Monotonic Job Recommendation with Separated Features

Kaito Hoshi and Takanari Seito, Hashi Yuichi,
Tatsuya Izumi, Shinji Iizuka
Research and Development Dept.
Hitachi Solutions East Japan, Ltd.
Miyagi, Japan
kaito.hoshi.fy@hitachi-solutions.com, takanari.seito.cw@hitachi-solutions.com,
yuichi.hashi.wg@hitachi-solutions.com, tatsuya.izumi.hz@hitachi-solutions.com,
shinji.iizuka.zt@hitachi-solutions.com

Abstract

In recent years, job recommender systems (JRSs) have been developed to compute matching scores of workers for projects. However, their RSs do not satisfy monotonicity. Satisfying monotonicity is important because it is a matter of course for users. In this study, we propose a monotonic JRS. The conventional monotonic model guarantees monotonicity for one type of input value. However, there is no study on a monotonic model that guarantees monotonicity for two types of input data: workers and projects. Our model can guarantee monotonicity for changes in two types of input data. Our study is novel with respect to the JRS and monotonic model.

Keywords
Recommender System, Job Recommender System, Deep Learning, Monotonicity, Monotonic Neural Network

1. Introduction
The information technology (IT) industry involves various projects such as system development and infrastructure construction. Recommending an appropriate worker to a project is important for the project's success. However, it is difficult to determine the matching score (measure of whether a worker matches a project) of the worker and project. There is a significant amount of information about workers (various skills, personality, and desired working environment) and information about projects (required skills and work environment), which makes this judgment complicated.
In recent years, there have been increasing cases of developing recommender systems (RSs) that compute the matching scores of the worker and project. However, these RSs have a problem in that they do not satisfy monotonicity of the worker and project. Monotonicity of the worker and project implies that if the skills of the worker improve, the matching score increases, and if the required skills of the project improve, the matching score decreases. This is because, if the skills of the worker improve, the project gets easier. Moreover, if the required skills of the project improve, the project becomes harder. Therefore, it is natural for people that RSs satisfy monotonicity. Creating an RS that satisfies monotonicity is necessary for RS users.

In this study, we propose an RS that satisfies monotonicity. The remainder of this paper is organized as follows: In Section 2, we present some works related to our study. In Section 3, we highlight the structure of our model. In Section 4, we present the data, as well as the content and results of the experiment. Section 5 summarizes the paper and presents the future prospects of our study.

2. Literature Review
An RS is used for recommending items (movies, books, etc.) to users. Recent studies on RS have mainstreamed the use of deep learning. Using deep learning, it is known that the accuracy is higher than that of the conventional model without deep learning (Shuai et al. 2019). The deep learning model first inputs the user and item information vectorized in the preprocessing. Thereafter, it extracts user and item features separately. Finally, the model uses the user and item features to generate the matching score. The reason for extracting the features separately is that the model can add various ingenious devices. For example, one can use cross-domain to filter out features specific to a particular domain (Heishiro et al. 2019), (Shuqing et al. 2019) , use an autoencoder to learn features from a small amount of data and remove data noise (Sheng et al. 2015), look for user or item with similar features, know which elements of the worker

or project data to focus on (Sungyong et al.2017), and so on. We cannot use these tricks when extracting features by concatenating the item and user input. Using these tricks, we can improve the model accuracy and explainability.
A job RS (JRS) is an RS in which the item is a project (job) and the user is a worker. For worker information, we use a worker's skills (e.g., programming and databases) and wage. For project information, we use the required skills of a project and project risk. As with RSs, studies using deep learning have become popular in JRS as well. The content of the processing is also the same. However, conventional models have a problem in that they cannot satisfy monotonicity. The monotonicity of JRS implies that (1) the higher the skills of the worker, the higher the matching score, and (2) the higher the required skill of the project, the lower the matching score. We show an example of a monotonic forecast in Figure 1. Assume that Worker B can use Python in addition to the skills of Worker A (Java). At this time, it is expected that Worker B will have a higher matching score for projects that require Python than Worker A. In contrast, even if the project does not require Python, the matching score of Worker B will not be lower because having extra skills is not a disadvantage. It is clear that humans with higher skills or more skills are more likely to succeed in various jobs. Satisfying monotonicity is related to trust in a JRS.

