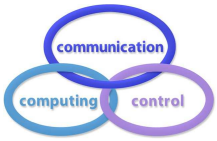
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 15, Issue: 2, Month: April, Year: 2020
Article Number: 3760, https://doi.org/10.15837/ijccc.2020.2.3760



CCC Publications

A Deep Choice Model for Hiring Outcome Prediction in Online Labor Markets

Y. Ma, Z. Zhang, A. Ihler

Yixuan Ma
1. School of Management and Economics
Beijing Jiaotong University, Beijing, 100044, China
2. International Center for Informatics Research
Beijing Jiaotong University, 100044, China
3. Department of Computer Science
University of California Irvine, 92617, USA
mayixuan@bjtu.edu.cn

Zhenji Zhang*
1. School of Management and Economics
Beijing Jiaotong University, Beijing, 100044, China
2. International Center for Informatics Research
Beijing Jiaotong University, 100044, China
*Corresponding author: zhjzhang@bjtu.edu.cn

Alexander Ihler
Department of Computer Science
University of California Irvine, CA, 92617, USA
ihler@ics.uci.edu

Abstract

A key challenge faced by online labor market researchers and practitioners is to understand how employers make hiring decisions from many job bidders with distinct attributes. This study investigates employer hiring behavior in one of the largest online labor markets by building a data-driven hiring decision prediction model. With the limitation of traditional discrete choice model (conditional logit model), we develop a novel deep choice model to simulate the hiring behavior from 722,339 job posts. The deep choice model extends the classical conditional logit model by learning a non-linear utility function identically for each bidder within of the job posts via a point-wise convolutional neural network. This non-linear mapping can be straightforwardly optimized using stochastic gradient approach. We test the model on 12 categories of job posts in the dataset. Results show that our deep choice model outperforms the linear-utility conditional logit model in predicting hiring preferences. By analyzing the model using dimensionality reduction and sensitivity analysis, we highlight the nonlinear combination of bidders' features in impacting employers' hiring decisions.

Keywords: deep choice model, hiring decision, convolutional neural network, conditional logit model, online labor markets

1 Introduction

With rapid growth of the Internet, an increasing portion of the workforce for professional services have migrated to online labor markets (OLMs). The global annual gross market size of the OLM was approximately $4.8 billion in 2016, with a total number of workers reaching approximately 48 million [21]. In U.K., approximately 11% of adults in U.K. were "freelancing", with 3% doing so at least weekly [16]. The thriving OLM economy is supported by a number of growing Internet platforms that serve to bridge connections between labor demands and supplies, such as Upwork.com and Freelancer.com. By 2018, 1.8 billion users were reported to be active on Upwork.com [44] and 31 million people had registered accounts on Freelancer.com [43].

Compared to the conventional labor market, OLMs have some peculiar features. First, the hiring processes are project-oriented, rather than based on a long-term working contract. For a typical hiring process on an OLM platform, the employer posts a job and receives bidders' applications. The hiring decision is made based on the bidders' attributes and the employer's requirements. After the work is completed, the selected bidder receives a reward and review based on his/her work performance. This project-oriented business mode greatly facilitates efficient matching of labor demands and supplies. Another characteristic feature of the OLM is the transparency of information to both employers and workers. The employers can reach a larger pool of applicants across geographical boundaries, and have access to more information on each applicant, such as skills, reputation, and work experience [5]. For the workers, they can adjust their bid price based on the other applicants' offers to make themselves more competitive in the bidding pool. The workers may also be guided to develop skills that are required for their preferred work.

A key question for OLM researchers and practitioners is to understand employers' hiring preferences. From a bidder's viewpoint, a realistic estimate of employer's hiring behavior sets up a benchmark for bidders to evaluate their competitiveness. The bidders can thus adjust their bidding offers to gain higher hiring opportunities. From an employer's viewpoint, a clarification of hiring mechanism helps them to easily access and focus on preferred bidders. Also, the OLM serving platform benefits from realistic estimates of hiring preferences for building efficient job recommendation systems.

Modeling the selection process from a pool of bidders is a classical choice problem in economics [33]. A common strategy for modeling choice behavior is to construct a utility function that summarizes different aspects of a bidder's attributes. The choices are then made based on a ranking of the bidders' utility function values. Conventionally, the utility function is composed of an optimized linear combination of the bidder's attributes. Such models can be easily implemented and offer high interpretability about the contribution of different attributes of the bidders.

Despite the appeal of such choice models, many researchers have pointed out that real-world hiring processes involve complex considerations regarding various aspects of the bidders, which cannot be adequately represented by simple linear models [35, 36, 39]. To capture non-linear impacts in hiring decision making, many machine learning based approaches have been introduced. Some of these approaches provide higher predictive accuracy, albeit at the expense of interpretability [26, 36]. Generally, it remains a question how machine learning based models can provide insights in understanding the hiring choices.

The purpose of this study is to make use of the rich online labor market data to improve our understanding of hiring decision making. To this end, we collected 1.4 million job posts from Freelancer.com. The posts range from January 2000 to March 2017. Among them, 722,339 job posts (48.6%) had responses by two or more bidders with a single winner fulfilling the project. Each of the fulfilled job posts provides a real-world hiring sample. With such abundant samples, existing discrete choice model, such as conditional logit model, can not reveal fine patterns of hiring behavior, and we need investigate hiring decision making processes in a data-driven manner. In this paper, we propose a deep choice model (DCM) that takes advantage of both classical economic choice models and machine learning based approaches. In this model, deep convolutional neural networks are applied to extend the conditional logit model by learning a non-linear utility function with various input. Our method combines the strengths of machine learning methods, such as flexible representations and improved accuracy, with those of choice models, such as interpretability and the ability to handle variable bid pool sizes.

The rest of the paper is organized as follows: Section 2 briefly reviews related work. Section 3 describes problem formulation. In Section 4, we present our deep choice model and its implementation using a deep neural network framework. The experiments in Section 5 demonstrate the out-performance of our proposed model, and we lay out some conclusions in Section 6.

2 Related Works

2.1 Hiring Decisions in Online Labor Markets

Online labor markets have recently become an active research topic, and related research spans a variety of problems. Among these problems, one of the most fundamental and challenging is to understand hiring behavior. Research related to the hiring process has long been constrained by expensive survey or administrative data [15]. Such data offer recruitment decisions associated with jobs, but it is often impossible to find the information for all candidates who applied for the job. With the digitization of labor activities, researchers can access cost-free and rich data sets, providing unique opportunities to analyze the hiring process and enhance our understandings of labor economics. Many factors can influence hiring decisions in online labor markets. A number of studies have been devoted to investigating how these factors affect hiring decisions or reveal hiring biases. For example, employers place significant weight on applicants' reputation [42], and are willing to accept more expensive bids if posted by more reputable bidders [34]. Researchers also found that workers from developing countries are less likely to be hired compared to developed countries [2]. On the other hand, Kokkodis et al. [19] analyzed freelancers' hourly billing, bidding rate, certifications, education, self-reported experience, and other factors. These many studies suggest that freelancer hiring decisions may arise from complex interactions between many worker and bid attributes, and suggest a number of attributes should be included when building a hiring prediction model.

