Talent Demand Forecasting with Attentive Neural Sequential Model

Qi Zhang[1,2,+], Hengshu Zhu[2,*], Ying Sun[2,3,+], Hao Liu[4], Fuzhen Zhuang[5,6], Hui Xiong[7,*]
[1]University of Science and Technology of China. [2]Baidu Talent Intelligence Center, Baidu Inc.
[3]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences,
Institute of Computing Technology, Chinese Academy of Sciences.
[4]The Hong Kong University of Science and Technology. [5] Institute of Artificial Intelligence, Beihang University.
[6] SKLSDE, School of Computer Science, Beihang University. [7]Rutgers, The State University of New Jersey.
zq26@mail.ustc.edu.cn,zhuhengshu@baidu.com,sunying17g@ict.ac.cn,liuh@ust.hk
zhuangfuzhen@buaa.edu.cn,hxiong@rutgers.edu

ABSTRACT

To cope with the fast-evolving business trend, it becomes critical for companies to continuously review their talent recruitment strategies by the timely forecast of talent demand in recruitment market. While many efforts have been made on recruitment market analysis, due to the sparsity of fine-grained talent demand time series and the complex temporal correlation of the recruitment market, there is still no effective approach for fine-grained talent demand forecast, which can quantitatively model the dynamics of the recruitment market. To this end, in this paper, we propose a data-driven neural sequential approach, namely Talent Demand Attention Network (TDAN), for forecasting fine-grained talent demand in the recruitment market. Specifically, we first propose to augment the univariate time series of talent demand at multiple grained levels and extract intrinsic attributes of both companies and job positions with matrix factorization techniques. Then, we design a Mixed Input Attention module to capture company trends and industry trends to alleviate the sparsity of fine-grained talent demand. Meanwhile, we design a Relation Temporal Attention module for modeling the complex temporal correlation that changes with the company and position. Finally, extensive experiments on a real-world recruitment dataset clearly validate the effectiveness of our approach for fine-grained talent demand forecast, as well as its interpretability for modeling recruitment trends. In particular, TDAN has been deployed as an important functional component of intelligent recruitment system of cooperative partner.

CCS CONCEPTS

• **Information systems** → **Data mining**.

[+]This work was accomplished when the first and the third authors working as interns in Baidu supervised by the second author.
[*] Corresponding authors.
This work was partially supported by grants from the National Natural Science Foundation of China (No.61836013, 91746301, 61773361).

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467131

KEYWORDS

Talent demand forecast, Neural sequential model, Attention mechanism, Recruitment market

ACM Reference Format:
Qi Zhang[1,2,+], Hengshu Zhu[2,*], Ying Sun[2,3,+], Hao Liu[4], Fuzhen Zhuang[5,6], Hui Xiong[7,*]. 2021. Talent Demand Forecasting with Attentive Neural Sequential Model. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467131

1 INTRODUCTION

In the era of knowledge economy, more and more attention has been paid to talent competitiveness. To cope with the fast-evolving business trend, it becomes critical for companies to continuously review their talent recruitment strategies [35]. As a major component in market intelligence, the timely forecast of talent demands in recruitment market can not only help companies to maintain the business competitive edge, but also provide an important indicator of economic prosperity by capturing the supply and demand relationship of talents.

Traditionally, the talent demand analysis in recruitment market is usually conducted by domain experts with simple statistic approaches [24]. With the recent development of online recruitment services and data mining technologies, researchers have explored talent recruitment related problems through a data-driven perspective, such as recruitment trend modeling [38], person-job fit [21], job skill demand analysis [32], and talent flow forecast [35]. Despite huge research efforts have been made in related directions, there still lacks an effective approach to fine-grained (i.e., specific position in specific company) talent demand forecasting that can quantitatively model the dynamics of recruitment market.

To this end, in this paper, we aim to forecast the talent demand in recruitment market through a fine-grained and dynamic manner. Along this line, some critical challenges cannot be neglected. On the one hand, the demand of recruitment for a single position in a company is usually small, which makes the fine-grained time series of talent demands (i.e., *company* $\times$ *position*) very sparse and difficult to model and predict. On the other hand, the talent demand of different companies and positions usually have different temporal correlations that may depend on various factors, e.g., external competition and marketing trend. This information is difficult to capture from the fine-grained time series itself. Therefore, it is

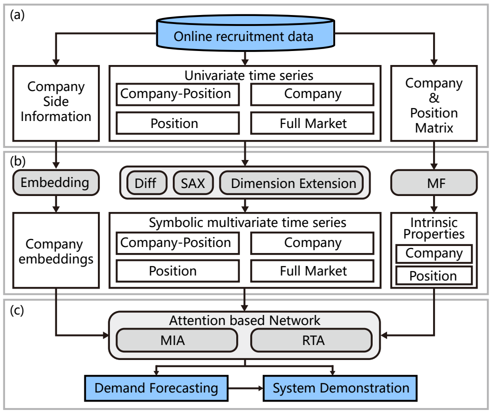

Figure 1: An overview of the TDAN framework.

important to augment and incorporate more information for alleviating the sparsity of fine-grained time series and flexibly modeling the different temporal correlation of different company positions.

To address the above challenges, we propose a data-driven sequential approach, namely Talent Demand Attention Network (TDAN), for forecasting the fine-grained talent demand in the recruitment market. Figure 1 overviews the workflow of our framework. To be specific, as shown in the part (a) of Figure 1, in order to augment and incorporate useful information, we first group online recruitment advertisements into univariate time series and augment the talent demand in multiple grained levels (i.e., company-position level, company level, position level, and full market level). The time series of high-level talent demand can reflect the recruitment environment of low-level talent demand. Meanwhile, we exploit the side information of companies which describes the attributes of different companies. We also construct the recruitment matrix of company and position which reflects the relationships of different companies and positions. Then, as shown in the part (b) of Figure 1, in order to improve the stationarity, reduce the influence of random noise, and better reflect the trend of recruitment, we calculate the first-order difference of the time series and symbolize them into five trend categories through Symbolic Aggregate Approximation (SAX) algorithm [27]. For each symbolic time series, we further expand the dimension of each time point by integrating the corresponding preceding values to incorporate the sequential pattern, which will benefit the training of neural network models. Meanwhile, we embed the side information of companies and extract intrinsic attributes from the recruitment matrix of company and position with Matrix Factorization (MF) techniques.

Based on these useful information, as shown in the part (c) of Figure 1, we propose an attention-based neural network model which contains two novel modules, namely Mixed Input Attention (MIA) and Relation Temporal Attention (RTA). First, because the coarse-grained talent demand time series contains more comprehensive recruitment information, the MIA module mixes different grained levels of talent demands together and captures company trends and industry trends to alleviate the sparsity of fine-grained talent demand. When mixing different grained levels of talent demands, the module applies the attention mechanism to automatically assign different weights to different levels of talent demands. Then, in the RTA module, we design a meta-temporal attention mechanism for flexibly modeling the temporal correlation that changes with the company and position. In detail, this meta-temporal attention mechanism mainly uses side information and intrinsic attributes to capture the characteristics of different companies and positions. Meanwhile, the multi-head attention mechanism will be adopted to model the pattern of fine-grained talent demand itself. The RTA module consists of the meta-temporal attention mechanism and multi-head attention mechanism, which complement each other. Finally, we conduct extensive experiments on a real-world online recruitment dataset, which clearly validate the effectiveness of our approach for fine-grained talent demand forecast, as well as its interpretability for modeling recruitment trends. In particular, TDAN has been deployed as an important functional component of intelligent recruitment system of cooperative partner, which can provide timely market insights for both hiring managers and recruiters.

