Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

RESERVE PRICE OPTIMIZATION IN DISPLAY ADVERTISING

by
Achir Kalra

Display advertising is the main type of online advertising, and it comes in the form of banner ads and rich media on publishers' websites. Publishers sell ad impressions, where an impression is one display of an ad in a web page. A common way to sell ad impressions is through real-time bidding (RTB). In 2019, advertisers in the United States spent nearly 60 billion U.S. dollars on programmatic digital display advertising. By 2022, expenditures are expected to increase to nearly 95 billion U.S. dollars. In general, the remaining impressions are sold directly by the publishers. The only way for publishers to control the price of the impressions they sell through RTB is by setting up a reserve price, which has to be beaten by the winning bids.

The two main types of RTB auction strategies are 1) first-price auctions, i.e., the winning advertiser pays the highest bid, and 2) second-price auctions, i.e., the winning advertiser pays the maximum of the second highest bid and the reserve price (the minimum price that a publisher can accept for an impression). In both types of auctions, bids lower than the reserve prices will be automatically rejected. Since both strategies are influenced by the reserve price, setting a good reserve price is an important, but challenging task for publishers. A high reserve price may lead to very few winning bids, and thus can decrease the revenue substantially. A low reserve price may devalue the impressions and hurt the revenue because advertisers do not need to bid high to beat the reserve. Reduction of ad revenue may affect the quality of free content and publishers' business sustainability. Therefore, in an ideal situation, the publishers would like to set the reserve price as high as possible, while ensuring that there is a winning bid.

This dissertation proposes to use machine learning techniques to determine the optimal reserve prices for individual impressions in real-time, with the goal of maximizing publishers' ad revenue. The proposed techniques are practical because they use data only available to publishers. They are also general because they can be applied to most online publishers. The novelty of the research comes from both the problem, which was not studied before, and the proposed techniques, which are adapted to the online publishing domain.

For second-price auctions, a survival-analysis-based model is first proposed to predict failure rates of reserve prices of specific impressions in second-price auctions. It uses factorization machines (FM) to capture feature interaction and header bidding information to improve the prediction performance. The experiments, using data from a large media company, show that the proposed model for failure rate prediction outperforms the comparative systems. The survival-analysis-based model is augmented further with a deep neural network (DNN) to capture the feature interaction. The experiments show that the DNN-based model further improves the performance from the FM-based one.

For first-price auctions, a multi-task learning framework is proposed to predict the lower bounds of highest bids with a coverage probability. The model can guarantee the highest bids of at least a certain percentage of impressions are more than the corresponding predicted lower bounds. Setting the final reserve prices to the lower bounds, the model can guarantee a certain percentage of outbid impressions in real-time bidding. The experiments show that the proposed method can significantly outperform the comparison systems.

RESERVE PRICE OPTIMIZATION IN DISPLAY ADVERTISING

by
Achir Kalra

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

August 2021

APPROVAL PAGE

RESERVE PRICE OPTIMIZATION IN DISPLAY ADVERTISING

Achir Kalra

Dr. Yi Chen, Dissertation Advisor — Date
Professor, Martin Tuchman School of Management; Joint Appointment in Department of Computer Science, NJIT

Dr. Cristian Borcea, Dissertation Co-Advisor — Date
Professor, Department of Computer Science, NJIT

Dr. Vincent Oria, Committee Member — Date
Professor, Department of Computer Science, NJIT

Dr. Chase Wu, Committee Member — Date
Professor, Department of Computer Science, NJIT

Dr. Michael Ehrlich, Committee Member — Date
Associate Professor, Martin Tuchman School of Management, NJIT

BIOGRAPHICAL SKETCH

Author: Achir Kalra

Degree: Doctor of Philosophy

Date: August 2021

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science
  New Jersey Institute of Technology, 2021
- Bachelor's in Information Technology
  University of Delhi, Delhi, India, 2002

Major: Computer Science

Presentations and Publications:

Shuai Zhao, Achir Kalra, Cristian Borcea, and Yi Chen. To be tough or soft: measuring the impact of counter-ad-blocking strategies on user engagement. In the Web Conference 2020, pp. 2690-2696, 2020.

Shuai Zhao, Achir Kalra, Chong Wang, Cristian Borcea, and Yi Chen. Ad blocking whitelist prediction for online publishers. The 2019 IEEE International Conference on Big Data (Big Data), pp. 1711-1716, 2019.

Achir Kalra, Chong Wang, Cristian Borcea, and Yi Chen. Reserve price failure rate prediction with header bidding in display advertising. The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019

Chong Wang, Shuai Zhao, Achir Kalra, Cristian Borcea, and Yi Chen. Webpage depth viewability prediction using deep sequential neural networks. IEEE Transactions on Knowledge and Data Engineering, 31(3), 601-614, 2018.

Chong Wang, Shuai Zhao, Achir Kalra, Cristian Borcea, and Yi Chen. Predictive models and analysis for webpage depth-level dwell time. Journal of the Association for Information Science and Technology, 69(8), 1007-1022, 2018.

Chong Wang, Achir Kalra, Li Zhou, Cristian Borcea., and Yi Chen. Probabilistic models for ad viewability prediction on the web. IEEE Transactions on Knowledge and Data Engineering, 29(9), 2012-2025, 2017.

Shuai Zhao, Chong Wang, Achir Kalra, Leon Vaks, Cristian Borcea, and Yi Chen. Ad blocking and counter-ad blocking: analysis of online ad blocker usage. The 23rd Americas Conference on Information Systems, 2017.

Chong Wang, Achir Kalra, Cristian Borcea, and Yi Chen. Webpage depth-level dwell time prediction. The 25th ACM International Conference on Information and Knowledge Management (pp. 1937-1940), 2016.

Chong Wang, Achir Kalra, Cristian Borcea, and Yi Chen. Revenue-optimized webpage recommendation. 2015 IEEE International Conference on Data Mining Workshop (ICDMW), 2015.

Chong Wang, Achir Kalra, Cristian Borcea, and Yi Chen. Viewability prediction for online display ads. the 24th ACM International on Conference on Information and Knowledge Management, 2015.

Ankur Gupta, Achir Kalra, Daniel Boston, and Cristian Borcea. MobiSoC: a middleware for mobile social computing applications. Mobile Networks and Applications, 14(1), 35-52, 2009.

Cristian Borcea, Ankur Gupta, Achir Kalra, Quentin Jones, and Iftode, L. The mobisoc middleware for mobile social computing: challenges, design, and early experiences. The 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (p. 27), 2008.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my advisor, Dr. Yi Chen and co-advisor Dr. Cristian Borcea, for their sustained direction and support.

Additionally, I would also like to thank Dr. Michael Ehrlich, Dr. Vincent Oria and Dr. Chase Wu for being on the dissertation committee and providing invaluable inputs and feedback. My sincere gratitude to David Tress and Ms. Clarisa Gonzalez-Lenahan for their support and guidance.

I would also like to thank Dr. Sotirios Ziavras and everyone with the Computer Science department, the teaching assistant ship and to the National Science Foundation for the research funding under Grants No. CNS-0454081, IIS-0534520, CNS-0520033, and CNS-0831753.

Additionally, I also wish to extend my sincere thanks to all my colleagues and team at Forbes Media, with whom I had the privilege of working, researching, developing and understanding new concepts and techniques.

I am thankful to all my wonderful friends and everyone in the family. I would especially like to thank my mother Madhu Kalra for her unconditional love and support and my Uncle, Praveen Kalra, who has always been a father figure for me and instrumental in encouraging and supporting all my endeavors since childhood.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Overview of The Research 6

1.3 Organization of the Dissertation 6

2 LITERATURE REVIEW 8

2.1 Reserve Price Optimization 8

2.2 Matrix Factorization 11

2.3 Feature Interaction in Deep Learning 12

2.4 Prediction Interval Estimation 12

3 RESERVE PRICE FAILURE RATE PREDICTION IN SECOND-PRICE AUCTIONS USING FACTORIZATION MACHINES 16

3.1 Background and Motivation 16

3.2 Reserve Price Failure Rate Prediction 19

3.2.1 Real-Life Datasets 19

3.2.2 Data Censorship 21

3.2.3 The Proposed Parametric Survival Model 22

3.2.4 Pairwise Interaction Tensor Factorization 26

3.2.5 Header Bidding Regularization 27

3.3 Evaluation 30

3.3.1 Experimental Dataset and Ground Truth 30

3.3.2 Implementation 31

3.3.3 Experimental Metrics 32

3.3.4 Comparison Systems 34

3.3.5 Comparison of Different Distributions 34

3.3.6 Comparison of Different Dimensionalities 37

TABLE OF CONTENTS
(Continued)

Chapter **Page**

3.3.7 Overall Comparison 38

3.4 Summary 40

4 RESERVE PRICE FAILURE RATE PREDICTION IN SECOND-PRICE AUCTIONS USING DEEP NEURAL NETWORKS 41

4.1 Motivation 41

4.2 Reserve Price Failure Prediction 41

4.2.1 Deep Neural Network for Feature Interaction 41

4.3 Evaluation 45

4.3.1 Comparison of Different Distributions 45

4.3.2 Comparison of Different Dimensionalities 48

4.3.3 Overall Comparison 51

4.4 Summary 53

5 RESERVE PRICE OPTIMIZATION IN FIRST-PRICE AUCTIONS . . . 54

5.1 The Proposed Multi-Task Learning Loss Function 57

5.1.1 Problem Definition 58

5.1.2 Multi-task Learning Framework 58

5.1.3 Highest Bid Lower Bound Prediction 59

5.1.4 Hazard Rate Prediction 61

5.2 Predicting Highest Bid Lower Bounds and Hazard Rates 63

5.2.1 Risk Level Selection 67

5.3 Evaluation 67

5.3.1 Experimental Data 67

5.3.2 Implementation 68

5.3.3 Evaluation Metrics 68

5.3.4 Comparison Systems 69

5.3.5 Performance by Varying λ and μ 71

TABLE OF CONTENTS
(Continued)

Chapter **Page**

5.3.6 Performance of Lower Bound Highest Bid Prediction 72

5.3.7 Performance of Different Risk Levels 74

5.3.8 Performance on Different Sizes of Training Data 77

5.3.9 Performance of Different Neural Network Parameters 78

5.4 Summary 79

6 CONCLUSIONS AND FUTURE WORK 82

REFERENCES 84

LIST OF TABLES

Table **Page**

3.1 Comparison for All Impressions 39

3.2 Comparison for Impressions with Header Bids Only 39

4.1 Comparison for All Impressions 52

4.2 Comparison for Impressions with Header Bids 52

5.1 Performance of Different Sizes of Training Data 79

5.2 Performance of Different Embedding Widths 80

5.3 Performance of Different Number of Abstraction Layers 81

LIST OF FIGURES

Figure **Page**

1.1 An example of a display ad. 2

1.2 Ad impression selling in RTB in second-price auctions. 3

3.1 Impression revenue in the second-price auctions. 17

3.2 Data preparation. 20

3.3 Three censorship cases. 21

3.4 Two cases for header bidding regularization. 28

3.5 Pairwise relationships. 33

3.6 Log-loss of different distributions with different shapes. 34

3.7 Accuracy of different distributions with different shapes. 35

3.8 C-index of different distributions with different shapes. 35

3.9 Log-loss of different distributions with different k. 35

3.10 Accuracy of different distributions with different k. 36

3.11 C-index of different distributions with different k. 36

4.1 The architecture of the deep neural network. 42

4.2 Log-loss of shapes 46

4.3 Accuracy of shapes. 46

4.4 C-Index of shapes. 47

4.5 Log-loss of ks. 48

4.6 Accuracy of ks. 48

4.7 C-Index of ks. 49

4.8 Log-loss of embedding width. 50

4.9 Accuracy of embedding width. 50

4.10 C-index of embedding width. 51

5.1 An example of a display ad. 55

5.2 The architecture of the proposed multi-task learning neural network. . . 65

LIST OF FIGURES
(Continued)

Figure **Page**

5.3 The PICPs of different λ and μ. 72

5.4 The MORPs of different λ and μ. 73

5.5 The revenue of different λ and μ. 74

5.6 The PICPs comparison of lower bound highest bid prediction. 75

5.7 The MORPs comparison of lower bound highest bid prediction. 76

5.8 The expected revenue comparison of lower bound highest bid prediction. 77

5.9 PICPs by varying the risk level α. 78

5.10 MORPs by varying the risk level α. 79

5.11 Expected revenue by varying the risk level α. 80

CHAPTER 1

INTRODUCTION

1.1 Motivation

Traditional advertising advertises a business through the use of newspaper advertisements, flyers, radio advertisements, and the yellow pages. However, compared with traditional methods, online advertising is a more efficient and cost-effective marketing solution for, especially, small businesses. The advantages of online advertising include reduced cost, measurability, targeting, and brand engagement.

Two main components in online advertising are search advertising and display advertising. Search advertising is a method of placing online advertisements on web pages that show results from search engine queries. Search ads are targeted to match key search terms entered on search engines. Advertisers are charged once their ads are clicked by users. In display advertising, advertisers (e.g., Volkswagen) pay publishers (e.g., Forbes) for showing banners, videos, or text on their webpages. Figure 1.1 is an example display ad. One display of an ad in a page view is called an *ad impression*.

The revenue of online display advertising in the U.S. is projected to be 7.9 billion U.S. dollars by 2022, and it has become the most critical revenue source for online publishers [15].

There are two channels to sell display ads: direct sale and real-time bidding. In direct sale, publishers make offline contracts with advertisers. Advertisers may set up their own impression volumes and target requirements. For example, an advertiser asks a publisher for 100,000 ad impressions, which are shown to male visitors in California within one month. In this case, the publishers have to forecast their future traffic and visitor profiles. Advertisers' costs are already finalized in the contracts. Publishers know the price of each ad beforehand.

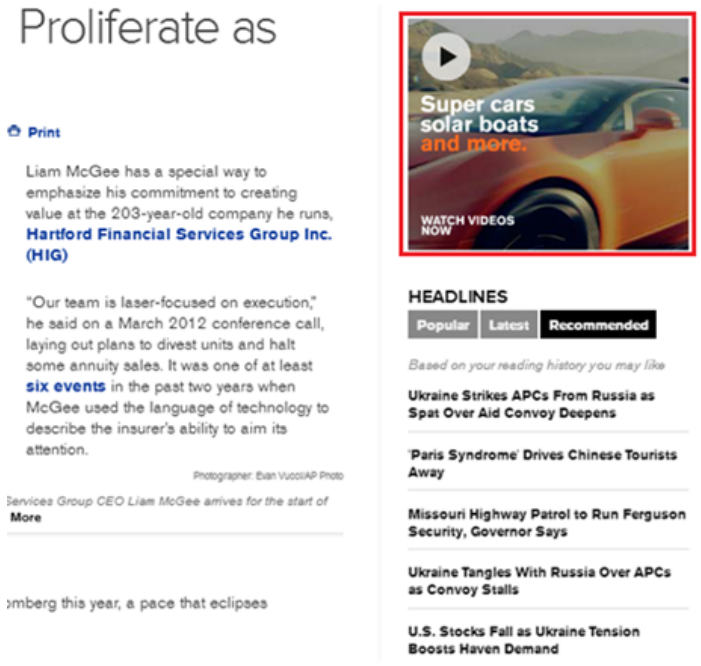

Figure 1.1 An example of a display ad.

The remaining inventory can be sold through auctions in the real-time. In other words, when a user clicks the link of a page. Ad impressions on the coming pages can be auctioned immediately, then the winning ads are returned and shown to the user. Both advertisers and publishers enjoy the benefits of real-time bidding. They provide publishers with a highly effective way to sell their 10% – 60% remnant inventory, help them cut down on the time and money associated with selling inventory to advertisers, and also offer a range of services, such as lead optimization and customer support. Advertisers benefit by receiving high cost savings and an easier, more effective way to connect with audiences on a much larger scale [60]. Globally, the real-time bidding (RTB) market is expected to grow to $18.56 billion by 2023 at the compound annual growth rate of 30.8% from 2018 to 2023 [3].

There are two main types of auctions: second-price auction and first-price auction. In first-price auction, the winning advertiser who bids the highest is charged by what it bid. In contrast, the second-price auction gives the winner a chance to pay

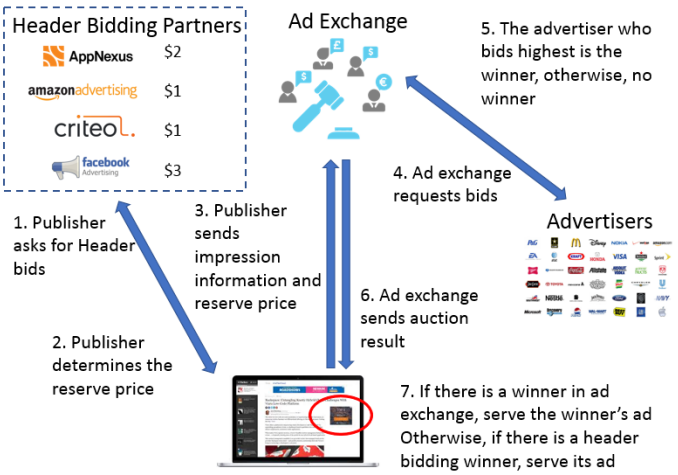

Figure 1.2 Ad impression selling in RTB in second-price auctions.

a little less than their original submitted offer. Instead of having to pay the full price, the winning bidder pays the price offered by the second-highest bidder plus \$0.01. According to existing studies [14], second-price auction gives bidders an incentive to bid their true value, i.e., it induces truthful bidding. The optimal strategy in a second price auction is to give bids that advertisers themselves value the given impressions.

However, recently, with the complexity of the RTB system, a lack of transparency is commonly cited as one of the main woes of programmatic RTB. Because of certain inconsistencies in the ways various exchanges manage auctions, there is a popular trend toward first-price auctions as an alternative, in a bid to create an even playing field for everyone, including advertisers, publishers, networks, ad exchanges, demand-side platforms (DSPs), and supply-side platforms (SSPs).

Figure 1.2 (steps 2-7) shows the basic process of ad impression selling (step 1 is an improvement, header bidding, that will be discussed later). When a user requests a webpage on a publisher's website, a *page view* is displayed on a screen.

To trigger real-time bidding in ad exchanges, for each ad impression, the publisher first determines a *reserve price*, the lowest price acceptable for an ad

impression (step 2). Then, the publisher sends the impression information (e.g. user profile, page metadata) and the reserve price to an ad exchange (step 3).

An ad exchange, denoted as *adx* in the rest of this research, is a digital marketplace that enables advertisers and publishers to buy and sell advertising space, often through real-time auctions. Adx requests bids from advertisers (step 4). Currently, most RTB ad exchanges use the second-price auction model (step 5), where the highest bidder wins if the bidding price is higher than the reserve price. The price paid is the higher between the second-highest bid and the reserve price [43]. In the first-price auction that may become common in the future, the advertiser who bids highest and higher than the reserve price still win the ad impression. However, the winner pays its own bid, instead of the second highest bid. If there is no bidding price higher than the reserve price, then the auction fails. If there is a winner in the auction (step 6), the winner's ad is shown as an ad impression.

Before triggering real-time bidding in ad exchanges, some publishers select to obtain some bids from header bidding first (step 1). The most obvious advantage of header bidding for publishers is increased advertising revenue. Publishers may increase their yield by 10% by adding just a single header bid source [11]. Header bidding allows the publisher to make the inventory available to multiple demand partners (ad networks, ad exchanges, and DSPs) at the same time. Greater number of advertisers participate in the auction automatically increases the probability that impressions are successfully monetized. Header bidding is similar to real-time bidding: An impression is sent to multiple header bidding partners. Each header bidding partner conducts one auction which is similar to the one occurs in the ad exchange later. Header bidding partners adopt first-price auctions. Thus, the advertiser who bids the highest in each header bidding auction is selected. Each header bidding partner returns its own highest bid. The publisher picks the highest winning bid across the header bidding partners.

In both second-price and first-price auctions, publishers have little control on the final revenue: advertisers determine whether and how much they are going to bid. The only thing that publishers can determine is the reserve price. Reserve price can impact the ad revenue for each impression. In second price auctions, since the impression revenue is the maximum of the reserve price and the second highest bid, increasing the reserve price can potentially maximize the impression revenue. However, a high revenue price that is more than the highest bid makes the impression unsold, in which case the impression revenue is zero. Likewise, in first price auctions, if the reserve price is the highest among the bids of all advertisers, the impression revenue is zero. Therefore, it is non-trivial to set reserve prices in both second-price and first-price auctions. A non-optimal reserve price may minimize the revenue of a publisher.

Therefore, it is significant to design algorithms for publishers to automatically set up a reserve price for a given impression in the real-time. The algorithms should be able to optimize publishers' revenue and minimize the auction failure rate (i.e., the chance that impressions are unsold).

The research is highly challenging: First, in second price auctions, the highest bids in ad exchange auctions are not accessible on the publisher's side. Publishers do not know the highest price advertisers would be willing to pay for each impression. Thus, it is difficult to learn the affordability of advertisers from historical data. Second, in both second-price and first-price auctions, advertisers utilize a large scale of user data to determine their bids. Advertisers buy user data from third-party companies and/or collect by themselves. On the other hand, publishers do not have much user data. The only information that publishers know about users are city-level geo locations, IP address, browser information, and so on. Furthermore, most publishers, especially online media, do not require user to log in. In this case, they can identify users only using cookies IDs, which is unreliable and expirable.

A user can delete cookies from the browser anytime. Thus, publishers do not have long browsing history of users even on their own websites. Third, few studies have been done in reserve price optimization. It is unknown whether machine learning algorithms can help determine optimal reserve price for individual impressions in the real-time.

1.2 Overview of The Research

The research studies the problem of reserve price optimization in online display advertising. The goal is to design machine learning algorithms that can automatically determine optimal reserve prices for specific impressions and thus optimize publishers' ad revenue.

The research looks into both second- and first-price auctions. As the main auction strategy as of 2019, second-price auctions are first studied: In Chapter 3, a parametric survival model has been proposed to predict the failure rate of a reserve price of a given impression (i.e., the probability that no advertisers will outbid the reserve price for the impression). If a reserve price is failed to be outbid, the impression revenue is zero. Thus, the failure rate indicates the risk of the reserve price of the impression. The outcome can help publishers estimate whether the reserve price that they determined for an impression is suitable. In Chapter 4, the proposed failure rate prediction model is then augmented by a deep neural network which capture more complicated joint effect between features. With the whole industry switching to first-price auctions, a reserve price optimization model is proposed in Chapter 5 to help publishers determine optimal reserve prices in real-time.