2.2 Modeling Approaches

Machine learning approaches. To understand how various factors affect hiring decisions, we can fit a variety of models using techniques from statistics and machine learning. Machine learning approaches have made significant progress in various fields by learning to approximate general functions from training data examples [9, 28, 29, 31, 37, 40]. Applied directly to model choices, these can be framed as various classical machine learning tasks, such as binary classification, multi-label classification, or preference learning.

Treating hiring prediction as a binary classification (hire/no-hire) problem frames hiring in a standard supervised learning format, to which researchers have applied binary logit models [19] or a variety of others (naíve Bayes, logistic regression, linear SVM, decision trees and random forests) [1] to predict hiring probability. However, this framework assumes that each applicant's decision in the bidding pool is independent of the others. Unfortunately, this assumption is unrealistic. Hiring one individual clearly depends on the pool of available alternatives, since an employer considers the other options in the application pool before making a final decision. Choosing the winner means giving up the other applicants.

Some researchers view choice problems as multi-label classification tasks, in which each class label corresponds to one of the choices. Then models, such as random forests or neural networks, are trained to predict the outcome of a new decision maker based on the labeled choices of previous decision makers and their associated characteristics. Unfortunately, this approach is not a good fit to many choice problems, since a key assumption of the approach is that each decision has the same, fixed set of possible outcomes. In some settings, such as decisions about which transportation mode to select [12, 13, 38], this assumption holds and a multi-label classifier can work well. However, in online labor hiring, each employer faces a completely different pool of bidders, varying in size from 2 up to hundreds, making it difficult to apply a standard classification framework.

Discrete choice models. These drawbacks to classical machine learning frameworks have made discrete choice models a preferred method to predict hiring decisions [3]. Discrete choice models posit that employers behave rationally, choosing the worker that maximizes their utility, then attempts

to fit a utility function to explain these choices by optimizing a probability model. This framework has the advantage that, although it estimates an individual quantity to each worker for a particular job, its prediction depends on the entire pool, making it a better fit to hiring decisions. Most of the aforementioned work on hiring (e.g., Section 2.1) apply discrete choice models to hiring problems, including the classic conditional logit model (CLM) [2, 6, 34], binary logit models [19], and mixed logit models [25]. However, crucially, these works use very simple, linear utility function models. In reality, the hirer utility is likely to be a more complex function of the job and worker attributes.

3 Problem Formulation

In this section, we describe our problem denotations, along with the classical conditional logit model. For a specific project, indexed by k, we have m bidders, where each bidder is characterized by n attributes. We represent the bidding pool as a $m \times n$ matrix $\mathbf{X}^k$:

$$\mathbf{X}^k = \begin{bmatrix} \mathbf{x}_1^k \\ \mathbf{x}_2^k \\ \vdots \\ \mathbf{x}_m^k \end{bmatrix} = \begin{bmatrix} x_{11}^k & x_{12}^k & ... & x_{1n}^k \\ x_{21}^k & x_{22}^k & ... & x_{2n}^k \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1}^k & x_{m2}^k & ... & x_{mn}^k \end{bmatrix} \tag{1}$$

Note that the bidding pool size m may vary with the job k, but we suppress this dependence for notational convenience. Each row of $\mathbf{X}^k$ corresponds to the attributes of one bidder, and each column represents one attribute across the pool of bidders. Our goal is to predict and explain the employers' choice decisions based the information provided in the bidding pools.

We assume that employers make rational choices among the bidders, i.e., that each bidder is able to provide some utility U_i^k to the employer, and that employer will act to select the bidder that provides the highest utility. Since not all employers will behave identically given the same pool of bidders, we assume that the utility involves some unknown aspects, modeled as noise ε, so that the true utility U_i^k is

$$U_i^k = U(\mathbf{x}_i^k; \boldsymbol{\beta}) + \varepsilon_i \tag{2}$$

For the conditional logit model, the deterministic component of the utility is assumed to be a linear function of the bidder's attributes,

$$U(\mathbf{x}_i^k; \boldsymbol{\beta}) = \boldsymbol{\beta}\mathbf{x}_i^k. \tag{3}$$

Here, $\boldsymbol{\beta} = [\beta_1, \beta_2, ...\beta_n]^T$, and each parameter β_ℓ can be viewed as representing the importance of the ℓth attribute to the employer's utility, by linearly scaling the observed features, while the noise ε_i captures the impact of all unobserved factors that affect the employer's choice. Moreover, if the ε for different bidders are i.i.d. random variables from a Gumbel distribution, MacFadden et al. [32] showed that the probability of an employer selecting bidder $\mathbf{b}_i^k$ is given by a "softmax" form,

$$P(\mathbf{b}_i^k | \mathbf{X}^k) = \frac{\exp(\boldsymbol{\beta}\mathbf{x}_i^k)}{\sum_{j=1}^m \exp(\boldsymbol{\beta}\mathbf{x}_j^k)} \tag{4}$$

This allows us to view the task of learning the employers' utility parameters $\boldsymbol{\beta} = [\beta_1, \beta_2, ...\beta_n]^T$ as a classical maximum likelihood learning problem,

$$\underset{\boldsymbol{\beta}}{\text{maximize}} \sum_k \log(P(\mathbf{b}_{i_k}^k | \mathbf{X}^k)) \tag{5}$$

$$\text{where} \quad P(\mathbf{b}_i^k | \mathbf{X}^k) \propto \exp(\boldsymbol{\beta}\mathbf{x}_i^k) \tag{6}$$

with i_k denoting the index of the bid selected by the employer for project k. The conditional logit model is a classical choice model that has been analyzed theoretically and used in online labor markets [2, 6, 34], and extensive applications as well [4, 14, 32].

The concept of the utility function in conditional logit models offers a powerful tool for measuring preferences within a choice set. The utility function works by summarizing the various attributes of a bidder to reflect his or her competitiveness in the bidding pool. By analyzing the contribution of different attributes on the utility function value, we can quantify the importance of those attributes, helping to explain the model or diagnose its results.

However, the problem of conditional logit model is that the utility takes a linear parametric form. While this enables the models to be easily interpreted and contributes to their wide use, human decisions are likely far more complicated than a simple linear weighting [18]. When only a few data are available, such linear models may still be relatively effective; but the recent exponential growth of data suggests that we have enough information to learn more flexible, more complex and nonlinear utility functions that can potentially better capture the human decision process.

The key question for developing a non-linear utility function lies in specifying 1) the parametric form for the utility function, and 2) the objective function and training process. For a given project, we need to apply the same utility function to each bidder's feature vector within the bidding pool. To train the model to predict accurately, the objective function should highlight the utility of the selected bidder compared to the competing bidders in that pool. This requires us to explicitly regulate the way in which model acts on the structured data.

