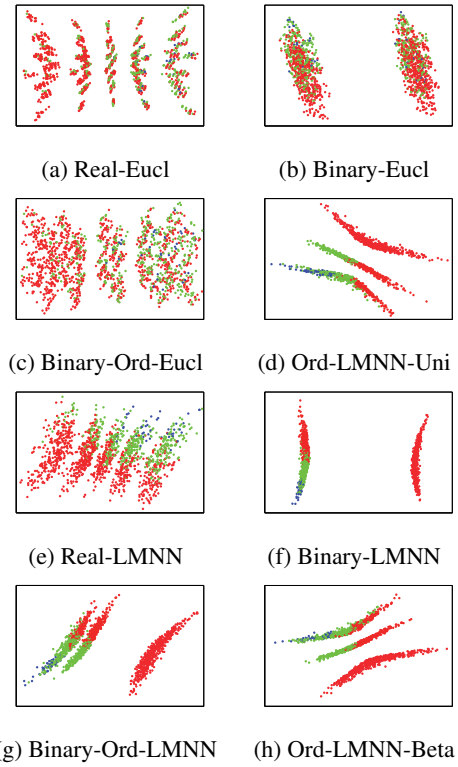


(a) Car: `person` (b) Car: `door`

(c) Nursery: `has-nurs` (d) Nursery: `health`

(e) Cancer: `unif-shape` (f) Cancer: `single-epi`

Figure 1: Learned intervals for selected features in different datasets. The x-axis represents the index of the thresholds for the corresponding feature, and the y-axis represents the learned value for the thresholds



(a) Real-Eucl (b) Binary-Eucl

(c) Binary-Ord-Eucl (d) Ord-LMNN-Uni

(e) Real-LMNN (f) Binary-LMNN

(g) Binary-Ord-LMNN (h) Ord-LMNN-Beta

Figure 2: Two-dimensional embeddings derived from the learned distances for Nursery dataset

using multidimensional scaling (Kruskal 1964). Figure 2 shows the embeddings from different methods on the Nursery dataset, where the embeddings are colored according to the class labels. We can see that Real-LMNN, Binary-LMNN, Binary-Ord-LMNN have better embeddings over Real-Eucl, Binary-Eucl, Binary-Ord-Eucl, respectively. The embeddings of our methods display the most clean class structures in which classes are more separable compared to those in other methods.

Conclusion

In this paper, we develop novel ways to learn distance metric on ordinal features. Our methods jointly learn the metric and latent interval variables from labled ordinal data. This provides important tools for ordinal data analysis which has many real-world applications. Our experiments show that the proposed methods not only improve the classification performance but also recover the latent thresholds. Our future work include extending our methods to the case where both input and output variables are ordinal, as well as experimenting on ordinal datasets with higher dimensionality.

Acknowledgements

This work is partially supported by an Alfred P. Sloan Research Fellowship, an ARO Young Investigator Award # W911NF-12-1-0241, ONR # N000141210066, and an NSF IIS Award #1065243.

References

Ananth, C. V., and Kleinbaum, D. G. 1997. Regression models for ordinal responses: a review of methods and applications. *International journal of epidemiology* 26(6):1323–1333.

1995. Generalization of the mahalanobis distance in the mixed case. *Journal of Multivariate Analysis* 53(2):332 – 342.

Boriah, S.; Chandola, V.; and Kumar, V. 2008. Similarity measures for categorical data: A comparative evaluation. In *SIAM International Conference on Data Mining*, 243–254.

Cardoso, J. S., and Pinto da Costa, J. F. 2007. Learning to classify ordinal data: The data replication method. *JMLR* 8:1393–1429.

De Leon, A. R., and Carriere, K. C. 2005. A generalized Mahalanobis distance for mixed data. *Journal of Multivariate Analysis* 92(1):174–185.

Fen Xia, Liang Zhou, Y. Y. W. Z. 2007. Ordinal regression as multiclass classification. *International Journal of Intelligent Control and Systems* 12(3):230–236.