1.3 Organization of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 provides a review of the literature related to this study. Chapter 3 present the complete work on reserve

price failure rate prediction. Given an impression and a reserve price, the proposed algorithms estimate the probability that no advertisers will outbid that reserve price. A parametric survival model with pairwise interaction tensor factorization and header bidding regularization is proposed. Chapter 4 shows an extended work that further unleash the potential of the proposed prediction framework. It proposes to replace the factorization machines with a deep neural network. With the evolution of real-time bidding, first-price auctions become the main ad selling mechanism replacing second-price auctions. Chapter 5 introduces reserve price optimization in first price auctions. The study propose an approach to estimate the lower bounds of the highest bids from historical censored dataset.

CHAPTER 2

LITERATURE REVIEW

2.1 Reserve Price Optimization

As a significant problem for online publishers, reserve price optimization has been studied in the past few years. Almost all existing related studies focus on second-price auctions, as it is the main auction strategy before 2020. The largest difficulty of reserve price optimization in second-prince auctions is that the highest bids of historical auctions are unknown, i.e., censored, on the publishers' side. Thus, it is not possible for publishers to learn advertisers' affordability directly from historical data.

Cui et al. [13] described a bid landscape forecasting system in non-guaranteed delivery marketplace for any advertiser campaign specified by a variety of targeting attributes. Given a set of targeting attributes, the proposed gradient boosting decision trees can predict the winning bid for a impression. However, in many cases, target attributes of an ad campaign are inaccessible to publishers. Publishers also have no ideas about a user's attribute values, especially personally identifiable information, due to privacy issues. Li et al. [40] evaluated the impact of reserve price on publisher revenue in real-time bidding. Yuan et al. [72] proposed a simple game-theoretic-based approach to obtain an optimal reserve price on the publishers' side. Mohri et al. [46] assumed that both the highest bids and the second highest bids are observed by publishers, and then proposed a machine learning approach to optimize publishers' real-time bidding revenue. Cesa-Bianchi et al. [8] showed a regret minimization algorithm for setting the reserve price in online second-price auctions. Austin et al. [5] described a scalable linear-function-based reserve price optimizer for real-time bidding. Medina et al. [48] assumed that bids are observed by publishers, and directly optimize the revenue by reducing reserve price optimization to the standard setting

of prediction under squared loss, which can be conveniently minimized. Xie et al. [70] came up with an efficient method of improving the publisher revenue by mainly focusing on adjusting the reserve price for the high-value impressions. Jauvion et al. [24] proposed a new online learning algorithm based on classical multi-armed bandit strategies. However, these existing studies assume that publishers know the highest winning bids and the second highest bids of historical impressions, which is not the case for most publishers. Most of them evaluate their approaches using either synthetic data (e.g., [46] and [48]) or ad exchange data (e.g., [5]). These approaches do not consider data censorship, i.e., publishers do not know all the information about RTB auctions. In contrast, this research addresses the reserve price optimization of second-price auctions based on censored data available to most publishers. Furthermore, our evaluation is done using real-life data from a large publisher.

Existing related work that takes censorship into account may have datasets that contain different combinations of right-, left-, and/or uncensored data, due to different practices and platforms used among publishers. Alcobendas et al. [4] proposed a game-theoretic-based model to optimize reserve prices in the context of online video advertising with left- and uncensored data. The model considers information about auctions (e.g., the number of bids higher than the reserve price) as input. Such information is typically only available for the publishers that own ad exchanges (e.g., Google and Yahoo). Wu et al. [69] studied how to predict the winning price such that the advertiser/demand-side platform (DSP) can win the bid by placing a proper bidding value in the real-time bidding (RTB) auction. To achieve this, the authors proposed a censored-regression-based mixture model which is deployed on the advertisers' side. The authors considered their data as a partially observed dataset, which contains right-, left-, and un-censored data. In other words, in some cases, the exact winning price is observed. In one component of the proposed method, the

authors used simple logistic regression to predict winning rate (i.e., the probability that an advertiser can win an impression with a bid price). Wu et al. [68] proposed a deep learning model that predict the winning price on the advertisers/DSPs' side. However, the proposed model assumes that the exact winning bids of the won impressions are observed. Also, it cannot output a probabilistic prediction. Chahuara et al. [9] described an engine to optimize web publisher revenue from second-price auctions. The authors first adopted a relatively simple non-parametric regression model of auction revenue based on an incremental time-weighted matrix factorization. , which implicitly builds adaptive users' and ad placements' profiles. The authors used an online extension of the Aalen's Additive model to estimate the first and second bids' distribution. This method cannot handle left- and right-censored data at the same time. In addition, the method has to discretize reserve prices due to the limitation of the Aalen's Additive model. In practice, feature binning is tricky and makes the model inflexible, as it is hard to determine the boundary of bins. Also, the method considers only user IDs and placement IDs. Thus, the model cannot make predictions for infrequent users [38].

In addition to reserve price optimization, another family of related work is the prediction of winning prices at the advertisers' side [13, 69, 75, 68]. These studies predict the adx bid of a given impression for an advertiser or a Demand-Side Platform (DSP) to win the ad opportunity. Reserve price optimization and RTB winning price prediction share a common feature: both evaluate the values of impressions. The algorithms in existing studies on winning price prediction do not make probabilistic predictions. Also, advertisers and DSPs typically know the highest bids (i.e., their own bids) when they win the impressions. Most publishers are not able to access the highest adx bids. This makes the problem more challenging.

Compared with the existing studies, this research makes prediction using data available at the publishers' side, which does not include details about the adx auctions.

The proposed methods are applicable to data with any type of censorship faced by a publisher. In addition, the proposed reserve price failure rate prediction model outputs probabilistic predictions.

2.2 Matrix Factorization

Matrix Factorization decomposes the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. Matrix Factorization algorithms factorize the user-item rating matrix as the product of two lower dimensional matrices, so that each user/item has a latent vector representing the characteristics. The dot product of a user's latent vector and an item's latent vector is the predicted rating given by the user to the item. The most fundamental matrix factorization in recommender systems is Singular Value Decomposition (SVD) [55] and its variation SVD++ [37]. As a more general-purpose supervised learning algorithm, factorization machines (FMs) [53] have been widely adopted to model the discrete feature interactions in recommender systems. FMs combine the high-prediction accuracy of factorization models with the flexibility of feature engineering. The core part of FMs is pairwise interaction tensor factorization (PITF) [54]. PITF uses matrix factorization to capture the interaction of pairwise features. FMs have been used in many applications and studies, such as user behavior prediction [62, 64] and click-through rate prediction [27].

In this work, we use use FMs, which learn latent vectors for users, pages, and ad placements from training data, to predict the scale parameter of a statistical distribution. Furthermore, we also propose a technique based on deep learning to better capture the feature interaction.

2.3 Feature Interaction in Deep Learning

We borrow ideas on capturing feature interactions from recommender systems studies. In recommender systems, feature interactions take various forms, depending on applications, such as user-product, user-content, and query-answer. Deep neural networks have been extensively used in recommender systems [7, 47]. There are several main methods to capture feature interaction: concatenation, multiplication, and distance function. Some studies [12, 20, 65, 10] concatenate low dimensional feature representations together and feed the results into the next layers, e.g., fully connected layers. Other studies use element-wise product of vectors [6, 19]. A few other studies use distance functions (e.g., cosine distance) [71, 22]. Model selection usually depends on performance and specific applications. In this work, we adopt concatenation due to simplicity and fast convergence.

2.4 Prediction Interval Estimation

Using the notation used in [33], machine learning based prediction often models prediction targets by

$$t_i = y_i + \epsilon_i \tag{2.1}$$

where t_i is the ith prediction target. ϵ_i is the error results from noise. The errors are assumed to be independently and identically distributed. The true regression y_i is often approximated by the model prediction $\hat{y}_i$, which is output by a prediction model. Thus, we get

$$t_i - \hat{y}_i = y_i - \hat{y}_i + \epsilon_i \tag{2.2}$$

Confidence intervals (CIs) measure the variance of $y_i - \hat{y}_i$, which is the uncertainty between the model prediction $\hat{y}_i$ and the true regressions y_i, i.e., $y_i - \hat{y}_i$.

In comparison, prediction intervals (PIs) reflect the uncertainty associated with the different between prediction target t_i and the model prediction $\hat{y}_i$, i.e., $t_i - \hat{y}_i$. Therefore, while a CI describes the uncertainty in the prediction of an unknown but fixed value, a PI measures the uncertainty in the prediction of a future realization of a random variable [45]. According to Section 2.2, PIs account for more sources of uncertainty (model misspecification and noise variance) than CIs which only capture model misspecification [21].

Several categories of methods have already been proposed for PI construction in the literature.

The Delta method is a strategy for constructing intervals through nonlinear regression. The Delta method linearizes a neural network model by learning optimal parameters. It approaches optimal parameters by minimizing the error-based cost function, sum square error (SSE). The Delta method is more computationally demanding in the offline training phase, especially the Jacobian matrix calculations and the estimation of the variance of the error term. Delta method has been used for the construction of prediction intervals in many applications, such as travel time prediction [29], uncertainty prediction in Adaptive Neuro Fuzzy Inference System (ANFIS) [31], and short simulation or rare event applications [56].

The Bayesian method assumes that the set of neural network parameters is a random set of variables with assumed a priori distributions. The predictive distribution of network outputs is then evaluated using the posterior distribution. The purpose of neural network training is to maximize the posterior probability of the neural network parameters. Generally, Bayesian methods can better alleviate overfitting. However, it is also computationally expensive because the cost function involves Hessian matrix. The Bayesian method has been applied in several applications [58, 61, 39].

The MVE method assumes that errors are normally distributed around the true mean of targets. The goal of the MVE method is to estimate the parameters of the distribution, i.e., mean and variance. Unlike Delta and Bayesian methods, the MVE method estimates the target variance using a dedicated neural networks. This enhances the flexibility for estimating the heteroscedastic variance of the targets. MVE is also very simple and computationally inexpensive. Its performance highly relies on the assumption that the error term obeys a normal distribution with the expectation of the true target [74]. In addition, due to the weak generalization ability of neural networks, MVE may underestimate or overestimate the actual prediction intervals. Example applications of the MVE method includes [30]

Bootstrap method ensemble a set of neural network models to output a less biased estimation of the true regression of the targets [21]. It provides reliable solutions to obtain the predictive distribution of the output variables in neural networks [44]. The Bootstrap method estimates the targets by averaging the predictions of all neural networks. It estimates the variance by calculating an unbiased variance based on the outcomes of all neural networks. Since it has to build N neural networks, it is computationally more expensive in the training phase. Given trained models, it only needs to make N point estimates in the online prediction phase, which is much less costly than the Delta and Bayesian techniques. The Bootstrap method so far is the most commonly-used one among the four in the literature for the construction of PIs [51, 59, 51, 16, 25].

The main strategy of the above four traditional PI construction methods is to minimize the prediction error, instead of trying to improve the PI quality [32]. Thus, the output PIs may not be optimal in terms of width and coverage: a high-quality PIs should be as narrow as possible and capture some specific proportion of data points. To overcome this issue, Khosravi et al. [32] propose a lower upper bound estimation method (LUBE) which builds a neural network with two outputs (i.e.,

lower and upper bound) for estimating the high-quality prediction interval. It is achieved through minimizing a PI-based loss function, which optimizes both PI width and coverage.

CHAPTER 3

RESERVE PRICE FAILURE RATE PREDICTION IN SECOND-PRICE AUCTIONS USING FACTORIZATION MACHINES

This chapter presents the completed work on reserve price failure rate prediction in second-price auctions. A survival-analysis based machine learning algorithm is proposed to predict how likely it is that a reserve price of a specific impression will not be beaten in a following second-price auction.

3.1 Background and Motivation

In the second-price auction, an effective strategy of setting the reserve price can significantly increase publishers' revenue. Figure 3.1 illustrates the three RTB cases, based on the relationship among the reserve price r, the highest bid b_1, the second highest bid b_2, and the revenue generated.

- In Case 1, $r < b_2 < b_1$, the revenue is b_2.
- In Case 2, the publisher increases the reserve price r such that $b_2 < r < b_1$; the revenue will be r, which gives a revenue boost of $r - b_2$ compared to case 1.
- In Case 3, $b_1 < r$, the publisher increases r too much, such that no advertisers in the auction are willing to bid more than r; the publisher obtains no revenue.

The objective of publishers is to set r as close as possible to b_1 but never higher than b_1. Then, the revenue will be boosted to nearly b_1. However, since publishers do not know a priori b_1 before an adx auction, it is difficult to set an optimal r.

To solve the problem of accurately predicting the reserve price value, we leverage *header bidding*, a recent improvement to the basic RTB process. Before selling impressions in ad exchanges, publishers send impression information to multiple *header bidding partners* to conduct an impression header auction. Header auctions

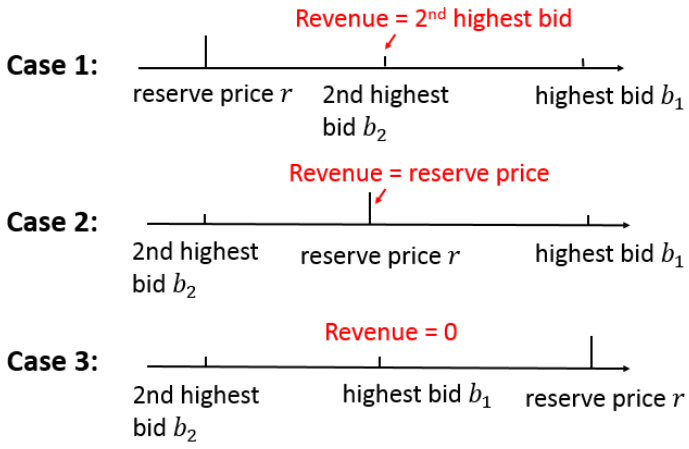

Figure 3.1 Impression revenue in the second-price auctions.

adopt first-price auction, instead of second-price auction as in RTB [14]. Without header bidding, advertisers can only bid leftovers after more premium channels, e.g., sponsored lines. With header bidding, advertisers have the benefit of looking first at the entire ad inventory: When a page loads, header bidding partners are called for all impressions in the pageview. Publishers can have more transparency into how much their impressions are worth and, thus, design adjust the reserve prices to increase revenue.

A simple strategy to benefit from header bidding information when setting the reserve prices is to always set r to h_1, the highest bidding price in header bidding (if available). In the example in Figure 1.2, the publisher can set $r = \$3$. If the publisher receives a higher winning bid in the RTB auction, the impression goes to the RTB winner; otherwise, it goes to the header bidding winner. If neither RTB nor header bidding has a winner, it triggers for a house ad or unfilled impression. This rarely happens in reality. Hence, no matter which case in Figure 3.1 occurs, the revenue is guaranteed to be $max(h_1, b_2)$. However, this is not optimal: it can be improved by predicting a better reserve price r', such that $b_2 \leq r' \leq b_1$. Thus, the expected revenue can be further boosted to $max(h_1, r')$. On the other hand, increasing r' increases the chance that r' fails to be outbid in adx and can be risky.

The goal of this chapter is to propose effective machine learning models for *r*eserve price failure rate prediction: Given information about an ad impression and a reserve price, the model outputs the probability that no advertisers in the RTB ad exchange will outbid the reserve price. The outcome of our approach has managerial implications for publishers to set appropriate reserve prices to maximize the expected revenue. It can also help publishers to find a balance between taking risks and maximizing revenue. For example, a conservative publisher may prefer to set a reserve price with a 20% failure rate, while a risk lover may set it to 80%.

Reserve price failure rate prediction is challenging due to three reasons. First, publishers do not know the bidding prices offered by RTB advertisers in past auctions, making the prediction of bidding price (and hence of the reserve price) for a target ad impression very challenging. We term this challenge "censorship" because the publishers are "censored" from knowing the highest bidding price; they just know the revenue generated by each ad impression (i.e., the second highest bid). There are two types of censorship in publishers' impression transaction data: left-censoring, where the unobserved highest adx bid is less than the known reserve price; and right-censoring, where the unobserved highest adx bid is greater than the observed revenue. Second, publishers do not have access to personally identifiable information of users, and thus do not know about users as much as advertisers and Demand-side Platforms (i.e., systems that help advertisers to buy impressions in real time). Without user profile information, it is difficult for publishers to predict the advertisers' bidding price in order to set the reserve appropriately. Third, although intuitively header bidding information can help with RTB prediction, it is not clear how to utilize this information. To the best of our knowledge, there is no existing work using header bidding in RTB-related predictions.

To address the censorship challenge, we propose to use parametric survival models. Unlike binary regressions, which cannot handle datasets with both censored

and uncensored data, parametric survival models are more generic and can handle any dataset with or without uncensored data [36]. To deal with the challenge of limited user profile information, we use latent vectors to capture feature characteristics and add factorized pairwise interaction between users and pages in the objective function. For the header bidding challenge, we exploit the similarities between header bidding and RTB in ad exchange and propose to improve the prediction model using header bidding regularization.

The chapter presents empirical results of the proposed approach on a real dataset from Forbes Media, a large online publisher, which logs daily ad impression transaction data. The concordance index (C-Index) is the standard performance measure for model assessment in survival analysis. We develop a customized C-Index for datasets containing only left- and right-censored instances. The experiments show that our models with the Weibull distribution significantly outperform the baselines, i.e., a Kaplan-Meier model and a logistic regression with observed reserve price/revenue as the feature. Adding interaction factorization and header bidding regularization reduces log-loss compared with the best baseline by 67%.

3.2 Reserve Price Failure Rate Prediction

This section first introduces our real-life dataset (Subsection 3.2.1) and discusses the data censorship (Subsection 3.2.2). We then present our parametric survival model (Subsection 3.2.3) with factorized pairwise interactions (Subsection 3.2.4) and header bidding regularization (Subsection 3.2.5).

3.2.1 Real-Life Datasets

We use two datasets collected in one day in April 2018 at Forbes Media's website: 1) NetworkImpressions and 2) NetworkBackfillImpressions [1]. These two datasets are

[1]The description and data samples are available at `https://support.google.com/admanager/answer/1733124` (Retrieval Date: July 23rd, 2021)

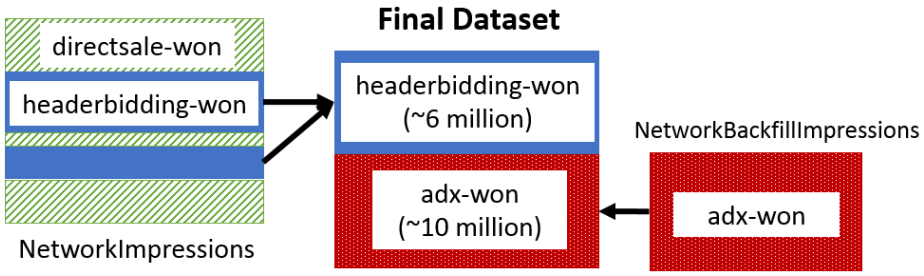

Figure 3.2 Data preparation.

provided by Google DoubleClick for Publishers (DFP) [1]. The data preparation is illustrated in Figure 3.2. The final dataset contains above 16 million impressions with 2.6 million unique users and 132 thousand unique pages.

NetworkImpressions records the impressions which were allocated to direct sale or header bidding winners. Headerbidding-won impressions failed to receive higher bids during RTB than the highest header bids (which is the reserve price of the headerbidding-won impressions in our dataset). Unlike headerbidding-won, directsale-won impressions were not sent to RTB. They were never bid by advertisers. Therefore, we filter out direct sale impressions by their order IDs. We get nearly 6 million headerbidding-won impressions.

NetworkBackfillImpressions records the impressions which were allocated to real-time bidding winners. In the context of this chapter, it contains adx-won impressions. It includes fields such as time, pageID, ad position, channel, section, geo-location, and header bids [2]. In addition, it records the revenue and reserve price of each impression. We obtain about 10 million adx-won impressions.

Although our method is based on Google DFP datasets (which has a dominant market share), it can be adopted by a large majority of the other publishers as well. This is because DFP's competitors (e.g., OpenX [2]) provide similar datasets,

[2]The default data report does not contain header bids. This information was added by the publisher.

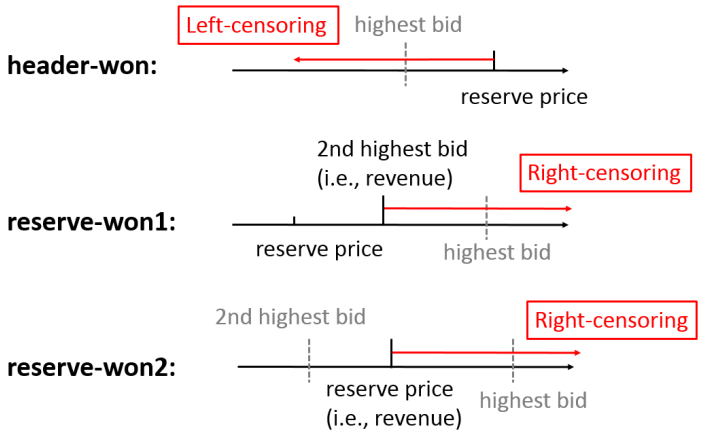

Figure 3.3 Three censorship cases.

and most publishers face the same restrictions/limitations in the datasets (e.g., censorship).

3.2.2 Data Censorship

The dataset has two types of impressions: adx-won and headerbidding-won. In the rest of the chapter, they are referred to as *reserve-won* and *header-won*, respectively. Without loss of generality, they are subdivided into three censorship cases in Figure 3.3.

header-won is similar to case 3 in Figure 3.1. The only difference is that, with header bidding available, the revenue is the maximum header bid. To increase revenue, a publisher wants to decrease the reserve price, so that it may be outbid in adx, and then the revenue will be the higher value between the revenue from adx and header bidding. Since the highest bid was not disclosed to publishers, the impressions are left-censored at the reserve prices.

reserve-won1 corresponds to case 1 in Figure 3.1. The reserve price and the revenue (i.e., the second highest bid) are known, but the highest bid is unknown. *Reserve-won1* impressions are right-censored at the second highest bid.

reserve-won2 corresponds to case 2 in Figure 3.1. Only the revenue (i.e. the reserve price) is known. That is, *reserve-won2* impressions are right-censored at the reserve price.

For both *Reserve-won1* and *reserve-won2*, to increase revenue, the publisher wants to increase the reserve price to be close to (but not exceed) the highest bid, which will then be the revenue [50].

3.2.3 The Proposed Parametric Survival Model

Problem Definition 1. *Given an ad impression A_i and a reserve price t_i, the goal is to predict the probability that no advertiser is willing to bid higher than t_i during RTB.*

Model Selection. Since our data are censored, it is natural to use survival analysis models [66]. The goal of survival analysis is to estimate the probability of a time-to-event of interest for a new instance with feature predictors. For instance, it can answer a question such as how likely it is that a patient has a disease at a certain point in time. To apply survival analysis to our case, we can make an analogy: one impression is an instance, which has a set of features. The event of interest is that all advertisers bid lower than the reserve price (i.e., reserve price failure). The time to event is the reserve price. With the increase in the reserve price (i.e., time to event), the probability of reserve price failure (i.e., event of interest) also increases. When the reserve price is \$0, it is most likely that the event of reserve price failure does not occur. If the reserve price is high (e.g., \$100), it can hardly receive a higher bid.