A very popular flexible model for dealing with structured data using non-linear functions is the convolutional neural network (CNN). As a particular class of neural network models, CNNs have been frequently applied for modeling various types of structured data, such as images, video, speech and audio [23]. Recently, deep CNNs have led to many of the most significant innovations in these fields [20].

Here we use a particular structure of CNN, called point-wise CNN, to learn a complex non-linear utility function applied identically to each bidder within the same project. We demonstrate in the sequel why this specific form of deep neural network is a good fit for our problem and how it can be applied to improve choice prediction accuracy while offering insight into the hiring choice process.

4 A Deep Choice Model

4.1 Convolutional Neural Networks

Neural networks are learning machines composed of a hierarchy formed of layers of connected computation units called *neurons*. Each neuron receives a set of signals from the previous layer, performs a simple transformation (a weighted sum, followed by a scalar nonlinear function called an *activation function*), and passes it to the next layer of neurons. With the composition of many simple but non-linear transformations through hierarchical layers of neurons, the network can represent complex functions, and internal neurons' responses can be interpreted as learning alternative representations or features of the data. The model's parameters are usually trained using (stochastic) gradient descent, often called *backpropagation* in the neural network literature.

Neural network models are a flexible framework, and neurons can be arranged to compose customized computing graphs that match specific learning tasks.

In many settings such as natural language, images, audio, and video, convolutional neural networks are a preferred variant, since they can explicitly encode some spatial or temporal invariance in the learned features. In lieu of individual neurons, convolutional networks define *channels* of neurons, which all share the same convolutional kernel (weights) and correspond to the response of that kernel at different shifts in the input data.

As a concrete example, consider a 2D image with $c = 3$ channels (for example, RGB color values). A single channel of the first layer would be defined by a convolutional kernel that takes in an $r_1 \times r_2$ region of the input image (called the *receptive field* of the kernel) and computes a weighted sum; this kernel can then be applied at each possible offset in the input image. If we have c' such channels, we can represent the parameters of this layer as a tensor $\mathbf{K}$ of size $c' \times c \times r_1 \times r_2$, and the channels'

response at any offset (a, b) is the convolution

$$\mathbf{O}[a,b] = (\mathbf{I} * \mathbf{K})[a,b] = \sum_{u=1}^{r_1}\sum_{v=1}^{r_2}(\mathbf{K}[u,v] \odot \mathbf{I}[a+u, b+v]) \tag{7}$$

Convolution layers are then interleaved with nonlinear activation function layers, for example, hyperbolic tangent or rectified linear unit (ReLU) transformations, or any of a number of other common building blocks.

Each building block is differentiable, so that backpropagation can be applied to efficiently compute the derivative with respect to each of the model parameters (the weights of the kernels $\mathbf{K}$) and optimize them.

While CNNs were developed and popularized for 2D image recognition tasks [24], the idea is easily applied to many types of array-form data of different dimensions, such as 1D sequence data [22] and 3D volumetric data [17]. By changing the shape of $\mathbf{K}$, we can carry out a spatial-only or *depthwise* convolution independently over each channel of an input, or perform *pointwise* convolution (convolution with receptive field size 1) to merge different channels of features at a same spatial position [8, 27].

We are particularly interested in point-wise convolution, given that it integrates multiple features while preserving the spatial structure of the data. For a 2D CNN, a point-wise convolution is also known as a 1×1 convolution, which has been applied for feature synthesis in [27] and many subsequent works. We illustrate how to use point-wise convolution to synthesize bidders' attributes and compose a non-linear utility function for predicting hiring.

4.2 Point-wise Convolution for Learning Utility

Consider a bidding pool represented by a $m \times n$ matrix, where there are m bidders, each with n attributes. We propose to fit a non-linear utility function that maps each bidder's attribute vector to their estimated utility. To do so, we process the bidding pool matrix using a sequence of computation blocks that composed of point-wise convolutions and nonlinear activation functions. The pointwise property ensures that the same nonlinear function is applied identically to each bidder, defining a fully connected neural network architecture that maps each individual bidder's attributes to a scalar utility value.

More precisely, starting from the bidding pool in the form of the $m \times n$ matrix $\mathbf{X}^k$, we treat the m bidders as m spatial points, with each point holding n channels of attributes. We slide through each bidder point using a point-wise convolution kernel with input channel size of n and output channel size of n_1. Mathematically, this is equivalent to performing a matrix multiplication on $\mathbf{X}^k$ with a $n \times n_1$ kernel matrix. We make this transformation nonlinear by including an element-wise nonlinear activation function f. The result Y_1^k is an $m \times n_1$ matrix:

$$\mathbf{Y}_1^k = f \odot (\mathbf{X}^k \cdot [\boldsymbol{\beta}_1^1, \boldsymbol{\beta}_1^2, \ldots, \boldsymbol{\beta}_1^{n_1}]) \tag{8}$$

Here, $[\boldsymbol{\beta}_1^1, \boldsymbol{\beta}_1^2, \ldots, \boldsymbol{\beta}_1^{n_1}]$ defines the point-wise convolution kernel. We note that after the transformation, each row of the matrix $\mathbf{Y}_1^k$ corresponds to a new attribute vector of a single bidder, and each column represents one such new attribute across the bidding pool.

Since the operation preserves the organization of the bidding pool, we can sequentially apply it multiple times to build up a more complex function that independently synthesizes each bidder's attributes:

$$\mathbf{Y}_{i+1}^k = f \odot (\mathbf{Y}_i^k \cdot [\boldsymbol{\beta}_{i+1}^1, \boldsymbol{\beta}_{i+1}^2, \ldots, \boldsymbol{\beta}_{i+1}^{n_{i+1}}]), i = 1, 2, \ldots \tag{9}$$

Here $\mathbf{Y}_{i+1}^k$ is the $(i+1)$th layer's representation of the bidders' attributes; $[\boldsymbol{\beta}_{i+1}^1, \boldsymbol{\beta}_{i+1}^2, \ldots, \boldsymbol{\beta}_{i+1}^{n_{i+1}}]$ is the $(i+1)$th layer point-wise convolution kernel.

For the final layer $\mathbf{Y}_{-1}^k$, we restrict $n_{-1} = 1$ and eliminate the non-linearity:

$$\mathbf{Y}_{-1}^k = \mathbf{Y}_{-2}^k \cdot \boldsymbol{\beta}_{-1}^1 \tag{10}$$

$\mathbf{Y}^k_{-1}$ is of dimension $m \times 1$. As in the conditional logit model, we apply a softmax function to normalize the vector into a probability distribution, with each element corresponding to the estimated hiring probability for the that bidder.

