Ordinal Hyperplane Loss

Bob Vanderheyden
Market Development & Insights
IBM
Brookhaven, GA, USA
rvanderh@us.ibm.com

Ying Xie
Dept. of Information Technology
Kennesaw State University
Kennesaw, GA, USA
ying.xie@kennesaw.edu

Abstract—**The problem of ordinal classification occurs in a large and growing number of areas. Some of the most common source and applications of ordinal data include rating scales, medical classification scales, socio-economic scales, meaningful groupings of continuous data, facial emotional intensity, facial age estimation, etc. The problem of predicting ordinal classes is typically addressed by either performing n-1 binary classification for n ordinal classes or treating ordinal classes as continuous values for regression. However, the first strategy doesn't fully utilize the ordering information of classes and the second strategy imposes a strong continuous assumption to ordinal classes. In this paper, we propose a novel loss function called Ordinal Hyperplane Loss (OHPL) that is particularly designed for data with ordinal classes. The proposal of OHPL is a significant advancement in predicting ordinal class data, since it enables deep learning techniques to be applied to the ordinal classification problem on both structured and unstructured data. By minimizing OHPL, a deep neural network learns to map data to an optimal space where the distance between points and their class centroids are minimized while a nontrivial ordinal relationship among classes are maintained. Experimental results show that deep neural network with OHPL not only outperforms the state-of-the-art alternatives on classification accuracy but also scales well to large ordinal classification problems.**

Keywords—ordinal hyperplane loss, ordinal classification, ordinal regression, deep learning, loss function, machine learning

I. Introduction

The problem of ordinal classification occurs in a large and growing number of areas. Some of the most common sources and applications of ordinal data are:

- Ratings scales (e.g. Likert scales), like customer satisfaction ratings, "promoter" ratings and quality ratings
- Medical classification scales (e.g. classification of disease stage/severity) and student performance (i.e., letter grades)
- Socio-Economic scale (e.g., high, medium and low)
- Meaningful groupings of continuous data (e.g., generational age groupings, grouping of noisy sensor data)
- Facial emotional intensity [1]
- Large storm severity ratings (e.g., Tropical Storms and Hurricanes)

Historically, data sources like surveys and medical ratings were relatively small in size, but this digitalized world has produced more and more truly big ordinal data sources, such as Amazon's purchase satisfaction surveys, Yelp's rating data, and electronic health records.

Ordinal data differ from nominal (unordered) data by providing additional information on the order of the classes, which leads to a different way to evaluate the results of classification. For instance, misclassifying a value of '3' as a value of '4' should be viewed as a "better" error than misclassifying it as a '5' for ordinal classification, although nominal classification treats these two error cases equally.

A popular strategy to address ordinal classification problem is to reduce the problem of ordinal classification to multiple binary classifications and then use machine learning methods such as Support Vector Machines (SVM) [2][3][4][5] or Gaussian Process [6] to perform those binary classifications. However, this strategy doesn't fully utilize the ordering information of classes. Furthermore, SVM or Gaussian Process based methods are not easily scalable to big data.

Another frequently used strategy for ordinal classification view ordered classes as integers and regression techniques to predict a continuous outcome [9]. However, this strategy assumes that equal "distances" between values have a consistent numerical meaning (i.e., all one unit differences having the same "meaning"). But this assumption is rarely true in ordinal data.

In recent years, deep learning has made breakthrough achievements on complex analytics problems with big data, such as image classification [23, 24], natural language processing [25, 26], and speech recognition [27]. However, to the best of our knowledge, there is no existing mechanism by which the learning power of deep neural network can be applied to ordinal classification problems.

The research goal of this work is to solve large-scale ordinal classification problems, such that the classification model can 1) establish and maintain the ordering of the classes without making assumption regarding distances among classes, 2) "pull" like samples together while "pushing" higher samples above the current class samples and "push" the lower class samples below the current class samples, 3) achieve higher classification accuracy than the state-of-the-art, 4) be scalable to very large classification problems, and 5) be applied to unstructured data such as images and text.

To achieve this research goal, we first proposed a novel loss function that is called Ordinal Hyperplane Loss (OHPL). OHPL is particularly designed for data with ordinal classes and enables deep learning techniques to be applied to the ordinal classification problems. Based on OHPL, we further design a deep learning strategy, by which a deep neural network learns to map data to an optimal space where the distance between points and their class centroids are minimized while a nontrivial ordinal relationship among classes are maintained. We also conducted experimental studies that demonstrated the deep learning strategy based on OHPL outperforms state-of-the-art alternatives on ordinal classification accuracies and are scalable to large ordinal classification problems.

The rest of the paper will be organized as follows. In section II, we report, in detail, our literature study. In section III, we will describe our proposed method, including Ordinal Hyperplane Loss (OHPL) and OHPL based deep learning strategy. Experimental studies on OHPL deep learning strategy will be provided in section IV. Finally, we conclude our paper in section V.

