Ordinal regression based on data relationship

Liu, Yanzhu

2019

Liu, Y. (2019). Ordinal regression based on data relationship. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/105479>

<https://doi.org/10.32657/10220/47844>

(On the Spine)

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

**ORDINAL REGRESSION BASED ON DATA
RELATIONSHIP**

LIU YANZHU

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2019

ORDINAL
REGRESSION
BASED ON DATA
RELATIONSHIP

LIU YANZHU

2019

ORDINAL REGRESSION BASED ON DATA RELATIONSHIP

LIU YANZHU

LIU YANZHU

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

14, March, 2019

.....
Date

Liu yanzhu

.....
LIU YANZHU

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

14, March, 2019

.....
Date

.....
Adams Wai Kin Kong

Authorship Attribution Statement

This thesis contains material from 3 papers published in the following peer-reviewed conferences in which I am listed as an author.

Chapter 3 is published as Yanzhu Liu, Xiaojie Li, Adams Wai Kin Kong, and Chi Keong Goh. Learning from small data: A pairwise approach for ordinal regression. In Proceedings of 2016 IEEE Symposium Series on Computational Intelligence (SSCI'16), December 2016, pp. 1-6.

The contributions of the co-authors are as follows:

- I proposed and implemented the method, performed the experiments, and wrote the manuscript draft.
- Xiaojie Li revised the manuscript.
- Prof. Adams discussed with me about the design details and revised the manuscript.
- Dr. Chi Keong suggested several applications about this topic.

Chapter 4 is published as Yanzhu Liu, Adams Wai Kin Kong, and Chi Keong Goh. Deep ordinal regression based on data relationship for small datasets. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17), August 2017, pp. 2372-2378.

The contributions of the co-authors are as follows:

- I proposed the method, performed the experiments, and wrote the manuscript draft.
- Prof. Adams revised the manuscript together with Dr. Chi Keong.

Chapter 5 is published as Yanzhu Liu, Adams Wai Kin Kong, and Chi Keong Goh. A Constrained Deep Neural Network for Ordinal Regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18), June 2018, pp. 831-839.

The contributions of the co-authors are as follows:

- I proposed the method, performed the experiments, and wrote the manuscript draft.
- Prof. Adams revised the manuscript together with Dr. Chi Keong.

14, March, 2019

.....
Date

Liu yanzhu

.....
LIU YANZHU

Acknowledgements

I would like to express my most sincere gratitude to my supervisor, Professor Adams Wai Kin Kong. Seven years ago, it was his enthusiasm, confidence and diligence on research that let me know first time what is a true researcher and what is innovation, which opened me a window towards the academic world that I had never thought before. Four years ago, it was his trust that gave me courage to bring my ten months old daughter together to start my PhD journal. During these four years, he guided me by his immense knowledge and provided me the maximum freedom to dive into the research topic I like. When the experiments of my idea failed a hundred times, his voice always encouraged me that never give up – “If you stick with one thing for ten years, you will be the best one in that area.” Professor Kong is not only my supervisor during my PhD study, but also the mentor of my life.

I am grateful to my co-supervisor, Dr. Chi Keong Goh. His advices usually inspired me from a different point of view. I also thank my friends and classmates from Rolls-Royce@NTU Corporate Lab and Cyber Security Lab, Pengfei, Shitala, Xiaojie, Hengyi, Chaoying, Xingpeng, Frodo, Guozhu, Peicong and Zehong. A lot of insightful suggestions came up from the discussions with them.

I want to express warm thanks to my family. My parents gave me unconditional trust and support. It is unforgettable that they commuted between Singapore and China every two months to help me take care my little daughter in the hardest first year. I cannot imagine how to start my first step of study without their help. Also, I don't know which words can express my thanks to my husband, Shifeng. His support afforded me the opportunity to explore, to experience, to grow up and find the best version of myself. Lastly, I want to thank my daughter, Zijin. No matter how much pressure I faced, when she wiggled her little body and put her cheek against mine, the light of hope was kindled immediately in my heart. I know it is love that pushes me forward, to adventure in the wonderland.

Contents

Summary	ix
1 Introduction	1
1.1 Motivation	1
1.2 Research Objectives	4
1.3 Contributions	5
1.4 Organization	6
2 Background	8
2.1 Definition of Ordinal Regression	8
2.2 Ordinal Regression Approaches	10
2.2.1 Max-margin Based Approaches	12
2.2.2 Projection Based Approaches	14
2.2.3 Ensemble Approaches	16
2.2.4 Probabilistic Approaches	17
2.2.5 Neural Network Approaches	18
2.2.6 Deep Learning Approaches	21
2.3 Issues in Ordinal Regression and Solutions	24
2.3.1 Incremental Ordinal Regression	24
2.3.2 Semi-supervised Ordinal Regression	25
2.3.3 Others	26

2.4	Applications of Ordinal Regression	27
2.5	Evaluation Metrics	30
3	Learning from Small Scale Data: A Pairwise Approach for Ordinal Regression	32
3.1	Overview	32
3.2	A Pairwise Approach for Ordinal Regression	34
3.2.1	A Pairwise Framework	35
3.2.2	A Pairwise SVM	37
3.2.3	A Decoder	39
3.3	Reducing the Training Complexity	43
3.4	Experimental Results	50
3.5	Summary	54
4	Deep Ordinal Regression Based on Data Relationship for Small Datasets	56
4.1	Overview	56
4.2	A Convolutional Neural Network for Ordinal Regression	58
4.2.1	The Proposed Approach	58
4.2.2	The Architecture of CNNOR	62
4.2.3	The Decoder Based on Majority Voting	64
4.3	Evaluation	65
4.3.1	Results on the Historical Color Images Dataset	65
4.3.2	Results on the Image Retrieval Dataset	69
4.4	Summary	71
5	A Constrained Deep Neural Network for Ordinal Regression	73
5.1	Overview	73
5.2	The Proposed Algorithm	76
5.2.1	The Proposed Optimization Formulation	76

5.2.2	The Proposed CNN based Optimization	78
5.2.3	Scalability of the Proposed Algorithm	81
5.3	Evaluation	83
5.3.1	Results on the Historical Color Images Dataset	83
5.3.2	Results on the Image Retrieval Dataset	86
5.3.3	Results on the Image Aesthetics Dataset	89
5.3.4	Results on the Adience Face Dataset	92
5.4	Summary	93
6	A Deep Ordinal Neural Network with Communication between Neurons of Same Layers	94
6.1	Overview	94
6.2	The Proposed Architecture	95
6.3	Evaluation	98
6.3.1	Results on the Historical Color Images Dataset	98
6.3.2	Results on the Image Retrieval Dataset	101
6.3.3	Results on the Image Aesthetics Dataset	103
6.3.4	Results on the Adience Face Dataset	105
6.3.5	Results Summary	107
6.4	Summary	108
7	Conclusion	109
7.1	Summary of the Main Contributions	109
7.2	Future Work	110
	Bibliography	113

List of Tables

2.1	Taxonomy of ordinal regression approaches	11
2.2	Applications of ordinal regression	28
2.3	Characteristic of datasets for ordinal regression applications	29
3.1	Coding matrix of one-against-all method for 4-class classification	41
3.2	Coding matrix of POR for 4-rank ordinal regression	42
3.3	Ordinal regression benchmarks	51
3.4	Mean zero-one error (MZE) on real ordinal regression datasets	52
3.5	Mean absolute error (MAE) on real ordinal regression datasets	53
3.6	Mean zero-one error (MZE) on discrete regression datasets	53
3.7	Mean absolute error (MAE) on discrete regression datasets	54
3.8	Win-loss summary	54
4.1	Coding matrix of CNNOR	64
4.2	Baseline methods and experimental results.	67
4.3	Accuracy performance of CNNm	69
4.4	Accuracy performance of Niu et al.'s method	69
4.5	Class distribution on MSRA-MM1.0 dataset	71
4.6	Accuracy (%) result on MSRA-MM1.0 dataset.	71
4.7	MAE result on MSRA-MM1.0 dataset.	72
5.1	Results on the historical image benchmark.	84

5.2	Class distributions on MSRA-MM1.0 dataset.	87
5.3	Accuracy (%) results on MSRA-MM1.0 dataset.	88
5.4	MAE results on MSRA-MM1.0 dataset.	88
5.5	Accuracy (%) results on the image aesthetics dataset.	90
5.6	MAE results on the image aesthetics dataset.	90
5.7	Results on the Adience face dataset.	93
6.1	Results on the historical image benchmark.	100
6.2	Accuracy (%) results on MSRA-MM1.0 dataset.	102
6.3	MAE results on MSRA-MM1.0 dataset.	103
6.4	Accuracy (%) results on the image aesthetics dataset.	104
6.5	MAE results on the image aesthetics dataset.	104
6.6	Results on the Adience face dataset.	105
6.7	Summary of accuracy results (%) on the four benchmarks.	107
6.8	Summary of MAE results on the four benchmarks.	107

List of Figures

1.1	Ordinal regression example: grading face images to four age groups. . . .	2
2.3	VGG-16 architecture.	23
3.1	The flowchart of POR framework for a training set $\{(\mathbf{x}_1, 1), (\mathbf{x}_2, 1), (\mathbf{x}_3, 2), (\mathbf{x}_4, 3), (\mathbf{x}_5, 3)\}$ and a testing point $\mathbf{x}_t$	36
4.1	Illustration of ordinal loss for a 3-rank problem.	60
4.2	The architecture of CNNOR for a 5-rank problem.	63
4.3	Historical color image dating dataset.	66
4.4	MSRA-MM1.0 dataset: cat subset.	70
5.1	The architecture of CNNPOR for a 3-rank ordinal regression problem. . .	80
5.2	Image Aesthetics dataset	91
5.3	Training curves of CNNPOR on the Adience face dataset.	93
6.1	The architecture of ORCNN for a 4-rank ordinal regression problem. . . .	96
6.2	The realization of ORCNN based on VGG-16 for a 5-rank dataset.	99
6.3	The realization of ORCNN based on LeNet for a 3-rank dataset.	101
6.4	The realization of ORCNN based on VGG-16 for a 8-rank dataset.	106

List of Publications

Yanzhu Liu, Adams Wai Kin Kong, and Chi Keong Goh. A Constrained Deep Neural Network for Ordinal Regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'18)*, June 2018, pp. 831-839.

Yanzhu Liu, Adams Wai Kin Kong, and Chi Keong Goh. Deep ordinal regression based on data relationship for small datasets. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, August 2017, pp. 2372-2378.

Yanzhu Liu, Xiaojie Li, Adams Wai Kin Kong, and Chi Keong Goh. Learning from small data: A pairwise approach for ordinal regression. In *Proceedings of 2016 IEEE Symposium Series on Computational Intelligence (SSCI'16)*, December 2016, pp. 1-6.

Summary

Ordinal regression is a supervised learning problem which aims to classify instances into ordinal categories. It is different from multi-class classification because there is an ordinal relationship between the categories. Moreover, it is different from metric regression because the target values to be predicted are discrete and the distances between different categories are not defined. Ordinal regression is an active research area because of numerous governmental, commercial and scientific applications, such as quality assessment, disease grading, credit rating, and risk stratification. The main challenge of ordinal regression is to model the ordinal information carried by the labels. Traditionally, there are two angles to tackle the ordinal regression problem: from metric regression perspective and classification perspective. However, most of existing works under both above categories are pointwise methods, in which the relationship between pairs or lists of data points is not explored sufficiently. Furthermore, learning models, especially deep neural network based models, on small datasets is challenging, but many real-world ordinal regression problems are in fact small data problems. The aim of this research is to propose ordinal regression algorithms by exploring data relationship and give consideration to suitability for small datasets and scalability for large datasets.

This thesis proposes four approaches for ordinal regression problems based on data relationship. The first approach is a pairwise ordinal regression approach for small datasets. In the training phase, the labeled instances are paired up to train a binary classifier, and the relationship between two data points in each pair is represented by a pairwise kernel. In the testing phase, a decoder algorithm is developed to recover the ordinal information from the binary outputs. By pairing up the training points, the size of the training dataset is squared, which alleviates the lack of training points in small datasets. A proof is presented that if the pairwise kernel fulfills certain properties, the time complexity solving the QP problem can be reduced from $O(n^4)$ to $O(n^2)$ without any loss of accuracy, where

n is the number of training points.

Motivated by the study of the pairwise relationship, the second approach extends the data relationship representation from pairs to triplets based on deep neural networks. The intuition is to predict rank labels by answering the question: “Is the rank of an input greater than $k - 1$ and smaller than $k + 1$?”. Therefore, the proposed approach transforms the ordinal regression problem to binary classification problems answering above question and uses triplets with instances from different categories to train deep neural networks such that high-level features describing their ordinal relationship can be extracted automatically. In the testing phase, triplets are formed by a testing instance and other instances with known ranks. A decoder is designed to estimate the rank of the testing instance based on the outputs of the network. Because of the data argumentation by permutation, deep learning can work for ordinal regression even on small datasets.

The third proposed approach is a constrained deep neural network for ordinal regression, which aims to automatically extract high-level features for representing intraclass information and interclass ordinal relationship simultaneously. A constrained optimization formulation is proposed for the ordinal regression problem which minimizes the negative loglikelihood for multiple categories constrained by the order between instances. Mathematically, it is equivalent to an unconstrained formulation with a pairwise regularizer. An implementation based on a convolutional neural network framework is proposed to solve the problem such that high-level features can be extracted automatically, and the optimal solution can be learned through the traditional back-propagation method. The proposed pairwise constraints as a regularizer make the algorithm work even on small datasets, and a proposed efficient implementation makes it be scalable for large datasets.

Furthermore, an ordinal network architecture is proposed for ordinal regression. The proposed approach embeds the ordinal relationship into the edges between nodes of the same layers in the neural network. Existing deep learning based ordinal regression approaches are implemented by traditional architectures for classification, in which no edges

exist between nodes of the same layers. The proposed architecture performs as a latent function mapping the instances to a real line, and the target categories are the intervals on this line which are decided by multiple boundaries. Most significant benefit is that the ordinal network is able to predict the rank labels directly by the outputs of the network without explicit predictions of the multiple boundaries.

This research breaks the limits of traditional ordinal regression approaches, and shows the effective and efficient performance of the proposed approaches comparing with the state-of-the-art ordinal regression approaches.

Chapter 1

Introduction

1.1 Motivation

Grading or rating is a significant capability of human intelligence, and it is used to assess quality or risk, organize items for better understanding or visualization, evaluate relative levels and so on. The machine learning technique to mimic this machinery of human being is named as ordinal regression or ordinal classification. It is a supervised learning problem, which aims to classify instances into ordinal categories. For example, as shown in Figure 1.1, the bottom line shows some example face images in a given training set, which consists of four categories of images labeled as 1 to 4 respecting to the age group “infant”, “child”, “teenager” and “adult”. Ordinal regression algorithms on this task target to learn from the training data and predict the age group for a test image, such as the face image in the first line of Figure 1.1. The problem setting is exactly same as the multi-class classification problem. However, the numerical order of category labels 1 to 4 indicates the natural order of age groups. The ordinal relationship between categories distinguishes ordinal regression problem from the multi-class classification problem. Moreover, ordinal regression is also different from the metric regression problem because the target values to be predicted are discrete and the distances between categories are not defined.

Ordinal regression technique has been widely used in numerous governmental, com-

Figure 1.1: Ordinal regression example: grading face images to four age groups.

mercial and scientific applications, such as disease grading [Kuranova and Hajdukova, 2014], sovereign credit rating [Fernandez-Navarro et al., 2013], and risk rating for beetle infestation [Robertson et al., 2009]. Many real-world ordinal regression applications lack of large training datasets. For example, in the computer-aided diagnostic problem, datasets of rare disease grading or tumor staging are usually fewer than 100 data points. These diseases affect a relatively small percentage of population, and in many cases, collecting data is difficult, expensive and invasive. Therefore, their clinical and experimental records are not large scale. Even for more common diseases, datasets of medical diagnostics are not big. For example, some researchers have explored solutions of automated grading age-related macular degeneration (AMD) [Deepak and Sivaswamy, 2012], which is one of the leading causes of central vision loss in people aged over 50 years. Two popular databases Diaretdb0 [DIARETDB,] and Diaretdb1 [Kälviäinen and Uusitalo, 2007] of AMD contain only 79 and 43 color fundus images separately. Besides of medical field, small dataset problems also arise in early prediction in engineering area. Because collect-

ing run-to-failure data is very expensive, datasets are extremely small. Small dataset is an important issue in many applications and is studying by other researchers for classification [Dougherty et al., 2015] and foresting [Chang et al., 2014]. Learning an effective ordinal classifier from a small dataset is a challenging task because lack of data increases uncertainty and easily causes overfitting.

One important reason why ordinal regression has been applied widely is that for a lot of real-world applications, it is extremely difficult to predict precise numbers for instances, but in many cases, a relative measure of goodness is sufficient. Therefore, it is observed that in most of ordinal regression applications, the number of categories is relatively small such as rating movies based on an ordinal scale 1 star to 5 stars, grading the risk level of a disease as “low”, “moderate” and “high”, ranking images in web search according to their relevance level to an input query as “very relevant”, “relevant” and “irrelevant”, categorizing images to five aesthetic grades: “Unacceptable”, “Flawed”, “Ordinary”, “Professional” and “Exceptional”, and so on. In most cases, the number of ranks is smaller than 10, because with increasing the number of ranks, it becomes harder to distinguish the difference between adjacent ranks even for human being.

Another characteristic of ordinal regression applications is that the difference between different adjacent ranks is not necessary to be equal. For example, given a query in image search, an “relevant” image is easier to be differentiated from “irrelevant” images than from “very relevant” ones. Therefore, although traditional metric regression can be viewed as a special case of ordinal regression which has equal rank intervals and the number of ranks approaches to infinity, it is not reasonable to employ the numeric order of regression values to express the ordinal relationship directly. From the other perspective, if the rank labels are viewed as discrete independent categories regardless of the order relationship, traditional multi-class classification algorithms can be arbitrarily applied in the ordinal regression task. However, classification algorithms do not consider the relationship of labels. In other words, taking the task in Figure 1.1 as an example, if the labels

are exchanged, the performance of classification will be exactly same. But as an ordinal regression problem, it will be totally different. Ordinal regression lies between regression and classification, but from both perspectives, how to model the ordinal information carried by the rank labels is challenging.

Recently, a number of machine learning approaches have been proposed for ordinal regression, and the main difference between these approaches is the way to model the ordinal relationship. For example, max-margin based approaches [Shashua and Levin, 2002] [Chu and Keerthi, 2007] predicted multiple boundaries to split the ordinal categories, and projection based approaches [Liu et al., 2011a] [Tian and Chen, 2015] mapped instances to a new space preserving the ordinal relationship between categories. However, most of existing works are pointwise methods, in which the relationship between pairs or lists of data points is not explored sufficiently. Being able to automatically extract high-level features from raw data, deep neural networks (DNNs), such as convolutional neural networks (CNNs), have attracted great attention in these several years and performed very well on many classification problems. However, instances are inputted individually into DNNs. To our best knowledge, there is no any existing work employing DNNs to extract and represent high-level features describing relationship among data instances for ordinal regression problems. Generally speaking, to train a deep neural network, a large training dataset is necessary. Learning deep neural networks on small datasets is challenging, and to design a method suitable for small datasets and also scalable for large datasets at the same time is another challenging task.

1.2 Research Objectives

The purpose of this research is to propose effective and efficient methods for ordinal regression problems based on data relationship, and it is aligned with the objective of the project RT3.1 "Virtual engine emulator by using data infusion" under cooperation with Rolls-Royce. The target of the project is to improve and speed up virtual engine design

process using machine learning methods. A relative measure of goodness is an important technique to achieve the goal, which is in fact an ordinal regression problem. Specifically, it is very useful to predict the design performance relatively in the early stage of engine design, such as estimating whether the performance of a design will be better than the threshold. For aircraft engine design, collecting data is very expensive, and for early prediction, few observations are available in the early stages of systems. Therefore, small dataset is an important issue.

The objectives of this thesis are twofold. The first objective is to tackle the ordinal regression problem by exploring data relationship, which are possibly encoded in pairs, triplets or lists of the data points. The second objective is to improve the performance of ordinal regression approaches for small datasets. Especially, this research aims to make use of the two advantages of deep neural networks to automatically extract features representing data relationship and keep scalability for large scale datasets, and adapt it suitable for small training sets scenario.

1.3 Contributions

In this research, four approaches are proposed for ordinal regression problems based on data relationship. The first approach is a pairwise ordinal regression approach for small datasets based on handcrafted features. The ordinal relationship of data points are represented in pairwise kernels in a revised support vector machine. By pairing up the training points, the size of the training set is squared, which alleviates shortage of data in small datasets. The second approach extends the data relationship representation from pairs to triplets based on deep neural networks. The proposed approach transforms the ordinal regression problem to binary classification problems answering the question “Is the rank of an input greater than $k - 1$ and smaller than $k + 1$?”. Triplets with instances from different categories are constructed to train deep neural networks so that high-level features describing their ordinal relationship can be extracted automatically. Because of the

data augmentation by permutation, deep learning can work for ordinal regression even on small datasets.

To waive the postprocessing procedure - decoding as previous two methods and implement a real end-to-end approach, data relationship is embedded into constraints of optimization objective in the third approach and it is introduced by revised weights connections of the network architecture in the last approach. A constrained optimization formulation is proposed for the ordinal regression problem in the third proposed approach, which minimizes the negative loglikelihood for multiple categories constrained by the order relationship between instances. The optimization is solved by a deep neural network which automatically extracted high-level features for representing intraclass information and interclass ordinal relationship simultaneously. Moreover, in the last proposed approach, we break the limits of traditional neural networks to connect nodes in same layers, which reflects the ordinal property that if the rank of an instance is greater than k , it will be definitely greater than $k - 1, \dots, 1$. The proposed connections formulate the ordinal relationship directly, and the architecture works as a standard multiple-class classification network which is scalable for large datasets.

The first proposed approach is evaluated on UCI feature based benchmarks, and all deep learning based approaches are tested on image datasets due to convolutional neural networks are used. The experimental results have validated the effectiveness and efficiency of the proposed approaches comparing with the state-of-the-art ordinal regression approaches.

1.4 Organization

The details of this research are shown in the following chapters. Chapter 2 reviews the literature of ordinal regression. Chapter 3 presents a pairwise ordinal regression approach based on kernel method for small datasets. Chapter 4 describes a deep ordinal regression approach based on triplets of instances, and Chapter 5 discusses a constrained deep neural

network for ordinal regression. Chapter 6 presents a deep ordinal neural network architecture with communication between neurons of same layers. Experimental evaluations with analyses and discussions are provided in chapters 3 to 6. Some conclusive remarks and discussions about future work are given in Chapter 7 as well.

Chapter 2

Background

2.1 Definition of Ordinal Regression

Ordinal regression aims to predict the rank y of an input vector $\mathbf{x}$, where $\mathbf{x}$ is in an input space $\mathcal{X} \subseteq \mathbb{R}^d$ and y is in a label space $\mathcal{Y} = \{1, 2, \dots, m\}$. The natural order of numbers in the label space indicates the order of ranks. Ordinal regression is a supervised learning problem, and a training set with labeled instances is given. The setting above is same as that of the *multi-class classification* problem. Nevertheless, the rank labels in the ordinal regression problem carry the ordinal information in addition to categorical information. Ordinal regression is also related to the *metric regression* problem, because the target values of metric regression are real numbers (i.e., $y \in \mathbb{R}$) which can be ordered. However, the discrete rank labels in ordinal regression do not carry metric information. In other words, the distances between different ranks are undefined.

In statistics and social sciences, *ordinal variables* have been widely investigated and there is a vast amount of literature describing theoretical and empirical issues about ordinal variables [Kampen and Swyngedouw, 2000] [Lalla, 2016]. However, ordinal variables usually refer to the features of instances with ordinal scale measures, in contrast to ordinal regression where ordinal scale measures are carried by the responses. The research of ordinal variables in statistics and social sciences mainly concerns ordinal variable mea-

surement, construction, and analysis. Ordinal variable measurement aims to propose an effective process to express a measured attribute by an ordered qualitative scale. Likert scale, semantic differential, Stapel scale, self-anchoring scale, feeling thermometer scale, and Juster scale are examples. Ordinal variable construction aims to generate the response of an attribute which is measured by one or more statements. Ordinal variables analysis has two purposes: one is to understand the underlying interactive process between variables, and the other one is to predict the response. Modeling ordinal variables to do prediction is similar to the ordinal regression problem if the ordinal variables are multivariate, except that the former carries ordinal scales in features and the latter carries ordinal scales in responses.

Another topic *ordinal optimization* is related but different from ordinal regression. Ordinal optimization can be viewed as soft computing for hard optimization problems [Ho, 1999]. In the search-based optimization method, certain real values indicating the system performance are needed to be estimated. Ordinal optimization estimates the discrete order level of the performance before estimating accurate values, which can quickly narrow down the search for the optimum performance to a “good enough” subset during the initial stages. This optimization strategy can be viewed as an application of ordinal regression, however, ordinal regression focuses on how to estimate the discrete order level, while ordinal optimization targets to find the precise real value by using this initial step as a speedup method.

Furthermore, it is necessary to distinguish ordinal regression from *learning to rank* in information retrieval. Learning to rank is an important technique behind search engines, and its purpose is to sort candidates (such as web documents) according to their relevance to a certain query. Learning to rank is different from ordinal regression as it is not able to predict the exact rank of an item but aims to predict the relative order between items. There are three main categories of learning to rank algorithms in literature: pointwise, pairwise and listwise [Liu et al., 2009]. Given a query, pointwise algorithms

first estimate a relevance value for each candidate and then sort the candidates based on the relevance values. The first step of this type of algorithms can be viewed as a special ordinal regression process if the relevance value is discrete and unique for each candidate.

2.2 Ordinal Regression Approaches

A comprehensive review of ordinal regression is given in [McCullagh and Nelder, 1989], [O’Connell, 2006] and [Gutierrez et al., 2016]. The reference [Gutierrez et al., 2016] is more recent, which classifies ordinal regression approaches into three categories: naïve, binary decomposition and threshold approach. The naïve approaches adapt nominal classification or metric regression methods to solve the ordinal regression problem. Binary decomposition separates the ordinal target labels to several binary ones, which are then estimated by a single or multiple binary models. Threshold approaches assume that there is a latent function mapping the instances to a real line, and the category of an instance is an interval on the line. The natural order of interval boundaries on the real line represents the ordinal relationship between categories. This taxonomy of ordinal regression approaches is based on the strategy how to deal with the ordinal information, while the ordinal regression approaches from the different categories may employ same fundamental learning technique. For example, support vector machine (SVM) was used in binary decomposition approaches as a basic binary classifier, and also used in threshold approaches as a boundary predictor. Therefore, in this thesis, recent ordinal regression approaches are summarized under a taxonomy in Table 2.1 based on the fundamental learning algorithms adapted.

For the sake of clear presentation, the problem setting and notations are given first. An ordinal regression problem with m ranks denoted by $\mathcal{Y} = \{1, 2, \dots, m\}$ is considered, where the natural order of numbers in $\mathcal{Y}$ indicates the order of ranks. Denote (X, Y) as a training set with n labeled instances: $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, and $Y = \{y_1, y_2, \dots, y_n\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is an input vector, and $y_i \in \mathcal{Y}$ is the rank label. $X_k \subseteq X$ is the subset of

input vectors whose rank labels are all k . n_k denotes its size and $\mathbf{x}_{kj}$ denotes its j -th input vector. It is assumed that each rank has at least one instance, i.e., $X_k \neq \emptyset$. The task of ordinal regression is to predict the rank label y_t for a new input vector $\mathbf{x}_t$.