Some studies ensure that the output value changes monotonically with respect to changes in the input value (Corné et al. 2021). Machine learning models that can guarantee monotonicity are being studied in fields such as business bankruptcy prediction, house evaluation, and medical diagnosis (Jose et al. 2019). It guarantees that the output value changes monotonically with respect to changes in one type of input data, such as companies, houses, and patients. The easiest way to construct a monotonic model is to constrain the weight to a positive value (Joseph 1998). Additionally, there are studies on monotonic models that approximate nonlinear monotonic functions (Seungil et al. 2017). Nonlinear monotonic models are more expressive and therefore more accurate than the model that constrains a parameter. We could not find a case of a monotonic model of JRS. There is no study that can guarantee monotonicity for two types of input data: worker and project.

In this study, we propose a monotonic JRS model. The model computes the worker and project features separately while satisfying monotonicity. The higher the skill, the bigger the feature. Finally, the model outputs a matching score from the ratio of the feature value of the worker to that of the project. The larger the feature of the worker, the larger the matching score. The larger the feature of project, the smaller the matching score. This processing ensures that the higher the worker's skills, the higher the matching score, and the higher the required skill of the project, the lower the matching score; in other words, the model satisfies JRS monotonicity. We mathematically prove that the model satisfies monotonicity. In summary, the novelty of our proposed model is as follows:

- We propose a JRS model that satisfies monotonicity.
- We construct a model that can guarantee monotonicity for changes in two types of input data.

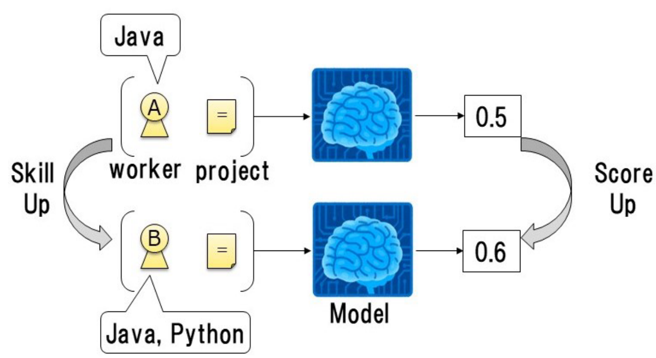

Figure 1. Mon**otonic forecast example.**

3. Proposed Model

In this section, we explain the method proposed in this study. In subsection 3.1, we describe (or define) a monotonic JRS function. In subsection 3.2, we describe the overall model and processing. In subsection 3.3, we prove that the model guarantees monotonicity.

3.1 Preliminary

Let $x_w \in \mathbb{R}^{n_w}$ and $x_p \in \mathbb{R}^{n_p}$ be input vectors of the worker and project, respectively, where n_w and n_p are dimensions of x_w and x_p respectively. The input vector of the worker is a vectorized version of the worker information (programming skills such as Java and Python, and wage). The input vector of the project is a vectorized version of the project information (required skills and risk). Each element of information is quantified as 1 if applicable and 0 if not applicable. An example of the input vector of the worker is listed in Table 1 (input vector of the project is omitted, but has a similar shape). Worker 1 has Python and C# skills. Worker 1 does not have Java skill. Let X, Y be any vector; then, $X \geq_{vec} Y$ implies that the value of each element of X is greater than that of each element of Y. $X \geq_{vec} 0$ implies that the value of each element of X is 0 or greater.

In JRS, monotonicity is guaranteed if the following two conditions hold:

i. If the skills of the worker improve, the project will be easy. Therefore, the matching score will increase.
ii. If the required skills of the project improve, the project will be difficult. Therefore, the matching score will decrease.