II. Literature Study

In 2016, Gutierrez, et al published an extensive examination of solutions to the Ordinal Classification/Regression problem [7], including benchmark performance metrics versus a set of standard datasets that were included in the work of Chu and Ghahramani [6]. In their review Gutierrez, et al grouped the existing top performing methodologies into three categories that address the Ordinal Classification problem: 1) Naïve Approaches, 2) Ordinal Binary Decompositions and 3) Threshold Models.

Naïve approaches use an appropriate simplifying assumption to cast the problem in such a manner that existing methodologies can be applied. For instance, if one assumes that the difference in classes is "close" to uniform they may transform the classes into sequential integers and apply regression analysis like ordinary least squares, neural nets or SVR. Cost sensitive methodologies which use different weights for different misclassification types also fall into this category [2]. For example, SVM with Ordered Partitions (SVMOP) uses class differences as weights, in an effort to not only provide correct classification, but to encourage misclassifications that are close in class number to the actual class [8].

The fundamental basis of binary decomposition is to recast the problem as a binary classification. The problem may be posed by comparing pairs of ordinal values with the higher value being assigned a value of 1 and then using either a single or multiple binary classification models. The earliest ordinal binary decomposition approaches used Ordinal Logistic Regression [9], which estimates binary probability for class ordering. More recent binary decomposition strategies using machine learning approaches like SVM algorithms create individual binary classifiers, combined with ensemble strategy that is based on the output of the binary classifiers. Deep Neural Nets allow of the output of multiple estimates that may be used to create class probabilities for all classes. Some researchers endeavored to use non-parallel hyperplanes in an SVM framework, but at a high cost of increased model complexity.

Threshold models include a large number of methodologies including:

1. SVMs: Chu & Keerthi developed two SVM algorithms that specifically address the ordinal classification problem through the estimated multiple hyperplanes that maintain the sequential ordering of the classes [4]. While successful in application to small datasets, SVMs are known to become impractical when data sets increase to above 100K records.
2. Boosting Models: RankBoost [9] attempts to improve a set of confidence functions, that maximize an ensemble of binary classifiers. Similarly, ORBoost [10] applies the same concepts to develop improved performance from ordinal regression models.
3. Gaussian Process: GPOR [6] uses a Bayesian framework to model a latent function via Gaussian Processes. Prior and posterior probabilities for class membership are estimated for a set of latent functions of the input features.

In late 2016, Hamsici and Martinez proposed a SVM based algorithm that attempted to maximize the margins between adjacent classes [11]. Their algorithm is similar to the one that was proposed by Keerthi and Chu [4], but with a notable and meaningful difference that their algorithm doesn't assume equal margins between adjacent classes. In addition, their algorithm includes weight parameters, which enable the prioritization of one of or more of the individual algorithms over others. This prioritization weighting allows one to focus on a specific pair of ordinal classes.

In 2017, Wang, et al used a nonparallel hyperplane assumption for the development of a specialized SVM algorithm to address the Ordinal classification problem [12]. For k ordinal classes, their algorithm estimates k-1 hyperplanes. For each, they include constraints that ensure that like labelled samples are within a prescribed margin of the hyperplane, while unlike labelled samples are one or more units away. They also include constraints to ensure the ordering of the hyperplanes reflect the ordering of the classes.

These algorithms provide a mixed performance across the standard test data sets that are used to benchmark performance of ordinal classifiers. Many are benchmarked using 20 or more small datasets, with performance that represents modest improvements. While these incremental improvements are impressive, they are being benchmarked against current "best in breed" classifiers, so as a rule, it's rare to find one that outperforms best benchmark classifier by 10% or more (in terms of decline in classification error).

In February 2018, Nguyen et al, incorporated triplet loss based constraints to what is similar as SVM solution [3]. Their algorithm employs triplet loss based constraints on local clusters of data points. The researchers produced both a linear version of their algorithm and a version that employs the kernel trick to produce a nonlinear mapping of the data into a higher dimensional space. Given the researcher's stated algorithm compute cost of $O(n^3)$, their solution while successful with relatively small datasets, may not be viable for large datasets.

"Triplet Loss" is a term that was first used in the FaceNet solution to the re-identification problem [13]. In developing FaceNet, Schroff et al leveraged the foundational work in Large Margin Nearest Neighbor (LMNN) Classification published by of Weinberger and Saul [14]. In [15], a framework of triplet loss was further proposed to provide a mechanism for applying a distance comparison between points without requiring the underlying distance assumptions for regression analysis. This framework of triplet loss makes it well suited to the ordinal classification problem, but triplet loss itself cannot be used, because it doesn't guarantee the ordering of classes.

