Optimization of Fuzzy Regression Transfer Learning using Genetic Algorithm for Cross-Domain Mapping

MENGCHUN XIE
Department of Electrical and Computer Engineering,
National Institute of Technology, Wakayama College,
77 Noshima, Nada, Gobo, Wakayama, 644-0023,
JAPAN

Abstract: - Artificial intelligence and big data have become widely utilized in industry and thus machine learning has been extensively researched. However, it is challenging to apply existing data-driven methods when the amount of data is insufficient. Therefore, transfer learning, which reuses knowledge acquired from domains with similar data characteristics and tasks, has gained attention for achieving fast and accurate model learning in new domains. Although numerous transfer learning methods have been proposed for classification problems, few have been proposed for regression problems. Moreover, conventional fuzzy regression transfer learning tends to work well only in limited domain environments with extremely limited target data, making its application to real-world data challenging. The present study applies a combination of regression models based on Takagi-Sugeno fuzzy theory and transfers learning to regression problems in domains with incomplete knowledge. We propose two methods, one based on a genetic algorithm and one based on differential evolution combined with a genetic algorithm, for optimizing mapping for input space modification and applying them to real datasets. The results of evaluation experiments demonstrate that the proposed methods have higher efficiency and learning accuracy than those of conventional methods.

Key-Words: - Transfer learning, Fuzzy, Regression, Genetic algorithm, optimization, cross-domain mapping, input space modification, Differential Evolution.

Received: August 14, 2023. Revised: October 29, 2023. Accepted: December 2, 2023. Published: December 31, 2023.

1 Introduction

Data science focuses on the processes and systems involved in extracting knowledge from vast amounts of data. Many machine learning methods assume that training data are collected from a similar feature space or distribution as that for the target domain. When the data distribution changes, most statistical models require the collection of new training data and a complete reconstruction of the model. However, these tasks are costly and sometimes impossible.

Transfer learning, a machine learning method, addresses this issue by leveraging knowledge from models built in a source domain where extensive data can be collected and applying this knowledge to construct models for the target domain. Transfer learning thus offers the potential for high-precision learning with minimal data and a short training time.

Currently, transfer learning is applied to tasks such as the classification of product reviews, spam emails, and web documents and the estimation of Wi-Fi location. Numerous methods have been proposed, particularly for text and image classification, [1], [2]. However, research on transfer learning for regression problems remains limited.

Transfer learning is often combined with deep neural networks; however, domains with insufficient information can introduce uncertainty in predictions. Fuzzy systems, capable of efficiently handling uncertainty, have gained attention in such situations. Traditional regression transfer learning based on the Takagi-Sugeno fuzzy theory is effective only when the target domain has extremely limited training data, [3]. This indicates a highly restricted domain environment, making application to real datasets challenging. Moreover, only particle swarm optimization and differential evolution (DE) methods have been used for optimizing mapping for input space modification, [4], [5].

The present study first verifies the relationships between learning data in different domains using a regression transfer learning method based on the Takagi-Sugeno fuzzy theory for domains with incomplete knowledge. Then, two methods, one based on a genetic algorithm (GA) and one based on DE combined with a GA, are proposed for optimizing mapping for input space modification.

We apply these methods to some real datasets and assess their effectiveness through evaluation experiments.

2 Related Research

2.1 Transfer Learning

Transfer learning is a machine learning approach distinct from conventional machine learning, where training and testing are typically conducted within the same task. In transfer learning, the knowledge gained from learning a given task is utilized for a new task. In other words, transfer learning effectively and efficiently addresses the data scarcity problem by reusing data and learning outcomes for a related problem, [1], [2].

In transfer learning, the source domain (where knowledge is transferred from) and the target domain (where knowledge is transferred to) are defined. The applicability of transfer learning depends on the strength of the relationship between the source and target domains; specifically, when there is a strong correlation between the tasks, data features, and distributions across domains, transfer learning is viable.

Many transfer learning methods have been developed. Instance-based methods attempt to select related source instances for transfer, [6], [7], [8], [9]. One study investigated the transferability of features within deep learning networks and the potential reuse of features across various tasks and domains, [10]. An unsupervised domain adaptation method that employs backpropagation has been developed, [11]. The amalgamation of transfer learning and semi-supervised learning, with model sharing across disparate tasks, has been investigated, [12]. A method for the efficient transfer of data across distinct domains using GAs for multiple-kernel learning has been proposed, [13].

