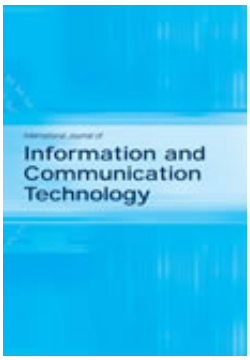
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Prediction of talent demand and job matching based on knowledge graph and attention mechanisms

Xiaoli Mei

Article History:

Received:	27 September 2024
Last revised:	09 October 2024
Accepted:	11 October 2024
Published online:	13 December 2024

Prediction of talent demand and job matching based on knowledge graph and attention mechanisms

Xiaoli Mei

School of Management,
Jiangxi University of Technology,
Jiangxi 330098, China
Email: 13979129140@163.com

Abstract: The real-time job data from recruitment platforms, reflecting the demands of enterprises for job seekers, can provide data support for the development of university training policies. This study proposes a knowledge representation model based on the self-attention mechanism for talent demand and job matching prediction method, where neural networks aggregate neighbourhood information to generate better node representations. In doing so, nodes can learn their local neighbourhood information and the whole graph structure. At the same time, using the self-attention mechanism, further extracts node features containing rich neighbourhood information. Deep interactions between nodes are used to calculate the central entity around the attention coefficients of neighbours to improve the accuracy of prediction. The experimental results show that the model prediction is highly fault-tolerant, short time-consuming, and can be widely used in the practical application of matching talent demand and university talent cultivation.

Keywords: knowledge graph; attention mechanism; talent demand; job matching.

Reference to this paper should be made as follows: Mei, X. (2024) 'Prediction of talent demand and job matching based on knowledge graph and attention mechanisms', *Int. J. Information and Communication Technology*, Vol. 25, No. 9, pp.76–87.

Biographical notes: Xiaoli Mei received her Bachelor's degree from Zhejiang University in 2010. She is currently a Lecturer at the School of Management at Jiangxi University of Science and Technology. Her research interests include business enterprise management, human resource management, and recruitment and talent assessment.

1 Introduction

In modern society, with the widespread use of the internet, an increasing number of enterprises through the recruitment network platform as a medium to release its recruitment information, the rise of online recruitment can effectively address the limitations of current data statistics, serving as a prominent channel that reflects market demand for talent. Information is centralised on the web, characterised by a large volume of data, it is difficult to accurately rely on search engines to obtain effective intelligence. In view of the above problems, this topic integrates the data of network recruitment

platform, integrates the development trends of the new information society and examines the professional alignment of college talents from both data and theoretical perspectives. This approach aims to precisely understand the characteristics of market demand and offer strategies for talent cultivation and curriculum development in colleges and universities.

Knowledge graph, as an emerging data representation, is able to structure discrete data information (Hao et al., 2021), reveal complex relationships between data, and perform well in the fields of information retrieval and recommendation system (Guo et al., 2020). The attention mechanism, on the other hand, is extensively utilised in areas like natural language processing and computer vision, improving the predictive ability and interpretability of models by focusing on important information (Xu et al., 2021). Combining knowledge graph with attention mechanism and applying it to talent demand and job matching prediction is expected to break through the limitations of traditional methods and provide more intelligent solutions.

In the field of talent demand and job matching prediction, traditional methods mainly include rule-based matching and statistical-based matching. These methods usually rely on manually formulated rules or simple statistical models, and lack a deep understanding of complex relationships and semantic information, resulting in poor prediction results. In recent years, as a result of the development of artificial intelligence and big data technologies, more and more researchers have begun to explore methods founded on machine learning and deep learning to improve the accuracy and efficiency of prediction (Usama et al., 2020).

Li et al. (2022) provided an in-depth study and analysis of combining convolutional neural networks (CNN) and BP neural network algorithms in a workforce resources recruitment demand forecasting model. BP neural network technology is introduced into the enterprise management talent assessment activities. Ni (2022) conducts job seeker matching and intelligent job recommendation based on deep learning algorithms to provide job seekers with more practical job search assistance. The intelligent job recommendation algorithm incorporates a bidirectional long short-term memory neural network (Bi-LSTM) and an attention-based neural network for person-job matching (APJFNN).

Based on the inspiration of the above work, this study proposes a talent demand and job matching prediction method that combines knowledge graph and attention mechanism, aiming to further improve the accuracy and efficiency of matching prediction, and provide a more intelligent solution for talent recruitment of enterprises.