One important feature of survival analysis models is that they can output probabilistic predictions. In our application, they can both infer the optimal reserve price and tell how likely it is that a better bid can be received from advertisers. This is helpful for publishers that would like to perform revenue optimization and/or find the trade-off between risk and return.

Survival analysis models have been used in many applications [66]. As a widely used semi-parametric survival model, Cox proportional hazards model [36] assumes that the baseline hazard is a function of time to event t, but does not involve the feature predictors X. Thus, it is not necessary to specify the form of the baseline hazard (i.e., how the probability of event occurrence is changing with t).

Although the Cox proportional hazards model is popular due to its flexibility, it is insufficient for our application for several reasons. First, it does not directly accommodate left- or interval-censored data (it does handle right-censor data, though). Second, in model inference, partial likelihood requires observed data whose times to event are known. However, our dataset consists of both right- and left-censored data and does not contain any observed data, i.e., the highest adx bids of all impression are not accessible on the publishers' side. In addition, we need to deal with continuous time points because the reserve price is a continuous variable. Also, as the size of the impression data is huge, our model should be able to be learned stochastically.

The Proposed Method. We propose to use the parametric likelihood of a *parametric survival model* since it can easily accommodate left- and right-censored data or uncensored data. Furthermore, parametric survival models are more general compared to binary regressions, which cannot handle data containing both censored and uncensored instances [36].

A parametric survival model is one in which the outcome is assumed to follow a known distribution family. In other words, it assumes the probability that no advertiser bids higher than a reserve price follows a certain distribution. Commonly used distributions include Weibull, Exponential, and Log-logistic. Individual instances typically share the same family of distributions of similar form but with different parameters. Distribution parameter values are determined based on the feature predictors of instances.

Left- and right-censoring can be considered as special cases of interval-censoring (with zero as the lower bound and positive infinity as the upper bound). Hence, without loss of generality, given the ith ad impression, we aim to predict the probability that the reserve price t_i fails within the interval $[a_i, b_i]$.

$$P(a_i \leq t_i \leq b_i) = \int_{a_i}^{b_i} f(t)dt \tag{3.1}$$

where $P(a_i \leq t_i \leq b_i)$ is the probability that the true failure reserve price t_i is between a_i and b_i. $f(t)$ is the probability density function (PDF) of the specified distribution. a_i and b_i are the lower bound and the upper bound of the reserve price for the ith ad impression, respectively. Our particular cases are handled as follows (refer to Figure 3.3):

- For header-won impressions (i.e., left-censoring), a_i is zero and b_i is the historical reserve price because the true failure reserve price t_i must be less than the observed reserve price b_i. For the ith impression, we want to maximize the probability that a reserve price t_i fails within the interval $[0, b_i]$, i.e., $P(0 \leq t_i \leq b_i)$.
- For reserve-won impressions (i.e., right-censoring), a_i is the historical revenue and b_i is positive infinity because the true failure reserve price t_i must be higher than the observed revenue a_i. For the ith impression, we want to maximize the probability that a reserve price t_i fails within the interval $[a_i, +\infty]$, i.e., $P(a_i \leq t_i \leq +\infty)$.

If a publisher's data have uncensored impressions (i.e., the highest bids $b_i^{(1)}$ are observed), we can use $P(t_i = b_i^{(1)}) = f(t)$.

Taking Weibull distribution as an example, its PDF is $f(t) = \frac{\beta}{\alpha}\left(\frac{t}{\alpha}\right)^{\beta-1} e^{-\left(\frac{t}{\alpha}\right)^{\beta}}$, where α is the scale parameter and β is the shape parameter. Both α and β are positive. Typically, for parametric survival models, the shape parameter β is pre-specified and held fixed. The scale parameter α can be re-parameterized in terms of feature predictors X_i and regression coefficients W_i:

$$\alpha_i = w_0 + \sum_{j=1}^{m} w_j x_{ij} \tag{3.2}$$

where m is the number of impression features reflecting the user, the page, and the context, which are defined next.

Features. The features we consider come from several aspects:

- **User**: User information plays an important role when advertisers/DSPs are assessing ad opportunities. Most publishers do not have access to personally identifiable information. Hence, we model users by 1) user IDs, 2) state-level location, 3) operating system and Internet browser, 4) network bandwidth, and 5) devices.
- **Ad Placement**: The ad placement sizes and positions determine if the ads are visible and thus convertible [63]. A small ad placement at the bottom of the page may not receive high adx bids. We model ad placements by 1) ad unit size, e.g., "123x324" and 2) ad position (On the publisher's page template, each ad slot has a unique name representing its position).
- **Page**: Page information reflects user interests and the information that a user is looking for at the moment. It may also impact impression valuation. We model pages by 1) page URLs, 2) channels, e.g., "business" and "lifestyle," 3) sections, i.e., sub-channels, and 4) the trending status of the page (i.e., if the page is labeled as trending by the publishers' editors).
- **Context**: 1) hour of the day and 2) referrer URLs, i.e., which page the current page request originated from.

All of these features are constant over the reserve prices. In other words, no matter what the reserve price is, the feature values of a given impression do not change. Since these features are all categorical, we convert them to dummy variables.

Inference. The log-likelihood function is defined as:

$$\ln L = \sum_{i=1}^{N} [y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i)] \cdot r_i \tag{3.3}$$

where y_i is the ground truth of the ith impression. If the ith impression is a header-won impression, $y_i = 1$; otherwise, it is 0. $\hat{y}_i$ is the prediction of the reserve

price failure rate. The corresponding success rate can be calculated by $1 - \hat{y}_i$. r_i is the reserve price for header-won impressions and to the revenue (i.e., second highest bid) for adx-won impressions.

Given Equation (3.1), $\hat{y}_i$ is calculated by $\hat{y}_i = \int_0^{t_i} f(t)dt$. As Figure 3.3 shows, if the ith impression is header-won, t_i is its reserve price. If it is reserve-won, t_i is its revenue. $f(t)$ is the PDF of a distribution. The scale parameter α_i in $f(t)$ is calculated by Equation (3.2).

An example weight r_i is assigned to each training impression i to represent how important the impression is. In our application, publishers care more about high-value impressions [70]. For example, correctly predicting that an impression values \$5 can bring more revenue to a publisher than correctly predicting that one values \$0.05. Thus, r_i serves as a weight in Equation (3.3).

Therefore, to learn $\mathbf{w}$, we can minimize the negative log-likelihood, where the log-likelihood function is as below:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{N} [y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i)] \cdot r_i \right\} \tag{3.4}$$

3.2.4 Pairwise Interaction Tensor Factorization

To improve the prediction performance, we further consider the interactions between users and pages. Indeed, the adx bids on an impression may be jointly determined by the user, the page, and the ad placement. Matrix factorization-based predictive models, e.g., Factorization Machines [53, 64], have been used at the advertisers' side to optimize RTB [26]. Considering pairwise interactions between features using latent vectors captures better the features' characteristics and thus boosts the performance of reserve price failure rate prediction. It can also overcome data sparsity, which is especially challenging for interactions between users and pages.

We add a term of pairwise interaction tensor factorization to Equation (3.2).

$$\alpha_i = w_0 + \sum_{j=1}^{m} w_j x_{ij} + \sum_{j=1}^{m} \sum_{h=j+1}^{m} \langle \mathbf{v}_j, \mathbf{v}_h \rangle x_{ij} x_{ih} \tag{3.5}$$

where m is the number of feature predictors. $\mathbf{v}_j$ describes the jth feature with k factors. k is a hyperparameter that defines the dimensionality of the factorization. $\langle \mathbf{v}_j, \mathbf{v}_h \rangle$ is the dot product of two vectors of size k, i.e., $\langle \mathbf{v}_j, \mathbf{v}_h \rangle := \sum_{f=1}^{k} v_{j,f} \cdot v_{h,f}$.

To alleviate the data sparsity and the cold-start problem [41], we assign occasional users and infrequent pages, whose occurrences are less than 5 times, to dummy features "rare_user" and "rare_page", respectively. In our dataset, 39.04% users and 74.94% pages are involved in at least 5 impressions.

Adding factorization does not change substantially Equation (3.3) for log-likelihood. The only part that needs to be revised is how to compute the scale parameter α in the distribution PDF $f(t)$, i.e., replace Equation (3.2) with Equation (3.5).

Adding factorization increases the number of parameters that need to be learned from data, especially if k is large. Thus, we add an L2 penalty term, i.e., $\lambda_1 \|\mathbf{w}\|^2$ and $\lambda_2 \|\mathbf{v}\|^2$, at the end of Equation (3.4) to avoid overfitting [53]. This limits $\mathbf{w}$ and $\mathbf{v}$ from becoming extremely large or small. λ_1 and λ_2 are pre-specified parameters controlling the strength of regularization. In this case, the new loss function is shown in Equation (3.6).

$$\mathbf{w}^* = \underset{\mathbf{w},\mathbf{v}}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{N} [y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i)] \cdot r_i + \lambda_1 \|\mathbf{w}\|^2 + \lambda_2 \|\mathbf{v}\|^2 \right\} \quad (3.6)$$

3.2.5 Header Bidding Regularization

Before undergoing a traditional RTB auction, an ad impression is offered for sale in a header auction. In fact, the same advertiser may join both auctions. Therefore, it can be assumed that header bids are pseudo-randomly sampled from adx bids. The header bids that publishers receive from header bidding partners are the maximums in the corresponding sample groups. Since header bidding uses first-price auctions, the

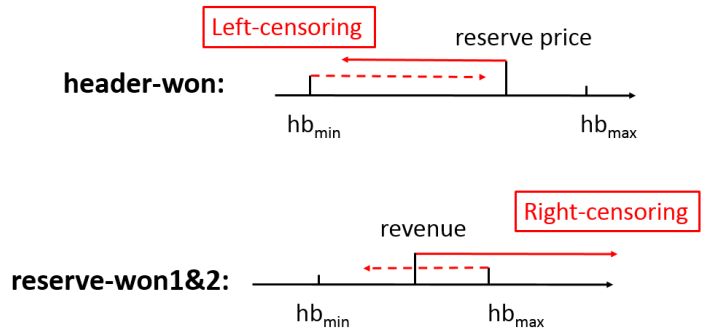

Figure 3.4 Two cases for header bidding regularization.

highest header bid in each header bidding partner is known, which can shed insights on advertiser's bids in the later RTB auctions.

This section discusses how to leverage the known bids in header bidding to improve reserve price failure rate prediction. One option is to add the winning header bids as feature vectors in the prediction model. However, in our dataset, the publisher receives winning header bids from five header bidding partners, who may not provide bids for every impression; this often happens due to network latency that prevents publishers from receiving header bids on time. Due to the presence of a substantial number of missing values, directly adding these bids into feature vectors may not work.

Instead, we propose to use header bidding information to regularize model learning. The intuition is shown in Figure 3.4. The minimum and the maximum header bids of an impression are first identified. They are denoted as hb_{min} and hb_{max}, respectively. For a header-won impression, its reserve price was set too high. The advertisers in the RTB auction were willing to pay less than the reserve price set by the publisher (i.e., left-censoring), while the advertisers in header bidding auctions were willing to pay at least hb_{min}. If hb_{min} is less than the reserve price, it can be assumed that likely advertisers in the RTB auction were also willing to pay more than hb_{min}. Thus, as Figure 3.4 shows, the header-won impression is strictly left-censored

at the reserve price, while the highest adx bid probably is more than hb_{min} (i.e., loosely right-censored at hb_{min}). For a reserve-won impression, its reserve price was set too low. The advertisers in the RTB auction were willing to pay more than the revenue received by the publisher (i.e., right-censoring), while the advertisers in header bidding auctions were willing to pay at most hb_{max}. If hb_{max} is less than the revenue, it can be assumed that likely advertisers in the RTB auction were willing to pay less than hb_{max}. Thus, as Figure 3.4 shows, the reserve-won impression is strictly right-censored at the revenue, while it is loosely left-censored at hb_{max}.

Thus, we add a term on header bidding regularization to the log-likelihood, i.e., from Equation (3.3) to Equation (3.7):

$$\begin{aligned}
\ln L = \sum_{i=1}^{N} & [y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i)] \cdot r_i \\
& + \lambda_3 \cdot I_{\{A_i \in H\}} \cdot \left[0 \cdot \ln \hat{y}_i^{min} + 1 \cdot \ln(1 - \hat{y}_i^{min})\right] \cdot r_i \\
& + \lambda_4 \cdot I_{\{A_i \in E\}} \cdot [1 \cdot \ln \hat{y}_i^{max} + 0 \cdot \ln(1 - \hat{y}_i^{max})] \cdot r_i \\
& = \sum_{i=1}^{N} [y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i)] \cdot r_i \\
& + \lambda_3 \cdot I_{\{A_i \in H\}} \cdot \ln(1 - \hat{y}_i^{min}) \cdot r_i + \lambda_4 \cdot I_{\{A_i \in E\}} \cdot \ln \hat{y}_i^{max} \cdot r_i \quad (3.7)
\end{aligned}$$

In Equation (3.7), the first term is the same as in Equation (3.3). $I_{\{A_i \in H\}}$ is an indicator variable, whose value is 1 if the impression A_i is in the header-won impressions H; otherwise, it is 0. $\hat{y}_i^{min}$ is the prediction at the minimum header bid of impression A_i (as shown in the first case of Figure 3.4). It is calculated by $\hat{y}_i^{min} = \int_0^{hb_{min}} f(t)dt$.

Since we assume adx advertisers are likely to bid more than hb_{min} (i.e., survive at hb_{min}), the closer $\hat{y}_i^{min}$ is to 0, the better. λ_3 is the strength of the regularization for header-won impressions. A large λ_3 encourages the model to predict correctly at

hb_{min}. $I_{\{A_i \in E\}}$ is an indicator variable, whose value is 1 if A_i is in the reserve-won impressions E; otherwise, it is 1. $\hat{y}_i^{max}$ is the prediction at the maximum header bid of A_i (as shown in the second case of Figure 3.4). It is calculated by $\hat{y}_i^{max} = \int_0^{hb_{max}} f(t)dt$.

Since we assume adx advertisers likely pay less than hb_{max}, the closer $\hat{y}_i^{max}$ is to 1, the better. They are all weighted by r_i. λ_4 is the strength of the regularization for reserve-won impressions. A large λ_4 encourages the model to predict correctly at hb_{max}.

The loss function is revised from Equation (3.6) to Equation (3.8).

$$\begin{aligned}\mathbf{w}^* = \underset{\mathbf{w},\mathbf{v}}{\operatorname{argmin}} - \Bigg\{ & \sum_{i=1}^{N} \left[y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i) \right] \cdot r_i \\ & + \lambda_3 \cdot I_{\{A_i \in H\}} \cdot \ln(1 - \hat{y}_i^{min}) \cdot r_i + \lambda_4 \cdot I_{\{A_i \in E\}} \cdot \ln \hat{y}_i^{max} \cdot r_i \\ & + \lambda_1 \left\| \mathbf{w} \right\|^2 + \lambda_2 \left\| \mathbf{v} \right\|^2 \Bigg\} \end{aligned} \tag{3.8}$$

3.3 Evaluation

3.3.1 Experimental Dataset and Ground Truth

The dataset has been described in Subsection 3.2.1. The dataset was collected over one day in April 2018 on Forbes.com. All header-won and reserve-won impressions are shuffled. Training data, validation data, and test data are randomly picked by 8:1:1. Nearly 13 million impressions are in the training data, and 1.6 million impressions are in the validation/test data.

Test impressions are also weighted by r_i, as it was already described for training data in Subsection 3.2.3. r_i is set to the reserve price if the ith impression is header-won or the revenue (second highest bid) if reserve-won.

We consider reserve price failure rate prediction as a classification problem. Our model outputs the probability that no advertisers will outbid a reserve price for an

impression. The *ground truth* is available in our dataset; it is known for each ad impression with a reserve price whether it is header-won or reserve-won.

3.3.2 Implementation

The proposed parametric survival model is implemented using Tensorflow. The experiments are run on a desktop with i7 3.60Hz CPU and 32GB RAM. The computation is sped up using NVIDIA GeForce GTX 1060 6G GPU. Running 5 epochs usually takes 5-6 hours depending on the parameter setting. In practice, the training phase can be offline. The prediction of an impression is done in less than 100ms, which demonstrates that the prediction phase can be deployed as an online process.

The training goal is to minimize the log-loss. Since the large training dataset does fit the memory, the optimizer we adopt is Stochastic Gradient Descent (SGD) with a learning rate of 10^{-3}.

Considering the training speed and memory consumption, we set the training batch size to 2048. Although using smaller batch sizes, in theory, can speed up convergence, it may also lead to longer training time for one epoch due to more I/O with the GPU. In this experiment, we find a batch size of 2048 is a good trade-off between convergence and training time for one epoch. The training process usually can converge at the second epoch.

To avoid overfitting, across all 10 epochs, the model that performs the best on the validation data is applied to the test data. The parameter values are empirically set: λ_1 and λ_2 (introduced in Subsection 3.2.4) are both set to 10^{-7}. They control the complexity of the feature weights, $\mathbf{w}$, and the complexity of the weights for feature latent vectors, $\mathbf{v}$. λ_4 (introduced in Subsection 3.2.5) and λ_5 are set to 10^{-4}. They control the strength of header bidding regularization for header-won and reserve-won impressions, respectively.

3.3.3 Experimental Metrics

Accuracy computes the percentage of the test instances correctly predicted at the reserve prices (for header-won impressions) or the revenues (for reserve-won impressions). Higher values are better.

Log-Loss is widely used in probabilistic classification. It penalizes a method more for being both confident and wrong. $logloss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$, where N is the number of test impressions. $\hat{y}_i$ is the probabilistic prediction and y_i is the ground truth (either 0 or 1). Lower values are better.

Concordance-Index (C-Index) [18] is a well-recognized measure of discrimination for models that predicts a time-to-event and equals the proportion of impression pairs in which the predicted event probability is higher for the subject who experienced the event of interest than that of the subject who did not.

The original C-Index requires uncensored instances. However, our dataset contains censored instances: either left- or right-censored. Figure 3.5 shows the four relationships of any pair of impressions, i.e., $< A_i, A_j >$. In the first case, A_i is left-censored (i.e., header-won), while A_j is right-censored (i.e., reserve-won). As it is known that A_i's reserve price t_i failed and A_j's reserve price t_j was outbid, we can expect that $1 \geqslant \hat{y}_i > \hat{y}_j \geqslant 0$, where $\hat{y}_i$ is the reserve price failure rate of A_i. The failure rates of $< A_i, A_j >$ are comparable. In the second case, both are left-censored because their reserve prices are higher than the (unobserved) highest adx bid prices.

The highest bid of A_i, $t_i{}'$, is within $[0, t_i)$. The highest bid of A_j, $t_j{}'$, is within $[0, t_j)$. In the case of $t_2{}' < t_1{}' < t_1 < t_2$, we may expect $0 \leqslant \hat{y}_i < \hat{y}_j \leqslant 1$ because $t_2 - t_2{}' > t_1 - t_1{}'$. In other words, $\hat{y}_i$ is farther from its highest bid price than $\hat{y}_j$. Thus, $\hat{y}_j$ should be closer to 1. However, in the case of $t_1{}' \ll t_1 < t_2{}' < t_2$, we expect $\hat{y}_i > \hat{y}_j$ because $t_2 - t_2 < t_1 - t_1{}'$. Since $t_1{}'$ and $t_2{}'$ are censored, The failure rates of $< A_i, A_j >$ is not comparable. Likewise, the third and the fourth cases are also not comparable on failure rates.

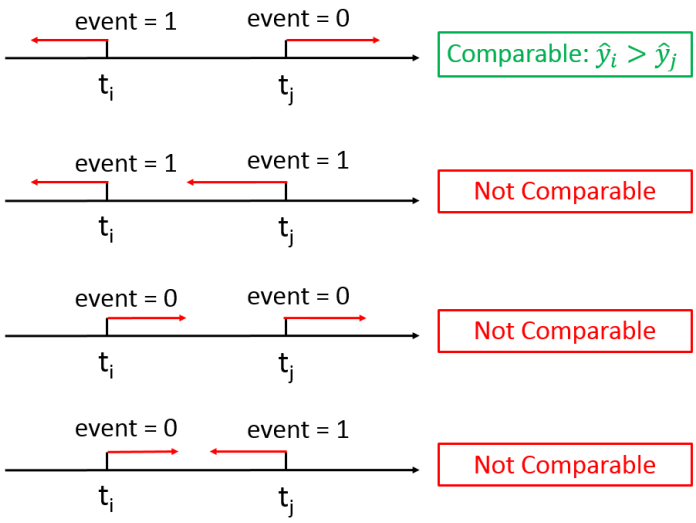

Figure 3.5 Pairwise relationships.

Thus, we create a customized C-Index based on the original one:

$$c\,index = \frac{1}{|\varepsilon|} \sum_{A_i \in L} \sum_{A_j \in R} I_{\{\hat{y}_i > \hat{y}_j\}} \tag{3.9}$$

where L and R are the set of left- and right-censored impressions, respectively. The customized C-Index only considers the first case, which is the only comparable one. Higher values are better.

This customized C-Index provides a measure that considers only the test instance pairs whose relationships are known. Along with log-loss, it is complementary for the data in which the highest bids are far away from the observed reserve prices/revenues, since the observed reserve prices/revenues are no longer reliable. Since the customized C-Index only considers one case (refer to Figure 3.5), it may be hacked by a naive method that always outputs low failure rates for large reserve prices and large failure rates for low reserve prices. Therefore, in the experiments, we evaluate the model using C-index along with log-loss and accuracy.

In our experiments, we ran each test three times and reported the averages.

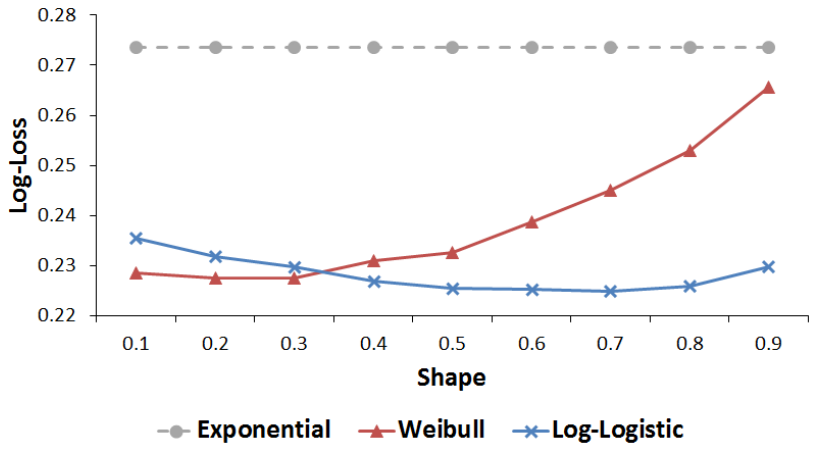

Figure 3.6 Log-loss of different distributions with different shapes.

