STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **12**, January 2024, pp 75–90.
Published online in International Academic Press (www.IAPress.org)

Hybrid GA–DeepAutoencoder–KNN Model for Employee Turnover Prediction

Chin Siang Lim [1], Esraa Faisal Malik [2], Khai Wah Khaw [2], Alhamzah Alnoor [3], XinYing Chew [1,*], Zhi Lin Chong [4], Mariam Al Akasheh [5]

[1]*School of Computer Sciences, Universiti Sains Malaysia, Penang, Malaysia*
[2]*School of Management, Universiti Sains Malaysia, Penang, Malaysia*
[3]*Management Technical College, Southern Technical University, Basrah, Iraq*
[4]*Department of Electronic Engineering, Faculty of Engineering and Green Technology, Universiti Tunku Abdul Rahman, Kampar, Malaysia*
[5]*Department of Analytics in the Digital Era, College of Business and Economics, United Arab Emirates University, UAE*

Abstract Organizations strive to retain their top talent and maintain workforce stability by predicting employee turnover and implementing preventive measures. Employee turnover prediction is a critical task, and accurate prediction models can help organizations take proactive measures to retain employees and reduce turnover rates. Therefore, in this study, we propose a hybrid genetic algorithm–autoencoder–k-nearest neighbor (GA–DeepAutoencoder–KNN) model to predict employee turnover. The proposed model combines a genetic algorithm, an autoencoder, and the KNN model to enhance prediction accuracy. The proposed model was evaluated and compared experimentally with the conventional DeepAutoencoder–KNN and k-nearest neighbor models. The results demonstrate that the GA–DeepAutoencoder–KNN model achieved a significantly higher accuracy score (90.95%) compared to the conventional models (86.48% and 88.37% accuracy, respectively). Our findings are expected to assist human resource teams identify at-risk employees and implement targeted retention strategies to improve the retention rate of valuable employees. The proposed model can be applied to various industries and organizations, making it a valuable tool for human resource professionals to improve workforce stability and productivity.

Keywords Autoencoder, Employee turnover, GA-DeepAutoencoder-KNN, Genetic algorithm, Hybrid machine learning architecture, KNN, Turnover prediction

AMS 2010 subject classifications 68T01, 68U01

DOI: 10.19139/soic-2310-5070-1799

1. Introduction

Employee turnover occurs when individuals leave their positions in business organizations due to several factors. There are two distinct categories of employee turnover, i.e., voluntary and involuntary, which are not dependent on the organization [1]. However, companies may experience employee turnover for various reasons within these categories, such as receiving better job offers from competing companies, dissatisfaction with salary, and/or strained relationships with management [2]. Involuntary turnover occurs when businesses terminate or lay off workers. In contrast, voluntary turnover occurs when a worker independently decides to resign or discontinue their engagement with the organization [3, 4]. Thus, the most significant difference between the two types of "turnover" is the initiator of the process. A previous study [5] employed a meta-analytic technique to identify which factors, such as age, pay, tenure, job satisfaction, and employment perception, have a high correlation with

*Correspondence to: XinYing Chew (Email: xinying@usm.my).School of Computer Sciences, Universiti Sains Malaysia, Penang, Malaysia

ISSN 2310-5070 (online) ISSN 2311-004X (print)

employee turnover. However, job satisfaction is a direct measure of the employees' attitudes toward their work, including various factors such as the work environment [6]. Job satisfaction serves as an indicator of how employees perceive their job and has been found to have a significant impact on the likelihood that an employee will leave the organization. Thus, job satisfaction is a critical factor to consider when predicting employee turnover [6]. Typically, high turnover has detrimental impacts on an organization because it may be difficult to find an appropriate candidate to replace an employee with specific expertise. In addition, new employees incur various costs, such as hiring and training costs [7]. The Society for Human Resource Management has stated that replacing an employee can cost an organization between six to nine months of the departing employee's salary [8]. For example, if an employee earns $60,000 annually, the recruitment and training expenses can range between $30,000 to $45,000 [8].

Involving employees in discussions about the reasons for their attrition is the most effective approach to improving retention [9]. In addition, a learning curve is involved when a new employee takes over a position, which can lead to reduced productivity. In the worst-case scenario, employees may even share their loyalties with competing corporations. The causes, importance and retention strategies of employee turnover was discussed by the previous study. [10] while Reference [[11]] investigated the case study of employee turnover at ice cream deli in Mexico. A previous study [12] classified turnover costs into direct and indirect costs. The direct turnover cost includes the cost of leaving as well as hiring, replacement, and transition costs, and the indirect turnover cost includes the cost of reduced productivity and performance [13]. Table 1 summarizes the direct and indirect costs associated with employee turnover. In summary, human resources can minimize the impact of employee turnover by taking proactive actions to retain employees and by planning for attrition risk analysis.

Table 1. Direct and Indirect Turnover Costs

Direct Turnover Cost	**Indirect Turnover Cost**
Hiring and recruiting costs	Loss of knowledge
Training and orientation costs	Loss of trade secrets
Severance cost	Loss of organization productivity and performance
Time spent on recruiting, hiring, and training processes	Loss of organization productivity and performance

As mentioned previously, employee turnover has a detrimental impact on productivity if experienced workers frequently leave the organization, which causes the organization to have a high percentage of inexperienced workers [12]. High turnover numbers can reduce an organization's efficiency due to the huge costs associated with candidate selection and recruitment [7]. Acquiring employee data is required to solve the employee turnover problem. However, data with high dimensionality tends to be a challenge for most machine learning tasks. Dimensionality reduction is a critical procedure that mitigates the detrimental impacts of a high-dimensional feature space and the data sparsity problem [14]. Reducing data dimensionality also ignores the data's noisy features while emphasizing highly relevant features [14].In addition, a large number of inputs increase model complexity and makes it more difficult to solve the turnover classification task. The curse of dimensionally occurs when a calculation becomes extremely complex with a high number of variables, which affects the performance of the model by making it increasingly difficult to discover the relations among data.

