Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Supplementary materials for

Jie CHEN, Dandan WU, Ruiyun XIE, 2023. Artificial intelligence algorithms for cyberspace security applications: a technological and status review. *Front Inform Technol Electron Eng*, 24(8):1117-1142. https://doi.org/10.1631/FITEE.2200314

1 Machine learning

1.1 Supervised learning

Supervised learning uses tagged data for training, mainly to settle classification and regression problems, such as text classification, spam filtering, medical diagnosis, pattern recognition, and precision marketing. The most widely used algorithms are the K-nearest neighbor (K-NN), support vector machine (SVM), naive Bayesian (NB), and decision tree (DT) algorithms.

K-NN algorithm uses Euclidean distance, Hamming distance, and other measures to traverse the whole dataset. It solves the classification problem by measuring the distance between different eigenvalues. K-NN has high precision and mature theory and is used to solve the problems of pattern recognition, text classification, precise marketing, and so on.

SVM algorithm uses the Gaussian kernel function to find a hyperplane to divide the samples into two categories with the most significant interval. It is applied to many classification problems, such as spam recognition and face recognition. It is usually used to solve the problems of small samples, nonlinearity, high dimensionality, and so on.

NB algorithm derived from the Bayesian theorem is suitable for independent scenes between features. Based on its prior probability, the theoretical probability is calculated to solve the classification and regression problems. Bayesian classifiers have played a good role in practical applications such as text classification, spam filtering, and medical diagnosis.

DT algorithm uses a tree structure to establish a decision model with the attributes of the data. It is a graphical method of intuitionistic application of probability analysis and is usually used to solve classification and regression issues. The main basic algorithms of DT (ID3, C4.5, and classification and regression tree (CART)) are different in the evaluation criteria of the current tree and are common in small-scale datasets. ID3 selects the attribute with the most significant information gain as the evaluation criterion. C4.5 selects the attribute with the most significant information gain rate as the evaluation criterion. CART is the most commonly used DT construction algorithm; it selects the attribute with the largest Gini gain as the evaluation criterion. Like NB, it is widely used to deal with classification and regression problems.

1.2 Unsupervised learning

Unsupervised learning uses untagged data for training, mainly to solve association analysis and dimension reduction problems, such as text recognition, image segmentation, and malicious traffic attack recognition. *K*-means and Gaussian mixture model (GMM) are the most widely used algorithms.

K-means algorithm classifies similar objects into the same cluster, which can find *K* different clusters. Each

cluster center is calculated by the average value contained in the dataset, mainly to solve the clustering problem.

GMM uses the combination of multiple Gaussian distributions to model the probability distribution of features and uses the maximum expected algorithm for training. It is a widely used clustering algorithm to identify malicious data samples in Internet traffic. For instance, when the distribution of attack samples is similar to that of standard samples, GMM can be used to model at the feature level, and the two types of samples can be distinguished (Bitaab and Hashemi, 2017).

Traditional supervised learning methods such as K-NN require many labeled samples for training. Although unsupervised learning methods do not need prior data, it is challenging to achieve high accuracy.

1.3 Semi-supervised learning

Semi-supervised learning is a mixture of supervised and unsupervised learning algorithms, applied to capture the potential distribution of the whole data in the case of a small number of label samples (Zhou and Belkin, 2014; Gupta and Agrawal, 2020). It introduces unlabeled samples into the model training process to avoid performance degeneration due to insufficient labeled samples. It is used mainly to solve the classification and anomaly detection problems. The most widely used algorithms include principal component analysis (PCA) (Kunal and Dua, 2019) and semi-supervised generative adversarial network (SGAN).

PCA is a commonly used feature extraction method. The primary purpose is to reduce the dimension of features and the training time.

SGAN is a variant of GAN. In SGAN, discriminators are trained in two modes simultaneously: unsupervised and supervised. In the unsupervised mode, authentic and generated images need to be distinguished, just as in traditional GAN. In the supervision mode, an image needs to be classified into several classes, as in the standard neural network classifier. It is used mainly to solve the classification and anomaly detection problems.

1.4 Reinforcement learning

The difference among supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning (RL) is that the main idea of RL learns how to choose the algorithm that can produce the maximum cumulative incentive action through reward or punishment, and the discrete mathematics, stochastic processes, and other mathematical methods are involved, while the others rely on a mass of statistical principles to solve optimization problems.

