Improving Neural Network Classifier using Gradient-based Floating Centroid Method

Mazharul Islam^{1,#}, Shuangrong Liu^{1,#}, Lin Wang^{1,*}, and Xiaojing Zhang¹

Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan, 250022, China *Corresponding Author: wangplanet@gmail.com

Abstract. Floating centroid method (FCM) offers an efficient way to solve a fixed-centroid problem for the neural network classifiers. However, evolutionary computation as its optimization method restrains the FCM to achieve satisfactory performance for different neural network structures, because of the high computational complexity and inefficiency. Traditional gradient-based methods have been extensively adopted to optimize the neural network classifiers. In this study, a gradient-based floating centroid (GDFC) method is introduced to address the fixed centroid problem for the neural network classifiers optimized by gradient-based methods. Furthermore, a new loss function for optimizing GDFC is introduced. The experimental results display that GDFC obtains promising classification performance than the comparison methods on the benchmark datasets.

Keywords: Neural Network Classifier \cdot Classification \cdot Loss Function \cdot Floating Centroid Method

1 Introduction

In the machine learning field, supervised classification is a classical topic among scientists. There are a considerable number of classification methods has been proposed during past years, such as naive bayes [13], k-nearest neighbor [3], decision tree [8], support vector machine [2], and neural network [9]. Among these methods, the neural network attracts substantial attention for addressing real-world classification problems [5–7, 14] because of its capability of learning non-linear relationship from real-world data.

Conventionally, the classification process of the neural network is explained from the probabilistic perspective. The values of output neurons are considered as the probabilities that a sample belongs to different classes. Meanwhile, this process is also can be described from a geometric perspective. The neural network is viewed as a mapping function f. Each class is coded as the unique binary string. As the input, the sample is mapped to a space by f, in which classes are

[#] Both authors contribute equally to this article.

represent by different fixed points, referred to centroids. The binary string of each class describes the position of centroids. The mapped sample is attached to the closest centroid (class) in according to the distance. Therefore, methods acting on the output layer of the neural network, such as one-per-class, softmax [1] and error-correcting output code (ECOC) [4], can also be viewed as the methods to distribute the centroids.

For the one-per-class, softmax and ECOC, they are widely used in different neural network models optimized by gradient-based optimization method, and obtains considerable successful stories [14, 15]. However, for these fixed centroid methods, the fixed centroid problem (FCP) [10–12], which refers to that the locations, labels, and number of centroids are prior set before the training, restrains their performance. Because the FCP results in the reduction of the size of the set consisting of optimal neural networks, and enlarges the complexity of optimization. Although the floating centroid method (FCM) [11] affords a way to solve the fixed centroid problem, the evolutionary computation is adopted as its optimization method that prevents FCM to employ with the neural networks optimized by gradient-based optimization method.

Considering the facts mentioned above, gradient-based floating centroid method (GDFC) is proposed in this study. The GDFC absorbs the advantages of fixed centroids methods and floating centroids method, and affords a way to assist the neural network classifiers optimized by gradient-based optimization method to address the fixed centroid problem. For this study, the major contributions are introduced as follows:

- The gradient-based floating centroid method is proposed to tackle fixedcentroid problem for the neural network classifiers optimized by gradientbased methods.
- A new loss function, named centroid loss function, is proposed to maximize compactness of within-class and separability of between-class during training process.

The proposed GDFC method is described in Section 2. Experiment is reported and results on benchmark datasets are analyzed in Section 3. We give the conclusion, and draw the future work in Section 4,.

2 Methodology

In this section, the framework of gradient-based floating centroid method is provide firstly. Subsequently, the centroid loss function is introduced. At the end of this section, we describe the optimization process of GDFC.

2.1 Gradient-based Floating Centroid Framework

The framework of the gradient-based floating centroid is shown in Fig. 1. The training part mainly includes 4 modules: mapping by neural network, generating centroids by K-means, coloring centroids, and calculating loss to update



Fig. 1. Architecture of the GDFC

neural network. At first, the neural network maps the samples to the partition space. Afterwards, the centroids are generated in the partition space by using k-means algorithm. The classes' number can smaller than the number of centroids. Subsequently, these centroids are labeled by different classes. The labeling strategy refer to coloration process. In the coloration process, if the mapped samples which is represented by one class are the majority then the corresponding centroid is colored by that class. Besides, one class can be used to label more than one centroids. After that, the neural network is iteratively updated by the proposed centroid loss function which uses the distribution information of the centroids. In the optimization process, the centroid loss function has the ability to maximize compactness of within-class and separability of between-class simultaneously; thus clear decision boundaries exist among different clusters. Finally, an optimal neural network and centroids decided by this optimal neural network are obtained.