2.2 Fuzzy Regression Transfer Learning

The regression model based on the Takagi-Sugeno fuzzy theory predicts outcomes according to the following fuzzy rules, [3], [14]:

$$\text{If } \boldsymbol{x_k} \text{ is } \boldsymbol{A_i}(\boldsymbol{x_k}, \boldsymbol{v_i}), \textit{then } \mathbf{y_k} \textit{ is } \boldsymbol{L_i}(\boldsymbol{x_k}, \boldsymbol{a_i}) \quad (1)$$

where v_i is a cluster centroid and a_i is the coefficient of a linear function.

This equation is constructed by forming a membership function A_i that represents the degree of membership of the input x_k to each cluster and estimating the parameters of the linear function L_i in the conclusion part of the rule. The fuzzy regression model predicts the output y_k.

Fuzzy regression transfer learning combines the Takagi-Sugeno fuzzy regression model and transfer learning, [3]. Figure 1 shows the relationship between the source and target domains. The characteristics of the target domain should approach those of the source domain because the characteristics of the two domains are similar but never the same. Therefore, the feature quantity of the target domain is made to approach the feature quantity of the source domain by passing the mapping Φ.

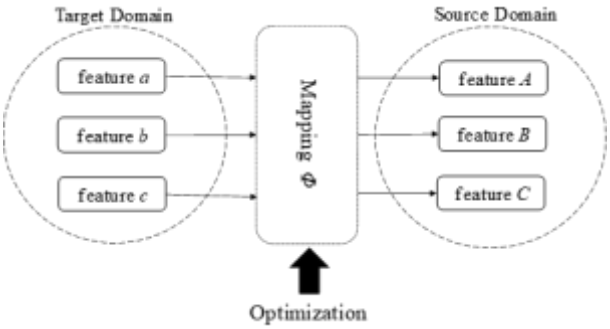

Fig. 1: Relationship between source domain and target domain

3 Fuzzy Regression Transfer Learning Using Genetic Algorithm

In fuzzy regression transfer learning, when applying the fuzzy regression model *M* constructed based on the Takagi-Sugeno fuzzy theory to the target domain, it is necessary to modify the input space.

3.1 Modification of Input Space

In the present study, the input space is modified as shown in Figure 2 to bring the characteristics of the target domain data closer to those of the source domain data.

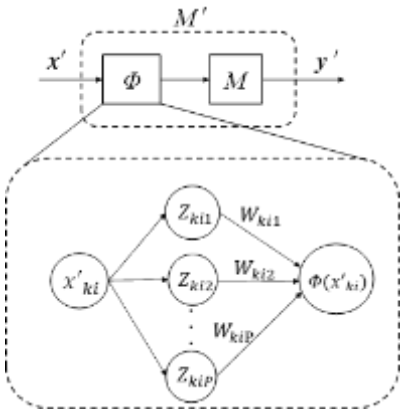

Fig. 2: Modification of input space using nonlinear continuous mapping

Here, M is a model built in the source domain. Model M' is model M combined with the nonlinear continuous mapping Φ.

Modifying the input space specifically means that the number of features in the input data in the target domain is modified so as to be adapted to model M. The input space x' for the target domain is modified by obtaining the mapping $\Phi(x'_k)$ of each input variable in the following equation:

$$\boldsymbol{\Phi}(\boldsymbol{x'_k}) = \begin{bmatrix} \Phi_1(x'_{k1}) \\ \Phi_2(x'_{k2}) \\ \vdots \\ \Phi_n(x'_{kD}) \end{bmatrix} = \begin{bmatrix} \sum_{p=1}^{P} w_{k1p} z_{k1p} \\ \sum_{p=1}^{P} w_{k2p} z_{k2p} \\ \vdots \\ \sum_{p=1}^{P} w_{kDp} z_{kDp} \end{bmatrix} \quad (2)$$

where x'_{ki} is the ith input variable of x'_k. The mapping for each input variable is constructed through a network that is composed of P nodes $(Z_{ki1}, Z_{ki2}, \ldots, Z_{kiP})$ located in the middle layer and a single node located in the output layer. The nodes in the middle layer are a sigmoid function consisting of the two parameters in the following equation:

$$Z_{kip} = \frac{1}{1 + e^{-\alpha_{ip}(x'_{ki} - \beta_{ip})}} \quad (3)$$

$$i = 1,2,\ldots,D, \quad p = 1,2,\ldots,P, \qquad \alpha_{ip} > 0$$

When modifying the input space, it is important to optimize the parameters α and β in the sigmoid function and the weight W. In the present study, we adopt a GA to optimize these parameters and evaluate the improvement of the resulting limited domain environment.