RL has led to a hot technical trend in ML. RL originates from the optimal control theory in cybernetics, which uses untagged data for training. Through continuous interaction with the outside, the optimal strategy for specific tasks is obtained, and the cumulative expected benefits of tasks are maximized (Buşoniu et al., 2010).

Unlike other ML algorithms, there is no supervisor in the RL model; there is only one reward signal, and the feedback is not generated immediately. Time is of great significance in RL, and the behavior of the agent will affect a series of subsequent data.

RL is divided into two categories centered on the strategy or value function: Policy-based method. This algorithm directly outputs the probability of the following action and selects the action according to the probability. However, the action is not necessarily selected with the highest probability, and usually needs to be considered as a whole and is suitable for discrete and continuous actions. The common algorithms are strategic gradient algorithms. Value-based method. This algorithm outputs the value of the action and the highest value, such as Q-learning and state-action-reward-state-action (SARSA), and is suitable for discrete movements.

The strategic gradient algorithm uses the Monte Carlo method to estimate the gradient strategy, and has good stability. Compared with Q-learning, it can deal with discrete/continuous space problems and achieve more rapid convergence.

Q-learning is an off-line RL algorithm based on the Berman equation, and is one of the core algorithms in RL. The algorithm learns the determination strategy (the target policy takes the maximum function value). It is widely used in the game field and path planning. The further improvement of Q-learning to achieve more accurate decision-making and improve user experience is the current research focus.

SARSA is an online RL algorithm. Both the action strategy and the evaluation strategy of the algorithm are used. The greedy strategy ε–greedy, whose application field is the same as that of Q-learning, is different from Q-learning in that it avoids pitfalls; it is also known as a conservative algorithm, while Q-learning is a greedy or brave algorithm because it selects the maximum Q value every time.

In addition, the actor–critic algorithm combines the policy-based method and the value-based method. The actor takes the action according to the probability, and the critic gives the value according to the action so that the learning process can be accelerated.

To further enhance the application scope and performance, researchers have introduced neural network algorithms into RL.

Deep reinforcement learning (DRL), which integrates deep networks and RL, has become a research hotspot so that the application scope of AI algorithms can be extended to solve the decision-making problem in high-dimensional states and action spaces. The algorithm has low requirements for specific domain knowledge and solves the perceptual decision problem in the high-dimensional discrete action space of complex systems. It has excelled in computer vision, robot control, large real-time strategy games, and other fields.

In 2013, the DeepMind team proposed the deep Q-network (DQN) model combining convolutional neural networks (CNNs) with Q-learning, which gives computers new skills to play Atari games by vision (Volodymyr et al., 2013). In 2015, the team improved the DQN model and achieved better results on the Atari game, which was reported by a cover paper of *Nature* (Volodymyr et al., 2015). DQN uses a deep neural network (DNN) to fit the state-action value function. Compared with the framework of Q-learning, an experience pool, a neural network to calculate the Q value, and temporary freezing of Q_target parameters are included. DQN is widely used in traffic signal control, video games, and robot manipulator path planning. Unfortunately, DQN cannot address continuous problems such as autopilot at present.

Since then, the team has developed various optimized versions of DQN. In 2016, the team combined DL with strategic search methods and launched AlphaGo, beating the champion Sedol Lee. The team selected six DQN extension algorithms and proposed a new variant: Rainbow. Instead of proposing new changes based on previous extensions, the new variant Rainbow integrates the six variants mentioned earlier into a separate agent (Hessel et al., 2018). In 2019, the team launched Alpha Star with multi-agent deep reinforcement learning (MADRL), which reached a human master in Star Craft II games and surpassed 99.8% of human players in the official ranking of Star Craft II.

2 Deep learning

DNNs are also called multi-layer perceptron (MLP) machines. According to the location of different layers, the internal neural network layer can be divided into three categories: input layer, hidden layer, and output layer. The first layer is the input layer, the last layer is the output layer, and the layers in the middle are all hidden layers. There is a complete connection between layers; that is, any neuron in layer i must be connected to any neuron in layer i+1. Because the lower neurons and all the upper neurons can form connections, the problem of parameter number expansion raises. For example, in the pixel image of 1000×1000, there are 10^{12} weights to be trained. The algorithm has been gradually replaced by other models with the increase of the number of parameters.