2 RELATED WORK

Generally, the related works of this paper can be grouped into three categories, including *analysis of online recruitment market*, *time series prediction* and *attention mechanism based neural networks*.

Analysis of Online Recruitment Market. As a crucial issue of human resource management, talent recruitment has attracted wide attention in the past decade. On the one hand, traditional recruitment analysis mainly depends on expert knowledge and statistical methods. For example, in [15], researchers introduced the multiple regression analysis to forecast human resource demand with enterprise strategic management theory. In [36], a human resource demand forecasting model was built based on Cob-Douglas production function and empirical analysis. On the other hand, machine learning methods mainly depend on big data. With the prevalence of online recruitment websites, a large number of digital job posts of recruitment can be collected. Thanks to the accumulation of recruitment data resources, there emerges a trend of studying online recruitment market with machine learning methods. For instance, in [38], authors proposed an unsupervised sequential latent variable model for automatically modeling the trend of recruitment market. However, this work mainly discussed the trend of recruitment topics but did not quantify the fine-grained talent demand at the company and position level, which is crucial for talent recruitment strategy. Then, [18] focused on career move prediction, [32] proposed a tensor factorization approach for job skill demand analysis, [35] designed a dynamic latent factor model for talent flow forecast, and so on [21, 23, 28, 39, 40]. It can be seen that though large efforts have been devoted to online recruitment market analysis, there still lacks an effective quantitative method for fine-grained talent demand forecasting in the recruitment market.

Time Series Prediction. Time series prediction is an active research area which covers many fields such as finance, market, transportation, clinic, and so on [1, 3, 26, 35]. In traditional statistical models, Autoregressive Integrated Moving Average (ARIMA) is one of the most important and widely used time series models [7, 34]. It can implement various exponential smoothing models [20]. Another work [29] designed a procedure, named Prophet, for time series forecasting based on an additive model. It can work effectively with time series that have strong seasonal effects and several seasons of historical data. However, most of the traditional statistical models are linear, in which predictions of the future values are constrained to be linear functions of past observations [34]. Therefore, the nonlinear patterns observed in real problems are almost missed.

In recent years, Artificial Neural Networks (ANNs) have emerged to be one of the most powerful tool for solving nonlinear machine learning tasks. Due to their flexible nonlinear modeling capability, many kinds of ANN models have been proposed for time series prediction [30]. Among these models, Recurrent Neural Networks (RNNs) [19] focus on sequential data modeling. The connections among nodes form a directed graph along a temporal sequence and exhibit temporal dynamic behavior. Specifically, Long Short-Term Memory (LSTM) [14] is a representative class of RNN variants, which has additional memory-control gates and memory cells to selectively store historical information and keep useful information. Bi-directional Long Short-Term Memory (BiLSTM) is an enhanced variant of LSTM, which considers bidirectional information in sequence data [12]. Since RNNs can capture the temporal information in sequence data, they have been widely used for time series prediction [11, 22]. However, RNNs are difficult to train in parallel and easy to forget long-term information. Convolutional Neural Networks (CNN) based time series prediction models were proposed for efficiently training and predicting [6]. TCN is a generic temporal convolutional network architecture that is applied across many tasks [4]. LSTNet combines CNN and RNN that can capture the short-term local dependency patterns and long-term time series trend patterns [17]. In recent years, some researchers applied attention models, which have shown remarkable success in Natural Language Processing (NLP) field, to time series modeling problems and achieved great performance. Specifically, authors in [25] developed the first simple attention-model based architecture for processing multi-variate clinical time-series data.

Along with these methods, many novel time series prediction models have been proposed for real-world applications. For example, in traffic flow prediction, the work [33] improved LSTM with feature enhancement. In stock price prediction, authors in [8] proposed to use a combination of MF and RNN for modeling the intrinsic properties of stocks. In supply chain prediction, researchers in [11] described one BiLSTM based model to propagate information about future input variables and better capture temporal patterns. However, there is still less use of the attention model for various time series forecasting. In this paper, we focus on the forecasting of online recruitment market. Due to the characteristics of time series data of labour market, we design one attention-based model for modeling fine-grained talent demand time series with different complex temporal correlations in different companies and positions.

Attention Mechanism based Neural Networks The attention mechanism had attracted wide attention in recent years, due to its powerful efficiency and interpretability. Moreover, researchers in [31] proposed one model, namely Transformers, that only stacked by the blocks of solely attention computations. Transformers use multi-head attention to describe the relationship between different positions in sequence data, so it can be trained in parallel and capture long-term patterns without decay. This kind of sequence model, which is based on the attention mechanism, has shown remarkable success in NLP field [9, 10]. Therefore, the researchers in [25] tried to apply the Transformers model in multivariate time-series data and achieved great performance. In this paper, we propose a novel attention-based model for time series forecasting, and apply it to online recruitment market.

Figure 2: An example of data format.

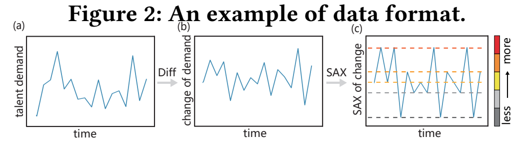

Figure 3: Extracting the trend of talent demand by SAX.

3 PRELIMINARIES

In this section, we will introduce the preliminaries of our TDAN framework, including data augmentation and problem formulation. To be specific, we first introduce our real-world online recruitment dataset. Then we propose to augment the talent demand time series and extract the static information of companies and positions. Based on transformed data, we formulate the modeling problem of fine-grained talent demand forecast. To facilitate illustration, details of some important mathematical notations are listed in Appendix A.

3.1 Data Description

The real-world dataset contains more than 355,465 job postings, which was collected from one of the largest Chinese online recruitment website. Specifically, each job posting mainly consists of the company name, job title and publish time. We extracted information from the job postings and formatted the data into three parts. The first is the talent demand time series of each position in each company, which is the goal we strive to forecast. The second is the side information of each company, including the year of establishment, company address, industry field, etc. The third is a *company* $\times$ *position* matrix that reflects the number of job postings each company posted in each position. An example of data format is shown in Figure 2.