In the testing process, an unknown sample as input of the optimal neural network are mapped to the partition space. This unknown sample is assigned to the centroid with closest distance. For example, in Fig.1, the unknown sample is close to the centroid, which represents class 2. So, this sample categorized to class 2.

4 M. Islam et al.

2.2 Centroid Loss Function

For the mapped samples in the partition space, the k-means algorithm is used to generate the centroids $C^{(k)}(k = 1, 2, ..., K)$ by clustering these mapped samples. Then, for each mapped sample, two centroids are selected from K centroids:

The first centroid is one with minimum value of the $|| \cdot ||_2$, and this centroid having same class as the mapped sample.

$$D_{\min}^{Self} = \arg\min_{C} ||\beta^{(j)} - C^{(k)}(g)||_2$$
(1)

where j=1,2,...,m, m is the number of samples, k=1,2,...,K, K is the number of centroids, $|| \cdot ||_2$ represents the distance. Note that we can obtain the centroid which has the same class and nearest distance to the mapped sample, denotes C^S . The second centroid is one with minimum value of the $|| \cdot ||_2$, and the class of this centroid is different with the mapped sample.

$$D_{\min}^{Noself} = \arg\min_{C} ||\beta^{(j)} - C^{(k)}(g)||_2$$
(2)

Note that we can select the centroid of the different class nearest to the mapped sample, denotes C^N . Since the target of GDFC is to put the points belongs to the same class closer and enlarge the distance among the points with different classes, thus, minimizing the D_{\min}^{Self} as well as maximizing the D_{\min}^{Noself} are expected. Adopting the method of stochastic gradient descent, which attempts to minimize the global error by updating the parameters of the neural network in an iterative process. Therefore, the loss function is listed as follow.

$$E = \frac{1}{2} \sum_{q=1}^{Q} \left[(\beta_{qj} - C_{qj}^S)^2 - \xi \cdot (\beta_{qj} - C_{qj}^N)^2 \right]$$
(3)

 β_{qj} denotes the mapped value of the qth (q=1,2,..,Q) neuron in the output layer, Q represents the output neurons' number, and is also the dimension of the partition space. ξ is a constant, which is used to adjust the weight between D_{\min}^{Self} and D_{\min}^{Nosolf} . Besides, the gradient descent method is prone to over-fitting, so L_2 regularization is applied to decrease over-fitting. From the above, the centroid loss function is essential to make the following reformulation as,

$$E = \frac{1}{2} \sum_{q=1}^{Q} \left[\left((\beta_{qj} - C_{qj}^S)^2 - \xi \cdot (\beta_{qj} - C_{qj}^N)^2 \right) \right] + \frac{\lambda}{2} \cdot \sum w^2$$
(4)

Where λ is the regularization parameter.

2.3 Optimization

Based on the gradient descent method, while following the back-propagation (BP) idea of training-error to iteratively update the parameters to obtain an optimal neural network. Without loss of generality, assuming that a feedforward

neural network has L layers. η is a global learning rate. From the L layer to the L-1 layer, the partial derivatives of the weights and biases are obtained respectively.

$$\Delta w_{qh}^{(L-1)} = -\eta \frac{\partial E}{\partial w_{qh}^{(L-1)}}$$

$$= -\eta \cdot \frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial z_{qj}} \frac{\partial z_{qj}}{\partial w_{qh}^{(L-1)}} - \lambda \cdot w_{qh}^{(L-1)}$$

$$= \eta \left[\delta_q^{(L)} \alpha_{hj}^{(L-1)} - \lambda w_{qh}^{(L-1)} \right]$$
(5)

$$\Delta \theta_q^{(L)} = \eta \frac{\partial E}{\partial \theta_q^{(L-1)}} = \eta \frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial \theta_q^{(L-1)}} = -\eta \delta_q^{(L)} \tag{6}$$