Deep learning techniques can be used to predict employee turnover by analyzing employee feedback and information. Deep learning techniques have succeeded in applications in various domain, such as healthcare [15], finance [16], cloud detection [17] and human resource, specifically employee turnover [18, 19, 20, 21].It can learn feature hierarchies from higher levels in the hierarchy generated by a combination of lower-level characteristics [22]. Deep learning techniques which rely on representation-learning to extract multiple levels of representation. This process involves the composition of uncomplicated yet nonlinear modules that transform the raw input representation at one level into a slightly more abstract representation at a higher level [23]. The concept of representation-learning encompasses a range of techniques that enable a machine to be supplied with raw data and autonomously uncover the necessary representations for classification or detection purposes [23]. These representation-learning approaches frequently employ many layers of representation generated from basic but nonlinear modules that translate representation at one level into a more basic level [23]. Previous studies have explored the fusion of autoencoder and ML techniques, such as the hybrid autoencoder–k-nearest neighbor

(KNN) model, [24] for improved breast cancer prediction on datasets with high-dimensional and uncertain data. In addition, another study [25] combined a sparse autoencoder and a support vector machine (SVM) for classification tasks, achieving an increased classification accuracy of 70.90%–75.90% compared to a single SVM model on a tabular dataset with 1000 observations and 24 features, which is similar to the dataset used in this study. Thus, in this study, we utilized a DeepAutoencoder in conjunction with a KNN model. As previous study [25] demonstrated the utility of genetic algorithms (GA) in terms of enhancing the performance of a deep autoencoder; thus, in this study, a GA is also utilized to determine the optimal hyperparameters for the DeepAutoencoder–KNN model.The objective of this study is to propose a hybrid model, which we refer to as the GA–DeepAutoencoder–KNN model, to address the employee turnover problem. While previous attempts have been made to predict employee turnover, to the best of our knowledge, no study has investigated the use of the GA–DeepAutoencoder–KNN model to predict employee turnover.

The remainder of this paper is organized as follows. Section 2 outlines the materials, methods, and proposed model. Section 3 presents the experimental results and a corresponding discussion. Finally, the paper is concluded in Section 4, including suggestions for potential future work.

2. Materials and methods

This section outlines the model design utilized to construct the proposed GA–DeepAutoencoder–KNN model. The primary aim of our experiment was to predict employee attrition, which is a binary classification prediction problem with outputs of "0" or "1," where "0" signifies that the employee remains with the organization and "1" represents that they have left the organization. To classify the output data from the DeepAutoencoder, several algorithms, such as random forest (RF), KNN, decision tree (DT), SVM, and naïve Bayes (NB), were evaluated to select the best technique. Note that the selected classifiers are the common algoritms that frequesntly applied in the existing research works [26, 27, 28] The KNN algorithm was selected as the classifier due to its promising performance (Table 2). The RF and KNN models exhibited high accuracy scores of 90.81% and 88.37%, respectively; however, we found that the RF model incurred considerably higher computational costs than the KNN model, as shown in Figure 1. The computation time of the RF model was 177 s, which is 89% more than that of the KNN classifier (only 18 s). Thus, the KNN model was used as the classifier to predict employee turnover owing to its superior accuracy and lower computational costs.

Table 2. Accuracy Scores of ML Algorithms

Model	**Accuracy (%)**
RF	90.81
KNN	88.37
DT	83.78
SVM	70.00
NB	68.24

According to the literature [29], the KNN model is very costly when facing the curse of dimensionality on a large number of datasets. The idea behind nearest neighbor classification is to categorize data items according to the class of their nearest neighbors, and it is frequently advantageous to include multiple neighbors [30]. However, the nearest neighbor calculation in the KNN method cannot discriminate candidate points when the distance between the nearest and farthest points from the query point is nearly equal in a high-dimensional space [31]. In addition to the curse of dimensionality, prediction accuracy can be reduced greatly due to the presence of noise in a dataset [32]. Thus, the proposed model utilizes the DeepAutoencoder before the KNN model to reduce the feature dimensionality by learning the relationships between data. Subsequently, the KNN algorithm is trained using training and test sets generated by the DeepAutoencoder. In addition, GridSearchCV is employed to select optimal hyperparameters for the KNN model, and the results are shown in Table 3.

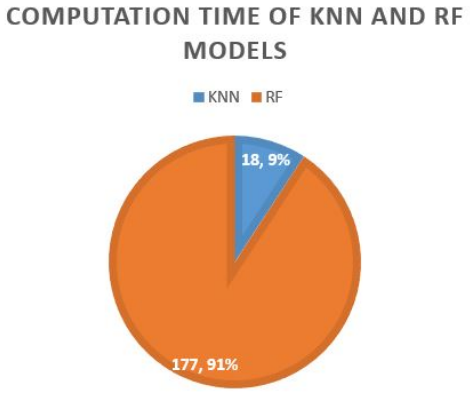

Figure 1. Computation time of KNN and RF models

Table 3. KNN Hyperparameter Selection Using GridSearchCV

Hyperparameter	Description	Value
n_neighbors	Number of neighbors	1
weights	Weight function used in prediction	Uniform
metric	Distance metric	Manhattan

2.1. Autoencoder Integration

An autoencoder is a type of neural network that has the same number of nodes in the input and output layers [33]. It learns the relationships of the data in an unsupervised learning manner. Interesting information about the data structure can be discovered by limiting the number of hidden nodes in an autoencoder method [34]. Here, the network is forced to learn a compressed representation of the input by configuring the smaller node in the hidden layer. Additionally, the correlation between the input features can be discovered when the number of hidden layers increases. Figure 2 shows the autoencoder's structure, which is composed of an encoder layer, code size layer, and decoder layer. The node located in the middle layer is referred to as the code size, and a smaller code size corresponds to a higher degree of data compression.

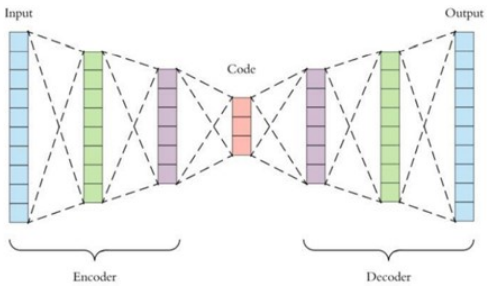

Figure 2. Autoencoder structure

The autoencoder equations can be expressed as follows:

$$z = f(w_e, b_e; x),$$

$$r = g(w_d, b_d; z),$$

where f(.) and g(.) represent the encoder and decoder functions of the autoencoder, respectively. In addition, w_e and b_e represent the parameter weight and bias for the encoder, respectively, and w_d and b_d represent the parameter weight and bias for the decoder, respectively [35]. In our experiment, the DeepAutoencoder was provided with a total of 30 features through its input layer, with a corresponding number of 30 nodes allocated to the output layer. Note that the activation function utilized in the proposed model is consistent with that of the autoencoder-SVM model [36]. Here, the ReLu activation function is used for the hidden layer, and the mean square function is used as the loss function. In summary, the proposed autoencoder model has the following training steps.