3.2 Time Series Augmentation

Formally, we suppose there are N companies and M job positions. Based on the online recruitment data, we can extract fine-grained talent demand time series of position j in company i, denoted by $D^T_{i,j}(1 \le i \le N, 1 \le j \le M)$. T is the last time point of the observed time. According to $D^T_{i,j}$, we further calculate three levels of talent demand time series. Specifically, we extract the company-level time series D^T_i to reflect the overall talent demand of different companies, the position-level talent demand time series D^T_j to reflect the overall recruitment demand of different positions, and the full market-level demand time series D^T to reflect the entire recruitment market, formulated as

$$\begin{aligned} &D^T_{i,j} = \left\{ d^{T-L}_{i,j}, \ldots, d^T_{i,j} \right\}, \\ &D^T_i = \sum_{j=1}^{M} D^T_{i,j}, \;\; D^T_j = \sum_{i=1}^{N} D^T_{i,j}, \;\; D^T = \sum_{i=1}^{N}\sum_{j=1}^{M} D^T_{i,j}, \end{aligned} \tag{1}$$

where $d^t_{i,j}$ means the number of corresponding job postings at time slice t, and L means the length of observed time. Then, we construct

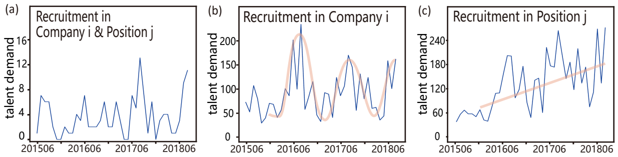

Figure 4: Examples of the talent demand time series with different levels.

the company-position matrix, R^T. More formally,

$$r_{i,j}^T = \sum_{t=1}^{T} d_{i,j}^t, \quad R^T = \begin{bmatrix} r_{1,1}^T & \cdots & r_{1,M}^T \\ \vdots & \ddots & \vdots \\ r_{N,1}^T & \cdots & r_{N,M}^T \end{bmatrix}_{N\times M}. \tag{2}$$

Intuitively, the talent demand of a company is affected by various complex factors, so that it is highly volatile and unstable. For better capturing the trend of talent demand, we calculate the first-order difference of the time series and symbolize them into five trend categories through SAX algorithm [27]. In Figure 3, we show an example of this procedure. Specifically, the five trends are *much less*, *less*, *hold*, *more*, and *much more*. We use $S_{i,j}^T$ to denote the symbolic time series in the last figure. The detailed formulation can be found in Appendix B.

However, both $D_{i,j}^T$ and $S_{i,j}^T$ are univariate time series, which only contain one value in each time point. This kind of time series makes it difficult for neural networks to effectively model each point. Therefore, we design a simple but useful up-sampling method. On each univariate time series, we pick the preceding values to expand the dimensions of each time point, which not only contains the sequential pattern but also benefits the training of neural network models. The extension time series $E_{i,j}^T$ can be formulated as

$$E_{i,j}^T = \left\{e_{i,j}^{T-L}, \ldots, e_{i,j}^T\right\}, \quad e_{i,j}^t = [\alpha_{i,j}^{t,1}, \ldots, \alpha_{i,j}^{t,dim}],$$
$$\alpha_{i,j}^{t,n} = \begin{cases} 0 & t+n \le dim \\ s_{i,j}^{t+n-dim} & t+n > dim \end{cases}, \tag{3}$$

where dim is the dimensions of $e_{i,j}^t$, which is the vector at time t in extension time series, $E_{i,j}^T$. Meanwhile, D_i^T, D_j^T, D^T are transformed to E_i^T, E_j^T, E^T in the same way.

3.3 Static Information Processing

In the online recruitment market, different companies have different preferences for talent demand in each position. As a result, the pattern of talent demand varies with respect to the companies and positions, which makes it difficult to try a unified and fine-grained talent demand model. Intuitively, the companies' preference is partly caused by their traits, e.g., the industry fields or company location. These traits are not only simple portraits, but also reflect the correlations between different companies. Along this line, we use these company features to enhance ability of TDAN on modeling company preferences. Specifically, we extract the side information I_i of companies from job posts, and map them into company embedding vectors, denoted by V_i, through two fully connected layers. Just as the *Embedding* module illustrated in Figure 1.

In addition to the static information, the companies' historical recruitment behaviors reflect the intrinsic properties of companies and job positions, which play quite an important role in explaining and even predicting online recruitment behaviors. For instance, an Internet company always has a high demand for algorithm engineers and a real estate company always has a high demand for sellers. Moreover, even similar Internet companies could have different preferences due to different specific industry fields. Correspondingly, various companies could recruit the same position, and the intrinsic properties of positions can reflect the job functions and the competitiveness for companies. To analyze the intrinsic properties of companies and job positions, based on the historical company & position matrix before time T, i.e., R^T, we can take advantage of MF approach to extract latent vectors of each company and position. These vectors can be viewed as the representation of the intrinsic properties of company i and position j, namely C_i and P_j. This is the *MF* module in Figure 1.

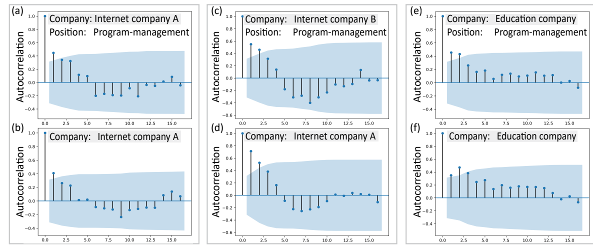

Figure 5: Autocorrelations of the talent demand time series.

3.4 Problem Formulation

Based on the Time Series Augmentation and the Processing of Static Information, here we can formulate the modeling problem of fine-grained talent demand forecast as:

Definition 1. **Talent Demand Forecast.** Given a set of extension talent demand time series $\{E_{1,1}^T, \ldots, E_{N,M}^T\}$, the corresponding embedding vectors $\{V_1, \ldots, V_N\}$ of companies, and the intrinsic properties of companies and positions $\{C_1, \ldots, C_N\}$ and $\{P_1, \ldots, P_M\}$ where N and M denote the total number of companies and job positions, and T means the observed time, the objective is to learn a predictive model $\mathcal{M}$ for forecasting the talent demand trend categories $\{s_{1,1}^{T+1}, \ldots, s_{N,M}^{T+1}\}$.

4 TECHNICAL DETAILS OF TDAN

In this section, Based on the multiple grained levels time series and various static information, we design an attention-based network, which includes the MIA module and the RTA module, for forecasting the trend of talent demand.

4.1 Mixed Input Attention Module

As discussed in Section 1, the demand of recruitment for a single position in a company is usually small, which makes the fine-grained time series of talent demands (i.e., $D_{i,j}^T$) very sparse to model, and different companies usually have different trends of talent demand that depend on various factors. However, the high-level time series of talent demands (i.e., D_i^T, D_j^T, D^T) are usually denser and can more clearly reveal the trend and periodicity. As shown in Figure 4, (a) is an example of fine-grained time series of talent demands (i.e., $D_{i,j}^T$), correspondingly, (b) is D_i^T and (c) is D_j^T. It can be seen that time series characteristics such as periodicity and trend are usually more obvious in high-level time series of talent demands. Therefore, high-level time series can enrich the characteristics and information of fine-grained time series. Meanwhile, in order to capture the relationship between different time series, the side information of companies and the intrinsic properties of companies and positions (i.e., I_i, C_i, P_j) can reflect the preferences of companies and the

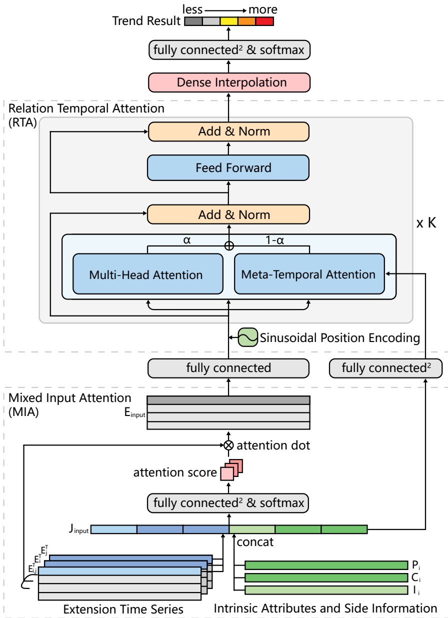

Figure 6: The structure of of attention-based network.

relationships between different companies, different positions, companies, and positions. Along this line, we develop a unique module in TDAN for better studying the fine-grained time series of talent demands, which is named MIA Module.