Note that,

$$\delta_{q}^{(L)} = -\frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial z_{qj}} = (C_{qj}^{S} - \beta_{qj}) \cdot \sigma'(z_{qj}) - \xi(C_{qj}^{N} - \beta_{qj})\sigma'(z_{qj})$$
(7)

Where Q is neurons' number in L layer, H represents the neurons' number in the L-1 layer, $\Delta w_{qh}^{(L-1)}$ is the weight change value from the *h*th (with h=1,2,...,H) neuron of the L-1 layer to the *q*th (q=1,2,...,Q) neuron of the L layer, $\Delta \theta_q^{(L)}$ is the bias change value of the *q*th (with q=1,2,...,Q) neuron in the L layer. $\sigma(\cdot)$ represents the activation function. $\beta_{qj} = \sigma(z_{qj}) = \sigma(\sum_{h=1}^{H} (w_{qh}^{(L-1)} \cdot \alpha_{hj}^{(L-1)} + \theta_q^{(L)}), \beta$ is the activation value of the L layer neurons, α is the activation value of the L-1 layer neurons. Thus, the weights and biases between the L layer and the L-1 layer are updated as

$$w_{qh}^{(L-1)}(g+1) = w_{qh}^{(L-1)}(g) + \Delta w_{qh}^{(L-1)}$$
(8)

$$\theta_q^{(L-1)}(g+1) = \theta_q^{(L-1)}(g) + \Delta \theta_q^{(L-1)}$$
(9)

From the L-1 layer to the L-2 layer, the partial derivatives of the weights and biases are obtained, respectively

$$\begin{aligned} \Delta w_{hp}^{(L-2)} &= -\eta \frac{\partial E}{\partial w_{hp}^{(L-2)}} \\ &= -\eta \cdot \left[\frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial z_{qj}} \frac{\partial z_{qj}}{\partial \alpha_{hj}^{(L-1)}} \frac{\partial \alpha_{hj}^{(L-1)}}{\partial y_{hj}^{(L-1)}} \frac{\partial y_{hj}^{(L-1)}}{\partial w_{hp}^{(L-2)}} \right] - \eta \cdot \lambda \cdot w_{hp}^{(L-2)} \end{aligned} \tag{10}$$
$$&= \eta \cdot \left[\delta_{h}^{(L-1)} \alpha_{pj}^{(L-2)} - \lambda w_{hp}^{(L-2)} \right]$$

$$\Delta \theta_h^{(L-1)} = -\eta \frac{\partial E}{\partial \theta_h^{(L-1)}} = -\eta \frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial z_{qj}} \frac{\partial z_{qj}}{\partial \alpha_{hj}} \frac{\partial \alpha_{hj}}{\partial y_{hj}^{(L-1)}} = -\eta \delta_h^{(L-1)} \tag{11}$$

6 M. Islam et al.

Note that,

$$\delta_{h}^{(L-1)} = -\frac{\partial E}{\partial \beta_{qj}} \frac{\partial \beta_{qj}}{\partial z_{qj}} \frac{\partial z_{qj}}{\partial \alpha_{hj}^{(L-1)}} \frac{\partial \alpha_{hj}^{(L-1)}}{\partial y_{hj}^{(L-1)}} = \sigma'(y_{hj}^{(L-1)}) \cdot \sum_{q=1}^{Q} w_{qh}^{(L-1)} \cdot \delta_{q}^{(L)}$$
(12)

Here, H is the neurons' number in the L-1, P is the neurons' number in the L-2 layer, $\Delta w_{hp}^{(L-2)}$ is the weight change value from the *p*th (with p=1,2,..., P) neuron of the L-2 layer to the *h*th (with h=1,2,..,H) neuron of the L-1 layer, $\Delta \theta_h^{(L-1)}$ is the bias change value of the *h*th (with h=1,2,..,H) neuron in the L-1 layer. $\sigma(\cdot)$ is the activation function. $\alpha_{pj}^{(L-1)} = \sigma(y_{hj}^{(L-1)}) = \sigma(\sum_{p=1}^{P} w_{hp}^{(L-2)} \alpha_{pj}^{(L-2)} + \theta_h^{(L-1)}), \alpha^{(L-1)}$ is the activation value of neurons in the L-1 layer. $\alpha^{(L-2)}$ is the activation value of neurons in the L-1 layer.

