Disruptive Human Resource Management
S. Patnaik (Ed.)

doi:10.3233/ATDE240418

A BP Algorithm for Human Resource Management Turnover and Salary Forecasting

Chun Hu[1] and Chuanjian Wu
Chengdu Textile College, Chengdu 611731, China

Abstract. Human resource management is a comprehensive process involving recruiting, training, evaluating and motivating employees to achieve organizational goals. In today's society, human resource management forecasts have low accuracy and high error rates. This article uses the BP algorithm to build a turnover and salary prediction model to predict human resource management. Finally, through experiments, it was concluded that the algorithm greatly reduced the error rate of human resource management predictions and improved its accuracy, with an accuracy rate as high as 98.9%. In the future, BP algorithm and human resource management prediction should be widely used.

Keywords. BP algorithm, BP neural network, Human Resource Management, forecasting, turnover and salary forecasting models

1. Introduction

Performance management in human resources, often referred to as performance appraisal or employee performance evaluation, is a systematic process designed to assess and enhance an employee's job performance. This concept originated in the 20th century when businesses began to recognize the tight linkage between employee performance and organizational success. Over time, it became evident that mere employee evaluations were insufficient for driving corporate success, and continuous development and growth of employees became paramount. As a result, performance management shifted towards identifying employee potential, offering consistent training, and providing feedback. In today's business environment, with heightened competition for talent, performance management has become a cornerstone of HR strategy, ensuring alignment between employee and company objectives.

The Backpropagation Neural Network [1-3], commonly referred to as the BP neural network, stands as a pivotal milestone in the domains of machine learning and artificial intelligence. This multi-layer feedforward neural network employs the

[1] Corresponding Author: Chun Hu, 2979978711 @qq.com

Backpropagation algorithm (often shortened to BP algorithm) for its learning and training processes, making it one of the few neural network models to achieve tangible success across diverse applications. McCulloch and Pitts first introduced a simplified neural network model [4] grounded in logic, laying the groundwork for subsequent neural network research. However, the true potential of neural networks wasn't fully unearthed until 1986. In a groundbreaking paper, Rumelhart, Hinton, and Williams detailed the Backpropagation algorithm. They demonstrated how this algorithm could effectively adjust weights in a multi-layered network, thereby minimizing the disparity between network outputs and true target values.

At the core of the BP algorithm [5] is the adoption of the gradient descent technique to fine-tune the neural network's weights and biases. In essence, the algorithm initiates with a forward propagation to measure the variance between the network's result and the true target. This divergence is subsequently relayed back through the network's tiers, from the output stage to the input stage, making provisions for tweaks to weights and biases in line with the derived gradients. From this juncture forward., the BP neural network [6-7] and the Backpropagation algorithm gradually became the standard in both neural network research and practical applications. With the advent of enhanced computational capabilities and the availability of vast datasets, this methodology further propelled the rapid advancement of the deep learning arena. Consequently, it laid the foundation for many breakthrough developments in modern AI applications.

The Backpropagation Neural Network (BPNN) stands as an epitome of computational intelligence in the annals of machine learning. Delving deeper into its technical roots, BPNNs are underpinned by the Backpropagation (BP) algorithm, a meticulously designed procedure to train multi-layer feedforward neural networks. Starting with the forward propagation phase, input data traverses the layers of the network, each layer acting as a transformative conduit applying its weights and activation functions (like ReLU, Sigmoid, or Tanh). Upon reaching the output layer, the network produces a provisional prediction [8-9]. The disparity between this prediction and the true target values is quantified by a chosen loss function, be it mean squared error, cross-entropy, or another suitable metric, representing the network's current inaccuracy. The magic unfurls during the backward propagation phase. This is when the BP algorithm [10] starts its meticulous task. The computed error is reverse propagated through the network, and during this retrogression, partial derivatives of the loss concerning each weight are determined. This gradient information serves as a compass, indicating the direction and magnitude for weight adjustments. With this roadmap in hand, optimization strategies, such as the foundational gradient descent or its evolved kin like Adam, RMSProp, or AdaGrad, undertake the task of iteratively refining these weights, inching the network closer to the desired accuracy. However, the landscape of prediction is vast and multi-faceted. Beyond the BPNN’s architectural nuances, the sanctity and robustness of predictions often hinge upon several auxiliary processes. Rigorous data preprocessing, including data cleansing, outlier detection, and handling missing values, ensures the purity of input data. Feature engineering, often termed an art in the ML community, distills raw data into meaningful and predictive features. Techniques like data normalization, standardization, and batch normalization ensure homogeneity and stability across datasets. To counteract overfitting, model regularization approaches like dropout, L1, and L2 regularization are indispensable, bestowing the model with a refined balance between specificity and generalizability. The labyrinth of hyperparameter tuning further refines model behavior, using strategies