1. Feed the 30-input feature into the input layer of the autoencoder
2. The input is transformed into a lower-dimensional layer (encoder) and then expanded to reproduce the initial data (decoder). The equations for the encoder and decoder layers are expressed as follows:

$$Y = f(x) = s(WX + b_x),$$

$$X^{'} = g(Y) = s(W^{'}Y + b_y),$$

where f(x) represents the decoded function, g(x) is the decoded function, W represents the weight, and b represents the bias. In addition, S is the activation function. In our experiment, ReLu was used as the activation function for the hidden layer of the autoencoder. The autoencoder training process was used to determine the parameter $\theta = (W, b_x, b_y)$ to minimize the reconstruction loss with the cost function, and the mean squared error was used as the cost function for the proposed autoencoder. The mean squared error is expressed as follows:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - Y)^2$$

where n is the number of data points, Y_i represents the observed value, and Y represents the predicted value.
3. The output of the decoder is fed into the output layer with 30 nodes (which is the same as the number of input nodes). Here, the sigmoid function is used as the activation function for the output layer; however, the related hyperparameters, such as the number of hidden layers, number of nodes in the hidden layers, number of epochs, and learning rate, can greatly affect the performance of the autoencoder. Thus, the GA was utilized to find the optimal hyperparameter values for the autoencoder (Figure 3).

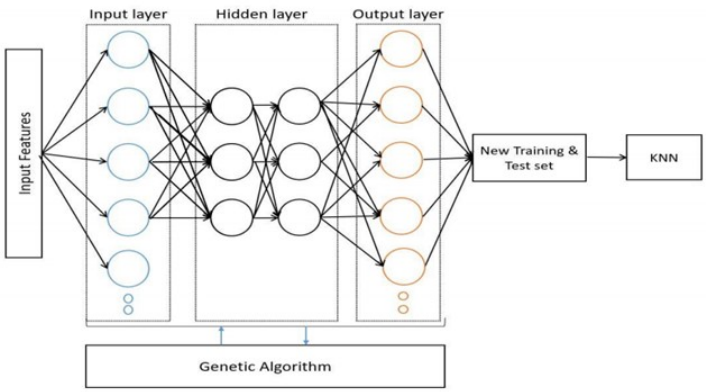

Figure 3. Architecture of the proposed model

2.2. GA Integration

The GA is an optimization algorithm that can discover the optimal areas of a search space by mimicking the process of Darwinian evolution [37].The individuals in the GA can be represented as a fixed-length string or vector that encodes a single possible solution to the given problem [38]. The GA begins by generating random individuals to form an initial population, and these individuals are then assigned a fitness value using a fitness function, where higher scores indicate better solutions, which increases the likelihood of being selected as parents. Then, mutation and recombination operators are used to promote solution diversity. Eventually, a new population (referred to as "children") is created to continue the iterative cycle. Figure 4 shows the GA process employed in the proposed model, beginning with the population initialization phase and continuing through fitness evaluation, selection, crossover, mutation, and termination.

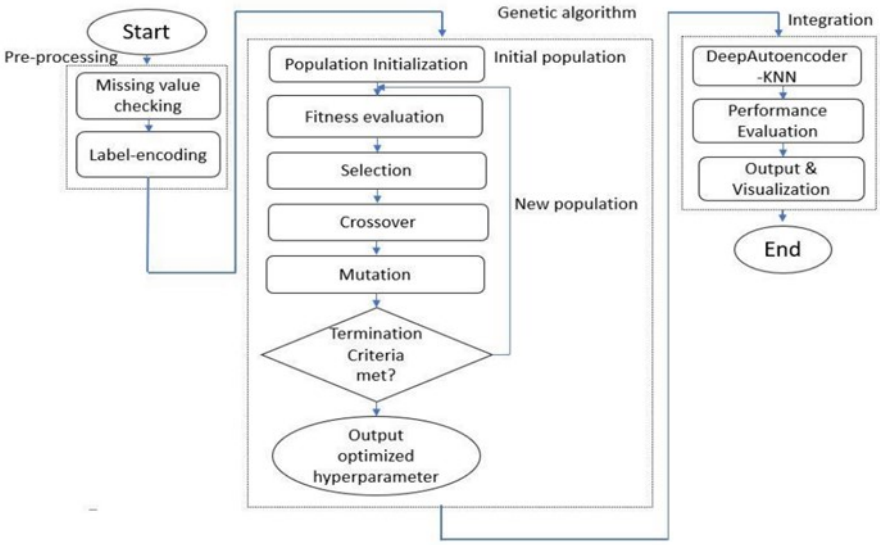

Figure 4. Process of GA

According to the literature [39], finding the optimal number of initial population is a difficult problem. The factors that could affect the initial population of the GA include the fitness function, search space, diversity, problem difficulty, selection pressure, and the number of individuals. Thus, in this study, the initial population was set to 10 despite the fact that different numbers of generations were tested and evaluated (Section 3). For example, if we set the initial population number to 10 and run 10 generations using the genetic algorithm. In this case, a total of 100 fitness functions will run through the GA. The reason being, with each generation generating 10 new populations, there will be a total of 42 fitness functions. The total fitness function can be calculated as follows. Furthermore, The decision variables are described in Table 4.

$$\begin{aligned} Total\ number\ of\ fitness\ functions &= initial\ population \times generation \\ &= 10 \times 10 \\ &= 100\ total\ fitness\ function \end{aligned} \tag{1}$$

A previous study [40] suggested that the number of nodes in the hidden layer is smaller than the number of nodes in the input layer to extract the correlation between inputs. Consequently, the decision variable for the number of nodes in the hidden layer was set between 5 and 25. Note that a small learning rate causes small weight changes, and a large learning rate causes larger changes [41]. In addition, an incorrect learning rate may cause the GA search to fall into a suboptimal solution. In the experiment, learning rates from 0.001 to 0.25 were set as the

Table 4. Decision Variables for the Hyperparameters

Hyperparameter	**Decision Variables**
Learning rate	0.001, 0.003, 0.004, 0.100, 0.150, 0.200, 0.250
Number of hidden layers	1-7
Number of nodes in hidden layer 1	5, 10, 15, 20, 25
Number of nodes in hidden layer 2	5, 10, 15, 20, 25
Number of nodes in hidden layer 3	5, 10, 15, 20, 25
Number of nodes in hidden layer 4	5, 10, 15, 20, 25
Number of nodes in hidden layer 5	5, 10, 15, 20, 25
Number of nodes in hidden layer 6	5, 10, 15, 20, 25
Number of nodes in hidden layer 7	5, 10, 15, 20, 25
Number of epochs	10, 20, 30, 40, 50, 60, 70, 80, 90, 100
Batch size	10, 20, 30, 40, 50, 60, 70, 80, 90, 100

decision variable. The generation of the solution permits an asymmetric autoencoder, which means that the number of layers in the encoder and decoder may not be equal. The GA will determine the hyperparameter values for the autoencoder using an approach that is similar to that reported in the literature [42], where the authors allow the solution of an asymmetric autoencoder to be obtained from an evolutionary algorithm. Chromosome encoding is the process of representing genetic information in a form that can be manipulated by an evolutionary algorithm, where the encoding scheme determines how the genes in the chromosome correspond to the traits or parameters of the individual being evolved. Figure 5 shows the encoding scheme for the chromosome.