Table 2.1: Taxonomy of ordinal regression approaches

	Algorithm	Publication
Max-margin based approaches	Support vector machine	1. [Herbrich et al., 1999]
		2. [Shashua and Levin, 2002]
		3. [Chu and Keerthi, 2007]
		4. [Zhao et al., 2009]
		5. [Gu et al., 2010]
		6. [Carrizosa and Martin-Barragan, 2011]
Projection based approaches	Discriminant learning	7. [Sun et al., 2010] 8. [Sun et al., 2015]
	Manifold	9. [Liu et al., 2011a]
	Nearest-centroid projection	10. [Tian and Chen, 2015]
Ensemble approaches	Reduction method	11. [Li and Lin, 2007]
	Boosting	12. [Lin and Li, 2009]
		13. [Riccardi et al., 2014]
Probabilistic approaches	Gaussian Process	14. [Chu and Ghahramani, 2005] 15. [Bao and Hanson, 2015]
	Non-parametric method	16. [DeYoreo et al., 2015]
	Semi-parametric method	17. [Donat and Marra, 2015]
	Graphical model	18. [Guo et al., 2015]
	Bayesian lasso	19. [Feng et al., 2015]
	Markov Chain Monte Carlo	20. [McKinley et al., 2015]

	Conditional random field	21. [Kim, 2014]
Neural network approaches	Artificial neural network	22. [Cheng et al., 2008] 23. [Fernandez-Navarro et al., 2014] 24. [Gutiérrez et al., 2014]
	Extreme learning machine	25. [Deng et al., 2010]
	Deep learning	26. [Niu et al., 2016]

2.2.1 Max-margin Based Approaches

Max-margin based approaches applied the principle of Structural Risk Minimization [Vapnik, 2013] to ordinal regression. As listed in Table 2.1, six representative publications are selected in this chapter to describe the basic ideas of max-margin based approaches. To our knowledge, the work in [Herbrich et al., 1999] was the first attempt to apply Structural Risk Minimization to ordinal regression. The loss function was based on pairs of instances as shown in Eq. 2.1:

$$\begin{aligned}
& \min_{\mathbf{w}, \xi} \quad \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_i \xi_i \\
& \text{s.t.} \quad \text{sign}(y_i^1 - y_i^2) \mathbf{w}^\top (\mathbf{x}_i^1 - \mathbf{x}_i^2) \geq 1 - \xi_i \\
& \quad \xi_i \geq 0
\end{aligned} \tag{2.1}$$

where $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$ denoted the first and second element of the i -th instance pair, $\text{sign}(\cdot)$ was the operator that returns 1 if the value of expression in the brackets was greater than 0, otherwise, returns -1. This optimization problem was a standard QP problem, and the optimal $\mathbf{w}$ can be solved analytically. The decision function to predict the rank of a testing point was given by Eq. 2.2:

$$h(\mathbf{x}) = k \Leftrightarrow \mathbf{w}^\top \mathbf{x} \in [b_{k-1}, b_k) \tag{2.2}$$

where b was the vector of thresholds with $b_k = \frac{\mathbf{w}^\top \mathbf{x}_s + \mathbf{w}^\top \mathbf{x}_r}{2}$, and $(\mathbf{x}_s, \mathbf{x}_r) = \text{argmin}_{\mathbf{x}_i \in X_k, \mathbf{x}_j \in X_{k+1}} (\mathbf{w}^\top \mathbf{x}_i - \mathbf{w}^\top \mathbf{x}_j)$.

The problem size of the formulation in [Herbrich et al., 1999] was the square of the size of training set. Shashua and Levin [Shashua and Levin, 2002] generalized the SVM formulation for ordinal regression by finding $m - 1$ thresholds where the problem size was two times of the size of training set. The proposed objective function was:

$$\begin{aligned}
\min_{\mathbf{w}, \xi, \xi^*} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_i \sum_k (\xi_i^k + \xi_i^{*k+1}) \\
\text{s.t.} \quad & \mathbf{w}^\top \mathbf{x}_i^k - b_k \leq -1 + \xi_i^k, \quad \xi_i^k \geq 0 \\
& \mathbf{w}^\top \mathbf{x}_i^{k+1} - b_k \geq 1 - \xi_i^{*k+1}, \quad \xi_i^{*k} \geq 0
\end{aligned} \tag{2.3}$$

where b_k was the k -th boundary dividing the intervals for rank k and $k + 1$. ξ_i^k was the error from the i -th sample of rank k , whose function value $\mathbf{w}^\top \mathbf{x}_i^k$ was greater than the margin $b_k - 1$. And ξ_i^{*k+1} was the error from i -th sample of rank $k + 1$, whose function value $\mathbf{w}^\top \mathbf{x}_i^{k+1}$ was less than the margin $b_k + 1$. The decision function was:

$$h(\mathbf{x}) = \operatorname{argmin}_{k \in \{1, \dots, m\}} \{k : \mathbf{w}^\top \mathbf{x} < b_k\} \tag{2.4}$$

The problem of the formulation in Eq. 2.3 was that it cannot be guaranteed to hold the inequalities $b_1 \leq b_2 \leq \dots \leq b_{m-1}$ in the solution. Chu and Keerthi improved the formulation in Eq. 2.3 further by adding $b_k \leq b_{k+1}$ into the constraints in SVOR [Chu and Keerthi, 2007], which was a state-of-the-art max-margin based approach for ordinal regression.

Carrizosa and Martin-Barragan [Carrizosa and Martin-Barragan, 2011] formulated all above approaches under one framework which maximizes the minimum of upgrading error (classifying an instance to a higher rank than its actual rank) and downgrading error (classifying an instance to a lower rank than its actual rank) simultaneously. The proposed framework was a bi-objective optimization problem, and main effort was taken to solve this optimization problem in the paper.

Because SVM based approaches involved solving QP problems, the computational complexity was high. Zhao et al. [Zhao et al., 2009] and Gu et al. [Gu et al., 2010]

focused on reducing the time complexity. Zhao et al. proposed a block-quantized support vector ordinal regression approach, which approximated the kernel matrix in SVOR [Chu and Keerthi, 2007] by a kernel composed by k^2 blocks. The training data set was separated into k clusters by a kernel k -means method and SVOR was performed on each of the k clusters. Therefore, the corresponding optimization problem was scaled with the number of clusters instead of the size of training set. Gu et al. extended Core Vector Machine [Tsang et al., 2005] to ordinal data, and the asymptotic time complexity was linear with the size of training set.

2.2.2 Projection Based Approaches

Projection based approaches mainly had three objectives: minimizing the intraclass distance, maximizing the interclass distance and preserving the order of ranks. Sun et al. [Sun et al., 2010] extended Kernel Discriminant Analysis to ordinal regression by introducing the ordinal information into constraints. The proposed optimization problem (KDLOR) was:

$$\begin{aligned} \min_{\mathbf{w}, \rho} \quad & \mathbf{w}^\top S \mathbf{w} - C\rho \\ \text{s.t.} \quad & \mathbf{w}^\top (\mathbf{o}_{k+1} - \mathbf{o}_k) \geq \rho \end{aligned} \quad (2.5)$$

where $S = \frac{1}{n} \sum_{k=1}^m \sum_{\mathbf{x} \in X_k} (\mathbf{x} - \mathbf{o}_k)(\mathbf{x} - \mathbf{o}_k)^\top$ was a intraclass scatter matrix, $\mathbf{o}_k$ was the mean vector of samples from the k -th rank, and C was a penalty coefficient. If $\rho > 0$, the projected mean vectors of all ranks can be sorted in order with their ranks. The rank of a testing point $\mathbf{x}$ was predicted by the decision function:

$$h(\mathbf{x}) = \operatorname{argmin}_{k \in \{1, \dots, m\}} \{k : \mathbf{w}^\top \mathbf{x} < b_k\} \quad (2.6)$$

which is the same formulation as Eq. 2.4 with $b_k = \frac{\mathbf{w}^\top (\mathbf{o}_{k+1} + \mathbf{o}_k)}{2}$.

KDLOR sought one direction where the projected samples were well ranked. Sun et al. [Sun et al., 2010] extended it further by constructing orthogonal projection vectors.

The first discriminant vector $\mathbf{w}_1$ was the solution of Eq. 2.5, and the other discriminant vectors $\mathbf{w}_{p+1}$ were obtained by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}_{p+1}, \rho_{p+1}} \quad & \mathbf{w}_{p+1}^\top S \mathbf{w}_{p+1} - C \rho_{p+1} \\ \text{s.t.} \quad & \mathbf{w}_{p+1}^\top \mathbf{w}_j = 0, \quad j = 1, 2, \dots, p \end{aligned} \quad (2.7)$$

For each $\mathbf{w}_p$, the rank of a testing point $\mathbf{x}$ was predicted by Eq. 2.6 with $\mathbf{w} = \mathbf{w}_p$. Once all predictions were obtained, majority voting or weighted averaging was used to get the final rank label for $\mathbf{x}$.

Liu et al. [Liu et al., 2011a] employed manifold to ordinal regression (MOR), and the basic idea was similar to the above approaches. The objective function was formulated as Eq. 2.8:

$$\begin{aligned} \min_{\mathbf{w}, \rho} \quad & \mathbf{w}^\top S \mathbf{w} - C \rho \\ \text{s.t.} \quad & \mathbf{w}^\top (\mathbf{o}_{k+1} - \mathbf{o}_k) \geq \rho \end{aligned} \quad (2.8)$$

where $S = \sum_{i,j=1}^m (\mathbf{x}_i - \mathbf{x}_j) A_{ij} (\mathbf{x}_i - \mathbf{x}_j)^\top$, $A_{ij} = \begin{cases} e^{-\frac{[(|y_i - y_j| + 1) \|\mathbf{x}_i - \mathbf{x}_j\|]^2}{2\sigma}} & \text{if } i \text{ and } j \text{ are neighbors} \\ 0 & \text{otherwise} \end{cases}$

and other parameters were same as those in Eq. 2.5. The difference between Eq. 2.5 and Eq. 2.8 was how to measure the intraclass distance or so called intrinsic manifold distance represented by S .

There are two limitations in both KDLOR and MOR. The first one is that the thresholds b_k associating with partial order constraints between every two neighboring ranks are optimized independently. The second limitation is that the partial order constraints on b_k are built upon the mean vectors of ranks, which are generally under-represented for distributions of data ranks. To avoid these shortages, Tian and Chen [Tian and Chen, 2015] proposed a combinatorial optimization formulation to jointly learn the thresholds across samples and rank centroids. The proposed objective function was relaxed to a QP problem and then was solved properly. It was shown in [Tian and Chen, 2015] that KDLOR and MOR were special cases of the QP problem.

2.2.3 Ensemble Approaches

The main idea of ensemble approaches for ordinal regression was to transform the classification problem into one or more nested binary classifiers, and then combine the resulting classifiers to obtain the final ensemble model. From this point of view, any binary decomposition approaches summarized in the survey paper [Gutierrez et al., 2016] can be classified under this category of approaches. Li and Lin [Li and Lin, 2007] proposed a reduction method RED-SVM to convert the ordinal regression problem to a binary classification problem based on SVM. A labeled instance $(\mathbf{x}, y)$ was extended to $m - 1$ binary instances via the following transformation:

$$\begin{aligned}\mathbf{x}^k &= (\mathbf{x}, e_k) \in \mathbb{R}^{d+m-1} \\ y^k &= 1 - 2[y \leq k]\end{aligned}\tag{2.9}$$

for $k = 1, \dots, m-1$, where $e_k \in \mathbb{R}^{m-1}$ denoted a vector with the k -th element being one, and the rests being zero. As an extended binary instance had a dimension of $d + m - 1$, the weighted vector $\mathbf{w}$ of SVM was also augmented to $(\mathbf{w}, -\theta)$, where θ is a vector of ordered thresholds: $\theta_1 \leq \theta_2 \leq \dots \leq \theta_{m-1}$. The binary prediction for $\mathbf{w}^k$ was given by:

$$f(\mathbf{x}^k) = \text{sign}((\mathbf{w}, -\theta)^\top \mathbf{x}^k) = \text{sign}(\mathbf{w}^\top \mathbf{x} - \theta_k)\tag{2.10}$$

Finally, the rank label of $\mathbf{x}$ was predicted as $\sum_{k=1}^{m-1} [f(\mathbf{x}^k) = 1] + 1$, where $[\cdot]$ was the indicator function.

Lin and Li [Lin and Li, 2009] proposed to extend the well-known AdaBoost algorithm for ordinal regression by using the reverse technique. In the training phase, AdaBoost was revised at two points: a weak ranker was used to replace the weak classifier; the target label was a rank instead of a binary class label. Thus, after T training iterations, a set of rankers $\{(r_t(\mathbf{x}), v_t) | t = 1, \dots, T\}$ were obtained. In the testing phase, by using the reverse technique, the prediction for $\mathbf{x}$ was given by:

$$h(\mathbf{x}) = 1 + \sum_{k=1}^{m-1} \left[\sum_{t=1}^T v_t [k < r_t(\mathbf{x})] \geq \frac{1}{2} \sum_{t=1}^T v_t \right]\tag{2.11}$$

Riccardi et al. [Riccardi et al., 2014] adapted the multi-class version of AdaBoost [Zhu et al., 2009] for ordinal regression, which directly handled the m -class problem by building a single m -class classifier instead of m binary ones. A new cost matrix was designed for ordinal regression in [Riccardi et al., 2014], and based on it, weights calculation in the multi-class AdaBoost were revised accordingly. To train the proposed ordinal regression AdaBoost efficiently, ELM was used as the weak classifier.

2.2.4 Probabilistic Approaches

Gaussian Process as a non-parametric Bayesian approach was successfully used in metric regression and classification problems. Chu and Ghahramani [Chu and Ghahramani, 2005] extended Gaussian Process to ordinal regression. They assumed a latent function $f(\mathbf{x}_i) \in \mathbb{R}$ associated with $\mathbf{x}_i$ was a Gaussian Process, and the rank label y_i was dependent on $f(\mathbf{x}_i)$. Firstly, the proposed method GPOR imposed a zero-mean Gaussian prior on $f(\mathbf{x}_i)$, and the covariance matrix was defined by a kernel function of input vectors. Then, GPOR defined a likelihood function for ordinal regression:

$$p(y_i|f(\mathbf{x}_i)) = \begin{cases} 1, & \text{if } b_{y_i-1} \geq f(\mathbf{x}_i) < b_{y_i} \\ 0, & \text{otherwise} \end{cases} \quad (2.12)$$

where b_{y_i} was the boundary dividing $f(\mathbf{x}_i)$ into two consecutive intervals respecting to the rank of y_i and $y_i + 1$. The vector b as a hyper parameter was approximated by MAP or EP algorithms. Finally, the estimated distribution $p(y_t|\mathbf{x}_t, D, \theta)$ for a testing point $\mathbf{x}_t$, where D was the training data and θ was the optimal hyper parameter vector, was calculated by substituting the likelihood on $\mathbf{x}_t$ and the posterior probability on the training set. The rank label y_t was predicted by $\text{argmax}_{k \in \{1, \dots, m\}} (p(y_t = k|\mathbf{x}_t, D, \theta))$. The key difference between GPOR and the traditional Gaussian Process Classification (GPC) was the definition of the likelihood function. Eq. 2.12 defined the likelihood for ordinal regression, and usually a probit function was used as a likelihood in GPC.

Two other Bayesian non-parametric algorithms for ordinal regression were proposed by Bao and Hanson [Bao and Hanson, 2015], and DeYoreo et al. [DeYoreo et al., 2015]. The latter algorithm was designed for the binary rank ordinal regression particularly. Moreover, a semi-parametric approach was applied for ordinal regression in the work [Donat and Marra, 2015] by Donat and Marra.

Graphical model is another powerful probabilistic tool to model and predict ordinal responses. Guo et al. [Guo et al., 2015] assumed that ordinal responses were generated by discretizing the marginal distributions of a latent multivariate Gaussian distribution. The ordinal information was then described by the underlying Gaussian graphical model which was inferred by estimating the corresponding concentration matrix. An approximate EM-like algorithm was developed to estimate parameters in [Guo et al., 2015]. Conditional Random Field (CRF) and Markov Chain Monte Carlo (MCMC) can be viewed as special graphical models. Kim [Kim, 2014] formulated a CRF model for ordinal regression by extending SVOR [Chu and Keerthi, 2007] and predicted ordinal labels for structured data such as sequences of images. McKinley et al. [McKinley et al., 2015] proposed a reversible-jump MCMC for ordinal regression. They relaxed the assumption of parallel borderlines in proportional odds model, and proposed a non-proportional odds model where the regression parameters were allowed to vary depending on the level of the response.

2.2.5 Neural Network Approaches

One appealing property of neural networks is that they can handle multiple responses in a seamless fashion. Usually, the number of neurons in the output layer of the network is as many as the number of ranks. The ordinal information is presented in the output vectors $\mathbf{t} = (t_1, t_2, \dots, t_m)$ of the neural network for ordinal regression. Different coding schemes for $\mathbf{t}$ have been used in literature. Cheng et al. [Cheng et al., 2008] encoded $\mathbf{t}$ by using the idea of cumulative link models: if a data point $\mathbf{x}$ belongs to the rank k ,

it is classified automatically into lower ranks $(1, 2, \dots, k-1)$ as well. Therefore, the output vector of $\mathbf{x}$ is $\mathbf{t} = (1, 1, \dots, 1, 0, \dots, 0)$, where t_i is set to 1 for $1 \leq i \leq k$ and 0 for others. In [Cheng et al., 2008], a standard neural network for classification with d input neurons, m output neurons and one hidden layer was applied, where d respected to the dimensions of input vectors and m was the number of ranks. Input neurons were fully-connected with the neurons in the hidden layer, which in turn were fully-connected with output neurons, and output neurons were activated by a sigmoid function without normalization. In the training phase, the encoded vectors $\mathbf{t}$ for training instances were used as labels. In the testing phase, the output vector $\mathbf{t}$ of a testing point was scanned from t_1 to t_m , and the predicted rank was the first k with t_k smaller than a predefined threshold (e.g., 0.5).

Gutiérrez et al. [Gutiérrez et al., 2014] encoded $\mathbf{t} = (t_1, t_2, \dots, t_m)$ using probabilities, where $t_i = p(y = i|\mathbf{x})$ was the conditional probability that $\mathbf{x}$ belonged to the rank i . Thus, the output vector of an instance $\mathbf{x}$ from the rank k was $\mathbf{t} = (0, \dots, 0, 1, 0, \dots, 0)$, where t_i was set to 1 for $i = k$ and 0 for others. Gutiérrez et al. [Gutiérrez et al., 2014] followed the idea of Proportional odds model (POM)[McCullagh, 1980] to estimate the probabilities by using Eq. 2.13:

$$\begin{aligned} p(y \leq k|\mathbf{x}) &= p(y = 1|\mathbf{x}) + \dots + p(y = k|\mathbf{x}) \\ p(y = k|\mathbf{x}) &= p(y \leq k|\mathbf{x}) - p(y \leq k-1|\mathbf{x}) \end{aligned} \quad (2.13)$$

where k was a rank index. They assumed the latent function $f(\mathbf{x})$ mapping instance to a real line followed a logistic cdf, such that $p(y \leq k|\mathbf{x}) = \frac{1}{1+e^{f(\mathbf{x})-b_k}}$ and $p(y = k|\mathbf{x}) = \frac{1}{1+e^{f(\mathbf{x})-b_k}} - \frac{1}{1+e^{f(\mathbf{x})-b_{k-1}}}$, where b_k was the boundary separating the intervals for the rank $k-1$ and k . A neural network with only one output neuron was used to estimate $f(\mathbf{x})$, such that $f(\mathbf{x}) = \sum_{j=1}^H \beta_j B_j(\mathbf{x}, \mathbf{w}_j)$, where H was the number of hidden neurons, $\mathbf{w}_j$ was the input weight vector for j -th hidden node, $B_j(\mathbf{x}, \mathbf{w}_j)$ was a basis function, and β_j was the j -th output weight. The constraint $b_k = b_{k-1} + \Delta_k^2$ was introduced to guarantee the order of the boundaries. In the training phase, the cross-entropy loss was used to per-

form maximum likelihood estimation of the parameter vector $\theta = (\beta, \mathbf{w}, b_1, \Delta)$, where $\beta = [\beta_1 \cdots \beta_H]$, $\mathbf{w} = [\mathbf{w}_1 \cdots \mathbf{w}_H]$, $\Delta = [\Delta_1 \cdots \Delta_m]$. In the testing phase, the prediction for a test point $\mathbf{x}_t$ was the rank k with the maximum conditional probability, i.e., $\operatorname{argmax}_{k \in \{1, \dots, m\}} p(y = k | \mathbf{x}_t, \theta)$. Moreover, a concentric hypersphere neural network was proposed to extend this idea to multiple projected real lines by approximating multiple $f(\mathbf{x})$ separately and using norm of them to obtain the final prediction.

Deng et al. [Deng et al., 2010] and Fernandez-Navarro et al. [Fernandez-Navarro et al., 2014] focused on training neural networks efficiently for ordinal regression by using extreme learning machine (ELM) [Huang et al., 2006], which was a fast learning algorithm for single hidden layer network and input weights can be set randomly and the output weights can be decided analytically. Deng et al. [Deng et al., 2010] defined a coding matrix for ordinal regression under Error Correcting Output Codes (ECOC) [Allwein et al., 2000] framework, and based on it, they encoded the output vectors $\mathbf{t} = (t_1, \dots, t_m)$ for training instances. Then a traditional neural network was employed. The prediction for a testing point $\mathbf{x}_t$ was the row index of the coding matrix which was nearest to the output vector $\mathbf{t} = (t_1, \dots, t_m)$ of $\mathbf{x}_t$.

The neural network for ordinal regression proposed in [Fernandez-Navarro et al., 2014] used the same coding scheme for $\mathbf{t}$ as Gutiérrez et al. [Gutiérrez et al., 2014]. The network was a standard single hidden layer neural network, and each output neuron was for one rank. Thus, $f_k(\mathbf{x}) = \sum_{j=1}^H \beta_{jk} B_j(\mathbf{x}, \mathbf{w}_j)$ was the projection for k -th output neuron, where β_{jk} was the weight on edge from j -th hidden neuron to k -th output neuron, $B_j(\mathbf{x}, \mathbf{w}_j)$ was a basis function. To introduce the ordinal information, the constraints $\beta_{jk} \leq \beta_{j(k+1)}$ were added to ensure the monotonicity. Finally, ELM was employed to learn the proposed network efficiently.

2.2.6 Deep Learning Approaches

In these several years, deep learning has attracted great attention and performed very well on many classification problems. Deep neural networks (DNNs) are extensions of artificial neural network (ANN) with deeper architectures. Thus, in Table 2.1, DNNs based ordinal regression methods are listed under the category “Neural network approaches”.

In DNNs, each layer learns to transform its input data into a slightly more abstract and composite representation. For example, in an image recognition application, the raw input may be a matrix of pixels, and the following layers recognize more abstract representation in images such as edges or directions, nose or eyes, and finally maybe faces. Importantly, a deep learning process can learn which features to optimally place in which level on its own. The “deep” in “deep learning” refers to the number of layers through which the data is transformed.

In the last decade, lots of research about deep learning have been done for the traditional multi-class classification task. Three well-known deep neural network architectures LeNet [LeCun et al., 1998], AlexNet [Krizhevsky et al., 2012] and VGG-16 [Simonyan and Zisserman, 2014] are shown in Figure 2.1, 2.2, and 2.3. LeNet architecture was proposed for handwritten digit recognition task and the inputs are small images with 32×32 pixels images. It is designed to apply two convolutional layers (i.e., layer 1 and layer 2 in Figure 2.1) and three fully-connected layers (i.e., layer 3-5 in the figure) to require minimal preprocessing. The convolutional layers apply a convolution operation on 5×5 patches of an image to capture the abstract spatial neighborhood features, and the numbers of feature maps in the convolutional layers are 6 and 16, which are denoted in the rectangular of the figure. There are two max-pooling layers between layer 1 and 2 and layer 2 and 3, which use the maximum value from each of a cluster of neurons at the prior layer as combination into a single neuron in the next layer. The numbers 120, 84 and n in the fully-connected layers in Figure 2.1 indicate the number of neurons. In the last layer, the number of neurons n is designed to be equal to the number of target categories of the

Figure 2.1: LeNet architecture.

Figure 2.2: AlexNet architecture.

Figure 2.3: VGG-16 architecture.

multi-class classification task. On 2012, Krizhevsky et al. proposed AlexNet , which has five convolutional layers and three fully-connected layers, as shown in Figure 2.2. And it introduced rectified linear unit (ReLU) as activation function following each convolutional layer and fully-connected layer. Furthermore, Simonyan and Zisserman proposed a much deeper architecture VGG-16 as shown in Figure 2.3.

However, very few works use DNNs for ordinal regression problem. Niu et al. [Niu et al., 2016] adapted CNN for age estimation and they claimed that their method was the first work to use DNNs for ordinal regression. They transformed the m -rank ordinal regression problem to $m - 1$ binary classifiers and the k -th classifier answered the question “Is the rank y_t of an instance greater than k ?”. The idea was very similar to RED-SVM [Li and Lin, 2007], but it adapted a single CNN to combine all classifiers and output the $k - 1$ predictions at the same time. A postprocessing step was required to decode the final predicted rank for a testing instance $\mathbf{x}_t$ from possible contradictory outputs. For example, the outputs of the CNN predicted that y_t was greater than $k + 1$ and smaller than $k - 1$. In [Niu et al., 2016], Niu et al. followed the decoding strategy of RED-SVM to assign $y_t = \sum_{k=1}^{m-1} [f_k(\mathbf{x}_t) = 1] + 1$, where $f_k(\mathbf{x}_t)$ was the k -th output of the CNN for $\mathbf{x}_t$.

2.3 Issues in Ordinal Regression and Solutions

2.3.1 Incremental Ordinal Regression

In some real-world ordinal regression tasks, training data are provided by one instance at one time. This is an incremental or online scenario. PRank [Crammer et al., 2001], a perception approach that generalized the binary perception algorithm to the ordinal regression situation, was a fast online algorithm. Like other threshold approaches, PRank aimed to find a weight vector $\mathbf{w}$ to project instances into subintervals defined by a threshold vector $\mathbf{b}$. At the i -th iteration, PRank took one instance $\mathbf{x}_i$ as an input and predicted its rank by selecting the smallest rank k such that $\mathbf{w}_i^T \mathbf{x}_i < b_k$, and $\mathbf{w}_i$ and b_k were up-

dated according to the mistake between the ground truth and the predicted rank of $\mathbf{x}_i$. There are two benefits of PRank: the inequalities $b_1 \leq b_2 \leq \dots \leq b_{m-1}$ is preserved at each iteration; the maximum number of updates for $\mathbf{w}_i$ and b_k at each iteration can be guaranteed.

Besides of the perceptron algorithms, Gu et al. [Gu et al., 2015] proposed an incremental learning algorithm based on SVM. They first modified the sum-of-margin SVM ordinal regression formulation in Eq. 2.3 [Shashua and Levin, 2002] such that each constraint included a mixture of one equality and one inequality. Then an online support vector classification algorithm [Gu and Sheng, 2013] was adapted to solve the modified quadratic problem. It should be pointed out that the proposed problem size was linear in the training data size.

2.3.2 Semi-supervised Ordinal Regression

In many real-world problems, labeled ordinal data are difficult to obtain, while a large amount of unlabeled data are available. Semi-supervised ordinal regression aims to make use of the unlabeled data for training to alleviate the challenge introduced by the lack of labeled data. Liu et al. [Liu et al., 2011b] extended their work MOR [Liu et al., 2011a] to the semi-supervised setting. The objective function to be minimized was:

$$\frac{1}{2} \sum_{i,j=1}^n \left\| \frac{F_i}{\sqrt{D_{ii}}} - \frac{F_j}{\sqrt{D_{jj}}} \right\|^2 A_{ij} + \mu \sum_{i=1}^n \|F_i - Y_i\|^2 \quad (2.14)$$

where Y denoted the $n \times m$ label matrix with $Y_{ij} = 1$ if $y_i = j$ and $Y_{ij} = 0$ otherwise. F denoted the final label matrix and $F_i(F_j)$ was its i -th (j -th) row. $D_{ii} = \sum_{j=1}^n A_{ij}$ where matrix A was same as that in Eq. 2.7. The first term of Eq. 2.14 ensured that nearby data points have similar labels, while the second term required the consistency between final labels and initial labels.

Srijith et al. [Srijith et al., 2013] proposed a semi-supervised ordinal regression approach based on Gaussian Process. This work was a combination of the transductive Gaussian Process regression (TGPR) [Le et al., 2006] and Gaussian Process for ordinal

regression (GPOR) [Chu and Ghahramani, 2005]. The assumption was that the estimating distribution on unlabeled data should have properties similar to the output distribution on labeled data. Original TGPR minimized the distance between distribution on labeled data and unlabeled data by using Eq. 2.15:

$$\operatorname{argmin}_q -\log p(D_l|\theta) + \lambda KL(q(Y_u) \parallel p(Y_u|X_u, D_l, \theta)) \quad (2.15)$$

where $KL(\cdot)$ was KL-divergence, D_l was the labeled data and X_u was the set of feature vectors of unlabeled data with Y_u as its unknown label set. And $p(Y_u|X_u, D_l, \theta) = \prod_{\mathbf{x}_u \in X_u, y_u \in Y_u} p(y_u|\mathbf{x}_u, D_l, \theta)$. To introduce the ordinal information into TGPR, GPOR algorithm was directly applied to calculate $p(y_u|\mathbf{x}_u, D_l, \theta)$. As a result, the optimization of Eq. 2.15 became a mixed integer programming problem, and Srijith et al. [Srijith et al., 2013] proposed an approximate method to solve it efficiently.