These conditions can be expressed mathematically as follows:

Definition. *Let f be a monotonic function of a JRS model, and let $a_w \in \mathbb{R}^{n_w}$ $(a_w \geq_{vec} 0)$ and $a_p \in \mathbb{R}^{n_p}$ $(a_p \geq_{vec} 0)$; then, f must satisfy the following:*

$$f(x_w + a_w, x_p) \geq f(x_w, x_p), \qquad (1)$$

$$f(x_w, x_p + a_p) \leq f(x_w, x_p), \qquad (2)$$

where $f(\cdot, \cdot) \in \mathbb{R}$ is the output (matching score) of the JRS model.

Details of function *f* are provided in subsection 3.2 and terms (3)–(5). The left side of (1) denotes the matching score when the skills of the worker improve. The left side of (2) denotes the matching score when the required skills of the project improve. By defining (1) and (2), we can demonstrate that as the skills of the worker improve, the matching score increases, and as required skills of the project increase, and the matching score decreases.

Table 1. Example of input vector of partner

	Python	Java	C#	…
Worker 1	1	0	1	…
Worker 2	0	1	1	…

3.2 Description of Model

We propose a model of a JRS using monotonic neural network (MNN) methods. Figure 2 shows our model, which comprises two parts: feature extraction and matching score function. The input of the model is vectors of worker and project information. The output of the model is a numerical value between 0 and 1, where 0 implies poor fit and 1 implies good fit. We can obtain the matching score of the worker and project by two processes.

For the first process, we input the vectors of the worker and project information, followed by feature extraction output feature vectors of the worker and project, respectively. The feature extraction uses an MNN, which is the feature extraction of the conventional monotonic model (Joseph 1998) , (Seungil et al. 2017). When using an MNN, the larger the skill value, the larger the feature. Any MNN model can be used as long as it is mathematically guaranteed to be monotonic. Notably, feature extraction is a separate MNN, which computes worker and project features separately. The conventional MNN model cannot output monotonic features for two inputs. The two features are output in the same dimension.

For the second process, the model inputs feature vectors of the worker and project into the matching score function, and thereafter outputs a scalar that is matching the score of worker and project. The model normalizes the features of the worker and project using the sigmoid function. Thereafter, using the sufficiency function, we compute the rate (sufficiency rate) of the worker feature value to that of the project. Sufficiency rate is a numerical value that indicates how well the skills of the worker meet the project requirements. The minimum value is 0 and the maximum value is 1. The greater the feature of the worker, the greater the degree of satisfaction, and the greater feature of the project, the smaller the degree of satisfaction. Finally, the model takes the product of all sufficiency rates and outputs the matching score. Here, in the conventional model of the JRS, there are two ways to compute matching scores. The first method (Chen et al. 2018) outputs the cosine similarity between worker and project features.

Using this method, monotonicity cannot be guaranteed because even if the input value or feature value increases, the positions of the two features do not become close. The second method (Shuqing et al. 2020), (Chuan et al. 2018) involves concatenating two feature vectors, multiplying the concatenated vectors by weights, and obtaining a matching score with a sigmoid function. Because the sigmoid function is a monotonic function, monotonicity can be guaranteed even in the conventional model by restricting the weights to positive values. Unlike traditional methods, our method allows users to intuitively determine whether a worker is fit for a project using the sufficiency rate. Figure 3 shows a radar chart with the values of the normalized worker and project features. It implies that the further to the right, the higher the value of that feature. By observing the radar chart, the user can intuitively determine the requirements (feature elements) of the project that the worker meets. For instance, in Figure 3, it is observed that the worker satisfies features 1 and 3, but does not satisfy feature 2. In contrast, because the value of feature 2 for the project is small, we can assume that B has less influence on the project. Therefore, even if the worker does not satisfy feature 2, it may match the project. In this way, several types of support can be provided to users.

