Novel Application of Mutual Information in Transfer Learning for Genetic Programming

Yilin Liu
Department of Electronic and Electrical Engineering
Brunel University
London, UK
Yilin.Liu@brunel.ac.uk

Gareth Taylor
Department of Electronic and Electrical Engineering
Brunel University
London, UK
Gareth.Taylor@brunel.ac.uk

Zhengwen Huang
Department of Electronic and Electrical Engineering
Brunel University
London, UK
Zhengwen.Huang@brunel.ac.uk

ABSTRACT

Genetic Programming (GP) faces two major challenges: the inability to use knowledge from past problems, as each run is independent, and the difficulty in identifying building blocks in the early stages of evolution. These limitations result in high computational costs and slower convergence. Transfer learning can provide a solution by leveraging past experiences to enhance performance in GP. A mutual-information-based transfer learning method is proposed in this paper to identify and transfer beneficial knowledge fragments. The method is evaluated on ten polynomial and trigonometric symbolic regression problems from previous literature and compared with standard GP and SubTree50 as state-of-the-art methods. Results of the above experiment demonstrate improved or comparable performance of the proposed method in terms of accuracy, statistical significance, and generalization. The contribution of this paper includes: a mutual-information-based technique for identifying and transferring knowledge fragments. Results highlight the potential of mutual information to address key challenges in GP effectively.

CCS CONCEPTS

• **Computing methodologies→Genetic programming**;

KEYWORDS

Genetic programming, transfer learning, mutual information

ACM Reference format:
Yilin Liu, Gareth Taylor, and Zhengwen Huang. 2025. Novel Application of Mutual Information in Transfer Learning for Genetic Programming. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '25 Companion), July 14-18, 2025, ACM, Málaga, Spain, 4 pages.* https://doi.org/10.1145/3712255.3726607

GECCO '25 Companion, July 14–18, 2025, Malaga, Spain

ACM ISBN 979-8-4007-1464-1/2025/07. $15.00
https://doi.org/10.1145/3712255.3726607

1 Introduction

Genetic Programming (GP) is an evolutionary computation paradigm that is designed to evolve computer programs to satisfy the objectives of users [1]. GP has achieved success in various domains like symbolic regression, but two challenges remain. Firstly, unlike humans, who learn from experience, GP must re-learn problems even with minor changes, leading to high time and computational costs. Secondly, how to identify Building Blocks in the early adaptation stage is still an open problem in GP, many theoretical studies in GP have agreed that high contributive segments of individuals, which are known as Building Blocks, can significantly promote GP to search the global optima and boot the searching speed if they can be found in the early stage [2], yet effective identification methods are still lacking. Transfer learning can help close the above research gaps. It leverages knowledge from past tasks to tackle similar ones, guiding optimization, and accelerating convergence [3]. In GP, many works related to transfer learning make efforts to identify and transfer key knowledge fragments carrying on various forms from the source domain to the target domain [4-6]. The key idea is that identifying meaningful fragments from solved problems and encouraging their early adaptation helps spread them in the population, enhancing GP performance in the target domain. The previous works also highlight that one of the challenges in this field lies in identifying valuable to-be-transferred knowledge for the target domain. Most works have explored various ways to extract useful components from the source domain. However, they often face issues such as lack of significance [7], convergence to local optima, and high dependence on specific source problems [6]. This suggests the need to explore alternative indicators to more effectively identify and transfer meaningful knowledge from the source to the target domain.

In this paper, we explore an approach for effectively identifying contributing components using a statistical index called Mutual Information (MI). In this paper, Building Blocks are defined as the program pieces that contain the information about the final solution. We propose a novel approach for transferring essential Building Blocks into GP, based on the MI between the data generated by these Building Blocks and the target problem.