III. Proposed Method

Given a data x that is not separable in its original space, we would like to find a transformation $\varphi(x)$ to map data to a high-dimensional feature space that is optimal for ordinal classification. Traditional kernel machines use a kernel function that is either pre-defined or pre-selected by the user to implicitly map the data from its original space to a high- dimensional feature space. However, this implicit mapping through kernel function does not guarantee that the mapped space is optimal for decision making [22]. In this research, we aim to design a new method that is able to automatically learn a transformation $\varphi(x)$ towards a learning objective that directly reflects an optimal distribution of the data in the mapped space with respect to ordinal classification.

A. *Geometric Illustration of an Optimal Data Distribution*

Given a data x that is not separable in its original space as shown in Fig. 1 (a), a transformation $\varphi(x)$ maps data to a feature space as shown in Fig. 1 (b). Now the question is how we evaluate the quality of the data distribution in this mapped space with respect to ordinal classification. For nominal classification, we can use measures based on intra-class density/inter-class distance to describe the quality of the data distribution produced by $\varphi(x)$. However, this type of measures does not work well for ordinal classification for the following reasons. First, increasing the inter-class distance does not guarantee the ordinal relationship is kept among classes; second, moving instances closer to the center of their own classes does not necessarily yield a better ordinal classification model if the moving is primarily along the dot-lines or dash-lines as shown in Fig. 1 (c).

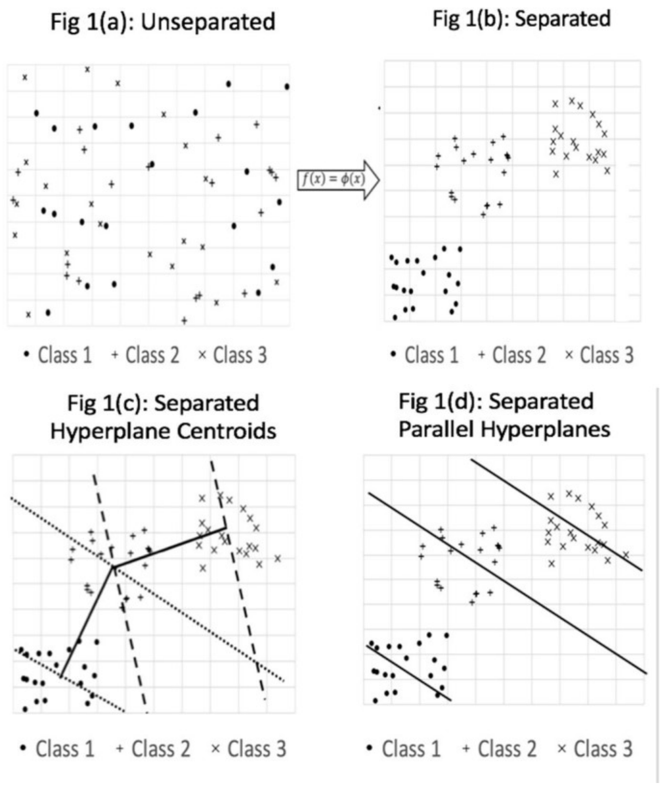


Fig 1(a): Unseparated
Fig 1(b): Separated
$f(x) = \phi(x)$
• Class 1 + Class 2 × Class 3
• Class 1 + Class 2 × Class 3
Fig 1(c): Separated Hyperplane Centroids
Fig 1(d): Separated Parallel Hyperplanes
• Class 1 + Class 2 × Class 3
• Class 1 + Class 2 × Class 3

Fig. 1 Geometric Illustration of an Optimal Data Distribution

Therefore, in order to describe an optimal data distribution in the mapped space towards ordinal classification, we propose to use a group of parallel hyperplanes to represent classes as shown in Fig. 1 (D). Now, we intuitively call a data distribution optimal for ordinal classification, if we can find a group of parallel hyperplanes in the mapped space, such that 1) if $class_i < class_j$, then the hyperplane for class i is lower than the hyperplane for class j in the mapped space for all i and j; and 2) an instance is closer to the hyperplane of its class than to any other hyperplane. If a transformation $\varphi(x)$ maps the data to a feature space, where some of the above criteria are not satisfied, this transformation brings loss. In the following subsection, we will mathematically define a loss function that is called Ordinal Hyperplane Loss to quantify such a loss for a data distribution that is produced by a transformation.

B. *Mathematical Definition of Ordinal Hyperplane Loss*

As the name implies that Ordinal Hyperplane Loss (OHPL) uses ordered linear hyperplanes, as the basis for calculating the loss for data distribution in the mapped space. The loss function is designed to utilize simple scalar distance calculations, combined with a standard application of large margin loss. The loss function enables the use of stochastic gradient descent, in optimizing data transformations.