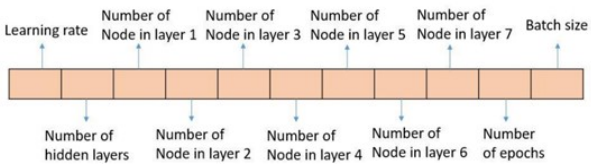

Figure 5. Chromosome encoding

2.2.1. Fitness Evaluation In the fitness evaluation phase, the hyperparameter of each population was input to the DeepAutoencoder–KNN model to obtain the model's testing accuracy score, where a higher accuracy score means the model exhibits better predictive performance. The accuracy score of each population was then stored in an array for the next parent selection process.

2.2.2. Parent And Survivor Selection The selection process in the GA ensures that an individual with a high fitness value is selected for the next generation. There are many types of selection methods for the GA, such as tournament selection, Boltzmann selection, rank selection, and steady-state selection. Tournament selection applies a roulette wheel approach that places all individuals on the roulette wheel according to their fitness value [43]. Here, the larger the fitness value, the large the segment of the roulette wheel will be assigned to the individual. Note that the operation of the roulette wheel is stochastic, and as such, the individuals with the highest probabilities are more likely to be selected for the next generation. This may lead to a biased selection toward individuals with high fitness; however, an individual with low probability may still have a chance to be selected for the generation cycle. Rank selection is another type of parent selection technique that involves sorting the population based on the individual's fitness value and assigning them a rank. Parents are selected based on rank rather than fitness. Rank selection is an explorative selection technique that can prevent quick convergence and can overcome the scaling

problem. The formula of the rank selection technique is expressed as follows [44]:

$$r_{sum_i} = \sum_{i=1}^{N} r_{i,j}, \tag{2}$$

$$PRANK_i = \frac{r_{i,j}}{r_{sum_i}}$$

where rsum represents the sum of ranks for all individuals, and PRANK is the selection probability. In this study, an existing tournament selection technique [43] was applied as the parent selection method, where the probability of an individual being selected for mating is based on the individual fitness value over the total fitness value, which is expressed as follows:

$$P_{FPS}(i) = \frac{f_i}{\sum_{j=1}^{\mu} f_i} \tag{3}$$

$$f_i = \text{individual fitness}; \qquad \sum_{j=1}^{\mu} f_i = \text{total fitness} \tag{4}$$

Note that a higher accuracy value indicates that the model exhibits better predictive performance. An individual with higher accuracy takes a large segment on the roulette wheel, and an individual with lower accuracy takes a small segment on the roulette wheel. As a result, an individual with higher accuracy will have a high chance of being selected as a parent. For the survivor selection, the design follows an age-based selection mechanism that produces the same number of offspring as parents and deletes all parents. Then, the surviving offspring form a new population in the next generation.

2.2.3. Crossover After the parent selection process, the GA begins the crossover phase, where two parents are combined to produce new children. Various crossover operations can be employed based on the representation of the solution, such as single-point, double-point, uniform, and cycle crossovers. Among these operations, the single-point crossover is the simplest operation in the GA, where a random cutting point is selected for both parents. Following this, new children are produced by taking the segment of both parents at the cutting point. In the double-point crossover operation, new children are produced by taking two cutting points on the parent [37]. In the proposed model, the crossover scheme follows the double-point crossover operation. Here, the cutting point of the chromosome is set as the third and seventh genes, as shown in Figure 6. The new offspring takes these parts from both parents. As a result, the new offspring has its own new element of information.

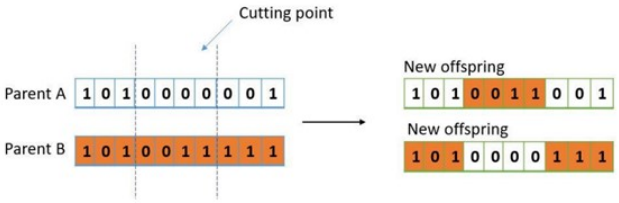

Figure 6. Double-point crossover operation

2.2.4. Mutation The mutation scheme follows a one-bit flip mutation operation, where a random point is selected and flips the individual bit at the random point (Figure 7). Additionally, the mutation probability of the child is set to 0.03, which means that not all children will proceed through the mutation process.In terms of the crossover and mutation probabilities, a previous study [45] suggested a range of 0.5–1.0 for the crossover probability and a range of 0.005–0.05 for the mutation probability. Thus, in this study, the probabilities of crossover and mutation in the GA were set to 0.8 and 0.03, respectively.

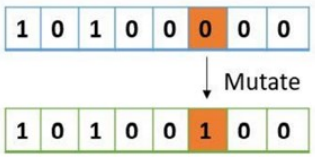

Figure 7. One-bit flip mutation

2.2.5. Termination Criteria The GA terminates when it reaches the maximum generation. In summary, the proposed GA–DeepAutoencoder–KNN model is a new architecture that has not been reported previously. The combination of the DeepAutoencoder and the KNN model solves the high dimensionality and high noise sensitivity problems of the conventional KNN technique. In addition, the GA is utilized to optimize the hyperparameters of DeepAutoencoder to increase the performance of employee turnover prediction. Details about the GA are summarized in Table 5.

Table 5. Summary of GA

Symbolic Parameters	
Representation	Bit-string
Survivor selection	Age-based
Parent selection	Tournament wheel selection
Mutation	One-bit flip
Crossover	Double-point crossover
Numeric Parameters	
Population size	10
Mutation probability	0.03
Crossover probability	0.80

The proposed GA–DeepAutoencoder –KNN model was compared to the single KNN model and the DeepAutoencoder–KNN model. Here, the DeepAutoencoder–KNN model included two hidden layers with five nodes in each layer. The output layer was set to 30 , which is the same as the input layer. The learning rate and batch size follow the default settings, i.e., 0.001 and 32, respectively. For the single KNN model, the GridsearchCV function was employed to select the best hyperparameters for the KNN model (Table 3).

2.3. Data Collection

The dataset used in this study was the IBM dataset [46], a medium-sized dataset comprising 1,500 observations and 35 features. The dataset contains a wide range of features, such as age, education, job role, monthly income, and others. In addition, the dataset includes a target feature called “attrition,” where the value “No” indicates an employee who has remained with the company, and “Yes” indicates that the employee has left the company. Other features in the dataset include business travel, daily rate, department, distance from home, employee count, employee number, satisfaction with the work environment, gender, hourly rate, job involvement, job level, job satisfaction, marital status, monthly rate, number of companies worked for, over 18 status, extent of overtime, percentage of salary hikes, performance rating, relationship satisfaction, standard hours, stock option level, total working years, and training time. Note that the feature variables vary between continuous, nominal, dichotomous, constant, and discrete.