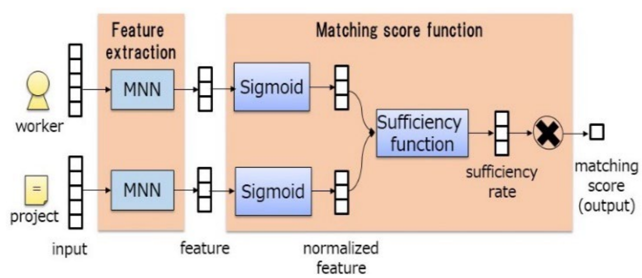

Figure 2. Structure of proposed model.

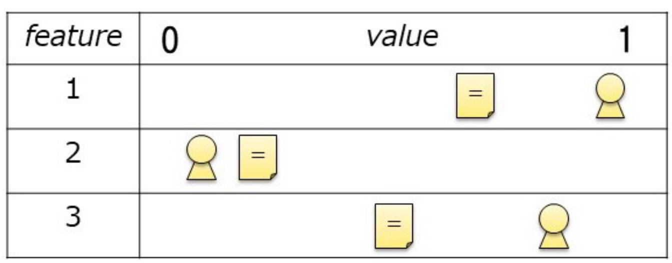

Figure 3. Radar chart of normalized feature matching score.

The computation process of our model is described as follows:

$$y_i = MNN_i(x_i)\ (i \in \{w, p\}), \qquad (3)$$

$$\tilde{z}[k] = min\left(\frac{\sigma(y_w[k]) + \varepsilon}{\sigma(y_p[k]) + \varepsilon}, 1\right), \qquad (4)$$

$$z = \prod_{j=1}^{m} \tilde{z}[k], \qquad (5)$$

where (3) denotes the term of feature extraction of the worker or project, (4) denotes the term of composite function for the sigmoid and sufficiency functions, and (5) denotes the term product of all sufficiency rates; $y_i \in \mathbb{R}^m$ denotes the feature of input vector x_i, m denotes the number of dimension of the feature vector, MNN_i denotes a feature extraction of any monotonic model (e.g., (Joseph 1998), (Seungil et al. 2017)), which can be written as a map $MNN_i : \mathbb{R}^{n_i} \rightarrow$

$\mathbb{R}^m$ defined by (3), σ denotes a sigmoid function, $y_i[k] \in \mathbb{R}$ denotes the k-th value of the vector, $\tilde{z}[k] \in [0, 1]$ denotes the sufficiency rate of the k-th vectors for the worker and project, $k \in \{1, .., m\}$, $\tilde{z}$ denotes a vector of sufficiency, ε denotes a constant number to prevent division by zero, $\min$ outputs minimum value in inputs, and $z \in [0, 1]$ denotes a matching score.

In summary, our model comprises feature extraction and matching score function. Feature extraction is the same as the feature extraction part of the conventional monotonic model (Joseph 1998), (Seungil et al. 2017). The matching score function computes the sufficiency rate that value of the skills of the worker meets the project requirements. We can obtain a radar chart from the sufficiency rate. Using the radar chart, the user can easily observe the skill level of the worker and the skill level required by the project. Our model can support user judgment.

Note that not all input elements need to satisfy monotonicity. For instance, there are cases when the wage of the worker should be cheaper owing to budget constraints, and there are cases when the wage should be higher owing to the use of a talented worker. Therefore, the wage of the worker does not need to be higher or lower. An MNN can be set such that it does not satisfy monotonicity for non-monotonic elements. The proposed model can also set the same.

3.3 Proof of Monotonicity

In this subsection, we will prove that our model is monotonic. Specifically, we will prove that future extraction and matching score function each satisfy monotonicity, thereby proving that our model is monotonic. In the conventional monotonic model (Joseph 1998), (Seungil et al. 2017), the feature extraction part and the output function each comprises monotone functions. Thus, in the feature extraction of our model, the larger the input value, the larger the feature value. We only prove that the matching score function satisfies monotonicity.