like grid search, random search, or more advanced Bayesian optimization to discover the golden configuration for optimal performance. Woven together, the sophisticated mechanics of BP neural networks, the algorithmic finesse of the BP algorithm, combined with the nuanced art and science of predictive techniques, provide a comprehensive arsenal for modern AI endeavors [11]. This intricate tapestry of methodologies and techniques not only facilitates accurate predictions but also catalyzes groundbreaking advancements across diverse AI-driven domains.

In this paper, BP algorithm [12] is used to build the salary and turnover prediction model, and the prediction experiment is carried out. The prediction system constructed by this algorithm greatly reduces the prediction error and improves the accuracy and efficiency of the prediction

2. Methodology

Within the BP neural network framework, initially, neurons accept input signals relayed from other neurons, paired with pertinent input connection weights. Consequently, neurons can compute the aggregate input signal using both the input signal and its related weights, subsequently contrasting this combined input value against the neuron's inherent reading. After undergoing the "activation function" procedure, the neuron's final output is obtained. Refer to Fig. 1.

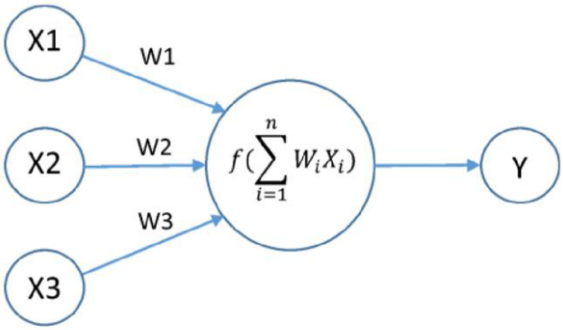

Figure 1. M-P neuron model

A neural network consists of numerous neurons linked in a specific hierarchical order, where neurons possessing activation functions are termed functional neurons or M-P neurons. Based on the inter-neuron hierarchy, neural networks can be classified into two types: layered network models and interconnected network models. In layered models, neurons are stratified, typically segmented based on their roles into layers such as the input, intermediate, and output layers, with each layer interconnected. On the other hand, in interconnected models, any two neurons can connect with a degree of arbitrariness. Due to its organized form and ease of analysis, the layered network is the predominant model structure. In this framework, a two-layered neural network is named a perceptron, where only the output layer neurons undergo activation function processes, meaning the output layer comprises M-P neurons. The multi-layered network expands on the perceptron by introducing one or more hidden layers between the input and output layers, with all neurons in these hidden layers being functional neurons with activation functions. Refer to Fig. 2.

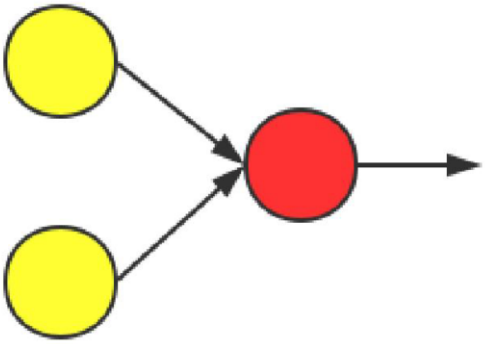

Figure 2. Perceptron

The multilayer feedforward neural network is a distinct type of multilayered network where each neuron layer fully interlinks with the subsequent layer. This implies every node in one layer connects with all nodes in the following layer, with no intra-layer or cross-layer connections.

In the context of feedforward networks, the absence of loops or cycles pertains to the network topology, not implying that signals can't be sent backward. Horik and colleagues demonstrated that a multilayer feedforward neural network, with just a single hidden layer containing sufficient neurons, can approximate any complex continuous function with utmost precision. The network depicted in Fig. 3 represents a multilayer feedforward neural network. While neural network topologies can differ, their foundational learning tenets remain consistent. The learning procedure of the neural network involves utilizing training data to modify the weights of neuronal input connections and the base values of neurons, essentially adjusting parameters of both hidden and output layer neurons.