In MIA module, as shown in Figure 6, there are two kinds of inputs, including a set of time series and a set of static information. The set of time series contains one fine-grained time series, $E_{i,j}^T$, and two high-level time series, E_i^T and E_j^T (if the fine-grained time series are set as E_i^T, the two high-level time series are both set as E^T, and so on). The set of static information contains one embedding of side information, V_i, and two intrinsic properties, C_i and P_i. Then, we concatenate them into a joint input, written as J_{input}, which will go through two fully connected layers and one softmax layer, with the output as A_{input}. The A_{inuput} means the attention scores for weighting the input set of time series, and the weighted result is denoted as E_{input}, which uses coarse-grained data to enhance fine-grained data. Meanwhile, the static information is skillfully incorporated into the weighted result while ensuring the time series data structure unchanged. More formally,

$$\begin{aligned}
J_{input} &= Concat(S_{i,j}^T, S_i^T, S_j^T, V_i, C_i, P_j),\\
Y &= FullyConnected^2(J_{input}),\\
A_{input} &= Softmax(Y),\\
E_{input} &= AttentionDot([E_{i,j}^T, E_i^T, E_j^T], A_{input}),
\end{aligned} \tag{4}$$

where Y represents one intermediate result in neural network. At the same time, functions in these formulas are defined as follows:

- *Concat*(∗): concat of all matrices along the last dimension.
- *FullyConnected*^{n}(∗): n fully connected layers.
- $Softmax(*)$: the softmax layer.
- $AttentionDot(X, Y)$: weight the X with the Y.

4.2 Relation Temporal Attention Module

In the MIA module, we mix the input information into a multivariate time series, namely E_{input}. Then, we use this enhanced time series and static information, namely J_{input}, to forecast the talent demand

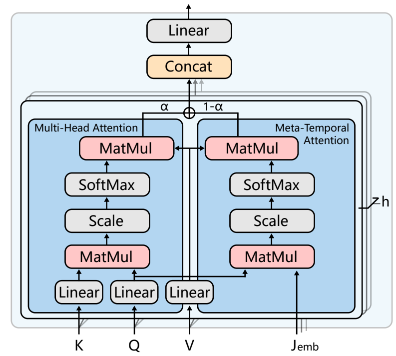

Figure 7: The structure of of Multi-Head Attention and Meta-Temporal Attention.

trend at time $T + 1$. In order to better model the talent demand time series, we further analyze the characteristics of recruitment market. In the traditional methods [5, 29, 34], autocorrelation is usually important to forecast future based on historical values. In Figure 5(a) shows the autocorrelation coefficient of fine-grained talent demand time series in a company A and a position. The Figure 5(b) shows the autocorrelation coefficient of talent demand time series in this company and all positions. Correspondingly, Figure 5(e),(f) show the autocorrelation coefficient in a different company. It can be seen that the autocorrelation characteristics of different companies are different. This is due to the different strategies of different companies in the recruitment market. How to flexibly model the different talent demand characteristics of different companies is a major challenge for talent demand forecasting. Meanwhile, it can be seen that although Figure 5(c),(d) is different from the company in Figure 5(a),(b), the autocorrelation characteristics are relatively similar. The reason is that the type of company in Figure 5(c),(d) is similar to the company in Figure 5(a),(b). How to further describe the recruitment relationship between different companies is also a big challenge. Similarly, different positions can also reflect characteristics of talent demand.

Along this line, we design a novel attention-based module in TDAN for better flexibly modeling the market patterns of talent demands, which is named RTA module. Specifically, the RTA module mainly consists of *multi-head attention* and *meta-temporal attention*. The multi-head attention [25] can effectively capture the autocorrelation between different time steps, however, the global attention and the characteristics of different companies and positions are missed. Therefore, we propose meta-temporal attention mechanism to capture these information and utilize sequential information and additional statistical information.

The *Relation Temporal Attention* part in Figure 6 shows the overall structure of RTA module, and Figure 7 shows the detailed structure of multi-head attention block and meta-temporal attention block. Firstly, we use one fully connected layer to transform E_{input} into the embedded inputs, named E_{emb}, which will feed to multi-head attention block and meta-temporal attention block. Meanwhile, the J_{input} will be transformed into the embedded static information, named J_{emb}, by two fully connected layers. Specifically,

$$\begin{aligned}
E_{emb} &= FullyConnected(E_{input}),\\
J_{emb} &= FullyConnected^2(J_{input}).
\end{aligned} \tag{5}$$

Then, after adding the sinusoidal position encoding [31] for involving absolute position information, E_{emb} will go through the multi-head attention block. In this block, the multi-head attention can use ensemble self-attention to capture dependencies of a single sequence. And the self-attention attention function is defined as:

$$Attention(\mathbf{Q},\mathbf{K},\mathbf{V}) = Softmax(\frac{\mathbf{Q}\mathbf{K}^{\mathbf{T}}}{\sqrt{d_k}})\mathbf{V}, \tag{6}$$

where $\mathbf{Q},\mathbf{K},\mathbf{V}$ are all correspond to three embeddings of E_{emb}(with position encoding) respectively, and d_k is the dimension of the embeddings vectors. The Equation 6 represents the *MatMul*, *Scale*, *SoftMax* and *MatMul* in multi-head attention block. This kind of mechanism can create a graph structure among the time points of the time series, where edges indicate the temporal dependencies. Moreover, we can create multiple attention graphs, i.e, h graphs in Figure 7 , each of which is defined by different parameters. Then, we ensemble them together to capture different types of dependencies, namely multi-head attention mechanisms [31]. Meanwhile, $\mathbf{Q}$, $\mathbf{V}$ and J_{emb} will be used in meta-temporal attention block. First, through *MatMul*, *Scale* and *SoftMax*, $\mathbf{Q}$ and J_{emb} will get the meta-temporal attention which can flexibly capture the information of the global attention and the characteristics of different companies and positions. And the meta-temporal attention will be applied to $\mathbf{V}$ by a *MatMul*. Thus, the attention function in meta-temporal attention block can be defined as:

$$MetaAttention(\mathbf{Q},\mathbf{V},J_{emb}) = Softmax(\frac{\mathbf{Q}{J_{emb}}^{\mathbf{T}}}{\sqrt{d_k}})\mathbf{V}. \tag{7}$$

Then the outputs of multi-head attention block and meta-temporal attention block, named E_{multi} and E_{meta} will be weighted by α and $1-\alpha$ as E_{union}. Then, after a *Concat*, a *Linear* transformation and a residual concatenate operation (i.e., *Add & Norm*), the outputs will go through a *Feed Forward* part and a residual concatenate operation to obtain the final output of the self-attention module. Simply, this RTA module can be formalized as:

$$\begin{aligned}
E_{multi} &= MultiHead(E_{emb}, W_{multi}),\\
E_{meta} &= MetaTemporal(E_{emb}, J_{emb}, W_{meta}),\\
E_{union} &= \alpha E_{multi} + (1-\alpha)E_{meta},
\end{aligned} \tag{8}$$

where the function $MultiHead(*)$ represents the whole *multi-head attention* and the function $MetaTemporal(*)$ represents the whole *meta-temporal attention*. Moreover, the W_{multi} and W_{meta} are all trainable parameters in two blocks, respectively.