Lemma. Let $h(y_w, y_p)$ be a composite function of *(4)* and *(5). Then, for any* $b \in \mathbb{R}^m (b \geq_{vec} 0)$, h *satisfies the following:*

$$(i) \qquad h(y_w + b, y_p) \geq h(y_w, y_p)$$

$$(ii) \qquad h(y_w, y_p + b) \leq h(y_w, y_p)$$

[proof] : We prove only (i) ((ii) can be proved in the same way as (i)).

We will prove that $h(y_w + b, y_p) - h(y_w, y_p) \geq 0$.

$y_w + b - y_w \geq_{vec} 0$

$\Leftrightarrow \ \sigma((y_w + b)[k]) - \sigma(y_w[k]) \geq 0$ for $\forall k \in \{1, \ldots, m\}$($\because \sigma$ is monotonic function)

$\Leftrightarrow (\sigma((y_w + b)[k]) + \varepsilon) - (\sigma(y_w[k]) + \varepsilon) \geq 0(\because \varepsilon > 0)$

$$\Leftrightarrow \frac{\sigma(y_w[k] + b) + \varepsilon}{\sigma(y_p[k]) + \varepsilon} - \frac{\sigma(y_w[k]) + \varepsilon}{\sigma(y_p[k]) + \varepsilon} \geq 0(\because \sigma(y_p[k]) + \varepsilon > 0)$$

$$\Leftrightarrow min\left(\frac{\sigma(y_w[k] + b) + \varepsilon}{\sigma(y_p[k]) + \varepsilon}, 1\right) - min\left(\frac{\sigma(y_w[k]) + \varepsilon}{\sigma(y_p[k]) + \varepsilon}, 1\right) \geq 0(\because \text{min is monotonic function})$$

$$\Leftrightarrow \prod_{j=1}^{m} min\left(\frac{\sigma(y_w[k] + b) + \varepsilon}{\sigma(y_p[k]) + \varepsilon}, 1\right) - \prod_{j=1}^{m} min\left(\frac{\sigma(y_w[k] + b) + \varepsilon}{\sigma(y_p[k]) + \varepsilon}, 1\right) (\because \text{inputs of min are positive})$$

Therefore, $h(y_w + b, y_p) \geq h(y_w, y_p)$.

■

In summary, the following hold:

1. The MNN used in the proposed model satisfies monotonicity.
2. As proved in the lemma, the output of the proposed model satisfies monotonicity.

From these points, we can prove the following main theorem:

Theorem. The proposed model is monotonic, that is, the model satisfies conditions (1) and (2) of the Definition.

4. Experiments

We examined the performance of our proposed method in the experiments on matching data of our company datasets.

4.1 Dataset Description

We used our company dataset. This dataset includes three types of data: matching, worker, and project. Matching data are previous matching results of the project and worker. We use matching data as training and test data. Worker data include information (programming and wage) about workers. Project data include information (required programming skills and risks) about projects. We read each item of the tabular data and preprocessed them with a one-hot vector. The number of data is 1,212 for worker, 1,673 for project, and 5,692 for matching (label 1: 5,033, label 0: 659). Table 2 shows the monotonicity of the input elements of the worker and project. Table 3 lists the details of matching data. We randomly selected 4,937 (approximately 85% of the total) data as training data from matching data (label 1: 4,481 and label 0: 456). We used the remaining matching data as test data.

Table 2. Data statistics

Dataset	Number of data	Monotonic	Non-monotonic	Total elements
Worker	1,212	96	2	98
Project	1,673	115	41	156
Matching	5,033			

Table 3. Breakdown of matching data

	Label 1	Label 0
Training	4,481	456
Test	552	203

4.2 Experimental Conditions

We selected the MNN in (3) to be a linear function with positive weight restrictions using a rectified linear unit (ReLU) function. The model learns to minimize the loss function using training data. The loss function is defined as the sum of binary cross entropy with the weight set to inverse class weight and L1-loss with weight of L1-loss. The model uses the Adam optimizer to optimize the parameters. Table 4 indicates the hyperparameters.