3.3.4 Comparison Systems

Observed Reserve Price/Revenues (OR). The simplest way to predict failure rates is to use reserve prices as the only feature in the model. Thus, we build a logistic regression with one feature: reserve prices for header-won and revenues for reserve-won.

Kaplan-Meier (KM): Kaplan-Meier [36] is an extensively used non-parametric statistic used to estimate the survival function from lifetime data. We slightly modify it: $\hat{y}_i = 1 - \prod_{j:t_j \leq t_i} \left(1 - \frac{d_j}{n_j}\right)$, where t_i is the reserve price/revenue of the ith test instance. t_j is a price less than t_i. d_j is the number of impressions that failed to be sold at t_j. n_j is the number of impressions that did not fail at t_j.

3.3.5 Comparison of Different Distributions

The proposed parametric survival model requires the assumption of a distribution of the reserve price failure rate. The distribution type can impact the prediction performance. In practice, publishers can plug in commonly used distributions for survival analysis and pick the one with the best performance [36]. We evaluate the performance of Exponential, Weibull, and Log-Logistic distributions. Subsection 3.2.3 described how Weibull distribution is used. The PDFs of Exponential and Log-

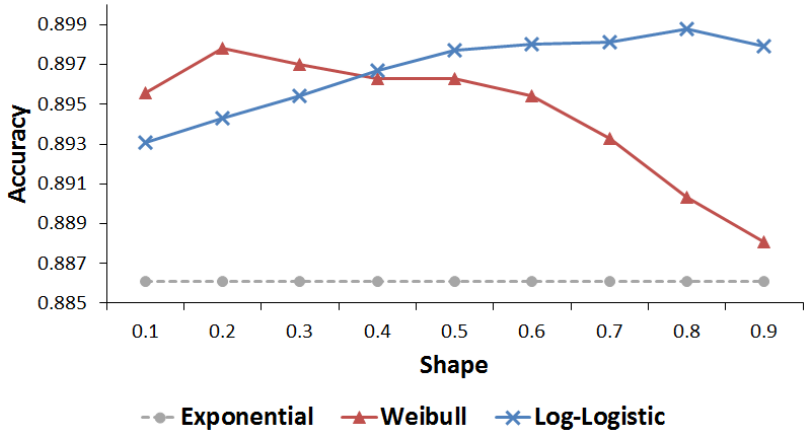

Figure 3.7 Accuracy of different distributions with different shapes.

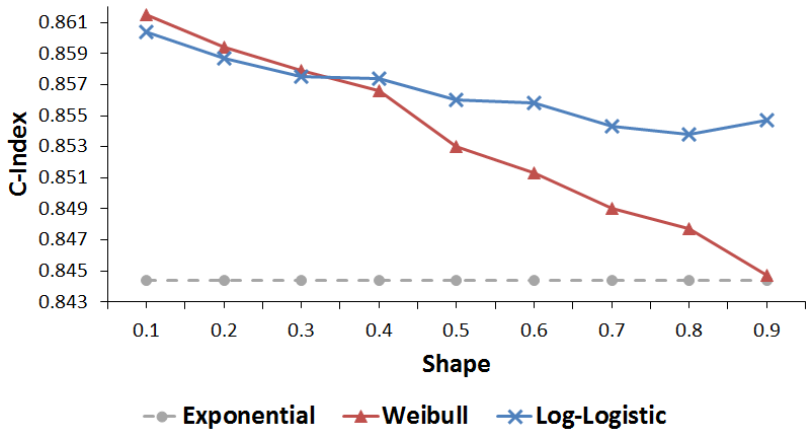

Figure 3.8 C-index of different distributions with different shapes.

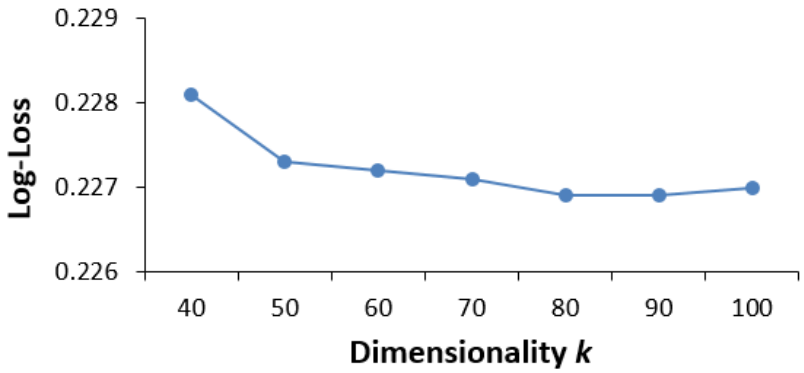

Figure 3.9 Log-loss of different distributions with different k.

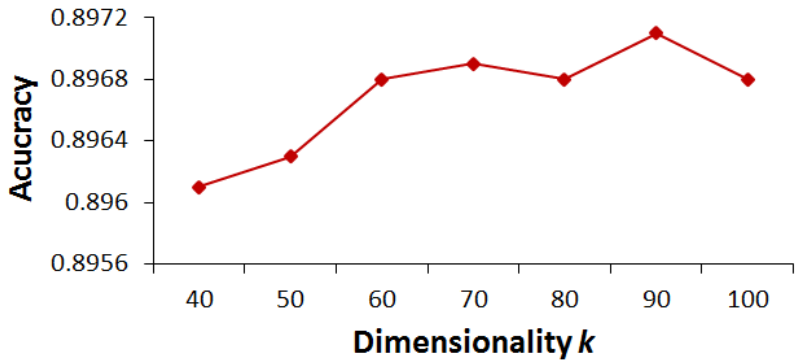

Figure 3.10 Accuracy of different distributions with different k.

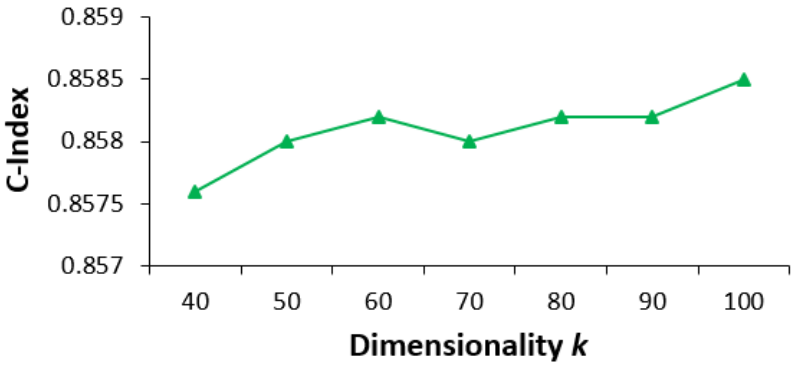

Figure 3.11 C-index of different distributions with different k.

Logistic distributions are $f(t) = \alpha e^{-\alpha t}$ and $f(t) = \left[\frac{\beta}{\alpha} \cdot (\frac{t}{\alpha})^{\beta-1}\right] / \left[(1 + (\frac{t}{\alpha})^{\beta})^2\right]$, respectively. The scale parameter, α, can be learned from the training data. It is re-parameterized in terms of feature predictors X_i and regression coefficients W_i, i.e., Equation (3.2). The optimal shape β, is found through experiments.

The performance of different distributions and different shape parameters is presented in Figures 3.6, 3.7, and 3.8. The performance is reported for the proposed model with factorization and header bidding regularization. As the Exponential distribution has no shape parameter, its performance does not change across different βs. As one can see, the performance of the Exponential distribution is not as good as Weibull and Log-Logistic.

The Log-Logistic obtains its lowest log-loss (0.2248) when $\beta = 0.7$ and the highest accuracy (0.8988) when $\beta = 0.8$. However, $\beta = 0.7$ and 0.8 make the model with the Log-Logistic misclassify more test instances of the "sure case" (case 1 in Figure 3.5). $\beta = 0.1$ leads to the highest C-Index, while its log-loss and accuracy are unsatisfactory: $\beta = 0.1$ tends to favor impressions with large reserve prices rather than those with low reserve prices (in terms of lowest failure rates).

The Weibull distribution obtains the lowest log-loss (0.2274) when $\beta = 0.3$, the highest accuracy (0.8978) when $\beta = 0.2$, and the highest C-Index (0.8615) when $\beta = 0.1$. The Weibull parametric survival model with a small β (e.g., 0.2) can obtain outstanding performance for all three metrics. Thus, in the rest of the experiments, we use the Weibull distribution with $\beta = 0.2$ as the model setting.

3.3.6 Comparison of Different Dimensionalities

In Subsection 3.2.4, each feature is represented by a k-length latent vector that carries characteristics of the feature. k is an important parameter which can significantly impact model performance. In theory, a larger k has better representation and results

in a more complex model; however, it has a higher chance to overfit. Dimentionality k is usually determined through experiments.

The results are shown in Figures 3.9, 3.10, and 3.11. They indicate that, with the increasing in vector dimensionality, the performance on the test data generally improves. The reason is that longer latent feature vectors can better capture the signals in the training data so as to improve model complexity. In addition, the performance of log-loss and accuracy increases fast when k grows from 40 to 60. After 60, the growth slows down and even reduces. This is because most signals in the training data have been captured, and further increasing k results in overfitting. C-Index also follows such a trend. We observe that $k = 100$ leads to a jump when compared with $k = 90$. The possible reason is that $k = 100$ may happen to correctly predict many test instances in the first case of Figure 3.5, which increases the customized C-Index. Finally, we notice that there is no single k that results in best performance for all three metrics: When $k = 80$, we get the lowest log-loss in the test data. $k = 90$ leads to the highest accuracy. $k = 100$ wins the C-Index test.

Publishers can select the k based on their objectives: Log-loss can make balance between ad revenue and failure risk. Accuracy maximizes the total number of impressions that are correctly classified. C-index can be used when the observed reserve prices and revenues are believed to be very off from the actual highest bids.

Since minimizing the overall log-loss is the training objective in Equation (3.8), we set $k = 80$ in the rest of the experiments.

3.3.7 Overall Comparison

We compare the proposed model with two baselines: *KM* and *OR*.

For our model, we use three versions: the plain parametric survival model *param-surv*, the model with interaction factorization *param-surv-f*, and the model

with both pairwise interaction factorization and header bidding regularization *param-surv-fhb*. All use Weibull distribution with $\beta = 0.2$.

Table 3.1 Comparison for All Impressions

	Log-Loss	Accuracy	C-Index
KM	1.5762	0.5495	0.0
OR	0.6883	0.5486	0.0
Param-surv	0.2425	0.8880	0.8532
Param-surv-f	0.2305	0.8953	0.8577
Param-surv-fhb	0.2266	0.8972	0.8583

Table 3.2 Comparison for Impressions with Header Bids Only

	Log-Loss	Accuracy	C-Index
KM	1.5766	0.5493	0.0
OR	0.6881	0.5487	0.0
Param-surv	0.2438	0.8879	0.8533
Param-surv-f	0.2311	0.8946	0.8573
Param-surv-fhb	0.2186	0.9011	0.8597

Table 3.1 presents the results. All three versions of our model outperform the two baselines. Among the three versions, *param-surv-fhb* is the best.

The C-Index values of the two baselines are all zero because: 1) KM makes predictions based on the percentages of impressions whose reserve prices that are less than a given price have already failed. As the overall failure rate increases with the increase of the reserve price, KM always "thinks" an impression with a higher reserve price has a higher failure rate (i.e., case 1 in Figure 3.5 never happens). 2) Likewise, OR is a linear model. It learns from the data that the failure rate is positively

correlated with the reserve price. It always gives a higher reserve price a greater failure rate than a lower reserve price.

The *param-surv* model has good performance and clearly outperforms the baselines. Adding interaction factorization, *param-surv-f* reduces the log-loss by 5% because latent feature vectors can better capture the regularities in the training data and overcome data sparsity compared to one-hot encoding. Furthermore, utilizing header bidding to regularize the model, *param-surv-fhb* can reduce the log-loss by an additional 1.7% from *param-surv-f*.

Header bidding regularization is only applicable on impressions with header bids. To fully present its effect, we filter out impressions without header bids from the test set. The results are shown in Table 3.2, and they demonstrate that *Param-surv-fhb* has a larger performance improvement compared with the other models.

3.4 Summary

This chapter proposes a parametric survival model to predict the failure rate of the reserve price of an online display ad impression in an ad exchange auction. The model is further augmented by user-page pairwise interaction tensor factorization and header bidding factorization. We also develop a customized C-Index for datasets containing only left- and right-censored instances. The experimental results show that the proposed models with the Weibull distribution significantly outperforms a Kaplan-Meier model and a logistic regression with observed reserve price/revenue as the feature. Adding factorized interaction and header bidding regularization further boost performance. Our model can be adopted by the majority of online publishers because similar data can be conveniently collected on most publishers' platforms.

CHAPTER 4

RESERVE PRICE FAILURE RATE PREDICTION IN SECOND-PRICE AUCTIONS USING DEEP NEURAL NETWORKS

4.1 Motivation

The previous chapter proposes a parametric survival analysis model that is powered by factorization machines (FM). Although FM models learn latent vectors to model input variables, it is still insufficient to capture the deep pattern among input variables. Hence, in this work, we replace the FM model with a deep neural network, which improves the end prediction performance. The deep neural network can be easily plugged into the previously proposed framework.

4.2 Reserve Price Failure Prediction

4.2.1 Deep Neural Network for Feature Interaction

Since the features introduced in Subsection 3.2.3 are mostly sparse inputs (i.e., categorical features with a large number of possible feature values, such as users, pages, and ad placements), we can also use a deep neural network (DNN) to infer scales, instead of the pairwise interaction tensor factorization proposed in Subsection 3.2.4. Unlike tensor factorization, our DNN technique captures the similarities of all features and better models the complex feature interactions. These two features are explained next.

In Subsection 3.2.3, the features, i.e., users, ad placements, pages, and context, are modeled by one-hot encoding. Subsection 3.2.4 proposes to use lower-dimensional dense representations (i.e., embeddings) to model individual pages and users. However, ad placements and context also need to be modeled by vectors: For instance, similar ad placements and referrer URLs may have similar impact on the scale and, thus, the final failure rate. Simply modeling them by one-hot encoding is insufficient

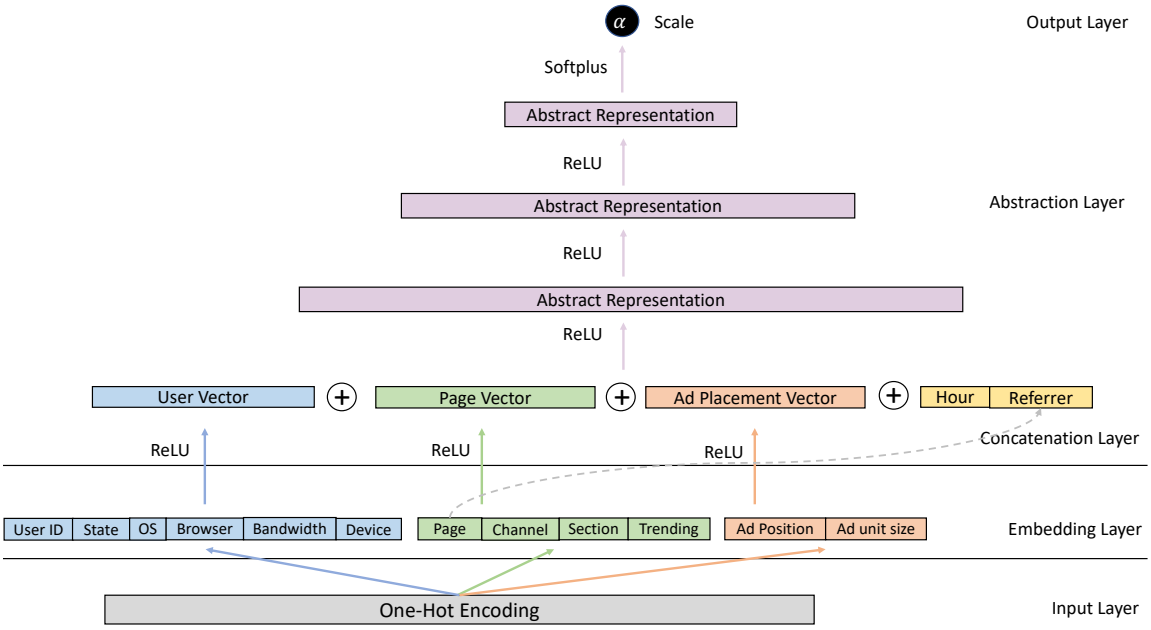

Figure 4.1 The architecture of the deep neural network.

to capture the similarities and their impact: features with similar contributions have similar coefficients, i.e., one scalar, which however are not able to fully capture the complex nature of the features. On the other hand, embeddings can capture the characteristics of features in a number of dimensions. Thus, features can be close to each other in some dimensions while different in other dimensions.

The pairwise interaction tensor factorization in Subsection 3.2.4 only captures pairwise interaction, e.g., users and pages. Neural networks allow all features to be modeled by embeddings, and use multiple layers of neurons to abstract the joint effects of all input features. The abstract representations of the interactions of input features are obtained through multiple layers of computation.

The DNN technique and the pairwise interaction tensor factorization in Subsection 3.2.4 achieve the same goal, despite their differences. When deploying the proposed model, practitioners can determine which of the two techniques to use through experiments and select a trade-off between prediction accuracy and training

efficiency. As it will be shown in Subsection 3.3, the DNN techniques achieves higher prediction accuracy, at the cost of slower training.

Figure 4.1 shows the architecture of our DNN technique.

Input Layer: All features at the input layer are represented by one-hot encoding, which is the same with what we did in Subsection 3.2.4. Note that on the Forbes webpages, the number of different ad placement shapes is finite due to a fixed number of page templates. Thus, we represent the ad unit size, e.g., "123x324", as a string feature.

Embedding Layer: The embedding layer takes one-hot vectors of input features, and it looks up the corresponding embedding matrix for the embeddings of the features. Since all input features are categorical, we use embeddings to represent all features, as proposed in [17]. The lengths of the embeddings are pre-specified parameters that can be tuned through experiments. The embeddings of the features representing the user, the page, and the ad placement are concatenated together, respectively. These features will be combined into unified vectors for the user, the page, and the ad placement by a Rectified Linear Unit (ReLU) layer. ReLU is an activation function that truncates its argument at zero. The negative part of the argument is forced to be zero, i.e., $f(x) = max(0, x)$, where x is the input argument of a ReLU. The advantages of ReLU include sparse activation, better gradient propagation, and efficient computation. Due to these advantages, it has been widely used in recent deep learning studies. Therefore, each neuron U_i in the user vector U is calculated by:

$$U_i = ReLU(\mathbf{w}_i \mathbf{x}_e^{(u)} + b_i) \tag{4.1}$$

where $\mathbf{w}_i$ is the weight vector for the ith neuron in the user vector, and $\mathbf{x}_e^{(u)}$ are the embeddings of the user features. The equations for the page vector and the ad placement vector are similar. b_i is the bias.

Concatenation Layer: This layer concatenates the user vector, the page vector, the ad placement vector, and the context information (i.e., the embedding of the hour of the day and the embedding on the referrer URL). To reduce the number of parameters that need to be learned from training data (thus improve training efficiency and alleviate overfitting), we share page embeddings with the referrer embeddings if the referrer is a Forbes page. Otherwise, the referrer embedding starts with a random value.

In addition to concatenation, another way to take into account feature interaction is to calculate the dot product of the user vector, the page vector, the ad placement vector, and the context vector. The output of the dot product can be the final scale prediction. However, through experiments, we see that it is hard to converge. The training error fluctuates even though a small learning rate is given. Also, it does not fit the data as well as concatenation.

Abstraction Layer: The concatenated vector is then further mapped to lower-dimension representations by multiple fully connected layers. Specifically, we use three layers with ReLU activation:

$$y_{hj} = ReLU(\sum_{f}^{\{u,p,a,c\}} \mathbf{w}_{hj}^{(f)} \mathbf{x}_{h-1}^{(f)} + b_{hj}) \tag{4.2}$$

where u, p, a, c are the vectors of the user, the page, the ad placement, and the context, respectively. $\mathbf{w}_{hj}^{(f)}$ is the weight vector for the jth neuron in the hth abstraction layer. $\mathbf{x}_{h-1}^{(f)}$ are the neurons of the previous layer. y_{hj} is the value of the jth neuron in the hth layer.

Output Layer: The range of the output of ReLU is $[0, +\infty)$. However, the range of the scale parameter in statistical distributions (e.g., Weibull distribution) must be $(0, +\infty)$. To predict scale, the output layer uses Softplus activation to convert the last abstraction layer to a positive scalar. Softplus is a smooth approximation to the rectifier. The analytic function is $f(x) = ln(1 + e^x)$. Thus, the range of the prediction $f(x)$ is $(0, +\infty)$.

The DNN technique predicts the scale parameter for the distribution, given impression information, and it is expected to perform better than the pairwise interaction tensor factorization because it leverages feature embeddings and multiple fully connected layers to capture sophisticated interactions between input features.

4.3 Evaluation

4.3.1 Comparison of Different Distributions

The proposed parametric survival model requires the assumption of a distribution of the reserve price failure rate. The distribution type can impact the prediction performance. In practice, publishers can plug in commonly-used distributions for survival analysis and pick the one with the best performance [36]. We evaluate the performance of Exponential, Weibull, and Log-Logistic distributions. Subsection 3.2.3 described how Weibull distribution is used. The PDFs of Exponential and Log-Logistic distributions are $f(t) = \alpha e^{-\alpha t}$ and $f(t) = \left[\frac{\beta}{\alpha} \cdot (\frac{t}{\alpha})^{\beta-1}\right] / \left[(1 + (\frac{t}{\alpha})^{\beta})^2\right]$, respectively. The scale parameter, α, can be learned from the training data. It is re-parameterized in terms of feature predictors X_i and regression coefficients W_i, i.e., Equation (3.2). The optimal shape β, is found through experiments.