A linear hyperplane can be expressed as a simple mathematical equation of the form: $w^T x + c = 0$, where w and x are vector valued and c is a scalar constant. A set of parallel hyperplanes of this form differ in their c values. As a direct consequence, the 'distance' between two parallel hyperplanes can be defined to be the absolute value of the difference in their values divided by $/w/$. Given w, we

further denote the hyperplane that goes through the *ith* data point x_i as

$$w^T x + c_i = 0 \quad (1)$$

then bring $\boldsymbol{x_i}$ into (1), we have

$$w^T x_i + c_i = 0 \ (2a)$$

$$c_i = -w^T x_i \ (2)$$

further bring (2) into (1), we have the expression of the hyperplane that goes through $\boldsymbol{x_i}$

$$w^T x - w^T x_i = 0 \ (3)$$

Given the hyperplanes going through each data point in a feature space, we can now represent a class in that feature space by calculating its **Hyperplane Centroid (HC)**. For instance, the hyperplane centroid for the *kth* class, denoted as HC_k, can be expressed as

$$HC_k : w^T x - \frac{1}{n_k} \mathbf{L}_{y_i=k} w^T x_i = 0 \ (4)$$

Given the definition in (4), all ordinal classes are represented as a group of hyperplane centroids, which are parallel to each other, in the feature space. Now we define OHPL, such that we can quantify the loss in a data distribution that is produced by a data transformation $\varphi(x)$ with respect to a given vector $\boldsymbol{w}$. According to the intuitive criteria of an optimal data distribution that are described in section 3.1, OHPL consists two components, namely Hyperplane Centroid Loss and Hyperplane Point Loss. Hyperplane Centroid Loss reflects the loss caused by non-optimal ordering of Hyperplane Centroids per the ordinal relationship of the classes, while Hyperplane Point Loss reflects the loss caused by non-optimal relationship between individual data points and the hyperplane centroids of their classes.

1) Hyperplane Centroid Loss(HCL)

Hyperplane Centroid Loss (HCL), the first component of OHPL, ensures that the hyperplane centroids are properly ordered, per the ordering of the classes. This ordering can be expressed as a difference in adjacent HCs. If the adjacent HCs are properly ordered, then the transitive property ensures that all HC's are properly ordered. Therefore, we require that the HCs for adjacent classes *k* and *k+1* adhere to: $HC_k - HC_{k+1} > 8$, for $\delta > 0$ This means, if HC_{k+1} is at least 8 from HC_k, then the ordering is correct with sufficient distance between the adjacent classes. Since the difference is unbounded from above, this formulation doesn't introduce a distance assumption. Given adjacent classes *k* and *k+1*, and $\delta > 0$ the Hyperplane Centroid Loss contribution of HC_k relative to HC_{k+1} is defined as: $max(HC_k - HC_{k+1} + 8, 0)$. Finally, for

the k ordinal class problem, the Hyperplane Centroid Loss (HCL) is defined as:

$$HCL = \mathbf{L}_{i=1}^{k-1} max(HC_i - HC_{i+1} + 8, 0) \quad (5)$$

2) Hyperplane-Point Loss (HPL)

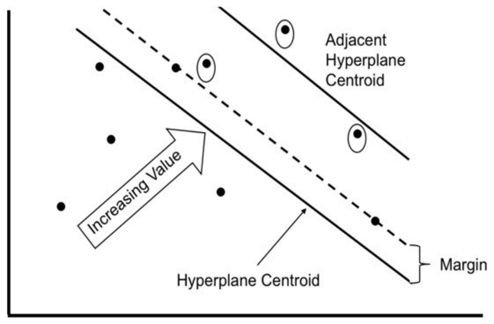
The second component of OHPL is "Hyperplane-Point Loss" (HPL). In calculating this loss component, individual data points are compared to a specific set of Hyperplane Centroids, to access the point's contribution to the loss of the data distribution. HPPL is actually, the sum of two analogous loss functions, that work in different "directions" a la the formulation of (5).

For the points, in a given class, if we "look" in the "increasing" direction (direction of larger ordinal class value), we only want the points that are higher than the HC for the point to potentially contribute to the loss (those below will be examined later). For points that are above their HC, but are already sufficiently close to their HC, there isn't much benefit in drawing them closer, so we want their loss contribution to be zero. Therefore, the HPL uses a margin to ensure that points that do not contribute to loss are closer to their HC than the

midpoint between the HC. In Fig 2 (a), below, the circled points are

higher than the margin above its HC, so they contribute to the total HPL value. Note that the dotted margin line/threshold is closer to the

HC, than to the adjacent HC.



(a) HPL Increasing Direction

Similarly, when we look in the decreasing direction, points that are further from their HC than the margin, will contribute to the HPL total. In Fig 2 (b), below, the three circled points contribute to HPL.