2.4. Data Preprocessing

To enhance the accuracy and computation time of the proposed model, several preprocessing steps were applied to the dataset to eliminate noise, such as missing values, empty columns, and meaningless constant features. Here,

each column of the dataset was examined thoroughly for empty values, and no empty values were detected. Then, irrelevant features, such as ID number, constant features, and features with unclear descriptions, were removed from the dataset because they do not provide useful information to a predictive model. Additionally, the numeric variables underwent standardization, and categorical features were transformed into model-understandable numerical data using a label encoder. After completing the removal of features with constant values, the resulting dataset comprised a total of 31 features. However, the dataset was found to be imbalanced, with only 237 observations in the "Yes" class and a significantly higher number of observations (1233) in the "No" class. To address this data imbalance problem, the dataset was resampled to ensure an equal number of observations in both classes, with each class having 1233 observations. Then, the entire dataset was partitioned randomly into training and testing sets at a ratio of 80:20, respectively. The training set was used to train the model, and the testing set was used to evaluate the model's performance in predicting the employee turnover status. Consequently, the training set comprised a total of 1726 observations, and the testing set comprised a total of 740 observations.

2.5. Performance Measurement

A confusion matrix was used to evaluate the performance of the classification model. The confusion matrix is structured as a table and contains values representing the true positive (TP), false positive (FP), true negative (TN), and false negative (FN) results produced by the model. The results are shown in Table 6.

Table 6. Confusion matrix

	Actual positive (1)	**Actual negative (0)**
Predicted positive (1)	True positives (TPs)	False positives (FPs)
Predicted negative (0)	False negatives (FNs)	True negatives (TNs)

where:
• TP: The actual class of the data point is true, and the predicted class is also true (positive class).
• TN: The actual class of the data point is false, and the predicted class is also false (negative class).
• FP: The actual class of the data point is false, and the predicted class is true.
• FN: The actual class of the data point is true, and the predicted class is false.

Actual positive ad actual negative refer to the real data in the test set whereas predicted positive and predicted negative denote the predicted results computed from the machine learning model. In addition, various performance evaluation metrics, such as accuracy, recall, and precision has been employed. Accuracy is calculated as the ratio of correct predictions to the total number of predictions, and recall measures the ability of the model to identify positive cases correctly. Precision is the measurement of how good the model is at whatever it predicts. The descriptions and corresponding formulas for these metrics are given as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

3. RESULTS AND DISCUSSION

We proposed a hybrid GA–DeepAutoencoder–KNN model for employee turnover prediction. In the proposed method, the KNN classifier is initially utilized to predict the attrition status of employees owing to its high accuracy and low computational cost, which is evident from the results shown in Table 2 and Figure 1. The results of the

following experiment indicated that the single KNN model achieves 88.37% accuracy with 304 TPs and 350 TNs, which is slightly lower in performance compared to the proposed GA–DeepAutoencoder–KNN model, as shown in Table 7 and Figures 8 - 10. This difference in performance is likely due to the KNN model's limited ability to handle high-dimensional data and noise. To enhance the KNN model's classification performance by reducing the dimensionality, the DeepAutoencoder was utilized before the KNN model. However, inaccurate selection of hyperparameters in the DeepAutoencoder can reduce performance, as shown in Table 7 and Figures 8 - 10. The accuracy of the DeepAutoencoder–KNN model was 86.48%, which is less than that of the KNN model. Finally, the GA was implemented to optimize the autoencoder's hyperparameters.

Table 7. Predictive performance of the models

Models	**TP**	**TN**	**FP**	**FN**	**Accuracy**	**Recall**	**Precision**
GA-DeepAutoencoder-KNN	319	354	63	4	90.95	98.76	83.51
KNN	304	350	78	8	88.37	97.44	79.58
DeepAutoencoder-KNN	290	350	92	8	86.48	97.32	75.92
DeepAutoencoder-RF	299	306	83	52	81.76	85.19	78.27
DeepAutoencoder-DT	267	326	115	32	80.14	89.30	69.90
DeepAutoencoder-SVM	284	281	98	77	76.35	78.67	74.35
DeepAutoencoder-NB	251	211	131	147	62.43	63.07	65.71

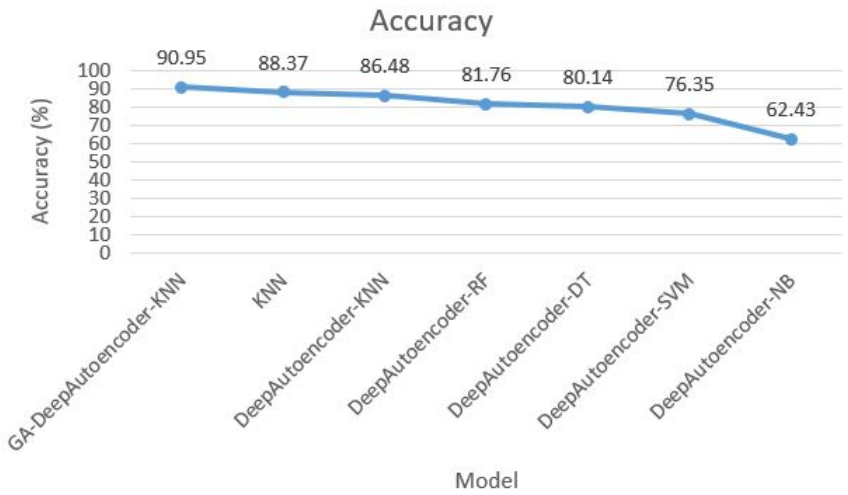

Figure 8. Comparison of Models in Accuracy

The performance of the proposed GA–DeepAutoencoder–KNN was compared to that of the DeepAutoencoder–KNN and single KNN models. The results show that the proposed GA–DeepAutoencoder–KNN model obtained the highest accuracy of 90.95% with 319 TPs and 354 TNs. The 90.95% accuracy result demonstrates the high effectiveness of the proposed model on correct prediction over total observation. Similarly, the proposed GA–DeepAutoencoder–KNN model demonstrated the highest precision of 83.51% with 319 TPs and 63 FPs. This demonstrates that the proposed model can correctly predict employee turnover. In addition, the 98.76% recall with 319 TPs and 4 FNs demonstrates the effectiveness of the proposed model at labeling actual positive samples. Compared to the DeepAutoencoder–KNN model, the proposed model proves that optimizing the hyperparameters using the GA can improve the baseline DeepAutoencoder–KNN model. Table 8 shows the optimal hyperparameters obtained by the GA with an accuracy score of 90.95%. This model comprised one input hidden layer, three hidden layers, and one output layer. The input and output layers follow the number of features in the training set, i.e., 25.