CNNs include convolution operations, pooling operations, full connection operations, and recognition operations for basic operation units (Fig. S1). Unlike DNNs, CNNs decrease the complexities of the model and the number of weights. CNNs use a convolution function; the convolution layer is connected by some neurons between the two layers, and the weight sharing network structure is closer to that of the biological neural network. CNNs are multi-layer preceptors designed to recognize two-dimensional images, so the structure is highly invariant to other forms of deformation, such as translation, proportional scaling, and tilt. It can well distinguish the data spatial relationship and is used mainly in the fields of image classification (Krizhevsky

et al., 2012), video recognition (Waibel et al., 1990), medical image analysis, and autopilot (Long et al., 2015). The accuracy is higher than that of human beings.

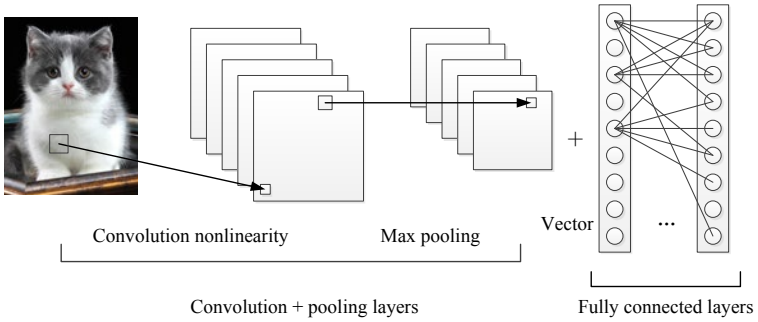

Fig. S1 Convolutional neural network method

By training the weights between the neurons, deep belief networks (DBNs) make the whole neural network generate the training data and are widely applied to identify features, classify data, and generate data.

The recurrent neural networks (RNNs) can simulate the changes in the time series, and the output of the neuron can act directly on itself at the next timestamp (Fig. S2). RNNs have the ability to remember historical data and apply them to prediction. It is widely used in natural language processing (Socher et al., 2011b; Sutskever et al., 2014), speech recognition (Graves et al., 2013), handwritten recognition, and other fields (Socher et al., 2011a) in which the time sequence of samples needs to be analyzed and learned.

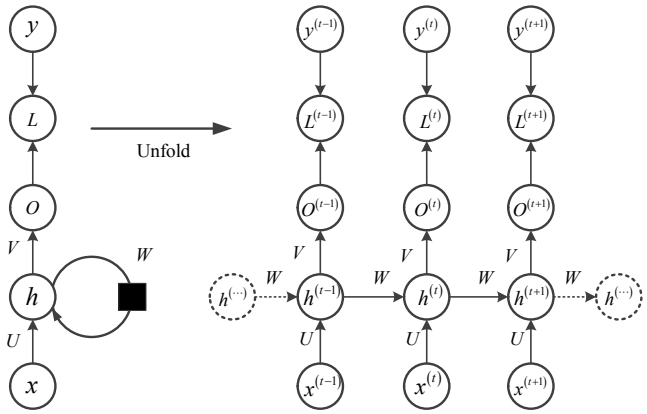

Fig. S2 Recurrent neural network structure

RNNs have some problems in practical applications, such as gradient disappearance and explosion and poor ability of long-distance dependence on information (Bengio et al., 1994), so long short-term memory (LSTM) was introduced (Hochreiter and Schmidhuber, 1997). LSTM can be similar to putting an enhanced component in the cyclic neural network framework. Specifically, it is a module that replaces the small circle of the hidden layer in the cyclic neural network with the LSTM module. LSTM is similar to RNNs in the main structure. The main improvement is to add three gate structures to the hidden layer *h*, and the input gate, output gate, and forget gate are also included. Meanwhile, a cell state is added. Due to LSTM's unique design structure and the advantage of long sequence data modeling, LSTM is fit for modeling context information and forecasting important events. It is widely used in natural language processing, such as translation and speech generation. However, the drawback is that LSTM cannot code backward information when modeling sentences, so bidirectional long short-term memory (Bi-LSTM) is built. This method can avoid the limitations of the LSTM algorithm and perform well in addressing the problem of finer granularity classification.