This study aims to propose a talent demand and job matching forecasting approach based on knowledge graph and attention mechanism. Neural networks are used to aggregate neighbourhood information to generate better node representations, so that nodes can learn their local neighbourhood information and the whole graph structure, while the self-attention mechanism is used to further extract node features containing rich neighbourhood information, and the deep interactions between nodes are used to compute the attention coefficients of the neighbours around the central entity, so as to enhance the accuracy of matching predictions. This approach not only increases the efficiency of prediction, but also significantly improves the accuracy of prediction to meet the dual needs of enterprises and job seekers. In this paper, the construction method of the model, the experimental process and the analysis of the results will be introduced in detail, in order to serve as a reference for the research and application in related fields.

The main innovations and contributions of this work include:

1 Constructed a complete knowledge graph: we structured information such as job descriptions, job seekers' resumes and enterprise requirements to construct a knowledge graph containing rich relationships and attributes, systematically integrating multi-dimensional information between talents and jobs.

2 Introduced the attention mechanism: building on graph neural network, we introduced the attention mechanism, which enables the model to automatically focus on the features that have a greater impact on the matching results, thus improving the accuracy and interpretability of the prediction.

3 Proposed a new talent matching prediction method: the method proposed this study combines the benefits of knowledge graphs, graph neural networks, and the attention mechanism, forming an efficient and accurate prediction method for talent demand and job matching, and experimental results demonstrate that this method outperforms traditional approaches.

4 Provides feasibility verification of practical application: through experimental verification, the method proposed in this study has good feasibility and effect in practical application, which provides powerful help for the phenomenon of the disconnection between talent cultivation in colleges and universities and the talent demand of enterprises at this stage.

2 Relevant technologies

2.1 Knowledge graph representation learning

A knowledge graph can be represented as $G = \{(h, r, t)\} \in E \times R \in E$, E and R denote the set of entity nodes as well as the set of relationship edges respectively. A triple (h, r, t) denotes a fact, which is represented in the graph as the existence of an edge r between nodes h and t, where $h, t \in E$, $r \in R$, h, and t are referred to as the subject and object, respectively. The relational properties between entities are crucial to what distinguishes knowledge graphs from other types of graphs. Examining the structure of knowledge graphs reveals that relationships are a vital component. An entity can have multiple representations based on its relationships (Ji et al., 2021).

The knowledge graph representation learning model projecting entities and relations projected as distributed vectors, thus simplifying computation in knowledge graphs and making full use of machine learning or deep learning techniques and big data techniques. Given an input triad (h, r, t), $f(h, r, t)$ outputs the likelihood score that (h, r, t) is a valid triad; it is also possible to project the probability of (h, r, t) being a valid triad by giving any two of the triad missing entities or relations (Chen et al., 2020; Scarselli et al., 2008).

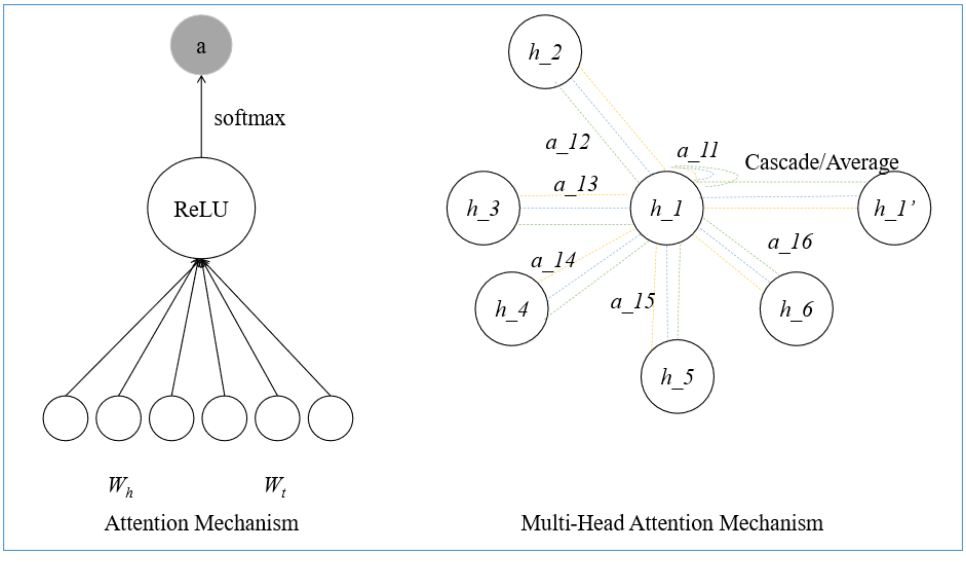
2.2 A knowledge representation model based on the self-attention mechanism