Thus, the weights and biases between the L-1 layer and the L-2 layer are updated as,

$$w_{hp}^{(L-2)}(g+1) = w_{hp}^{(L-2)}(g) + \Delta w_{hp}^{(L-2)}$$
(13)

$$\theta_h^{(L-1)}(g+1) = \theta_h^{(L-1)}(g) + \Delta \theta_h^{(L-1)}$$
(14)

Through the above derivation, we can generalize the general update formulas for the L-l layer to the L-(l-1) with l=1,2,..,L-2 layer weights and biases, can be written as below:

$$\delta^{(L-l)} = \sigma'(y^{(L-l)}) \cdot \sum w^{(L-l)} \cdot \delta^{(L-l+1)}$$
(15)

then,

$$\Delta w^{(L-l)} = \eta \tau(x) [\delta^{(L-l)} \alpha^{(L-l+1)} - \lambda w^{(L-l+1)}]$$
(16)

$$\Delta \theta^{(L-l+1)} = -\eta \delta^{(L-l+1)} \tag{17}$$

Thus, the weights and biases are updated as

$$w^{(L-l)}(g+1) = w^{(L-l)}(g) + \Delta w^{(L-l)}$$
(18)

$$\theta^{(L-l+1)}(g+1) = \theta^{(L-1)}(g) + \Delta \theta^{(L-l+1)}$$
(19)

3 Experiments

3.1 Overview of the Datasets

Ten classification benchmark datasets is employed to evaluate models. Table 1 describes the characteristics of the datasets, containing datasets name (Data set), abbreviation of the datasets (Abbr.), size of the datasets (Size), dimensions of the datasets (Dim.), and number of classes (Class).

Abbr.	Size	Dim.	Class
Diabetes	392	8	2
Vote	232	16	2
RFCC	668	33	2
SPECT	267	22	2
CMSC	540	18	2
Web	1353	9	3
$_{\rm HR}$	160	3	3
Balance	625	4	3
Wine	178	13	3
UKM	403	5	4
	Abbr. Diabetes Vote RFCC SPECT CMSC Web HR Balance Wine UKM	Abbr. Size Diabetes 392 Vote 232 RFCC 668 SPECT 267 CMSC 540 Web 1353 HR 160 Balance 625 Wine 178 UKM 403	Abbr. Size Dim. Diabetes 392 8 Vote 232 16 RFCC 668 33 SPECT 267 22 CMSC 540 18 Web 1353 9 HR 160 3 Balance 625 4 Wine 178 13 UKM 403 5

Table 1. Datasets Descript

,

 Table 2. Comparative Methods

Туре	Method	
Neural Network-based Methods	Feed-forward Neural Network (FNN)	
	Nearest Neighbor Partitioning (NNP)	
	Floating Centroid Method (FCM)	
Other Classification Methods	Nave Bayes (NB)	
	Support Vector Machine (SVM)	
	K-nearest Neighbor (KNN)	

3.2 Comparison Methods

We choose different types of classifier in the experiment to compare the efficiency with the GDFC method. Table 2 is used to introduce these methods. For a fair comparison with the proposed method, the potentiality of these methods is explored, and their parameters are tuned by trial and error.

- KNN is a classification method. For the KNN, the number of nearest neighbors is selected in the range {1, 30}.
- For the SVM, cost parameter is selected from the range $\{2^{-2}, 2^{-5}\}$.
- For the GDFC and FCM, the number of hidden layer neurons is set from range $\{1, 40\}$. the number of dimensions of the partition space is selected from $\{N, 10N\}$, and for the number of centroids is from the range $\{N, 5N\}$, where N is equal to the number of classes.
- For NNP, hidden layer neurons number is selected from the range $\{1, 40\}$. The dimension of partition space is chosen from the range $\{N, 10N\}$, and the number of centroids from the range N, 5N. The value of parameter p fixed at 3.
- For FNN, hidden layer neurons number is selected from the range $\{1, 40\}$.