The performance of different distributions and different shape parameters is presented in Figure 4.2, 4.3, and 4.4. The performance is reported for the proposed model with factorization and header bidding regularization (i.e., *ps-fhb*) and the model with DNN and header bidding regularization (i.e., *ps-nhb*). As the Exponential

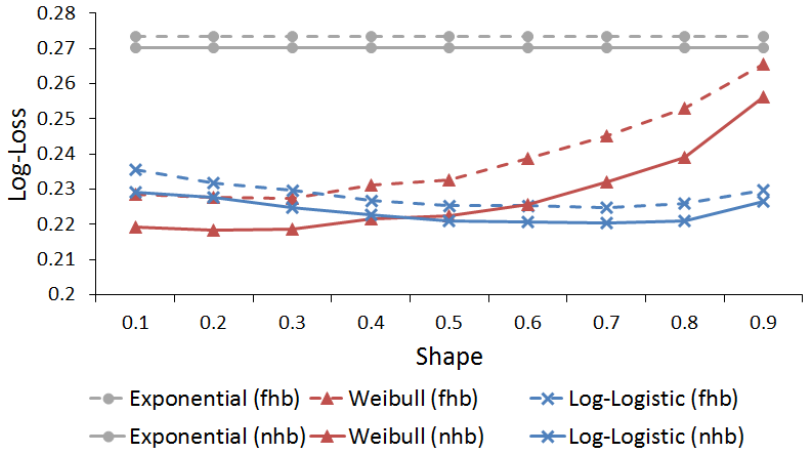

Figure 4.2 Log-loss of shapes .

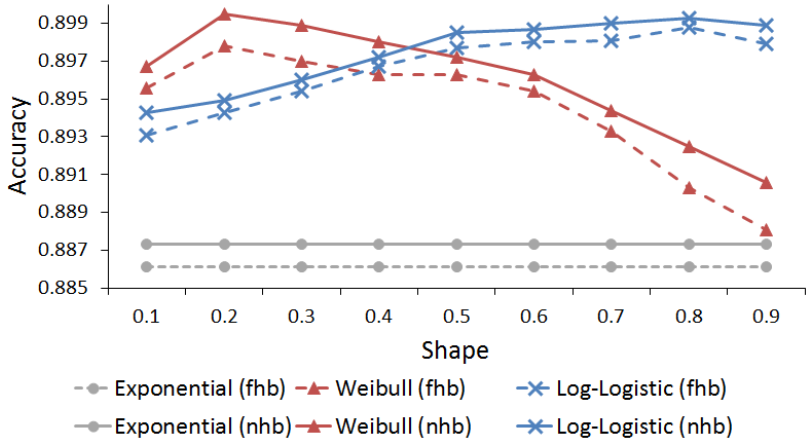

Figure 4.3 Accuracy of shapes.

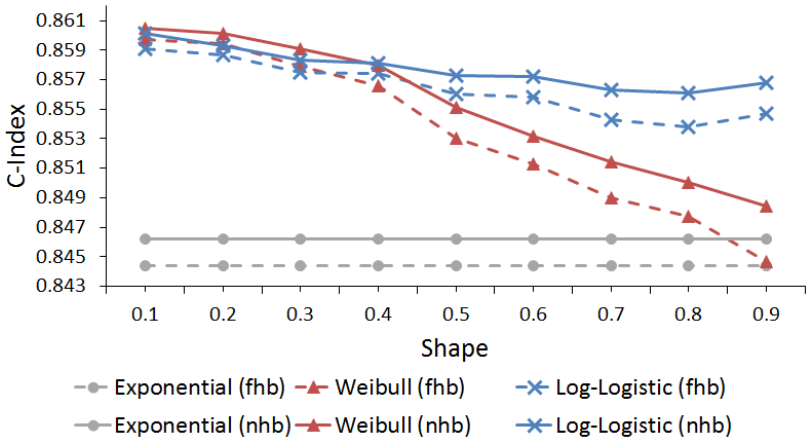

Figure 4.4 C-Index of shapes.

distribution has no shape parameter, the performance does not change across different βs. As one can see, the performance of the Exponential is not as good as Weibull and Log-Logistic.

With *ps-fhb*, the Log-Logistic obtains its lowest log-loss (0.2248) when $\beta = 0.7$ and the highest accuracy (0.8988) when $\beta = 0.8$. However, $\beta = 0.7$ and 0.8 make the model with the Log-Logistic misclassify more test instances of the "sure case" (case 1 in Figure 3.5). $\beta = 0.1$ leads to the highest C-Index, while its log-loss and accuracy are unsatisfactory: $\beta = 0.1$ tends to favor impressions with large reserve prices rather than those with low reserve prices (in terms of lowest failure rates). The Weibull distribution obtains the lowest log-loss (0.2274) when $\beta = 0.3$, the highest accuracy (0.8978) when $\beta = 0.2$, and the highest C-Index (0.8615) when $\beta = 0.1$. The Weibull parametric survival model with a small β (e.g., 0.2) can obtain outstanding performance for all three metrics. Thus, in the rest of the experiments, we use the Weibull distribution with $\beta = 0.2$ as the model setting.

With *s-nhb*, the Log-Logistic obtains its best log-loss (0.2203) when $\beta = 0.7$, the highest accuracy (0.8993) when $\beta = 0.8$, and the best C-Index (0.8601) when $\beta = 0.1$. The Weibull distribution obtains the lowest log-loss (0.2183) when $\beta = 0.2$, the highest accuracy (0.8995) when $\beta = 0.2$, and the highest C-Index (0.8605) when

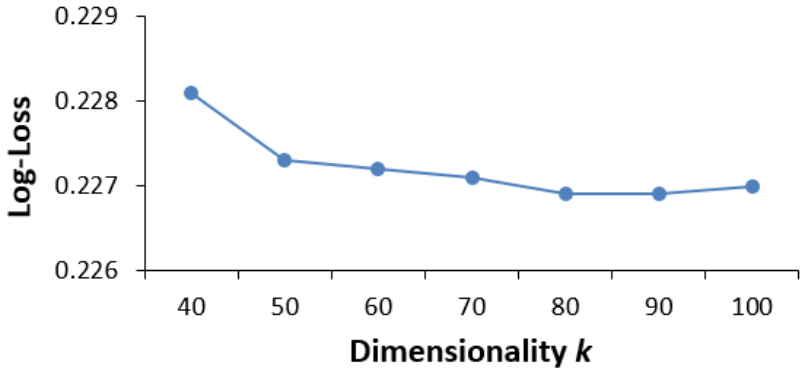

Figure 4.5 Log-loss of ks.

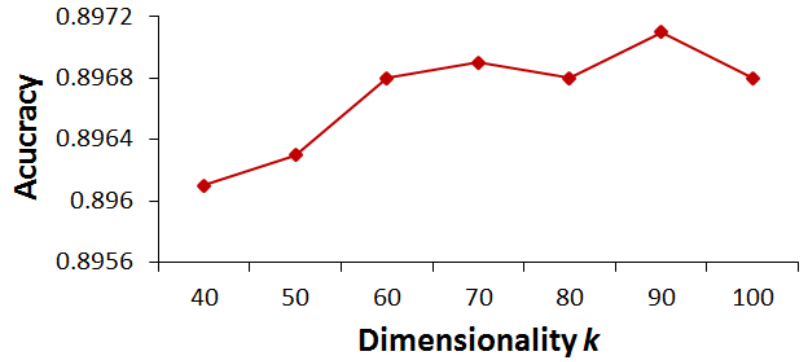

Figure 4.6 Accuracy of ks.

$\beta = 0.1$. Thus, the best β settings for *ps-fhb* and *ps-nhb* are almost the same. Similar to *ps-fhb*, since $\beta = 0.2$ leads to decent performance for all metrics, we use the Weibull distribution with $\beta = 0.2$ for *ps-nhb* in the rest of the experiments.

4.3.2 Comparison of Different Dimensionalities

Both proposed techniques, pairwise interaction tensor factorization and DNN, have parameters to control the complexity of the models. In pairwise interaction tensor factorization, each feature is represented by a k-length latent vector that carries characteristics of the feature. k is an important parameter which can significantly impact model performance. In theory, a larger k has better representation and results in a more complex model; however, it has a higher chance to overfit.

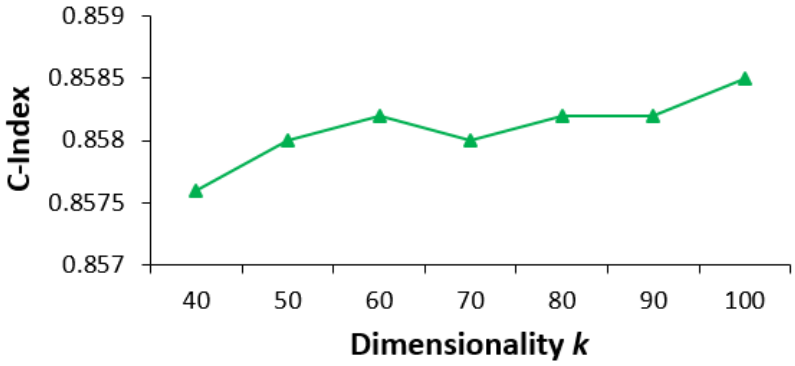

Figure 4.7 C-Index of ks.

Dimensionality k is usually determined through experiments. The DNN technique has more parameters, such as the length of the embeddings and the length of each abstraction layer. We carefully fine tune all parameters in order to find the best performance. Due to space constraints, we only present the effects of the most important parameters in the network structure: the embedding length of user, page, and ad placement.

We first present the impact of k in *ps-fhb*. The results are shown in Figure 4.5, 4.6, and 4.7. They indicate that, with the increasing in vector dimensionality, the performance on the test data generally improves. The reason is that longer latent feature vectors can better capture the signals in the training data so as to improve model complexity. In addition, the performance of log-loss and accuracy increases fast when k grows from 40 to 60. After 60, the growth slows down and even reduces. This is because most signals in the training data have been captured, and further increasing k results in overfitting. C-Index also follows such a trend. In addition, we observe that $k = 100$ leads to a jump when compared with $k = 90$. The possible reason is that $k = 100$ may happen to correctly predict many test instances in the first case of Figure 3.5, which increases the customized C-Index. Finally, we notice that there is no single k that results in best performance for all three metrics: When

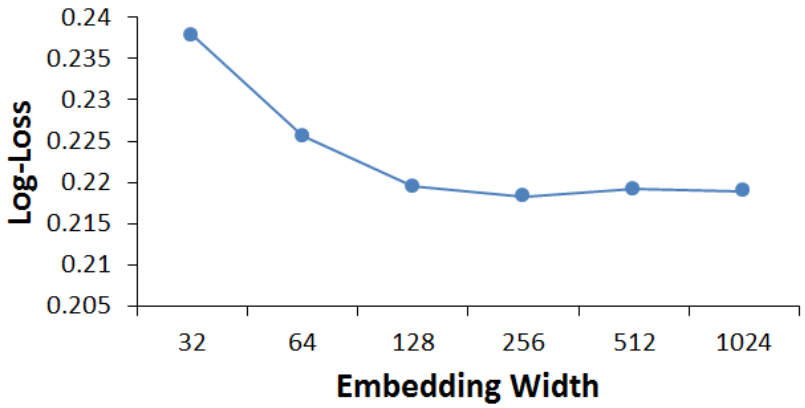

Figure 4.8 Log-loss of embedding width.

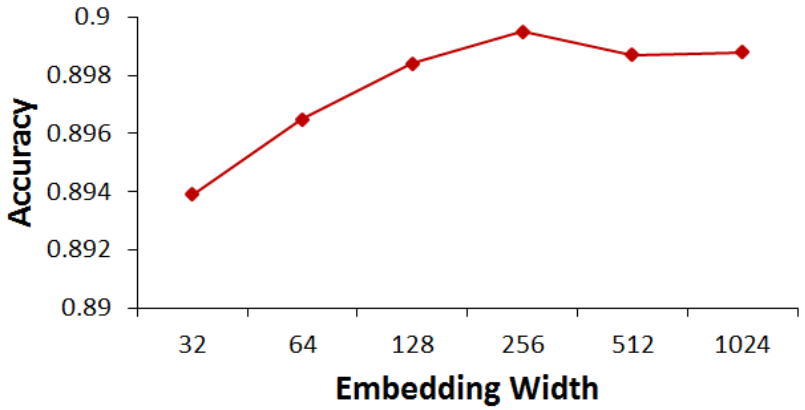

Figure 4.9 Accuracy of embedding width.

$k = 80$, we get the lowest log-loss in the test data. $k = 90$ leads to the highest accuracy. $k = 100$ wins the C-Index test.

The effects of different embedding lengths in *ps-nhb* are presented in Figure 4.8, 4.9, and 4.10. The overall trends are similar to those in Figures 4.5, 4.6, and 4.7. However, the curves in Figures 4.8, 4.9, and 4.10 are smoother than those in Figure 4.5, 4.6, and 4.7. The reason is that DNN has a much more complex structure than factorization machines. Therefore, the changes of the embedding lengths may not necessarily have drastic changes on the final performance. When the embedding length is set to 256, the model achieves the best log-loss and accuracy. It can also

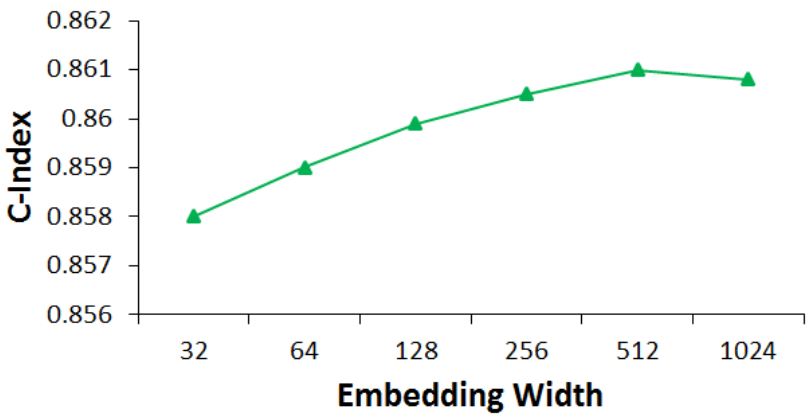

Figure 4.10 C-index of embedding width.

obtain decent performance by C-index. To get the highest C-Index, one can increase the embedding length to 512.

Publishers can select the right dimensionality based on their objectives: Log-loss can make balance between ad revenue and failure risk. Accuracy maximizes the total number of impressions that are correctly classified. C-index can be used when the observed reserve prices and revenues are believed to be very off from the actual highest bids.

As minimizing overall log-loss is the training objective, in the rest of the experiments, we set $k = 80$ for *ps-fhb* and embedding length to 256 for *ps-nhb*.

4.3.3 Overall Comparison

We compare the proposed model with two baselines: *KM* and *OR*. Table 4.1 presents the results. All three versions of our model outperform the two baselines. Among the three versions, *ps-fhb* is the best.

The C-Index values of the two baselines are all zero because: 1) KM makes predictions based on the percentages of impressions whose reserve prices that are less than a given price have already failed. As the overall failure rate increases with the increase of the reserve price, KM always "thinks" an impression with a higher reserve

Table 4.1 Comparison for All Impressions

	Log-Loss	Accuracy	C-Index
KM	1.5762	0.5495	0.0
OR	0.6883	0.5486	0.0
Ps	0.2425	0.8880	0.8532
Ps-f	0.2305	0.8953	0.8577
Ps-fhb	0.2266	0.8972	0.8583
Ps-nhb	0.2183	0.8995	0.8590

Table 4.2 Comparison for Impressions with Header Bids

	Log-Loss	Accuracy	C-Index
KM	1.5766	0.5493	0.0
OR	0.6881	0.5487	0.0
Ps	0.2438	0.8879	0.8533
Ps-f	0.2311	0.8946	0.8573
Ps-fhb	0.2186	0.9011	0.8597
Ps-nhb	0.2017	0.9102	0.8615

price has a higher failure rate (i.e., case 1 in Figure 3.5 never happens). 2) Likewise, OR is a linear model. It learns from the data that the failure rate is positively correlated with the reserve price. It always gives a higher reserve price a greater failure rate than a lower reserve price.

The *ps* model has good performance and clearly outperforms the baselines. Adding interaction factorization, *ps-f* reduces the log-loss by 5% because latent feature vectors can better capture the regularities in the training data and overcome data sparsity compared to one-hot encoding. Furthermore, utilizing header bidding to regularize the model, *ps-fhb* can reduce the log-loss by additional 1.7% from *ps-f*.

The *ps-nhb* model can further reduce the log-loss by 3.7%. It learns multi-dimensional representation to capture the characteristics of input features. The *ps-nhb* model also leverages deep structures to compute the joint effects among the input features, which is the main reason that it outperforms *ps-fhb* by all three metrics.

Header bidding regularization is only applicable on impressions with header bids. To fully present its effect, we filter out impressions without header bids from the test set. The results are shown in Table 4.2, and they demonstrate that *Ps-fhb* has a larger performance improvement compared with the other models.

4.4 Summary

This chapter proposes parametric survival models to predict the failure rate of the reserve price of an online display ad impression. The model is further augmented by a deep neural network (DNN) technique to capture the feature interaction and header bidding factorization. We also develop a customized C-Index for datasets containing only left- and right-censored instances. The experiments show that the proposed models with the Weibull distribution significantly outperforms a Kaplan-Meier model and a logistic regression with observed reserve price/revenue as the feature. The DNN technique and header bidding regularization further boost performance. Our model can be adopted by the majority of online publishers because similar data can be conveniently collected on most publishers' platforms.

CHAPTER 5

RESERVE PRICE OPTIMIZATION IN FIRST-PRICE AUCTIONS

Online display advertising is the most important revenue stream of most online websites. Websites, such as Facebook, Google, and Yahoo, provide free information and services run on display ads. In display advertising, advertisers (e.g., Volkswagen) pay publishers (e.g., Forbes) for showing banners, videos, or text on their webpages. Figure 5.1 is an example display ad. One display of an ad in a page view is called an *ad impression*. Display ads allow for catchy messaging, plus graphics, video, and advertisers' branding to stand out and attract attention.

One of the main ad selling methods is real time bidding: An impression triggered in the real-time is sent to ad exchanges with a reserve price provided by the publisher. Reserve price is the minimum price that the publisher would be willing to accept from advertisers for this impression. The impression is then bid by advertisers. The winning advertiser is allowed to show the ad on the publisher's webpage.

Second-price auctions and first-price auctions are two main auctions used in online display advertising. Over the last few years, the whole display advertising market is switching from second-price auctions to first-price auctions. Unlike second-price auctions in which winners are charged by the second highest bids, in first-price auctions, the winning advertisers pay the prices that they just bid if they outbid the publisher's reserve price. If all bids are lower than the reserve price (i.e., underbid), the impression is unsold. If at least one bid is equal to or higher than the reserve price (i.e., outbid), the impression is regarded as sold through real-time bidding.

Although the highest bid is accessible to publishers (i.e., no censorship exists) in first-price auctions, setting good reserve prices is still tricky and worth of exploring: Similar as what happens in second-price auctions, a too high reserve price may fail to

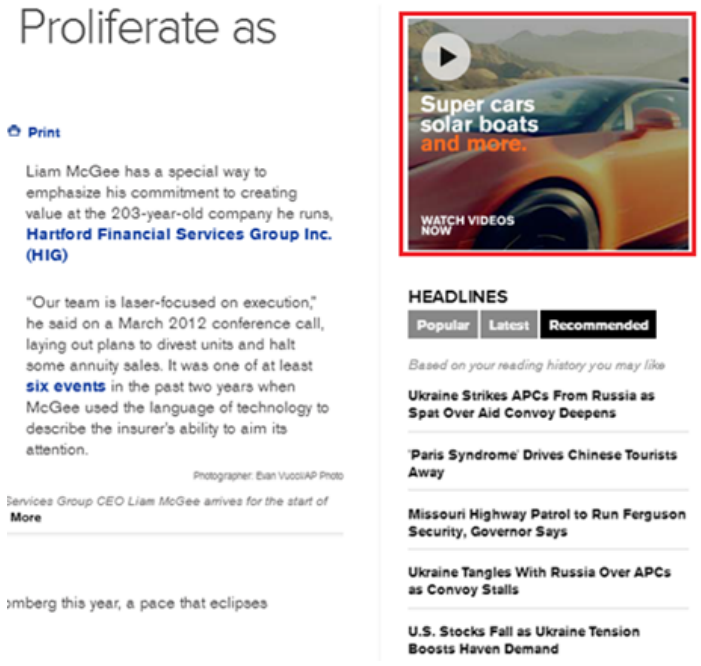

Figure 5.1 An example of a display ad.

be outbid, in which case the publisher receives zero revenue from the RTB auction. On the other hand, a too low reserve price is not able to stimulate advertisers to bid higher: To have a chance to win, advertisers may only need to bid slightly higher than the reserve price, which results in a sub-optimal impression revenue. In preliminary data analysis, when a reserve price was outbid, highly likely its final revenue was slightly higher than the reserve price. This means that advertisers' bids are correlated to the reserve prices. Hence, a good reserve price can optimize the publisher's impression revenue: Always setting their reserve prices slightly less than the highest prices can motivate advertisers to bid higher next time.

Since the highest bids are uncensored in first-price auctions, publishers can build a machine learning model to predict the highest bids using their historical real-time bidding transaction data. Given an impression, publishers just set their reserve prices right below the predicted highest bids. In this case, publishers can win all impressions and try to push future bids higher. However, due to uncertainty in

the ad market, data noise, and model misspecification, predicted highest bids are not always equal to true highest bids. Publishers may fail to sell ad impressions. Directly predicting highest bids is risky. Therefore, instead of point estimation, we do interval estimation. i.e., $[\hat{b}_L, +\infty]$. Namely, instead of predicting highest bids, we propose to predict the lower bound of the highest bids, b_L, with a pre-specified confidence level, $(1-\alpha)\%$. Publishers can set their own confidence level according to their revenue goal: decreasing the confidence level leads to a more aggressive strategy (i.e., less risk), while increasing the confidence level leads to a more conservative strategy (i.e., higher risk).

In statistical inference, a prediction interval is an estimate of an interval in which a future observation will fall, with a certain confidence level $(1-\alpha)\%$ (i.e., coverage probability). For instance, a prediction interval $[0.1, +\infty]$ with 80% confidence level means at least 80% chance that the highest bid is not less than \$0.1. Publishers can consider α as the risk level. The risk is that the highest bid is likely to be less than the reserve price. In this case, a publisher can set the reserve price to the lower bound. For instance, a publisher sets the risk level to 20%, i.e., $\alpha = 20\%$. Assuming the model predicts $\hat{b}_L = 0.1$, a reserve price equal to \$0.1 can guarantee that it is at least 80% chance that the highest bid will be equal to or higher than \$0.1, i.e., the reserve price can be outbid. To build such model, we propose to modify an existing loss function QD [52] for computing prediction intervals, fed by historical impressions whose reserve prices were outbid.

Besides outbid impressions, those underbid impressions also carry important information on advertisers' bidding behavior. To this end, we propose a multi-task learning neural network to predict both b_L and the hazard rate h (i.e., the probability of being underbid). The final loss function combines the loss of prediction interval estimation and the loss of a proportional hazards model.