2.3.3 Others

As a supervised learning problem, ordinal regression suffers common issues existed in standard classification or regression problems. *Data imbalanced* refers to the problem that the numbers of training points in some ranks are large while those in other ranks are very small. Kim et al. [Kim, 2014] proposed a method to solve the data imbalanced problem for ordinal regression based on KNN. *Feature selection* is also an important topic in ordinal regression. Baccianella et al. [Baccianella et al., 2014] focused on feature selection for ordinal text instances, and Feng et al. [Feng et al., 2014] proposed a supervised feature subset selection method with ordinal optimization. Moreover, Perez-Ortiz et al. [Perez-Ortiz et al., 2015] proposed an approach for instance selection in the context of ordinal regression. *Model selection* between multiple ordinal classifiers has been considered in ensemble approaches, and the measures of models for ordinal regression were summarized by Nelson and Edwards [Nelson and Edwards, 2015]. *Robust* ordinal regression [Corrente et al., 2013] was investigated for multiple criteria decision aiding. Robust ordinal regression mainly focuses on the challenge of uncertainty in the ordinal

scenario. Recently, large scale problem has attracted a lot of attentions in the machine learning field. For ordinal regression, on one hand, several *sparse* algorithms were presented, such as Chang et al. [Chang et al., 2009] and Sriji et al. [Sriji et al., 2012]; on the other hand, some efficient *parallel* or *distributed* implementations [Wang et al., 2014][Wang et al., 2015] were designed by using some popular software frameworks, such as MapReduce.

2.4 Applications of Ordinal Regression

Numerous governmental, commercial and scientific applications involve grading or rating, such as quality assessment, disease grading, credit rating, and risk stratification. Table 2.2 summarizes recent main applications of ordinal regression and lists one representative publication for each application, which are published from year 2010 to 2016. Moreover, the fundamental algorithms employed in these approaches are shown in the column “Algorithm”. Table 2.2 indicates that the application areas of ordinal regression are very wide, including: medicine (application 4, 10 and 12), education (application 11), environment (application 9, 14 and 16), politics (application 13), finance (application 19), economics (application 8, 15 and 20) and human life (application 18). Most of the applications in Table 2.2 focus on collecting data and extracting field related features. From the algorithm perspective, most of them directly apply the existing algorithms or compare multiple ordinal regression algorithms on the specific datasets.

Table 2.3 describes the characteristics of datasets used to evaluate the approaches in Table 2.2. The id number in the first column respects to the application id in Table 2.2, and the second column lists the datasets used in the corresponding publications. For each dataset, its reference is cited following the dataset name, except of those cannot be accessed publicly. The column “R” in Table 2.3 indicates the number of ranks in the applications, and the last column “N” shows the size of the dataset. Column “N” also describes the data types in different applications: applications 1-5 focus on images,

applications 6-9 work on time series, application 10 works on text and the rests are on field specific feature vectors (denoted as “patterns” in the column). Based on Table 2.3, it can be concluded that usually the numbers of ranks in these ordinal regression applications are not very large, most of which are from 3 to 6.

Table 2.2: Applications of ordinal regression

Application	Publication	Algorithm
1. Image retrieval/ranking	[Liu et al., 2011b]	Manifold
2. Age estimation	[Tian et al., 2016]	Max-margin based
3. Facial beauty assessment	[Yan, 2014]	Projection based
4. Medical prognostic	[Doyle et al., 2013]	Gaussian process
5. Facial expression analysis	[Rudovic et al., 2012]	CRF
6. Action analysis	[Chen et al., 2014]	CRF
7. Speech quality prediction	[El Asri et al., 2014]	SVM
8. Rating rich countries	[Sianes et al., 2014]	Neural network
9. Weather prediction	[Ogawa et al., 2014]	Discriminant learning
10. Emergency and disaster information	[Kim et al., 2016]	KNN
11. Medical risk stratification	[Tran et al., 2015]	Sparse method
12. Educational data mining	[Gómez-Rey et al., 2015]	Gravitational model
13. Policy public perception	[Martin et al., 2014a]	Cumulative link model
14. Deforestation processes	[Minetos and Polyzos, 2010]	Cumulative link model
15. Measure of globalization	[Dorado-Moreno et al., 2015]	Neural network
16. Drought characterization	[Hao et al., 2016]	Cumulative link model
17. Fisheries	[Perez-Ortiz et al., 2015]	Methods comparison
18. Rating quality of life	[Crane et al., 2016]	Logistic regression

19. Remitting behaviour analysis	[Campoy-Muñoz et al., 2013]	Methods comparison
20. Sustainable development process estimation	[Pérez-Ortiz et al., 2014]	Ensemble learning

Table 2.3: Characteristic of datasets for ordinal regression applications

ID	Datasets	R	N
1	UMIST face dataset [Graham and Allinson, 1998]	2	575 images
	FG-NET aging dataset [FGNet, 2010]	6	1,002 images
	MSRA-MM dataset [Wang et al., 2009]	10	10,000 images
	Mixed gambles task dataset [Tom et al., 2007]	16	N.A.
2	FG-NET aging dataset [FGNet, 2010]	4	1,002 images
	Morph Album I [Guo et al., 2009]	5	1,690 images
	Morph Album II [Guo et al., 2009]	5	55,000 images
	UMIST	6	564 images
3	MBW and FBW	5	10,400 images
4	Brain images	3	450 images
5	UNBC-MacMaster shoulder pain expression archive dataset [Lucey et al., 2011]	6	200 image sequences
	Denver intensity of spontaneous facial actions dataset [Mavadati et al., 2013]	27	videos
6	Stanford 40 actions [Yao et al., 2011]	40	9,532 images
	UCF sports [Rodriguez et al., 2008]	10	150 videos
	Hollywood1 human action datasets [Laptev et al., 2008]	8	430 videos
7	LEGO corpus [Schmitt et al., 2012]	5	200 dialogues

8	Commitment to development index [Easterly and Pfutze, 2008]	5	9 time series
9	Japan weather association weather prediction	7	4 time series
10	Radio distress-signalling and infocommunications [ROSE, 2014]	5	2,034 text documents
11	Mental health dataset	3	17, 566 patterns
12	Turkiye student evaluation dataset [Fokoue and Gündüz, 2013]	5	5,820 patterns
13	Feebate policy data	2	2,473 patterns
14	Case study	4	N.A
15	KOF index of globalisation [Dreher et al., 2008]	6	1,470 patterns
16	U.S. Drought Monitor - drought categories [USDM, 2010]	6	168 patterns
17	Simulation data	3	1,022 patterns
18	Sydney travel and health study [Rissel et al., 2013]	5	846 patterns
19	National immigrant survey	3	1,303 patterns
20	European council EU SDS [Eurostat, 2010]	4	162 patterns

2.5 Evaluation Metrics

Most of evaluation metrics for nominal classification can be used for ordinal regression, Baccianella [Baccianella et al., 2009] and George et al. [George et al., 2016] summarized common metrics as follows:

Mean Zero-One Error (MZE) is defined as the fraction of incorrect predictions:

$$\frac{1}{|T|} \sum_{\mathbf{x}_t \in T} [\hat{y}_t \neq y_t] \quad (2.16)$$

where T is a test dataset whose size is $|T|$, y_t is the ground truth for the test vector $\mathbf{x}_t$ and $\hat{y}_t$ is its predicted rank label. $[\cdot]$ is the indicator function that returns 1 if the expression inside is true, otherwise, returns 0.

Mean Squared Error (MSE) is defined as:

$$\frac{1}{|T|} \sum_{\mathbf{x}_t \in T} (\hat{y}_t - y_t)^2 \quad (2.17)$$

A variant of MSE is *Root Mean Square Error (RMSE)* corresponds to the square root of MSE.

Mean Absolute Error (MAE) is the average absolute deviation of the prediction from the ground truth:

$$\frac{1}{|T|} \sum_{\mathbf{x}_t \in T} |\hat{y}_t - y_t| \quad (2.18)$$

where $|\cdot|$ is the absolute value operator.

Average Mean Absolute Error (AMAE) evaluates the mean of MAE across classes as a more robust measure for imbalanced datasets:

$$\frac{1}{m} \sum_{i=1}^m \frac{1}{|T_i|} \sum_{\mathbf{x}_t \in T_i} |\hat{y}_t - y_t| \quad (2.19)$$

where m is the number of ranks in the testing set, and T_i is the subset with all instances from rank i .

Kendall's correlation coefficient $\tau_b \in [-1, 1]$ measures the rank correlation:

$$\frac{C - D}{\sqrt{(C + D - T_t)(C + D - T_p)}} \quad (2.20)$$

where C is the number of concordant pairs, D is the number of discordant pairs, T_t is the number of tied pairs in the true class membership, and T_p is the number of tied pairs in the predicted class membership.

Some other measurements relate to the receiver operating characteristic (ROC) curves. For example, Waegeman et al. [Waegeman et al., 2006] used the volume under the modified ROC surface to evaluate the ordinal regression performance.

Chapter 3

Learning from Small Scale Data: A Pairwise Approach for Ordinal Regression

3.1 Overview

As a supervised learning problem, a large number of labeled data is needed to train an accurate model. Learning an effective ordinal classifier from a small dataset is a challenging task. However, as discussed in Chapter 1, numerous of real-world ordinal regression applications are in fact small data problems. For general supervised learning problems, learning from a small dataset is challenging because lack of data increases uncertainty and easily causes overfitting. However, for ordinal regression problems, the ordinal relationship between instances in different categories is valuable information, which can be used to alleviate problems of small training sets. To deal with lack of training data, semi-supervised learning and transfer learning have been applied to ordinal regression [Srijith et al., 2013] [Seah et al., 2012]. However, small dataset problems are different from these two settings. Semi-supervised learning aims to make use of unlabeled data for training, typically given a small amount of labeled data with a large amount of unlabeled data, but

in small dataset problems, both labeled and unlabeled data are few. Taking rare diseases as an example, the labeled data are instances with grading labels, such as low, moderate and high, and the unlabeled data are from patients with the disease but without the severity level labels. These unlabeled data are also difficult to obtain. Transfer learning aims to make use of other data from related domains for training. However, it is difficult to measure whether a dataset is related or not and hard to guarantee no negative transfer.

This work proposes a framework to transform the ordinal regression problem to a binary classification problem and then recover the ordinal information from the binary outputs. The labeled instances are paired up to train a binary classifier, and therefore, the number of training points is squared, which alleviates the lack of training points. The transformed binary classification problem is solved by a pairwise SVM method. The ordinal regression approaches in literature are summarized in a taxonomy containing naïve, binary decomposition and threshold approaches, which has been briefly discussed in Section 2.2. The proposed approach is in the binary decomposition category. RED-SVM [Lin and Li, 2012] is a state-of-the-art binary decomposition approach, which extended each instance to m instances for m ranks and based on the derived dataset, a binary classifier was trained to predict the rank of an instance is greater or smaller than each rank. The proposed method in this chapter converts the original regression problem to a single binary classification problem which answers the question for every two instances: “Whose rank is greater?” and it uses a tailor-made decoder to recover the rank labels from the outputs of the classifier. Because of the high generalization performance of SVM, several SVM-based formulations have been proposed for ordinal regression. SVOR [Chu and Keerthi, 2007] is a state-of-the-art SVM-based ordinal regression method, which generalized SVM with multiple thresholds. The authors proposed two formulations: fixed-margin-based and the other one is sum-of-margins-based. The details have been discussed in Section 2.2. The binary classifier in the proposed algorithm is also SVM-based, but other binary classifiers with good performance can be applied.

The proposed approach makes use of relationship between two instances in every possible pairs to extend knowledge with small datasets. The framework can transform an ordinal regression problem on a dataset with n instances to a binary classification problem on a dataset with $n^2 - \sum_{k=1}^m n_k^2$ instances (m is the number of ranks). A decoder is developed to predict the category of an instance from the binary outputs of the classifier. Although we employ a revised SVM method for classification and do further analysis based on that, other binary classification methods can be considered. The experimental results have shown that in 12 small scale benchmarks (with less than 300 data points), our method has gained comparable performance.

The contributions of this work include:

1. Increase the number of training points from n to n^2 by transforming the ordinal regression problem to a binary classification problem.
2. Modify support vector machine classifier by employing pair-wise kernels and introducing distances between different ranks into the constraints.
3. Reduce time complexity of solving the proposed QP problem from $O(n^4)$ to $O(n^2)$ (n is the number of training points) without any accuracy loss, provided that the pair-wise kernel in SVM fulfills certain properties.
4. Develop a decoder to predict the rank of a test point from the binary classifier's outputs.

3.2 A Pairwise Approach for Ordinal Regression

For the sake of clear presentation, the problem setting and notations given in Section 2.2 is presented here again. An ordinal regression problem with m ranks denoted by $\mathcal{Y} = \{1, 2, \dots, m\}$ is considered, where the natural order of numbers indicates the order of ranks. Denote (X, Y) as a training set with n labeled instances: $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$,

and $Y = \{y_1, y_2, \dots, y_n\}$, where $\mathbf{x}_i \in R^d$ is an input vector, and $y_i \in Y$ is the rank label. $X_k \subseteq X$ is the subset of input vectors whose rank labels are all k , and denote n_k as the size of X_k and $\mathbf{x}_{kj}$ as the j -th input vector of it. It is assumed that each rank has at least one instance, i.e. $X_k \neq \emptyset$. $(\mathbf{x}_i, \mathbf{x}_j)$ refers to a pair of input vectors, and $I = \{(i, j) | \mathbf{x}_i, \mathbf{x}_j \in X\}$ is defined as the index set of input pairs in the training set. The task of ordinal regression is to predict the rank label y_t for an new input vector $\mathbf{x}_t$, and the prediction is denoted by $\hat{y}_t$.

3.2.1 A Pairwise Framework

The proposed pairwise ordinal regression (POR) framework is based on a simple intuition. Because there are some training instances in each rank, the prediction of a test point can be obtained by comparing it to each of training instances. More precisely, given a test point $\mathbf{x}_t$, if we can answer “Is $y_t > y_i$ true?” for any $\mathbf{x}_i \in X$, the rank label y_t will be easily inferred from these answers. Figure 3.1 shows the POR framework, which is a realization of this intuition. The framework contains four steps: the first three steps solve the binary classification problem to answer above question, and the final step is to determine the rank label of a test point based on the answers.

POR starts from deriving new datasets from the original dataset in step 1. Any two instances from different ranks in original training set are paired up to form one new instance, and the new instance is labeled as positive or negative according to the original ranks of the two entries. For example, two instances $\mathbf{x}_i$ and $\mathbf{x}_j$ come from different ranks y_i and y_j . If $y_i < y_j$, the new instance $(\mathbf{x}_i, \mathbf{x}_j)$ will be labeled as $+1$; otherwise, it will be labeled as -1 . The number of instances in the derived dataset is $n^2 - \sum_{k=1}^m n_k^2$, which is $O(n^2)$. The dataset of the new binary problem has been extended quadratically to relieve the difficulties brought by small datasets. The second step of POR is to train a pairwise SVM on the derived dataset. The input of this SVM is a pair $(\mathbf{x}_i, \mathbf{x}_j)$, and its output is a binary label $+1$ or -1 . The SVM kernels are pairwise: $K(X \times X) \times (X \times X) \rightarrow R$.

Figure 3.1: The flowchart of POR framework for a training set $\{(\mathbf{x}_1, 1), (\mathbf{x}_2, 1), (\mathbf{x}_3, 2), (\mathbf{x}_4, 3), (\mathbf{x}_5, 3)\}$ and a testing point $\mathbf{x}_t$.

The distance between rank y_i and y_j is introduced into the constraints of SVM, and the details will be discussed in section 3.2.2. In the third step, given a test instance $\mathbf{x}_t$, it is paired up with all the training points to form a set of pairs $\{(\mathbf{x}_t, \mathbf{x}_i) | \mathbf{x}_i \in X\}$. Then the predicted binary label l_{ti} for all $(\mathbf{x}_t, \mathbf{x}_i)$ will be computed and these labels indicate the rank of $\mathbf{x}_t$ should be larger or smaller than the rank of $\mathbf{x}_i$. Because the ranks of all the training points are known, the fourth step of POR is to decode the estimated rank $\hat{y}_t$ from $\{((\mathbf{x}_t, \mathbf{x}_i), l_{ti}) | \mathbf{x}_i \in X, l_{ti} \in \{+1, -1\}\}$ for $\mathbf{x}_t$.

3.2.2 A Pairwise SVM

Brunner et al. [Brunner et al., 2012] proposed pairwise support vector machines (SVM) to track pairwise classification problem. Task of pairwise classification is to predict whether the instances (a, b) of a pair (a, b) belong to the same class or to different classes. We adopt this pairwise SVM to address rank comparison problem, which is to predict whether the rank of instance a of a pair (a, b) is smaller than the rank of instance b . The following formulation is defined for the rank comparison problem on the derived dataset.

$$\begin{aligned}
\min_{\mathbf{w}, \varepsilon} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{(i,j) \in I} \varepsilon_{i,j} \\
\text{s.t.} \quad & \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{x}_j) \geq d_{i,j} - \varepsilon_{i,j}, & \text{if } y_i > y_j \\
& \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{x}_j) \geq -d_{i,j} + \varepsilon_{i,j}, & \text{if } y_i < y_j \\
& \varepsilon_{i,j} \geq 0
\end{aligned} \tag{3.1}$$

where I is the index set, $\varepsilon_{i,j}$ is the slack variable with respect to pair $(\mathbf{x}_i, \mathbf{x}_j)$, $\phi(\cdot, \cdot)$ is a function mapping pair $(\mathbf{x}_i, \mathbf{x}_j)$ to a high dimension, and $d_{i,j} = |y_i - y_j|$. In the first two constraints, $d_{i,j} = |y_i - y_j|$ is used to quantify the distance between rank y_i and y_j . By introducing the distance into the constraints, the contributions of different pairs are distinguished. For example, assuming that $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are from rank 1, 2, 3 respectively, both $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_1, \mathbf{x}_3)$ are positive instances for the new binary classification problem, but the contributions of these two instances should not be same because their rank distances

are different.

The index set I is symmetric. In other words, if $(i, j) \in I$, then $(j, i) \in I$, and $\forall (i, j) \in I, i \neq j$. Then, Eq. 3.1 can be rewritten as following:

$$\begin{aligned} \min_{\mathbf{w}, \varepsilon} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{(i,j) \in I} \varepsilon_{i,j} \\ \text{s.t.} \quad & \text{sign}(y_i - y_j) \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{x}_j) \geq d_{i,j} - \varepsilon_{i,j} \\ & \varepsilon_{i,j} \geq 0 \end{aligned} \quad (3.2)$$

Eq. 3.2 is the pairwise SVM formulation proposed for rank comparison. The input of this SVM is a pair $(\mathbf{x}_i, \mathbf{x}_j)$, and the corresponding kernels are defined on pairs: $K : (X \times X) \times (X \times X) \rightarrow R, K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = \langle \phi(\mathbf{x}_i, \mathbf{x}_j), \phi(\mathbf{x}_k, \mathbf{x}_l) \rangle$. Pairwise kernels for ordinal regression must fulfill the following properties, (a) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_k, \mathbf{x}_l), (\mathbf{x}_i, \mathbf{x}_j))$, because according to Mercer's theorem, kernel matrix must be symmetric and (b) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_j, \mathbf{x}_i), (\mathbf{x}_l, \mathbf{x}_k))$. Brunner et al. [Brunner et al., 2012] gave several examples of pairwise kernels, such as metric learning pairwise kernel, direct sum learning pairwise kernel, and tensor metric learning pairwise kernel. Any traditional kernel $K : X \times X \rightarrow R$ can in fact be used if a new feature vector representing pair $(\mathbf{x}_i, \mathbf{x}_j)$ is constructed in advance. In implementation, a lot of methods can be used to construct a feature vector $\mathbf{x}'$ from the pair $(\mathbf{x}_i, \mathbf{x}_j)$, such as $\mathbf{x}' = \mathbf{x}_i - \mathbf{x}_j$ and $\mathbf{x}' = [\mathbf{x}_i; \mathbf{x}_j]$ which appends $\mathbf{x}_j$ to $\mathbf{x}_i$.

After training a pairwise SVM, the decision function defined in Eq. 3.3 is used to predict the rank of a test point.

$$f(\mathbf{x}_k, \mathbf{x}_l) \triangleq \sum_{(i,j) \in I} \text{sign}(y_i - y_j) \alpha_{i,j} K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) \quad (3.3)$$

where $\alpha_{i,j}$ is the Lagrange multiplier of $(\mathbf{x}_i, \mathbf{x}_j)$. For the ordinal regression problem, the decision function must fulfill the property $f(\mathbf{x}_k, \mathbf{x}_l) = -f(\mathbf{x}_l, \mathbf{x}_k)$, which is called negative symmetric. Obviously, if the kernel used in SVM fulfills the property (c) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = -K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_l, \mathbf{x}_k))$, the decision function will be always negative symmetric. Moreover, if $\forall (k, r), (r, l) \in I$, the kernel satisfies the property (d) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$

$= K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_r)) + K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_r, \mathbf{x}_l))$, then $f(\mathbf{x}_k, \mathbf{x}_l) = f(\mathbf{x}_k, \mathbf{x}_r) + f(\mathbf{x}_r, \mathbf{x}_l)$. In section 3.3, we will prove that if the kernel used in pairwise SVM satisfies this property (d), the training pairs can be reduced from $O(n^2)$ to $O(\frac{n^2}{m})$ without any accuracy loss, where n is the number of training points and m is the number of ranks. Furthermore, if the kernel is combined by standard pointwise kernels in a certain form, the training complexity can be reduced from $O(n^4)$ to $O(n^2)$, which is near to that of SVM with n training points.

3.2.3 A Decoder

In the POR framework, once the binary prediction values are obtained, a rule is applied to determine the rank labels. In the prediction phase, a testing point $\mathbf{x}_t$ is paired up with all training points, and the pairs are inputted to the pairwise SVM. The decoder is designed to estimate the rank of $\mathbf{x}_t$ from the SVM outputs. Algorithm 1 presents the proposed decoding algorithm, where the rank of $\mathbf{x}_t$ is determined by majority voting. Assuming that the rank of $\mathbf{x}_t$ is equal to k , the decoder calculates how many SVM outputs from the input pairs fit the assumption. The rank label is assigned to the k who fits the SVM outputs best.

In the literature, the prediction phase of the decomposition methods for ordinal regression using a single multi-output classifier and multiple binary classifiers can be unified under the Error Correcting Output Codes (ECOC) framework developed by Allwein [Allwein et al., 2000]. The ECOC framework is proposed for multi-class classification originally. Take an one-against-all method for 4 class classification problem as an example and let $h_j(\cdot), j = 1, \dots, 4$ be a decision function of the binary classifier distinguishing whether an instance belongs to class j or not. A coding matrix $M \in \{+1, -1, 0\}^{m \times s}$ is defined associating with the decomposition method, where m is the number of classes, and s is the number of the decision functions. Table 3.1 is a coding matrix for 4 class classification problems solved by one-against-all methods. Each row is for one class and

Algorithm 1 Pseudo code for decoder

Input: $Y = \{y_1, \dots, y_n\}$, $L = \{l_1, \dots, l_n | l_i \in \{1, -1\}\}$. l_i is the prediction of pair-wise SVM for $(\mathbf{x}_t, \mathbf{x}_i)$, $\mathbf{x}_t$ is the test point, $\mathbf{x}_i \in X$, $l_i = 1$ indicates the rank of $\mathbf{x}_t$ is higher than that of $\mathbf{x}_i$, otherwise, $\mathbf{x}_t$'s rank is lower.

Output: $\hat{y}_t$, the predicted rank of $\mathbf{x}_t$.

for $k = 1$ **to** m **do**

Let r_k be the number of correct predictions in L assuming the rank of $\mathbf{x}_t$ is k .

Initialize $r_k = 0$.

for $i = 1$ **to** n **do**

if $y_i < k$ **and** $l_i = -1$ **then**

$r_k = r_k + 1$

else

if $y_i > k$ **and** $l_i = 1$ **then**

$r_k = r_k + 1$

end if

end if

end for

Assign prediction accuracy $p_k = \frac{r_k}{n}$.

end for

return $\hat{y}_t = \operatorname{argmax}_k(p_k)$.

each column is for one decision function. For the sake of clear presentation, the first column is labeled as $[1|2, 3, 4]$ for $h_1(\cdot)$, which considers class 1 as a positive class and the rest are negative classes. The rest of the columns are for $h_2(\cdot)$, $h_3(\cdot)$ and $h_4(\cdot)$. The elements in the matrix represent the training targets for different functions and different classes. For example, M_{11} is +1, meaning that $h_1(\cdot)$ uses +1 as a training target for class 1. In the prediction phase, given a test point $\mathbf{x}_t$, the estimated values are obtained from all the classifiers, i.e., $h_1(\mathbf{x}_t), \dots, h_s(\mathbf{x}_t)$. A predefined similarity metric $d(\cdot, \cdot)$ is used to measure the closeness between the k -th row of the coding matrix $M(k)$ and the estimated vector $Z = [h_1(\mathbf{x}_t), \dots, h_s(\mathbf{x}_t)]$. The rank of $\mathbf{x}_t$ is determined by $\text{argmax}_k(d(M(k), Z))$.

Table 3.1: Coding matrix of one-against-all method for 4-class classification

Class	Coding values			
	$[1 2, 3, 4]$	$[2 1, 3, 4]$	$[3 1, 2, 4]$	$[4 1, 2, 3]$
1	+1	-1	-1	-1
2	-1	+1	-1	-1
3	-1	-1	+1	-1
4	-1	-1	-1	+1

The decoder of the POR framework given in Algorithm 1 can be formulated as a special case under the ECOC framework. Table 3.2 is the coding matrix M' of POR for 4 ranks ordinal regression. Each row is for one rank and the proposed decision function $f(\cdot, \cdot)$ in Eq. 3.3 is run on all columns. Different columns represent different subsets of data with different ranks. Column $(\mathbf{x}, \{k\})$ represents $f(\cdot, \cdot)$ running on pairs formed by an input vector and all instances from rank k . $M'_{ij} = +1$ indicates that $(\mathbf{x}_{ik}, \mathbf{x}_{jk})$ whose ranks are i and j respectively is used as a positive training sample, while $M'_{ij} = -1$ indicates that $(\mathbf{x}_{ik}, \mathbf{x}_{jk})$ is used as a negative training sample. $M'_{ii} = 0$ means that samples from the same rank are not used in training. The elements in the i -th row, except for the

diagonal elements, are expected outputs for a test sample with rank i .

Table 3.2: Coding matrix of POR for 4-rank ordinal regression

Rank	Coding values			
	$(\mathbf{x}, \{1\})$	$(\mathbf{x}, \{2\})$	$(\mathbf{x}, \{3\})$	$(\mathbf{x}, \{4\})$
1	0	-1	-1	-1
2	+1	0	-1	-1
3	+1	+1	0	-1
4	+1	+1	+1	0

As described before, in the ECOC framework, the rank of a test point is determined by $\text{argmax}_k(d(M(k), Z))$, where $M(k)$ is the k -th row vector of M and Z is the estimation vector of a test point $\mathbf{x}_t$. For the decoder of POR, let $Z' = [\mathbf{z}_1, \dots, \mathbf{z}_m]$ be the estimation tensor of a test point $\mathbf{x}_t$ and each element $\mathbf{z}_i$ be the prediction vector for the pairs of $\mathbf{x}_t$ and all instances from rank i , i.e., $\mathbf{z}_i = [f(\mathbf{x}_t, \mathbf{x}_{i1}), \dots, f(\mathbf{x}_t, \mathbf{x}_{in_i})]$. Define

$$d(M'(k), Z') = \frac{1}{n - n_k} \sum_{i=1, \dots, m, i \neq k} \frac{n_i + M'_{k,i} \mathbf{1}_{[n_i]} \mathbf{z}_i^T}{2} \quad (3.4)$$

where $\mathbf{1}_{[n_i]}$ is the all-ones vector with length n_i , $M'_{k,i}$ is the element in the k -th row and the i -th column in the coding matrix M' , and n_k and n_i are the number of instances with rank k and i in the training set. If $M'_{k,i} = 1$, the equation $\frac{n_i + M'_{k,i} \mathbf{1}_{[n_i]} \mathbf{z}_i^T}{2}$ inside the summation in Eq. 3.4 returns the number of ones in $\mathbf{z}_i$, while if $M'_{k,i} = -1$, it returns the number of negative ones in $\mathbf{x}_i$. Therefore, Eq. 3.4 is a mathematical representation of line 2 to line 11 in the Algorithm 1. In the Algorithm 1 the rank of a test point is determined by $\text{argmax}_k d(M'(k), Z')$. Clearly, the coder of POR is a special case of the ECOC framework.