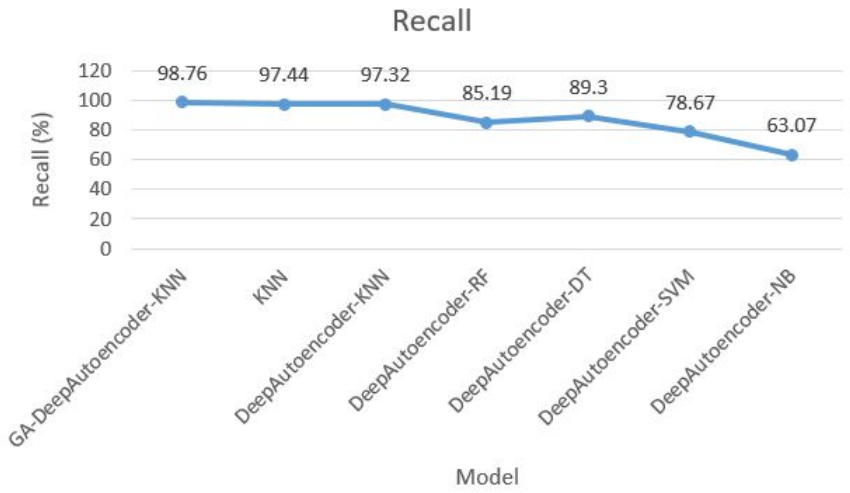

Figure 9. Comparison of Models in Recall

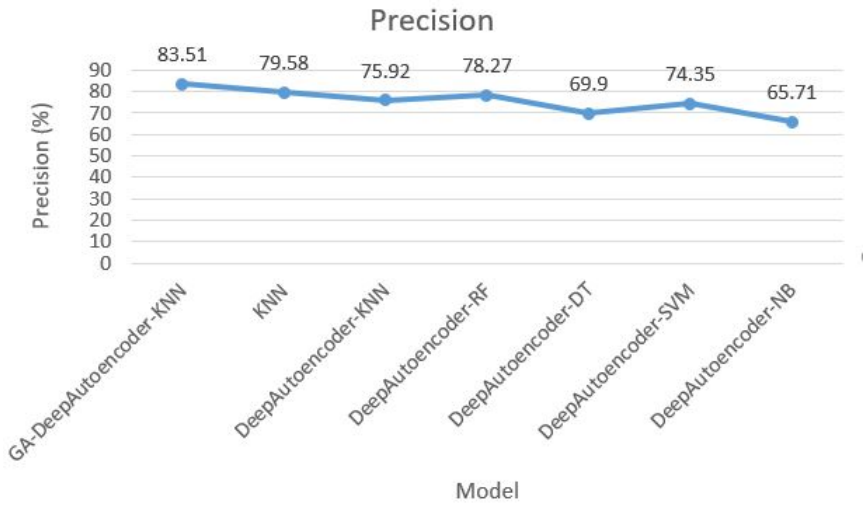

Figure 10. Comparison of Models in Precision

In addition, the number of nodes in the three hidden layers was 20, 5, and 20 nodes. The ideal learning rate was 0.001, and the number of epochs was 20. Finally, the batch size was set to 60.

In addition, the GA–DeepAutoencoder model was combined with various other ML models, such as the RF, DT, SVM, and NB models, to evaluate and compare their effectiveness. When comparing the GA–DeepAutoencoder-based ML models, we found that the DeepAutoencoder–KNN model achieved the highest accuracy score of 90.95%. The second and third highest models in terms of accuracy were the GA–DeepAutoencoder–RF and GA–DeepAutoencoder–DT models with 81.76% and 80.14% accuracy, respectively. However, a reduction in the accuracy performance was observed in tree hybrid models (GA–DeepAutoencoder–RF and GA–DeepAutoencoder–DT) compared to the single RF and single DT models. This proves that the single RF and DT models are less affected by the high dimensionality problem. Additionally, the GA–DeepAutoencoder–SVM models showed a 6.35% accuracy improvement compared to the single SVM

Table 8. Optimized hyperparameters

Hyperparameter	Value
Learning rate	0.001
Number of nodes in hidden layer	3
Number of nodes in hidden layer 1	20
Number of nodes in hidden layer 2	5
Number of nodes in hidden layer 3	20
Number of epochs	20
Batch size	60

model. For the NB combination, the accuracy decreased from 68.24% to 62.23% compared to the single NB model. In summary, the combination of the DeepAutoencoder, KNN, and SVM obtained accuracy improvement in employee turnover prediction because the KNN and SVM models are limited by the high dimensionality problem and noise in the dataset. In this study, an initial population of 10 was selected for the genetic algorithm. While a smaller population size can result in rapid convergence, it is important to note that the model may potentially be confined to local optima within the search space. Thus, the number of generations versus the average accuracy graph is plotted to observe the performance of the GA. Here, the number of generations was set between 1 and 50, as shown in Figure 11. The average fitness of the population increased as the number of generations increased. The increment in accuracy by increasing the number of generations explains the increasing fitness of the population in each generation because the GA utilizes the tournament selection technique, which favors individuals with higher fitness to be selected as the parent for the next generation. In addition, the one-bit flip mutation operation in the GA increases the variance of the population, which speeds up the search process.

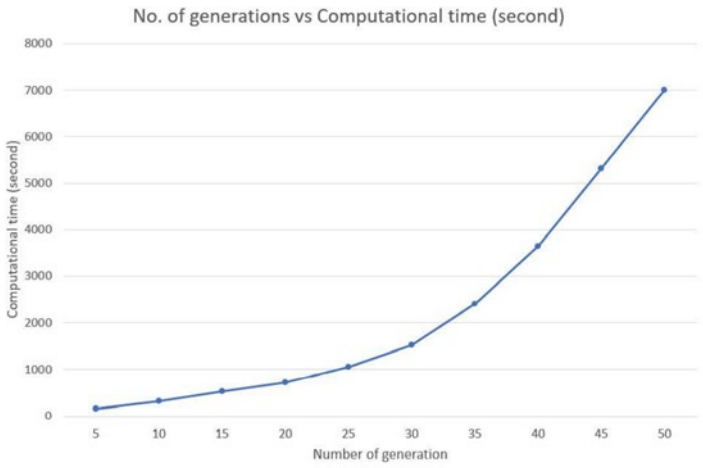

Figure 11. Number of generations vs. computation time

Moreover, a low crossover rate of 0.1 is compared to the 0.8 crossover rate in Figure 9. In 50 generations, the average speed of discovering a higher optimum solution at a 0.1 crossover rate is low, with a success rate of only 87.98%. This can be attributed to the decrease in the number of chromosome crossovers when the crossover rate is reduced. If an individual does not undergo crossover, it will have the same chromosomes in the next generation. These results demonstrate that the selection of an appropriate crossover rate is vital to avoid falling into the local optima. Figure 12 shows that the computational time of the proposed model increased as the number of generations increased. This is because more fitness function evaluations occurred in the population as the number of generations increased. For example, if the GA starts with an initial population of 10 individuals and runs for 20 generations, it will perform a total of 200 fitness evaluations. The quality of the epoch solutions generated for each individual

can also impact the computational time. If the GA generates a low epoch number of 10 for a solution, the fitness evaluation process will be faster due to the reduced number of epochs required for training the solution. The execution time of the GA was very high (2 h or (7005 s) to complete the entire execution of 50 generations.