Thus, the sequential application of point-wise convolutions and nonlinear activations actually composes a small fully connected neural network, which is mapped independently to each bidder's attribute vector, while defining a network on the full bidding pool matrix. We call this model a deep choice model. If the point-wise convolution is applied only once, with no nonlinearity, the deep choice model reduces to the classical conditional logit model. On the other hand, by stacking several layers of simple but non-linear transformations, we can potentially learn very complex functions to better capture employers' hiring decisions.

4.3 Training and Model Regularization

We train the deep choice model by maximizing the choice decision likelihood as defined in Equation (5). We apply backpropagation to optimize the kernel parameters: the gradient component of the likelihood function with respect to the kernel weights can be derived by applying the chain rule to the computation graph, and we iteratively adjust the kernel parameters via stochastic gradient to maximize the likelihood function.

For implementation convenience, we make use of available deep neural network libraries to implement our deep choice model. Most neural network libraries provide off-the-shelf 2D convolutional neural network modules. As is shown in Figure 1, to adapt the 2D CNN modules for our setting, we first reshape the $m \times n$ bidding pool matrix to size $n \times m \times 1$, with n representing the channel size, and $m \times 1$ representing the spatial domain. Next, we sequentially apply 1×1 convolution and a nonlinear activation, which slides through each "spatial point" (bidder) to compute that bidder's next layer of attributes. We then normalize the final layer's output with a softmax function, and train the model using backpropagation in the library's optimizer.

Since the deep choice model includes multiple layers of point-wise convolutions and thus may have many parameters, there is significant potential for *overfitting*, in which a model fits the data used for training well but performs badly on an independent test set. Regularization methods refer to approaches that prevent model overfitting and help the model generalize better to unseen data. Modern regularization approaches for conventional convolutional neural networks can be directly applied to our deep choice model.

In this work we employ a popular regularization approach, called Dropout [41]. Dropout randomly censors (or zeros out) the responses of each neuron with some probability $1 - p$ during training. Intuitively, this prevents the model from relying too much on any particular neuron's response values, encouraging redundancy in the learned representation. To compensate for the dropout effect applied during training, at test time all neurons are included, but multiplied by the probability of non-censoring, p. Dropout has shown good performance in reducing overfitting in many settings.

5 Experiments

5.1 Dataset and Variables

Dataset. The dataset we use for experiments are gathered from one of largest online labor marketplace, Freelancer.com through open API. We collected 1,486,801 job posts ranging from January 2000 to March 2017. To investigate hiring behavior, we focus on projects to which at least two bidders respond, with one winner who successfully completed the project. After data cleaning and filtering on these conditions, $722,339$ valid hiring samples are left for use.

Considering the fact that different categories of jobs may have distinct hiring behaviors, we further divide this set of valid projects into 12 categories. Figure 1 shows the number of projects falling into each category.

Attributes that influence hiring decisions. For each project in the data set, the hiring decision might be impacted by the bidders' intrinsic attributes, the bidder's commitment and bidding price, as well as how the bidders' skills and experiences match the project requirements. To model

Table 1: Project Amount for Each Category

Abbr.	Full Name	Counts
WIS	Website, IT and Software	303,802
DMA	Design, Media and Architecture	179,955
WC	Writing and Content	123,977
DEA	Data Entry and Admin	35,811
SM	Sales and Marketing	31,901
MPC	Mobile Phones and Computing	18,987
ES	Engineering and Science	12,230
BAHRL	Business, Accounting, Human Resources and Legal	7,448
O	Others	5,680
TL	Translation and Languages	1,601
PSM	Product Sourcing and Manufacturing	702
LJS	Local Jobs and Services	245

this decision making process, we quantify the relevant attributes that have potential influence on the hiring decision. A summary of all variables, their descriptions and method of computation are given in Table 2.

Table 2: Summary of Variables

Variable	Description
Bidder Attributes	
Membership	The worker's membership plan. Encoding as: Free-1, Intro-2, Basic-3, Plus (Starter)-4, Professional (Standard)-5, Premier (Premium)-6[a] ([1-6])
Verification Completeness	Ratio of worker's completed verification to total number of verification. Verification contains payment, email, phone, Facebook, deposit and profile completion. ([0-1])
Earning Score	Amount earned doing projects in certain skill or category. Increased as projects are successfully completed and paid through the platform. ([0-10])
On-time Rate	Ratio of completed projects on time to completed projects. ([0-1])
On-budget Rate	Ratio of completed projects within budget to completed projects. ([0-1])
Job Count	Number of the worker's received jobs. [0-5,855]
Completed Job Count	Number of the worker's completed jobs. [0-5,080]
Completion Rate	Ratio of Completed Job Count to Job Count. ([0-1])
Uncompleted Job Count	Number of the worker's uncompleted jobs. [0-857]
Communication	Average rating on communication for all completed projects. ([0-5])
Expertise	Average rating on expertise for all completed projects. ([0-5])
Rehiring	Average rating on rehiring for all completed projects. ([0-5])
Professionalism	Average rating on professionalism for all completed projects. ([0-5])
Quality	Average rating on quality for all completed projects. ([0-5])
Overall Reputation	Average overall rating for all completed projects. ([0-5])
Review Count	Number of the worker's received reviews. [0-4,341]
Uncompleted Review Count	Number of the worker's uncompleted reviews. [0-141]
Bid Attributes	
Working Period	Working period of the project proposed by the worker (in days). [0-10,000]
Bid Price	Bid price for the project proposed by the worker. $[0 - 1.12 \times 10^9]$
Bidder-Project/Job Related Attributes	
Profile Skill Match	Ratio of the number of skills required in the project to the number of skills the worker declared in the profile. ([0-1])
History Skill Match	Ratio of the number of skills required in the project to the number of skills in the worker's working history. ([0-1])
Country Match	The match degree of Employer's country and worker's country[b] Calculated by $\frac{Pr(freelancer's country, employer's country)}{Pr(freelancer's country)Pr(employer's country)}$ [-2.5-9.15]
Bid Price-Budget	How much the worker's bid price within the budget of the project provided by the employer. Calculated by $\frac{(BidPrice-BudgetMinimum)}{(BudgetMaximum-BudgetMinimum)}$. [-982 - 1.85×10^6]

[a] There were four membership plans (Free, Starter, Standard and Premium) on the Freelancer.com platform by 2012. In 2012, the platform canceled its Starter Plan and introduced a Basic Plan. Then, two new plans (Intro and Plus) were added in 2014. Next, Standard and Premium were replaced by Professional and Premier, respectfully in 2016. Since the data set contains freelancers from 2000 to 2017, we classified membership plans in different time periods according to the similarity of the packages.

[b] We assume that different country match degrees appear in various types of project. So we compute country match within each type of project.

5.2 Data Preparation

The raw dataset is composed of job posts that are tendered by different numbers of bidders. For computational convenience, the data are padded to have the same number of bidders in each job post. The padding is implemented by adding fictional bidders whose attributes are assigned the least competitive value of all the existing bidders, for instance, the lowest earning score, professionalism and highest uncompleted job counts.