3.3 Reducing the Training Complexity

All the instances from different ranks are paired up in POR to train the pairwise SVM. Thus, the total number of training pairs is $O(n^2)$, where n is the number of instances in the original training dataset. However, there are redundant information from these pairs for training. For example, let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ be instances from rank 1, 2, 3 respectively. Intuitively, the ordinal information given by $(\mathbf{x}_1, \mathbf{x}_3)$ has been covered by $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_2, \mathbf{x}_3)$ together. The following theorems prove that this intuition is true, provided that the kernel used in the pairwise SVM satisfies the property (d): $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_r)) + K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_r, \mathbf{x}_l))$. To show that the kernels fulfilling this property exist, a type of pairwise kernels is defined in Eq. 3.5 as an example:

$$K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))^T (\phi(\mathbf{x}_k) - \phi(\mathbf{x}_l)) \quad (3.5)$$

It is necessary to show that the function defined in Eq. 3.5 is a kernel. If it exists a mapping function $\phi(\mathbf{x})$, such that $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$ can be written as the right side of Eq. 3.5, $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$ will be a kernel because it is an explicit inner product in the high dimensional space. Denote $K(\mathbf{x}_i, \mathbf{x}_j)$ as any standard kernel. According to the Mercer Theorem, it exists a mapping function $\phi(\mathbf{x})$ such that $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, and due to $K(\mathbf{x}_i, \mathbf{x}_k) - K(\mathbf{x}_i, \mathbf{x}_l) + K(\mathbf{x}_j, \mathbf{x}_k) - K(\mathbf{x}_j, \mathbf{x}_l) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_k) - \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_l) + \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_k) - \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_l) = (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))^T (\phi(\mathbf{x}_k) - \phi(\mathbf{x}_l)) = K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$, the mapping function $\phi(\mathbf{x})$ in Eq. 3.5 exists. Therefore, $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$ in Eq. 3.5 is a kernel function. It can be checked that besides of property (d), the kernel function in Eq. 3.5 also fulfills the property (a) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_k, \mathbf{x}_l), (\mathbf{x}_i, \mathbf{x}_j))$, (b) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_j, \mathbf{x}_i), (\mathbf{x}_l, \mathbf{x}_k))$ and (c) $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = -K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_l, \mathbf{x}_k))$. Therefore, for any test pair $(\mathbf{x}_t, \mathbf{x}_k)$, the value of decision function $f(\mathbf{x}_t, \mathbf{x}_k) = -f(\mathbf{x}_k, \mathbf{x}_t)$, where $f(\cdot, \cdot)$ is the decision function defined in Eq. 3.3.

The following theorems show that if the kernel defined in Eq. 3.5 is used in the proposed pairwise SVM, the training complexity can be reduced from $O(n^4)$ to $O(n^2)$,

which is same as the complexity of a standard pointwise SVM. Theorem 1 shows that the total number of training points can be reduced to $O(\frac{n^2}{2})$ without any accuracy loss. Based on that, Theorem 2 further proves that the number can be averagely reduced to $O(\frac{n^2}{2m})$, where m is the number of ranks. For the QP problem of a standard pointwise SVM with n training points, the training complexity is $O(n^2)$. If the number of training points increases to $O(n^2)$, the training time is $O(n^4)$. For the proposed pairwise SVM, even though the number of training points is reduced to $O(\frac{n^2}{2m})$, the training complexity is still huge. However, Theorem 3 shows the proposed pairwise SVM with the defined pairwise kernel can be transformed as a new QP problem and $O(n^2)$ time is needed to solve it on the reduced training datasets instead of $O(n^4)$.

In the following part, we will show the proofs. The dual problem of the proposed pairwise SVM can be derived by the standard Lagrangian techniques. The Lagrangian of problem 3.2 is:

$$L_p = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{(i,j) \in I} \varepsilon_{i,j} - \sum_{(i,j) \in I} \alpha_{i,j} (\text{sign}(y_i - y_j) \mathbf{w}^\top \phi(\mathbf{x}_i, \mathbf{x}_j) - (d_{i,j} - \varepsilon_{i,j})) - \sum_{(i,j) \in I} \mu_{i,j} \varepsilon_{i,j} \quad (3.6)$$

The KKT conditions for the primal problem require the following to hold:

$$\begin{aligned} \frac{\partial L_p}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{(i,j) \in I} \text{sign}(y_i - y_j) \alpha_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) = 0 \\ \frac{\partial L_p}{\partial \varepsilon_{i,j}} &= C - \mu_{i,j} - \alpha_{i,j} = 0 \end{aligned} \quad (3.7)$$

By substituting $\mathbf{w}$ and μ of Eq. 3.7 into Eq. 3.6, the dual problem becomes the problem:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{(i,j)(k,l) \in I} \text{sign}(y_i - y_j) \text{sign}(y_k - y_l) \alpha_{i,j} \alpha_{k,l} K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) \\ & - \sum_{(i,j) \in I} \alpha_{i,j} d_{i,j} \\ \text{s.t.} \quad & 0 \leq \alpha_{i,j} \leq C \end{aligned} \quad (3.8)$$

Following the proof in [Brunner et al., 2012], Lemma 1 and Theorem 1 can be proved based on the facts that $d_{i,j} = d_{j,i}$ and $\text{sign}(y_i - y_j) = -\text{sign}(y_j - y_i)$. The proofs are omitted in this thesis.

Lemma 1 *If I is a symmetric index set and $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) = K((\mathbf{x}_j, \mathbf{x}_i), (\mathbf{x}_l, \mathbf{x}_k))$ holds, then there is a solution $\hat{\alpha}$ of the problem 3.8 with $\hat{\alpha}_{ij} = \hat{\alpha}_{ji}$ for all $(i, j) \in I$.*

Theorem 1 *For any solution α^* of the problem 3.8 and for any solution β^* of the following problem 3.9, it holds that $f_{\alpha^*}(a, b) = f_{\beta^*}(a, b)$, where I is the symmetric index set, and I' is defined as $I' = \{(i, j) | (i, j) \in I, (i, j) \in I' \Rightarrow (j, i) \notin I'\}$. $f(\cdot, \cdot)$ is the decision function in Eq. 3.3, i.e., $f_{\alpha^*}(a, b) = \sum_{(i,j) \in I} \alpha_{ij}^* \text{sign}(y_i - y_j) K((\mathbf{x}_i, \mathbf{x}_j), (a, b))$ and $f_{\beta^*}(a, b) = \sum_{(i,j) \in I'} \beta_{ij}^* \text{sign}(y_i - y_j) K((\mathbf{x}_i, \mathbf{x}_j), (a, b))$.*

$$\begin{aligned}
\min_{\beta} \quad & \frac{1}{2} \sum_{(i,j)(k,l) \in I'} \text{sign}(y_i - y_j) \text{sign}(y_k - y_l) \beta_{i,j} \beta_{k,l} K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) \\
& - \sum_{(i,j) \in I'} \beta_{i,j} d_{i,j} \\
\text{s.t.} \quad & 0 \leq \beta_{i,j} \leq 2C
\end{aligned} \tag{3.9}$$

Theorem 1 shows that if there are pairs $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}_j, \mathbf{x}_i)$ in the training set, one of them can be omitted for training. It means that only half of pairs in the training set are necessary to train the pairwise SVM to obtain the same performance as that trained by all pairs. Therefore, the index set of the training set can be reduced to $J = \{(i, j) | i, j \in I, y_i < y_j\}$, so that $\text{sign}(y_i - y_j) < 0$ for all $(i, j) \in J$. Thus, the problem 3.9 is equivalent to the following problem:

$$\begin{aligned}
\min_{\beta} \quad & \frac{1}{2} \sum_{(i,j)(k,l) \in J} \beta_{i,j} \beta_{k,l} K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) - \sum_{(i,j) \in J} \beta_{i,j} d_{i,j} \\
\text{s.t.} \quad & 0 \leq \beta_{i,j} \leq 2C
\end{aligned} \tag{3.10}$$

and the decision function in Eq. 3.3 becomes:

$$f(a, b) = \sum_{(i,j) \in J} \beta_{i,j} K((\mathbf{x}_i, \mathbf{x}_j), (a, b)) \tag{3.11}$$

Lemma 2 *For any solution α^* of the problem 3.12, there is a solution $(\hat{\alpha}_1, \hat{\alpha}_2, \dots, \hat{\alpha}_n)$ of the problem 3.13 such that $\alpha^* = \hat{\alpha}_1 + \hat{\alpha}_2 + \dots + \hat{\alpha}_n$.*

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^\top Q \alpha - b^\top \alpha \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq nC \end{aligned} \quad (3.12)$$

$$\begin{aligned} \min_{\alpha_1, \alpha_2, \dots, \alpha_n} \quad & \frac{1}{2} (\alpha_1 + \dots + \alpha_n)^\top Q (\alpha_1 + \dots + \alpha_n) - b^\top (\alpha_1 + \dots + \alpha_n) \\ \text{s.t.} \quad & 0 \leq \alpha_{1i} \leq C \\ & \dots \\ & 0 \leq \alpha_{ni} \leq C \end{aligned} \quad (3.13)$$

Proof Denote α^* as a solution of the problem 3.12, $G(\alpha) := \frac{1}{2} \alpha^\top Q \alpha - b^\top \alpha$ and $H(\alpha_1, \dots, \alpha_n) := \frac{1}{2} (\alpha_1 + \dots + \alpha_n)^\top Q (\alpha_1 + \dots + \alpha_n) - b^\top (\alpha_1 + \dots + \alpha_n)$. Thus, $G(\alpha) = H(\alpha_1 + \dots + \alpha_n)$, given that $\alpha = \alpha_1 + \dots + \alpha_n$.

Let us define $\hat{\alpha}_1 = \hat{\alpha}_2 = \dots = \hat{\alpha}_n = \frac{1}{n} \alpha^*$. $(\hat{\alpha}_1, \dots, \hat{\alpha}_n)$ is a feasible point of the problem 3.13. Assume that $(\tilde{\alpha}_1, \dots, \tilde{\alpha}_n)$ is a solution of problem 3.13, it will hold that $H(\tilde{\alpha}_1, \dots, \tilde{\alpha}_n) \leq H(\hat{\alpha}_1, \dots, \hat{\alpha}_n)$.

Because $\tilde{\alpha} = \tilde{\alpha}_1 + \dots + \tilde{\alpha}_n$ is a feasible point of the problem 3.12, and $\alpha^* = \hat{\alpha}_1 + \dots + \hat{\alpha}_n$ is an optimal point, it will hold that $H(\hat{\alpha}_1, \dots, \hat{\alpha}_n) = G(\alpha^*) \leq G(\tilde{\alpha}) = H(\tilde{\alpha}_1, \dots, \tilde{\alpha}_n)$. Thus, $H(\tilde{\alpha}_1, \dots, \tilde{\alpha}_n) = H(\hat{\alpha}_1, \dots, \hat{\alpha}_n)$, and $(\hat{\alpha}_1, \dots, \hat{\alpha}_n)$ is a solution of problem 3.13. ■

Theorem 2 For any solution β^* of the problem 3.10 and for any solution γ^* of the following problem 3.14, if the kernels fulfill property (d), then it holds that $f_{\beta^*}(a, b) = f_{\gamma^*}(a, b)$, where $f(\cdot, \cdot)$ is the decision function defined in Eq. 3.11, i.e., $f_{\beta^*}(a, b) = \sum_{(i,j) \in J} \beta_{i,j}^* K((\mathbf{x}_i, \mathbf{x}_j), (a, b))$ and $f_{\gamma^*}(a, b) = \sum_{(i,j) \in L} \gamma_{i,j}^* K((\mathbf{x}_i, \mathbf{x}_j), (a, b))$.

$$\begin{aligned} \min_{\gamma} \quad & \frac{1}{2} \sum_{(i,j)(k,l) \in L} \gamma_{i,j} \gamma_{k,l} K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)) - \sum_{(i,j) \in L} \gamma_{i,j} \\ \text{s.t.} \quad & 0 \leq \gamma_{i,j} \leq 2C + \frac{2C}{n_w} \sum_{t=1}^{w-1} n_t + \frac{2C}{n_{w+1}} \sum_{t=w+1}^m n_t + \frac{2C}{n_w n_{w+1}} \sum_{t=1}^m n_t \quad \text{for } w = y_i \end{aligned} \quad (3.14)$$

where $L = \{(i, j) \in J | y_j = y_i + 1\}$, m is the number of ranks and n_w is the number of instances in the rank w .

Proof The first step is to prove $\sum_{(i,j) \in J} \beta_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) = \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j)$, if define $\gamma_{ij} := \gamma_{p,q}^{w,w+1} := \beta_{p,q}^{w,w+1} + \frac{1}{n_w} \sum_{r=1}^{w-1} \sum_{s \in L_r} \beta_{s,q}^{r,w+1} + \frac{1}{n_{w+1}} \sum_{u=w+1}^m \sum_{s \in L_u} \beta_{p,v}^{w,u} + \frac{1}{n_w n_{w+1}} \sum_{r=1}^{w-1} \sum_{u=w+2}^m \sum_{\substack{s \in L_r \\ v \in L_u}} \beta_{s,v}^{r,u}$ where $w = y_i$.

Define $\beta_{i,j} := \beta_{p,q}^{y_i, y_j}$, which is the Lagrangian multiplier for pair $(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i$ is the p -th instance of rank y_i , and $\mathbf{x}_j$ is the q -th instance of rank y_j . Define $\phi(\mathbf{x}_i, \mathbf{x}_j) := \phi_{p,q}^{y_i, y_j}$ similarly. Denote $B := \sum_{(i,j) \in J} \beta_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) = \sum_{w,z=1 \dots m} \sum_{p \in X_w, q \in X_z} \beta_{p,q}^{w,z} \phi_{p,q}^{w,z}$, where $w = y_i$.

Because the kernel fulfills the property (d), it can be derived that $\phi(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i, \mathbf{x}_r) + \phi(\mathbf{x}_r, \mathbf{x}_j)$. Therefore, if $z - w \geq 2$, then $\beta_{p,q}^{w,z} \phi_{p,q}^{w,z} = \beta_{p,q}^{w,z} \times \frac{1}{n_{w+1}} \sum_{c=1}^{n_{w+1}} (\phi_{p,c}^{w,w+1} + \phi_{c,q}^{w+1,z})$. Then, if $z - (w+1) \geq 2$, the $\phi_{c,q}^{w+1,z}$ in the above equation is expanded recursively. After expansion and merging same terms base on $\phi_{p,c}^{w,w+1}$, the equation of B becomes:

$$\begin{aligned}
B &= \sum_{w=1 \dots m} \sum_{p \in X_w, q \in X_{w+1}} \phi_{p,q}^{w,w+1} (\beta_{p,q}^{w,w+1} + \frac{1}{n_w} \sum_{r=1}^{w-1} \sum_{s \in X_r} \beta_{s,q}^{r,w+1} + \\
&\quad \frac{1}{n_{w+1}} \sum_{u=w+1}^m \sum_{s \in X_u} \beta_{p,v}^{w,u} + \frac{1}{n_w n_{w+1}} \sum_{r=1}^{w-1} \sum_{u=w+2}^m \sum_{\substack{s \in X_r \\ v \in X_u}} \beta_{s,v}^{r,u}) \\
&= \sum_{w=1 \dots m} \sum_{p \in X_w, q \in X_{w+1}} \phi_{p,q}^{w,w+1} \gamma_{p,q}^{w,w+1} \\
&= \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) \tag{3.15}
\end{aligned}$$

Therefore, $\sum_{(i,j) \in J} \beta_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) = \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j)$.

Assume that β^* is a solution of the problem 3.10, and rewrite $\beta_{i,j}^*$ as $\beta_{p,q}^{*y_i, y_j}$. Define $\bar{\gamma}_{ij} := \bar{\gamma}_{p,q}^{w,w+1} := \beta_{p,q}^{*w,w+1} + \frac{1}{n_w} \sum_{r=1}^{w-1} \sum_{s \in X_r} \beta_{s,q}^{*r,w+1} + \frac{1}{n_{w+1}} \sum_{u=w+1}^m \sum_{s \in X_u} \beta_{p,v}^{*w,u} + \frac{1}{n_w n_{w+1}} \sum_{r=1}^{w-1} \sum_{u=w+2}^m \sum_{\substack{s \in X_r \\ v \in X_u}} \beta_{s,v}^{*r,u}$, where $(i, j) \in L$ and $w = y_i$. Because $0 \leq \beta_{i,j}^* \leq 2C$ for any $(i, j) \in J$, $\bar{\gamma}_{ij}$ is a feasible point of 3.14. Therefore, $f_{\beta^*}(a, b) = \phi(a, b) \sum_{(i,j) \in J} \beta_{i,j}^* \phi(\mathbf{x}_i, \mathbf{x}_j) = \phi(a, b) \sum_{(i,j) \in L} \bar{\gamma}_{ij} \phi(\mathbf{x}_i, \mathbf{x}_j) = f_{\bar{\gamma}}(a, b)$.

The next step is to prove that $\bar{\gamma}$ is a solution of 3.14. Let $K_{ij,kl}$ be short for $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$. Denote $L_D(\beta)$ and $L'_D(\gamma)$ as the objective functions of problems 3.10 and 3.14, and they can be written as two parts as following,

$$\begin{aligned} L_D(\beta) &= \underbrace{\frac{1}{2} \sum_{(i,j)(k,l) \in J} \beta_{i,j} \beta_{k,l} K_{ij,kl}}_{L_{D1}(\beta)} - \underbrace{\sum_{(i,j) \in J} \beta_{i,j} d_{i,j}}_{L_{D2}(\beta)} \\ L'_D(\gamma) &= \underbrace{\frac{1}{2} \sum_{(i,j)(k,l) \in L} \gamma_{i,j} \gamma_{k,l} K_{ij,kl}}_{L'_{D1}(\gamma)} - \underbrace{\sum_{(i,j) \in L} \gamma_{i,j}}_{L'_{D2}(\gamma)} \end{aligned}$$

It can be shown that $L_{D1}(\beta) = L'_{D1}(\gamma)$:

$$\begin{aligned} L_{D1}(\beta) &= \frac{1}{2} \sum_{(i,j)(k,l) \in J} \beta_{i,j} \beta_{k,l} K_{ij,kl} \\ &= \frac{1}{2} \sum_{(k,l) \in J} \beta_{k,l} \phi(\mathbf{x}_k, \mathbf{x}_l) \sum_{(i,j) \in J} \beta_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) \\ &= \frac{1}{2} \sum_{(k,l) \in J} \beta_{k,l} \phi(\mathbf{x}_k, \mathbf{x}_l) \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) \\ &= \frac{1}{2} \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) \sum_{(k,l) \in J} \beta_{k,l} \phi(\mathbf{x}_k, \mathbf{x}_l) \\ &= \frac{1}{2} \sum_{(i,j) \in L} \gamma_{i,j} \phi(\mathbf{x}_i, \mathbf{x}_j) \sum_{(k,l) \in L} \gamma_{k,l} \phi(\mathbf{x}_k, \mathbf{x}_l) \\ &= L'_{D1}(\gamma) \end{aligned} \tag{3.16}$$

Similarly, based on $d_{i,j} = d_{i,r} + d_{r,j}$, it can be proved that $L_{D2}(\beta) = L'_{D2}(\gamma)$. Therefore, based on Lemma 2, $\bar{\gamma}$ is a solution of 3.14 and for any solution γ^* , $f_{\gamma^*}(a, b) = f_{\bar{\gamma}}(a, b)$. ■

Theorem 2 pinpoints that if the kernel fulfilling the property (d), the pairwise SVM can be trained by pairs of instances from adjacent ranks instead of all pairs without any performance loss.

Theorem 3 *For any solution γ^* of the problem 3.14 and for any solution μ^* of the following problem 3.17, if the kernels can be written as the formulation in Eq. 3.5, then*

it holds that $f_{\gamma^*}(a, b) = g_{\mu^*}(a, b)$, where $f_{\gamma^*}(a, b) = \sum_{(i,j) \in L} \gamma_{i,j}^* K((\mathbf{x}_i, \mathbf{x}_j), (a, b))$ and $g_{\mu^*}(a, b) = \sum_i \mu_i^* (K(\mathbf{x}_i, a) - K(\mathbf{x}_i, b))$ where $K(\mathbf{x}_i, a)$ is a standard pointwise kernel.

$$\begin{aligned}
& \min_{\mu} \quad \frac{1}{2} \mu^\top Q \mu - \mathbf{1}^\top z \\
& \text{s.t.} \quad \mathbf{1}^\top \mu = 0 \\
& 0 \leq z_i \leq 2Cn + 2Cn \frac{\sum_{t=1}^{w-1} n_t}{n_w} + 2Cn \frac{\sum_{t=w+1}^m n_t}{n_{w+1}} + 2Cn \frac{\sum_{t=1}^m n_t}{n_w n_{w+1}} \quad \text{for } w = y_i \\
& 0 \leq z_i - \mu_{i,j} \leq 2Cn + 2Cn \frac{\sum_{t=1}^{w-1} n_t}{n_w} + 2Cn \frac{\sum_{t=w+1}^m n_t}{n_{w+1}} + 2Cn \frac{\sum_{t=1}^m n_t}{n_w n_{w+1}} \quad \text{for } w = y_i
\end{aligned} \tag{3.17}$$

where Q is the kernel matrix with $Q_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$ and m is the number of ranks. n is the total number of training points and n_w is the number of instances in the rank w .

Proof Let $K_{ij,kl}$ be short for $K((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))$, K_{ij} be short for $K(\mathbf{x}_i, \mathbf{x}_j)$. Set $\gamma_{1i} = \sum_j \gamma_{i,j}$, $\gamma_{2j} = \sum_i \gamma_{i,j}$, $\gamma_{1k} = \sum_l \gamma_{k,l}$, $\gamma_{2l} = \sum_k \gamma_{k,l}$. Thus, $\sum_i \gamma_{1i} = \sum_j \gamma_{2j}$. The objective function of problem 3.14 can be rewritten as following:

$$\begin{aligned}
J(\gamma) &= \min_{\gamma} \frac{1}{2} \sum_{(i,j)(k,l) \in L} \gamma_{i,j} \gamma_{k,l} K_{ij,kl} - \sum_{(i,j) \in L} \gamma_{i,j} \\
&= \min_{\gamma} \frac{1}{2} \sum_{(i,j)(k,l) \in L} \gamma_{i,j} \gamma_{k,l} (K_{ik} - K_{il} - K_{jk} + K_{jl}) - \sum_{(i,j) \in L} \gamma_{i,j} \\
&= \min_{\gamma} \frac{1}{2} \left(\sum_{i,k} \gamma_{1i} \gamma_{1k} K_{ik} - \sum_{i,l} \gamma_{1i} \gamma_{2l} K_{il} - \sum_{j,k} \gamma_{2j} \gamma_{1k} K_{jk} + \sum_{j,l} \gamma_{2j} \gamma_{2l} K_{jl} \right) - \sum_i \gamma_{1i} \\
&= \min_{\gamma} \frac{1}{2} (\gamma_1 - \gamma_2)^\top Q (\gamma_1 - \gamma_2) - \mathbf{1}^\top \gamma_1
\end{aligned} \tag{3.18}$$

where $\gamma_1 = [\gamma_{11}, \dots, \gamma_{1n}]$, $\gamma_2 = [\gamma_{21}, \dots, \gamma_{2n}]$.

Let $\mu = \gamma_1 - \gamma_2$, $z = \gamma_1$, then $J(\gamma) = \frac{1}{2} \mu^\top Q \mu - \mathbf{1}^\top z$, which is the objective function of problem 3.17. Assume γ^* is a solution of the problem 3.14, according to Lemma 2, $\bar{\mu} = \gamma_1^* - \gamma_2^*$ is a solution of 3.17, and $g_{\mu^*}(a, b) = g_{\bar{\mu}}(a, b)$.

The next step is to prove that $g_{\bar{\mu}}(a, b) = f_{\gamma^*}(a, b)$:

$$\begin{aligned}
f_{\gamma^*}(a, b) &= \sum_{(i,j) \in L} \gamma_{i,j}^* K((\mathbf{x}_i, \mathbf{x}_j), (a, b)) \\
&= (\phi(a) - \phi(b)) \sum_{(i,j) \in L} \gamma_{i,j}^* (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \\
&= (\phi(a) - \phi(b)) \left(\sum_i \gamma_{1i}^* \phi(\mathbf{x}_i) - \sum_j \gamma_{2j}^* \phi(\mathbf{x}_j) \right) \\
&= (\phi(a) - \phi(b)) \sum_i (\gamma_{1i}^* - \gamma_{2i}^*) \phi(\mathbf{x}_i) \\
&= \sum_i \bar{\mu} (K(\mathbf{x}_i, a) - K(\mathbf{x}_i, b))
\end{aligned} \tag{3.19}$$

■

Theorem 3 pinpoints that if the proposed algorithm employs a kernel with the format of Eq. 3.5, the pairwise SVM can be trained directly by the original training instances via solving the QP problem 3.17. Therefore, the number of training pairs is reduced to $O(n)$, and the training complexity is $O(n^2)$ as the standard SVM.

3.4 Experimental Results

To evaluate the performance of the proposed algorithm, it is compared with three state-of-the-methods on 12 widely used benchmark datasets. Table 3.3 lists their details. Each of them has no more than 300 data points. Table 3.3 lists the partitions between training set and testing set. K indicates the feature dimensions and Q indicates the number of ranks. The last column shows the number of points in different ranks. Note that some of them are highly imbalance. More information of the datasets can be found in [Chu and Keerthi, 2007] and [Sánchez-Monedero et al., 2013]. Two metrics are used to evaluate the performance of the methods. The first one is mean zero-one error (MZE) defined by $e = \frac{1}{|T|} \sum_{\mathbf{x}_t \in T} \mathbb{1}[\hat{y}_t \neq y_t]$, where T is the testing set, $|T|$ is its size. y_t is the ground truth of $\mathbf{x}_t$, $\hat{y}_t$ is the prediction for y_t , and $\mathbb{1}[\cdot]$ is the indicator function. The second one is mean absolute error (MAE) defined by $e = \frac{1}{|T|} \sum_{\mathbf{x}_t \in T} |\hat{y}_t - y_t|$.

Table 3.3: Ordinal regression benchmarks

Dataset	Partition	K	Q	Class Distribution
contact-lenses	18/6	6	3	(15,5,4)
pasture	27/9	25	3	(12,12,12)
squash-stored	39/13	51	3	(23,21,8)
squash-unstored	39/13	52	3	(24,24,4)
tae	113/38	54	3	(49,50,52)
newthyroid	161/54	5	3	(30,150,35)
bonrate	42/15	37	5	(6,33,12,5,1)
automobile	153/52	71	6	(3,22,67,54,32,27)
pyrim5	50/24	27	5	(7,28,17,12,10)
machine5	150/59	6	5	(152,27,13,7,10)
pyrim10	50/24	27	10	(2,2,14,14,13,5,10,4,3,7)
machine10	150/59	6	10	(115,37,21,6,8,5,3,4,4,6)

Three state-of-the-art methods, SVOR [Chu and Keerthi, 2007], RED-SVM [Lin and Li, 2012], and PCDOC [Sánchez-Monedero et al., 2013] are compared with the proposed algorithm POR. Gaussian kernel is used in all of the methods. The hyper-parameters C and γ are selected respectively from $\{2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^0, 2^1, 2^2, 2^3, 2^4, 2^5\}$ and $\{2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^0\}$ using 5-fold cross validation. Some features in these datasets are binary but the others are real numbers. Normalization does have impacts of most of classifiers, including SVM. Two types of normalization, normalizing all features and normalizing only none-binary features are considered. Including the original features, there are three options. They are selected through cross-validation. More clearly, cross-validation is used to select the best hyper-parameters and the best corresponding normalization scheme simultaneously. $\mathbf{x}_i = \frac{\mathbf{x}_i - \mu_i}{\theta_i}$, where μ_i and θ_i are respectively the mean and variance of i -th feature and $\mathbf{x}_i$ is the i -th feature value of an input vector, are used

as the normalization method. For SVOR [Chu and Keerthi, 2007], RED-SVM [Lin and Li, 2012], and PCDOC [Sánchez-Monedero et al., 2013], the model selection methods reported in the original papers are used. For fair comparison, the same experimental settings reported in [Sánchez-Monedero et al., 2013] and [Chu and Keerthi, 2007] are applied to the first eight datasets and the rest, respectively. According to the experimental settings in [Sánchez-Monedero et al., 2013] and [Chu and Keerthi, 2007], the results from the first eight datasets are average of 30 trials, while the results from the last four datasets are average of 20 trials. Tables 3.4 and 3.5 list the mean errors and their standard derivation/variance from the first eight databases. The best performance is highlighted. None of the algorithms can get best results for all eight datasets. In terms of MZE, POR can achieve the best results on three datasets, while the others can achieve the best results on no more than two datasets. In terms of MAE, PCDOC wins four times, while POR is the best for three datasets.