The brief diagram of the GAT model is shown in Figure 1. This diagram illustrates the working principles of the attention mechanism (left) and the multi-head attention mechanism (right).

The left side shows how to calculate the attention weight a between a node and its neighbouring nodes through the attention mechanism. First, the features of the neighbouring nodes are weighted (using W_k and W_t), passed through the ReLU activation function, and then normalised using the softmax function to obtain the attention weights.

The right side illustrates the structure of the multi-head attention mechanism. Here, node n_1 interacts with several neighbouring nodes (h_1, h_2, h_3, h_4, h_5, h_6), and the attention weights (a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{16}) are calculated for each interaction. These attention weights are used to compute the weighted sum of the neighbouring node features, which are then combined using either concatenation or averaging to generate the new node feature h_1'.

Figure 1 Brief diagram of GAT model (see online version for colours)



3 Job prediction based on self attention mechanism of knowledge representation model

3.1 Knowledge graph representation model

Symbolically, we define knowledge graph $G = \{(h, r, t)\} \in E \times R \in E$, E and R, which respectively represent entity node combinations and sets of relationship edges. (h, r, t) represents a factual triplet, indicating the existence of an edge r between nodes h and t in the graph, where h, $t \in E$, $r \in R$, h and t are respectively referred to as the subject and object, and r represents the relational predicate. The entity embedding matrix $E \in \Re^{N_e \times D_e}$, the relation embedding matrices $R \in \Re^{N_r \times D_r}$; in addition, define the word embedding matrix $M \in \Re^{N_w \times D_w}$. The embedding matrix after the graph attention layer, $E' \in \Re^{N_e \times D_e'}$ and $R' \in \Re^{N_r \times D_r'}$.

Next, we will offer an in-depth introduction to the various layers in the text enhanced knowledge graph representation learning algorithm TGAT based on graph attention networks.

3.1.1 Text convolutional layer

In order to extract more implicit information from descriptive information and improve model performance, we use text convolutional layers to extract information from text content. Firstly, all the processed text's words are converted into word vectors using the embedding layer to produce the embedding matrix *M*. Here, $M \in \Re^{N_w \times D_w}$ is the BERT pre trained word vector model (Huang et al., 2020; Gao et al., 2022).

For each entity's descriptive text information, sparse representations of several words are obtained through data preprocessing X_{sparse}. For each word, word vectors are extracted from the embedding matrix M, resulting in an embedding matrix A for each entity's descriptive text

$$A = X_{sparse} M \tag{1}$$

Here, $A \in \Re^{N_s \times D_w}$ and $X_{sparse} \in \Re^{N_s \times N_w}$ are sparse representations of input words, and N^s is the number of words segmented from the descriptive text of each entity. Similar to the convolution operation in images, we use convolution layer to extract the features of image A in computer vision. Considering the high correlation between adjacent words in the sentence, text convolution is performed along the text sequence, with the convolution kernel width matching the word vector dimension D_w. The height of the convolution kernel can be configured with various hyper parameters. Multiple convolution kernels of varying heights are set to perform convolution operations on the text features, resulting in multiple feature maps (Dhillon and Verma, 2020). Assuming the convolution kernel height is set to *h*, the corresponding convolution kernel matrix is $W \in \Re^{h \times D_w}$, and the embedding matrix obtained for each entity's descriptive text is $A \in \Re^{N_s \times D_w}$, the convolution operation is defined as:

$$o_i = W \circ A[i : i + h - 1], \qquad i = 1, 2, \ldots, s - h + 1 \tag{2}$$

Here, $A[i: j]$ represents the i^{th} to j^{th} rows of matrix A, and the extracted feature map is generated through a nonlinear activation function:

$$c_i = f(o_i + b) \tag{3}$$

For each convolution kernel, a feature vector $C \in \Re^{N_s - h + 1}$ can be generated to obtain multiple different feature maps (Vishnu et al., 2022; Zhang et al., 2023).