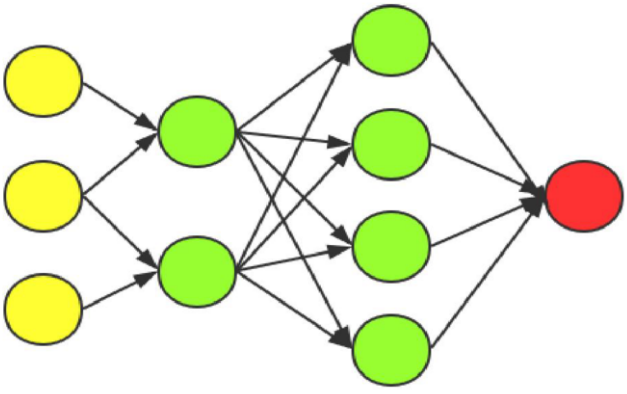

Figure 3. Multilayer network

Using the three-layer BP neural network as an illustration, the computational concepts of forward and backward propagation are detailed. Let's assume the output

from the input layer neuron of this three-layer BP network is, the output from the hidden layer neuron is, the value of the hidden layer node is, and the output layer node value is. The connection weight from the input layer node to the hidden layer node is denoted as w; the weight from the hidden layer node to the output layer node is represented by vik. Given that the input layer node undergoes no processing, the output from the JTH node of the hidden layer is expressed as. This is depicted in Formula 1 below:

$$l_j = f(\sum_i (w_{ij}x_i - \alpha_j)) = f(net_j) \tag{1}$$

The result from the KTH node in the output layer is denoted as. Refer to Formula 2 below:

$$Z_j = f(\sum_i (V_{ij}xL_i - B_j)) = f(net_j) \tag{2}$$

Where Z represents the computed outcome using the forward propagation technique. Given discrepancies between the predicted Z and the true output layer value, a mechanism to quantify this error gap is essential. Typically, the error magnitude is gauged using the cost function (or loss function) E. Let's denote the network's cost function (or loss function) as. See Formula 3 below:

$$\mathrm{E} = \mathrm{C}(\boldsymbol{\theta}; \boldsymbol{z}) \tag{3}$$

Where @=0(w; a; v; B) stands for all parameters and z=(z; z;; zk; ..) represents the output vector. The essence of the BP algorithm lies in diminishing the value of E through consistent iterative refinement. Typically, layer parameters are modified using the gradient descent technique to continuously decrease E until its lowest point is achieved. The computation procedure is outlined below: 1) Determine the error gradient value for nodes in the output layer. Relevantly, and met relates exclusively with z, hence the relation. Reference Formula 4 below:

$$\frac{\partial E}{\partial v_{jk}} = \frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial v_{jk}} = \frac{\partial E}{\partial z_k}\frac{\partial z_k}{\partial net_k}\frac{\partial net_k}{\partial v_{jk}} \tag{4}$$

Formula 5 can be obtained from formula 2 and 3:

$$\begin{aligned} obj^{(t)} &= \sum_{j=1}^{t} [G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2] + \gamma T \\ &= -\frac{1}{2}\sum_{j=1}^{T} \frac{G_j^2}{H_j+\lambda} + \gamma T \end{aligned} \tag{5}$$

Given that the network's learning rate is n, this rate signifies the step size during the iterative phase of the algorithm, having a direct influence on the speed of convergence. Different layers of the network can adopt varying learning rates for enhanced tuning. The BP algorithm shifts parameters against the gradient direction of

the objective. If we presume a consistent learning rate across all layers, then equations 6 and 7 are provided:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \eta \gamma_j x_i \tag{6}$$

$$\Delta \alpha_j = -\eta \frac{\partial E}{\partial \theta_j} = -\eta \gamma_j \tag{7}$$

Likewise, the gradient minimum for nodes within the hidden layer can be derived. This is depicted in equations 8 and 9:

$$\Delta v_{jk} = -\eta \frac{\partial E}{\partial v_{jk}} = \eta \delta_k l_j \tag{8}$$

$$\Delta \beta_k = -\eta \frac{\partial E}{\partial \lambda_k} = -\eta \delta_k \tag{9}$$

3. Experiment

From the preceding discussion, it's evident that some attributes of the original dataset for salary and turnover predictions are non-numeric, primarily character-based. Hence, there's a need for numeric conversion preprocessing to turn these non-numeric values into numeric ones. Depending on the feature, conversion methods vary. Some attributes undergo direct discretization, while others necessitate ordinal mapping. Take gender as an instance: post-discretization, it yields two outcomes, allowing a straightforward conversion where male is 0 and female is 1. Conversely, for education institutions, post-dispersion, they are ranked on a scale from 1 to 4. A mapping table is employed for ordinal mapping, categorizing institutions accordingly. For instance, top-tier institutions like Tsinghua University might be mapped to 4, universities like Huazhong University of Science and Technology to 3, institutions such as Guangxi University to 2, and other universities to 1, thus classifying all into these four tiers.