Tables 3.6 and 3.7 list the results on the discrete regression datasets. In terms of both

Table 3.4: Mean zero-one error (MZE) on real ordinal regression datasets

	contact-lenses	pasture	squash-stored	squash-unstored
RED-SVM	0.300±0.111	0.352 ± 0.134	0.336 ± 0.104	0.251 ± 0.086
SVOR	0.367 ± 0.127	0.333 ± 0.120	0.361 ± 0.118	0.236 ± 0.103
PCDOC	0.311 ± 0.095	0.344 ± 0.103	0.315±0.123	0.305 ± 0.084
POR	0.344 ± 0.166	0.304±0.149	0.359 ± 0.111	0.226±0.093
	Tae	newthyroid	bondrate	automobile
RED-SVM	0.478 ± 0.074	0.031 ± 0.022	0.447±0.073	0.316 ± 0.055
SVOR	0.410±0.066	0.031 ± 0.021	0.453 ± 0.092	0.361 ± 0.076
PCDOC	0.418 ± 0.064	0.027±0.020	0.460 ± 0.101	0.322 ± 0.060
POR	0.420 ± 0.094	0.030 ± 0.024	0.449 ± 0.150	0.303±0.102

Table 3.5: Mean absolute error (MAE) on real ordinal regression datasets

	contact-lenses	pasture	squash-stored	squash-unstored
RED-SVM	0.378 ± 0.169	0.359 ± 0.142	0.346 ± 0.110	0.251 ± 0.086
SVOR	0.506 ± 0.167	0.333 ± 0.120	0.372 ± 0.126	0.239 ± 0.109
PCDOC	0.367 ± 0.154	0.348 ± 0.104	0.326 ± 0.141	0.305 ± 0.084
POR	0.489 ± 0.113	0.304 ± 0.149	0.369 ± 0.104	0.226 ± 0.093
	Tae	newthyroid	bondrate	automobile
RED-SVM	0.515 ± 0.087	0.031 ± 0.022	0.598 ± 0.088	0.393 ± 0.073
SVOR	0.461 ± 0.081	0.031 ± 0.021	0.591 ± 0.102	0.424 ± 0.090
PCDOC	0.457 ± 0.071	0.027 ± 0.020	0.568 ± 0.126	0.397 ± 0.093
POR	0.539 ± 0.072	0.030 ± 0.024	0.556 ± 0.114	0.431 ± 0.062

Table 3.6: Mean zero-one error (MZE) on discrete regression datasets

	pyrim5	machine5	pyrim10	machine10
RED-SVM	0.413 ± 0.063	0.264 ± 0.010	0.762 ± 0.021	0.572 ± 0.013
SVOR	0.517 ± 0.086	0.431 ± 0.054	0.719 ± 0.066	0.655 ± 0.045
PCDOC	0.483 ± 0.088	0.178 ± 0.036	0.704 ± 0.071	0.385 ± 0.054
POR	0.442 ± 0.075	0.180 ± 0.048	0.641 ± 0.196	0.333 ± 0.064

of MZE and MAE, POR wins two times; RED-SVM and PCDOC wins one time, while SVOR cannot get anyone best results. Table 3.8 summarizes the number of wins from the 12 datasets. The average error differences defined by $v_j = \frac{1}{12} \sum_{i=1}^{12} e_{ji} - e_i^{best}$, where e_{ji} is the mean error of method j on i -th dataset, e_i^{best} is the lowest error on i -th dataset, are also given. For MZE, POR performs better than all the baselines For MAE, the proposed

and algorithm PCDOC performs similarly.

Table 3.7: Mean absolute error (MAE) on discrete regression datasets

	pyrim5	machine5	pyrim10	machine10
RED-SVM	0.454±0.086	0.478 ± 0.031	1.304 ± 0.040	0.842 ± 0.022
SVOR	0.615 ± 0.127	0.462 ± 0.062	1.294 ± 0.046	0.990 ± 0.026
PCDOC	0.552 ± 0.116	0.202 ± 0.046	1.088 ± 0.159	0.494 ± 0.082
POR	0.541 ± 0.049	0.200 ± 0.038	1.133 ± 0.091	0.476 ± 0.029

Table 3.8: Win-loss summary

	#win on MZE	Var on MZE	#win on MAE	Var on MAE
RED-SVM	3	0.0535	2	0.0897
SVOR	1	0.0874	0	0.1203
PCDOC	3	0.0290	5	0.0214
POR	5	0.0176	5	0.0350

3.5 Summary

In this chapter, a new pairwise algorithm is proposed to solve the ordinal regression problems with small datasets. The number of training data is increased quadratically using a pairwise method. SVM is revised such that the ordinal information is embedded in the constraints. Pairwise kernel is also used in this revised SVM. In fact, any binary classifier with good performance can be used to predict the larger or smaller relationship of pairs. Finally, a decoder is designed to recover the category of a test point from the bi-

nary predictions for pairs of this test points and all training points. It is proved that time complexity of solving the proposed QP problem can be reduced from the $O(n^4)$ to $O(n^2)$ without any accuracy loss if suitable kernels are used. The experiments show that the proposed algorithm is comparable with the state-of-the-art methods.

Chapter 4

Deep Ordinal Regression Based on Data Relationship for Small Datasets

4.1 Overview

From the data point of view, most of existing ordinal regression approaches are pointwise. Their input spaces are the raw feature spaces and the parameters are learned from individual data points. Therefore, the relationship between instances is not explored explicitly. In the previous work, the pairwise relationship between instances has been studied for ordinal regression. In this chapter, we propose the second approach extends the data relationship representation from pairs to triplets based on deep neural networks. The triplets whose elements are from different ranks are used to explore the ordinal data relationship. The intuition is simple: if a method can predict that the rank of an input $\mathbf{x}$ is greater than $k - 1$ and smaller than $k + 1$, then the rank label of $\mathbf{x}$ will be k . Therefore, the proposed approach transforms a m -rank ordinal regression problem to m binary classification problems with triplets as inputs. And the k -th classifier answers the question: “Is the rank of $\mathbf{x}$ greater than $k - 1$ and smaller than $k + 1$?” One of state-of-the-art ordinal regression methods RED-SVM [Lin and Li, 2012] trained a binary classifier to answer “Is the rank of $\mathbf{x}$ greater than k or not?” Comparing with RED-SVM, the proposed question

in this chapter is more precise so that it is straightforward to recover the rank label from the answers.

Being able to automatically extract high-level features from raw data, deep neural networks (DNNs), such as convolutional neural networks (CNNs), have attracted great attention in these several years and performed very well on many classification problems. However, there are very few works to employ DNNs to ordinal regression problems. Niu et al. [Niu et al., 2016] have recently adopted CNNs for age estimation and claimed that they are the first one to address ordinal regression problems by using CNNs. Their method is a pointwise approach. The training instances are inputted to CNNs individually and the relationships between input instances are not considered. Moreover, as traditional deep learning methods, their method is more applicable for large scale datasets. To our best knowledge, there is no any existing work employing DNNs to extract and represent high-level features describing relationship among data instances for ordinal regression problems.

Generally speaking, to train a deep neural network, a large training dataset is necessary. Using deep learning on small datasets is challenging, but many real-world ordinal regression problems are in fact small data problems. In this work, representing instances relationship and small dataset problems are two challenges that we are targeting. In the proposed approach, DNNs are adopted to automatically extract high-level features from the triplets with instances from different categories, and a m -rank ordinal regression problem is transformed to m binary classification problems. For each rank k , a separate DNN is trained for the corresponding binary classification problem. Because the distance between every two adjacent ranks can be different, separate DNNs are used instead of one multi-class CNN. In the testing phase, triplets are formed by a testing instance and other instances with known ranks. A decoder is designed to estimate the rank of the testing instance based on the outputs of the network. A significant benefit of the proposed approach is data augmentation. If the size of a training dataset is n , the number of triplets used for

training the CNNs will be $O(n^3)$. Therefore, the proposed approach makes deep learning on small datasets possible. Experimental results on the historical color image benchmark and MSRA image search datasets demonstrate that the proposed algorithm outperforms the traditional deep learning approach and is comparable with other state-of-the-art methods, which are highly based on prior knowledge to design effective features.

4.2 A Convolutional Neural Network for Ordinal Regression

An ordinal regression problem with m ranks denoted by $Y = \{1, 2, \dots, m\}$ is considered, where the natural order of the numbers in Y indicates the order of the ranks. A training set with labeled instances $T = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in X, y_i \in Y\}$ is given, where $\mathbf{x}$ is the input space. The target is to predict the rank $y_t \in Y$ of an input $\mathbf{x}_t \in X$. In the rest of this section, the outline of the proposed approach will be provided first, and then its key components, including a pre-trained CNN and a decoding scheme will be presented.

4.2.1 The Proposed Approach

The proposed deep neural network for ordinal regression is to predict rank label by answering the questions: “Is $k - 1 < y_t < k + 1$ true?” for all $k \in Y$. Obviously, if the answer is “yes” for a certain k , then the predicted rank label y_t will be k . In the proposed approach, for each rank k in Y , a separate binary classifier is trained to answer the above question. Since convolutional neural networks are used in the current implementation, the proposed method is named convolutional neural network for ordinal regression (CNNOR). Algorithm 2 gives the pseudo code of the CNNOR. It consists of three steps: pre-training (line 1), training (line 2-6) and decoding (line 7-11). Given a training set T with labeled instances and a testing point $\mathbf{x}_t$, the goal of CNNOR is to predict the rank of $\mathbf{x}_t$ being most likely between which two ranks. In other words, CNNOR aims to predict

whether a triplet $(\mathbf{x}_i, \mathbf{x}_t, \mathbf{x}_j)$ is consistent in order, where “consistent” means $\mathbf{x}_i \in X_{k-1}$, $\mathbf{x}_t \in X_k$, and $\mathbf{x}_j \in X_{k+1}$. Therefore, Algorithm 2 starts from training CNN_{reg} which aims to make the ranks of $\mathbf{x}_i$ and $\mathbf{x}_j$ in order first for all possible triplets $(\mathbf{x}_i, \mathbf{x}_t, \mathbf{x}_j)$.

Algorithm 2 Pseudo code of CNNOR

Input: Training set $T = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in X, y_i \in Y\}$ where X is an input space, $Y = \{1, 2, \dots, m\}$, and a test point $\mathbf{x}_t \in X$.

Output: y_t , the predicted rank of $\mathbf{x}_t$.

- 1: Train CNN_{reg} on T by minimizing Eq. 4.1.
 - 2: Split T into m subsets $X_1, \dots, X_m$ with $X_k = \{\mathbf{x}_i | (\mathbf{x}_i, y_i) \in T, y_i = k\}$ for $k = 1, \dots, m$.
 - 3: **for** $k = 1$ to m **do**
 - 4: Construct positive and negative triplet sets D_k^+ and D_k^- using Eq. 4.2. (Denote $D_k = D_k^+ \cup D_k^-$)
 - 5: Train CNN_k on D_k with weights initialized from the pre-trained CNN_{reg} model.
 - 6: **end for**
 - 7: **for** $k = 1$ to m **do**
 - 8: Construct d test triplets D_k^t by Eq. 4.3.
 - 9: Input D_k^t to CNN_k and assign $p_k = \frac{c_k}{d}$ where c_k is the number of positive predictions for rank k .
 - 10: **end for return** $y_t = \arg\max_k(p_k)$.
-

Let $\phi(\mathbf{x})$ denote the output of the network CNN_{reg} which is an one-dimensional real value, and $\mathbf{x}$ be an instance. The objective of CNN_{reg} is to learn a function $\phi(\cdot)$ which maps $\mathbf{x}_i$ and $\mathbf{x}_j$ to a real line such that $\phi(\mathbf{x}_i) < \phi(\mathbf{x}_j)$ if $y_i < y_j$. Hence, we construct the training set with lists $(\mathbf{x}_i^1, \mathbf{x}_i^2, \dots, \mathbf{x}_i^k, \dots, \mathbf{x}_i^m)$ as instances, where $\mathbf{x}_i^k$ is from rank k .

Figure 4.1: Illustration of ordinal loss for a 3-rank problem.

The loss function is defined by Eq. 4.1,

$$l = \sum_{i=1}^n \sum_{k=1}^{m-1} \max(0, g + \mathbf{w}^\top \phi(\mathbf{x}_i^k) - \mathbf{w}^\top \phi(\mathbf{x}_i^{k+1})) \quad (4.1)$$

where $\mathbf{x}_i^k$ is from the k -th rank, m is the number of ranks and n is the size of training set (i.e., the number of lists in the training set), g is a hyperparameter which controls the margin of mapped value between adjacent ranks. Eq. 4.1 is named as Ordinal Loss, which calculates the total error of all pairs of instances if their orders are incorrect or their margin is smaller than g . Figure 4.1 illustrates the Ordinal Loss for a 3-rank ordinal regression problem. The axis is the real line which is the range of $\phi(\cdot)$. The red circles are the set of points mapped from instances of the rank 1, i.e., $\{\phi(\mathbf{x}_i) | y_i = 1\}$. The green crosses are the mapped points from the rank 2, and the blue stars are those from the rank 3. g in Eq. 4.1 is the expected margin between two adjacent ranks as shown in Figure 4.1. The Ordinal Loss only counts the errors from pairs of adjacent ranks, e.g., e_1 , e_2 and e_3 . But other errors such as e_4 are not considered explicitly, because if both $(\phi(\mathbf{x}_1), \phi(\mathbf{x}_2))$ and $(\phi(\mathbf{x}_2), \phi(\mathbf{x}_3))$ are in order, it can be inferred that $(\phi(\mathbf{x}_1), \phi(\mathbf{x}_3))$ are also in order. In other words, when e_2 is minimized, e_4 is minimized implicitly.

Then, the next step of Algorithm 2 is to learn a separate CNN for each rank to extract high level features describing ordinal relationship. For each rank k , a new training set with triplets as instances is derived. In line 2 of Algorithm 2, the training set T is split into m subsets $X_1, \dots, X_m$ according to the rank labels. All the instances in the subset

X_k have the same rank label which is k . The new training set D_k is derived as Eq. 4.2:

$$\begin{aligned}
D_k &= D_k^+ \cup D_k^- \\
D_k^+ &= \begin{cases} \{((\mathbf{x}_p, \mathbf{x}_j), +1) | \mathbf{x}_p \in X_1, \mathbf{x}_j \in X_2\} & \text{if } k = 1 \\ \{((\mathbf{x}_i, \mathbf{x}_p), +1) | \mathbf{x}_i \in X_{m-1}, \mathbf{x}_p \in X_m\} & \text{if } k = m \\ \{((\mathbf{x}_i, \mathbf{x}_p, \mathbf{x}_j), +1) | \mathbf{x}_i \in X_{k-1}, \mathbf{x}_p \in X_k, \mathbf{x}_j \in X_{k+1}\} & \text{otherwise} \end{cases} \\
D_k^- &= \begin{cases} \{((\mathbf{x}_p, \mathbf{x}_j), -1) | \mathbf{x}_p \in X_r, 1 < r \leq m, \mathbf{x}_j \in X_2\} & \text{if } k = 1 \\ \{((\mathbf{x}_i, \mathbf{x}_p), -1) | \mathbf{x}_p \in X_r, 1 \leq r < m, \mathbf{x}_i \in X_{m-1}\} & \text{if } k = m \\ \{((\mathbf{x}_i, \mathbf{x}_p, \mathbf{x}_j), -1) | \mathbf{x}_i \in X_{k-1}, \mathbf{x}_j \in X_{k+1}, \mathbf{x}_p \in X_r, r \neq k\} & \text{otherwise} \end{cases}
\end{aligned} \tag{4.2}$$

D_k^+ is a positive training set which includes triplets with three elements from ranks $k - 1$, k and $k + 1$ and the negative training set D_k^- includes the triplets whose middle elements are from other ranks and the other two elements are from the ranks $k - 1$ and $k + 1$. Because there is no previous rank for rank 1, the training set is formed by pairs $(\mathbf{x}_p, \mathbf{x}_j)$ where $\mathbf{x}_j \in X_2$. The classifier for rank 1 aims to decide whether the rank of $\mathbf{x}_i$ is smaller than $\mathbf{x}_j$. Similarly, the training samples for rank m is the pairs $(\mathbf{x}_i, \mathbf{x}_p)$ where $\mathbf{x}_i \in X_{m-1}$. Based on the derived training set, CNN_k is fine-tuned from the pre-trained CNN_{reg} in line 5 of Algorithm 2. Each CNN_k has a binary output, indicating whether the input is consistent in order or not.

In the testing phase, as shown in line 7-11, given a testing point $\mathbf{x}_t$, the test triplets are formed for each CNN_k according to Eq. 4.3.

$$D_k^t = \begin{cases} \{(\mathbf{x}_t, \mathbf{x}_j) | \mathbf{x}_j \in X_2\} & \text{if } k = 1 \\ \{(\mathbf{x}_i, \mathbf{x}_t) | \mathbf{x}_i \in X_{m-1}\} & \text{if } k = m \\ \{(\mathbf{x}_i, \mathbf{x}_t, \mathbf{x}_j) | \mathbf{x}_i \in X_{k-1}, \mathbf{x}_j \in X_{k+1}\} & \text{otherwise} \end{cases} \tag{4.3}$$

For each rank k , d pairs of $\mathbf{x}_i$ and $\mathbf{x}_j$ are randomly selected from X_{k-1} and X_{k+1} and they are constructed as triplets with the testing point $\mathbf{x}_t$. The decision for the rank label of $\mathbf{x}_t$ i.e., y_t is made by majority voting. Each classifier k predicts how many triplets (c_k) of the d inputs are consistent in order respect to rank k , and the percentage $\frac{c_k}{d}$ indicates the probability that $\mathbf{x}_t$ belongs to rank k . Finally, the prediction of y_t is assigned to the rank k with maximum probability. It should be pointed out that the proposed approach produces a set of testing instances for one testing point. Through reusing training data, it increases both training and testing data to overcome the weaknesses of traditional deep learning for small dataset problems. Increasing the size of the testing set is not always an issue because of the advancement of hardware and no real-time requirements in some applications e.g., healthcare.

4.2.2 The Architecture of CNNOR

For a m -rank ordinal regression problem, CNNOR includes one CNN_{reg} and m CNNs, each of which is a binary classifier for one rank. Figure 4.2 shows the architecture of CNN_{reg} for a 5-rank ordinal regression problem and the network CNN_k for the rank k . The input of CNN_{reg} is a list $(\mathbf{x}_1, \dots, \mathbf{x}_m)$ which consists of m images for m -rank ordinal regression problem. Each image $\mathbf{x}_i$ in $(\mathbf{x}_1, \dots, \mathbf{x}_m)$ is inputted into one of the networks, and all the m networks share the same weights. All the networks in CNN_{reg} have one output neuron, and the output value of the i -th network represents $\phi(\mathbf{x}_i)$. All the m outputs are inputted to the Ordinal Loss layer to minimize the loss function in Eq. 4.1. At each iterations of training, the error of the loss layer is back propagated to all networks in CNN_{reg} .

Once the well-trained CNN_{reg} is obtained, its weights are used to initialize the weights of the networks in CNN_k . As shown in Figure 4.2b, the input of CNN_k is a triplet $(\mathbf{x}_i, \mathbf{x}_k, \mathbf{x}_j)$, which consists of three images. Each of the three images is inputted to one of the networks denoted as G_i , G_k and G_j in Figure 4.2b, and all the three networks share the

(a) The architecture of CNN_{reg}

(b) The architecture of CNN_k

Figure 4.2: The architecture of CNNOR for a 5-rank problem.

same weights. CNN_k with $k = 1$ ($k = m$) in Figure 4.2b only has G_k and G_j (G_i). The layers and the parameters of the networks in CNN_k are same as those in CNN_{reg} except for the last fully-connected layer. The last fully-connected layer of CNN_{reg} has only one output neuron, but it is removed in CNN_k and the features extracted from the previous fully-connected layer are passed to the Diff layer as shown Figure 4.2b. Because CNN_k is trained to model the rank order of $(\mathbf{x}_i, \mathbf{x}_k, \mathbf{x}_j)$ by exploring the data relationship, the Diff layer is used to represent the ordinal information within triplets. Based on the assumption that the distance between instances in the mapped feature space indicates the distance

between their ranks, the Diff layer combines two parts of features $\psi(\mathbf{x}_k) - \psi(\mathbf{x}_i)$ and $\psi(\mathbf{x}_j) - \psi(\mathbf{x}_k)$, where $\psi(\mathbf{x}_i)$ is the feature vectors exacted from the last fully-connected layers of G_i . It can be concluded from the proposed architecture, although the augmented dataset has n^3 triplets, CNNOR is computational feasible even for large datasets. As shown in Figure 4.2b, the elements of a triplet are processed individually before the Diff layer. For all $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$, only unique $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_k$ are necessary to be computed. And G_i , G_k and G_j share weights. Therefore, before the Diff layer, the computation cost of CNN_k is same as a standard CNN. The operation of the Diff layer is a simple subtraction, which is not an issue for a modern hardware even for huge data.

4.2.3 The Decoder Based on Majority Voting

A simple decoder is designed to predict the rank label of a testing point $\mathbf{x}_t$ from the outputs of CNN_k as shown in line 7-11 of Algorithm 2. Table 4.1 is the coding matrix of CNNOR for a 5-rank example. Each row is for one rank and each column is for one CNN_k . The elements in the matrix represent the training targets for different CNN_k and different ranks. For example, the first column of Table 4.1 is labeled as $(\mathbf{x}_t, 2)$, which represents a testing triplet constructed as Eq. 4.3 for CNN_I , and rank 1 is considered as a positive rank and the rest are negative ranks. In the testing phase, d triplets for each

Table 4.1: Coding matrix of CNNOR

	$(\mathbf{x}_t, 2)$	$(1, \mathbf{x}_t, 3)$	$(2, \mathbf{x}_t, 4)$	$(3, \mathbf{x}_t, 5)$	$(4, \mathbf{x}_t)$
1	+1	-1	-1	-1	-1
2	-1	+1	-1	-1	-1
3	-1	-1	+1	-1	-1
4	-1	-1	-1	+1	-1
5	-1	-1	-1	-1	+1

column are predicted and the rank with maximum positive predications is regarded as the rank label of $\mathbf{x}_t$.

4.3 Evaluation

The proposed CNNOR framework is evaluated on a historical color image dataset [Palermo et al., 2012] and an image retrieval dataset MSRA-MM1.0 [Wang et al., 2009]. Two metrics are used as performance indexes. The first one is accuracy defined by $acc = \frac{1}{|T|} \sum_{\mathbf{x}_t \in T} [\hat{y}_t = y_t]$, where T is a testing set, $|T|$ is its size, y_t is the ground truth of $\mathbf{x}_t$, $\hat{y}_t$ is the predicted label for $\mathbf{x}_t$ and $[\cdot]$ is the indicator function. The second one is mean absolute error (MAE) defined by $e = \frac{1}{|T|} \sum_{\mathbf{x}_t \in T} |\hat{y}_t - y_t|$.

4.3.1 Results on the Historical Color Images Dataset

The historical color image dataset [Palermo et al., 2012] is a benchmark dataset for algorithm evaluation, which includes historical color images photographed in different decades. As shown in Figure 4.3, the dataset consists of five ordinal categories corresponding to five decades from 1930s to 1970s. Each category has 265 color images downloaded from Flickr and non-photographic content was removed manually. For fair comparison, the same experimental setting as [Palermo et al., 2012] is taken in this study. In each category, 215 images are selected for training and the rest 50 images are for testing. The same training and testing image partitions published by Palermo et al. (<http://graphics.cs.cmu.edu/projects/historicalColor/>) are used.

Two categories of baselines are used for comparison: hand-crafted feature based methods and deep learning methods. For each category, state-of-the-art multi-class classification methods and ordinal regression methods are evaluated as shown in Table 4.2. Palermo et al. [Palermo et al., 2012] designed six categories of features for this task: color co-occurrence histogram (3072 features), conditional probability of saturation given hue

(a) 1930s

(b) 1940s

(c) 1950s

(d) 1960s

(e) 1970s

Figure 4.3: Historical color image dating dataset.

(512 features), hue histogram (128 features), gist descriptor (600 features), tiny image (3072 features), and $L^*a^*b^*$ color histogram (784 features), which are 8168 dimensions in total. For fair comparison, all hand-crafted feature based methods in Table 4.2 are tested on these same features. Palermo et al. [Palermo et al., 2012] focused on feature design for this specific task, and based on that they tackle the task as a multi-class classification problem by using a linear multi-class SVM as the classifier. Martin et al. [Martin et al., 2014b] improved the ordinal regression method for this particular task and got better MAE result. RED-SVM [Lin and Li, 2012] is a state-of-the-art general ordinal regression method but performs worse than Palermo et al. [Palermo et al., 2012] and Martin et al. [Martin et al., 2014b] on this dataset.

The deep multi-class classification method (CNNm in Table 4.2) and the deep ordinal regression method (Niu et al.’s method in Table 4.2) based on CNN are implemented for comparison. The layers, parameters and organizations of the networks in CNNOR (i.e., G_1 - G_5 , G_i , G_j and G_k in Figure 4.2) are implemented exactly same as that in CNNm and

Table 4.2: Baseline methods and experimental results.

Features	Methods		Accuracy (%)	MAE
Hand-crafted	Classification	Palermo et al.’s method [Palermo et al., 2012]	44.92	0.93
	Ordinal regression	Martin et al.’s method [Martin et al., 2014b]	42.76	0.87
		RED-SVM [Lin and Li, 2012]	35.92	0.96
Deep learning	Classification	CNNm	41.07	1.06
	Ordinal regression	Niu et al.’s method [Niu et al., 2016]	38.65	0.95
		CNNOR	41.56	1.04

Niu et al.’s method, except for the number of output neurons and the loss layer. AlexNet architecture [Krizhevsky et al., 2012] is employed in all the comparison methods. The image size of the historical dataset is equal or greater than 315×315 pixels, and we crop the images to 227×227 pixels for inputting to AlexNet architecture. The reasons why we apply AlexNet for this dataset are two-fold. On the one hand, the size of images is suitable for AlexNet. On the other hand, CNNOR has m separate DNNs for a m -rank ordinal regression problem. AlexNet is a mediate deep network and thus it is efficient for training on this dataset. If a much deeper architecture such as VGG-16 is applied, much more weights need to be trained. For each training/testing image partition, the last 5 images in the training set are used as the validation images, i.e., 210 images for training, 5 images for validation, and 50 images for testing in each rank. Thus, the total sizes of training, validation and testing sets for CNNm and Niu et al.’s method are 1050, 25 and 250 images. For CNN_{reg} of CNNOR, all the possible permutations of the images in the

five ranks produce 210^5 training instances (i.e., the list $(\mathbf{x}_i^1, \mathbf{x}_i^2, \dots, \mathbf{x}_i^k, \dots, \mathbf{x}_i^m)$ in Eq. 4.1).