As a preprocessing step, we normalize the data to ensure commensurability in the attribute values. The attribute values of all bidders are linearly scaled to a unit interval within each project. We normalize within each project because each project can be very different, especially their bid price and working period. Some projects are short term and low budget, e.g., "proofread a 1500 word text translated from English to Italian with budget of $10"; some projects are relatively long term and high budget, such as "design a car booking app like Uber with budget of $25000 within half a year". Bidder attribute values for these two kinds of projects are quite different. Hence it is reasonable to scale data within each pool.

After data padding and normalization, we randomly shuffle the data and divide them into training (65%), validation (15%), and test (20%) sets. The training and validation sets are used to calibrate the model parameters and prevent overfitting. The test set is not used during the training process,

only for the final experimental analysis.

5.3 Hyperparameters

The hyperparameters are the variables that determine the model structure (e.g., the channel size of each point-wise convolutional layer) and the variables that determine how the network is trained and regularized (e.g., the learning rate and dropout rate) [11]. For the DCM with multiple convolutional layers, the hyperparameters we considered are listed in Table 3.

Table 3: Hyperparameters considered for the experiments

Hyperparameters	Descriptions	Values
Channel size of the convolution kernel	The number of attributes in the next layer's representation.	15, 50, 100, 200
Learning Rate for SGD	How quickly we adjust the parameters to follow the loss gradient.	0.1, 0.01, 0.001
Dropout Rate	The existence probability of each neuron and its connections in dropout.	0.1, 0.3, 0.5

For simplicity, in our multi-layer models, we only consider architectures with the same number of channels (convolutional kernels) at each layer of the model. For training, we use stochastic gradient descent (SGD) with mini-batch size of 64 to maximize the likelihood of the observed hiring choice decisions. Mini-batch SGD is a standard choice for large data sets. We test several values of learning rate for the SGD optimization, which influences how rapidly the network parameters evolve during training. We also try different dropout probabilities (0.1, 0.3 and 0.5) for regularization.

5.4 Evaluation Metrics

Two evaluation metrics are used to quantify the models' performances: the normalized log likelihood L,

$$L = \frac{1}{N} \sum_{k} \log(P(\mathbf{b}_{i_k}^k | X^k)) \tag{11}$$

and the Top n Accuracy (ACC_n), which measures the fraction of the N projects in which the model assigns the winning bid $\mathbf{b}_{i_k}^k$ a probability that is smaller than at most $n-1$ other bids. So, for example, ACC_1 corresponds to the fraction of correct hiring predictions, while ACC_2 is the probability that the winner is in the top two bids in each bidding pool.

5.5 Model Architectures

We design a series of experiments to test the models' performances for each of the 12 job categories listed in Table 1. Four architectures are considered with increasing network depths, as shown in Figure 1. The simplest network architecture is the conditional logit model, as is labeled with the red arrows. It is composed of the reshape layer and a single point-wise convolutional layer with kernel size of $1 \times n \times 1 \times 1$. The more complex architectures are composed of stacked point-wise convolutional layers and non-linearities. For instance, the blue arrows represent a deeper model with 2 convolutional layers, the first convolutional layer transfers the bidding pool matrix into the form of $n_1 \times m \times 1$, where n_1 denotes the attribute counts of the new representation of the bidding pool.

5.6 Results

DCMs using different architectures and hyperparameters are tested to evaluate the impact of structural variations of the model, as measured by four different metrics (normalized log likelihood L and top n accuracy ACC_n for $n = 1, 2, 3$). When training the model, we gradually increase model complexity by adding more convolutional layers. For each architecture, we use the validation data to adjust the learning rate and dropout rate, and use validation-based early stopping when training the model. We chose the final model according to the performance on the validation set. We train the models 30 times and averaged performance is reported. Table 4 shows the models' validation prediction performances for the job posts under the BAHRL (business, accounting, human resources,

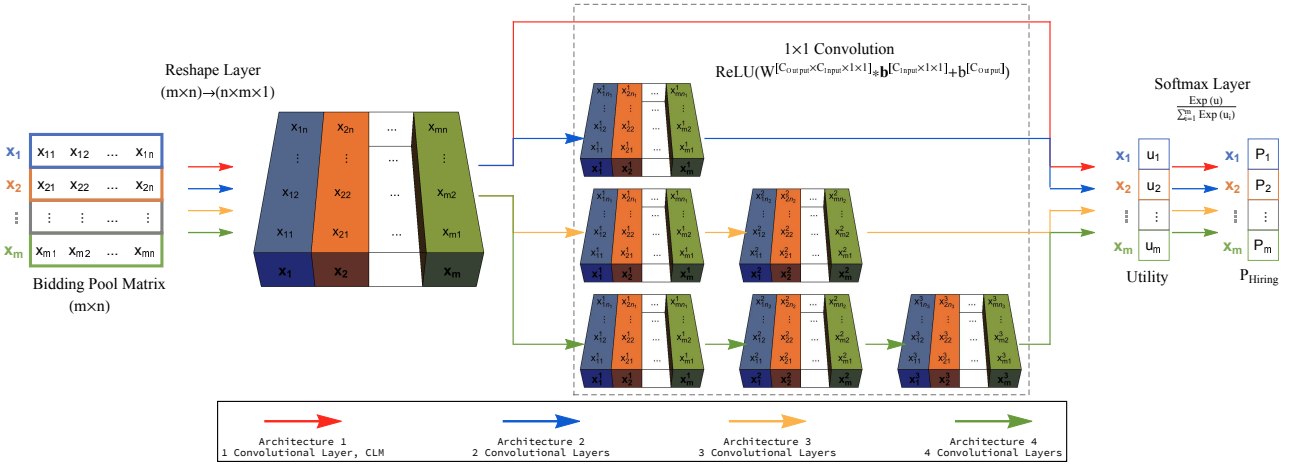


Figure 1: The architectures of our model. From left to right are input data with size of $m \times n$, reshape layer which changes the input to a $n \times m \times 1$ tensor, followed by four choices of convolutional layers which compute bidders' information to bidders' utilities, and softmax layer last to transform the estimated utility values into hiring probabilities. The red arrows represent Architecture 1, where only 1 convolutional layer is used, and is equivalent to the classical CLM. The blue arrows represent Architecture 2 with 2 convolutional layers, yellow arrows Architecture 3 with 3 layers, and green arrows Architecture 4 with 4 layers.

and legal) category. For computational reasons, we select not to test all these model variants for the other job categories.