The two components of the HPL (an increasing and a decreasing) that are summed to arrive at the total HPL. Formally, given a dataset S, let γ to be the proportion of distance between adjacent HCs, HC be the hyperplane centroid that represents the class that $x_i \in S$ belongs to, HC_{+1} is the higher hyperplane centroid that is adjacent to HC, and HCL^+ be the HPL for the point $x_i \in S$ in the increasing direction, then we have:

i

$$0.5 < y < 1.0$$

$$point\ margin = y(HC_{+1} - HC)$$

$$HCL^+ = max((f(x_i) - HC) - (HC_{+1} - HC) + y(HC_{+1} - HC), 0)$$

i

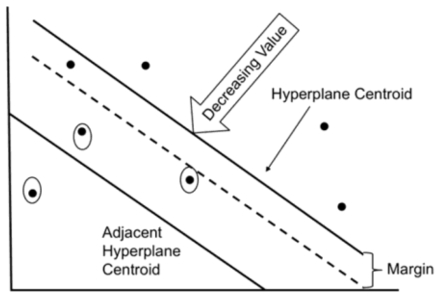
$$= max(f(x_i) - yHC - (1 - y)HC_{+1}, 0)$$

Similarly, in the decreasing direction,

$$HPL^{-} = max(yHC - f(x_i) + (1-y)HC_{-1}, 0)$$

Then, the overall HPL will be the aggregation of $(HPL^{+} + HPL^{-})$ over all data points in S.

$$HPL = \sum_{x_i \in S} HPL^{+} + HPL^{-} \quad (6)$$



(b) HPL Decreasing Direction

Fig. 2 Computing HPL in Two Directions

3) Ordinal Hyperplane Loss (OHPL)

Finally, the Ordinal Hyperplane Loss (OHPL) is defined as the weighted aggregation of HCL and HPL, as shown below, where a �1 reflects the importance of HCL in OHPL with respect to HPL.

$$OHPL = aHCL + HPL \quad (7)$$

C. OHPL-Net: OHPL Deep Learning Strategy

Given the definition of OHPL in (17), this section describes a deep learning strategy for ordinal classification based on OHPL. Figure 3 shows a simple deep neural network (DNN) model that represents a non-linear transformation ϕ that maps input data from their original space to a n-dimensional space. We further add the last layer $w^T</(x)$ on the top of the transformation $</(x)$. Then we use the weights of the last layer, namely $\boldsymbol{w}$, to define m parallel hyperplanes to represent m ordinal classes, such that the kth class will be represented by the hyperplane whoes expression is shown in (4).

Based on the hyperplane representations of the ordinal classes, we can calculate the Ordinal Hyperplane Loss (OHPL) based on the formula (7). Then the DNN can learn both an optimal transformation $<I$ and an optimal vector $\boldsymbol{w}$ by minimizing the OHPL (recall that $\boldsymbol{w}$ determines the direction of those parallel hyperplanes in the feature space that is mapped by $<I$).

In our practical algorithm design, the HCL component of OHPL is estimated on the entire dataset, while the HPL component is applied to batches. To ensure that the ordering relationship is achieved as early as possible and maintained over the entire training process, a large weight value $\alpha > 10$ is used to prioritize HCL loss over point loss. The algorithmic description of the OHPL deep learning strategy is given as follows.

OHPL Deep Learning Algorithm

```
For parameters: $O$ ordinal classes
h – number of hidden layers
$l_k$ – number of nodes in each layer
$w$ – prioritization wgt for HCL
$lr$ – learning rate
$m$ – HC margin
$\gamma$ – point margin proportion
$bs$ – batch size
**Input:** Rescaled training data $\{(x_i, y_i) | i = 1, \dots, n\}$
Parameters $h, l_k, wgt, lr, \{l_k = 1,\dots, h\}$
**Begin:**
1) Randomize node weights (W) and bias (b) values
2) While not converged do
    HPPL = 0, HCL = 0, OHPL = 0
    Select mini-batch
        Calc mini-batch Hyperplane Centroids
        Calc HCL
        Calc HPPL for batch
        Update OHPL
        Calc SGD*
        Update $\boldsymbol{W}$ and $\boldsymbol{b}$
    Repeat until training sample exhausted
Check convergence
**End:** Output **W** and **b**
```

* – Stochastic Gradient Descent

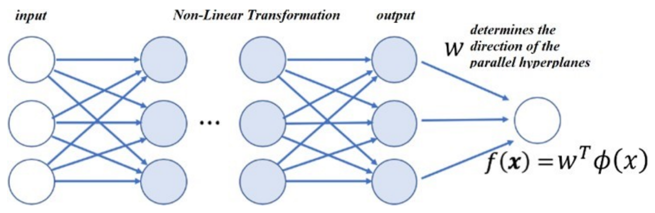

input
Non-Linear Transformation
output
W
determines the direction of the parallel hyperplanes
...
$f(x) = w^T\phi(x)$