Fitkov-Norris, E.; Vahid, S.; and Hand, C. 2012. Evaluating the impact of categorical data encoding and scaling on neural network classification performance: The case of repeat consumption of identical cultural goods. In *Engineering Applications of Neural Networks*, volume 311 of *Communications in Computer and Information Science*. Springer. 343–352.

Frank, E., and Hall, M. 2001. A simple approach to ordinal classification. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, 145–156. London, UK, UK: Springer-Verlag.

Gerd Ronning, M. K. 1996. Efficient estimation of ordered probit models. *Journal of the American Statistical Association* 91(435):1120–1129.

Goldberger, J.; Roweis, S.; Hinton, G.; and Salakhutdinov, R. 2004. Neighborhood Component Analysis. In *NIPS*.

Gower, J. C. 1971. A general coefficient of similarity and some of its properties. *Biometrics* 857–871.

Johnson, T. R. 2006. Generalized linear models with ordinally-observed covariates. *British Journal of Mathematical and Statistical Psychology* 59:275–300.

Kedem, D.; Tyree, S.; Sha, F.; Lanckriet, G. R.; and Weinberger, K. Q. 2012. Non-linear metric learning. In *NIPS*. 2573–2581.

Kottas, A.; Mller, P.; and Quintana, F. 2005. Nonparametric bayesian modeling for multivariate ordinal data. *Journal of Computational and Graphical Statistics* 14(3):pp. 610–625.

Kruskal, J. B. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29(1):1–27.

Kukuk, M. 1998. Indirect estimation of linear models with ordinal regressors: A monte carlo study and some empirical illustrations. Technical report, University of Tbingen, School of Business and Economics.

Kurczynski, T. W. 1970. Generalized distance and discrete variables. *Biometrics* 26(3):pp. 525–534.

Leon, A. R. D., and Carriere, K. C. 2007. General mixed-data model: extension of general location and grouped continuous models . *The Canadian Journal of Statistics* 35(4):533–548.

Li, L., and Lin, H.-T. 2006. Ordinal regression by extended binary classification. In Schlkopf, B.; Platt, J.; and Hoffman, T., eds., *NIPS*, 865–872. MIT Press.

Lichman, M. 2013. UCI machine learning repository.

McCullagh, P. 1980. Regression models for ordinal data. *J. Roy. Statist. Soc. B* 42:109–142.

O'Brien, R. M. 1979. The use of pearson's with ordinal data. *American Sociological Review* 44(5):851–857.

Olkin, I., and Tate, R. F. 1961. Multivariate correlation models with mixed discrete and continuous variables. *Ann. Math. Statist.* 32(2):448–465.

Paquet, U.; Thomson, B.; and Winther, O. 2012. A hierarchical model for ordinal matrix factorization. *Statistics and Computing* 22(4):945–957.

Podani, J. 1999. Extending gower's general coefficient of similarity to ordinal characters. *Taxon* 48(2):pp. 331–340.

Poon, W.-Y., and Wang, H.-B. 2012. Latent variable models with ordinal categorical covariates. *Statistics and Computing* 22(5):1135–1154.

Torra, V. C.; Domingo-Ferrer, J.; i Sanz, J. M. M.; and Ng, M. 2003. Regression for ordinal variables without underlying continuous variables. *Information sciences* (176):465–474.

Tran, T.; Phung, D. Q.; and Venkatesh, S. 2012. Cumulative restricted boltzmann machines for ordinal matrix data analysis. In *ACML*, volume 25 of *JMLR Proceedings*, 411–426. JMLR.org.

Webb, E. L., and Forster, J. J. 2008. Bayesian model determination for multivariate ordinal and binary data. *Computational Statistics & Data Analysis* 52(5):2632–2649.

Weinberger, K. Q., and Saul, L. K. 2009. Distance metric learning for large margin nearest neighbor classification. *JMLR* 10:207–244.

Weinberger, K. Q., and Tesauro, G. 2007. Metric learning for kernel regression. In *AISTATS*, 612–619.

Winship, C., and Mare, R. D. 1984. Regression models with ordinal variables. *American Sociological Review*.

Yuille, A. L.; Rangarajan, A.; and Yuille, A. 2002. The concave-convex procedure (cccp). *NIPS* 2:1033–1040.