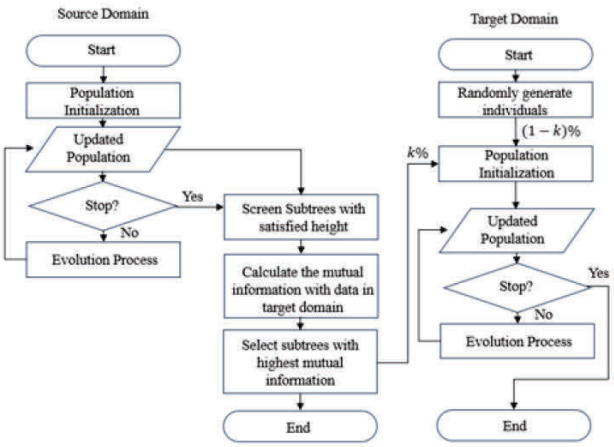

Figure 1: The proposed transfer learning method.

MI rooted in information theory, measures data dependence and remains invariant under smooth, invertible transformations of random variables [8]. Leveraging this invariance, potential Building Blocks that are beneficial for the target problem can be identified within the source problem [9]. Our experiment has shown that, compared with state-of-the-art (SOTA), our transferring method can be effective with more significant and robust performance.

3 Proposed Method

3.1 Overall Framework

The proposed method follows the framework summarized in the previous chapter. As demonstrated in Figure 1, the details of proposed transfer learning are described in the following steps:

1. Combining the final population and elite pool in the source domain, all subtrees are extracted from the combined population.
2. For each subtree do:
   a. Examine whether the subtree height is within the required range. If it is, proceed to the next step; if not, return to step 1.
   b. Calculate the MI between the data generated by subtree and target data.
   c. Check if the calculated MI value has appeared before: If it has, compare the height of this subtree to the previously recorded one. If this subtree's height is smaller, update the record with this subtree; otherwise, return to step 1. If the MI value is new, go to step 2d.
   d. Record the subtree along with its MI value and height.
3. Rank and select the top n subtrees in the record according to their MI scores.
4. In the target domain, the population initializes as follows: $k\%$ of the initial population will use those subtrees, and each subtree will generate $k\%N/n$ individuals in the initial population, where N is the size of the population. Then, randomly generate $(1-k)\%N$ individuals in the initial population of the target domain.

3.2 What to Transfer: Subtrees with High Mutual Information from Source Domain

In the proposed method, the MI identifier screens and extracts source domain subtrees, acting as a subtree extractor that prioritizes their relevance to target data. Theoretically, the requirements for source tasks are they should share or be a subset of the target domain's primitives and provide sufficient high-mutual-information subtrees deemed "similar" or "correlated" based on prior user knowledge.

Studies have reported that the entire population transfer is only transferring the sample point to the target domain [7], thus, in the proposed method, the target of transfer learning is the subtrees.

The MI between datasets generated by a tree $\hat{Y}$ and the target dataset Y can be expressed as [9]:

$$I(\hat{Y};Y)=\sum_{\hat{y}\in\hat{Y}}\sum_{y\in Y}p(\hat{y},y)\log\left(\frac{p(\hat{y},y)}{p(\hat{y})p(y)}\right) \tag{1}$$

where $\hat{y}\in\hat{Y}$ and $y\in Y$ are separately the predicted data generated by individual and target data, $p(y)$ and $p(\hat{y})$ are accordingly their probability distribution, $p(\hat{y},y)$ is their joint probability distribution. Different ways of MI normalization can emphasize the description the dependency between the individual and the target focusing on different perspectives as below [10]:

$$I_N(\hat{Y};Y)=\frac{I(\hat{Y};Y)}{H(Y)}=1-\frac{H(Y|\hat{Y})}{H(Y)} \tag{2}$$

$$I_N(\hat{Y};Y)=\frac{I(\hat{Y};Y)}{H(\hat{Y};Y)} \tag{3}$$

The first normalization divides MI by the entropy of the target data, emphasizing the proportion of information from a subtree that contributes specifically to the target. In the second normalization method, MI is divided by joint information entropy, considering both specificity and redundancy.

Based on the above definitions of MI, the global optimal model, which describes the distribution of target data, has the lowest error and should have the maximal MI with the target data. According to the invariance of MI, the Building Blocks with the high MI contain most of the information from global optima, it can be transformed into the global optimal by crossover and mutation within the smooth invertible structural change [11]. On the other hand, more overlap provides individuals with more relevant information about the target, which helps to reduce the individual's error, making it more distinguishable among populations. As a result, the error-based fitness of the individuals improves, reflecting its closer alignment with the target solution.