Table 4. Condition of hyperparameter

Hyperparameter	Condition
Dimension of feature	4
Epoch	20,000
Weight of L1-loss	10^{-6}
Optimizer	Adam

4.2 Evaluation Metric

We evaluated the possibility of correctly predicting the presence or absence of the matching score through an analysis. Because label data are unbalanced, we used balanced accuracy (BA) as the evaluation metric, which is as follows:

$$BA = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right)$$

where TP (true positive) denotes the number that the model predicted as 1 for data with label 1, FN (false negative) denotes the number that the model predicted as 1 for data with label 0, TN (true negative) denotes the number that the model predicted as 0 for data with label 0, and FP (false positive) denotes the number that the model predicted as 1 for data with label 0. Inside the parentheses to the right of the BA equation, the first expression implies the accuracy of the positive example, whereas the second expression implies the accuracy of the negative example. Thus, BA is the average accuracy of the positive and negative examples.

To understand the accuracy of the proposed model's recommendation, we compare the accuracy when predicting all 1s and the accuracy of the cut-off model. Here, the cut-off model comprises feature extraction without weight constraints and an evaluation function that outputs the Euclidean distance between two features as a sigmoid function. We compare the precision with a t-test with a 5% significance level. In general, we also need to compare the accuracy of the model to the accuracy of predicting all 0s. The accuracy of BA is 50% when the model predicts all 0s. The accuracy of BA when the model predicts all 1s is also 50%; thus, we ignore that the model predicts all 0s. To verify whether there is a significant difference in accuracy, the training data were divided into 10 parts, and with the data excluding one part, the training was performed 10 times while changing the division.

4.2 Experimental Results

The experimental results are summarized in TABLE V. The BA column in TABLE V lists the mean and standard deviation (in parentheses) of the BA for 10 assessments. Column P-value lists the P-values obtained from the t-test with a 5% significance level with the proposed model.

The P-value of proposed model and cut-off model is 1.38*10^{-4}, which is less than 0.05. Moreover, the average value of the BA is higher in the cut off model. Therefore, it is observed that the cut-off model is more accurate.

The P-value of the proposed model that can predict all 1s is 5.29*10^{-9}, which is less than 0.05. Moreover, the average value of the BA is higher in the proposed model. Therefore, it is observed that the proposed model is more accurate.

From these results, it is observed that the proposed model is less accurate than the cut-off model, but it can guarantee the minimum accuracy. In contrast, the accuracy is higher than when all 1s or 0s are predicted; thus, it is observed that it has the minimum accuracy.

Table 5. Evaluation results

	BA	P-value
Proposed model	0.824(0.045)	
Cut-off model	0.942(0.024)	1.38*10^{-4}
Predict all 1s	0.500(0)	5.29*10^{-9}

5. Conclusions

In this study, we proposed a JRS model that guarantees monotonicity. Our model makes predictions satisfying monotonicity for changes in two inputs, the worker and project. In the conventional monotonic model, it was not possible to maintain monotonicity against changes in two input values. By constraining the weights to be positive, we designed a feature extraction such that the higher the skill, the greater the feature. The matching score function was designed to determine the suitability by comparing the value of the two features. We mathematically proved that the proposed model satisfies monotonicity.
As a result of the analysis, we confirmed that the prediction accuracy of the model is higher than the prediction of all 1s. Thus, the minimum degree of the accuracy is assumed to be guaranteed. In contrast, we confirmed that the proposed model is less accurate than the cut-off model. A future work will involve eliminating the difference in prediction accuracy between with and without monotonicity.