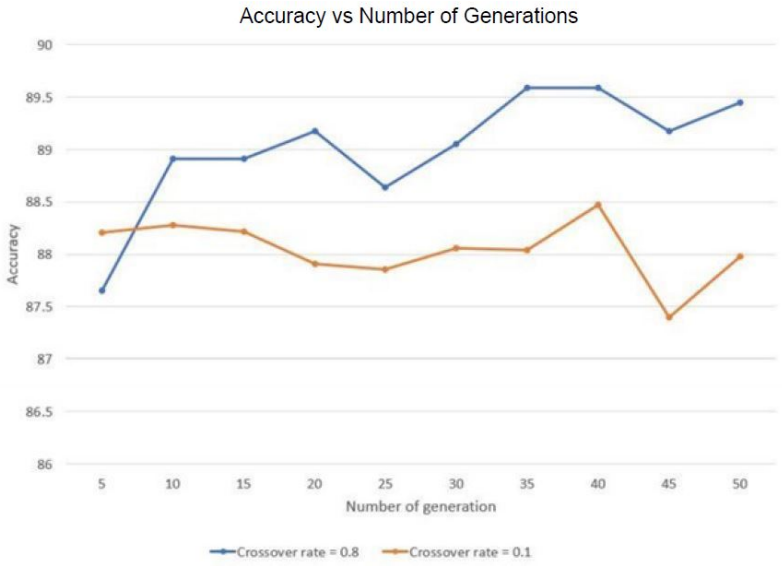

Figure 12. Accuracy vs. number of generations

4. Conclusion

The impact of employee turnover on businesses is significant and multifaceted, resulting in reduced productivity, performance, and revenue, along with increased costs associated with the recruitment, training, and replacement of employees, both in terms of time and money. In addition, finding a replacement employee with similar knowledge and expertise can be difficult and expensive. Thus, to address a research gap in the field, we proposed a GA–DeepAutoencoder–KNN model to predict employee turnover status. We realized two primary contributions. The novel GA–DeepAutoencoder–KNN model was developed to theoretically contribute on realizing effective employee turnover prediction. The proposed model's performance was evaluated and compared to the single KNN and DeepAutoencoder–KNN models. We found that the proposed model obtained promising results, practically demonstrating its ability to overcome the single KNN model's limitations in terms of high-dimensional and noisy data. In addition, integrating a GA into the DeepAutoencoder–KNN model enhanced its performance by identifying optimal hyperparameter values. The proposed model was evaluated and compared experimentally, and the results demonstrate its effectiveness for employee turnover prediction, thereby meeting the research objectives.

However, two limitations should be acknowledged. The first limitation is the experimental sample size. In this study, a low sample size was utilized because the proposed model is a proof-of-concept architecture. Thus, in the future, more data will be required for real-world implementation. In n the future, we plan to investigate various algorithms, including extreme gradient boosting and adaptive boosting, as well as use different methods, such as reinforcement learning.

Acknowledgement

This work is supported by the Ministry of Higher Education Malaysia, Fundamental and Research Grant Scheme under Grant No. FRGS/1/2019/STG06/USM/02/6 for the project entitled "A New Hybrid Model for Monitoring the Multivariate Coefficient of Variation in Healthcare Surveillance".

COMPLIANCE WITH ETHICS GUIDELINES

The authors declare they have no conflicts of interest or financial conflicts to disclose.

REFERENCES

1. F. D. Modau, N. Dhanpat, P. Lugisani, R. Mabojane, and M. Phiri, "Exploring employee retention and intention to leave within a call centre," *SA Journal of Human Resource Management*, vol. 16, no. 1, pp. 1–13, 2018.
2. S. N. Khera and Divya, "Predictive modelling of employee turnover in indian it industry using machine learning techniques," *Vision*, vol. 23, no. 1, pp. 12–21, 2018.
3. M. H. Rahman, M. A.-A. Al-Amin, M. A. Salam, T. Saha, and T. Dey, "Addressing voluntary turnover in manufacturing sectors: An empirical study," *International Fellowship Journal of Interdisciplinary Research*, vol. 1, no. 1, pp. 48–65, 2021.
4. S.-H. An, "Employee voluntary and involuntary turnover and organizational performance: Revisiting the hypothesis from classical public administration," *International Public Management Journal*, vol. 22, no. 3, pp. 444–469, 2019.
5. J. L. Cotton and J. M. Tuttle, "Employee turnover: A meta-analysis and review with implications for research," *Academy of management Review*, vol. 11, no. 1, pp. 55–70, 1986.
6. Z. He, L. Chen, and Z. Shafait, "How psychological contract violation impacts turnover intentions of knowledge workers? the moderating effect of job embeddedness," *Heliyon*, vol. 9, no. 3, 2023.
7. F. K. Alsheref, I. E. Fattoh, and W. M Ead, "Automated prediction of employee attrition using ensemble model based on machine learning algorithms," *Computational Intelligence and Neuroscience*, vol. 2022, 2022.
8. iGrad, "The cost of replacing an employee and the role of financial wellness," *https://www.enrich.org/blog/The-true-cost-of-employee-turnover-financial-wellness-enrich*, 2022.
9. A. Frye, C. Boomhower, M. Smith, L. Vitovsky, and S. Fabricant, "Employee attrition: what makes an employee quit?," *SMU Data Science Review*, vol. 1, no. 1, p. 9, 2018.
10. W. A. Al-Suraihi, S. A. Samikon, A.-H. A. Al-Suraihi, and I. Ibrahim, "Employee turnover: Causes, importance and retention strategies," *European Journal of Business Management Research*, vol. 6, no. 3, pp. 1–10, 2021.
11. L. H. v. Wobeser, G. R. Escamilla, and I. v. Wobeser, "Case study of employee turnover at ice cream deli in mexico," *Journal of Business Case Studies*, vol. 9, pp. 193–202, 2013.
12. Y. Zhao, M. K. Hryniewicki, F. Cheng, B. Fu, and X. Zhu, "Employee turnover prediction with machine learning: A reliable approach," in *Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys) Volume 2*, pp. 737–758, Springer, 2019.
13. C. P. Maertz Jr and M. A. Campion, "25 years of voluntary turnover research: A review," *Management Journal,(35)*, pp. 1036–1054, 1998.
14. H. Gunduz, "An efficient dimensionality reduction method using filter-based feature selection and variational autoencoders on parkinson's disease classification," *Biomedical Signal Processing and Control*, vol. 66, p. 102452, 2021.
15. O. Mehrpour, C. Hoyte, A. Al Masud, A. Biswas, J. Schimmel, S. Nakhaee, M. S. Nasr, H. Delva-Clark, and F. Goss, "Deep learning neural network derivation and testing to distinguish acute poisonings," *Expert Opinion on Drug Metabolism Toxicology*, vol. 19, pp. 367–380, 2023.
16. A. W. Lo and M. Singh, "Deep-learning models for forecasting financial risk premia and their interpretations," *Quantitative Finance*, vol. 23, pp. 917–929, 2023.
17. K. W. Li, N. Ma, and L. Sun, "Cloud detection of multi-type satellite images based on spectral assimilation and deep learning," *International Journal of Remote Sensing*, vol. 44, pp. 3106–3121, 2023.
18. Q. Meng, H. Zhu, K. Xiao, L. Zhang, and H. Xiong, "A hierarchical career-path-aware neural network for job mobility prediction," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 14–24, 2019.
19. D. K. Srivastava and P. Nair, "Employee attrition analysis using predictive techniques," in *Information and Communication Technology for Intelligent Systems (ICTIS 2017)-Volume 1 2*, pp. 293–300, Springer, 2018.
20. S. Al-Darraji, D. G. Honi, F. Fallucchi, A. I. Abdulsada, R. Giuliano, and H. A. Abdulmalik, "Employee attrition prediction using deep neural networks," *Computers*, vol. 10, no. 11, p. 141, 2021.
21. M. Teng, H. Zhu, C. Liu, and H. Xiong, "Exploiting network fusion for organizational turnover prediction," *ACM Transactions on Management Information Systems (TMIS)*, vol. 12, no. 2, pp. 1–18, 2021.
22. H. Moeini and F. M. Torab, "Comparing compositional multivariate outliers with autoencoder networks in anomaly detection at hamich exploration area, east of iran," *Journal of Geochemical Exploration*, vol. 180, pp. 15–23, 2017.
23. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
24. K. Adem, "Diagnosis of breast cancer with stacked autoencoder and subspace knn," *Physica A: Statistical Mechanics and its Applications*, vol. 551, p. 124591, 2020.