In addition to uncertainty in prediction, it is always challenging to model the characteristics of users, pages and ad slots, which play a significant role in determining impression prices. Especially, publishers have very limited information about users. Thus, it is not possible to explicitly build user features which may impact advertisers' bids. Moreover, advertisers are using different real-time bidding algorithms (from deterministic methods to machine learning models), which are black-box to publishers. It is not feasible to reverse-engineer their algorithms on the publisher side. To overcome these issues, a deep neural network is used to capture the complicated joint effect between the features. The deep neural network models users, pages, and ad placements features using embeddings. These latent vectors will be able to learn latent features from massive historical transaction data on the publisher side.

The contributions of this work are as follows: 1) This is the first work that attempts to optimize reserve prices for first price auctions. 2) We use prediction interval estimation, instead of point estimation, to better quantify the uncertainty in the highest bid prediction. Publishers can arbitrarily adjust the risk level based on their business strategies. 3) We propose a multi-task learning algorithm which predicts the highest bid lower bound and hazard rate using both outbid and underbid impressions. 4) The proposed method is evaluated on a real-life transaction dataset, which contains tens of millions of impression transactions.

5.1 The Proposed Multi-Task Learning Loss Function

In this section, we define the research problem, the censorship issue in the data, and the reason of using prediction interval estimation. We then introduce the proposed multi-learning framework which combines prediction interval estimation and survival analysis.

5.1.1 Problem Definition

Problem Definition 2. *Given an ad impression A_i and a risk level $\alpha\%$, the goal is to determine a reserve price so that there is at least $(1-\alpha)\%$ chance that the reserve price will be outbid.*

In first-price auctions, the real-time bidding transaction data on the publisher side contains censorship. The highest bids (i.e., how much advertisers are willing to bid at most) are not always visible: if a reserve price was outbid, the highest bid and the reserve price are known, i.e., uncensored. If a reserve price was underbid, the publisher knows the reserve price only. The highest bid is not accessible on the publisher's side, i.e., left-censored.

Directly predicting the exact highest bids is risky. Too high reserve prices make publishers lose significant revenue. Therefore, it is proposed to predict ranges of the highest bids with a certain confidence level. The final reserve price is the lower bound so that the highest bids will be guaranteed to be more than the lower bound with a certain probability. To achieve his goal, we propose to construct prediction intervals, instead of traditional point estimation.

5.1.2 Multi-task Learning Framework

Historical outbid impressions can be used to learn for constructing prediction intervals because their highest bids (i.e., revenue) are known. However, historical underbid impressions cannot be directly used because their highest bids are censored. Only utilizing outbid impressions may lose significant information about advertisers' bidding behaviors. Therefore, to more efficiently leverage historical information, we propose to use a multi-task learning neural network which predicts both the lower bound of the highest bids b_L and the hazard rate h. The loss of the lower bounds estimation is computed based on only outbid impressions, while the loss of the hazard rate estimation is computed based on both outbid and underbid impressions.

5.1.3 Highest Bid Lower Bound Prediction

This section introduces an existing loss function for prediction interval construction, which is used in this research to predict lower bounds of highest bids with a certain confidence level. Highest bid lower bound prediction requires true highest bids. Historical underbid impressions are not used in training because it is impossible to determine whether estimated prediction intervals are appropriate without true highest bids. Hence, only historical outbid impressions are available because their highest bids are known.

Pearce et al proposed this loss function for Quality-Driven prediction interval estimation, denoted as QD in [52]. It has been applied in existing studies, such as [57, 23, 42] for wind power interval prediction and it is modified in this project to be used in reserve price optimization: the upper bound included in the original loss function is canceled because the ranges of the highest bids are one-sided, i.e., the upper bound is $+\infty$.

Pearce et al. [52] proposed this quality-driven and distribution-free loss function that can generate high-quality prediction intervals which are as narrow as possible and meanwhile capture some specified proportion of data points, i.e., High-Quality (HQ) principle [32]. Compared with traditional prediction interval (PI) construction methods, which minimize prediction errors, QD directly improves PI quality. The constructed PIs are guaranteed to be optimal in terms of their key characteristics: width and coverage probability.

In general, the QD loss function has two components: Mean Prediction Interval Width (MPIW) and Prediction Interval Coverage Probability (PICP). The overall loss is the sum of MPIW and PICP.

MPIW measures the width of the average prediction intervals. The assumption is that a wide prediction interval (e.g., $[0, +\infty]$) is not informative and useful at all. Therefore, a good prediction interval should be as narrow as possible. QD defines

a captured MPIW, denoted as $MPIW_{capt.}$, which represents the average width of prediction intervals that correctly include the ground truth labels:

$$MPIW_{capt.} = \frac{1}{\sum_{i=1}^{n} k_i} \left(\hat{b}_{U_i} - \hat{b}_{L_i} \right) \cdot k_i \tag{5.1}$$

where k_i is a Boolean indicator representing whether the ground truth label of the ith sample out of n samples is correctly captured in the estimated PI. $\hat{b}_{U_i}$ and $\hat{b}_{L_i}$ are the upper and lower bound of the estimated PI, respectively. The higher $MPIW_{capt.}$, the better PI quality.

In our case, since we only care about the the lower bound, b_{U_i} is $+\infty$, (i.e., the interval is $[b_{L_i}, +\infty]$). It is removed from Section 5.1. Note that $k_i = 1$ if $b_{L_i} \leqslant b_i$, where b_i is the actual highest bid:

$$MPIW_{capt.} = -\frac{\hat{b}_{L_i} k_i}{\sum_{i=1}^{n} k_i} \tag{5.2}$$

PICP measures the coverage probability of the estimated PIs, i.e., how many ground truth labels are correctly captured (Equation. 5.3).

$$PICP = \frac{1}{n} \sum_{n}^{i=1} k_i \tag{5.3}$$

PICP is the most important indicator on the quality of PIs. [52] tries to learn parameters θ that can minimize $L_\theta = L\left(\theta | \mathbf{k}, \alpha\right)$. It can be further represented by a binomial distribution: $L_\theta = \binom{n}{c}(1-\alpha)^c \alpha^{n-c}$, where $c = \sum_{i=1}^{n} k_i$. Using the de Moivre-Laplace theorem, it can further be approximated by a normal distribution. Therefore, the negative log likelihood is updated to:

$$-logL_\theta \propto \frac{n}{\alpha(1-\alpha)}((1-\alpha) - PICP)^2 \tag{5.4}$$

Putting the MPIW and PICP terms together, the loss of highest bid lower bound prediction considers both width and coverage, as shown in Equation (5.5). λ is a parameter controlling the importance of PICP.

$$Loss_{qd} = MPIW_{capt.} + \lambda \cdot PICP = -\frac{\hat{b}_{L_i} k_i}{\sum_{i=1}^{n} k_i} + \frac{n}{\alpha(1-\alpha)} max(0, (1-\alpha) - \frac{1}{n}\sum_{n}^{i=1} k_i)^2 \tag{5.5}$$

5.1.4 Hazard Rate Prediction

Section 5.1.3 uses only historical outbid impressions. However, historical underbid impressions also carry important information on advertisers bidding patterns. Since the highest bids of underbids impressions are censored, we proposed to use survival models, which have been adopted in [67, 73, 28] in the online advertising field. Thus, to leverage underbid impressions, the proposed loss function incorporates a loss function of a Cox's proportional hazards model (Cox PH model).

$$h(t, X_i) = h_0(t) e^{\hat{y}_i} \tag{5.6}$$

The Cox PH model gives an expression for the hazard at time t for an individual with a given specification of a set of explanatory variables [35]. The Cox PH model formula is shown in Equation (5.6). The Cox PH model consists of two parts:
1) the underlying baseline hazard function, $h_0(t)$, is called the baseline hazard function. It is the cumulative hazard rate, i.e., the percentage of the training instances whose events have already occurred at t. $h_0(t)$ describing how the risk of event per time unit changes over t at baseline levels of explanatory variables; 2) the exponential term is the exponential expression e to $\hat{y}_i$, which is computed from trainable parameters θ and explanatory variables X_i (the feature values of the ith

impression). $\hat{y}_i$ describes how the hazard varies in response to explanatory variables. The output $h(t, X_i)$ is the hazard rate of X_i at time t.

Note that $h_0(t)$ is an unspecified function. It is this property that makes the Cox PH model a semiparametric model. We do not need to assume that the baseline hazard follows any specific distribution. $h_0(t)$ can be easily computed from existing observations. This is one of the reasons that the Cox PH model is so popular. It is important in the online display advertising scenario because the ad market is highly complicated and dynamic. The highest bids may not always be drawn from a specific distribution.

To handle both uncensored and censored data, the parameters θ are estimated using the Cox PH partial likelihood function. The partial likelihood function considers probabilities only for those training instances whose events happen and does not explicitly consider probabilities for those whose are censored. In particular, although the partial likelihood focuses on instances which fail, survival time information prior to censorship is used for those which are censored.

To apply survival analysis to our case, we can make an analogy: one impression is an instance, which has a set of features. The event of interest is that all advertisers bid lower than the reserve price r (i.e., underbid). The time to event is the reserve price. Left-censored instances are underbid impressions (only the reserve price r is known), while uncensored instances are outbid impressions (the highest bid b is known). With the increase in the reserve price (i.e., time to event), the event of interest also increases. When the reserve price is \$0, it is most likely that the event of reserve price failure does not occur. If the reserve price is high (e.g., \$100), it can hardly receive a higher bid.

Therefore, for an underbid impression A_i whose reserve price is r_i, we find all outbid impressions A_j whose highest bid is b_j, where $b_j \geqslant r_i$. Since it is already known that A_i was underbid before r_i and A_j was not underbid at r_i, the goal is to

find θ that can maximize $h(r_i, X_i) - h(r_i, X_j)$. Therefore, the partial likelihood of the event at a price r_i is:

$$L_i = \frac{h(r_i, X_i)}{\sum_{j:b_j \geqslant r_i} h(r_i, X_j)} = \frac{h_0(r_i)e^{\hat{y}_i}}{\sum_{j:b_j \geqslant r_i} h_0(r_i)e^{\hat{y}_i}} = \frac{e^{\hat{y}_i}}{\sum_{j:b_j \geqslant r_i} e^{\hat{y}_i}} \tag{5.7}$$

Treating the impressions as if they are statistically independent of each other, the joint probability of all uncensored cases is $L_\theta = \prod_{A_i \in U} L_i$. U is the set of underbid impressions. The loss of the hazard rate prediction, (i.e., negative log partial likelihood) is Equation (5.8):

$$Loss_{cox} = \sum_{A_i \in U} (log \sum_{j:b_j \geqslant r_i} e^{\hat{y}_i} - \hat{y}_i) \tag{5.8}$$

Therefore, incorporating Equations (5.5) and (5.8) together, the final loss is Equation (5.9). μ is a parameter controlling the importance of the hazard rate prediction.

$$\begin{aligned} Loss &= Loss_{qd} + Loss_{cox} \\ &= -\frac{\hat{b}_{L_i} k_i}{\sum_{i=1}^{n} k_i} + \lambda \frac{n}{\alpha(1-\alpha)} max(0, (1-\alpha) - \frac{1}{n}\sum_{n}^{i=1} k_i)^2 \\ &+ \mu \sum_{A_i \in U} (log \sum_{j:b_j \geqslant r_i} e^{\hat{y}_i} - \hat{y}_i) \end{aligned} \tag{5.9}$$

$\hat{b}_{L_i}$ and $\hat{y}_i$ are computed from a mapping of θ and the features X. Next section introduces the proposed mapping.

5.2 Predicting Highest Bid Lower Bounds and Hazard Rates

Section 5.1 introduces a multi-task learning loss function that predicts both lower bounds of the highest bids and hazard rates. Given $\hat{b}_{L_i}$ and $\hat{y}_i$, the final loss function Equation 5.9 calculates the training loss. The next task is to predict $\hat{b}_{L_i}$ and $\hat{y}_i$.

The highest bid reflects the value of an ad impression. Advertisers will bid high if they believe showing their ad in an impression can benefit the advertising campaigns. The value of an impression is determined by multiple factors: 1) User: User interest is the most important factor. If advertisers think the user in the impression likely has interest in their products, they will be willing to pay more for this impression. 2) Ad placement (e.g., position): if an ad placement on top of a page is much more viewable than one at the bottom [63]. 3) Page (e.g., topics): an ad opportunity on a web article about electronic products may be more attractive than one on a political article because the user who reads the former more likely has shopping intent. 4) Context (e.g., time): the time that an impression occurs also matters. The average bid price varies over the day.

It is highly challenging to predict advertisers' bidding behavior on the publisher side. Most advertisers either collect by themselves or buy from third-party companies excessive information about users and real-time ad market. Hence, advertisers are using massive data and building complicated algorithms to determine their bids in real-time. However, most publishers do not have detailed user data. Many online media websites even do not require visitors to log in before reading articles. This study is to analyze the problem from the publishers' point of view. Thus, only limited user information is utilized.

To estimate the value of ad impressions, features about these four factors are taken into account. User features include 1) user IDs, 2) state-level location, 3) operating system and Internet browser, 4) network bandwidth, and 5) devices. Note that, unlike advertisers, publishers usually do not have access to user personal data. The above features are identified from the user cookies. Users' personally identifiable information (PII) is not used in this study. Ad placement features include 1) ad unit size, e.g., 123x324 and 2) ad position (On the publisher's page template, each ad slot has a unique name that represents its position). Page features include 1) page URLs,

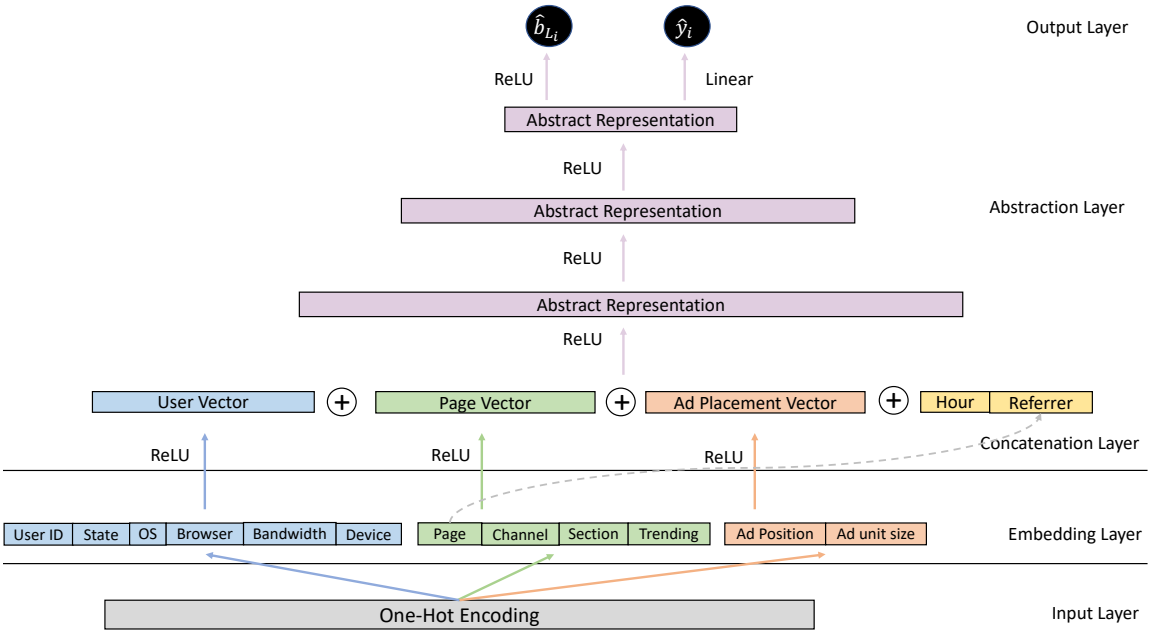

Figure 5.2 The architecture of the proposed multi-task learning neural network.

2) channels, e.g., business and lifestyle, 3) sections, i.e., sub-channels, and 4) the trending status of the page (i.e., if the page is labeled as trending by the publishers' editors). Context features include 1) hour of the day and 2) referrer URLs, i.e., which page the current page request originated from.

In addition, due to the complexity of advertisers' real-time bidding algorithms and the uncertainty in the ad market, publishers have to learn latent features, instead of using explicit features, from historical data. It is also important to capture interaction among latent features. Therefore, simple linear models are not sufficient for this task. This work proposes to use deep neural networks, which has been extensively studied and deployed in many applications, e.g., recommender systems. To reduce the complexity of the model and improve prediction performance, we adopt the multi-task learning framework. It predicts $\hat{b}_{L_i}$ and $\hat{y}_i$ using a set of shared parameters. Figure 5.2 presents the architecture of the proposed multi-task learning neural network.

Input Layer: All features listed above are either categorical or can be easily converted to categorical variables. For instance, ad unit size is converted to a string feature, such as "747x413". These categorical variables are then one-hot encoded. A publisher usually has fixed number of standard ad unit sizes on the templates. Using one string feature, instead of two integer features, can capture the interaction of width and height.

Embedding Layer: User, page, and ad placement features are represented by latent embedding vectors. Embedding vectors can significantly enhance model complexity and the ability of fitting training data. It is also able to capture the sophisticated ad market from historical data. The embedding layer retrieves feature embeddings based on the one-hot input. The lengths of the embeddings are pre-specified parameters that can be tuned by experiments. The feature embeddings of the three factors are then concatenated, respectively. These features will be combined into unified vectors for the user, the page, and the ad placement by a Rectified Linear Unit (ReLU) layer. The Rectified Linear Unit is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back. ReLU is written as $f(x) = max(0, x)$. It can allow the model to account for non-linearities and interactions. In theory, ReLU is able to approximate any function. Thus, it is suitble for our application, in which we have to mimic advertisers' black-box bidding algorithms.

Concatenation Layer: Besides concatenating user, page, and ad placement vectors, the concatenation layer also incorporate context information. To reduce the number of parameters that need to be learned from training data. Page and referrer share the same embedding matrix. Parameter sharing can reduce overfitting and decrease training cost.

Abstraction Layer: The concatenated vector is then further mapped to lower-dimension representations by multiple fully connected layers with ReLU activation.

Output Layer: The output layer outputs two values: 1) the lower bound of the highest bid $\hat{b}_{L_i}$, whose range is $[0, +\infty]$. Hence, a ReLU function is used as the activation function. 2) the exponential of the Cox PH model $\hat{y}_{U_i}$, whose range is $[-\infty, +\infty]$. Hence, a linear function is used as the activation function.

The predicted $\hat{b}_{L_i}$ and $\hat{y}_{U_i}$ are then fed into the final loss function, i.e., Equation (5.9). Stochastic gradient descent is used to train the neural network.

5.2.1 Risk Level Selection

The proposed model allows publishers first specify a risk level $\alpha \in (0, 1)$. The risk level determines predicted lower bounds of the highest bids, which is the predicted reserve price. Thus, the risk level can be explained as the percentage of underbid impressions the publisher can tolerate. This risk level is a trade-off: a high risk level causes many underbid impressions. On the other hand, a low risk level may incur low highest bids due to low predicted lower bounds $\hat{b}_{L_i}$ (i.e., predicted reserve prices). The optimal risk level can be determined based on the publisher's business strategy (i.e., aggressive or conservative). It can also be set empirically by A/B testing (i.e., select the one that can maximize total revenue).

5.3 Evaluation

5.3.1 Experimental Data

The data used in this project is collected in four days in March 2021 at Forbes Media's website. Forbes Media switched to first-price auctions later 2020. The transactions in the dataset are all first-price auctions. The dataset contains nearly 60 million impressions on average per day, in which the ratio of outbid impressions and underbid impressions is about 3:2.

The models are trained over both outbid and under impressions, while being evaluated using only outbid impressions due to presence of the highest bids. The evaluation is conducted three times: a model trained using the all impressions on one day is tested on the outbid impressions of the next day. As we have four days data, each model is tested on three days test data.

5.3.2 Implementation

The proposed model is implemented using Tensorflow. The experiments are run on a desktop with i7 3.60Hz CPU and 32GB RAM. The computation is sped up using NVIDIA GeForce GTX 1060 6G GPU.

The training goal is to minimize the total loss in Equation (5.9). Since the large training dataset does fit the memory, the optimizer we adopt is Stochastic Gradient Descent (SGD) with a learning rate of 10^{-3}. The training batch size is set to 256.

To avoid overfitting, across all 10 epochs, the model that performs the best on the validation data is applied to the test data.

The widths of the embeddings and abstraction layers are empirically set to be 128. We adapt two abstraction layers.

5.3.3 Evaluation Metrics

The main purpose of the proposed multi-task learning model is to predict the lower bounds of highest bids b_L with a risk level α. It is a special case of prediction interval estimation, in which the upper bound is $+\infty$.

Following the HQ principle [32, 52], two evaluation metrics are used: Mean Prediction Interval Width (MPIW) and *Prediction Interval Coverage Probability (PICP)*.

PICP measures the coverage probability of the estimated PIs, i.e., how many ground truth labels are correctly captured. It is defined in Equation (5.3). PICP indicates the percentage of outbid impressions in the prediction results.

MPIW measures the average width of the prediction intervals. It is defined in Equation (5.2). In our application, publishers expect the lower bound (i.e. reserve price) to be as high as possible so that they can optimize long-term ad revenue. Since the upper bound is $+\infty$, MPIW is negative in this application. Also, as the distribution of reserve prices and highest bids are highly left skew, median is a better measure than mean. Thus, to build a more intuitive metric, we use *Median Outbid Reserve Price (MORP)*, which is the median of all $\hat{b}_L$ which $\hat{b}_{L_i} \leqslant b_i$ (i.e., $median(\left\{\hat{b}_{L_i} | \hat{b}_{L_i} \leqslant b_i\right\})$). MORP is a positive value.

PICP signals the percentage of outbid impressions. MORP reflects the median unit price of outbid impressions. Let N is the total number of impressions. Therefore $(PICP * N) * MORP$ is the expected lower bound revenue that the publisher got received from real-time bidding. The reason that it is the lower bound because $\hat{b}_{L_i}$ is the reserve price, which is mostly less than the highest bid (i.e., final impression revenue). To fairly compare results datasets with different sizes, N can be canceled. So, we use the third experimental metric, *Expected Revenue (ER)*, which more intuitively shows the impact of the proposed method to the final revenue.

$$\begin{aligned} ER &= PICP \cdot MORP \cdot N \\ &\sim PICP \cdot MORP \end{aligned} \tag{5.10}$$

5.3.4 Comparison Systems

As two classic methods of prediction interval estimation, MVE and Bootstrap are selected as comparison systems. The proposed method is also compared with the existing QD loss. Note that these methods use different loss functions with the same

set of features, which are constructed using the deep neural network proposed in Section 5.2.

MVE: The MVE method was originally proposed by Nix et al. [49] for construction of PIs. This method assumes that errors are normally distributed around the true mean of targets, $y(x)$. The MVE method estimates the target variance using a dedicated neural network. The dependence of the target variance on the set of inputs is the fundamental assumption of this method for PI construction. The outputs of the neural network are the predicted mean $\hat{\mu}$ and the predicted variance $\hat{\sigma}^2$ of the normal distribution. The final reserve price is $\hat{b}_{L_i}$ which makes $\Phi(\hat{b}_{L_i}) = \alpha$ (Φ is the cumulative distribution function (CDF) of the standard normal distribution).