3.2 Optimization of Mapping using Genetic Algorithm

GAs are optimization algorithms inspired by natural selection and genetics. They belong to the broader category of evolutionary algorithms and have been widely used to find approximate solutions to optimization and search problems. GAs are based on the principles of evolution, including selection, crossover (recombination), and mutation, [15], [16], [17], [18], [19], [20].

In this study, we apply a GA to optimize the parameters for the mapping Φ. The representation of individuals in the GA uses a matrix whose size is determined by the number of attributes and the number of nodes. The representation of an individual for parameter α is shown in Figure 3. Similar representations are used for β and W. Here, since we use a real-valued GA, each element in the matrix is a random real number within a specified range.

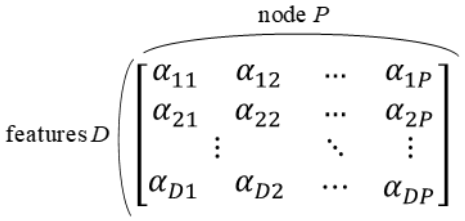

Fig. 3: Representation of individual for parameter α in genetic algorithm

The fitness function uses the mean squared error (MSE):

$$MSE = \frac{1}{N} \sum_{k=1}^{N} (\boldsymbol{y_k} - \boldsymbol{y_k}')^2 \quad (4)$$

where y_k' is a predicted value based on the input space modified using the parameters of the individual in the current generation and y_k is a target value.

Finally, for genetic manipulation, single-point crossover is performed, as shown in Figure 4. Each element of the matrix is stochastically mutated one by one. The individuals are then compared and the one with the best fitness is selected. Additional individuals are selected by roulette selection. These individuals make up the next generation.

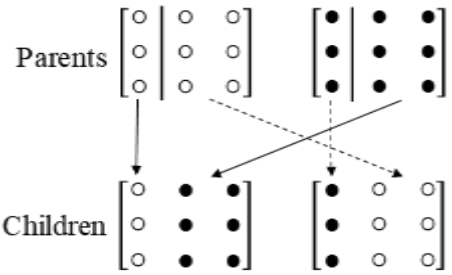

Fig. 4: Diagram of single-point crossover

3.3 Optimization of Mapping Using Differential Evolution Combined with Genetic Algorithm

In the optimization of mapping using a GA, it is considered that in a real-valued GA, the fitness of the initial individuals has a significant impact on the final accuracy. Increasing the population size, mutation probability, and number of generations in the GA can enhance the coverage of the search space; however, this may also increase the computation time.

DE is a population-based optimization algorithm that falls under evolutionary algorithms. It iteratively evolves a population of candidate

solutions through processes such as mutation, crossover, and selection. DE is known for its simplicity and effectiveness in solving optimization problems across various domains, [21], [22], [23], [24].

In this study, we propose a mapping optimization method that combines DE and GA, denoted as DE-GA. Initially, optimization is conducted using DE. Then, further optimization is conducted by applying a GA to the solution set obtained from DE. In this approach, the initial optimization using DE is performed to enhance the fitness of the initial individuals for the real-valued GA. It is expected that combining DE with a GA will improve the prediction accuracy (even though a smaller solution set and fewer generations are used compared to those for GA alone) and reduce the computation time.

3.4 Proposed Model and Evaluation Metric

In this study, we propose five patterns for model M' in the target domain, where the model is transferred from the constructed model in the source domain.