25. Y. Ju and W. Sun, "Plasma assisted combustion: Dynamics and chemistry," *Progress in Energy and Combustion Science*, vol. 48, pp. 21–83, 2015.
26. E. F. Malik, K. W. Khaw, and X. Chew, "A new hybrid data preprocessing technique for fraud detection prediction," *Computing and Informatics*, vol. 41, pp. 981–1001, 2022.
27. S. Shafie, S. P. Ooi, and K. W. Khaw, "Prediction of employee promotion using hybrid sampling method with machine learning architecture," *Malaysian Journal of Computing*, vol. 8, pp. 1264–1286, 2023.
28. A. Et-taleby, Y. Chaibi, M. Benslimane, and M. Boussetta, "Applications of machine learning algorithms for photovoltaic fault detection: a review," *Statistics, Optimization and Information Computing*, vol. 11, pp. 168–177, 2023.
29. P. Zhu, X. Zhan, and W. Qiu, "Efficient k-nearest neighbors search in high dimensions using mapreduce," in *2015 IEEE fifth international conference on big data and cloud computing*, pp. 23–30, IEEE, 2015.
30. P. Ajit, "Prediction of employee turnover in organizations using machine learning algorithms," *algorithms*, vol. 4, no. 5, p. C5, 2016.
31. N. Ukey, Z. Yang, B. Li, G. Zhang, Y. Hu, and W. Zhang, "Survey on exact knn queries over high-dimensional data space," *Sensors*, vol. 23, no. 2, p. 629, 2023.
32. A. Atla, R. Tada, V. Sheng, and N. Singireddy, "Sensitivity of different machine learning algorithms to noise," *Journal of Computing Sciences in Colleges*, vol. 26, no. 5, pp. 96–103, 2011.
33. J. Zhao, Y. Kim, K. Zhang, A. Rush, and Y. LeCun, "Adversarially regularized autoencoders," in *International conference on machine learning*, pp. 5902–5911, PMLR, 2018.
34. A. Gudigar, U. Raghavendra, T. N. Rao, J. Samanth, V. Rajinikanth, S. C. Satapathy, E. J. Ciaccio, C. Wai Yee, and U. R. Acharya, "Ffcaes: An efficient feature fusion framework using cascaded autoencoders for the identification of gliomas," *International Journal of Imaging Systems and Technology*, vol. 33, no. 2, pp. 483–494, 2023.
35. J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *2018 IEEE international conference on systems, man, and cybernetics (SMC)*, pp. 415–419, IEEE, 2018.
36. Y. N. Kunang, S. Nurmaini, D. Stiawan, A. Zarkasi, *et al.*, "Automatic features extraction using autoencoder in intrusion detection system," in *2018 International Conference on Electrical Engineering and Computer Science (ICECOS)*, pp. 219–224, IEEE, 2018.
37. S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, pp. 8091–8126, 2021.
38. G. Jones, "Genetic and evolutionary algorithms," *Encyclopedia of Computational Chemistry*, vol. 2, no. 1127-1136, p. 40, 1998.
39. P. A. Diaz-Gomez and D. F. Hougen, "Initial population for genetic algorithms: A metric approach.," in *Gem*, pp. 43–49, Citeseer, 2007.
40. A. Ng, "Cs294a lecture notes: Sparse autoencoder," *URL: https://web. stanford. edu/class/cs294a/sparseAutoencoder 2011new. pdf*, 2010.
41. C. F. Rodríguez-Hernández, M. Musso, E. Kyndt, and E. Cascallar, "Artificial neural networks in academic performance prediction: Systematic implementation and predictor evaluation," *Computers and Education: Artificial Intelligence*, vol. 2, p. 100018, 2021.
42. F. Assuncao, D. Sereno, N. Lourenco, P. Machado, and B. Ribeiro, "Automatic evolution of autoencoders for compressed representations," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2018.
43. S. Mirjalili, "Evolutionary algorithms and neural networks," in *Studies in computational intelligence*, vol. 780, Springer, 2019.
44. W. Duch, T. Wieczorek, J. Biesiada, and M. Blachnik, "Comparison of feature ranking methods based on information entropy," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, vol. 2, pp. 1415–1419, IEEE, 2004.
45. M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *computer*, vol. 27, no. 6, pp. 17–26, 1994.
46. P. Subhash, "Ibm hr analytics employee attrition & performance," *Retrieved on July*, vol. 10, 2017.