At the same time, corresponding to the (*R*, *G*, *B*) channels in image convolution, we also use multiple different channel modes to enable the convolution kernel to generate multiple feature maps during convolution operations, thus more comprehensively mining the potential features contained in the text.

According to different convolution kernels, feature maps of different sizes can be obtained. In order to make the feature dimensions the same and reduce model parameters to avoid overfitting problems, pooling operations are used to extract important information from multiple feature maps of the same dimension:

$$h_i = \max\left(c_0, \ldots, c_{N_s - h + 1}\right) \tag{4}$$

For e different sizes of convolution kernels, assuming each kernel contains 3 channels, concatenate the pooled feature map and activate it with a softmax function to extract the feature representation:

$$x_{text} = softmax\left(Wconcat\left(h_1, \ldots, h_{3k}\right) + b\right) \tag{5}$$

Common pooling operations include average pooling, max pooling, K-max pooling, and dynamic pooling, among others. Average pooling calculates the final feature value by taking the average; max pooling operation is commonly used in computer vision because it can extract strong features regardless of position, and the extracted features have rotational invariance; K-max pooling avoids the disadvantage of missing information in max pooling by taking K highest scoring features and preserving the order of concatenation of these feature values, thus preserving global information. We adopted this method to perform pooling operations on feature maps (Li et al., 2016).

In summary, text convolution first segments and encodes a serialised text to obtain a text vector matrix (Dubey et al., 2019). When constructing the convolution kernel, its width is set to the dimension D_w of the word vector, and its height can be set to different sizes such as 2, 3, 4, …. For serialised text, it is equivalent to contextual windows of different sizes, which can be considered a special case of convolution kernels in images (Ciancetta et al., 2020); next, pooling and concatenation operations are performed on the feature maps obtained from different convolution kernels to ultimately obtain low dimensional dense feature vectors of the text vector matrix.

3.1.2 Knowledge graph attention layer

For nodes and relationships, the vector representation of nodes and relationships is first trained through the TransE model using triplet structured data. Since this training process only includes triplet structured data, the vectors obtained by this method are defined as knowledge graph structural features (denoted as v^s).

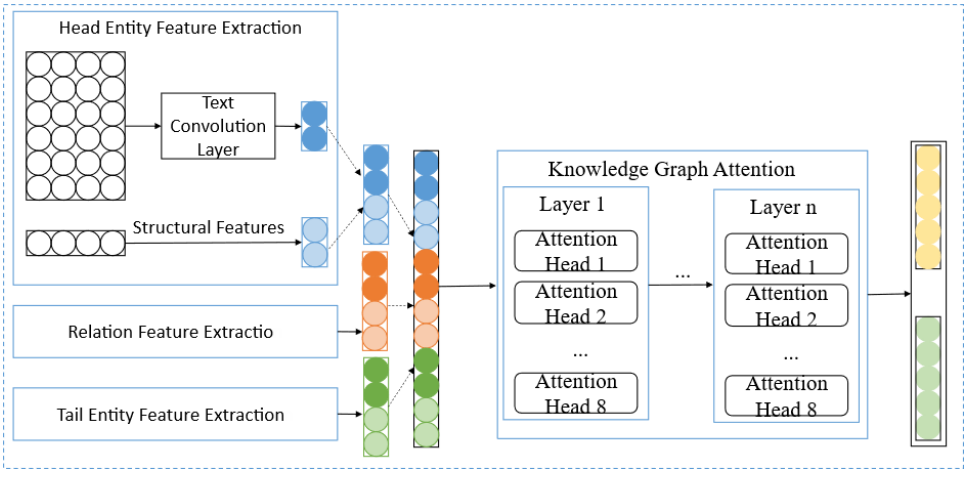
Then, corresponding text features (denoted as v^t) are obtained through text convolution operations for the corresponding text descriptions. Since this method obtains more semantic features of the text through natural language processing models, we define the vectors obtained by this method as knowledge graph text features.

Finally, in order to learn features from two different dimensions simultaneously, and considering the potential performance improvement caused by feature crossover, we performed feature extraction through concatenation operations and multi-layer perceptrons, ultimately obtaining a vector representation that integrates the two vector representations.