M'_T: model built with only target data (Φ=1)
M'_S: model built with only source data (Φ=1)
M'_{DE}: Φ and M optimized using DE
M'_{GA}: Φ and M optimized using GA
M'_{DE_GA}: Φ and M optimized using DE-GA

The evaluation metric used in the validation experiments on a real dataset is the root-mean-square error (RMSE) between the target values y and the predicted values y', as defined in (5). In addition, the standard deviation (SD) of the prediction errors, defined in (6), is calculated to assess the variability of the predictions. $\bar{y'}$ represents the mean value of the predicted values. Furthermore, to assess the efficiency of the models, a comparison of the computation time from model learning to prediction is conducted for M'_{DE}, M'_{GA}, and M'_{DE_GA}.

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_k - y_k')^2} \quad (5)$$

$$SD = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(y_k' - \bar{y'}\right)^2} \quad (6)$$

4 Experiment and Results

Validation experiments were conducted using the proposed five models on real datasets. The five models were applied to various tasks for real datasets. The proposed method was evaluated in terms of prediction accuracy and computation time. Cross-validation was also performed. The experimental conditions are shown in Table 1.

Table 1. Experimental conditions

Fuzzy Regression		**DE**	
Cluster *C*	*	Solution Group	200
Fuzzy Degree *m*	*	Generation	200
Mapping		Scaling Factor *F*	*
Node	3	Crossover Rate *CR*	*
GA		**DE-GA**	
Individuals	50	Solution Group (Individuals)	50
Generation	1000	DE generation	100
Elite Rate	0.2	GA generation	100
Mutation Rate	0.05	No other changes	

Since M'_S did not require five-fold cross-validation, all target data were used as the test data for this model. In addition, the parameters were set to values believed to be sufficient for the learning of each model based on preliminary experiments. For parameters *C* and *m* for the fuzzy regression model and parameters *F* and *CR* for DE, the values were set to accommodate the given dataset.

4.1 Experiment 1: Boston Housing Dataset

In this experiment, the Boston housing dataset, an open dataset for regression problems, was utilized. For the transfer learning process, the input space was constructed using two attributes from this dataset, namely ROOM (average number of rooms in each neighborhood) and DISTANCE (distance from each neighborhood to five employment centers in Boston).

The target variable was the median housing price in each neighborhood. For the transfer learning task, instances with the attribute TAX (property tax rate per $10,000) below 600 were considered as the source data (one dataset with 370 instances) and those with TAX equal to or above 600 were considered as the target data (three datasets with 30, 60, and 90 instances, respectively). The input space for the two domains is shown in Figure 5. As shown, the two domains have different data distributions.

Table 2. Results for Boston housing dataset

Target	M'_T	M'_S	M'_{DE}	M'_{GA}	M'_{DE_GA}
30	104.32		8.44	**5.38**	9.75
	50.24		6.65	12.69	**4.71**
	37.80		**26.33**	32.47	33.61
	32.50		34.54	18.61	**17.51**
	24.85		39.33	**20.00**	35.51
Mean	49.94±28.42	**16.71**	23.06±13.34	17.83±8.96	20.22±12.41
60	13.32		9.20	**6.52**	7.21
	27.63		24.47	**21.94**	22.40
	11.80		4.83	6.70	**4.05**
	5.64		7.46	**4.96**	10.39
	9.58		**6.42**	9.18	6.77
Mean	13.59±7.48	13.62	10.48±7.14	**9.86±6.19**	13.37±5.27
90	24.71		25.68	21.08	**19.53**
	17.21		12.86	12.58	**11.33**
	7.35		9.13	**7.86**	9.05
	5.32		4.85	5.57	**4.14**
	5.12		4.78	5.27	**4.43**
Mean	11.94±7.78	12.22	11.46±7.72	10.47±5.92	**9.70±5.63**
Time Ratio			1.00	4.66	0.56

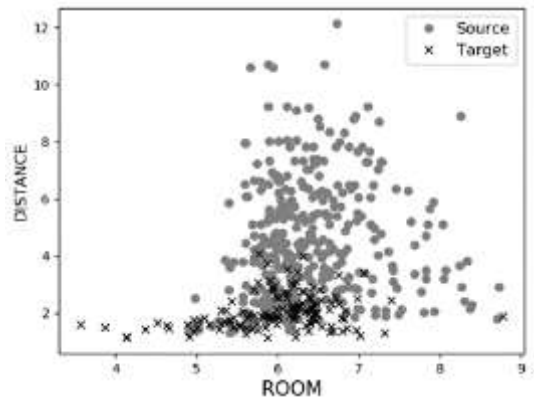

Fig. 5: Input space for source and target domains in Experiment 1

Table 2 shows the results of five-fold cross-validation for each trial, including the RMSE, average, and SD for various target data sizes and the ratio of the average computation time. The parameters used were C = 6, m = 1.8, F = 0.5, and CR = 0.9. As shown in the table, $M'_{GA}$and M'_{DE_GA} often have smaller errors than those for the other models, indicating higher prediction accuracy. However, for the target dataset with 30 instances, bothM'_{GA} and M'_{DE_GA} are inferior to M'_S.