Fig 3: OHPL Deep Learning Strategy Illustration

In order to facilitate the application of OHPL deep learning strategy on different types of data for ordinal classification, we further brand this strategy as OHPL-Net, a deep architecture that users can directly apply to their ordinal classification problems. An OHPL-Net contains two components. The first component is called $</$layers, which are fully connected deep nets that represents a non-linear transformation of the input data. The second component is called Hyperplane layer, which is a one-layer one-output neuron network representing the direction of Hyperplane Centroids. Again OHPL-Net uses OHPL to learn optimal $</$and optimal parallel hyperplanes. If users' classification tasks involve unstructured data, such as medical diagnosis of Alzheimer's disease (mild, moderate and severe) based on MRI images or anger level detections based on tweets, OHPL-Net can be put upon those deep neuron architectures that are built on specific unstructured data, such

as Convolutional Neuron Network (CNN) [23] [24] on image data and Recurrent Neural Network (RNN) [25] [26] [27] on text data.

IV. Experimental Results

OHPL was tested against seven ordinal classification datasets that are found in a number of related studies, including CPU Small [16], Census 10 [16], ERA (Employee Rejection/Acceptance) [17], LEV (Lecturers Evaluation) [18], SWD (Social Worker Decisions) [19], Cars [20], and Red Wine [20]. Characteristics of the seven data sets are given in Table 1.

Table 1: Test Dataset Key Characteristics

Dataset	# Records	# Features	# Classes	Class Distribution
CPU Small	8,192	12	10	~820 per class
Census 10	22,784	16	10	~2,278 per class
Cars	1,728	6	4	(1,210, 384, 69, 65)
Wine-Red	1,599	11	6	(10, 53, 681, 638, 199, 18)
ERA	1,000	4	9	(92, 142, 181, 172, 158, 118, 88, 3, 18)
LEV	1,000	4	5	(93, 280, 403, 197, 270)
SWD	1,000	10	4	(32, 352, 399, 217)

A. Assessment Measures

Mean Zero-One Error (MZE) is used to test classification of nominal data. This measure reports the proportion of misclassifications when scoring the validation samples, which can be computed as:

$$MZE = \frac{1}{N}\sum(y_i \neq \hat{y}_i) = 1 - accuracy$$

As discussed earlier, ordinal data differs from nominal data and shares some characteristics with continuous data while also differing from continuous data. One of the key similarities with continuous data is the concept of being "close" if the prediction is incorrect. As such, Mean Absolute Error (MAE) may be a more meaningful way to access model performance. As a minimum, it's a powerful way to distinguish among models that have comparable MZE performance. MAE can be computed as:

$$MAE = \frac{1}{N}\sum|y_i - \hat{y}_i|$$

Table 1 illustrates the fundamental difference between MAE and MZE, for the OHPL results when applied to the Social Work Decisions dataset. A standard methodology to assess classifier performance is the use of a "confusion" matrix. The basic principle is to use the classifier to score a dataset that has known labels, giving each record an actual and a predicted class value. The actual values correspond to the rows of the matrix and the predicted classes are represented in the columns. Every record is an ordered pair that occurs within the matrix. Cells of the matrix are filled with counts of the corresponding ordered pairs. Assuming that the row and column sequence is the same, then the diagonal (darkest colored cells in the matrix below) represents the correctly classified counts, which sum to the MZE value, before dividing by the total number of records.

Table 2. Social Work Decisions OHPL Confusion Matrix

	Predicted			
Actual	2	3	4	5
2	16	15	4	0
3	29	202	100	20
4	6	79	237	83
5	0	7	79	123

As you move further from the diagonal of the matrix, the values get lighter in color (further in color from the diagonal). This color change represents increasing error, in the classification and the lighter the color, the higher the error for points that are represented in the cells. An ideal classifier, that isn't a perfect classifier, will have zeros, in the three lightest colors in Table 2.

B. Benchmark Algorithms

The following benchmark algorithms are included in our experimental studies: 1) Support Vector Machines with Ordered Partitions (SVMOP) [8], 2) GPOR (Gaussian Process for Ordinal Regression) [6], 3) ORBALL (Ordinal Regression Boosting with All margins) [10], and 4) LODML (linear classifiers using triplet loss constraints on Mahalanobis distance within an optimization framework) [3]. To ensure a justifiable benchmark comparison, a 5-fold cross validation was used when testing OHPL based classifiers.