In the salary prediction framework, given that the final prediction doesn't fall within a normalized range, it's essential to revert the normalized data back to its original spectrum upon model conclusion. To bolster model adaptability and ease future data incorporations, xmax and min consider feature value boundaries, not the dataset's extreme feature values. The primary goal of normalization is dimension scaling for diverse features, making different metric features comparable while preserving the initial data distribution. Zero-mean normalization, or standard deviation normalization, which centers raw data around 0, is the prevalent method.

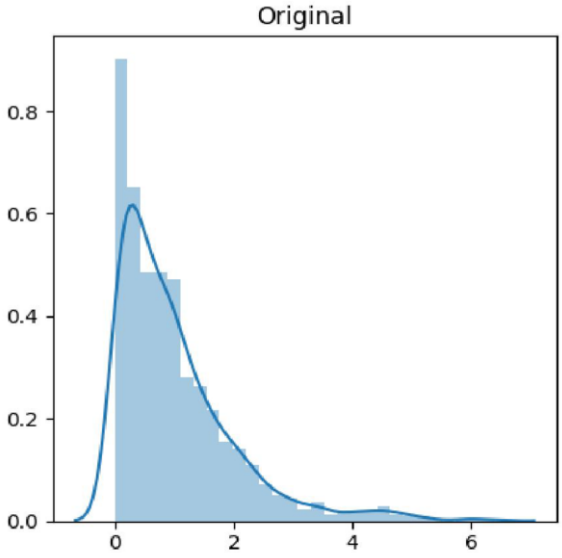

Figure 4. Raw data distribution

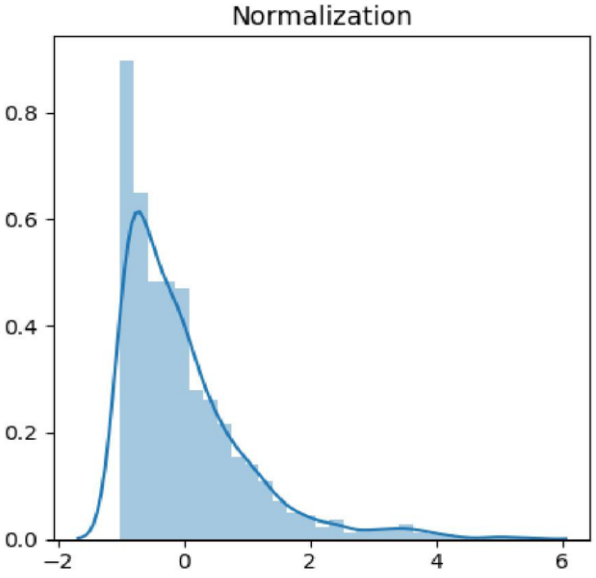

Figure 5. z-score standardized results

To demonstrate the data spread after applying z-score normalization, we present an example detailing the actual processing impact, as depicted in Fig.4 and 5. Observably, z-score normalization centers the original data distribution around 0, enhancing the contrast of diverse data attributes, yet without altering the inherent data distribution.

Relying on the empirical formula, the neuron count range in the hidden layer is approximated to be within [5,1]. This range is then marginally broadened to [3,16] during the training phase. Upon iterative training using the BP neural network, we derive TABLE Ⅰ. Notably, TABLE Ⅰ comprises 1000 sample entries, with the training dataset holding 500 samples, and the validation dataset another 500. For each hidden layer, 30 training iterations are conducted on neuron counts, and the mean of these 30 outcomes is ultimately considered. Training concludes either upon reaching 1000 generations or when the loss in training accuracy, denoted as E, drops below 0.01.

Table 1. Comparison of training of different number of neurons in hidden layer

Number of neurons	Number of iterations	Training loss	Proof loss
3	218	0.009975	0.010725
4	173	0.009932	0.010724
5	130	0.009966	0.010653
6	138	0.009945	0.010754
7	137	0.009911	0.010519
8	120	0.009935	0.010806
9	92	0.009958	0.010515
10	94	0.009948	0.010510

Typically, the count of neurons in the hidden layer that exhibit superior performance on the validation set is chosen. As per the results in TABLE Ⅰ, with 7 neurons in the hidden layer, there's an accelerated training pace coupled with minimal verification loss. Consequently, for the three-layer BP neural network employed in the salary prediction model, the neuron counts for the input, hidden, and output layers are (10, 9, 1), in that order.