4.2 Experiment 2: Diabetes Dataset

In this experiment, we utilized an open dataset related to diabetes for regression analysis. The input space was constructed using three attributes, namely BMI (body mass index indicating obesity), BP (blood pressure), and GLU (blood glucose level). The target variable was the progression of diabetes after 1 year.

Here, instances with ages below 60 were used as source data (one dataset with 339 instances) and those with ages equal to or above 60 were used as target data (three datasets with 30, 60, and 90 instances, respectively). The input space for the two domains is shown in Figure 6. In contrast to Experiment 1, the two domains have similar data distributions.

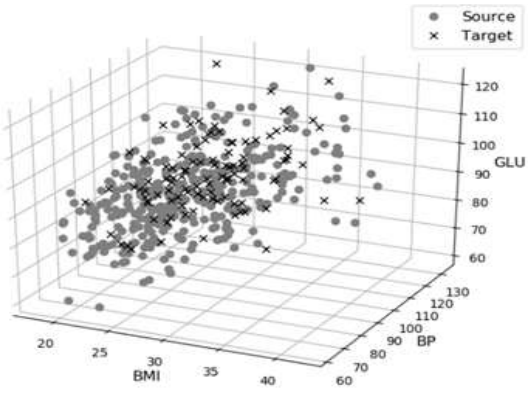

Fig. 6: Input space for source and target domains in Experiment 2

Table 3 shows the results of five-fold cross-validation for each trial. The parameters used were C = 6, m = 1.8, F = 0.5, and CR = 0.7. A shown in the table, depending on the data split, there are cases where M'_{GA} and M'_{DE_GA} have smaller errors than those for the other models. However, on average, regardless of the number of instances in the target dataset, M'_S has the best prediction accuracy in all cases.

Table 3. Results for diabetes dataset

Target	M'_T	M'_S	M'_{DE}	M'_{GA}	M'_{DE_GA}
30	4376.53		62.83	84.05	**52.41**
	678.69		149.21	128.89	**114.99**
	193.28		62.55	**42.67**	47.36
	5554.48		89.92	76.84	**75.85**
	91.44		87.94	**48.28**	51.75
Mean	2178.88±2314.08	**60.60**	90.49±31.63	76.15±30.79	68.47±25.30
60	73.64		66.10	63.60	**62.93**
	64.41		66.77	**61.50**	65.95
	56.75		**65.61**	65.63	73.66
	70.60		64.17	67.98	**54.82**
	74.97		63.10	59.24	**52.68**
Mean	68.08±6.73	**56.63**	65.15±1.33	63.59±3.06	62.01±7.63
90	68.49		72.35	57.32	**55.19**
	54.30		**61.14**	69.75	67.02
	54.44		62.46	64.66	**54.76**
	47.18		44.61	55.29	**42.57**
	64.67		**49.48**	59.45	54.22
Mean	57.82±7.72	**53.61**	58.01±9.88	61.29±5.26	54.75±7.74
Time Ratio			1.00	3.60	0.48

Table 4. Results for cancer dataset

Target	M'_T	M'_S	M'_{DE}	M'_{GA}	M'_{DE_GA}
30	50.42		27.58	29.54	**25.53**
	60.93		44.51	**38.29**	39.71
	2362.65		71.81	**63.92**	100.46
	2208.44		**27.60**	42.51	40.22
	208.68		46.50	46.06	**44.37**
Mean	978.22±1070.00	**38.32**	43.60±16.24	44.06±11.36	50.06±25.99
60	33.38		37.10	36.22	**34.87**
	31.84		**33.11**	34.61	35.92
	35.80		40.33	**39.39**	34.37
	503.46		26.89	**23.15**	26.33
	48.28		**23.07**	35.52	32.59
Mean	130.55±186.54	36.15	**32.10±6.36**	33.78±5.55	32.81±3.42
90	39.58		36.71	32.96	**32.02**
	37.47		36.54	36.75	**35.01**
	34.56		**28.12**	32.62	28.18
	31.91		29.18	36.90	**28.04**
	36.59		29.28	28.99	**28.08**
Mean	36.02±2.61	34.67	31.97±3.82	33.64±2.95	**30.27±2.81**
Time Ratio			1.00	3.36	0.42