Besides, subtree selection considers that large transferred trees may cause bloat, while small subtrees are less prone to disruption by genetic operators [5]. In addition, the ability of generalization will be reduced when transferring trees with the same MI values. Thus, the to-be-transferred subtrees are always the subtree with the smallest height among an MI value.

3.3 How to Transfer: Initial Population Biasing

The proposed method utilizes initial population biasing. Although, misidentifying Building Blocks may lead to the loss of the initial advantage, correctly identifying and extracting Building Blocks

could greatly benefit the process. According to the Building Blocks hypothesis [2], accurate Building Blocks tend to proliferate exponentially within the population. Additionally, this technique avoids increasing the number of primitives, which could lead to higher dimensionality. Instead, it specifies initial samples within the existing search space to let the evolution notice the subspaces with high mutual information but could take very small portions. This approach is less likely to disrupt the dimensions of the search space, preserving its distribution if it is already well-formed. Notably, when optimizing an evolutionary algorithm for a specific application class, biasing the search towards regions with high solution density is a promising strategy [12]. In transfer learning, the target domain can be seen as such a class, making this initialization suitable. Meanwhile, differences between source and target problems may introduce irrelevant or harmful elements in transferred knowledge. The evolution process helps adapt and refine these, focusing on beneficial components.

Table 1: Parameter Configurations in the Experiment.

Parameter	Configurations
Population Size	500
Generations	50
Initialization	Half and Half, Depth of tree range = [0,6]
Selection	Tournament, Tournament size=3
Fitness Function	Mean Absolute Error on all fitness cases
Crossover	Standard crossover, Crossover rate = 0.9
Mutation	Standard mutation, Mutation rate = 0.1, Depth of mutation tree range = [0,5]
Bloat Control	*StaticLimit* [1], Max height = 15
Elitism	On, Hall-of-fame size = 50
Train Data	100 uniformly sampled points in [-1,1], as [4]
Test Data	100 uniformly sampled points in [-3,3], as [4]
Test Error	Mean Absolute Error in all test cases.

4 Experiment Design

The proposed method uses 10 univariable symbolic regression problems from [4] as trial problems. The standard GP and *SubTree50* [4] are compared with the proposed transfer learning methods as baselines. *SubTree50* transfers a random subtree from each individual in the top 50% of the source domain's last generation and de-abstracts them as individuals as part of the initial population in the target domain. Each algorithm is independently run 100 times per regression problem, with the best individual in the final population taken as the solution. For transfer learning, each run also includes re-executing the source problem and subtree extraction before solving the target. The experiment parameter settings for all the algorithms used are listed in Table 1. For each problem, the terminal set consists of a single element: $\{x\}$. The function set includes: $\{+, -, \times, \div, sin, cos, exp, log_e x\}$, where division and logarithm function are protected versions: if the calculation is invalid, it returns 1. For transfer learning algorithms, both the source and target domains in the transfer learning algorithms share the same general configurations. Poly0 and Trig0 are used separately as source domain problems for polynomial and trigonometric symbolic regression tasks. In the proposed method, only subtrees with heights in the range of [1,4] are considered for MI calculation. The top 10 subtrees with the highest MI relative to the target domain data are selected. These subtrees are then utilized to equally generate individuals, forming 50% of the initial population in the target domain. MI, a discrete domain index, is calculated by discretizing each dataset into 10 uniform bins.

All the experiments are implemented by DEAP [13] 1.4.1 on Python 3.12. If an individual's fitness or test error compilation fails due to values that exceed Python's limit during the calculations, the individual is assigned an infinite fitness/test error to indicate the overflow/underflow. The experiments were conducted in batches at different times and performed under consistent hardware and software environments to ensure fairness and reliability.

5 Result and Analysis

5.1 Performance on Regression Tasks

Table 2 shows the results for 10 symbolic regression tasks. "no-norm MI", "target-norm MI", and "joint-norm MI" refer to no normalization, normalization by target entropy, and by joint entropy, respectively. "+", "=" and "-" indicate the method is better than, similar, or worse than Standard GP or *Subtree50*. The double-sided Wilcoxon test assesses significance between Standard GP and transfer learning methods, with significant results ($PVal < 0.05$) marked by a down slash. About 10% of final individuals marked with infinity test error are excluded when doing the Wilcoxon test.