Autoencoder (AE) includes encoder and decoder. The input data are mapped from the encoder to the decoder. AE can learn automatically without the need for labeled data, and with enough data, it can easily achieve high performance on specific types of input data. It is used mainly for data reduction and feature learning.

The GANs built by Goodfellow et al. (2014) are the most promising and hot DL model for unsupervised learning in complex distribution in recent years (Goodfellow et al., 2016). The model is composed of two basic neural networks. The generator neural network is used to generate content, and the discriminator neural network is used to distinguish the generated content. This significant characteristic is that the model is a powerful sample generation model. To solve the problem of insufficient tagged samples, simulation samples with the same distribution as that of the real samples are generated, thus expanding the scale of tagged samples. GANs are used to generate small sample categories of simulation samples. It is widely used in small sample learning, picture synthesis (Ledig et al., 2017), language processing, and other scenes. GANs have an increasing number of in-depth applications in computer vision, natural language processing, human-computer interaction, and so on, and continue to extend to other fields.

3 Swarm intelligence optimization algorithm and population search algorithm

In 1989, Gerardo Beni and Jing Wang first proposed the concept of swarm intelligence (SI) (Hinchey et al., 2007; Wang et al., 2017). The algorithm simulates the group behavior of insects, fish, herds, and birds. Each group always alters the search direction by learning its own experience and others, also known as the metaheuristic method. The basic entity of SI algorithms is the population of agents, which work together following some rules and there is no centralized controlling authority to dictate the individual behaviors in the environment. The collective local behaviors of agents (animals, insects, birds, etc.) emerge as a global intelligent behavior that is often not known to individuals. These include the particle swarm optimization (PSO) algorithm, ant colony optimization algorithm, artificial bee swarm optimization algorithm, artificial fish swarm optimization algorithm, frog jump optimization algorithm, firefly optimization algorithm, cuckoo search algorithm, bat algorithm, and gray wolf optimizer (GWO). For example, the PSO algorithm optimizes the problem by simulating bird swarm cooperation in the air to find food-seeking behavior; the ant colony algorithm optimizes the problem by simulating the ant colony by planning the shortest path to find the food-seeking process; the artificial bee swarm algorithm optimizes the problem by simulating the position of the food source transmitted by other bees in flight to judge and find the optimal honey source; the artificial fish swarm algorithm optimizes the problem by simulating the aggregation behavior of the fish swarm in the water area and finding the optimal food source; the frog jump algorithm optimizes the problem by simulating the process of frog jumping in the wetland to find the optimal food source (Dash, 2017); the firefly algorithm optimizes the problem by simulating the flash behavior of fireflies using brightness to attract other firefly aggregation population behavior. The SI optimization algorithm generally does not require the continuity and convexity of the objective function and constraint and has strong adaptability to uncertainty in computation. It has been widely applied for many areas, such as function optimization, data mining, and path planning. The PSO algorithm has been employed in various domains such as attack detection, classification and clustering, digital signal processing, control application, design of antenna, data mining, fault diagnosis, robotics, network design, and wind farm design (Mohiuddin et al., 2016; Elbes et al., 2019; Shafiqur et al., 2020).

The group search optimizer algorithm and a series of group search algorithms based on competition mechanisms, the coevolution multi-objective optimization algorithm, and the group search optimizer with multiple producers (GSOMP) are popular optimization algorithms in modern times. The key idea is to optimize the problem by simulating the foraging behavior of the animal population. The members of the population are composed of leaders, followers, and deserters. The leader randomly explores three points in the search space, and when it finds a better place, it directly enjoys the food in this location. Followers will follow suit, searching for better food in the neighborhood where the leader is located. To shun the traditional method in the population, the evaders carries out a free search of space. At the same time, the roles of leaders, followers, and deserters can

change according to the quality of the food they receive. Compared with other optimization algorithms, such as genetic algorithms and PSO, the swarm search algorithm is superior in exploitation ability. It has been comprehensively researched and used to solve problems of load economic dispatching in power systems in recent years, including multi-objective decision-making optimization (Park et al., 2010) and other large-scale complex engineering optimization.

4 Main AI algorithms and applications

The summary of common AI algorithms is given in Table S1.