Then, the RTA module will be stacked K times, and the shape of the outputs is too large to flat into one vector. Therefore, we use *Dense Interpolation* [25] to reduce the dimensions and only select the last vector as the F_{union}. Then, we let it go through a fully connected layer and a softmax layer for the final trend result, named y. More formally,

$$\begin{aligned}
F_{union} &= DenseInterpolation(E_{union}),\\
p &= Softmax(FullyConnected(F_{union})).
\end{aligned} \tag{9}$$

The main optimization goal in our network is the cross-entropy loss, which is widely used in classification problems. The loss function of TDAN can be written by:

$$Loss_{main} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{m} y_{i,k}\log p_{i,k} + \lambda||w||^2, \tag{10}$$

where $p_{i,k}$ represents the prediction, $y_{i,k}$ represents the input label, n is the number of the input batch, and m is the number of categories. The second part is the $L2$ regularization term for trainable parameters, and λ is the regularization coefficient, reflecting the degree of regularization constraints.

5 EXPERIMENTS

In this section, we validate the performance of TDAN framework through experiments on a real-world data set. After that, some intuitive case studies and discussions will be presented.

5.1 Dataset

In our research, the experiments were conducted on a real-world data set collected from an online recruiting market. Specifically, the raw data contain 355,465 job postings created by 152 companies from June 2015 to August 2018, involving 16 categorized job titles. we got 2,432 fine-grained talent demand time series, where we set each time interval to be one month. The details of data preprocessing can be found in Appendix C.

5.2 Experimental Settings

In this subsection, we introduce the settings of our experiments[1] in the talent demand forecast. Our model was trained with Adam Optimization Algorithm [16] and the learning rate was set as 0.002, with the batch size as 32 and the dropout as 0.1. All neural network models are implemented by the TensorFlow framework [2]. Since we have regarded the demand forecast as a multi-class classification task, we use *micro-F1* metric to measure the overall effectiveness. Considering the classification is unbalanced, we also use *macro-F1* and *weighted-F1* metrics to measure the overall performance in each class. The main parameters of the proposed TDAN framework and the details of evaluation metrics can be found in Appendix D.

Baseline methods. To comprehensively validate the performance of TDAN framework, we compared it with eight types of baseline methods as follows:

- *Benchmark*, which is the fundamental method that simply use t value to predict $t+1$ value.
- *ARIMA* [34] and *Prophet* [29], which are well-known traditional solutions for time series forecast task.
- *CNN* [37], which is a widely used neural network machine learning method for regression and classification problems.
- *LSTM* [13, 14] and *BiLSTM* [12], which are representative RNN methods for capturing the temporal information in sequence data, and widely used for time series prediction.
- *LSTNet* [17], which is a CNN+RNN time series modeling method that has been applied in many tasks. This method can capture the short-term local dependency patterns and long-term time series trend patterns.
- *Transformer* [25], which is a sequence method that introduces attention-model based architecture for processing multi-variate time-series data.
- *TCN* [4], which is a generic temporal convolutional network architecture that is applied across many tasks.
- *TDAN-MIA* and *TDAN-RTA*, which are our TDAN framework without RTA and our TDAN framework without MIA, respectively, for evaluating the performance of RTA module and MIA module.

5.3 Overall Results

The overall results of different approaches are shown in Table 1. Obviously, our TDAN method consistently outperforms all the baselines in all metrics, which validates the effectiveness of our

[1]Details can be found at https://github.com/7-Z-7/TDAN.

Table 1: **The overall performance of different approaches.**

Method	Micro-F1	Macro-F1	Weighted-F1
Benchmark	0.2569	0.0817	0.1050
ARIMA	0.2811	0.1666	0.1850
Prophet	0.3175	0.2800	0.2770
LSTM	0.3619	0.3479	0.3536
BiLSTM	0.3672	0.3496	0.3583
CNN	0.3847	0.3194	0.3314
LSTNet	0.3784	0.3538	0.3640
Transformer	0.3901	0.3193	0.3361
TCN	0.4034	0.3145	0.3333
TDAN-MIA	0.4198	0.3534	0.3712
TDAN-RTA	0.4207	0.3547	0.3726
TDAN	**0.4355**	**0.3693**	**0.3868**

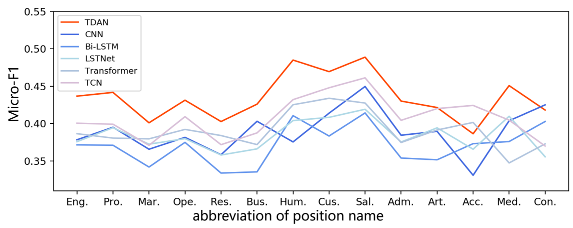

Figure 8: Evaluation on each position.

approach for fine-grained talent demand forecast. And from the results of TDAN-MIA and TDAN-RTA, we can find that MIA module and RTA module also can improve the prediction performance respectively, which validates the effectiveness of each single module. To be specific, the micro-F1 value of our method is 15.09% higher than LSTNet, 7.96% higher than TCN, and 11.64% higher than Transformer, the macro-F1 value of our method is 15.66% higher than Transformer, 17.42% higher than TCN and 4.38% higher than LSTNet, and the weighted-F1 value of our method is 15.08% higher than Transformer, 16.05% higher than TCN and 6.26% higher than LSTNet. It is notable that the Transformer model does not perform well in terms of macro-F1 and weighted-F1. This is because the model only focuses on the sequence itself, and the trained model is difficult to flexibly capture different characteristics of talent demands in different recruitment environment, so the model prefers to the common large categories. Our model not only improves the performance on micro-F1, but also greatly improves the performance on macro-F1 and weighted-F1. The MIA module can use the coarse-grained recruitment environment time series to enhance the fine-grained time series, and the RTA module can use the static information to flexibly capture the relationships and differences of companies and positions. Therefore, our TDAN method which combines MIA and RTA modules comprehensively improves the performance of fine-grained talent demand forecast.

5.4 Detailed Evaluations and Discussions

Evaluation on each positions. To further investigate the performance of our framework, we evaluate models on each position. As shown in Figure 8, we mainly compare our model with 4 representative models with good overall performance. The abbreviation of position names in abscissa are the first three characters of the job names, which refer to Table 7 for detail. There are only 14 positions in the figure because the time series of two positions are filtered out as shown in Appendix C. It can be seen that TDAN model achieves significant and stable improvement in almost every position. This further proves the effectiveness and stability of TDAN model.

Table 2: Performance evaluation with different *dim*.

dim	Micro-F1	Macro-F1	Weighted-F1
1	0.4218	0.3521	0.3699
2	0.4294	0.3518	0.3684
4	**0.4355**	**0.3693**	**0.3868**
6	0.4311	0.3659	0.3838
8	0.4226	0.3406	0.3606

Table 3: Performance evaluation with different L.