In order to enable the neighbouring nodes around the entity to have varying degrees of influence on the entity, on the one hand, we have mined the n-hop neighbours around the entity, thereby extending the concept of edges to directed paths. The embedding of auxiliary relationships is derived by summing the embeddings of all relationships in the path according to their attributes. At the same time, to avoid the accumulation of incoming information in the n-hop neighbourhood of the typical n-layer model, we standardise the entity embeddings after each graph attention layer; on the other hand, we designed a multi head attention neural network layer to represent entities, thereby

stabilising the learning process and encapsulating more information about the neighbourhood; at the same time, a multi-layer knowledge graph attention network layer is used to explore entity features at a deeper level, resulting in the final node vector representation $E' \in \Re^{N_e \times D'_e}$ and relationship vector representation $R' \in \Re^{N_r \times D'_r}$. The brief diagram of the model is shown in Figure 2.

Figure 2 Text enhancement graph attention network model brief diagram (see online version for colours)



We extract the text description feature v^t corresponding to entities in the knowledge graph through text convolution operation. Simultaneously, to integrate the structural feature v^s of the knowledge graph (initialised as TransE pre trained structural vector here), we concatenate the normalised structural feature and normalised text description feature, and finally merge them into entity vectors:

$$v_h = concat\left(v_h^t, v_h^s\right) \tag{6}$$

correspondingly, the definition of relationship vectors is as follows:

$$v_r = concat\left(v_r^t, v_r^s\right) \tag{7}$$

For the graph attention network layer in the TGAT model, the brief diagram is shown in Figure 3. The first step is the multi head attention mechanism. For each entity node e_i in the knowledge graph, its neighbours are defined as N_i, and R_i is defined as a set of relationships connecting entities e_i and e_j. To calculate the influence weight of neighbour N_i on node e_i, we introduce graph attention mechanism, defined as follows:

$$score_{(h,r,t)} = a\left(v_h, v_r, v_t\right) \tag{8}$$

Using attention mechanism, calculate an unnormalised attention coefficient of $score_{(h,r,t)}$ for each tuple (h, r, t) near node i. To avoid self influence, we use a single-layer feedforward neural network to calculate the attention weight values:

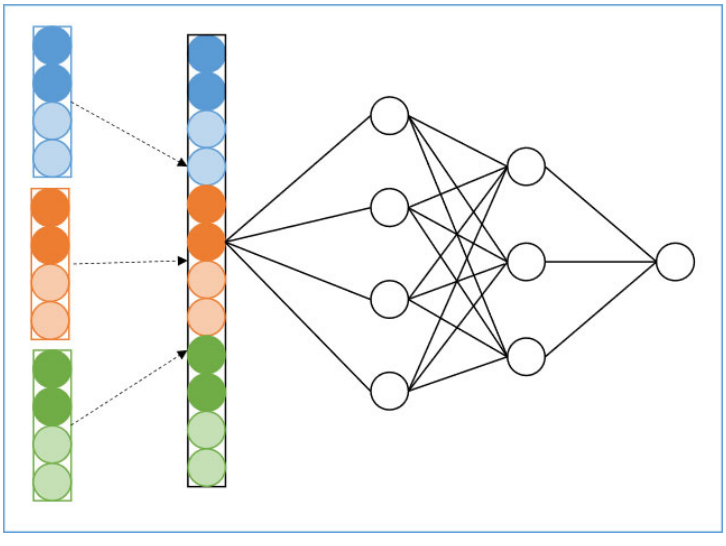
$$score_{(h,r,t)} = LeakyReLU\left(W_1, v_{(h,r,t)}\right) \tag{9}$$

Here, $v_{(h,r,t)}$ is the feature representation of the triplet (h, r, t), obtained by concatenating the entity and relationship feature vectors corresponding to the triplet (h, r, t) and performing a linear transformation:

$$v_{(h,r,t)} = W_2 \cdot concat\left(v_h, v_r, v_t\right) \tag{10}$$

Here, vectors v_h and v_t represent the embeddings of entities h and t, v_r represents the embeddings of edge r, and W_1 and W_2 represent the linear transformation matrices.