Bootstrap: To make less biased estimation, it builds B neural network models using different subsets of the parameter space. Collective decisions are made by the ensemble of neural networks [34]. The predicted mean is $\hat{y} = \sum_{B}^{h=1} \hat{y}_h$. The predicted variance $\hat{\sigma}^2_{\hat{y}} = \frac{1}{B-1} \sum_{h=1}^{B} (\hat{y}_h - \hat{y})$. One separate individual neural network is built to estimate the variance of errors $\hat{\sigma}^2_{\epsilon}$. Once both $\hat{\sigma}^2_{\hat{y}}$ and $\hat{\sigma}^2_{\epsilon}$ are known, the ith PI with a confidence level of $(1-\alpha)\%$ can be constructed

$$\hat{y} \pm t_{1-\frac{\alpha}{2},df} \sqrt{\hat{\sigma}^2_{\hat{y}} + \hat{\sigma}^2_{\epsilon}} \tag{5.11}$$

where $t_{1-\frac{\alpha}{2},df}$ is the $1-\frac{\alpha}{2}$ quantile of a cumulative t-distribution function with df degrees of freedom. df is defined as the difference between the number of training samples and the number of parameters of neural networks. The final reserve price is the lower bound: $\hat{y} - t_{1-\frac{\alpha}{2},df} \sqrt{\hat{\sigma}^2_{\hat{y}} + \hat{\sigma}^2_{\epsilon}}$.

LUBE: LUBE [32] was developed based on the HQ principle (as described in Subsection 5.1.3). It considers PICP and normalized MPIW (NMPIW). NMPIW is equal to MPIW divided by the range of the underlying target. Since the target range in our application is indefinite. In the experiments, we use MPIW instead. LUBE

tries to minimize coverage width-based criterion (CWC) which is used for evaluation of PIs:

$$CWC = MPIW(1 + \gamma(PICP)e^{-\eta(PICP-\mu)}) \tag{5.12}$$

where $MPIW = \frac{1}{n}\sum_{i=1}^{n} \hat{b}_{U_i} - \hat{b}_{L_i}$, PICP is the same as the one in the proposed method. The constant η and μ are two hyperparameters determining how much penalty is assigned to PIs with a lower coverage probability. $\gamma(PICP)$ is a step function that is 1 if $PICP \geq \mu$ otherwise 0.

Since LUBE is not differentiable everywhere, it is proposed with Simulated Annealing (SA) as the training method.

QD: QD is a quality-driven distribution-free loss function is proposed in [52]. It is proposed based on LUBE. It has been described in Subsection 5.1.3. The formula is shown in Equation (5.5). QD actually is a model without the failure rate prediction part described in Subsection 5.1.4.

5.3.5 Performance by Varying λ and μ

Before comparing with comparison systems, we first seek for the optimal λ and μ through experiments. To compare them fairly, the risk level α is controlled to 30%. In other words, the prediction intervals, i.e., $[\hat{b}_L, +\infty]$, are expected to cover the highest bids of at least 70% impressions. Figures 5.3, 5.4 and 5.5 shows the PICP, MORPs and expected revenue of different combinations of λ and μ, respectively.

The PICPs of all combinations are more than 70%, which meet the minimum coverage. Increasing λ and μ can promote coverage, thereby enhancing PICP: High λ awards the model to focus more on the PICP part in Equation (5.5). High μ stimulates the model to predict failure rate more accurate. Thus, the PICP of the proposed model goes up with the increasing of λ and μ.

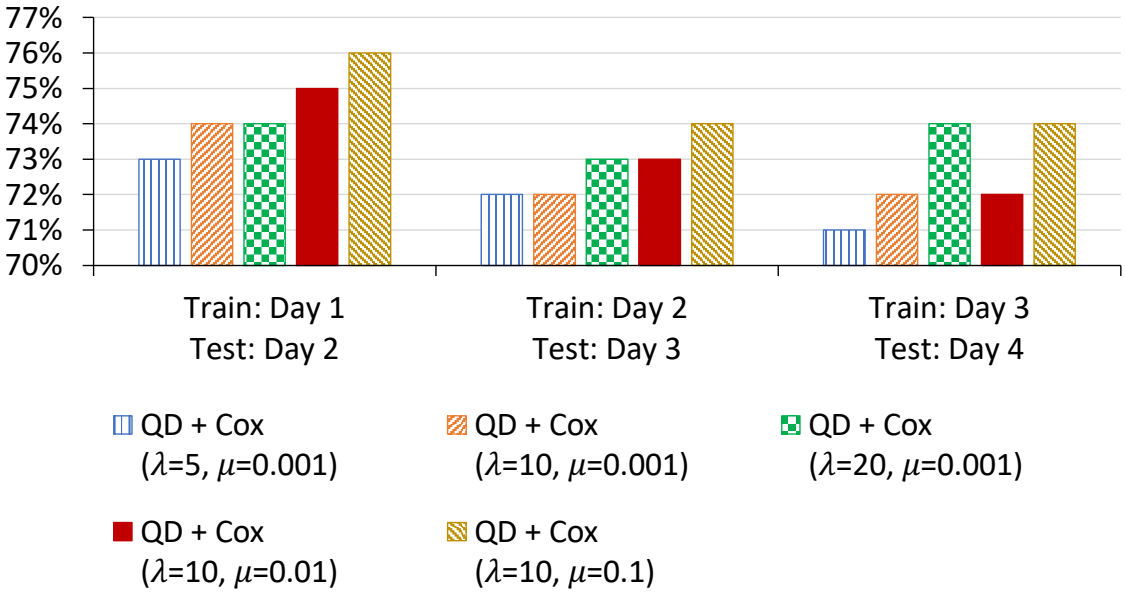

Figure 5.3 The PICPs of different λ and μ.

The proposed method QD+Cox receives the best MORP and expected revenue when $\lambda = 10$ and $\mu = 0.01$ across all three days. In terms of the expected revenue, QD+Cox(10, 0.01) wins the first two days, while QD+Cox(20, 0.001) wins the last one (because its PICP on the last days is much higher). On average, QD+Cox(10, 0.01) has the highest mean expected revenue on the selected three test days.

5.3.6 Performance of Lower Bound Highest Bid Prediction

In this section, QD+Cox(10, 0.01) is compared with the comparison systems (MVE, Bootstrap, LUBE, and QD). The risk level is fixed to 30%. Figures 5.6, 5.7 and 5.8 shows the comparison of PICP, MORPs and expected revenue of the outbid impressions, respectively.

As shown in Figure 5.6, when $\alpha = 30\%$, the PICPs of most methods are more than 70%, which meet the minimum coverage. In other words, the highest bids of 70% impressions are more than or equal to the corresponding $\hat{b}_L$. However, the PICPs of MVE and Bootstrap are less than 70% on the test data of two days.

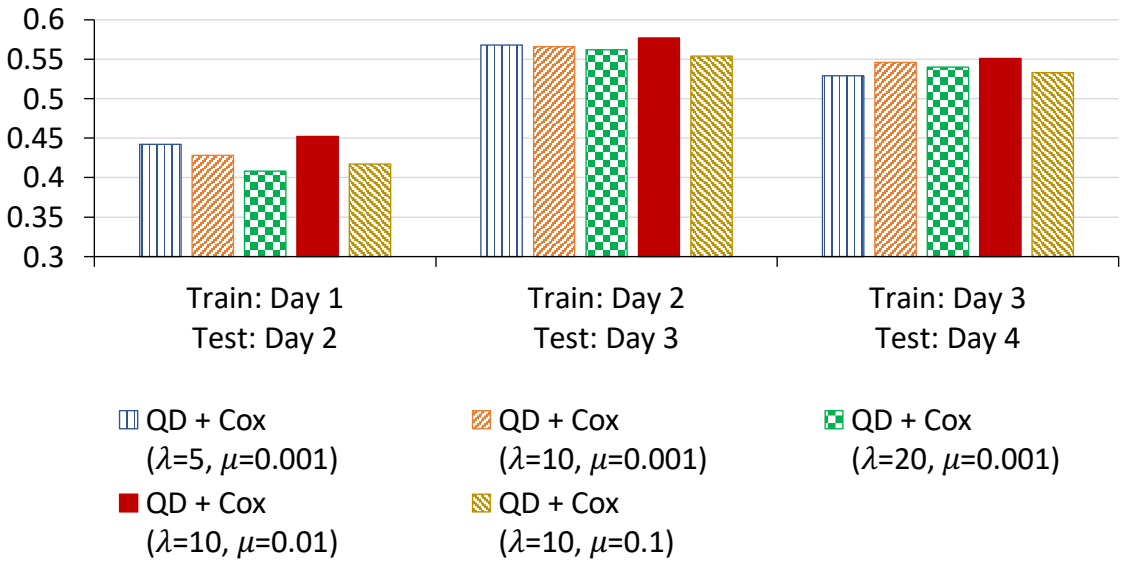

Figure 5.4 The MORPs of different λ and μ.

The MORPs of MVE and Bootstrap are less than other models. It is interesting to see that MVE and Bootstrap do not get high PICPs in return for low MORPs. This indicates that their predicted $\hat{b}_L$ are mostly low. They also produces too high $\hat{b}_L$ which makes many impressions underbid. In theory, MVE has a strong assumption that the variance of the highest bids follows Gaussian distribution. It is an improper estimation in our application because bids are skew to the lower left tail. On the other hand, bootstrap does not have any assumption. However, due to limited number of bootstrap neural networks and the potential bias, leading to an inaccurate estimation of the model misspecification variance [33]. This may lead to either too wide or too narrow prediction intervals.

Since LUBE and QD are recognisable as having the similar objective [52], they have similar performance on our data set (QD is slightly better on average). QD+Cox(10, 0.01) significantly outperforms QD, which does not utilize underbid impressions in the training phase. It indicates that considering underbid impressions

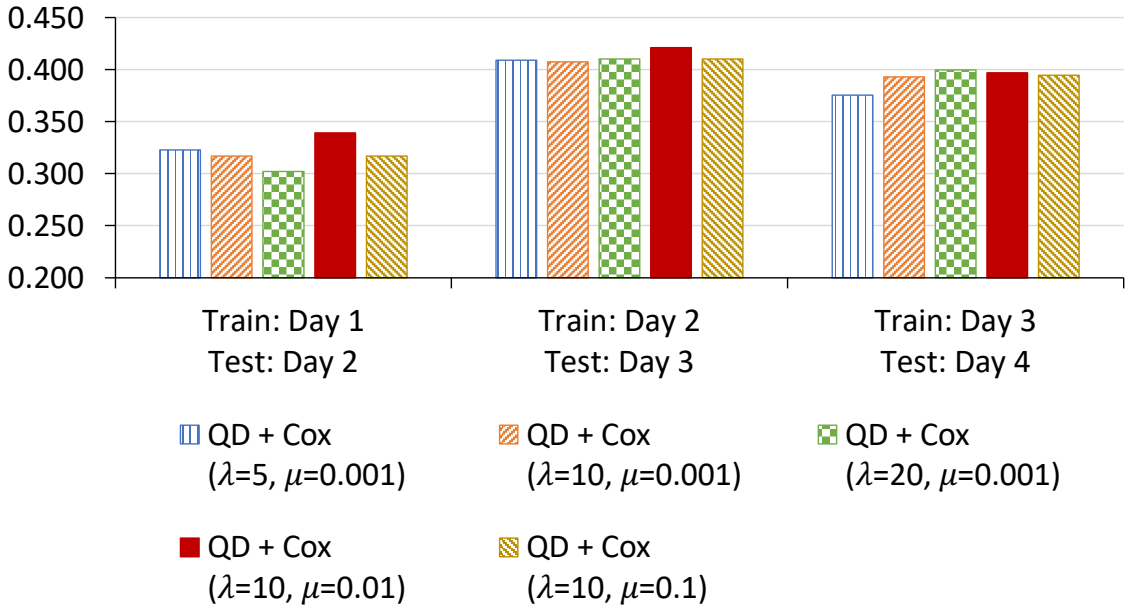

Figure 5.5 The revenue of different λ and μ.

can significantly improve the final prediction performance. Underbid impressions also contain useful patterns on advertisers' bidding behavior.

5.3.7 Performance of Different Risk Levels

The risk level α determines the minimum coverage, i.e., the minimum percentage of predicted outbid impressions. In theory, there is often an inverse relationship between a model's coverage (i.e., PICP) and its lower bounds (i.e., MORP, final reserve prices), where it is possible to increase one at the cost of reducing the other. Reducing the lower bounds can increase the chance of covering the actual highest bids. On the other hand, setting up high reserve prices for individual impressions likely causes more underbid impressions. Therefore, for publishers, setting α is a trade-off between selling more outbid impressions and motivating advertisers to bid high in a long run.

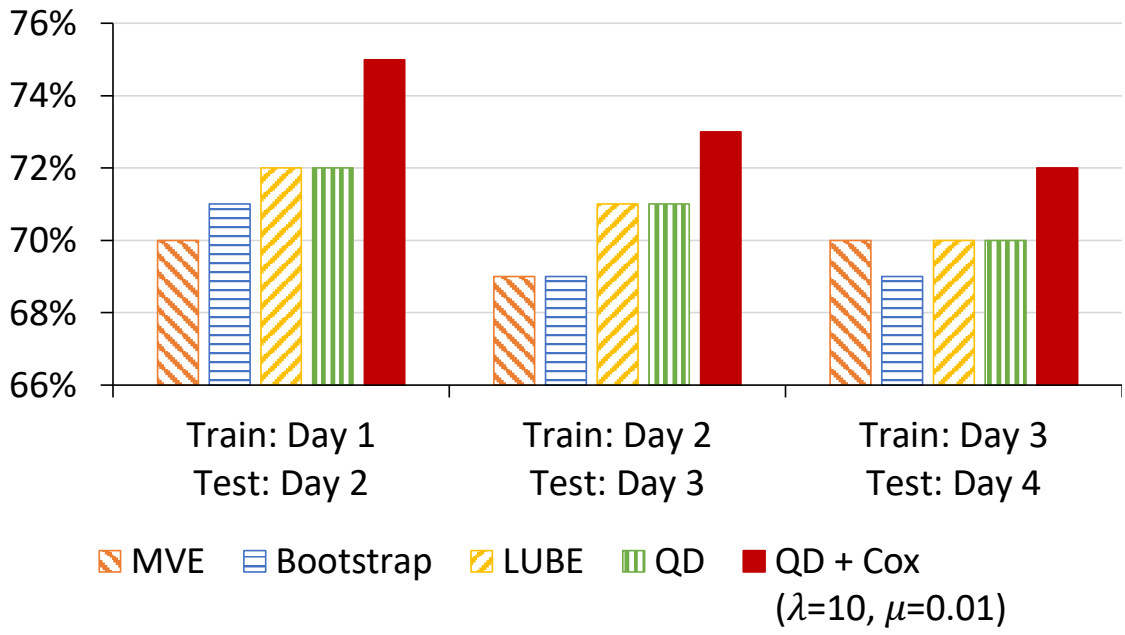

Figure 5.6 The PICPs comparison of lower bound highest bid prediction.

In this experiment, we observe the performance of the proposed method by varying α. The model we use is QD+Cox(10, 0.01) which has the best performance in last section.

Figure 5.9 shows the PICP performance of different α's. When $\alpha = 30\%$ and 40%, the PICPs of QD+Cox(10, 0.01) on the test data can meet the required minimum coverage. However, it is interesting to observe that when $\alpha = 10\%$ and 20% the PICPs on the test data (except $\alpha = 20\%$ on the first day) are less than the required minimum coverage (i.e., 90% and 80%, respectively). The reason is that covering more than 80% on test data is highly difficult due to the challengingness in the data. During model training, the model has to either overfit the training data or fail to meet the minimum coverage (i.e., get converged with a relatively large loss). In the result on the test data, the model fail to meet the minimum coverage.

Figures 5.10 and 5.11 present the MORPs and revenue of different αs on three days. $\alpha = 40\%$ has the highest MORPs, i.e., the median reserve price is the highest.

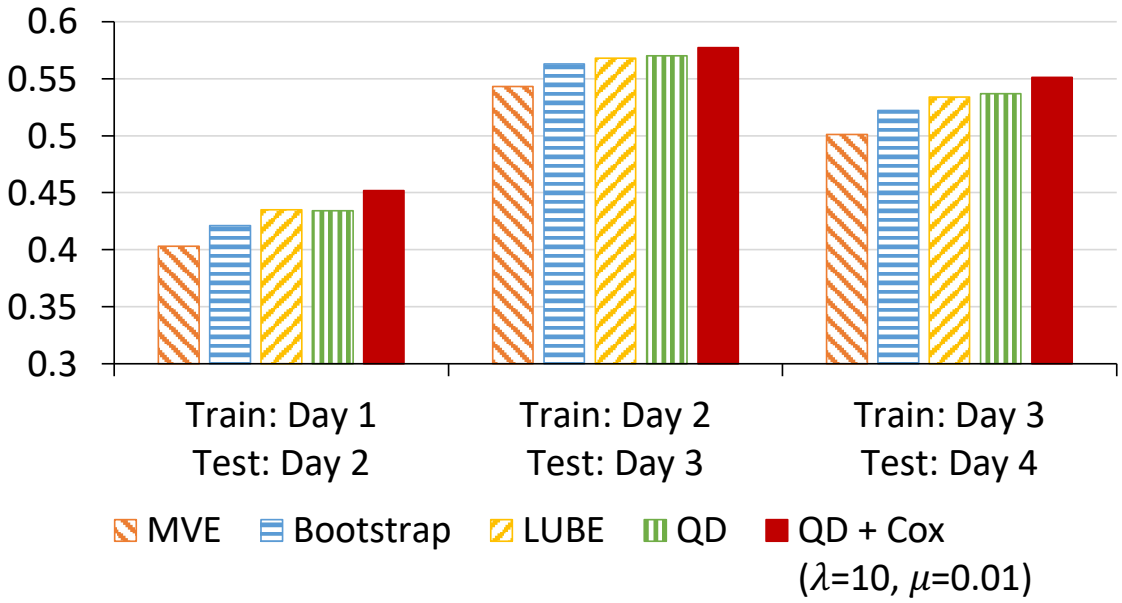

Figure 5.7 The MORPs comparison of lower bound highest bid prediction.

This is because that the required minimum coverage is as low as 60% when $\alpha = 40\%$. Thus, the model has large room to push the predicted lower bounds much higher. In contrast, when $\alpha = 10\%$, to reach the minimum coverage 90%, the model has to largely shrink predicted lower bounds. This causes low MORPs. On the other hand, the expected revenue of $\alpha = 40\%$ is slightly less than that of $\alpha = 30\%$ because PICP of $\alpha = 40\%$ is lower. In other words, when $\alpha = 40\%$, although individual reserve prices are higher, much fewer impressions are outbid.

Therefore, setting α is a trade-off between harvesting many outbid impressions and increasing the unit price of impressions: A high α causes many underbid impressions. However, once an impression gets outbid by advertisers, the publisher can own more revenue. On the other hand, a low α makes many impressions will be sold through real-time bidding, while lower reserve prices may not motivate advertisers to keep their high bids. Real-time bidding algorithms of advertisers may quickly learn that high bids are unnecessary and try to bid lower next time.

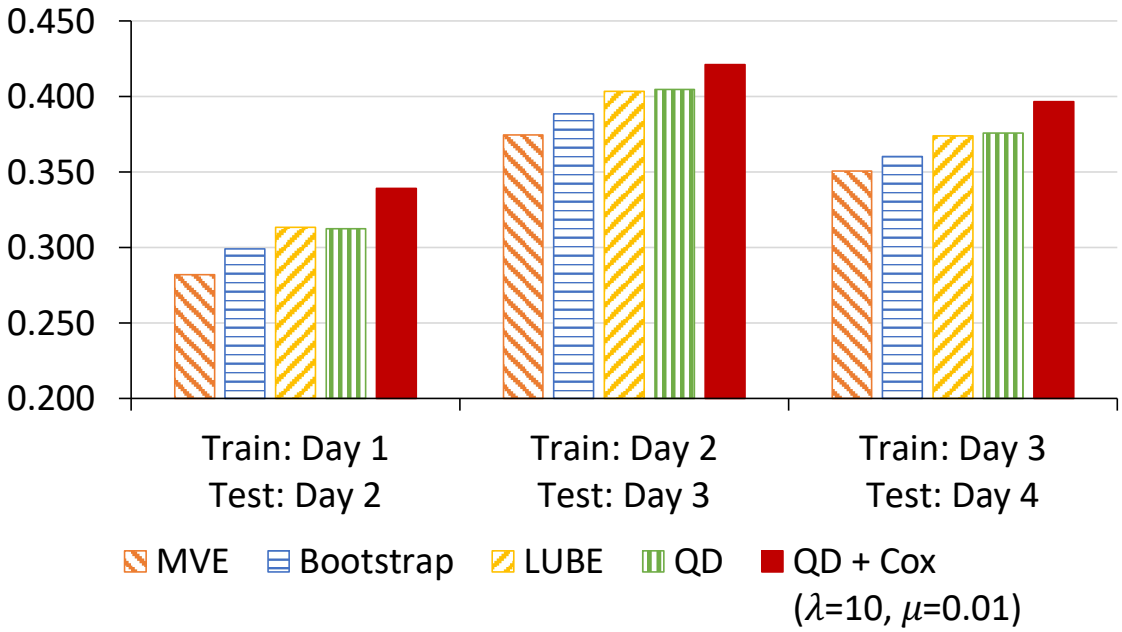

Figure 5.8 The expected revenue comparison of lower bound highest bid prediction.

5.3.8 Performance on Different Sizes of Training Data

This section investigates the impact of the training sizes on the model performance. This may tell publishers how many data they have prepare for model training and how often they have to refresh their models. We vary the training size from one day to three days. Since the total dataset is 4-day, we obtain two groups of data: one uses Day 1 and 2 as the training pool and tests on Day 3; the one uses Day 1-3 as the training pool and tests on Day 4. The model we select is the best model so far we found: QD+Cox ($\lambda = 10, \mu = 0.01$).

Table 5.1 presents that training on one day or two days generates similar results: each has a narrow win on Day 3 and Day 2, respectively. This indicates that publishers (especially those who have limited computational resources) may only need to train on the previous one day to get decent performance out of the proposed model.