8 M. Islam et al.

3.3 Evaluation Metrics

As the evaluation metrics, Generalization Accuracy (GA) and Average F-measure (Avg.FM) is adopted to evaluate the efficiency of the GDFC method.

$$GA = \frac{TC}{TN} \times 100\% \tag{20}$$

$$Avg.FM = \frac{2 \times \left(\frac{precision \times recall}{precision + recall}\right)}{Num} \times 100\%$$
(21)

The performance of all methods is evaluated by using ten-fold cross-validation. At first, the whole dataset is randomly split into ten subsets and then one subset is chosen for testing, and other subsets are employed for training. This whole process repeated for ten times, and the final result is considered by the mean of ten results.

 Table 3. Accuracy comparison with neural network-based methods. The unit of results is percentage.

Method	FNN	NNP	FCM	GDFC
Diabetics	75.5	75	77.75	79.5
Vote	78.75	92.9	92.9	94.17
RFCC	88.82	87.06	90	92.35
SPECT	80.71	78.21	80.71	82.07
CMSC	92.55	92.73	94	96.09
Web	86.64	84.5	85.11	88.41
HR	71.11	79.44	77.22	81.11
Balance	95.08	94.6	95.87	96.35
Wine	94.44	98.89	98.89	98.89
UKM	89.05	95.24	95.95	96.18
MEAN	85.27	87.86	88.84	90.51

3.4 Results Analysis

We demonstrate the experimental results and the findings in this subsection. In our experiments, we compared the proposed GDFC method with six different classifiers, including SVM, NB, KNN, FNN, NNP, and FCM. Table 3 and 4 exhibit the testing accuracy and Avg.FM of neural network-based methods on each dataset.

From Table 3, our proposed GDFC achieved better generalization accuracy on nine datasets out of ten. Only in Wine dataset, the generalaization performance of NNP, and FCM is alike to the proposed method. That phenomenon clarifies that the GDFC is promising compared with comparative methods in terms of generalization performance. Table 4 shows the comparison between the proposed method with neural network-based methods based on Average Fmeasure. As compared with other methods, GDFC has better performance on eight of the ten datasets. Only in Balance and Wine dataset FCM and NNP has slightly better performance respectively. That reveals GDFC has better generalization ability than other omparative methods. For classification task, Avg.FM is a significant appraisal for the classifiers because Avg.FM summaries both precision and recall to evaluate the performance. Which demonstrates that, the GDFC has a better balance between precision and recall compared to other competing methods.

Furthermore, based on mean accuracy which is the average results of each method on ten datasets. Gradient-based floating centroid method improved about **3.50%** (versus FNN), **1.77%** (versus NNP) and **1.15%** (versus FCM) in terms of testing accuracy. Moreover, GDFC improved about **7.81%** (versus FNN), **1.82%** (versus NNP) and **3.33%** (versus FCM) in terms of Avg.FM. Fig.

MEAN	74.55	83.54	81.27	86.27
UKM	82.13	94.88	96.11	97.08
Wine	93.6	98.97	98.88	98.93
Balance	84.87	90.23	92.81	89.96
Hr	69.6	80.46	77.24	81.48
Web	74.83	81.4	66.55	85.14
CMSC	62.38	77.95	75.84	83.79
SPECT	63.16	68.12	67.46	73.07
RFCC	59.17	77.23	72.65	83.76
Vote	82.69	92.9	92.9	94.14
Diabetics	73.07	73.25	72.21	75.29
Method	FNN	NNP	FCM	GDFC

Table 4. Avg.FM comparison with neural neteork-based methods. The unit of results is percentage.

2 and Fig.3 displays generalization accuracy and Avg.FM of other existing classification methods, including SVM, NB and KNN. GDFC method achieves better performance on ten datasets in terms of classification accuracy. For Avg.FM, GDFC achieves higher performance of eight datasets out of ten datasets. SVM achieves better performance on RFCC dataset, and in Wine dataset SVM has similar performance as GDFC. Furthermore, it is noticeable that the average accuracy of GDFC is higher than the other methods. Gradient-based floating centroid method improved about 2.83% (versus SVM), 4.24% (versus NB), 3.71% (versus KNN) and 4.02% (versus SVM), 6.58% (versus NB), 7.54% (versus KNN) in terms of Avg.FM. That concludes our proposed GDFC method is superior to the other competing methods.



Fig. 2. Comparison between GDFC and the other classication methods in terms of generalization accuracy.