C. Performance Comparisions

Table 3 shows the comparison results on MZE. As can be seen, OHPL based classifier has the lowest average MZE across the 7 data sets. Furthermore, OHPL achieves lowest MZE on 3 out of 7 data sets. Especially on the two largest data sets, CPU Small and Census 10, OHPL outperforms the second best by 24.8% and 13% respectively. Table 4 shows the comparison results on MAE. As can be seen, OHPL based classifier has the lowest average MAE across the 7 data sets. Furthermore, OHPL achieves lowest MAE on 4 out of 7 data sets. Especially on the two largest data sets, CPU Small and Census 10, OHPL outperforms the second best by 25.8% and 40.9% respectively. Therefore, for these larger datasets, OHPL represents a significant improvement over the best existing algorithms.

D. Scalling to Big Data

To demonstrate that OHPL can scale to large datasets, a collection of "synthetic" datasets was created, from the largest of the standard datasets that are used to benchmark ordinal classification/regression algorithms.

From the Chu and Ghahramani research [6], Census 10 dataset is among the largest for which benchmark results are available. A standard benchmark assessment for this dataset is included in the next section. For the purposes of assessing the scaling of OHPL, the data set was split, 80:20, into single training and test datasets. The synthetic datasets were created by replicating the training data. In each set, the 1st replica is the original training dataset. For each subsequent, replica a small amount of random noise was added to each data value. The original training sample contains just 20K records. The largest synthetic dataset contains just over 500K records. Time to algorithm execution times are reported in Figure 6 below.

V. Conclusion and Future Remarks

OHPL directly addresses the unique requirements of the ordinal classification by using a point specific large margin loss function to group classes, while directly adhering to the ordinal information that are represented in the data. DNN's built using OHPL perform on par with existing high performing ordinal classifiers on small datasets, while demonstrating vastly improved results on larger standard benchmark large datasets. Through the application to a very large (500K+ records) synthetic dataset example, OHPL was demonstrated to effectively classify ordinal data.

From a technical perspective, immediate plans include continue to develop the algorithm, to gain an additional insight into the impact of DNN structure (assess whether or not guidelines for number of layers and number of nodes per layer to achieve optimal performance can be established). In addition, early testing suggests that allowing some "flexibility" regarding the minimum margin between hyperplane centroids may provide the benefit of faster convergence to an optimal solution.

As mentioned earlier, OHPL uses a DNN with a loss function that has been specifically developed for the Ordinal Classification problem. As such, it can handle large datasets (200K+ records). Testing against large datasets is critical, but ideally after some additional work to improve algorithm efficiency. Experimentation with algorithmdifferent/additional strategies, to improve algorithm speed would be important when applying it to large datasets.

Table 3. MZE Comparison

	Algorithm				
Dataset	*SVMOP*	*GPOR*	*ORBALL*	*LODML*	*OHPL*
CPU Small	0.631	0.588	0.654	0.569	**0.428**
Census 10	0.771	0.749	0.774	0.737	**0.641**
Cars	0.003	0.037	**0.012**	0.028	0.024
Wine-Red	0.358	0.394	**0.334**	0.432	0.431
ERA	0.745	**0.712**	0.76	0.828	0.744
LEV	**0.367**	0.388	0.391	0.49	0.399
SWD	0.424	**0.422**	0.439	0.526	**0.422**
Average	*0.471*	*0.47*	*0.481*	*0.516*	***0.441***

Table 4. MAE Comparison

	Algorithm				
Dataset	*SVMOP*	*GPOR*	*ORBALL*	*LODML*	*OHPL*
CPU Small	1.680	1.565	1.590	1.656	**1.161**
Census 10	2.127	2.190	1.951	2.174	**1.153**
Cars	**1**	1.081	**1**	1.053	**1**
Wine-Red	1.145	**1.066**	1.108	1.123	1.159
ERA	1.664	1.742	1.645	1.816	**1.034**
LEV	1.090	**1.082**	1.100	1.097	1.098
SWD	1.061	**1.043**	1.048	1.062	1.158
Average	*1.395*	*1.396*	*1.349*	*1.426*	***1.109***

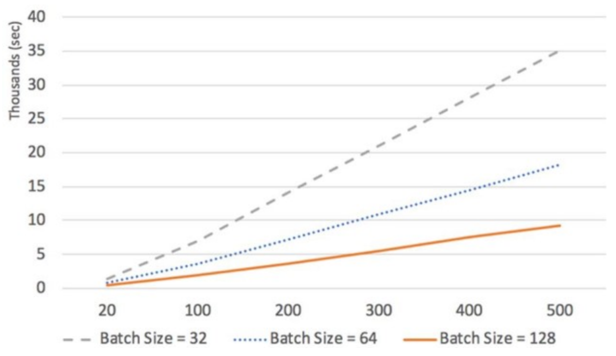


Fig 6: Time to Complete 500 Epochs by Number of Records (K records)

References

[1] M. Kim and V. Pavlovic, "Structured output ordinal regression for dynamic facial emotion intensity prediction," in Proceedings of the 11th European Conference on Computer Vision, Heraklion, Crete, 2010.