Figure 3 Brief diagram illustrating the attention value calculation process (see online version for colours)



Meanwhile, α_{ij} is defined as:

$$\alpha_{ij} = \frac{\exp\left(score_{(i,j,k)}\right)}{\sum_{n\in N_i}\sum_{r\in R_{in}} \exp\left(score_{(i,n,r)}\right)} \tag{11}$$

Here, N_i is defined as the neighbourhood of entity i, and R_{in} is defined as a set of relationships connecting entities i and n.

The definition of using multi head attention mechanism is as follows:

$$v_i' = \left(\Big|\Big|_{m=1}^{M} \sigma\left(\sum_{j\in N_i}\sum_{k\in R_{ij}} \alpha_{ij}^m v_{(i,j,k)}^m\right)\right)W \tag{12}$$

For the relationship embedding matrix, perform linear transformation to obtain the transformed relationship embeddings $R' \in \Re^{N_r \times D_t}$ and D_t, which are the shared output dimensions of entity and relationship embeddings:

$$R' = RW \tag{13}$$

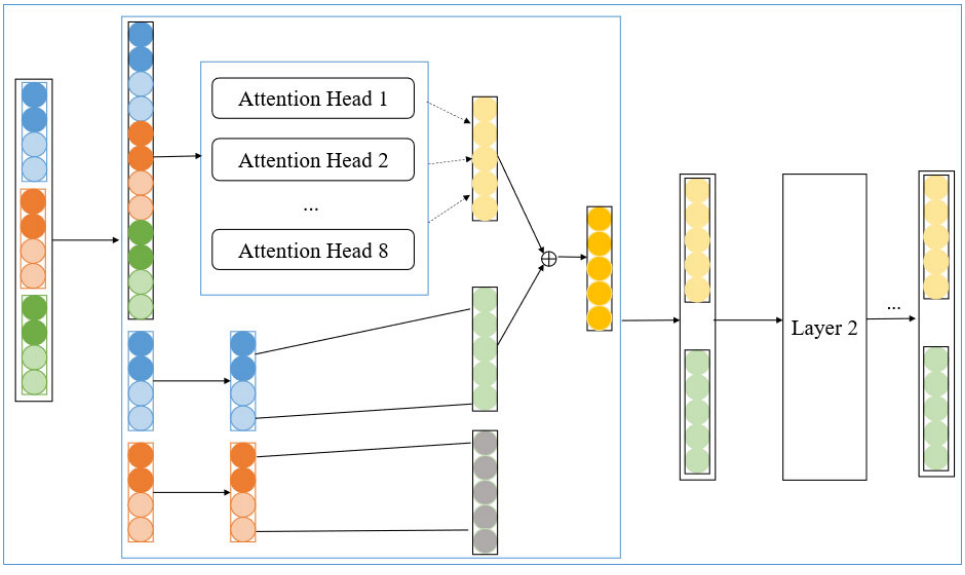
In undirected graphs that do not consider relationship categories, in order to enable the information of its own entities to play a role in the new embedded representation, a self looping strategy is usually added. However, performing self looping operations in a

knowledge graph would mean adding meaningless new relationship types, thereby allowing the information of its own entities to influence new embeddings; conversely, disregarding the previously stored information in entity embeddings can result in information loss. We use the entity embeddings obtained from the last attention layer overlaid with the original entity embeddings that have undergone linear transformation to solve this problem:

$$E' = EW^E + E^f \tag{14}$$

Here, E represents the entity embedding input into the model, $W^E \in \Re^{D_e \times D'_e}$ is defined as the linear transformation matrix, therefore EW^E represents the entity embedding after linear transformation, $E^f \in \Re^{N_e \times D'_e}$ is defined as the entity embedding obtained from the last attention layer, D_e represents the dimension of the initial entity embedding, and D'_e represents the dimension of the entity embedding generated by the graph attention layer. In this way, the entire text enhanced knowledge graph representation learning graph attention neural network layer is defined. In order to explore deeper information and make training more stable, we deepen the graph attention neural network layer to obtain a stable vector representation containing rich features. The brief diagram is shown in Figure 4.