Table S1 Main AI algorithms and applications

Main AI algorithm	Application
K-nearest neighbor (K-NN)	Text classification, pattern recognition, precision marketing
Support vector machine (SVM)	Spam recognition, face recognition
Decision trees (DT)	Classification and regression problems, such as text classification, spam filtering, and medical diagnosis
Naive Bayesian (NB)	
K-means	Text classification, image segmentation, map coordinate aggregation, malicious traffic attack recognition
Gaussian mixture model (GMM)	Malicious traffic attack recognition
Principal component analysis (PCA)	Intrusion detection
Semi-supervised GAN (SGAN)	Classification, clustering enhancement, anomaly detection
Policy gradient	Discrete/continuous space problems, with rapid convergence
Q-learning	Game field, path planning
SARSA/Actor-Critic	Timing decision problems, game field, path planning
Deep reinforcement learning (DRL)	High-dimensional space decision-making problems, such as computer vision, robot control, and large real-time strategic games
Deep Q-network (DQN)/Rainbow	Traffic signal control, video games, robot manipulator, path planning
Convolutional neural networks (CNNs)	Image classification, video recognition, medical image analysis, autopilot
Deep belief networks (DBNs)	Identifying features, classifying data, generating data
Recurrent neural networks (RNNs)	Natural language processing, speech recognition, handwritten recognition, and other scenes in which the time sequence of samples needs to analyzed and learned
Long short-term memory (LSTM)	Natural language processing, such as translation and speech generation
Bidirectional long short-term memory (Bi-LSTM)	Long sequence data modeling, finer granularity classification
Generative adversarial networks (GANs)	Small sample learning, picture synthesis, language processing, computer vision, natural language processing, human-computer interaction
Traditional optimization algorithm (linear programming; dynamic programming; mountain climbing method; fastest drop method; simulated annealing; genetic algorithm (GA); tabu search)	Linear programming, quadratic programming, convex function optimization
Particle swarm optimization (PSO)	Function optimization, data mining, path planning, continuous optimization problem
Ant colony optimization (ACO)	Small scale problem, combinatorial optimization, discrete optimization problem
Cuckoo search (CS)	Continuous optimization problem, facility layout, clustering problem
Artificial bee colony (ABC)	Image signal processing, feature selection, resource scheduling
Firefly algorithm (FA)	Function optimization, data mining, path planning
Frog leaping optimization (FLO)	Clustering problems, resource network optimization, image segmentation
Fruit fly optimization algorithm (FOA)	Structural engineering design optimization, wireless sensor network layout, resource scheduling
Group search optimizer (GSO)	Multi-objective optimization (the problem of load economic dispatching in a power system), large-scale complex engineering optimization, decision-making problems

References

Bengio Y, Simard P, Frasconi P, 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans Neur Netw*, 5(2):157-166. https://doi.org/10.1109/72.279181

Bitaab M, Hashemi S, 2017. Hybrid intrusion detection: combining decision tree and Gaussian mixture model. Proc 14th Int ISC (Iranian Society of Cryptology) Conf on Information Security and Cryptology, p.8-12. https://doi.org/10.1109/ISCISC.2017.8488375

Buşoniu L, Babuška R, de Schutter B, 2010. Multi-agent reinforcement learning: an overview. In: Srinivasan D, Jain LC (Eds.), Innovations in Multi-agent Systems and Applications. Springer, Heidelberg, p.183-221. https://doi.org/10.1007/978-3-642-14435-6_7

Dash R, 2017. An improved shuffled frog leaping algorithm based evolutionary framework for currency exchange rate prediction. *Phys A Statist Mech Appl*, 486:782-796. https://doi.org/10.1016/J.PHYSA.2017.05.044

Elbes M, Alzubi S, Kanan T, et al., 2019. A survey on particle swarm optimization with emphasis on engineering and network applications. *Evol Intell*, 12(2):113-129. https://doi.org/10.1007/S12065-019-00210-Z

Goodfellow IJ, Pouget-Abadie J, Mirza M, et al., 2014. Generative adversarial nets. Proc 27th Int Conf on Neural Information Processing Systems, p.2672-2680.

Goodfellow IJ, Bengio Y, Courville A, 2016. Deep Learning. MIT Press, Cambridge, USA.