L	Micro-F1	Macro-F1	Weighted-F1
2	0.4075	0.3469	0.3643
4	0.4241	0.3585	0.3748
6	**0.4355**	**0.3693**	**0.3868**
8	0.4190	0.3538	0.3715
10	0.4075	0.3413	0.3604

Table 4: Performance evaluation with different α.

α	Micro-F1	Macro-F1	Weighted-F1
0	0.4122	0.3469	0.3643
0.25	0.4239	0.3559	0.3742
0.5	**0.4355**	**0.3693**	**0.3868**
0.75	0.4248	0.3527	0.3709
1.0	0.4198	0.3535	0.3712

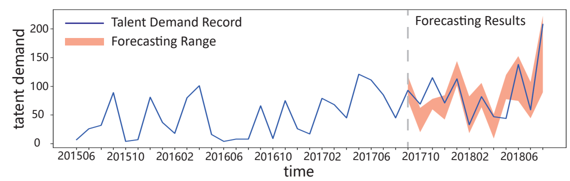

Figure 9: An example of talent demand forecast.

Parameter sensitiveness of *dim*. To evaluate the parameter sensitiveness, first, we conducted experiments with five different *dim*, i.e., 1, 2, 4, 6 and 8. The number 1 of *dim* means no dimension extension of univariate time series. In univariate time series, each time point has only one value, which makes the information of each point very limited and difficult to be modeled by neural network. The dimension expansion method we designed can use the time series itself for upsampling, so that each point contains a sequence of short time. This can not only enrich the information at each time point, but also help to learn the temporal patterns. The results are shown in Table 2, and the dimension expansion can improve the forecast performance. According to the results, it seems that a value around 4 might be the best choice.

Parameter sensitiveness of L. Then, we conducted experiments with five different observed time length L, i.e., 2, 4, 6, 8 and 10. The results are shown in Table 3. It can be seen that a larger observed time could provide more information that could improve the prediction accuracy, but when L is larger than 6, the accuracy starts to decline. This is because in the process of data set, a larger L could lead to the smaller number of training set when the number of testing set fixed. For example, when $L = 6$, 16,477 complete training samples can be obtained, and when $L = 10$, there are only 13,647 complete training samples. However, this also shows that when L is larger than 6, the improvement of prediction accuracy is insignificant. Therefore, in the talent demand forecasting problem, the last 6 months are important, which can reflect the recruitment trend.

Parameter sensitiveness of α. Furthermore, we conducted experiments with five different weight coefficient α of RTA module, i.e., 0, 0.25, 0.5, 0.75 and 1.0. The number 0 of α means only meta-temporal attention block, and the number 1.0 of α means only multi-head

Table 5: Average talent demands in different attention distributions of MIA module.

	$A_{com}, A_{pos} < 0.1$	$A_{com} \geq 0.1$	$A_{pos} \geq 0.1$
Average Talent Demand	9.52	3.67	3.84

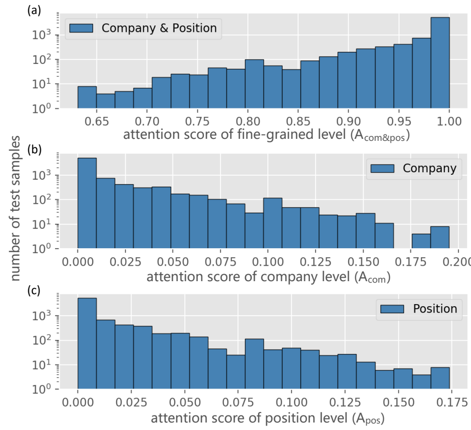

Figure 10: Distributions of attentiion score in MIA module.

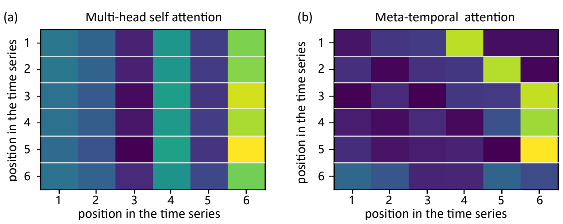

Figure 11: Attention maps of multi-head attention and meta-temporal attention.

attention block. The results are shown in Table 4. It can be seen that the performance of using each block alone is limited, and the combination of two blocks can greatly improve the prediction performance, such as $\alpha = 0, 5$. It is because these two blocks focus on different important features. The multi-head attention block focuses on the information of sequence with absolute position information, and the meta-temporal attention block focuses on various environmental and relative positional information. The balanced combination of these two blocks can capture more comprehensive and effective features for talent demand forecasting.

Case study. Based on the forecast results of five trend categories, we can estimate the scope of future talent demand, as shown in Figure 9. This figure shows the real talent demand of one company in the engineering position (i.e., the blue line) and the forecast results of our model (i.e., the red area). This red area is drawn based on the classification results. It can be seen that our trend forecast method can be effectively transformed into the forecast of the talent demand number.

Effect of attention in MIA module. In the process of mixing input, the attention mechanism in MIA module gives different attention scores to different levels of talent demands. In order to further discuss the effects of the MIA module, we count the distribution of attention scores. In Figure 10, (a) shows the attention scores distribution of fine-grained level talent demand (i.e, $E_{i,j}$), which is the subject time series for forecasting. And Figure 10(b) and (c) show

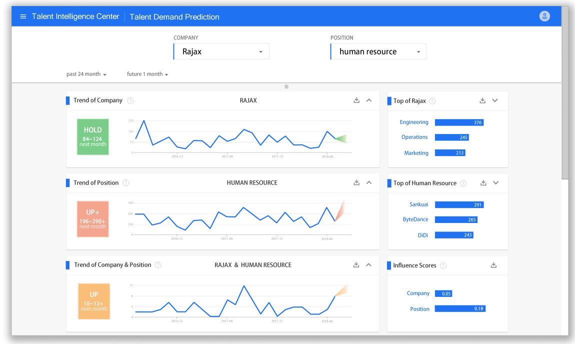

Figure 12: The screenshot of our deployed system.

the attention score distribution of coarse-grained talent demands (i.e, E_i, E_j), which are used for enriching the fine-grained talent demand time series. It can be seen that many samples do not have high attention of coarse-grained talent demands. This result means that not all fine-grained talent demands need a large amount of coarse-grained information, and many fine-grained talent demands can be effectively predicted with only a small amount of enriching. Then, Table 5 further shows the average value of fine-grained talent demand under different attention scores. It can be seen that when the average talent demand is small, the attention mechanism of MIA will automatically pay more attention to coarse-grained talent demand time series for enriching the fine-grained talent demand, thereby improving the prediction performance. When the average talent demand is large, the MIA module will automatically pay less attention to coarse-grained talent demand time series.