4.3 Experiment 3: Cancer Dataset

In this experiment, a dataset related to cancer incidence rates by county in the United States was utilized. The input space was constructed using two attributes, namely IR (cancer diagnosis rate per capita for each county) and INCOME (median income of residents in each county). The target variable was the per capita cancer mortality rate for each county. Instances with a poverty rate of less than 30% were used as source data (one dataset with 2930 instances) and those with a poverty rate equal to or above 30% were used as target data (three datasets with 30, 60, and 90 instances, respectively). The input space for the two domains is shown in Figure 7. The two domains have different data distributions.

Table 4 shows the results of five-fold cross-validation for each trial. The parameters were set as $C = 6$, $m = 1.4$, $F = 0.5$, and $CR = 0.7$. As shown in the table, there are cases where M'_{DE_GA} has smaller errors than those for the other models. In addition, M'_{DE_GA} often has the smallest errors (highest prediction accuracy). However, for the target dataset with 30 instances, M'_S has the best results.

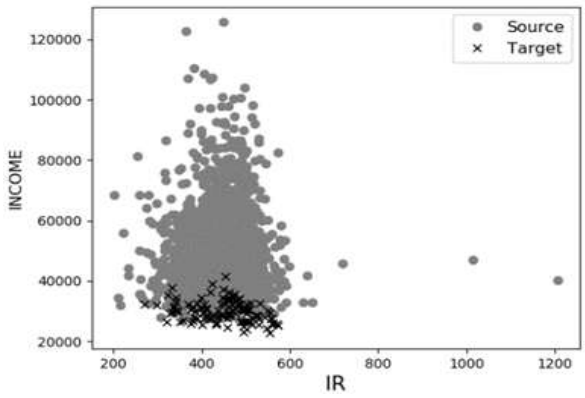

Fig. 7: Input space for source and target domains in Experiment 3

The results for the three experiments confirm that fuzzy regression transfer learning is effective for real datasets. However, in Experiment 2, for which the input space distributions of the source and target domains overlapped, the modification of the input space adversely affected the learning process. In addition, due to the large variability in input data in a real dataset, the method performed well even with a large number of instances in the target dataset. On the other hand, for a target dataset with 30 instances, learning was insufficient. These observations suggest that depending on the conditions (e.g., data distribution and the number of instances), negative transfer through mapping may occur.

Furthermore, a comparison of the computation times indicates that M'_{GA} was approximately 3.5 to 4.5 times faster than M'_{DE} in all experiments. On the other hand, M'_{DE_GA} was about half as fast as M'_{DE}. Considering both prediction accuracy and computation time, it can be concluded that M'_{DE_GA} is the best model, verifying the efficient optimization of parameters with DE-GA.

Although the proposed method requires certain conditions (e.g., the number and distribution of data instances in the source and target domains) to be met, it is applicable to various real-world regression problems.

5 Conclusion

In the present study, we applied fuzzy regression transfer learning to regression problems in domains with incomplete knowledge. First, for the target task, we found that the impact of the ratio of data between the source and target domains was negligible, but the distribution of data between the two domains had a significant effect. Incorporating a GA for optimizing the mapping for input space modification improved prediction accuracy and reduced computation time. The effectiveness of the proposed method was confirmed on real-world datasets. The proposed transfer learning method can only be applied when the dimensions and attributes of the input space are the same.

The following challenges will be considered in future studies:

(1) Handling datasets with different dimensions and attributes. The current method is applicable only when the dimensions and attributes of the input space are the same. Methods that can handle datasets with different dimensions and attributes are required.

(2) Experiments using more diverse domains. Although the experiments in this study were conducted on real-world datasets, the effectiveness of the proposed method should be further verified using more diverse domains and datasets.

(3) Automated parameter tuning. Some parameters in the current method are manually set. Methods for the automatic adjustment of these parameters should be developed.