Specifically, we envision that proposed model intends to increase the expressive power of feature extraction. Feature extraction of the model used in the experiment comprises a linear layer and an active function. We observe that the configuration is extremely simple. Previous studies on monotonic models involved complex models, such as the deep lattice network (DLN) (Seungil et al. 2017). These models are more expressive; thus, their accuracy is also higher. Replacing the construction of feature extraction from this construction to the DLN (or other complex monotonic model) will improve accuracy.

References

Shuai, Z. Lina Y., Aixin S., and Yi T., Deep Learning based Recommender System: A Survey and New Perspectives, *ACM Computing Surveys (CSUR)*, vol. 52(1), pp. 1–38, 2019.

Corné, D, R., and Sandjai, B., Job Recommender Systems: A Review*, arXiv preprint* arXiv:2111.13576, 2021.

Joseph, Sill., Monotonic Networks, *Advances in neural information processing systems*. pp. 661–667, 1998.

Seungil, Y., David, D., Kevin, C., Jan, P., and Maya, R, G., Deep Lattice Networks and Partial Monotonic Functions, *arXiv e-prints*, *Article arXiv:1709.06680*, 2017.

Jose, C., Pedro, A, G., Bartosz, K., Michal, W., and Salvador G., Monotonic classification: an overview on algorithms, performance measures and data sets, *Neurocomputing*, Vol. 341, pp. 168–182, 2019.

Heishiro, K., Hayato, K., Nobuyuki, S., Yukihiro, T., and Taiji, S., Cross-domain Recommendation via Deep Domain Adaptation, *European Conference on Information Retrieval*, pp. 20–29, 2019.

Shuqing, B., Wayne, X, Z., Yang, S., Tao, Z., and Ji, R, W., Domain adaptation for person-job fit with transferable deep global match network., *in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4810–4820, 2019.

Sheng, L., Jaya, K., and Yun, F, Deep collaborative filtering via marginalized denoising auto-encoder, *in CIKM*, pp. 811–820, 2015.

Chen, Z., Hengshu, Z., Hui, X., Chao, M., Fang, X., Pengliang, D., Pan, LI., Person-job fit: Adapting the right talent for the right job with joint representation learning., *ACM Transactions on Management Information Systems (TMIS)*, vol. 9(3), pp. 1–17, 2018.

Shuqing, B., Xu, C., Wayne, X, Z., Kun Z., Yupeng, H., Yang, S., Tao, Z. and Ji, R, W., Learning to match jobs with resumes from sparse interaction data using multi-view co-teaching network., *in Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 65–74, 2020.

Chuan, Q., Hengshu, Z., Tong, X., Chen, Z., Liang, J., Enhong, Chen., and Hui, X., Enhancing person-job fit for talent recruitment: An ability-aware neural network approach, *in The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 25–34, 2018.

Sungyong, S., Jing, H., Hao, Y., and Yan, L., Interpretable convolutional neural networks with dual local and global attention for review rating prediction, *in Proceedings of the 11th ACM Conference on Recommender Systems. ACM*, pp 297–305, 2017.

Biographies

Kaito Hoshi is a researcher in the Research and Development Dept. at Hitachi Solutions East Japan, Ltd. He obtained Master of Science from Akita University, Japan. His research interests include deep learning.

Takanari Seito is a researcher in the Research and Development Department at Hitachi Solutions East Japan, Ltd. He obtained Master of Information Science from Tohoku University, Japan. His research interests include mathematical optimization and data analysis.

Yuichi Hashi is a researcher in the Research and Development Department at Hitachi Solutions East Japan, Ltd. He obtained Doctor of Engineering and Master of Science from Tohoku University. His research interests include data analysis and numerical simulation.

Tatsuya Izumi is a manager in the Research and Development Department at Hitachi Solutions East Japan, Ltd. He obtained Bachelor of Science from Tohoku University. His research interests include project management and application of AI in business.

Shinji Iizuka is an expert (manager equivalent) in the Research and Development Department at Hitachi Solutions East Japan, Ltd. He obtained Ph.D. from Tohoku University. His research interests include data analysis, artificial intelligence, and quantum computing.