In the experiments, 40960 instances are randomly selected from them to train CNN_{reg} and 40960 training triplets and 2100 validation triplets with equal numbers of positive and negative triplets are randomly selected to train CNN_k . In the testing phase, 30 triplets for each CNN_k are used to infer the label. The mini-batch size is set to 64 in all the experiments. The learning rate for CNN_m , Niu et al.’s method and CNN_{reg} are 0.01, and the learning rate for fine-tuning CNN_k from CNN_{reg} is 0.001. In the training phase, it is observed that the accuracies on validation sets of both CNN_m and Niu et al.’s method are fluctuated dramatically because the validation sets are too small to reliably estimate the performance on the testing set. However, for CNNOR , the size of validation set is 33 mini-batches, and therefore, we can use the early stopping strategy to stop training when accuracy converges on the validation set. This is a benefit from the proposed method for small datasets for training and validation. In the experiments, the numbers of training iterations for CNN_m and Niu et al.’s method need to be predefined. Table 4.3 and Table 4.4 show the test accuracies of CNN_m and Niu et al.’s method on different numbers of iterations. After 7500 iterations for CNN_m and 12500 iterations for Niu et al.’s method the losses on both training sets are smaller than 0.01. In Table 4.2, we choose the best accuracy from Table 4.3 and Table 4.4 for comparison. For CNNOR , the number of iterations to train CNN_{reg} is 2500, and to train CNN_k is 2111, which is the average value on all 20 training/testing partitions. It shows that CNNOR outperforms the other two deep learning methods in terms of accuracy. Though Niu et al.’s method performs better than CNNOR in terms of MAE, its accuracy is significantly lower. It should be emphasized that the CNN_m and Niu et al.’s results in Table 2 are the best results selected from Tables 3 and 4, where predefined iterations are used because the small validation sets cannot reliably estimate the testing accuracy. Comparing with the methods based on the hand-crafted features, CNNOR performs 5.64% better than RED-SVM in terms of accuracy

and its performance is also comparable with the other two methods. Note that RED-SVM is an ordinal regression method for general ordinal regression problems but Palermo et al. and Martin et al.’s methods, which are highly based on prior knowledge to design the classifiers and the features, are tailored-made for this dataset.

Table 4.3: Accuracy performance of CNNm

#Iterations	2500	5000	7500	10000	12500	15000
Accuracy(%)	38.77	39.26	40.33	41.07	40.22	41.07

Table 4.4: Accuracy performance of Niu et al.’s method

#Iterations	7500	10000	12500	15000	17500	20000
Accuracy(%)	34.17	36.41	37.29	38.30	38.59	38.65

4.3.2 Results on the Image Retrieval Dataset

Microsoft Research Asia Multimedia 1.0 (MSRA-MM 1.0) dataset [Wang et al., 2009] is a benchmark dataset to evaluate multimedia information retrieval algorithms and includes an image subset and a video subset. In the image dataset, 68 representative queries are selected based on the query log of Microsoft Live Search and then about 1000 images for each query are collected from the image search engine of Microsoft Live Search. The relevance annotations of the images are provided. For each image, its relevance to the corresponding query is labeled with three levels: very relevant, relevant and irrelevant. These three levels are indicated by rank 2, 1 and 0, respectively. Figure 4.4 shows an example of the “cat” query, where the first row lists images labeled as “very relevant”, the second row shows some images labeled as “relevant”, and the last row is “irrelevant”

Figure 4.4: MSRA-MM1.0 dataset: cat subset.

images. Given a testing image in a query set, predicting its relevance to the query is an ordinal regression problem. Hence, five subsets of MSRA-MM 1.0 image dataset which are “cat”, “baby”, “beach”, “rose” and “tiger” are used to evaluate the performance of algorithms. The number of images in each rank of the five subsets is shown in Table 4.5. The total number of images in each subset is less than 1100 which is also a small dataset. The experiments on MSRA-MM 1.0 evaluate the CNNOR on data with different properties, including the different number of ranks, non-equal number of images in each rank and smaller image size.

The images in MSRA-MM 1.0 are thumbnails (i.e., the small images displayed on Microsoft Live Search) which are cropped to 3-channel 60×60 pixels in the experiments. The LeNet architecture [LeCun et al., 1998] is employed in all the comparison methods: CNNm, Niu et al.’s method and CNNOR. The setting of mini-batch and learning rate are same as those in Section 4.3.1. Tables 4.6 and 4.7 summarize the results, which are the mean values on three random training/testing partitions. For accuracy, CNNOR performs 5.16% higher than CNN and 5.20% than Niu et al.’s methods averagely on the five subsets.

Table 4.5: Class distribution on MSRA-MM1.0 dataset

#Images	Rank 1	Rank 2	Rank 3	Total
Baby	379	295	277	951
Beach	336	398	213	947
Cat	243	344	378	965
Rose	222	418	329	969
Tiger	277	408	335	1020

Table 4.6: Accuracy (%) result on MSRA-MM1.0 dataset.

	Baby	Beach	Cat	Rose	Tiger
CNNm	48.00	50.67	47.56	55.11	53.33
Niu et al.	47.33	51.11	48.44	55.78	51.78
CNNOR	51.56	56.45	52.67	59.78	60.00

For MAE, CNNOR achieves better results on three of the five subsets. Because there are not handcrafted features for MSRA-MM 1.0 dataset published in literature, to evaluate non-deep methods, the best baseline method in Table 4.2 using the 8868 features proposed for the historical dataset is tested. The accuracy on “cat” subset is 37.11%, which is 15.56% lower than CNNOR. The results indicate that CNNOR performs better than the two deep learning based multi-class classification method and ordinal regression method.

4.4 Summary

In this work, a new ordinal regression algorithm is proposed for small data problems. CNNs are adapted to automatically extract high-level features to describe the ordinal relationship. To increase training data, this chapter proposes a new network organization

Table 4.7: MAE result on MSRA-MM1.0 dataset.

	Baby	Beach	Cat	Rose	Tiger
CNNm	0.667	0.598	0.676	0.522	0.571
Niu et al.	0.647	0.576	0.620	0.500	0.562
CNNOR	0.640	0.551	0.627	0.513	0.523

with triplets as instances and employs a new objective to pre-train the networks. Thus, deep learning can be applied more effectively on small dataset problems. The experimental results show that the proposed algorithm is comparable with the state-of-the-art methods.

Chapter 5

A Constrained Deep Neural Network for Ordinal Regression

5.1 Overview

In previous two chapters, pairs and triplets of instances are explored to represent data relationship for ordinal regression. However, a decoder is needed in both approaches to recover the rank prediction from the outputs of ordinal classifiers, which increases the instability of algorithms. The proposed approach in this chapter aims to embed the data relationship into constraints of optimizing formulation, and obtain predictions for instances in an end-to-end process without any preprocessing such as feature extraction or any postprocessing such as decoding.

As the problem setting of ordinal regression lies between multi-class classification and metric regression, most approaches solve the ordinal regression problem either from regression prospective or from classification prospective. The approaches from regression prospective aim to learn a function mapping the instances to a real line and predict multiple boundaries to discretize the mapped value. For example, the max-margin based approaches [Shashua and Levin, 2002][Chu and Keerthi, 2007] adapted the support vector regression to predict contiguous boundaries splitting the ordinal classes. The ap-

proaches from classification prospective embed ordinal information between class labels into the traditional classification methods. For example, neural network based approaches [Gutiérrez et al., 2014][Cheng et al., 2008] use different coding schemes to encode the ordinal information of class labels into the output vectors of the networks. However, few of existing work combine classification and regression parts in the optimization objective explicitly. This chapter proposes a constrained optimization formulation for the ordinal regression problem which minimizes the negative loglikelihood for multiple categories constrained by the order relationship between instances. The proposed constrained optimization problem can be converted to an unconstrained problem with two terms in the objective function: one is a logistic regression loss for classification and the other is a pairwise hinge loss for regression. Therefore, the proposed approach optimizes classification and regression objectives simultaneously, which targets to the problem setting of ordinal regression more directly.

As discussed in Chapter 4, most of existing ordinal regression approaches in literature are based on handcrafted features, which are labor-intensive and highly rely on the prior knowledge. The first deep ordinal regression approach [Niu et al., 2016] is proposed to adapt CNN for age estimation. They proposed to apply a single CNN with $m - 1$ outputs for m -rank ordinal regression problems. However, a postprocessing step is required to decode the final predicted rank for a testing instance. Therefore, it is not an end-to-end approach.

As ordinal regression is aiming to classify instances to ordinal categories, this work targets to automatically extract high-level features for representing intraclass information and interclass ordinal relationship simultaneously by using revised CNNs. In the traditional deep learning approaches, the learning objectives are formulated as unconstrained optimization problems. Usually, the objective function is a loss designed for the task. This work first formulates the ordinal regression problem as a constrained optimization problem. A CNN can be viewed as a combination of multiple convolution layers which map

instances to a high dimensional feature space and multiple fully-connected layers which perform as a classifier. The proposed method aims to learn the mapping function to maximize the probability of training instances belong to their category under constraints that in the high dimensional space, the instances from ordered ranks are mapped to a real line in order. Then, an equivalent unconstrained formulation with a pairwise regularizer is derived. Based on this formulation, CNN is adapted to solve the problem such that high-level features can be extracted automatically, and the optimal solution can be learned through the traditional back-propagation method. The proposed pairwise constraints as data augmentation make the algorithm work even on small datasets, and a proposed efficient implementation make it be scalable for large datasets.

In terms of ranking order, a related research topic is learning to rank for information retrieval. Its target is to learn the relevance between a document and a given query, and to predict the relative order of the documents based on the relevance. However, learning to rank is different from ordinal regression because it is not able to predict the exact ranks of documents. A comprehensive survey [Liu et al., 2009] summarizes the approaches of learning to rank as pointwise, pairwise and listwise approaches. RankingSVM [Joachims, 2002], an pairwise approach, introduced the ranking constraints into SVM as shown in Eq. 5.1:

$$\begin{aligned}
& \min_{\mathbf{w}, \xi} \quad \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i,j} \xi_{i,j} \\
& \text{s.t.} \quad \mathbf{w}^\top \phi(q, d_i) \geq \mathbf{w}^\top \phi(q, d_j) + 1 - \xi_{i,j} \\
& \quad \quad \xi_{i,j} \geq 0
\end{aligned} \tag{5.1}$$

where q is a query, d_i and d_j are two documents, $\mathbf{w}$ is the weight vector and $\phi(q, d_i)$ is a mapping function. This work adapts the pairwise constraints in Eq. 5.1 for ordinal regression and solves the proposed optimization problem under the deep learning framework.

The contributions of this work are summarized as following:

1. The proposed approach adapts DNNs to solve a constrained optimization problem

for ordinal regression.

2. The proposed approach is an end-to-end approach without any preprocessing such as feature extraction or any postprocessing such as decoding for predictions.
3. The proposed pairwise regularizer makes deep learning on small datasets possible.
4. The proposed approach is suitable for small datasets and scalable for large datasets.

5.2 The Proposed Algorithm

An ordinal regression problem with m ranks denoted by $Y = \{1, 2, \dots, m\}$ is considered, where the natural order of the numbers in Y indicates the order of the ranks. A training set with labeled instances $T = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in X, y_i \in Y\}$ is given, where X is the input space. The target is to predict the rank $y_t \in Y$ of an input $\mathbf{x}_t \in X$. Let $X_k \subseteq X$ be the subset of training instances whose rank labels are k and $I_k = \{i | \mathbf{x}_i \in X_k\}$ be the index set of X_k . Denote $\mathbf{x}_i^k \in X_k$ as an input from rank k . In the rest of this section, the outline of the proposed approach will be provided first, and then the DNN architecture used to solve the optimization problem will be presented.

5.2.1 The Proposed Optimization Formulation

The intuition of the proposed approach is to learn a multi-class classifier by constraining the instances being mapped to a real line in order. Eq. 5.2 shows the optimization problem:

$$\begin{aligned}
\min_{f, \phi, w, \xi} \quad & - \sum_{k=1}^m \sum_{i \in I_k} \log \frac{e^{f_k \circ \phi(\mathbf{x}_i^k)}}{\sum_{r=1}^m e^{f_r \circ \phi(\mathbf{x}_i^k)}} + C \sum_{k=1}^{m-1} \sum_{\substack{i \in I_k \\ j \in I_{k+1}}} \xi_{i,j}^k \\
\text{s.t.} \quad & \mathbf{w}^\top \phi(\mathbf{x}_j^{k+1}) - \mathbf{w}^\top \phi(\mathbf{x}_i^k) \geq 1 - \xi_{i,j}^k, \\
& \xi_{i,j}^k \geq 0, \quad k = 1 \dots m-1, i \in I_k, j \in I_{k+1}
\end{aligned} \tag{5.2}$$

where m is the number of ranks, and I_k is the index set of X_k . $f_k(\cdot)$ and $\phi(\cdot)$ are mapping functions, and $\circ$ is the function composition operator. $\mathbf{w}$ is the weight vector mapping $\phi(\mathbf{x})$ to a real line. $\phi(\cdot)$ can be considered as a feature extractor and $f_k(\cdot)$ is a classifier for label k . The first term of the objective function in Eq. 5.2 is the composition of softmax function and multinomial logistic regression loss, and the second term is the sum of slack variables $\xi_{i,j}^k$ where C is a hyperparameter. The constraints in Eq. 5.2 define the condition that the mapped values of instances from rank $k + 1$ should be equal or larger than those of instances from rank k with an margin of 1 and tolerance $\xi_{i,j}^k$. Once the optimal solution of Eq. 5.2 is obtained, the rank label of a test instance $\mathbf{x}_t$ is predicted as the category with the maximum likelihood. More precisely, Eq. 5.3 is the decision function:

$$\hat{y}_t = \operatorname{argmax}_k \frac{e^{f_k \circ \phi(\mathbf{x}_t)}}{\sum_{j=1}^m e^{f_j \circ \phi(\mathbf{x}_t)}} = \operatorname{argmax}_k f_k \circ \phi(\mathbf{x}_t) \quad (5.3)$$

The constraints in the proposed approach enforce that all pairs of instances from adjacent ranks are mapped in order with a tolerance, and they are similar to those in RankingSVM [Joachims, 2002] as shown in Eq. 5.1. However, the proposed optimization problem is different from RankingSVM in the following four perspectives:

1. Given a query, the target of RankingSVM is to predict the order of test instances based on relevance. It is not able to predict the exact rank of a test instance.
2. The objective of RankingSVM is to minimize the margin (i.e., $\|\mathbf{w}\|_2^2$) based on the large-margin theory in support vector regression. However, the objective of the proposed approach is to maximize the loglikelihood which is always used for classification problems.
3. The constraints of RankingSVM are applied to all possible pairs for a given query, but the proposed constraints applied to pairs of instances from adjacent ranks.
4. The mapping function $\phi(\cdot)$ in RankingSVM is predefined by a kernel function, but in the proposed approach $\phi(\cdot)$ is learned automatically by a deep neural network.

In the proposed optimization problem, the constraints only count on pairs of instances from adjacent ranks, but other pairs of instances, such as instances from rank k and rank $k + 2$, are not considered explicitly. The reason is that if both $(\mathbf{w}^\top \phi(\mathbf{x}_1^k), \mathbf{w}^\top \phi(\mathbf{x}_2^{k+1}))$ and $(\mathbf{w}^\top \phi(\mathbf{x}_2^{k+1}), \mathbf{w}^\top \phi(\mathbf{x}_3^{k+2}))$ are in order, it can be inferred that $(\mathbf{w}^\top \phi(\mathbf{x}_1^k), \mathbf{w}^\top \phi(\mathbf{x}_3^{k+2}))$ are also in order.

The slack variables in Eq. 5.2 and the slack variables in SVM have the same meaning. They both are used as tolerances for non-separable instances. In Eq. 5.2, if $\mathbf{w}^\top \phi(\mathbf{x}_j^{k+1}) - \mathbf{w}^\top \phi(\mathbf{x}_i^k) \geq 1$, the error $\xi_{i,j}^k$ should be 0. Otherwise, the error $\xi_{i,j}^k$ should be $1 - (\mathbf{w}^\top \phi(\mathbf{x}_j^{k+1}) - \mathbf{w}^\top \phi(\mathbf{x}_i^k))$. Therefore, the proposed optimization problem can be rewritten as an unconstrained optimization problem in Eq. 5.4.

$$\min_{f, \phi, \mathbf{w}} - \sum_{k=1}^m \sum_{i \in I_k} \log \frac{e^{f_k \circ \phi(\mathbf{x}_i^k)}}{\sum_{r=1}^m e^{f_r \circ \phi(\mathbf{x}_i^k)}} + C \sum_{k=1}^{m-1} \sum_{\substack{i \in I_k \\ j \in I_{k+1}}} \max(0, 1 + \mathbf{w}^\top \phi(\mathbf{x}_i^k) - \mathbf{w}^\top \phi(\mathbf{x}_j^{k+1})) \quad (5.4)$$

The first term in Eq. 5.4 is same as the the first term in Eq. 5.2 and the second term can be viewed as a pairwise hinge loss for regression. Therefore, the proposed approach optimizes the weighted combination of classification loss and regression loss explicitly, which directly represents the definition of ordinal regression problem.

5.2.2 The Proposed CNN based Optimization

Traditional feature based large-margin approaches often employ a function $\psi(\mathbf{x}_i)$ mapping the input feature vector $\mathbf{x}_i$ to a high dimensional space. And a predefined kernel is used to represent the mapping function based on the kernel trick. The form of the kernel function and its hyperparameters affect the performance a lot. Deep neural networks are able to learn the high level features and weights of classifiers simultaneously. Therefore, a deep neural network is designed to learn the mapping function $\phi(\cdot)$, the weight $\mathbf{w}$ and $f_k(\cdot)$ in the proposed optimization problem in Eq. 5.4. Since convolutional neural networks are used in the current implementation, the proposed method is named convolu-

tional neural network with pairwise regularization for ordinal regression (CNNPOR).

The new loss function defined in Eq. 5.4 is implemented in CNNPOR, which is a weighted combination of a softmax logistic regression loss and a pairwise hinge loss. It should be pointed out that the scales of the two losses are not same. Therefore, a new training set is constructed by pairing up the instances from adjacent ranks, i.e., $X' = \{(\mathbf{x}_s^k, \mathbf{x}_s^{k+1}) | \mathbf{x}_s^k \in X_k, \mathbf{x}_s^{k+1} \in X_{k+1}, k = 1, \dots, m-1\}$. Define $P_k = \{(\mathbf{x}_s^k, \mathbf{x}_s^{k+1})\}$, and $I_k^p = \{s | (\mathbf{x}_s^k, \mathbf{x}_s^{k+1}) \in P_k\}$ as the index set of P_k . All the elements $\mathbf{x}_s^k$ and $\mathbf{x}_s^{k+1}$ in the pairs are used as input. Using this training set, the two losses are scaled automatically i.e. Eq. 5.5.

$$\begin{aligned} \min_{f, \phi, \mathbf{w}} & - \sum_{k=1}^{m-1} \sum_{s \in I_k^p} \log \frac{e^{f_k \circ \phi(\mathbf{x}_s^k)}}{\sum_{r=1}^m e^{f_r \circ \phi(\mathbf{x}_s^k)}} - \sum_{s \in I_{m-1}^p} \log \frac{e^{f_m \circ \phi(\mathbf{x}_s^m)}}{\sum_{r=1}^m e^{f_r \circ \phi(\mathbf{x}_s^m)}} \\ & + C \sum_{k=1}^{m-1} \sum_{s \in I_k^p} \max(0, 1 + \mathbf{w}^\top \phi(\mathbf{x}_s^k) - \mathbf{w}^\top \phi(\mathbf{x}_s^{k+1})) \end{aligned} \quad (5.5)$$

Figure 5.1 shows the architecture of CNNPOR for a 3-rank ordinal regression problem. The input instances are organized in a list, as $(\mathbf{x}_i^1, \mathbf{x}_i^2, \mathbf{x}_i^3)$ in the figure, where $\mathbf{x}_i^1, \mathbf{x}_i^2, \mathbf{x}_i^3$ are from rank 1, 2 and 3, respectively. They are individually inputted to the convolution net G_h , which represents the mapping function $\phi(\cdot)$ in Eq. 5.5. The outputs of G_h as the high dimensional features are passed to the fully-connected layer G_c , which represents the mapping function $f_k(\cdot)$. There is a softmax logistic regression loss and the number of output neurons equals to the number the ranks. The combination of the convolution net G_h and the fully-connected layer G_c is a standard multi-class CNN. Then the instances from adjacent ranks (i.e., $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$, $\mathbf{x}_i^2$ and $\mathbf{x}_i^3$) are paired up and inputted into the convolution nets G_{11} to G_{22} . The outputs of all G_{11} to G_{22} are mapped into one dimensional space by the mapping vector w , and then the pairwise hinge loss layer receives all the outputs to calculate the last term in Eq. 5.5. The final loss layer sums up the two losses at weights 1:C. All the convolution nets ($G_h, G_{11}-G_{22}$) have the same architecture which consists of layers before the last fully-connected layer in a standard CNN, and they share the same weights. In the training phase, the standard backpropagation technique is

Figure 5.1: The architecture of CNNPOR for a 3-rank ordinal regression problem.

used and the loss is back propagated to all the convolution nets. In the testing phase, a testing point $\mathbf{x}_t$ is inputted into G_h and the output of the G_c is the prediction. Therefore, CNNPOR is different from other pairwise methods such as Niu et al.’s method [Niu et al., 2016], RED-SVM [Lin and Li, 2012] and Liu et al.’s method [Liu et al., 2016], because it is an end-to-end approach for ordinal regression, which does not require any postprocess step to achieve the predictions.

5.2.3 Scalability of the Proposed Algorithm

The proposed pairwise constraints as a regularizer make learning CNNPOR on small datasets possible, while the proposed architecture is also computationally feasible for large datasets. It should be emphasized that, for a training set with n images, the number of input images of CNNPOR is n not n^2 . As shown in Figure 5.1, all the convolution layers G_h , G_{11} , G_{12} , G_{21} and G_{22} share weights, meaning that there is only one unique standard CNN to be trained. The pairwise constraints which require quadratic number of operations are applied on the features inputted to the pairwise loss layer, not on the raw input images.

Algorithm 3 describes the implementation of one training iteration in CNNPOR, which reorganizes the instances as each batch having d images from each rank (i.e., set D^r in Algorithm 3) and n images from all ranks randomly (i.e., set D^c). Assume that a standard CNN structure such as the VGG-16 [Simonyan and Zisserman, 2014] or the LeNet [LeCun et al., 1998] are used. All layers before the last fully-connected layer are named as G_h , which also represents for G_{11} to G_{22} in Figure 5.1, and the last fully-connected layer is named as G_c . In CNNPOR, one more fully-connected layer G_r with one output node is connected to G_h , and its weights are the w in Figure 5.1. As shown in line 1-2 of Algorithm 3, all instances of D are propagated to G_h . Then the instances of D^c are propagated to G_c to calculate the softmax loss l_1 and the instances of D^r are propagated to G_r to calculate the pairwise hinge loss l_2 in line 6-12. Finally, the weighted loss $l_1 + C \times l_2$

Algorithm 3 Pseudo code of one training iteration in CNNPOR

Input: Training set $D = D^c \cup D^r$ with n instances in D^c and $m \times d$ instances in D^r ,

where $D^r = D_1 \cup D_2 \cdots \cup D_m$, $D_k \subseteq X_k$ and the size of D_k is d .

Output: Update the network weights.

- 1: Initialize or update all weights in a CNN consisting of convolution net G_h and two fully-connected layers G_c and G_r both connected to G_h .
 - 2: Forward propagate all instances of D into G_h .
 - 3: Forward propagate instances of D^c into G_c .
 - 4: Calculate the softmax loss l_1 of D^c .
 - 5: Forward propagate instances of D^r into G_r .
 - 6: **procedure** PAIRWISEHINGELOSS
 - 7: Initialize pairwise hinge loss $l_2 \leftarrow 0$.
 - 8: $O_k \leftarrow$ the outputs of G_r for D_k .
 - 9: **for** $k = 1$ to $m - 1$ **do**
 - 10: $l_2 = l_2 + SUM(MAX(0, 1 + O_k - O_{k+1}))$
 - 11: **end for**
 - 12: **end procedure**
 - 13: Backward propagate of $l_1 + C \times l_2$.
-

is back propagated to the whole network. O_k in line 9 is a vector where each element is the one-dimensional output of G_r for one instance of D_k , i.e., $w \cdot \phi(\mathbf{x}_s^k)$ in Eq. 5.5. The operations ‘ $-$ ’, MAX and SUM in line 10 are elementwise subtraction, maximum and summation. Therefore, comparing to a standard m -class CNN, for a mini-batch with $n + m \times d$ instances, CNNPOR does not calculate the softmax loss for $m \times d$ instances but calculates the hinge loss for them by using $m - 1$ element-wise vector subtraction, maximum and summation operations instead. Thus, although CNNPOR introduces the pairwise regularizer, by employing the proposed architecture and implementation, it is scalable for large scale datasets.

5.3 Evaluation

The proposed CNNPOR approach is evaluated on four benchmarks - a historical color image dataset [Palermo et al., 2012], an image retrieval dataset MSRA-MM1.0 [Wang et al., 2009], an image aesthetic dataset [Schifanella et al., 2015] and the Adience Face Dataset [Levi and Hassner, 2015]. Accuracy and mean absolute error are used as performance indexes. Accuracy is defined by $\frac{1}{|T|} \sum_{\mathbf{x}_t \in T} [\hat{y}_t = y_t]$, where T is a testing set and $|T|$ is its size, $[\cdot]$ is the indicator function, y_t is the ground truth of $\mathbf{x}_t$, and $\hat{y}_t$ is its predicted label. Mean absolute error (MAE) is defined by $\frac{1}{|T|} \sum_{\mathbf{x}_t \in T} |\hat{y}_t - y_t|$. Three baseline methods are employed for comparison: the state-of-the-art handcrafted feature based ordinal regression method - RED-SVM [Lin and Li, 2012], the traditional CNN method for multi-class classification - CNNm and the CNN based ordinal regression method - Niu et al.’s method [Niu et al., 2016].

5.3.1 Results on the Historical Color Images Dataset

The historical color image dataset [Palermo et al., 2012] is used to evaluate the proposed approach. The benchmark is described in Section 4.3.1. The evaluation protocol reported

Table 5.1: Results on the historical image benchmark.

Methods	Accuracy(%)	MAE
Palermo et al.’s method [Palermo et al., 2012]	44.92±3.69	0.93±0.08
Martin et al.’s method [Martin et al., 2014b]	42.76±1.33	0.87±0.05
Frank and Hall [Frank and Hall, 2001]	41.36±1.89	0.99±0.05
Cardoso and Pinto da Costa [Cardoso and Costa, 2007]	41.32±2.76	0.95±0.04
RED-SVM@8168 [Lin and Li, 2012]	35.92±4.69	0.96±0.06
RED-SVM@deep [Lin and Li, 2012]	25.38±2.34	1.08±0.05
CNNm	48.94±2.54	0.89±0.06
Niu et al.’s method [Niu et al., 2016]	44.67±4.24	0.81±0.06
CNNPOR	50.12±2.65	0.82±0.05

in [Palermo et al., 2012] is taken in this study for fair comparison. In each category, 215 images are employed for training and the rest 50 images are for testing.

Table 5.1 lists the experimental results on the historical color image dataset. Besides the results of the three baseline methods, i.e, RED-SVM, CNNm and Niu et al.’s method, the results from the previous methods on this dataset are also reported. Palermo et al.’s method [Palermo et al., 2012] and Martin et al.’s method [Martin et al., 2014b] are proposed for this particular task, and Frank and Hall’s method [Frank and Hall, 2001] and Cardoso and Pinto da Costa’s method [Cardoso and Costa, 2007] are for general ordinal regression problems. Palermo et al. [Palermo et al., 2012] designed 8168 features for this task. In the experiments, all handcrafted feature based methods listed in Table 5.1 use the same features for fair comparison. RED-SVM [Lin and Li, 2012] is a state-of-the-art handcrafted feature based method for general ordinal regression problems. To evaluate the performance of CNNPOR achieved by the deep features and by the algorithm, CNNPOR is compared with RED-SVM with the inputs of the 8168 handcrafted features (RED-SVM@8168 in Table 5.1) and the deep features extracted from the tradi-

tional CNN which are the 512 dimensional output values before the first fully-connected layer in the VGG-16 architecture [Simonyan and Zisserman, 2014] (RED-SVM@deep in Table 5.1).