[2] C.-S. C. a. Y.-P. H. K.-Y. Chang, "Ordinal Hyperplanes Ranker with Cost Sensitivities for Age Estimation," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, , 2011.

[3] B. Nguyen, Morell, Carlos and De Baetsa, Bernard , "Distance metric

learning for ordinal classification based on triplet constraints," Knowledge-Based Systems, no. 142, p. 17–28, 2018.

[4] W. Chu and S. S. Keerthi, "New approaches to support vector ordinal regression," in Proceedings of the 22nd International Conference on Machine learning, Bonn, Germany, 2005.

[5] R. Herbrich, T. Graepel and K. Obermayer, "Support Vector Learning for Ordinal Regression," in Ninth International Conference on Artificial Neural Networks, ICANN 99, Edinburgh, UK, 1999.

[6] W. Chu and Z. Ghahramani, "Gaussian Processes for Ordinal Regression," Journal of Machine Learning Research, no. 6, p. 1019–1041, 2005.

[7] P. Gutiérrez, M. Pérez-Ortiz and J. Sánchez-Mone, "Ordinal regression methods: survey and experimental study," IEEE Trans. Knowl. Data Eng. 28, no. 1, p. 127–146, 2016.

[8] E. Frank and M. Hall, "A Simple Approach to Ordinal Classification," in 12th European Conference Machine learning: ECML, Freiburg, Germany, 2001.

[9] Y. Freund, R. Iyer, R. E. Schapire and Y. Singer, "An Efficient Boosting Algorithm for Combining Preferences," Journal of Machine Learning Research, vol. 4, no. 4, pp. 933-969, 2003.

[10] . H.-T. Lin and L. Li, "Large-margin thresholded ensembles for ordinal regression: Theory and practice," in Proceedings 17th Algorithmic Learning Theory Int. Conf, Barcelona, Spain, 2006.

[11] O. C. Hamsici and A. M. Martinez, "Multiple Ordinal Regression by Maximizing the Sum of Margins," IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, vol. 27, no. 10, pp. 2072-2083, 2016.

[12] Y. S. L. N. a. Y. T. H. Wang, "Nonparallel Support Vector Ordinal Regression," IEEE TRANSACTIONS ON CYBERNETICS,, vol. 47, no. 10, pp. 3306-3317, 2017.

[13] Schroff, Florian ; Kalenichenko, Dmitry; Philbin, James ;, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, 2015.

[14] K. Q. Weinberger and L. K. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification," Journal of Machine Learning Research, no. 10, pp. 207-244, 2009.

[15] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe and S. Singh, "No Fuss Distance Metric Learning using Proxies," arXiv:1703.07464v3, 2017.

[16] W. Chu, "BENCHMARK of ORDINAL REGRESSION: datasets and results," UCL Gatsby Computational Neuroscience Unit, 11 April 2004. [Online]. Available: http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html. [Accessed 1 April 2018].

[17] A. B. David, "View datasets-arie_ben_david ERA (public)," Business Administration School Tel Aviv University, 06 11 2010. [Online]. Available: http://mldata.org/repository/data/viewslug/datasets-arie_ben_david-era/. [Accessed 1 4 2018].

[18] A. B. David, "View datasets-arie_ben_david LEV (public)," Holon Inst. of Technology, 06 11 2010. [Online]. Available: http://mldata.org/repository/data/viewslug/datasets-arie_ben_david-lev/. [Accessed 01 04 2018].

[19] A. B. David, "View datasets-arie_ben_david SWD (public)," Business Administration School Tel Aviv Univerity, 06 11 2010. [Online]. Available: http://mldata.org/repository/data/viewslug/datasets-arie_ben_david-swd/. [Accessed 01 04 2018].

[20] D. D. Taniskidou and E. Karra, "UCI Machine Learning Repository," University of California, School of Information and Computer Science, 2017. [Online]. Available: http://archive.ics.uci.edu/ml. [Accessed 15 March 2018].

[21] J. D. M. Rennie, "Ordinal Logistic Regression," MIT, 16 February 2005. [Online]. Available: http://people.csail.mit.edu/jrennie/writing/olr.pdf. [Accessed 20 March 2018].

[22] L. Le and Y. Xie "Deep Embedding Kernel," arXiv:1804.05806, 2018

[23] L. Yann, B. Yoshua & H. Geoffrey, "Deep Learning", Nature, Vol. 521, 2015

[24] S. Christian, I. Sergey, V. Vincent & A. A. Alexander, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning", arXiv:1602.07261, 2017

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997

[26] K. Cho, B. van Merrienboer, C. Gulcehre, et. al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," arXiv:1406.1078, 2014

[27] A. Graves, A. Mohamed, G. Hinton, "Speech Recognition with Deep Recurrent Nueral Networks," Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, Canada, May 2013