4. Discussion

This article delves into the application of the BP neural network in salary prediction. Initially, the choice of the salary forecasting model is presented. Subsequently, the creation of the salary prediction model, rooted in the standard BP neural network [13], is elaborated, encompassing aspects like the neural network's topology, activation function choice, and parameter initialization. Addressing the model's limitations, optimization techniques like mini-batch gradient descent and the hybrid Nadam optimization algorithm are introduced. The subsequent sections shed light on experimental testing of the salary prediction model, contrasting its performance with other machine learning regression models. The manuscript also highlights the requirements analysis for the human resource management system and the necessary preprocessing of experimental data. It underscores the needs analysis for salary forecasting, extracting information from resume files, and data feature selection. The demands for turnover forecasting and associated data feature selection are also detailed, culminating in a discussion on data engineering techniques, encompassing data type conversions, standardization, and normalization. The central aim of this chapter is to furnish pertinent data for subsequent experiments, enhancing the precision of experimental outcomes.

5. Conclusion

The salary and turnover prediction models in this study are built upon the BP neural network. Utilizing the BP algorithm, the three-layer BP neural network presented in this paper possesses the capability to approximate functions with arbitrary precision. Owing to its commendable performance in data regression prediction, it's frequently chosen for crafting data regression predictive models. This study specifically constructs a salary and turnover predictive framework using the BP algorithm and leverages mini-batch gradient descent along with the hybrid Nadam optimization algorithm for enhancements. Experimental simulations reveal that the refined salary and turnover prediction model exhibits stellar outcomes in both learning phases and predictive accuracy. The robustness of the salary and turnover prediction model is validated, achieving a notable accuracy rate of 98.9%. Anticipating the trajectory ahead, it's projected that the BP algorithm will garner substantial adoption in future human resource management forecasting endeavors.

References

[1] Kumar KV, Ravi V, Carr M, Kiran NR. Software development cost estimation using wavelet neural networks. Journal of Systems and Software. 2008; 81(11), p. 1853-1867.
[2] Lumini A, Nanni L. Convolutional neural networks for ATC classification. Current pharmaceutical design. 2018; 24(34), p. 4007-4012.
[3] Göller AH, Kuhnke L, Montanari F, Bonin A, Schneckener S, Ter Laak A, Wichard J, Lobell M, Hillisch A. Bayer's in silico ADMET platform: a journey of machine learning over the past two decades. Drug Discovery Today. 2020; 25(9), p. 1702-1709.
[4] Chen J, Wang Z, Pollastri G. A neural network approach to ordinal regression. A neural network approach to ordinal regression. IEEE edn; 2021. 1279-1284 p.
[5] Ortiz-Rodríguez J, Alfaro AR, Haro AR, Viramontes JC, Vega-Carrillo H. A neutron spectrum unfolding computer code based on artificial neural networks. Radiation physics and chemistry. 2022; 95, p. 428-431.
[6] Kanika S. Chakraverty, Chakraborty P. Tools and techniques for teaching computer programming: A review. Journal of Educational Technology Systems. 2020; 49(2), p. 170-198.
[7] Tajik, AJ. Machine learning for echocardiographic imaging: embarking on another incredible journey. Machine learning for echocardiographic imaging: embarking on another incredible journey. DC: American College of Cardiology Foundation Washington, edn; 2021. 2296-2298 p.
[8] Roccetti M, Delnevo G, Casini L, Cappiello G. Is bigger always better? A controversial journey to the center of machine learning design, with uses and misuses of big data for predicting water meter failures. Journal of Big Data. 2022; 6(1), p. 1-23.
[9] Tajik AJ. Machine learning for echocardiographic imaging: embarking on another incredible journey. Machine learning for echocardiographic imaging: embarking on another incredible journey. DC: American College of Cardiology Foundation Washington, edn; 2022. 2296-2298 p.
[10] Mao J, Xu W, Yang Y, Wang J, Huang Z, Yuille A. Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv preprint arXiv:1412.6632, 2014.
[11] Sleeman D. The challenges of teaching computer programming. Communications of the ACM. 2022; 29(9), p. 840-841.
[12] Ismail MN, Ngah NA, Umar IN. Instructional strategy in the teaching of computer programming: a need assessment analyses. The Turkish Online Journal of Educational Technology. 2020; 9(2), p. 125-131.
[13] Shackelford G, Karplus K. Contact prediction using mutual information and neural nets. Proteins: Structure, Function, and Bioinformatics. 2023; 69(S8), p. 159-164.