The deep multi-class classification method (CNNm in Table 5.1) and the deep ordinal regression method (Niu et al.’s method in Table 5.1) are implemented for comparison. For the historical image dataset, the VGG-16 architecture is employed for CNNm, Niu et al.’s method and CNNPOR. For CNNPOR, as shown in Figure 5.1, G_h and G_c are linked together and implemented through the VGG-16 architecture, i.e., G_h consists of the thirteen convolution layers and the ReLU and pooling layers in between, and G_c includes the three fully-connected layers and the layers in between. The implementation of $G_{11} - G_{22}$ is same as G_h . The reasons why we choose VGG-16 are that the size of images is suitable for VGG-16 and there are pre-trained VGG-16 models to be finetuned. And it was claimed that VGG-16 with a deeper architecture perform better than AlexNet. The images in the historical image dataset are resized to 256×256 pixels. For all the three deep learning methods, the image size of the input layer is set to 224×224 3-channel pixels, and the input images are cropped further at random positions during the training phases for data augmentation. For each training/testing image partition, the last 5 images in the training set are used as the validation images, i.e., 210, 5 and 50 images respectively for training, validation and testing in each rank. In total, the sizes of training, validation and testing sets for CNNm and Niu et al.’s method are 1050, 25 and 250 images, respectively. For CNNPOR, all the possible permutations of the images in the five ranks produce $4 * 210^2$ training pairs (i.e., the pair $(\mathbf{x}_i^k, \mathbf{x}_j^{k+1})$ in Eq. 5.5) and $4 * 5^2$ validation pairs. All three deep methods are fine-tuned from the pretrained ImageNet model. The C in Eq. 5.5 is set to 1 in the experiments. The learning rate of all layers, except for the last fully-connected layer, is set to 0.0001. Because the number of output nodes for the historical image dataset is different from that for the ImageNet, the learning rate of the last fully-connected layer is set as 10 times of the learning rate of other layers, i.e., 0.001.

Table 5.1 summarizes the results and the number after $\pm$ is the standard deviation values. CNNPOR outperforms RED-SVM on handcrafted features and deep features, CNNm, and Niu et al.’s method by 14.2%, 24.74%, 1.18%, and 5.45%, respectively in terms of accuracy. The mean MAE result of CNNPOR is 0.01 higher than that of Niu et al.’s method, which outperforms all other methods, but it is within two standard deviations of Niu et al.’s method. Overall, CNNPOR achieves the best results on the historical color image dataset. As shown in Table 5.1, CNNm performs much better than RED-SVM on deep features (RED-SVM@deep). The deep features for training RED-SVM are extracted from the well-trained CNNm. The results show RED-SVM cannot fully utilize the deep network, because during the training phase of RED-SVM, it cannot adjust the mapping from the raw images to deep features. As shown in Table 5.1, Niu et al.’s method achieves better performance for MAE than for accuracy. It is originally proposed for age estimation problem, which has larger number of ranks and is more similar to regression. Thus, it focuses on minimizing the absolute error, instead of zero one error.

5.3.2 Results on the Image Retrieval Dataset

The other small scale benchmark Microsoft Research Asia Multimedia 1.0 (MSRA-MM 1.0) dataset [Wang et al., 2009] which has been illustrated in the previous work is used in the experiments for CNNPOR. Seven subsets - “cat”, “baby”, “beach”, “rose”, “tiger”, “fish” and “golf” in MSRA-MM 1.0 image benchmark are used to evaluate the performance CNNPOR. Table 5.2 summarizes the size of the seven subsets and the number of images in each rank. These datasets are small with less than 1100 images. To evaluate the algorithms on imbalanced datasets, “fish” and “golf” subsets are tested, respectively, 69.4% and 81.5% images in one rank. Besides images content and task differences, the images in MSRA-MM 1.0 are different from the historical images in three properties: different number of ranks, non-equal number of images in each rank or very imbalanced, and smaller image size.

Table 5.2: Class distributions on MSRA-MM1.0 dataset.

	Rank 1	Rank 2	Rank 3	Total
Baby	379	295	277	951
Beach	336	398	213	947
Cat	243	344	378	965
Rose	222	418	329	969
Tiger	277	408	335	1020
Fish	130	669	165	964
Golf	777	97	79	953

Because the size of MSRA-MM 1.0 images is quite small, the LeNet architecture [Lecun et al., 1998] is employed in all deep learning methods: CNNm, Niu et al.’s method and CNNPOR. The images are cropped to 60×60 pixels in the experiments. For each rank of the first five datasets in Table 5.2, the images are randomly split to 10 images for validation, 50 images for testing and the rest for training. For the two imbalanced datasets “fish” and “golf”, 75%, 5% and 20% images in each rank are randomly selected for training, validation and testing, respectively. In each training set, 40960 pairs of instances from adjacent ranks are constructed as training instances for CNNPOR. Mini-batch size is set to 64 and the learning rate is set to 0.01. To evaluate RED-SVM method on handcrafted features, the same 8168 features as used for the historical image dataset are employed. RED-SVM is also tested on the features extracted before the first fully-connected layer of the LeNet architecture, which is 50 dimensional features. All methods are examined on three random training/testing partitions for all datasets and the mean results are summarized in Tables 5.3 5.4. CNNPOR performs better than all the baseline methods on five subsets in terms of accuracy, and on three subsets in terms of MAE. The results on MSRA-MM 1.0 dataset indicate that CNNPOR performs averagely better than the baseline methods.

Table 5.3: Accuracy (%) results on MSRA-MM1.0 dataset.

	RED-SVM@8168	RED-SVM@deep	CNNm	Niu et al.	CNNPOR
Baby	36.99	32.66	48.00	47.33	50.00
Beach	35.64	34.00	50.67	51.11	51.11
Cat	40.22	34.89	47.56	48.44	52.89
Rose	42.05	34.22	55.11	55.78	56.67
Tiger	35.57	33.56	53.33	51.78	52.89
Fish	68.66	68.89	63.95	66.16	66.33
Golf	80.45	80.17	83.08	83.93	84.96
Overall	48.51	45.48	57.39	57.79	59.26

Table 5.4: MAE results on MSRA-MM1.0 dataset.

	RED-SVM@8168	RED-SVM@deep	CNNm	Niu et al.	CNNPOR
Baby	0.630	0.699	0.667	0.647	0.636
Beach	0.648	0.673	0.598	0.576	0.596
Cat	0.633	0.662	0.676	0.620	0.598
Rose	0.582	0.664	0.522	0.500	0.500
Tiger	0.644	0.673	0.571	0.562	0.578
Fish	0.313	0.311	0.378	0.357	0.355
Golf	0.283	0.289	0.229	0.219	0.197
Overall	0.533	0.567	0.520	0.497	0.494

5.3.3 Results on the Image Aesthetics Dataset

The image aesthetic benchmark [Schifanella et al., 2015] consists of 10800 Flickr photos of four categories, i.e., “animals”, “urban”, “people” and “nature”, and is constructed originally to retrieve beautiful yet unpopular images in social networks. The ground truths of the photos in the benchmark are five aesthetic grades: “Unacceptable” - images with extremely low quality, out of focus or underexposed, “Flawed” - images with some technical flaws and without any artistic value, “Ordinary” - standard quality images without technical flaws, “Professional” - professional-quality images with some artistic value, and “Exceptional” - very appealing images showing both outstanding professional quality and high artistic value. Figure 5.2 shows an example from the “urban” category with one photo from each aesthetic level. Each photo in the dataset is labeled by five graders of an online crowdsourcing platform to one of the five aesthetics levels. If the level of agreement is low, two more graders are recruited to perform the evaluation. In the experiments, these five aesthetic levels are indicated by rank 1 to 5, and the median rank of each image given by the graders is used as the ground truth. In each rank 75%, 5% and 20% images are randomly selected for training, validation and testing, respectively. All comparison methods are tested on five random training/testing partitions.

In the experiments, all the deep learning methods, including CNNm, Niu et al.’s method and CNNPOR, employ the VGG-16 architecture and are fine-tuned from the ImageNet model. The images are resized to 256×256 pixels and are randomly cropped to 224×224 pixels further during the learning. The learning rate is set to 0.001 for the last fully-connect layer and 0.0001 for all other layers. RED-SVM is tested on the same 8168 features listed in Section 5.3.1 and the deep features extracted right before the first fully-connected layer. Table 5.5 and 5.6 summarize the results in terms of accuracy and MAE. For both performance indexes, CNNPOR outperforms all the baseline methods on three categories. CNNm achieves the best performance for one category in terms of accuracy and Niu et al.’s method achieves the best performance for one category in terms of MAE.

Table 5.5: Accuracy (%) results on the image aesthetics dataset.

	RED-SVM@8168	RED-SVM@deep	CNNm et al.	Niu	CNNPOR
Nature	69.73	70.72	70.97	69.81	71.86
Animal	61.14	61.05	68.02	69.10	69.32
Urban	63.88	65.44	68.19	66.49	69.09
People	60.06	61.16	71.63	70.44	69.94
Overall	63.70	64.59	69.45	68.96	70.05

Table 5.6: MAE results on the image aesthetics dataset.

	RED-SVM@8168	RED-SVM@deep	CNNm	Niu et al.	CNNPOR
Nature	0.319	0.309	0.305	0.313	0.294
Animal	0.407	0.410	0.342	0.331	0.322
Urban	0.391	0.374	0.356	0.349	0.325
People	0.421	0.412	0.315	0.312	0.321
Overall	0.385	0.376	0.330	0.326	0.316

(a) Unacceptable

(b) Flawed

(c) Ordinary

(d) Professional

(e) Exceptional

Figure 5.2: Image Aesthetics dataset

5.3.4 Results on the Adience Face Dataset

To evaluate the scalability of CNNPOR, the Adience face dataset [Levi and Hassner, 2015] is employed, which consists of 26580 Flickr photos of 2284 subjects and the ordinal ranks are eight age groups. In the experiments, the images alignment and five-fold partition follow [Levi and Hassner, 2015]. Because the VGG-16 net for multi-class classification has been verified scalable for large datasets, the training phase of CNNPOR is compared with CNNm and both methods are fine-tuned from the VGG-16 ImageNet pretrained model. Same mini-batch size 96 is used for both CNNm and CNNPOR (i.e., $n = 32, m = 8, d = 8$ in Algorithm 3). Same learning rate 0.001 is applied for the last fully-connected layer of CNNm and G_c, G_r of CNNPOR, and 0.0001 for the rest layers. The C in Eq.5.5 is set to 1. Caffe package on Tesla M40 GPU is run for the experiments, and the average training time for one iteration of CNNm and CNNPOR is 3.3 and 3.6 seconds respectively. Figure 5.3 shows the training curves on one fold, which indicate the converge speed of CNNm and CNNPOR are similar, and in the experiments, both methods are trained for the same number of iterations 2000. Therefore, by employing the proposed efficient implementation, the scalability of CNNPOR is similar as CNNm. As shown in Figure 5.3 and Table 5.7, the training error of CNNPOR is higher than CNNm, but CNNPOR achieves better performance on the testing set, which indicates the proposed method avoids overfitting effectively. RED-SVM is not scalable for this dataset, and the accuracy of state-of-the-art handcrafted feature-based method for this dataset is cited from [Eidinger et al., 2014] for comparison in Table 5.7. G. Levi and T. Hassner proposed a lean DNN [Levi and Hassner, 2015] particularly for this dataset. They did not report MAE results in their papers. It is observed that CNNPOR achieves overall best performance consistently for all the benchmarks.

Figure 5.3: Training curves of CNNPOR on the Adience face dataset.

Table 5.7: Results on the Adience face dataset.

Methods	Accuracy(%)	MAE
Feature-based		
[Eidinger et al., 2014]	45.1 ± 2.6	-
Lean DNN [Levi and Hassner, 2015]	50.7 ± 5.1	-
CNNm	54.0 ± 6.3	0.61 ± 0.08
Niu et al.	56.7 ± 6.0	0.54 ± 0.08
CNNPOR	57.4 ± 5.8	0.55 ± 0.08

5.4 Summary

This chapter proposes a new constrained optimization formulation for ordinal regression problems, and transforms it to an unconstrained optimization formulation with an effective deep learning implementation. The experimental results show that CNNPOR achieves overall the best results on all the four benchmarks, demonstrating the generality and scalability of the proposed method.

Chapter 6

A Deep Ordinal Neural Network with Communication between Neurons of Same Layers

6.1 Overview

The proposed deep learning based approaches CNNOR (as discussed in Chapter 4) and CNNPOR (as discussed in Chapter 5) adapted DNNs from the input space and the loss function respectively. For implementation, any deep neural network for multi-class classification can be used for them. In fact, in the experiments, CNNOR and CNNPOR were implemented based on AlexNet and VGG-16. However, this work aims to design an architecture for ordinal regression particularly, where the new connections between neurons reflect the ordinal information.

As summarized in Chapter 2, binary decomposition approaches and threshold approaches are two main categories of ordinal regression methods in literature. However, most of binary decomposition methods require a decoding process, which leads to non end-to-end methods. Threshold methods suffer the limitation that except of regression function $f(\mathbf{x})$, it is necessary to estimate multiple boundaries b_k to divide the $f(\mathbf{x})$ value

to intervals, so that the prediction accuracy for b_k affects the ordinal regression performance a lot. The motivation of the proposed approach in this chapter is to learn $f(\mathbf{x}) - b_k$ by a neural network directly, which is the decision function that the rank of $\mathbf{x}$ is greater than or equal to k , where $k \in [1, m]$ for a m -rank ordinal regression problem.

A standard deep neural network for the m -class classification problem with m output neurons (denoted as CNN m) is used as a base, where each output node refers to the decision function for one rank as mentioned above. The property of ordinal regression problem motivating us to revise the connections between neurons is that if the rank of an instance $\mathbf{x}$ is greater than or equal to k , then it will be definitely greater than or equal to $1, \dots, k - 1$. This is a critical property distinguishing ordinal regression problem from multi-class classification problem, where exchanging the category labels does not change the problem and the performance of any method. Therefore, new edges are added to connect adjacent groups of neurons in the fully-connected layers of CNN m . In other words, it is proposed to connect neurons in the same layers of a neural network, which is a novel and interesting neural network architecture.

6.2 The Proposed Architecture

The basic idea of the proposed approach is based on the following two points. First, the output nodes of the network represent the decision function $o_{y \geq k}(\mathbf{x})$ instead of $o_{y=k}(\mathbf{x})$ as previous chapters, where y is the predicted rank of an instance $\mathbf{x}$ and k is a rank label. Second, decision function $o_{y \geq i}(\mathbf{x})$ and $o_{y \geq j}(\mathbf{x})$ are dependent, where i, j are rank labels. Figure 6.1 illustrates the proposed ordinal architecture based on CNN (ORCNN). L_s denotes a standard CNN, for example, consisting of convolution layers, fully-connected layers, and max pooling and ReLU layer in between. L_s does not include the output layer such as that inputted to softmax in CNN m . Several special fully-connected layers follow L_s , which are named as horizontal layers. Each horizontal layer includes m groups of neurons for a m -rank ordinal regression problem, and each group of neurons makes up a

Figure 6.1: The architecture of ORCNN for a 4-rank ordinal regression problem.

separate layer fully-connected to the upper layer. The groups in the same horizontal layer are fully-connected in a pipeline style. As shown in Figure 6.1, orange dash rectangle marks one horizontal layer L_t for a 4-rank ordinal regression problem, where $G_1, \dots, G_4$ are four groups of neurons for rank 1 to 4 respectively. G_k is fully-connected to G_{k+1} , where $k = 1, \dots, m - 1$. It should be emphasized that the direction of connections within layer L_t is from the group for the lower rank to that for the upper rank. From the vertical view, one group of neurons from each horizontal layer is connected to the next as a standard neural network, as shown in the blue dash rectangle in Figure 6.1, ending up with one output neuron for one rank (i.e., $O_1, \dots, O_4$). It should be pointed out that $O_1, \dots, O_4$ are named as output neurons in ORCNN, but they are not the final neurons of the network. Substraction operation is employed between two adjacent output neurons, i.e., $O_k - O_{k+1}$, as shown by the diamonds in Figure 6.1. The $m - 1$ substraction outcomes and the last output neuron (i.e., O_4) are concatenated and inputted into a softmax layer. The last output neuron is directly inputted into softmax because $O_{y \geq m}(\mathbf{x}) = O_{y=m}(\mathbf{x})$ for a m -rank ordinal regression problem. In the testing phase, the i with maximum value R_i for an instance $\mathbf{x}$ in Figure 6.1 is the predicted rank. Therefore, ORCNN is an end-to-end approach which does not require any postprocessing steps.

Each output neuron in ORCNN represents the decision function $o_{y \geq k}(\mathbf{x})$, which can be viewed as a regression function learned from the network. For example, in Figure 6.1, O_1 is the regression function learned from the subnetwork consisting of layers L_s , G_1 and H_1 . Threshold approaches for ordinal regression in literature formulate $o_{y \geq k}(\mathbf{x}) = f(\mathbf{x}) - b_k$, where $f(\mathbf{x})$ is the regression function to learn and b_k is the boundary to divide $f(\mathbf{x})$ for rank k and $k + 1$. b_k is a hyperparameter tuned during training and affects the performance a lot. While $o_{y \geq k}(\mathbf{x})$ in ORCNN is learned from the network directly. The rank predictions of ORCNN are based on the rule $p(y = k|\mathbf{x}) = p(y \geq k|\mathbf{x}) - p(y \geq k + 1|\mathbf{x})$. ORCNN formulates $p(y = k|\mathbf{x}) = \text{softmax}(o_{y \geq k}(\mathbf{x}) - o_{y \geq k+1}(\mathbf{x}))$. Moreover, the weights connecting the groups of neurons in the horizontal layers quantify

the dependence between $o_{y \geq k}(\mathbf{x})$ and $o_{y \geq k+1}(\mathbf{x})$, $k = 1, \dots, m - 1$. As shown in Figure 6.1, in the backpropagation of training ORCNN, the weight updates raised by instance $\mathbf{x} \in X_k$ will modify the weights for $o_{y \geq 1}(\mathbf{x}), \dots, o_{y \geq k}(\mathbf{x})$, which is based on the fact if $\mathbf{x} \in X_k$, then the rank of $\mathbf{x}$ is greater than or equal to rank $1, \dots, k$.

6.3 Evaluation

The proposed ORCNN architecture is evaluated on four benchmarks - a historical color image dataset [Palermo et al., 2012], an image retrieval dataset MSRA-MM1.0 [Wang et al., 2009], an image aesthetic dataset [Schifanella et al., 2015] and the Adience Face Dataset [Levi and Hassner, 2015], and the experimental settings are same as those in Chapter 5. Also, accuracy and mean absolute error are used as performance indexes. The three baseline methods compared in Chapter 5 - RED-SVM [Lin and Li, 2012], CNNm, Niu et al.’s method [Niu et al., 2016] - are employed, and the ordinal regression approach CNNPOR proposed in the previous chapter is also compared.

6.3.1 Results on the Historical Color Images Dataset

The first benchmark used to evaluate the proposed approach ORCNN is the historical color images dataset [Palermo et al., 2012]. The evaluation protocol is kept same as the one taken in previous chapters for fair comparison. Table 6.1 lists all the baseline methods, and the DNN based methods CNNm, Niu et al.’s method, and CNNPOR are all based on VGG-16 architecture in the experiments for the historical color images dataset. Thus, ORCNN is also realized based on VGG-16 architecture as shown in Figure 6.2. We follow the common abbreviations in Figure 6.2: “conv” is for convolution, “fc” is for fully-connected, and “pool” is for pooling. The number of horizontal layers in ORCNN for the historical images dataset is 1, which is realized as “layer 16” as shown in Figure 6.2. Since there are five categories of images in the dataset, five groups of neurons are

Figure 6.2: The realization of ORCNN based on VGG-16 for a 5-rank dataset.

arranged in “layer 16”, where each group is for one rank. The number of neurons in each group (shown as rectangles of “layer 16” in Figure 6.2) is 256, and the neurons between adjacent groups are fully-connected. The implementation and training details of the baseline methods have been reported in Section 5.3.1 and the results are directly cited in Table 6.1.

Table 6.1: Results on the historical image benchmark.

Methods	Accuracy(%)	MAE
Palermo et al.’s method [Palermo et al., 2012]	44.92±3.69	0.93±0.08
Martin et al.’s method [Martin et al., 2014b]	42.76±1.33	0.87±0.05
Frank and Hall [Frank and Hall, 2001]	41.36±1.89	0.99±0.05
Cardoso and Pinto da Costa [Cardoso and Costa, 2007]	41.32±2.76	0.95±0.04
RED-SVM@8168 [Lin and Li, 2012]	35.92±4.69	0.96±0.06
RED-SVM@deep [Lin and Li, 2012]	25.38±2.34	1.08±0.05
CNNm	48.94±2.54	0.89±0.06
Niu et al.’s method [Niu et al., 2016]	44.67±4.24	0.81±0.06
CNNPOR	50.12±2.65	0.82±0.05
ORCNN	50.23±1.90	0.86±0.06

ORCNN is finetuned from the VGG-16 model pretrained for ImageNet. During the training phase, the learning rate is set to 0.0001 for layers “layer 1” to “layer 15” and 0.001 for “layer 16” and “layer 17”, because these two layers are different from the pretrained model. The size of mini batches is 150, and the number of training iterations is set to 600 for all 20 folds. As shown in Table 6.1, ORCNN outperforms all handcrafted feature based methods. In terms of accuracy, ORCNN outperforms CNNm, Niu et al.’s method and CNNPOR by 1.29%, 5.56% and 0.11% respectively. The mean MAE result of ORCNN

Figure 6.3: The realization of ORCNN based on LeNet for a 3-rank dataset.

is 0.03 lower than CNNm, but it is 0.05 and 0.04 higher than that of Niu et al.’s method and CNNPOR.

6.3.2 Results on the Image Retrieval Dataset

Because the images in MSRA-MM 1.0 dataset are thumbnails in an image retrieval system and the size of images are quite small (cropped as 60×60 pixels in the experiments), LeNet architecture is used to evaluate deep learning methods CNNm, Niu et al.’s method and CNNPOR on this dataset. For fair comparison, ORCNN is realized based on LeNet architecture for MSRA-MM 1.0 dataset as shown in Figure 6.3. “layer 4” is the realization of horizontal layers of ORCNN, which consists of three groups with 32 neurons for three categories of images in MSRA-MM 1.0 dataset.

The training details of baseline deep learning methods have been reported in Section 5.3.2 and the results are directly cited in Table 6.2 and Table 6.3. For ORCNN, the size of training mini batches is set to 64. The training phase is not finetuned from any model, and

the learning rate is set to 0.001 for all layers. The number of training iterations is 2000 for subsets “baby”, “beach”, “cat” and “fish”. For subsets “rose”, “tiger” and “golf”, the number of iterations is 1500, because the training accuracies reach 100% before 1500 iterations.

As shown in Table 6.2, in terms of accuracy, ORCNN outperforms all baseline methods for “beach”, “rose” and “tiger” subsets. None of methods outperforms others on all seven subsets of MSRA-MM 1.0 dataset. Overall, CNNPOR achieves the best results, and ORCNN is the second best one which is 0.59% lower than CNNPOR in terms of accuracy. In terms of MAE, ORCNN wins two times for the seven subsets, and it is 0.010 and 0.013 higher than the MAE results of Niu et al.’s method and CNNPOR.

Table 6.2: Accuracy (%) results on MSRA-MM1.0 dataset.

	RED-SVM @8168	RED-SVM @deep	CNNm	Niu et al.	CNNPOR	ORCNN
Baby	36.99	32.66	48.00	47.33	50.00	48.22
Beach	35.64	34.00	50.67	51.11	51.11	52.67
Cat	40.22	34.89	47.56	48.44	52.89	52.44
Rose	42.05	34.22	55.11	55.78	56.67	56.67
Tiger	35.57	33.56	53.33	51.78	52.89	54.00
Fish	68.66	68.89	63.95	66.16	66.33	65.14
Golf	80.45	80.17	83.08	83.93	84.96	81.54
Overall	48.51	45.48	57.39	57.79	59.26	58.67

Table 6.3: MAE results on MSRA-MM1.0 dataset.

	RED-SVM @8168	RED-SVM @deep	CNNm	Niu et al.	CNNPOR	ORCNN
Baby	0.630	0.699	0.667	0.647	0.636	0.633
Beach	0.648	0.673	0.598	0.576	0.596	0.593
Cat	0.633	0.662	0.676	0.620	0.598	0.618
Rose	0.582	0.664	0.522	0.500	0.500	0.516
Tiger	0.644	0.673	0.571	0.562	0.578	0.553
Fish	0.313	0.311	0.378	0.357	0.355	0.379
Golf	0.283	0.289	0.229	0.219	0.197	0.258
Overall	0.533	0.567	0.520	0.497	0.494	0.507

6.3.3 Results on the Image Aesthetics Dataset

The image aesthetic benchmark is also used to test the performance of ORCNN. The 5-fold random training/testing partitions follows those in the experiments of Section 5.3.3. The number of ranks in image aesthetic benchmark is 5, which is same as the historical color image dataset, and the size of images is also similar as that of historical images. Therefore, the realization of ORCNN architecture is exactly same as the implementation shown in Figure 6.2.

In the experiments, all the deep learning methods, including CNNm, Niu et al.’s method, CNNPOR and ORCNN, are finetuned from the VGG-16 ImageNet model. The images are resized to 256×256 pixels and are randomly cropped to 224×224 pixels further during the training. The learning rate is set to 0.001 for the fully-connected layers “layer 16” and “layer 17” in Figure 6.2, and 0.0001 for all other layers. The number of training iterations is 1000. Table 6.4 and 6.5 summarize the results in terms of accuracy and MAE. For both performance indexes, ORCNN outperforms all the baseline

methods. Overall, ORCNN achieves 1.71% higher accuracy than CNNPOR which is the best baseline method in terms of accuracy, and outperforms 0.018 lower MAE result than CNNPOR.

Table 6.4: Accuracy (%) results on the image aesthetics dataset.

	RED-SVM @8168	RED-SVM @deep	CNNm	Niu et al.	CNNPOR	ORCNN
Nature	69.73	70.72	70.97	69.81	71.86	72.98
Animal	61.14	61.05	68.02	69.10	69.32	72.54
Urban	63.88	65.44	68.19	66.49	69.09	69.44
People	60.06	61.16	71.63	70.44	69.94	72.06
Overall	63.70	64.59	69.45	68.96	70.05	71.76

Table 6.5: MAE results on the image aesthetics dataset.

	RED-SVM @8168	RED-SVM @deep	CNNm	Niu et al.	CNNPOR	ORCNN
Nature	0.319	0.309	0.305	0.313	0.294	0.281
Animal	0.407	0.410	0.342	0.331	0.322	0.292
Urban	0.391	0.374	0.356	0.349	0.325	0.317
People	0.421	0.412	0.315	0.312	0.321	0.302
Overall	0.385	0.376	0.330	0.326	0.316	0.298

6.3.4 Results on the Adience Face Dataset

The adience face dataset used in the evaluations of Chapter 5 is also applied. ORCNN is realized based on VGG-16 architecture as shown in Figure 6.4. The horizontal layers include one layer - “layer 16” with eight groups for eight categories of images in the adience face dataset. The number of neurons in each group is 256. Each group is followed a ReLU layer where works same as the ReLU layers for standard fully-connected layers.

In the experiments, ORCNN is also finetuned from the VGG-16 model for ImageNet. The learning rate for “layer 16” and “layer 17” is 10 times as that for layers “layer 1” to “layer 15” (denoted as base learning rate), which is the same strategy as that to train CNNPOR in Section 5.3.4. In the training phase of ORCNN, the base learning rate is set to 0.001 for first 1850 iterations and 0.0001 to 6000 iterations. The size of mini-batches is 64. Table 6.6 summarizes the results. For both indexes accuracy and MAE, ORCNN is slightly worse than Niu et al.’s method and CNNPOR, but outperforms all other approaches.

Table 6.6: Results on the Adience face dataset.

Methods	Accuracy(%)	MAE
Feature-based		
[Eidinger et al., 2014]	45.1 ± 2.6	-
Lean DNN [Levi and Hassner, 2015]	50.7 ± 5.1	-
CNNm	54.0 ± 6.3	0.61 ± 0.08
Niu et al.	56.7 ± 6.0	0.54 ± 0.08
CNNPOR	57.4 ± 5.8	0.55 ± 0.08
ORCNN	56.5 ± 6.4	0.55 ± 0.09

Figure 6.4: The realization of ORCNN based on VGG-16 for a 8-rank dataset.

6.3.5 Results Summary

The four benchmarks in this chapter evaluate the proposed approach comprehensively from different image content, different image sizes, different number of ranks and different data scales. The experimental results on the four benchmarks are summarized in Table 6.7 and Table 6.8. From the results of Section 6.3.1 to Section 6.3.4, deep learning approaches outperform handcrafted feature based approaches. Therefore, the results of deep learning based methods for the four benchmarks are listed and the overall result is the mean value of the four results for each method. From Table 6.7 and Table 6.8, it can

Table 6.7: Summary of accuracy results (%) on the four benchmarks.