The proposed methods outperform *SubTree50* in most symbolic regression tasks, significantly improving best fitness and median test error. On the means of best fitness for training, the proposed methods outperform standard GP and *SubTree50* in most cases, suggesting that MI-based methods effectively leverage transferable subtrees to enhance fitness performance. In terms of the median test error, the proposed method can reduce the error in most cases. However, in some tasks, especially in Poly5, the performance is not as good as *SubTree50* suggesting the potential overfitting. Secondly, the proposed method surpasses *SubTree50* in the significance level test, demonstrating its ability to maintain the advantage of subtrees identified via MI. These subtrees may better align with Building Blocks. However, while the method succeeded in some problems, it failed in others, especially on the test set. The low significance or overfitting in some tasks may stem from MI's strong data distribution descriptor increasing overfitting risk, or initial influences evolution only at the start, which influences evolution only at the start. Thirdly, normalization by joint entropy consistently achieves competitive or best performance, particularly in training fitness and trigonometric test errors, compared to no normalization or target-data normalization. Those observations are consistent with the theoretical analysis mentioned in [10].

Table 2: The performance of proposed methods on 10 benchmark tasks, compared with SOTAs.

Problems	Poly-1	Poly-2	Poly-3	Poly-4	Poly-5	Trig-1	Trig-2	Trig-3	Trig-4	Trig-5
Mean of Best Fitness										
Standard GP	0.0180	0.0204	0.0379	0.0506	0.0622	0.0365	0.0088	0.0159	0.0156	0.0947
SubTree50	0.0139	0.0138	0.0258	0.0419	0.0563	0.0230	0.0057	0.0177	0.0105	0.0933
no-norm MI	0.0139 (+,=)	0.0129 (+,+)	0.0238 (+,+)	0.0418 (+,+)	0.0516 (+,+)	0.0141 (+,+)	0.0052 (+,+)	0.0103 (+,+)	0.0043 (+,+)	0.0470 (+,+)
target-norm MI	0.0127 (+,+)	0.0128 (+,+)	0.0254 (+,+)	0.0377 (+,+)	0.0449 (+,+)	0.0172 (+,+)	0.0066 (+,-)	0.0126 (+,+)	0.0075 (+,+)	0.0485 (+,+)
joint norm MI	0.0117 (+,+)	0.0129 (+,+)	0.0222 (+,+)	0.0488 (+,-)	0.0540 (+,+)	0.0129 (+,+)	0.0057 (+,=)	0.0089 (+,+)	0.0070 (+,+)	0.0627 (+,+)
Median of Test Error										
Standard GP	126.965	47.900	91.983	156.348	5955826.581	2.892	4.973	6.426	8.910E-17	1.200
SubTree50	116.750	38.099	90.793	155.527	5169346.282	2.854	7.912	6.172	8.351E-17	1.286
no-norm MI	125.207 (+,-)	26.601 (+,+)	82.773 (+,+)	132.057 (+,+)	23010016.817 (-,-)	2.614 (+,+)	2.766 (+,+)	4.077 (+,+)	8.042E-17 (+,+)	1.137 (+,+)
target-norm MI	126.428 (+,-)	43.425 (+,-)	67.320 (+,+)	139.002 (+,+)	23010016.840 (-,-)	2.732 (+,+)	3.175 (+,+)	4.796 (+,+)	8.188E-17 (+,+)	1.000 (+,+)
joint norm MI	126.529 (+,-)	34.971 (+,+)	76.556 (+,+)	157.080 (-,-)	23010016.801 (-,-)	2.662 (+,+)	2.746 (+,+)	4.062 (+,+)	8.243E-17 (+,+)	1.080 (+,+)