The simplest model in this experiment is the CLM, which is implemented using a single layer convolutional network with kernel size $1 \times n \times 1 \times 1$. The CLM model's performance forms a baseline for comparison, and is listed for each metric in the second column of Table 4. Compared to the basic CLM, choice models with multiple convolutional layers achieve better performance. Considering model architectures, deeper models with dropout techniques tend to perform better. For fixed layer sizes, increasing the number of layers tends to increase model performance with dropout regularization. For a fixed number of layers, increasing the number of convolutional channels per layer from 15 to 100 tends to increase model performance, while model performances become worse when keep increasing neuron size to 200. We note that for the evaluation metrics of accuracy, we achieve the best performance using DCMs with 4 convolutional layers, each with channel size of 100, trained with learning rate of 0.01 and dropout rate 0.3. The best performing models improve L and ACC_1 by 7.17% and 10.00%, respectively, with similar improvements in ACC_2 and ACC_3 (5.76% and 4.23%). Based on this evidence, we apply the same architecture and hyperparameters to the remaining categories of job posts.

Table 5 compares the performances of the basic CLM and our DCM, using the selected architecture and hyperparameters (4 convolutional layers, each with channel size of 100, trained with learning rate of 0.01 and dropout rate of 0.3), across the 12 categories of job posts. Results evaluated using L and $ACC_{1,2,3}$ are given separately for the training, validation and test sets. Generally, almost all job categories show better prediction accuracy using the DCM compared to CLM for all four performance measures, L and $ACC_{1,2,3}$. On average, the L metric improved by $0.090 - 0.253$ and the $ACC_{1,2,3}$ metrics improved by $0.015 - 0.041$.

Comparing performance on the training, validation and test sets, we see that the CLM results show no significant overfitting, since the performances over the three different sets are generally close. For the DCM, although we include regularization and adopt an early stopping strategy, we see some overfitting in certain categories, such as TL. For categories DMA and WIS, where we have enough samples, the DCM results show no evident overfitting.

5.7 Attribute Encoding Performance

To gain an intuitive understanding of the learned feature representations from our deep choice model, we visualize the bidders' features using the t-distributed stochastic neighbor embedding (t-

Table 4: The validation set performances on four evaluation metrics and several configurations of models for BAHRL Job Posts. We can see that DCM with more layers and incorporating dropout tend to perform better than CLM; the best model under each metric is highlighted. For all metrics, the 4 layer model with $C = 100$ and $P_{\text{Dropout}} = 0.3$ performs best, and we select this for subsequent experiments.

Evaluation Metric	CLM	Dropout	Convolutional Layers											
			2 Layers				3 Layers				4 Layers			
			$C = 15$	$C = 50$	$C = 100$	$C = 200$	$C = 15$	$C = 50$	$C = 100$	$C = 200$	$C = 15$	$C = 50$	$C = 100$	$C = 200$
L	-1.771	No Dropout	-1.672	-1.672	-1.654	-1.658	-1.653	-1.665	-1.665	-1.658	-1.652	-1.659	-1.656	-1.657
		$P_{\text{Dropout}} = 0.1$	-1.671	-1.658	-1.646	-1.651	-1.649	-1.654	-1.658	-1.653	-1.651	-1.652	-1.657	-1.652
		$P_{\text{Dropout}} = 0.3$	-1.702	-1.665	-1.655	-1.650	-1.646	-1.698	-1.645	-1.649	-1.644	-1.678	**-1.639**	-1.648
		$P_{\text{Dropout}} = 0.5$	-1.719	-1.673	-1.655	-1.652	-1.652	-1.717	-1.683	-1.663	-1.656	-1.697	-1.690	-1.672
ACC_1	0.360	No Dropout	0.370	0.374	0.378	0.385	0.385	0.381	0.382	0.374	0.385	0.385	0.388	0.377
		$P_{\text{Dropout}} = 0.1$	0.374	0.393	0.392	0.379	0.392	0.389	0.376	0.388	0.380	0.389	0.376	0.378
		$P_{\text{Dropout}} = 0.3$	0.374	0.380	0.382	0.394	0.372	0.375	0.393	0.392	0.393	0.376	**0.396**	0.389
		$P_{\text{Dropout}} = 0.5$	0.363	0.380	0.389	0.393	0.380	0.365	0.365	0.372	0.382	0.370	0.371	0.390
ACC_2	0.573	No Dropout	0.588	0.582	0.602	0.594	0.588	0.589	0.595	0.577	0.596	0.590	0.593	0.599
		$P_{\text{Dropout}} = 0.1$	0.585	0.590	0.605	0.590	0.588	0.602	0.598	0.592	0.577	0.603	0.583	0.593
		$P_{\text{Dropout}} = 0.3$	0.582	0.588	0.586	0.601	0.599	0.582	0.603	0.602	0.593	0.585	**0.606**	0.603
		$P_{\text{Dropout}} = 0.5$	0.578	0.583	0.586	0.596	0.592	0.574	0.580	0.588	0.577	0.588	0.587	0.585
ACC_3	0.709	No Dropout	0.722	0.722	0.723	0.721	0.722	0.723	0.724	0.705	0.722	0.714	0.705	0.720
		$P_{\text{Dropout}} = 0.1$	0.713	0.712	0.732	0.709	0.720	0.734	0.719	0.722	0.718	0.712	0.724	0.725
		$P_{\text{Dropout}} = 0.3$	0.734	0.733	0.733	0.729	0.730	0.727	0.736	0.731	0.736	0.723	**0.739**	0.727
		$P_{\text{Dropout}} = 0.5$	0.710	0.714	0.713	0.722	0.727	0.711	0.710	0.717	0.718	0.715	0.714	0.710

Table 5: Comparing CLM and DCM performances for 12 categories of job posts. DCM improves over CLM across all types of jobs.