	CNNm	Niu et al.' method	CNNPOR	ORCNN
Historical	48.94	44.67	50.12	50.23
MSRA-MM1.0	57.39	57.79	59.26	58.67
Aesthetics	69.45	68.96	70.05	71.76
Face	54.0	56.7	57.4	56.5
Overall	57.45	57.03	59.27	59.29

Table 6.8: Summary of MAE results on the four benchmarks.

	CNNm	Niu et al. method	CNNPOR	ORCNN
Historical	0.61	0.54	0.55	0.55
MSRA-MM1.0	0.33	0.33	0.32	0.30
Aesthetics	0.52	0.50	0.50	0.51
Face	0.89	0.81	0.82	0.86
Overall	0.59	0.55	0.55	0.56

be concluded that in terms of accuracy ORCNN overall outperforms state-of-the-art methods. For MAE, ORCNN achieves the second best result. The performance of ORCNN is competitive, but in future work, it needs to be improved further especially for MAE.

6.4 Summary

This chapter proposes a new neural network architecture for ordinal regression problems, which is easy to implement and effective. The input and loss function used in the architecture is exactly same as standard multi-class classification. The experimental results show that ORCNN achieves competitive results comparing with state-of-the-art methods.

Chapter 7

Conclusion

This thesis focuses on proposing algorithms for ordinal regression problems on image datasets. The main contributions are to explore the ordinal relationship between categories and adapt deep neural networks to ordinal regression problems from three aspects: input space, loss function and architecture. Moreover, because many real-world ordinal regression applications are in fact small data problems, new approaches are proposed to make possible deep neural networks work on small datasets. In the preceding chapters, we have presented and discussed the proposed four ordinal regression approaches and their performance on different benchmarks. In this chapter, I conclude the main contributions of this thesis and suggest directions for future work.

7.1 Summary of the Main Contributions

The following is a summary of the main contributions of this thesis:

- **A pairwise approach for ordinal regression** proposed in Chapter 3 represents the ordinal relationship by pairing up the instances. A pairwise kernel formulation is proposed for ordinal regression, and this approach is suitable for small datasets because the number of training instances is squared.
- **Deep ordinal regression based on data relationship** presented in Chapter 4 uses

triplets of instances to explore the ordinal relationship, and deep neural networks are adapted to accept triplets as inputs. It makes deep learning work on small datasets because of data augmentation.

- **A constrained deep neural network for ordinal regression** discussed in Chapter 5 embeds the ordinal relationship into constraints of the optimization objective. An equivalent unconstrained formulation is derived and the standard backpropagation training can be used to solve the optimization problem. An efficient method is proposed to train the network, which makes it suitable for small datasets and also scalable for large datasets.
- **A deep ordinal Neural Network with communication between neurons of same layers** is described in Chapter 6, which directly formulates the ordinal relationship by the revised architecture. It works as a standard multiple-class classification network which is scalable for large datasets.

7.2 Future Work

The ordinal regression approaches proposed in this thesis take advantage of deep learning, which is able to automatically extract features from raw data, afford high-level representation by multiple-layers architecture and provide scalable solutions for large scale data problems. However, deep neural networks trained with backpropagation have some disadvantages, e.g., having to tune a large number of hyperparameters to the data, lack of calibrated probabilistic predictions, and a tendency to overfit the training data. Currently, most of deep neural networks work as a block box. Although deep networks are powerful for lots of tasks, it is difficult to clearly indicate when and where the networks work. The deep learning based ordinal regression approaches proposed in this thesis inherit the limits of deep neural networks.

Bayesian deep learning as a marriage of the Bayesian technique and deep neural net-

works has both advantages of calibrated probabilistic predictions and scalability to large datasets. However, it is challenging to learn a Bayesian network for the ordinal regression problem which aims to classify instances into ordinal categories, because the target is not only to learn the regression function but also the multiple boundaries to discretize the function value. In the future work, I am interested in investigating the bayesian deep learning for ordinal regression aiming to effectively predict the ordinal labels for instances, and also be able to provide the confidence of the predictions.

From data scale point of view, in this thesis, to make the proposed ordinal regression approaches work on small datasets, the efforts are mainly done from two angles: augmenting data in the input space by encoding the ordinal relationship among instances; introducing constraints into the loss function to regularize the networks to avoid overfitting. However, to tackle the problem of lack of training data, transfer learning is a convincing method which makes use of data from other relevant domains as discussed in Section 2.3. It is a future direction to employ the transfer learning technique into the ordinal regression problem. Recently, generative adversarial network (GAN) has been successfully applied in both transfer learning and ordinal regression problems. Thus, an immediate attempt is to adapt GAN to solve ordinal regression problems in the domain transfer setting.

Another future work is to extend ordinal regression approaches to two more application scenarios, i.e., time series ordinal regression and circular ordinal regression. Ordinal regression on time series datasets focuses on predicting the rank label for a whole series, not for the future time stamps of a certain series. For example, risk level estimation of simulated aircraft flights aims to grade the discrete probabilities that an aircraft will crash based on the time series features during flights. Another example is to rate the interaction speech quality as described in Section 2.4, based on the series of speech records. Circular ordinal regression involves rank labels in a loop. For example, to predict the time when outdoor images are photographed as “morning”, “noon”, “afternoon”, “evening”, “midnight” and “early morning”. After “early morning”, it will be “morning” again. The

distance between the first rank “morning” and the last rank “early morning” is one interval, so they are circular rank labels. There are very few ordinal regression approaches on these two application scenarios, and it will be helpful to extend the proposed approaches in this thesis to wider fields.

Bibliography

- [Allwein et al., 2000] Allwein, E. L., Schapire, R. E., and Singer, Y. (2000). Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141.
- [Baccianella et al., 2009] Baccianella, S., Esuli, A., and Sebastiani, F. (2009). Evaluation measures for ordinal regression. In *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*, pages 283–287. IEEE.
- [Baccianella et al., 2014] Baccianella, S., Esuli, A., and Sebastiani, F. (2014). Feature selection for ordinal text classification. *Neural computation*, 26(3):557–591.
- [Bao and Hanson, 2015] Bao, J. and Hanson, T. E. (2015). Bayesian nonparametric multivariate ordinal regression. *Canadian Journal of Statistics*, 43(3):337–357.
- [Brunner et al., 2012] Brunner, C., Fischer, A., Luig, K., and Thies, T. (2012). Pairwise support vector machines and their application to large scale problems. *Journal of Machine Learning Research*, 13(Aug):2279–2292.
- [Campoy-Muñoz et al., 2013] Campoy-Muñoz, P., Gutiérrez, P. A., and Hervás-Martínez, C. (2013). Addressing remitting behavior using an ordinal classification approach. In *Natural and Artificial Computation in Engineering and Medical Applications*, pages 326–335. Springer.

- [Cardoso and Costa, 2007] Cardoso, J. S. and Costa, J. F. (2007). Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research*, 8(Jul):1393–1429.
- [Carrizosa and Martin-Barragan, 2011] Carrizosa, E. and Martin-Barragan, B. (2011). Maximizing upgrading and downgrading margins for ordinal regression. *Mathematical Methods of Operations Research*, 74(3):381–407.
- [Chang et al., 2014] Chang, C.-J., Li, D.-C., Chen, C.-C., and Chen, C.-S. (2014). A forecasting model for small non-equigap data sets considering data weights and occurrence possibilities. *Computers & Industrial Engineering*, 67:139–145.
- [Chang et al., 2009] Chang, X., Zheng, Q., and Lin, P. (2009). Ordinal regression with sparse bayesian. In *Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence*, pages 591–599. Springer.
- [Chen et al., 2014] Chen, W., Xiong, C., Xu, R., and Corso, J. (2014). Actionness ranking with lattice conditional ordinal random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 748–755.
- [Cheng et al., 2008] Cheng, J., Wang, Z., and Pollastri, G. (2008). A neural network approach to ordinal regression. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1279–1284. IEEE.
- [Chu and Ghahramani, 2005] Chu, W. and Ghahramani, Z. (2005). Gaussian processes for ordinal regression. In *Journal of Machine Learning Research*, pages 1019–1041.
- [Chu and Keerthi, 2007] Chu, W. and Keerthi, S. S. (2007). Support vector ordinal regression. *Neural computation*, 19(3):792–815.

- [Corrente et al., 2013] Corrente, S., Greco, S., Kadziński, M., and Słowiński, R. (2013). Robust ordinal regression in preference learning and ranking. *Machine Learning*, 93(2-3):381–422.
- [Crammer et al., 2001] Crammer, K., Singer, Y., et al. (2001). Pranking with ranking. In *Nips*, volume 1, pages 641–647.
- [Crane et al., 2016] Crane, M., Rissel, C., Greaves, S., and Gebel, K. (2016). Correcting bias in self-rated quality of life: an application of anchoring vignettes and ordinal regression models to better understand qol differences across commuting modes. *Quality of Life Research*, 25(2):257–266.
- [Deepak and Sivaswamy, 2012] Deepak, K. S. and Sivaswamy, J. (2012). Automatic assessment of macular edema from color retinal images. *IEEE Transactions on medical imaging*, 31(3):766–776.
- [Deng et al., 2010] Deng, W.-Y., Zheng, Q.-H., Lian, S., Chen, L., and Wang, X. (2010). Ordinal extreme learning machine. *Neurocomputing*, 74(1):447–456.
- [DeYoreo et al., 2015] DeYoreo, M., Kottas, A., et al. (2015). A fully nonparametric modeling approach to binary regression. *Bayesian Analysis*, 10(4):821–847.
- [DIARETDB,] DIARETDB, D. Evaluation database and methodology for diabetic retinopathy algorithms may 2007.
- [Donat and Marra, 2015] Donat, F. and Marra, G. (2015). Semi-parametric bivariate polychotomous ordinal regression. *Statistics and Computing*, pages 1–17.
- [Dorado-Moreno et al., 2015] Dorado-Moreno, M., Sianes, A., and Hervás-Martínez, C. (2015). From outside to hyper-globalisation: an artificial neural network ordinal classifier applied to measure the extent of globalisation. *Quality & Quantity*, pages 1–28.

- [Dougherty et al., 2015] Dougherty, E. R., Dalton, L. A., and Alexander, F. J. (2015). Small data is the problem. In *Signals, Systems and Computers, 2015 49th Asilomar Conference on*, pages 418–422. IEEE.
- [Doyle et al., 2013] Doyle, O. M., Ashburner, J., Zelaya, F., Williams, S. C., Mehta, M. A., and Marquand, A. F. (2013). Multivariate decoding of brain images using ordinal regression. *NeuroImage*, 81:347–357.
- [Dreher et al., 2008] Dreher, A., Gaston, N., and Martens, P. (2008). *Measuring globalisation: Gauging its consequences*. Springer Science & Business Media.
- [Easterly and Pfutze, 2008] Easterly, W. and Pfutze, T. (2008). Where does the money go? best and worst practices in foreign aid. *Journal of Economic Perspectives*, 22(2).
- [Eidinger et al., 2014] Eidinger, E., Enbar, R., and Hassner, T. (2014). Age and gender estimation of unfiltered faces. *IEEE Transactions on Information Forensics and Security*, 9(12):2170–2179.
- [El Asri et al., 2014] El Asri, L., Khouzaimi, H., Laroche, R., and Pietquin, O. (2014). Ordinal regression for interaction quality prediction. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 3221–3225. IEEE.
- [Eurostat, 2010] Eurostat (2010). Eurostat official website. Available at <http://epp.eurostat.ec.europa.eu/portal/page/portal/sdi/>.
- [Feng et al., 2014] Feng, D., Chen, F., and Xu, W. (2014). Supervised feature subset selection with ordinal optimization. *Knowledge-Based Systems*, 56:123–140.
- [Feng et al., 2015] Feng, X.-N., Wu, H.-T., and Song, X.-Y. (2015). Bayesian adaptive lasso for ordinal regression with latent variables. *Sociological Methods & Research*, page 0049124115610349.

- [Fernandez-Navarro et al., 2013] Fernandez-Navarro, F., Campoy-Munoz, P., la Paz-Marin, M.-d., Hervás-Martínez, C., and Yao, X. (2013). Addressing the eu sovereign ratings using an ordinal regression approach. *Cybernetics, IEEE Transactions on*, 43(6):2228–2240.
- [Fernandez-Navarro et al., 2014] Fernandez-Navarro, F., Riccardi, A., and Carloni, S. (2014). Ordinal neural networks without iterative tuning. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(11):2075–2085.
- [FGNet, 2010] FGNet (2010). The fg-net aging database. Available at <http://www.fgnet.rsunit.com/>.
- [Fokoue and Gündüz, 2013] Fokoue, E. and Gündüz, N. (2013). Data mining and machine learning techniques for extracting patterns in students’ evaluations of instructors.
- [Frank and Hall, 2001] Frank, E. and Hall, M. (2001). *A simple approach to ordinal classification*. Springer.
- [George et al., 2016] George, N. I., Lu, T.-P., and Chang, C.-W. (2016). Cost-sensitive performance metric for comparing multiple ordinal classifiers. *Artificial Intelligence Research*, 5(1):p135.
- [Gómez-Rey et al., 2015] Gómez-Rey, P., Fernández-Navarro, F., and Barberà, E. (2015). Ordinal regression by a gravitational model in the field of educational data mining. *Expert Systems*.
- [Graham and Allinson, 1998] Graham, D. B. and Allinson, N. M. (1998). Characterising virtual eigensignatures for general purpose face recognition. In *Face Recognition*, pages 446–456. Springer.
- [Gu and Sheng, 2013] Gu, B. and Sheng, V. S. (2013). Feasibility and finite convergence analysis for accurate on-line-support vector machine. *Neural Networks and Learning Systems, IEEE Transactions on*, 24(8):1304–1315.

- [Gu et al., 2015] Gu, B., Sheng, V. S., Tay, K. Y., Romano, W., and Li, S. (2015). Incremental support vector learning for ordinal regression. *Neural Networks and Learning Systems, IEEE Transactions on*, 26(7):1403–1416.
- [Gu et al., 2010] Gu, B., Wang, J.-D., and Li, T. (2010). Ordinal-class core vector machine. *Journal of Computer Science and Technology*, 25(4):699–708.
- [Guo et al., 2009] Guo, G., Mu, G., Fu, Y., and Huang, T. S. (2009). Human age estimation using bio-inspired features. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 112–119. IEEE.
- [Guo et al., 2015] Guo, J., Levina, E., Michailidis, G., and Zhu, J. (2015). Graphical models for ordinal data. *Journal of Computational and Graphical Statistics*, 24(1):183–204.
- [Gutierrez et al., 2016] Gutierrez, P. A., Perez-Ortiz, M., Sanchez-Monedero, J., Fernandez-Navarro, F., and Hervas-Martinez, C. (2016). Ordinal regression methods: survey and experimental study. *Knowledge and Data Engineering, IEEE Transactions on*, 28(1):127–146.
- [Gutiérrez et al., 2014] Gutiérrez, P. A., Tiño, P., and Hervás-Martínez, C. (2014). Ordinal regression neural networks based on concentric hyperspheres. *Neural Networks*, 59:51–60.
- [Hao et al., 2016] Hao, Z., Hong, Y., Xia, Y., Singh, V. P., Hao, F., and Cheng, H. (2016). Probabilistic drought characterization in the categorical form using ordinal regression. *Journal of Hydrology*, 535:331–339.
- [Herbrich et al., 1999] Herbrich, R., Graepel, T., and Obermayer, K. (1999). Large margin rank boundaries for ordinal regression. *Advances in neural information processing systems*, pages 115–132.

- [Ho, 1999] Ho, Y.-C. (1999). An explanation of ordinal optimization: soft computing for hard problems. *information Sciences*, 113(3):169–192.
- [Huang et al., 2006] Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501.
- [Joachims, 2002] Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM.
- [Kälviäinen and Uusitalo, 2007] Kälviäinen, R. and Uusitalo, H. (2007). Diaretdb1 diabetic retinopathy database and evaluation protocol. In *Medical Image Understanding and Analysis*, volume 2007, page 61. Citeseer.
- [Kampen and Swyngedouw, 2000] Kampen, J. and Swyngedouw, M. (2000). The ordinal controversy revisited. *Quality and quantity*, 34(1):87–102.
- [Kim, 2014] Kim, M. (2014). Conditional ordinal random fields for structured ordinal-valued label prediction. *Data mining and knowledge discovery*, 28(2):378–401.
- [Kim et al., 2016] Kim, S., Kim, H., and Namkoong, Y. (2016). Ordinal classification of imbalanced data with application in emergency and disaster information services. *IEEE Intelligent Systems*, 31(5):50–56.
- [Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- [Kuranova and Hajdukova, 2014] Kuranova, P. and Hajdukova, Z. (2014). Ordinal regression for classification of patients into one of the individual phadiatop test groups. In *Digital Technologies (DT), 2014 10th International Conference on*, pages 174–178. IEEE.

- [Lalla, 2016] Lalla, M. (2016). Fundamental characteristics and statistical analysis of ordinal variables: a review. *Quality & Quantity*, pages 1–24.
- [Laptev et al., 2008] Laptev, I., Marszałek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.
- [Le et al., 2006] Le, Q. V., Smola, A. J., Gärtner, T., and Altun, Y. (2006). Transductive gaussian process regression with automatic model selection. In *Machine Learning: ECML 2006*, pages 306–317. Springer.
- [LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [Levi and Hassner, 2015] Levi, G. and Hassner, T. (2015). Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–42.
- [Li and Lin, 2007] Li, L. and Lin, H.-T. (2007). Ordinal regression by extended binary classification. In *Advances in neural information processing systems*, pages 865–872.
- [Lin and Li, 2009] Lin, H.-T. and Li, L. (2009). Combining ordinal preferences by boosting. In *Proceedings ECML/PKDD 2009 Workshop on Preference Learning*, pages 69–83.
- [Lin and Li, 2012] Lin, H.-T. and Li, L. (2012). Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Computation*, 24(5):1329–1367.
- [Liu et al., 2009] Liu, T.-Y. et al. (2009). Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

- [Liu et al., 2016] Liu, Y., Li, X., Kong, A. W. K., and Goh, C. K. (2016). Learning from small data: A pairwise approach for ordinal regression. In *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*, pages 1–6. IEEE.
- [Liu et al., 2011a] Liu, Y., Liu, Y., and Chan, K. C. (2011a). Ordinal regression via manifold learning. In *Twenty-fifth AAAI conference on artificial intelligence*.
- [Liu et al., 2011b] Liu, Y., Liu, Y., Zhong, S., and Chan, K. C. (2011b). Semi-supervised manifold ordinal regression for image ranking. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1393–1396. ACM.
- [Lucey et al., 2011] Lucey, P., Cohn, J. F., Prkachin, K. M., Solomon, P. E., and Matthews, I. (2011). Painful data: The unbc-mcmaster shoulder pain expression archive database. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 57–64. IEEE.
- [Martin et al., 2014a] Martin, E., Shaheen, S., Lipman, T., and Camel, M. (2014a). Evaluating the public perception of a feebate policy in california through the estimation and cross-validation of an ordinal regression model. *Transport Policy*, 33:144–153.
- [Martin et al., 2014b] Martin, P., Doucet, A., and Jurie, F. (2014b). Dating color images with ordinal classification. In *Proceedings of International Conference on Multimedia Retrieval*, page 447. ACM.
- [Mavadati et al., 2013] Mavadati, S. M., Mahoor, M. H., Bartlett, K., Trinh, P., and Cohn, J. F. (2013). Disfa: A spontaneous facial action intensity database. *Affective Computing, IEEE Transactions on*, 4(2):151–160.
- [McCullagh, 1980] McCullagh, P. (1980). Regression models for ordinal data. *Journal of the royal statistical society. Series B (Methodological)*, pages 109–142.
- [McCullagh and Nelder, 1989] McCullagh, P. and Nelder, J. A. (1989). *Generalized linear models*, volume 37. CRC press.

- [McKinley et al., 2015] McKinley, T. J., Morters, M., Wood, J. L., et al. (2015). Bayesian model choice in cumulative link ordinal regression models. *Bayesian Analysis*, 10(1):1–30.
- [Minetos and Polyzos, 2010] Minetos, D. and Polyzos, S. (2010). Deforestation processes in greece: A spatial analysis by using an ordinal regression model. *Forest Policy and Economics*, 12(6):457–472.
- [Nelson and Edwards, 2015] Nelson, K. P. and Edwards, D. (2015). Measures of agreement between many raters for ordinal classifications. *Statistics in medicine*, 34(23):3116–3132.
- [Niu et al., 2016] Niu, Z., Zhou, M., Wang, L., Gao, X., and Hua, G. (2016). Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4920–4928.
- [O’Connell, 2006] O’Connell, A. A. (2006). *Logistic regression models for ordinal response variables*. Number 146. Sage.
- [Ogawa et al., 2014] Ogawa, T., Takahashi, S., Takahashi, S., and Haseyama, M. (2014). A new method for error degree estimation in numerical weather prediction via mkda-based ordinal regression. *Eurasip journal on advances in signal processing*, 2014(1):1–17.
- [Palermo et al., 2012] Palermo, F., Hays, J., and Efros, A. A. (2012). Dating historical color images. In Fitzgibbon, A., Lazebnik, S., Sato, Y., and Schmid, C., editors, *ECCV (6)*, volume 7577 of *Lecture Notes in Computer Science*, pages 499–512. Springer.
- [Pérez-Ortiz et al., 2014] Pérez-Ortiz, M., de la Paz-Marín, M., Gutiérrez, P. A., and Hervás-Martínez, C. (2014). Classification of eu countries’ progress towards sustainable development based on ordinal regression techniques. *Knowledge-Based Systems*, 66:178–189.

- [Perez-Ortiz et al., 2015] Perez-Ortiz, M., Gutierrez, P. A., Hervás-Martínez, C., and Yao, X. (2015). Graph-based approaches for over-sampling in the context of ordinal regression. *Knowledge and Data Engineering, IEEE Transactions on*, 27(5):1233–1245.
- [Riccardi et al., 2014] Riccardi, A., Fernández-Navarro, F., and Carloni, S. (2014). Cost-sensitive adaboost algorithm for ordinal regression based on extreme learning machine. *Cybernetics, IEEE Transactions on*, 44(10):1898–1909.
- [Rissel et al., 2013] Rissel, C., Greaves, S., Wen, L. M., Capon, A., Crane, M., and Standen, C. (2013). Evaluating the transport, health and economic impacts of new urban cycling infrastructure in sydney, australia—protocol paper. *BMC public health*, 13(1):963.
- [Robertson et al., 2009] Robertson, C., Wulder, M. A., Nelson, T. A., White, J. C., et al. (2009). *Preliminary risk rating for mountain pine beetle infestation of lodgepole pine forests over large areas with ordinal regression modelling*, volume 2009. Pacific Forestry Centre.
- [Rodriguez et al., 2008] Rodriguez, M. D., Ahmed, J., and Shah, M. (2008). Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.
- [ROSE, 2014] ROSE (2014). A dataset of text documents collected from the hungarian national association of radio distress-signalling and infocommunications (rsoe) over a period of 36 months from january 2012 to december 2014. Available at <https://hisz.rsoe.hu/>.
- [Rudovic et al., 2012] Rudovic, O., Pavlovic, V., and Pantic, M. (2012). Kernel conditional ordinal random fields for temporal segmentation of facial action units. In *Computer Vision—ECCV 2012. Workshops and Demonstrations*, pages 260–269. Springer.

- [Sánchez-Monedero et al., 2013] Sánchez-Monedero, J., Gutiérrez, P. A., Tiño, P., and Hervás-Martínez, C. (2013). Exploitation of pairwise class distances for ordinal classification. *Neural computation*, 25(9):2450–2485.
- [Schifanella et al., 2015] Schifanella, R., Redi, M., and Aiello, L. M. (2015). An image is worth more than a thousand favorites: Surfacing the hidden beauty of flickr pictures. In *ICWSM*, pages 397–406.
- [Schmitt et al., 2012] Schmitt, A., Ultes, S., and Minker, W. (2012). A parameterized and annotated spoken dialog corpus of the cmu let’s go bus information system. In *LREC*, pages 3369–3373.
- [Seah et al., 2012] Seah, C.-W., Tsang, I. W., and Ong, Y.-S. (2012). Transductive ordinal regression. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(7):1074–1086.
- [Shashua and Levin, 2002] Shashua, A. and Levin, A. (2002). Taxonomy of large margin principle algorithms for ordinal regression problems. *Advances in neural information processing systems*, 15:937–944.
- [Sianes et al., 2014] Sianes, A., Dorado-Moreno, M., and Hervás-Martínez, C. (2014). Rating the rich: an ordinal classification to determine which rich countries are helping poorer ones the most. *Social indicators research*, 116(1):47–65.
- [Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- [Srijith et al., 2012] Srijith, P., Shevade, S., and Sundararajan, S. (2012). A probabilistic least squares approach to ordinal regression. In *AI 2012: Advances in Artificial Intelligence*, pages 683–694. Springer.

- [Srijith et al., 2013] Srijith, P., Shevade, S., and Sundararajan, S. (2013). Semi-supervised gaussian process ordinal regression. In *Machine Learning and Knowledge Discovery in Databases*, pages 144–159. Springer.
- [Sun et al., 2010] Sun, B.-Y., Li, J., Wu, D. D., Zhang, X.-M., and Li, W.-B. (2010). Kernel discriminant learning for ordinal regression. *IEEE Transactions on Knowledge and Data Engineering*, 22(6):906–910.
- [Sun et al., 2015] Sun, B.-Y., Wang, H.-L., Li, W.-B., Wang, H.-J., Li, J., and Du, Z.-Q. (2015). Constructing and combining orthogonal projection vectors for ordinal regression. *Neural Processing Letters*, 41(1):139–155.
- [Tian and Chen, 2015] Tian, Q. and Chen, S. (2015). A novel ordinal learning strategy: Ordinal nearest-centroid projection. *Knowledge-Based Systems*, 88:144–153.
- [Tian et al., 2016] Tian, Q., Chen, S., and Qiao, L. (2016). Ordinal margin metric learning and its extension for cross-distribution image data. *Information Sciences*, 349:50–64.
- [Tom et al., 2007] Tom, S. M., Fox, C. R., Trepel, C., and Poldrack, R. A. (2007). The neural basis of loss aversion in decision-making under risk. *Science*, 315(5811):515–518.
- [Tran et al., 2015] Tran, T., Phung, D., Luo, W., and Venkatesh, S. (2015). Stabilized sparse ordinal regression for medical risk stratification. *Knowledge and Information Systems*, 43(3):555–582.
- [Tsang et al., 2005] Tsang, I. W., Kwok, J. T., and Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. In *Journal of Machine Learning Research*, pages 363–392.
- [USDM, 2010] USDM (2010). The u.s. drought monitor (usdm) website. Available at <http://droughtmonitor.unl.edu/>.

- [Vapnik, 2013] Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.
- [Waegeman et al., 2006] Waegeman, W., De Baets, B., and Boullart, L. (2006). A comparison of different roc measures for ordinal regression. In *Proceedings of the 3rd International Workshop on ROC Analysis in Machine Learning*, N. Lachiche, C. Ferri, and S. Macskassy, Eds, pages 63–69. Citeseer.
- [Wang et al., 2009] Wang, M., Yang, L., and Hua, X.-S. (2009). Msra-mm: Bridging research and industrial societies for multimedia information retrieval. *Microsoft Research Asia, Tech. Rep.*
- [Wang et al., 2015] Wang, S., Zhai, J., Zhang, S., and Zhu, H. (2015). An ordinal random forest and its parallel implementation with mapreduce. In *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*, pages 2170–2173. IEEE.
- [Wang et al., 2014] Wang, S., Zhai, J., Zhu, H., and Wang, X. (2014). Parallel ordinal decision tree algorithm and its implementation in framework of mapreduce. In *Machine Learning and Cybernetics*, pages 241–251. Springer.
- [Yan, 2014] Yan, H. (2014). Cost-sensitive ordinal regression for fully automatic facial beauty assessment. *Neurocomputing*, 129:334–342.
- [Yao et al., 2011] Yao, B., Jiang, X., Khosla, A., Lin, A. L., Guibas, L., and Fei-Fei, L. (2011). Human action recognition by learning bases of action attributes and parts. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1331–1338. IEEE.
- [Zhao et al., 2009] Zhao, B., Wang, F., and Zhang, C. (2009). Block-quantized support vector ordinal regression. *Neural Networks, IEEE Transactions on*, 20(5):882–890.
- [Zhu et al., 2009] Zhu, J., Zou, H., Rosset, S., and Hastie, T. (2009). Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360.