Figure 4 Brief diagram of the knowledge graph attention layer (see online version for colours)



3.1.3 Knowledge graph attention layer

After building the TGAT model, in order to optimise the feature vectors, we borrowed the idea of the translation distance model scoring function in the definition of the scoring function. By learning the embeddings of entities and relationships, we make node h the nearest neighbour of node t connected by relationship r for a given effective triad (h, r, t). The distance function is defined as the l_2 norm of the distance between node h and node t:

$$d_{(h,r,t)} = \|h + r - t\|_2 \tag{15}$$

Here, we add l_2 constraints to the embedding representation, and the loss function is defined as:

$$\begin{aligned} L &= \sum_{(h,r,t)\in G} \sum_{(h,r,t)'\in G'} \left[d_{(h,r,t)'} - d_{(h,r,t)} + \gamma \right] + G' \\ &= \left\{ (h', r, t) \middle| h' \in E \backslash h \right\} \cup \left\{ (h, r, t') \middle| t' \in E \backslash t \right\} \end{aligned} \tag{16}$$

Here, G represents the collection of valid triples, G' represents the group of invalid triples with either the head or tail entities are randomly replaced, $[x]_+$ represents the positive part of x, and γ is the edge hyper parameter.

4 Experimental results and analysis

4.1 Dataset

This article takes big data positions as an example and constructs a talent demand and job matching knowledge graph through steps such as data information acquisition, knowledge extraction, and knowledge fusion. The graph data is stored locally using knowledge storage technology, and can also be stored in a graph database for easy visualisation. After obtaining knowledge graph data, we can apply knowledge representation algorithms to map entity/relational data in the knowledge graph to a dense vector space, which can be used as input for deep neural networks, providing the possibility for knowledge inference.

4.2 Evaluation indicators

This article employs the root mean square error (RMSE) as a key evaluation metric to assess the performance of the enterprise job demand forecasting model. RMSE is a widely-used statistical measure that quantifies the difference between the values predicted by a model and the actual values observed. The calculation formula for RMSE is as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \left(x_{obj,i} - x_{model,i} \right)^2}{n}} \tag{17}$$

Among them, $x_{obj,i}$ is the original value, $x_{model,i}$ is the predicted value, and n is the data length.

4.3 Analysis of experimental results

Invoke the model described in this article to obtain the predicted output and compare it with the actual values, as illustrated in the figure below. By plotting the predicted values against the actual values, we can visually assess the accuracy and reliability of the forecasting model, as illustrated in Figure 5.

Figure 5 Predictive trend chart (see online version for colours)

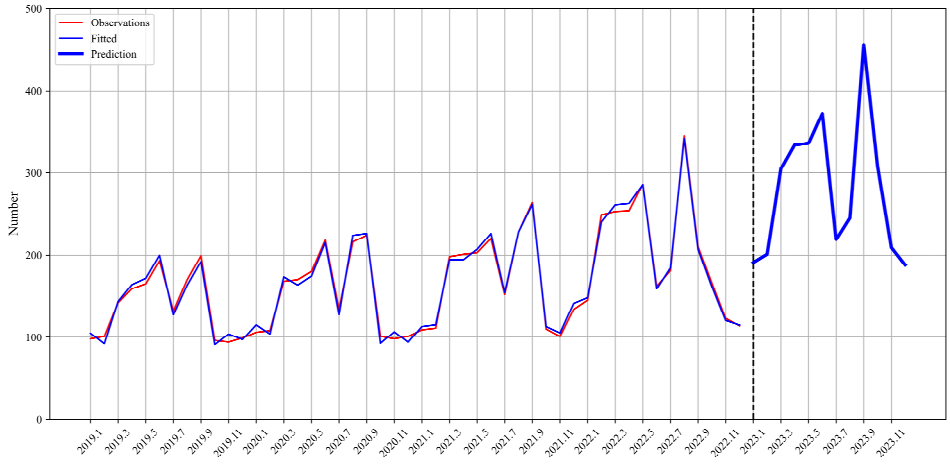


Figure 5 shows a comparison between the actual observations (red), fitted values (purple), and predicted values (blue) in a time series. The dashed line separates the fitted data (past period) from the predicted data (future period). The red line represents the observed values (actual data), while the purple line represents the model's fitted values. The high overlap between the two before the dashed line indicates that the model has fitted the past data very accurately, capturing the overall trend and fluctuations. The blue prediction curve after the dashed line reflects the model's forecast for trends beyond 2023. While the model predicts a slight decline after 2023, the overall values are still expected to remain at a high level.