Effect of attention in RIA module. Then, for discussing attention mechanism in the RTA module, we visualize the attention maps of two blocks in this module. Figure 11 shows a representative example. The abscissa and ordinate in the figure are the positions in the time series, and each block represents the attention score of the ordinate position to the abscissa position (e.g., the first row represents the attention of the zeroth time point in the time series to other time points). The darker color of the square means a lower attention score, and vice versa. It can be seen that two blocks of RTA module are focus on different kinds of features. Specifically, The vertical pattern in Figure 11(a) shows the multi-head attention block could pay similar attention to the same absolute position of time series, which contains the characteristic of the time series itself. Meanwhile, The diagonal pattern in Figure 11(b) shows the meta-temporal attention block prefers to pay similar attention to the same relative position, which also contains the various characteristics of recruitment environment. The combination of these two kinds of blocks could help our model to capture more comprehensive information for improving the forecasting performance.

5.5 System Demonstration

We have deployed TDAN as an important functional component of intelligent recruitment system of cooperative partner, which can provide timely market insights for both hiring managers and recruiters. Specifically, users can search company and position names in the system, then, the system will show the detailed talent demand analyses of the company, the position and the position in the company (company & position) respectively. The analyses contain the trend forecasting for next month, the prediction of recruitment numbers in next month and the historical talent demand sequence. Based on these analyses, human resource experts can grasp the trends of the recruitment market and formulate appropriate recruitment strategies to maintain the business competitive edge. If the user only searches company or position, there is only one talent demand analysis of the company or the position. We regularly update the database with new transaction data and train a new model upon each data refresh. The TDAN model is trained offline so that users can get an instant result after submitting company and position names. One example of our prototype system is shown in Figure 12.

6 CONCLUSION

In this paper, we proposed a data-driven neural sequential approach, namely Talent Demand Attention Network (TDAN), for forecasting the fine-grained talent demand in recruitment market. Specifically, we first propose to augment the univariate time series of talent demand at multiple grained levels and extract intrinsic attributes of both companies and job positions with matrix factorization techniques. Then, we design a Mixed Input Attention module to capture company trends and industry trends to alleviate the sparsity of fine-grained talent demand. Meanwhile, we design a Relation Temporal Attention module for modeling the complex temporal correlation that changes with the company and position. In particular, the mixed input attention module can use intrinsic properties, high-level time series and side information to enrich the trend information and capture the relationship between different companies and job positions. Finally, extensive experiments on a large-scale real-world online recruitment dataset clearly validate the effectiveness of our approach for fine-grained talent demand forecast, as well as its interpretability for modeling recruitment trends.

REFERENCES

[1] Afshin Abadi, Tooraj Rajabioun, and Petros A Ioannou. 2014. Traffic flow prediction for road transportation networks with limited traffic data. *IEEE transactions on intelligent transportation systems* 16, 2 (2014), 653–662.

[2] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *USENIX 2016*. 265–283.

[3] Ayodele Ariyo Adebiyi, Aderemi Oluyinka Adewumi, and Charles Korede Ayo. 2014. Comparison of ARIMA and artificial neural networks models for stock price prediction. *Journal of Applied Mathematics* 2014 (2014).

[4] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Recurrent Networks for Sequence Modeling. *arXiv:1803.01271* (2018).

[5] Domenico Benvenuto, Marta Giovanetti, Lazzaro Vassallo, Silvia Angeletti, and Massimo Ciccozzi. 2020. Application of the ARIMA model on the COVID-2019 epidemic dataset. *Data in brief* (2020), 105340.

[6] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. 2017. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691* (2017).

[7] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

[8] Chi Chen, Li Zhao, Jiang Bian, Chunxiao Xing, and Tie-Yan Liu. 2019. Investment behaviors can tell what inside: Exploring stock intrinsic properties for stock trend prediction. In *KDD 2019*. 2376–2384.

[9] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[11] Chenyou Fan, Yuze Zhang, Yi Pan, Xiaoyue Li, Chi Zhang, Rong Yuan, Di Wu, Wensheng Wang, Jian Pei, and Heng Huang. 2019. Multi-horizon time series forecasting with temporal attention learning. In *KDD 2019*. 2527–2535.

[12] Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm.. In *Interspeech*. 715–719.

[13] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. 2020. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting* (2020).

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[15] Ji-nan Jia and Zhen-ping Liu. 2010. Study on demand forecasting of enterprise management human resources based on MRA-OLS. In *2010 International Conference on Management and Service Science*. IEEE, 1–4.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR 2018*.

[18] Liangyue Li, How Jing, Hanghang Tong, Jaewon Yang, Qi He, and Bee-Chung Chen. 2017. Nemo: Next career move prediction with contextual embedding. In *WWW 2017*. 505–513.

[19] Zachary C Lipton, John Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019* (2015).

[20] ED McKenzie. 1984. General exponential smoothing and the equivalent ARMA process. *Journal of Forecasting* 3, 3 (1984), 333–344.

[21] Chuan Qin, Hengshu Zhu, Tong Xu, Chen Zhu, Liang Jiang, Enhong Chen, and Hui Xiong. 2018. Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In *SIGIR 2018*. 25–34.

[22] Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. 2017. Stock price prediction using LSTM, RNN and CNN-sliding window model. In *icacci 2017*. IEEE, 1643–1647.

[23] Baoxu Shi, Jaewon Yang, Feng Guo, and Qi He. 2020. Salience and Market-aware Skill Extraction for Job Targeting. In *KDD 2020*. 2871–2879.

[24] Joseph Sia. 2013. University choice: Implications for marketing and positioning. *Education* 3, 1 (2013), 7–14.

[25] Huan Song, Deepta Rajan, Jayaraman J Thiagarajan, and Andreas Spanias. 2018. Attend and diagnose: Clinical time series analysis using attention models. In *AAAI 2018*.

[26] Ewout W Steyerberg et al. 2019. *Clinical prediction models*. Springer.

[27] Youqiang Sun, Jiuyong Li, Jixue Liu, Bingyu Sun, and Christopher Chow. 2014. An improvement of symbolic aggregate approximation distance measure for time series. *Neurocomputing* 138 (2014), 189–198.

[28] Ying Sun, Fuzhen Zhuang, Hengshu Zhu, Qi Zhang, Qing He, and Hui Xiong. 2021. Market-oriented job skill valuation with cooperative composition neural network. *Nature communications* 12, 1 (2021), 1–12.

[29] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72, 1 (2018), 37–45.

[30] José F Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. 2020. Deep Learning for Time Series Forecasting: A Survey. *Big Data* (2020).

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS 2017*. 5998–6008.

[32] Xunxian Wu, Tong Xu, Hengshu Zhu, Le Zhang, Enhong Chen, and Hui Xiong. 2019. Trend-aware tensor factorization for job skill demand analysis. In *IJCAI 2019*. AAAI Press, 3891–3897.

[33] Bailin Yang, Shulin Sun, Jianyuan Li, Xianxuan Lin, and Yan Tian. 2019. Traffic flow prediction using LSTM with feature enhancement. *Neurocomputing* 332 (2019), 320–327.

[34] G Peter Zhang. 2003. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* 50 (2003), 159–175.

[35] Le Zhang, Hengshu Zhu, Tong Xu, Chen Zhu, Chuan Qin, Hui Xiong, and Enhong Chen. 2019. Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model?. In *The World Wide Web Conference*. 2312–2322.

[36] Yuan ZHANG, LI Feng, and LIU Bo. 2017. Research on the Forecasting Model of Total Human Resource Demand of Large Central Enterprise Group Based on CD Production Function. *DEStech Transactions on Computer Science and Engineering* itme (2017).