Fig. 3. Comparison between GDFC and the other classication methods in terms of average F-measure.

4 Conclusions

In this study, a novel gradient-based floating centroid (GDFC) method is introduced to solve the fixed-centroid problem for gradient-based neural network classifier. Moreover, the centroid loss function is introduced for maximizing the compactness of within-class and the separability of between-class during the optimization process. Experimental results indicated that proposed gradient-based floating centroid (GDFC) method outperformed the competing methods on the majority of the datasets in terms of classification accuracy and Avg.FM. In the future, different neural network structures will be investigated and employed in GDFC, considering that the mapping ability of the neural network is one of the vital factors for the classification performance of GDFC.

Acknowledgements. This work was supported by National Natural Science Foundation of China under Grant No. 61573166, No. 61572230, No. 61872419, No. 61873324, No. 81671785, No. 61672262. Project of Shandong Province Higher Educational Science and Technology Program under Grant No. J16LN07. Shandong Provincial Natural Science Foundation No. ZR2019MF040, No. ZR2018LF005. Shandong Provincial Key R&D Program under Grant No. 2018GGX101048, No. 2016ZDJS01A12, No. 2016GGX101001, No. 2017CXZC1206. Taishan Scholar Project of Shandong Province, China.

References

- Bridle, J.S.: Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In: Soulié, F.F., Hérault, J. (eds.) Neurocomputing. pp. 227–236. Springer Berlin Heidelberg, Berlin, Heidelberg (1990)
- 2. Chua, K.S.: Efficient computations for large least square support vector machine classifiers. Pattern Recogn. Lett. **24**(1), 75–80 (Jan 2003)
- Cunningham, P., Delany, S.: k-nearest neighbour classifiers. Mult Classif Syst 34(8), 1–17 (March 2007)
- Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via errorcorrecting output codes. J. Artif. Int. Res. 2(1), 263–286 (Jan 1995)
- Jiang, G., He, H., Yan, J., Xie, P.: Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox. IEEE Transactions on Industrial Electronics **PP**, 1–1 (06 2018)
- Kamilaris, A., Prenafeta-Bold, F.X.: A review of the use of convolutional neural networks in agriculture. The Journal of Agricultural Science 156(3), 312–322 (2018). https://doi.org/10.1017/S0021859618000436
- Nazari, M., Oroojlooy, A., Snyder, L., Takac, M.: Reinforcement learning for solving the vehicle routing problem. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31, pp. 9839–9849. Curran Associates, Inc. (2018)

- 12 M. Islam et al.
- Quinlan, J.R.: Induction of decision trees. MACH. LEARN 1(1), 81–106 (March 1986)
- Rafiei, M.H., Adeli, H.: A new neural dynamic classification algorithm. IEEE Transactions on Neural Networks and Learning Systems 28(12), 3074–3083 (Dec 2017)
- Wang, L., Yang, B., Chen, Y., Zhang, X., Orchard, J.: Improving neural-network classifiers using nearest neighbor partitioning. IEEE Transactions on Neural Networks and Learning Systems 28(10), 2255–2267 (Oct 2017)
- Wang, L., Yang, B., Chen, Y., Abraham, A., Sun, H., Chen, Z., Wang, H.: Improvement of neural network classifier using floating centroids. Knowledge and Information Systems 31(3), 433–454 (Jun 2012)
- Wang, L., Yang, B., Chen, Z., Abraham, A., Peng, L.: A novel improvement of neural network classification using further division of partition space. In: Mira, J., Álvarez, J.R. (eds.) Bio-inspired Modeling of Cognitive Tasks. pp. 214–223. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
- Wang, T., Yang, J.: A heuristic method for learning bayesian networks using discrete particle swarm optimization. Knowledge and Information Systems 24(2), 269–281 (Aug 2010)
- Wibowo, A., Wiryawan, P.W., Nuqoyati, N.I.: Optimization of neural network for cancer microRNA biomarkers classification. Journal of Physics: Conference Series 1217, 012124 (may 2019)
- Wong, Y.J., Arumugasamy, S.K., Jewaratnam, J.: Performance comparison of feedforward neural network training algorithms in modeling for synthesis of polycaprolactone via biopolymerization. Clean Technologies and Environmental Policy 20(9), 1971–1986 (Nov 2018)