Category	Model	Training				Validation				Test			
		L	ACC_1	ACC_2	ACC_3	L	ACC_1	ACC_2	ACC_3	L	ACC_1	ACC_2	ACC_3
BAHRL	CLM	-1.725	0.390	0.600	0.716	-1.771	0.360	0.573	0.709	-1.798	0.348	0.576	0.704
	DCM	-1.515	0.454	0.654	0.765	-1.639	0.396	0.606	0.739	-1.670	0.372	0.603	0.725
DEA	CLM	-2.030	0.319	0.514	0.636	-2.037	0.320	0.505	0.628	-2.051	0.313	0.515	0.632
	DCM	-1.846	0.354	0.549	0.672	-1.881	0.335	0.525	0.652	-1.893	0.336	0.530	0.656
DMA	CLM	-2.200	0.276	0.445	0.562	-2.203	0.275	0.444	0.561	-2.203	0.275	0.446	0.561
	DCM	-2.099	0.295	0.467	0.583	-2.110	0.292	0.463	0.579	-2.113	0.290	0.462	0.581
ES	CLM	-1.794	0.383	0.589	0.707	-1.807	0.382	0.576	0.701	-1.770	0.391	0.603	0.708
	DCM	-1.630	0.411	0.617	0.735	-1.679	0.389	0.593	0.725	-1.641	0.414	0.619	0.724
LJS	CLM	-1.185	0.553	0.799	0.855	-1.132	0.567	0.810	0.863	-1.294	0.449	0.857	0.918
	DCM	-1.174	0.503	0.780	0.874	-1.117	0.595	0.865	0.892	-1.143	0.490	0.878	0.939
MPC	CLM	-1.971	0.344	0.535	0.645	-1.978	0.346	0.535	0.651	-1.981	0.346	0.536	0.653
	DCM	-1.760	0.392	0.589	0.700	-1.830	0.373	0.549	0.663	-1.833	0.378	0.566	0.677
O	CLM	-1.518	0.448	0.679	0.788	-1.527	0.444	0.673	0.783	-1.577	0.428	0.672	0.790
	DCM	-1.357	0.480	0.700	0.810	-1.397	0.461	0.673	0.783	-1.396	0.468	0.697	0.801
PSM	CLM	-1.669	0.450	0.638	0.713	-1.701	0.415	0.575	0.708	-1.708	0.404	0.621	0.750
	DCM	-1.435	0.476	0.682	0.746	-1.553	0.434	0.632	0.736	-1.507	0.443	0.657	0.764
SM	CLM	-2.040	0.314	0.524	0.644	-2.014	0.321	0.529	0.649	-2.044	0.313	0.525	0.651
	DCM	-1.735	0.383	0.588	0.702	-1.776	0.361	0.566	0.685	-1.791	0.342	0.549	0.675
TL	CLM	-1.522	0.470	0.707	0.817	-1.672	0.471	0.662	0.762	-1.647	0.378	0.647	0.791
	DCM	-1.381	0.495	0.715	0.830	-1.580	0.488	0.658	0.771	-1.503	0.419	0.669	0.800
WC	CLM	-1.959	0.324	0.515	0.636	-1.965	0.326	0.517	0.636	-1.952	0.333	0.518	0.639
	DCM	-1.771	0.374	0.568	0.688	-1.819	0.35	0.545	0.667	-1.806	0.359	0.554	0.672
WIS	CLM	-1.772	0.392	0.597	0.710	-1.777	0.391	0.597	0.708	-1.772	0.392	0.596	0.709
	DCM	-1.649	0.408	0.615	0.726	-1.660	0.405	0.614	0.722	-1.659	0.408	0.613	0.723

SNE) [30], which is a nonlinear dimensionality reduction technique that attempts to accurately reflect the distances among high-dimensional data in a low-dimensional space. Here, we use t-SNE to visualize how the original and the learned features of the DCM capture bidder quality for the BAHRL category.

To do so, for each project, we pick the actual winner (selected bidder) and randomly choose a non-selected bidder from the same project. Then, we visualize the distribution of these bids (both selected and non-selected) in the original feature space, using the normalized attribute vector, and using the new attribute vectors corresponding to the responses (channel values) at the final convolutional layer of our four-layer DCM (each with channel size 100, learning rate 0.01, and dropout rate 0.3) using t-SNE to embed into two dimensions. For each embedding, we plot the selected bidders in red, and then non-selected in blue.

Figure 2 shows the result. Here we can see that the embedding using the DCM's features (lower

panel) show much better separation between the selected and non-selected bidders than the embedding based on the original features (upper panel). In the upper panel, the two types of bidders are overlapping and mixed, suggesting the difficulty of separating the best bids using a simple linear model. In contrast, the DCM-processed features show much better separation (with actual winners more concentrated on the right). This suggests that the DCM's layers compute a better representation of bidders' attributes for distinguishing preferred versus non-preferred bidders.

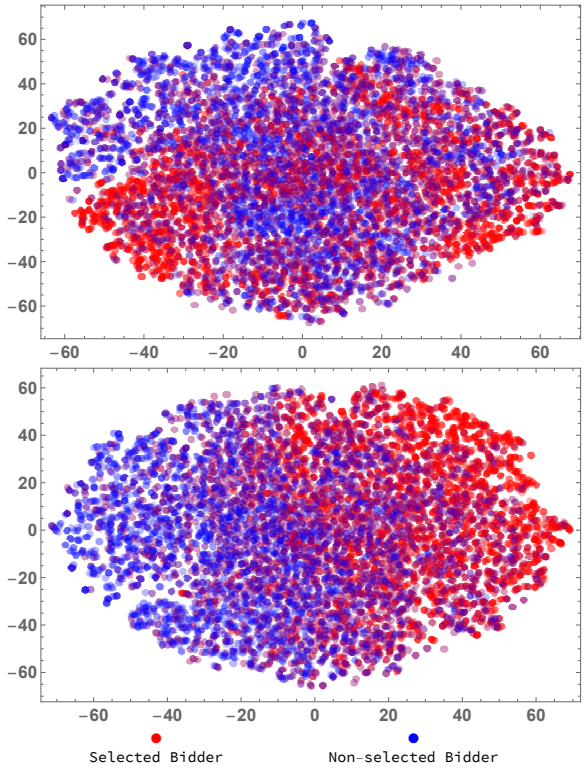


Figure 2: Visualizations of attributes for conditional logit model and deep choice model from the test set using t-SNE. Upper panel is input space for CLM and lower one is new attributes encoded by DCM. We can see that after attributes encoding, selected bidders and non-selected bidders are more easy to cluster.

6 Conclusion

The online labor market is a fast-growing economic system that is notable for its information availability and transparency. People can compete for the same job across geographical boundaries, which significantly enhances the efficient matching of labor demand and supply. In this work, we investigated hiring decision behavior using real-world hiring samples from freelancer.com. The classic conditional logit model's linear form limits its representation of the choice utility. In response, we propose a deep choice model based on a point-wise convolutional neural network to represent the utility function associated with each bid, allowing the model to capture more complex employer decision processes. Experimental results on 12 different categories of real jobs provide validation of the benefits of our non-linear utility model over the conditional logit model. By visualizing attributes for conditional logit model and our deep choice model using t-SNE, we see that after attribute encoding, bidders are more coherent in quality, with clear differences between selected and non-selected bidders.

Acknowledgment

This work was supported by the Fundamental Research Funds for the Central Universities (2018YJS052), Defense Advanced Research Projects Agency (W911NF-18-C-0015), National Science Foundation (IIS-1254071), and Beijing Social Science Fund (16JDGLA010).

References

[1] Abhinav, K.; Dubey, A.; Jain, S.; Virdi, G.; Kass, A.; Mehta, M. (2017). Crowdadvisor: A framework for freelancer assessment in online marketplace, *Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track*, Buenos Aires, Argentina, 93–102, 2017.

[2] Agrawal, A.; Lacetera, N.; Lyons, E. (2016). Does standardized information in online markets disproportionately benefit job applicants from less developed countries? *Journal of international Economics*, 103, 1–12, 2016.

[3] Akiva, M. E.; Lerman, S. R.; Lerman, S. R. (1985). *Discrete choice analysis: theory and application to travel demand*, MIT press, vol. 9, 1985.