[37] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. 2014. Time series classification using multi-channels deep convolutional neural networks. In *International Conference on Web-Age Information Management*. Springer, 298–310.

[38] Chen Zhu, Hengshu Zhu, Hui Xiong, Pengliang Ding, and Fang Xie. 2016. Recruitment market trend analysis with sequential latent variable models. In *KDD 2016*. 383–392.

[39] Chen Zhu, Hengshu Zhu, Hui Xiong, Chao Ma, Fang Xie, Pengliang Ding, and Pan Li. 2018. Person-job fit: Adapting the right talent for the right job with joint representation learning. *ACM Transactions on Management Information Systems (TMIS)* 9, 3 (2018), 1–17.

[40] Hengshu Zhu, Hui Xiong, Fangshuang Tang, Qi Liu, Yong Ge, Enhong Chen, and Yanjie Fu. 2016. Days on market: Measuring liquidity in real estate markets. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 393–402.

APPENDIX

A Mathematical Notations

Table 6: Some important mathematical notations.

Symbol	Description
$D_{i,j}^T$	The talent demand time series of company i and position j before time T
$d_{i,j}^t$	The job posts number at time slice t in $D_{i,j}^T$
D_i^T	The talent demand time series of company i before time T
D_j^T	The talent demand time series of position j before time T
$F_{i,j}^T$	The differential time series of $D_{i,j}^T$
$S_{i,j}^T$	The SAX time series of $F_{i,j}^T$
$s_{i,j}^t$	The talent demand trend at time slice t in $S_{i,j}^T$
$E_{i,j}^T$	The extension time series of $S_{i,j}^T$
I_i	The side information of company i
V_i	The company embedding vector of I_i
R^T	The company & position matrix before time T
$r_{i,j}$	The talent demand number of company i and position j in R
C_i	The intrinsic properties of company i
P_j	The intrinsic properties of position j
N	The number of companies
M	The number of job positions
dim	The dimensions of extension time series in a time
L	The length of the $D_{i,j}^T$
α	the weight coefficient of the RTA module

B SAX Algorithm

We use $S_{i,j}^T$ to denote the symbolic time series in the last figure. Formally, the procedure can be formulated as

$$\begin{aligned}
S_{i,j}^T &= \left\{s_{i,j}^{T-L}, \ldots, s_{i,j}^T\right\}, \\
std_{i,j} &= STD(D_{i,j}^T), \\
s_{i,j}^1 &= SAX(0), \\
s_{i,j}^t &= SAX\left(\frac{d_{i,j}^t - d_{i,j}^{t-1}}{std_{i,j}}\right) \quad (t > 1), \\
SAX(x) &= \begin{cases} -1 & x < -0.84 \\ -0.5 & -0.84 \le x < -0.25 \\ 0 & -0.25 \le x < 0.25 \\ 0.5 & 0.25 \le x < 0.84 \\ 1 & 0.84 \le x \end{cases},
\end{aligned} \tag{11}$$

where $STD(*)$ means calculating the standard deviation, and the range of SAX (i.e., −1, −0.5, 0, 0.5, 1) corresponds to the five trends in order. The $S_{i,j}^T$ directly represents the trend of talent demand. Meanwhile, D_i^T, D_j^T, D^T are transformed in the same way.

C Data Pre-processing

To avoid noise and format the data for experiments, we pre-processed the data set by *job title categorization.*

Job title categorization. In each job posting, the job title was manually filled by the user. Therefore, different job titles may correspond to the similar job position. To reduce noise and normalize the job titles, we categorized the original job titles into 16 categories by the classification approach mentioned in [35]. The detailed results of job title categorization in our dataset are shown in Table 7.

Table 7: Job title category and distribution.

Category	Number
Business-development	13,181
Human Resources	10,837
Accounting	5,548
Media-and-communication	10,024
Customer-service	12,166
Engineering	88,355
Marketing	18,188
Legal	1,326
Research	29,191
Arts-and-design	16,584
Administrative	11,776
Operations	35,563
Purchasing	1,225
Sales	15,208
Program-and-project-management	28,697
Consulting	14,100

After that, we got 2,432 fine-grained talent demand time series, where we set each time interval to be one month. Moreover, the companies usually only have continuous talent demand in some positions (not all 16 positions) according to their main businesses, so there are few job postings in other positions. Such as some catering companies may not recruit many research positions, and the prediction in this situation is not needed because the demands are always zero. For this consideration, we further filtered the company-position pairs to guarantee the effectiveness of evaluation. Specifically, we removed the company-position pair if its monthly averaged talent demand of it is less than 2. Then, we retained 638 fine-grained talent demand time series. Meanwhile, we got 152 company talent demand time series and 16 position talent demand time series and 1 full talent market demand time series (i.e., 638 $D_{i,j}^T$, 152 D_i^T, 16 D_j^T, 1 D^T). In the experiments, we set the input length of time window as 6 months, then each sample in our data set is composed of a time series of length 6 and a classification label. Moreover, we used the last 12 months (i.e., 2017.09 to 2018.08) for the test. Finally, we obtained 16,477 samples for training and validation, and 9684 samples for testing.

Table 8: The values of parameters in our experiments.

Parameter	Value
The Dimensions of Intrinsic Properties (dim_{MF})	8
The Dimensions of Input Embedding ($dim_{embedding}$)	16
The Dimensions of Extension Time Series (dim)	4
The Length of Observed Time (L)	6
The Number of Head in Multi-head Attention ($head_n$)	8
The Dimensions of Head in Multi-head Attention ($head_d$)	8
The Dimensions of Feed Forward ($feed_d$)	128
The Weight Coefficient of RTA Module (α)	0.5
The Regularization Coefficient of All Trainable Parameters (λ)	0.0005

D Experimental Settings

Details of implementation. The main parameters of the proposed TDAN framework are shown in Table 8. Under these settings, the model was trained with Adam Optimization Algorithm [16] and the learning rate was set as 0.002, with the batch size as 32 and the dropout as 0.1. All neural network models are implemented by the TensorFlow framework [2].

Evaluation metrics. Since we have regarded the demand forecast as a multi-class classification task, we use *micro-F1* metric to measure the overall effectiveness. Considering the classification is unbalanced, we also use *macro-F1* and *weighted-F1* metrics to measure the overall performance in each class. The number of true positive examples in category i is denoted as TP_i. Similarly, the number of false positive examples, the number of false negative examples and the number of true negative examples in category i are denoted as FP_i, FN_i and TN_i, respectively. More formally,

$$
\begin{aligned}
& micro-F1 = \frac{2 \cdot Pr \cdot Re}{Pr + Re}, \\
& Pr = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FP_i}, \\
& Re = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FN_i}, \\
& macro-F1 = \frac{1}{\sum_i 1} \sum_i \frac{2 \cdot Pr_i \cdot Re_i}{Pr_i + Re_i}, \\
& Pr_i = \frac{TP_i}{TP_i + FP_i}, \\
& Re_i = \frac{TP_i}{TP_i + FN_i}, \\
& Cn_i = TP_i + TN_i + FP_i + FN_i, \\
& macro-F1 = \frac{1}{\sum_i Cn_i} \sum_i \frac{2 \cdot Pr_i \cdot Re_i}{Pr_i + Re_i} Cn_i .
\end{aligned}
\tag{12}
$$