(4) Enhancing robustness to noise and missing data. Real-world datasets often contain noise and/or missing data. Improving the robustness of the proposed method in such situations is important.

References:

[1] S. J. Pan and Q. Yang, A Survey on Transfer Learning, *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, 1345-1359, October 2010.

[2] Weiss, K., Khoshgoftaar, T.M. & Wang, D. A survey of transfer learning. *J. Big Data*, 3, no. 9, March 2016.

[3] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, J. Lu, Fuzzy Regression Transfer Learning in Takagi-Sugeno Fuzzy Models, *IEEE Transactions on Fuzzy Systems*, vol.25, no.6 December 2017.

[4] Ahmed G. Gad, Particle swarm optimization algorithm and its application: a systematic review, *Archives of Computational Methods in Engineering, Springer Link*, Vol. 29, 2531-2561, April 2022.

[5] F. zhao, C. Wang, H. Liu, Differential evolution - based transfer rough clustering algorithm, Vol. 9, 5033-5047, February 2023

[6] Y. Wang, J. Zhai, Y. Li, K. Chen, H. Xue, Transfer learning with partial related "instance-feature" knowledge, *Neurocomputing*, 310, 115-124, 2018.

[7] J. Huang, A. J. Smola , A. Gretton, K.M. Brorgwardt, B. Scholkopf, Correcting sample

selection bias by unlabeled data, Proceedings of the *International Conference on Neural Information Processing Systems*, 601-608, 2006.

[8] M.E. Taylor, P. Stone Transfer learning for reinforcement learning domain: a survey, *J. Math. Learn*, *Res.*, no. 10 1633-1685, 2009.

[9] R.E. Schapire, Y. Freund, P. Bartlett, W.S. Lee, Boosting the margin: a new explanation for the effectiveness of voting methods, *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, p.322-330, 1997.

[10] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, vol. 27, 3320-3328, 2014

[11] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, p.1180-1189, 2015.

[12] S. Ruder, *An overview of multi-task learning in deep neural networks*. arXiv preprint arXiv:1706.05098, 2017

[13] W. Zhang, Z. Li, Y. Chen, Domain transfer multiple kernel learning using genetically evolved kernels. *Neurocomputing*, 171, 303-312, 2016.

[14] T. Takagi, M. Sugeno, Fuzzy Identification of Systems and Its Applications to Modeling and Control, *IEEE Transactions on System, Man, and Cybernetics*, vol. SMC-15, no.1, 1985.

[15] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975

[16] A. E. Eiben, J. E. Smith, *Introduction to Evolutionary Computing*, Springer Link, 2015, https://doi.org/10.1007/978-3-662-44874-8.

[17] M. Wooldridge, *Intelligent Agents, Multiagent Systems: A modern Approach to Distributed Artifical Intelligence*, edited by Gerhard Weiss, The MIT Press, 2000

[18] M. Xie, H. Ogura, T. Odaka, J. Nishino, "Application of Genetic Algorithm to Inter-correlated Nonlinear Knapsack Problem", *1996 International Symposium on Nonlinear Theory and its Applications*, p.145-148, 1996

[19] M.C. Xie, Cooperative Behavior Rule Acquisition for Multi-Agent Systems Using a Genetic Algorithm, *Proceedings of the IASTED International Conference on Advances in Computer Science and Technology*, p.124-128 ,2006

[20] S. M. Elsayed, R. A Sarker, and D. L. Essam, A new genetic algorithm for solving optimization problems, *Engineering Application of Artificial Intelligence*, Vol. 27, p.57-69, 2014

[21] R. Storn, K. Price, Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, vol.11, no. 4, 341-359, 1997

[22] S. Das, P.N. Suganthan, Differential Evolution: A Survey of the State-of-the-Art. *IEEE Transactions on Evolutionary Computation*, vol. 15, no.1, p.4-31, 2011.

[23] A.K. Qin, V. Huang, P.N. Suganthan, Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization. *IEEE Transactions on Evolutionary Computation*, vol. 13, no.2, p.398-417, 2009.

[24] T. Eltaeib, A. Mahmood, Differential Evolution: A Survey and Analysis, *Applied Sciences*, vol.8, no.10, 2018, https://doi.org/10.3390/app8101945.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.