[4] Boskin, M. J. (1974). A conditional logit model of occupational choice. *Journal of Political Economy*, 82(2, Part 1), 389–398, 1974.

[5] Brynjolfsson, E.; Hitt, L. M. (2003). Computing productivity: Firm-level evidence. *Review of economics and statistics*, 85(4), 793–808, 2003.

[6] Chan, J.; Wang, J. (2017). Hiring preferences in online labor markets: Evidence of a female hiring bias, *Management Science*, 64(7), 2973–2994, 2017.

[7] Chollet, F. (2015). Keras. `https://keras.io/`, 2015.

[8] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, USA, 1251–1258,. 2017.

[9] Sousa, W.; Montevechi, J.; Miranda, R.; Rocha, F.; Vilela F. (2019). *Economic Lot-size using machine learning, parallelism, metaheuristic and simulation*, 18(2), 205–216, 2019.

[10] Fernández-Delgado, M.; Cernadas, E.; Barro, S.; Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1), 3133–3181, 2014.

[11] Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. (2016). *Deep learning*, MIT press Cambridge, Vol.1, 2016.

[12] Hagenauer, J. and Helbich, M. (2017). A comparative study of machine learning classifiers for modeling travel mode choice. *Expert Systems with Applications*, 78, 273–282, 2017.

[13] Hensher, D. A.; Ton, T. T. (2000). A comparison of the predictive potential of artificial neural networks and nested logit models for commuter mode choice. *Transportation Research Part E: Logistics and Transportation Review*, 36(3), 155–172, 2000.

[14] Hoffman, S. D.; Duncan, G. J. (1988). Multinomial and conditional logit discrete-choice models in demography. *Demography*, 25(3), 415–427, 1988.

[15] Horton, J. J.; Tambe, P. (2015). Labor economists get their microscope: big data and labor market analysis. *Big data*, 3(3), 130–137, 2015.

[16] Huws, U.; Joyce, S. (2016). Crowd working survey: Size of the uk's 'gig economy' revealed for the first time. *University of Hertfordshire*, 2016.

[17] Ji, S.; Xu, W.; Yang, M.; Yu, K. (2013). 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1), 221–231, 2013.

[18] Klein, G. A.; Orasanu, J. E.; Calderwood, R. E.; Zsambok, C. E. (1993). *Decision making in action: Models and methods*, Ablex Publishing, 1993.

[19] Kokkodis, M.; Papadimitriou, P.; Ipeirotis, P. G. (2015). Hiring behavior models for online labor markets. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, Shanghai, China, 223–232, 2015.

[20] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, Nevada, USA, 1097–1105, 2012.

[21] Kuek, S. C.; Paradi-Guilford, C.; Fayomi, T.; Imaizumi, S.; Ipeirotis, P., Pina, P.; Singh, M. (2015). *The global opportunity in online outsourcing*, Technical report, World Bank, Washington, DC, 2015.

[22] LeCun, Y.; Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10), 255–258, 1995.

[23] LeCun, Y.; Bengio, Y.; Hinton, G. (2015). Deep learning, *Nature*, 521(7553), 436–444, 2015.

[24] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324, 1998.

[25] Lazear, P.; Shaw, L.; Stanton, T. (2016). *Who gets hired? the importance of finding an open slot*, Technical report, National Bureau of Economic Research, 2016.

[26] Lhéritier, A.; Bocamazo, M.; Delahaye, T.; Acuna-Agost, R. (2019). Airline itinerary choice modeling using machine learning. *Journal of Choice Modelling*, 31, 198-209, 2019.

[27] Lin, M.; Chen, Q.; Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[28] Lin, W.-Y.; Hu, Y.-H.; Tsai, C.-F. (2012). Machine learning in financial crisis prediction: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4), 421–436, 2012.

[29] Ma, Y.; Zhang, Z.; Ihler, A.; Pan, B. (2018). Estimating warehouse rental price using machine learning techniques. *International Journal of Computers Communications & Control*, 13(2), 235–250, 2018.

[30] Maaten, L. V. D.; Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9, 2579–2605, 2008.

[31] Mali, P.; Kuzmanovic, B.; Nikolic, M.; Mitic, S.; Terek, E. (2019). Model of leadership and entrepreneurial intentions among employed persons. *International Journal of Simulation Modelling*, 18(3), 385–396, 2019.

[32] McFadden, D. (1974). *Conditional logit analysis of qualitative choice behavior*. New York: Academic Press, Chapter 4, 1974.

[33] McFadden, D. (2001). Economic choices. *American economic review*, 91(3), 351–378, 2001

[34] Moreno, A.; Terwiesch, C. (2014). Doing business with strangers: Reputation in online service marketplaces. *Information Systems Research*, 25(4), 865–886, 2014.

[35] Mottini, A.; Acuna-Agost, R. (2017). Deep choice model using pointer networks for airline itinerary prediction. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Halifax, NS, Canada, 1575–1583, 2017.

[36] Nam, D.; Kim, H.; Cho, J.; Jayakrishnan, R. (2017). A model based on deep learning for predicting travel mode choice. *Proceedings of the Transportation Research Board 96th Annual Meeting Transportation Research Board*, Washington, DC, USA, 8–12, 2017.

[37] Olugboyega O. (2016). Examining the protocols and platforms adopted for building information modelling processes by Nigerian construction professionals *Journal of System and Management Sciences*, 6(4), 1–45, 2016.

[38] Omrani, H. (2015). Predicting travel mode of individuals by machine learning. *Transportation Research Procedia*, 10, 840–849, 2015.

[39] Otsuka, M.; Osogami, T. (2016). A deep choice model. *AAAI*, Phoenix, Arizona USA, 850–856, 2016.

[40] Pan, B.; Hsu, K.; AghaKouchak, A.; Sorooshian, S. (2019). Improving precipitation estimation using convolutional neural network. *Water Resources Research*, 55(3), 2301–2321, 2019.

[41] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929–1958, 2014.

[42] Yoganarasimhan, H. (2013). The value of reputation in an online freelance marketplace. *Marketing Science*, 32(6), 860–891, 2013.

[43] Freelancer.com (2019). Freelancer.com reveals the fastest growing online jobs in 2018. `http://www.freelancer.com`. Accessed April 30 2019.

[44] Upwork (2019). Upwork to report first quarter 2019 financial results on may 8, 2019. `https://investors.upwork.com/news-releases`. Accessed April 30 2019.

Journal's webpage: http://univagora.ro/jour/index.php/ijccc/

This journal is a member of, and subscribes to the principles of,
the Committee on Publication Ethics (COPE).
https://publicationethics.org/members/international-journal-computers-communications-and-control

Cite this paper as:
Ma, Y.; Zhang, Z.; Ihler, A. (2020). A Deep Choice Model for Hiring Outcome Prediction in Online Labor Markets, *International Journal of Computers Communications & Control*, 15(2), 3760, 2020. https://doi.org/10.15837/ijccc.2020.1.3760