6 Conclusions and Future Works

In this paper, we proposed a mutual-information-guided subtree transfer learning method for genetic programming, introducing a new perspective for finding valuable transfer targets. The proposed approach utilizes subtrees from the source domain that exhibit high MI values, transferring them as part of the initial population in the target domain to enhance the evolutionary process. Experimental results on ten symbolic regression tasks highlighted the effectiveness of the proposed approach, showing improved training and testing accuracy as well as better significance levels compared to state-of-the-art methods. These results underscore the potential of MI as a reliable indicator for identifying and leveraging useful subtrees. This method not only offers a promising strategy for transfer learning in GP but also emphasizes the importance of integrating MI into identifying useful segments for enhancing evolutionary performance. However, the experiments also indicated that there is room for further improvement in performance on testing, as not all cases outperform *SubTree50*. Additionally, the computational cost of evaluating all subtrees in the final generation of the source domain can be reduced by calculating MI during the evaluation stage. Since the compiler follows the precedence of parentheses when computing values, it can simultaneously obtain each subtree's value and compute its MI during evaluation, eliminating the need for separate MI calculations. Thirdly, while some theoretical analyses suggest how MI can guide the adaptation process, more works are needed to establish a stronger theoretical link between MI and the search dynamics of GP. Future research should focus on enhancing the generalization ability of the proposed method and deepening the theoretical understanding of GP's search behavior from the perspective of MI.

ACKNOWLEDGMENTS

The author would like to show many thanks to all helping authors get through life when writing this paper. In addition, particularly, many thanks to the author's university.

REFERENCES

[1] John R. Koza. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. (1st. ed.). MIT Press.

[2] David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. (1st. ed.). Addison-Wesley Longman Publishing Co., Inc.

[3] Abhishek Gupta, Yew Soon Ong, and Liang Feng. 2018. Insights on Transfer Optimization: Because Experience is the Best Teacher. *IEEE Transactions on Emerging Topics in Computational Intelligence* 2, 1, (2018),51-64.

[4] T. T. Huong Dinh, T. H. Chu, and Q. U. Nguyen. 2015. Transfer learning in genetic programming. In *Proceedings of 2015 IEEE Congress on Evolutionary Computation*. IEEE, 1145-1151.

[5] M. A. Ardeh, Y. Mei, and M. Zhang. 2019. Transfer Learning in Genetic Programming Hyper-heuristic for Solving Uncertain Capacitated Arc Routing Problem. In *Proceedings of 2019 IEEE Congress on Evolutionary Computation*. IEEE, 49-56.

[6] Brandon Muller, Harith Al-Sahaf, Bing Xue, and Mengjie Zhang. 2019. Transfer learning: a building block selection mechanism in genetic programming for symbolic regression. In *Proceedings of Genetic and Evolutionary Computation Conference Companion*. ACM, 350–351.

[7] M. A. Ardeh, Y. Mei, and M. Zhang. 2020. Genetic Programming Hyper-Heuristics with Probabilistic Prototype Tree Knowledge Transfer for Uncertain Capacitated Arc Routing Problems. In *Proceedings of 2020 IEEE Congress on Evolutionary Computation*. IEEE, 1-8.

[8] Jorge R. Vergara, and Pablo A. Estévez. 2014. A review of feature selection methods based on mutual information. *Neural Computing and Applications* 24, 1, (2014),175-186.

[9] Zahra Zojaji, and Mohammad Mehdi Ebadzadeh. 2016. Semantic schema theory for genetic programming. *Applied Intelligence* 44, 1, (2016),67–87.

[10] S. W. Card, and C. K. Mohan. 2005. Information theoretic indicators of fitness, relevant diversity & pairing potential in genetic programming. In *Proceedings of 2005 IEEE Congress on Evolutionary Computation*. IEEE, 2545-2552 Vol. 2543.

[11] Zahra Zojaji, and Mohammad Mehdi Ebadzadeh. 2018. Semantic schema modeling for genetic programming using clustering of building blocks. *Applied Intelligence* 48, 6, (2018),1442-1460.

[12] William B. Langdon, and Riccardo Poli. 2002. *Lessons from the GP Schema Theory* Foundations of Genetic Programming, Springer Berlin Heidelberg, Berlin, Heidelberg.

[13] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner Gardner, Marc Parizeau, and Christian Gagné. 2012. DEAP: evolutionary algorithms made easy. *J. Mach. Learn. Res.* 13, 1, (2012),2171–2175.