Graves A, Mohamed AR, Hinton G, 2013. Speech recognition with deep recurrent neural networks. Proc IEEE Int Conf on Acoustics, Speech and Signal Processing, p.6645-6649. https://doi.org/10.1109/ICASSP.2013.6638947

Gupta ARB, Agrawal J, 2020. A comprehensive survey on various machine learning methods used for intrusion detection system. IEEE 9th Int Conf on Communication Systems and Network Technologies, p.282-289. https://doi.org/10.1109/CSNT48778.2020.9115764

Hessel M, Modayil J, van Hasselt H, et al., 2018. Rainbow: combining improvements in deep reinforcement learning. Proc AAAI Conf on Artificial Intelligence, 32(1):3215-3222. https://doi.org/10.1609/aaai.v32i1.11796

Hinchey MG, Sterritt R, Rouff C, 2007. Swarms and swarm intellgence. *Computer*, 40(4):111-113. https://doi.org/10.1109/MC.2007.144

Hochreiter S, Schmidhuber J, 1997. Long short-term memory. *Neur Comput*, 9(8):1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Krizhevsky A, Sutskever I, Hinton GE, 2012. ImageNet classification with deep convolutional neural networks. Proc 25th Int Conf on Neural Information Processing Systems, p.1097-1105. https://doi.org/10.1145/3065386

Kunal, Dua M, 2019. Machine learning approach to IDS: a comprehensive review. 3rd Int Conf on Electronics Communication and Aerospace Technology, p.117-121. https://doi.org/10.1109/ICECA.2019.8822120

Ledig C, Theis L, Huszár F, et al., 2017. Photo-realistic single image super-resolution using a generative adversarial network. IEEE Conf on Computer Vision and Pattern Recognition, p.105-114. https://doi.org/10.1109/CVPR.2017.19

Long J, Shelhamer E, Darrell T, 2015. Fully convolutional networks for semantic segmentation. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.3431-3440. https://doi.org/10.1109/CVPR.2015.7298965

Mohiuddin MA, Khan SA, Engelbrecht AP, 2016. Fuzzy particle swarm optimization algorithms for the open shortest path first weight setting problem. *Appl Intell*, 45(3):598-621. https://doi.org/10.1007/s10489-016-0776-0

Park JB, Jeong YW, Shin JR, et al., 2010. Closure to discussion of "An improved particle swarm optimization for nonconvex economic dispatch problems." *IEEE Trans Power Syst*, 25(4):2010-2011. https://doi.org/10.1109/TPWRS.2010.2069890

Shafiqur R, Salman K, Luai MA, 2020. The effect of acceleration coefficients in particle swarm optimization algorithm with application to wind farm layout design. *FME Trans*, 48(4):922-930. https://doi.org/10.5937/fme2004922r

Socher R, Huang EH, Pennington J, et al., 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. Proc 24th Int Conf on Neural Information Processing Systems, p.801-809.

Socher R, Lin CCY, Ng AY, et al., 2011b. Parsing natural scenes and natural language with recursive neural networks. Proc 28th Int Conf on Machine Learning, p.129-136.

Sutskever I, Vinyals O, Le QV, 2014. Sequence to sequence learning with neural networks. Proc 27th Int Conf on Neural Information Processing Systems, p.3104-3112.

Volodymyr M, Koray K, David S, 2013. Playing Atari with deep reinforcement learning. https://doi.org/10.48550/arXiv.1312.5602

Volodymyr M, Koray K, David S, 2015. Human-level control through deep reinforcement learning. *Nature*, 518:529-533. https://doi.org/10.1038/nature14236

Waibel A, Hanazawa T, Hinton G, et al., 1990. Phoneme recognition using time-delay neural networks. In: Waibe A, Lee KF

(Eds.), Readings in Speech Recognition. Elsevier, Amsterdam, the Netherlands, p.393-404. https://doi.org/10.1016/B978-0-08-051584-7.50037-1

Wang H, Jing X, Niu B, 2017. A discrete bacterial algorithm for feature selection in classification of microarray gene expression cancer data. *Knowl-Based Syst*, 126:8-19. https://doi.org/10.1016/j.knosys.2017.04.004

Zhou XY, Belkin M, 2014. Semi-supervised learning. *Acad Press Libr Signal Process*, 1:1239-1269. https://doi.org/10.1016/B978-0-12-396502-8.00022-X