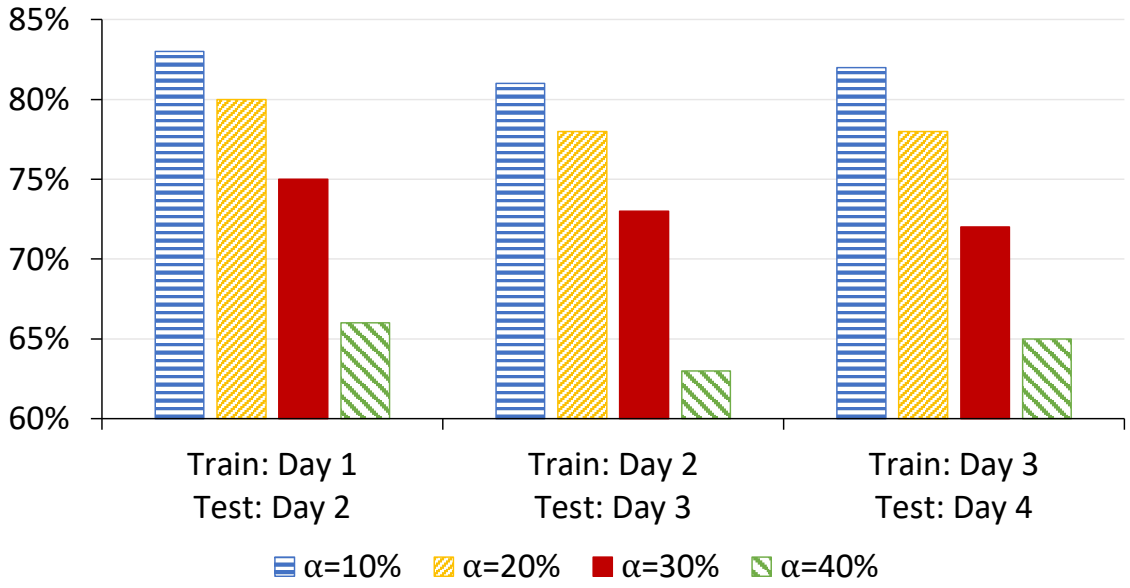

Figure 5.9 PICPs by varying the risk level α.

5.3.9 Performance of Different Neural Network Parameters

We show the impact of the neural network parameters on the performance of lower bound highest bid prediction. We use the best model we find, i.e., QD+Cox($\lambda = 10, \mu = 0.01$). For the sake of simplicity, we report the 3-day average metrics values.

We first vary the embedding width, fixing the number abstraction layers to 2. Table 5.2 shows the performance by varying the width of the embeddings for users, pages, and ad placements. The performance significantly improves with the embedding width being increased from 64 to 128. Wider embeddings than 128 generates subtly worse results.

We then fix the embedding width to 128. Table 5.3 presents the performance by varying the number of abstraction layers. It tells that using more than one abstraction layers generates very similar performance.

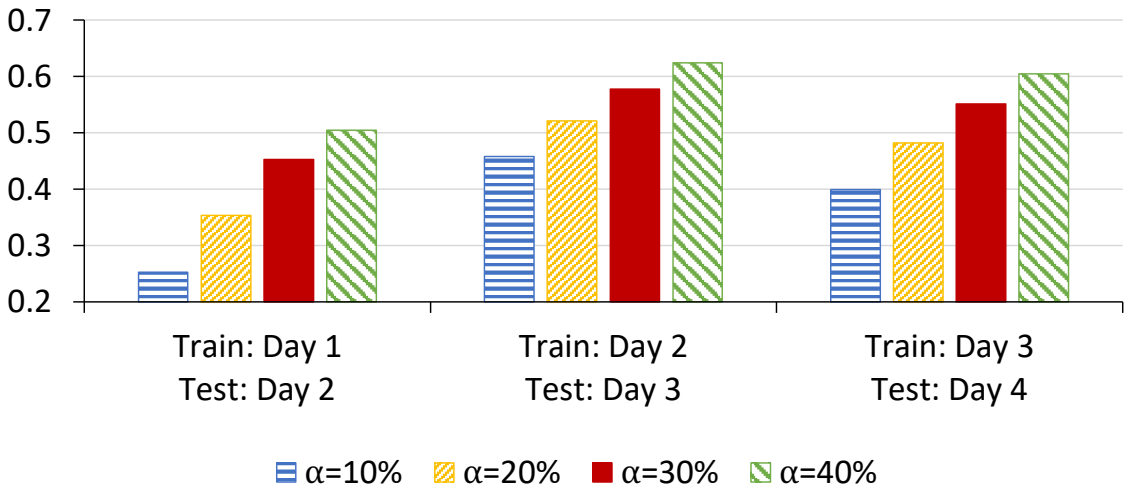

Figure 5.10 MORPs by varying the risk level α.

Table 5.1 Performance of Different Sizes of Training Data

Data Partitions		PICP	MORP	Expected Revenue
Train	Test			
Day 2	Day 3	73%	0.577	0.421
Day 1&2		73%	0.578	**0.422**
Day 3	Day 4	72%	0.551	**0.397**
Day 2&3		72%	0.548	0.395
Day 1&2&3		72%	0.542	0.390

5.4 Summary

The entire display advertising industry has been bracing first-price auctions, in which the advertisers who bid the highest win the ad opportunities. Publishers can know the highest bids if their reserve prices were outbid. Such information can be used to estimate the highest bid of future impressions. The estimation can help publishers find optimal reserve prices which optimize outbid rate and motivate advertisers to bid high in the future. This chapter proposes a multi-task learning framework that predicts the lower bounds of highest bids with a coverage probability $(1-\alpha)\%$. The

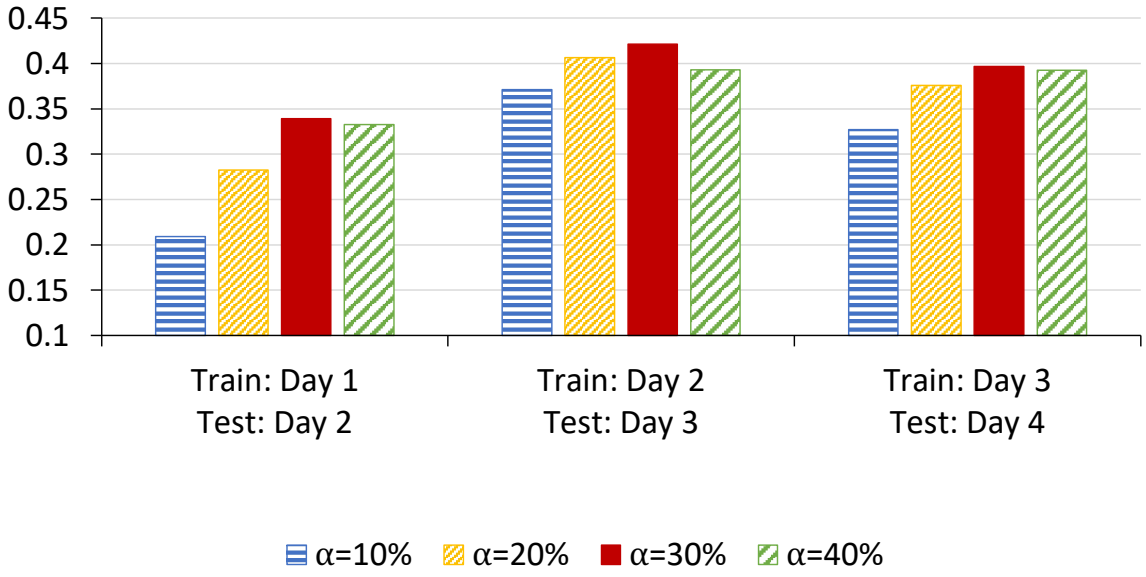

Figure 5.11 Expected revenue by varying the risk level α.

Table 5.2 Performance of Different Embedding Widths

Embedding Width	Mean PICP	Mean MORP	Mean Expected Revenue
64	72%	0.517	0.372
128	73%	0.527	**0.386**
256	74%	0.520	0.385
512	73%	0.522	0.381

lower bounds is expected to be lower than at least $(1-\alpha)\%$ highest bids. Publishers can set the lower bounds as the final reserve prices. In this case, at least $(1-\alpha)\%$ impressions will be successfully outbid by advertisers. The risk level α can be adjusted based on publishers' business strategy. In addition to predict lower bounds, the proposed method also predict hazard rates using Cox PH model in order to utilize historical underbid impressions. The experiments show that the proposed method can significantly outperform the comparison systems.

Table 5.3 Performance of Different Number of Abstraction Layers

#Abstraction layers	Mean PICP	Mean MORP	Mean Expected Revenue
1	72%	0.520	0.374
2	73%	0.527	**0.386**
3	73%	0.523	0.384
4	73%	0.525	0.385

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

The aim of this dissertation is to advance the state-of-the-art research on reserve price optimization in online display advertising. Online display advertising is a million-dollar industry. It is the main revenue source of most online websites. It allows online publishers to continue providing high-quality and free content service to users. A main display ad selling method is real-time bidding: publishers provide reserve prices with impression information and sell their ad impressions through real-time auctions at ad exchanges. There are two popular auctions: second-price auctions and first-price auctions. In second-price auctions, optimal reserve prices can help publishers boost their ad revenue. In first-price auctions, optimal reserve prices can help publishers not only defend minimum ad revenue but also motivate advertisers to keep high bids. Therefore, it is important for publishers to optimize reserve prices in real-time bidding.

Chapter 3 proposes a parametric survival model to predict the failure rate of the reserve price of an online display ad impression in second-price auction. The model is further augmented by user-page pairwise interaction tensor factorization and header bidding factorization. Chapter 4 further augments the previously proposed parametric survival model by a deep neural network (DNN) to capture the feature interaction. The experiments show that the proposed models with the Weibull distribution significantly outperforms a Kaplan-Meier model and a logistic regression with observed reserve price/revenue as the feature. The DNN technique and header bidding regularization further boost performance.

With the entire industry moving from second-price auctions to first-price auctions, Chapter 5 proposes a multi-task learning framework that predicts the lower

bounds of highest bids with a coverage probability $(1-\alpha)\%$. The lower bounds is expected to be lower than at least $(1-\alpha)\%$ highest bids. Publishers can set the lower bounds as the final reserve prices.

As future work, we will deploy the multi-task learning model proposed for first-price auctions in a real business platform. We are going to measure the actual revenue lift.

REFERENCES

[1] Google doubleclick for publishers. https://www.google.com/dfp/, 2018. Retrieval Date: 2021-07-23.

[2] Openx ad server for publishers. https://community.openx.com/s/article/Ad-Server-Report/, 2018. Retrieval Date: 2021-07-23.

[3] Real-time bidding market 2019 global trends, segments, growth, size, regional analysis, landscape and demand by forecast to 2023. https://www.marketwatch.com/press-release/real-time-bidding-market-2019-global-trends-segments-growth-size-regional-analysis-landscape-and-demand-by-forecast-to-2023-2019-05-16, 2019. Retrieval Date: 2021-07-23.

[4] Miguel Angel Alcobendas Lisbona, Sheide Chammas, and Kuang-chih Lee. Optimal reserve prices in upstream auctions: Empirical application on online video advertising. In *2016 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1395–1404, 2016.

[5] Daniel Austin, Sam Seljan, Julius Monello, and Stephanie Tzeng. Reserve price optimization at scale. In *2016 IEEE 3rd International Conference on Data Science and Advanced Analytics (DSAA)*, pages 528–536, 2016.

[6] Ting Bai, Ji-Rong Wen, Jun Zhang, and Wayne Xin Zhao. A neural collaborative filtering model with interaction-based neighborhood. In *the 2017 ACM on Conference on Information and Knowledge Management*, pages 1979–1982, 2017.

[7] Zeynep Batmaz, Ali Yurekli, Alper Bilge, and Cihan Kaleli. A review on deep learning for recommender systems: challenges and remedies. *Artificial Intelligence Review*, 52(1):1–37, 2019.

[8] Nicolo Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. Regret minimization for reserve prices in second-price auctions. *IEEE Transactions on Information Theory*, 61(1):549–564, 2015.

[9] Pedro Chahuara, Nicolas Grislain, Gregoire Jauvion, and Jean-Michel Renders. Real-time optimization of web publisher rtb revenues. In *2017 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1743–1751, 2017.

[10] Liang Chen, Yang Liu, Xiangnan He, Lianli Gao, and Zibin Zheng. Matching user with item set: Collaborative bundle recommendation with deep attention network. In *the 2019 International Joint Conference on Artificial Intelligence*, pages 2095–2101, 2019.

[11] Emmanuel Colliot. Pros and cons of header bidding. https://admetricspro.com/pros-and-cons-of-header-bidding/, 2017. Retrieval Date: 2018-10-22.

[12] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *the 10th ACM conference on recommender systems*, pages 191–198, 2016.

[13] Ying Cui, Ruofei Zhang, Wei Li, and Jianchang Mao. Bid landscape forecasting in online ad exchange marketplace. In *2011 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 265–273, 2011.

[14] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review*, 97(1):242–259, 2007.

[15] eMarketer. Us digital display ad spending to surpass search ad spending in 2016. https://www.emarketer.com/Article/US-Digital-Display-Ad-Spending-Surpass-Search-Ad-Spending-2016/1013442, 2016. Retrieval Date: 2021-07-23.

[16] Patrícia L Espinheira, Silvia LP Ferrari, and Francisco Cribari-Neto. Bootstrap prediction intervals in beta regressions. *Computational Statistics*, 29(5):1263–1277, 2014.

[17] Cheng Guo and Felix Berkhahn. Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*, 2016.

[18] Frank E Harrell. Regression modeling strategies, with applications to linear models, survival analysis and logistic regression. *New York, NY: Springer*, 2001.

[19] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 355–364, 2017.

[20] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *the 26th international conference on world wide web*, pages 173–182, 2017.

[21] Tom Heskes. Practical confidence and prediction intervals for prediction tasks. *Progress in Neural Processing*, pages 128–135, 1997.

[22] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. Collaborative metric learning. In *the 26th international conference on world wide web*, pages 193–201, 2017.

[23] Jianming Hu, Yingying Lin, Jingwei Tang, and Jing Zhao. A new wind power interval prediction approach based on reservoir computing and a quality-driven loss function. *Applied Soft Computing*, 92:106327, 2020.

[24] Grégoire Jauvion, Nicolas Grislain, Pascal Dkengne Sielenou, Aurélien Garivier, and Sébastien Gerchinovitz. Optimization of a ssp's header bidding strategy using thompson sampling. In *2018 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 425–432, 2018.

[25] Jian Ji, Yong Sun, Fandong Kong, and Qiguang Miao. A construction approach to prediction intervals based on bootstrap and deep belief network. *IEEE Access*, 7:124185–124195, 2019.

[26] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. Field-aware factorization machines in a real-world online advertising system. In *2017 International World Wide Web Conference*, pages 680–688. International World Wide Web Conferences Steering Committee, 2017.

[27] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. Field-aware factorization machines for ctr prediction. In *the 10th ACM Conference on Recommender Systems*, pages 43–50, 2016.

[28] Achir Kalra, Chong Wang, Cristian Borcea, and Yi Chen. Reserve price failure rate prediction with header bidding in display advertising. In *2019 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[29] Abbas Khosravi, Ehsan Mazloumi, Saeid Nahavandi, Doug Creighton, and JWC Van Lint. Prediction intervals to account for uncertainties in travel time prediction. *IEEE Transactions on Intelligent Transportation Systems*, 12(2):537–547, 2011.

[30] Abbas Khosravi and Saeid Nahavandi. An optimized mean variance estimation method for uncertainty quantification of wind power forecasts. *International Journal of Electrical Power & Energy Systems*, 61:446–454, 2014.

[31] Abbas Khosravi, Saeid Nahavandi, and Doug Creighton. Prediction interval construction and optimization for adaptive neurofuzzy inference systems. *IEEE Transactions on Fuzzy Systems*, 19(5):983–988, 2011.

[32] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE Transactions on Neural Networks*, 22(3):337–346, 2010.

[33] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on Neural Networks*, 22(9):1341–1356, 2011.

[34] Abbas Khosravi, Saeid Nahavandi, Dipti Srinivasan, and Rihanna Khosravi. Constructing optimal prediction intervals by using neural networks and bootstrap method. *IEEE Transactions on neural networks and learning systems*, 26(8):1810–1815, 2014.

[35] David G Kleinbaum and Mitchel Klein. *Survival Analysis*. New York, NY: Springer, 2010.

[36] David G Kleinbaum and Mitchel Klein. Introduction to survival analysis. In *Survival Analysis*, pages 1–54. New York, NY: Springer, 2012.

[37] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *the 14th ACM SIGKDD International Conference on Knowledge discovery data mining*, pages 426–434, 2008.

[38] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[39] Anabelle Laurent, Fernando Miguez, Peter Kyveryga, and David Makowski. Going beyond mean effect size: Presenting prediction intervals for on-farm network trial analyses. *European Journal of Agronomy*, 120:126127, 2020.

[40] Juanjuan Li, Xiaochun Ni, Yong Yuan, Rui Qin, Xiao Wang, and Fei-Yue Wang. The impact of reserve price on publisher revenue in real-time bidding advertising markets. In *2017 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1256–1261, 2017.

[41] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[42] Fangjie Liu, Chaoshun Li, Yanhe Xu, Geng Tang, and Yuying Xie. A new lower and upper bound estimation model using gradient descend training method for wind speed interval prediction. *Wind Energy*, 2020.

[43] Brendan Lucier, Renato Paes Leme, and Eva Tardos. On revenue in the generalized second price auction. In *2012 International World Wide Web Conference*, pages 361–370, 2012.

[44] Tullio Mancini, Hector Calvo-Pardo, and Jose Olmo. Prediction intervals for deep neural networks. *arXiv preprint arXiv:2010.04044*, 2020.

[45] Nigel Meade and Towhidul Islam. Prediction intervals for growth curve forecasts. *Journal of Forecasting*, 14(5):413–430, 1995.

[46] Andres M Medina and Mehryar Mohri. Learning theory and algorithms for revenue optimization in second price auctions with reserve. In *2014 International Conference on Machine Learning*, pages 262–270, 2014.

[47] Ruihui Mu. A survey of recommender systems based on deep learning. *IEEE Access*, 6:69009–69022, 2018.

[48] Andres Munoz and Sergei Vassilvitskii. Revenue optimization with approximate bid predictions. In *2017 The Conference and Workshop on Neural Information Processing Systems*, pages 1858–1866, 2017.

[49] David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *the 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pages 55–60, 1994.

[50] Michael Ostrovsky and Michael Schwarz. Reserve prices in internet advertising auctions: A field experiment. In *2011 ACM Conference on Economics and Computation*, pages 59–60, 2011.

[51] Li Pan and Dimitris N Politis. Bootstrap prediction intervals for linear, nonlinear and nonparametric autoregressions. *Journal of Statistical Planning and Inference*, 177:1–27, 2016.

[52] Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *International Conference on Machine Learning*, pages 4075–4084. PMLR, 2018.

[53] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, 3(3):57, 2012.

[54] Steffen Rendle and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *2010 ACM International Conference on Web Search and Data Mining*, pages 81–90, 2010.

[55] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender Systems Handbook*, pages 1–35. New York, NY: Springer, 2011.

[56] Christoph Roser and Masaru Nakano. Confidence intervals from a single simulation using the delta method. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 46(1):67–72, 2003.

[57] Tárik S Salem, Helge Langseth, and Heri Ramampiaro. Prediction intervals: Split normal mixture from quality-driven deep ensembles. In *Conference on Uncertainty in Artificial Intelligence*, pages 1179–1187. PMLR, 2020.

[58] AR Shafay and N Balakrishnan. One-and two-sample bayesian prediction intervals based on type-i hybrid censored data. *Communications in Statistics-Simulation and Computation*, 41(1):65–88, 2012.

[59] Rifat Sonmez. Range estimation of construction costs using neural networks with bootstrap prediction intervals. *Expert systems with applications*, 38(8):9913–9917, 2011.

[60] Michael Sweeney. How real-time bidding (rtb) changed online display advertising. `https://clearcode.cc/blog/real-time-bidding-online-display-advertising/`, 2014. Retrieval Date: 2021-07-23.

[61] Olga Vigiak and Ulrike Bende-Michl. Estimating bootstrap and bayesian prediction intervals for constituent load rating curves. *Water Resources Research*, 49(12):8565–8578, 2013.

[62] Chong Wang, Achir Kalra, Cristian Borcea, and Yi Chen. Webpage depth-level dwell time prediction. In *the 25th ACM International on Conference on Information and Knowledge Management*, pages 1937–1940, 2016.

[63] Chong Wang, Achir Kalra, Li Zhou, Cristian Borcea, and Yi Chen. Probabilistic models for ad viewability prediction on the web. *IEEE Transactions on Knowledge and Data Engineering*, 29(9):2012–2025, 2017.

[64] Chong Wang, Shuai Zhao, Achir Kalra, Cristian Borcea, and Yi Chen. Predictive models and analysis for webpage depth-level dwell time. *The Journal of the Association for Information Science and Technology*, 2018.

[65] Chong Wang, Shuai Zhao, Achir Kalra, Cristian Borcea, and Yi Chen. Webpage depth viewability prediction using deep sequential neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(3):601–614, 2018.

[66] Ping Wang, Yan Li, and Chandan K. Reddy. Machine learning for survival analysis. In *ACM Computing Surveys*, volume 1, pages 1–38, 2017.

[67] Yuchen Wang, Kan Ren, Weinan Zhang, Jun Wang, and Yong Yu. Functional bid landscape forecasting for display advertising. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 115–131. New York, NY: Springer, 2016.

[68] Wush Wu, Mi-Yen Yeh, and Ming-Syan Chen. Deep censored learning of the winning price in the real time bidding. In *2018 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2526–2535, 2018.

[69] Wush Chi-Hsuan Wu, Mi-Yen Yeh, and Ming-Syan Chen. Predicting winning price in real time bidding with censored data. In *2015 ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1305–1314, 2015.

[70] Zhihui Xie, Kuang-Chih Lee, and Liang Wang. Optimal reserve price for online ads trading based on inventory identification. In *The 2017 AdKDD Workshop in conjunction with The 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, page 6, 2017.

[71] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *2017 International Joint Conference on Artificial Intelligence*, volume 17, pages 3203–3209. Melbourne, Australia, 2017.

[72] Shuai Yuan, Jun Wang, Bowei Chen, Peter Mason, and Sam Seljan. An empirical study of reserve price optimisation in real-time bidding. In *2014 ACM*

SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1897–1906, 2014.

[73] Weinan Zhang, Tianxiong Zhou, Jun Wang, and Jian Xu. Bid-aware gradient descent for unbiased learning with censored data in display advertising. In *the 22nd ACM SIGKDD International Conference on Knowledge discovery data mining*, pages 665–674, 2016.

[74] Hua Zhong and Li Xu. An all-batch loss for constructing prediction intervals. *Applied Sciences*, 11(4):1728, 2021.

[75] Wen-Yuan Zhu, Wen-Yueh Shih, Ying-Hsuan Lee, Wen-Chih Peng, and Jiun-Long Huang. A gamma-based regression for winning price estimation in real-time bidding advertising. In *IEEE Big Data'17*, 2017.