5 Conclusions

This study proposes a knowledge representation model built on self attention mechanism, aiming to solve the problem of predicting talent demand and job matching. With population growth and job saturation, companies are increasingly demanding higher quality job seekers. Real time job data from recruitment platforms can provide effective data support for universities to formulate training policies. This model aggregates neighbourhood information through neural networks to generate better node representations, enabling nodes to learn their local neighbourhood information and the entire graph structure. At the same time, the self attention mechanism further extracts node features containing rich neighbourhood information, and uses deep interactions between nodes to compute the attention coefficients for the neighbours surrounding the central entity, thereby improving the accuracy of prediction. The experimental results indicate that the model demonstrates high predictive fault tolerance and can be widely applied in matching actual talent demand with university talent cultivation. This study effectively alleviates the problem of the disconnect between talent cultivation in universities and talent demand in enterprises, providing important reference for the formulation of university policies.

Acknowledgements

This work is supported by the 2023 Annual Project of Jiangxi Education Planning (No. 23YB315).

References

Chen, Z., Wang, Y., Zhao, B. et al. (2020) 'Knowledge graph completion: a review', *IEEE Access*, Vol. 8, pp.192435–192456.

Ciancetta, F., Bucci, G., Fiorucci, E. et al. (2020) 'A new convolutional neural network-based system for NILM applications', *IEEE Transactions on Instrumentation and Measurement*, Vol. 70, pp.1–12.

Dhillon, A. and Verma, G.K. (2020) 'Convolutional neural network: a review of models, methodologies and applications to object detection', *Progress in Artificial Intelligence*, Vol. 9, No. 2, pp.85–112.

Dubey, S.R., Chakraborty, S., Roy, S.K. et al. (2019) 'diffGrad: an optimization method for convolutional neural networks', *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 31, No. 11, pp.4500–4511.

Gao, C., Zhang, N., Li, Y. et al. (2022) 'Self-attention-based time-variant neural networks for multi-step time series forecasting', *Neural Computing and Applications*, Vol. 34, No. 11, pp.8737–8754.

Guo, Q., Zhuang, F., Qin, C. et al. (2020) 'A survey on knowledge graph-based recommender systems', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 34, No. 8, pp.3549–3568.

Hao, X., Ji, Z., Li, X. et al. (2021) 'Construction and application of a knowledge graph', *Remote Sensing*, Vol. 13, No. 13, p.2511.

Huang, N., Nie, F., Ni, P. et al. (2020) 'SACall: a neural network basecaller for Oxford nanopore sequencing data based on self-attention mechanism', *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 19, No. 1, pp.614–623.

Ji, S., Pan, S., Cambria, E. et al. (2021) 'A survey on knowledge graphs: representation, acquisition, and applications', *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 33, No. 2, pp.494–514.

Li, H., Wang, Q., Liu, J. et al. (2022) 'A prediction model of human resources recruitment demand based on convolutional collaborative BP neural network', *Computational Intelligence and Neuroscience*, Vol. 2022, No. 1, p.3620312.

Li, Y., Hao, Z. and Lei, H. (2016) 'Survey of convolutional neural network', *Journal of Computer Applications*, Vol. 36, No. 9, p.2508.

Ni, Q. (2022) 'Deep neural network model construction for digital human resource management with human-job matching', *Computational Intelligence and Neuroscience*, Vol. 2022, No. 1, p.1418020.

Scarselli, F., Gori, M., Tsoi, A.C. et al. (2008) 'The graph neural network model', *IEEE Transactions on Neural Networks*, Vol. 20, No. 1, pp.61–80.

Usama, M., Ahmad, B., Xiao, W. et al. (2020) 'Self-attention based recurrent convolutional neural network for disease prediction using healthcare data', *Computer Methods and Programs in Biomedicine*, Vol. 190, p.105191.

Vishnu, P., Vinod, P. and Yerima, S.Y. (2022) 'A deep learning approach for classifying vulnerability descriptions using self attention based neural network', *Journal of Network and Systems Management*, Vol. 30, No. 1, p.9.

Xu, C., Feng, J., Zhao, P. et al. (2021) 'Long-and short-term self-attention network for sequential recommendation', *Neurocomputing*, Vol. 423, pp.580–589.

Zhang, Y., Yang, B., Liu, H. et al. (2023) 'A time-aware self-attention based neural network model for sequential recommendation', *Applied Soft Computing*, Vol. 133, p.109894.