UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

COS700 Research Proposal

A Comparative Study: Transfer Learning in Genetic Programming for Data Classification

Student number: u17023239

Supervisor(s):
Prof. N. Pillay

Abstract

Multi-class data classification is probably the second most common machine learning problem and is applicable in a wide array of use cases. Research has been done, to adopt Genetic Programming (GP) to solve the multi-class data classification problem; however GP is one of the least effective techniques, which can be used to solve this problem. In recent research, it has been shown that various transfer learning methods can be used to improve the performance of GP in various problem domains. This research proposal will investigate and compare different transfer learning methods, which can be adapted and used to improve GP, for the problem domain of multi-class data classification. It will focus on the following transfer learning methods: BestGen, Fulltree, Sub-Tree, Genetic Programming with Code Reuse (GPCR) and Primitives from Sub-trees (PST).

Keywords:

Classifier, Genetic programming, Transfer learning, Multi-objective.

1 Introduction

Multi-class data classification is probably the second most common machine learning problem and is applicable in a wide array of use cases. For example, an entity might use multi-class data classification to classify incoming emails and attachments based on various risk levels. Research has been done to adopt Genetic Programming to solve the multi-class data classification problem [LC01].

Genetic Programming (GP) is an evolutionary algorithm introduced by John Koza [Koz94], which can be applied to various problem domains, due to the fact that it evolves solutions in the programming space. It has been shown that GP's performance can be improved by using transfer learning in the problem domains of symbolic regression and image classification [7 - 12].

Transfer learning aims to enhance the performance of GP by transferring knowledge from previously solved problems [HDCN15]. The two main areas in which transfer learning improves GP performance are achieving better training errors and improving models, which are trained using incomplete data.

Due to the fact that most transfer learning methods for GP are knowledge independent, it can be adopted to solve the problem of multi-class data classification. It would be of great benefit to a vast array of use cases to conduct research on the subject, if transfer learning will enhance the performance of GP in the problem domain of multi-class data classification.

This research proposal is structured as follows: *Section 2* of the research proposal defines the problem statement. Thereafter, in *Selection 3* an in-depth literature study is performed of GP, Multi-class classification in GP and transfer learning in GP. *Section 4* describes the methodology and process, which will be used in the proposed project. The research proposal finishes with a planning schedule, which shows the time frame of the proposed project in *Section 5*.

2 Problem Statement

Which, if any, Transfer Learning method improves the performance of Genetic Programming on the problem domain of multi-class data classification?

A study of the available literature shows that this question in its entirety has not yet been researched. Therefore, in order to answer this question, it needs to be divided into two sub-questions.

The first sub-question is: Can GP be used to effectively solve the problem domain multi-class data classification? This question will be answered by investigating the original GP methods used to solve the multi-class data classification problem. The proposed research will not focus on finding the most optimal GP method, because the proposed research's main purpose is to find the best transfer learning method which can be used to improve GP in the problem domain of multi-class data classification and not the best multi-class data classification

method. The selected method will be used as a baseline to which the different transfer learning methods will be compared.

The second sub-question is: Which, if any, transfer learning method can be adapted and used to improve GP for the problem domain of multi-class data classification? This question will be addressed by an in-depth literature study on the different transfer learning methods, which can be adapted for and used to improve GP in different problem domains. Due to the lack of research done on transfer learning methods for the problem domain of multi-class data classification, the research will focus on transfer learning methods, which were used in other problem domains, namely linear regression and image classification. The use of different problem domains will not present a problem, because the studied methods' knowledge that gets transferred will be problem domain independent.

Based on these questions, the proposed research will have the main objectives of:

1. Performing a literature study on the following subjects: genetic programming, multi-class classification methods for genetic programming and transfer learning methods, which can be adapted for and used to improve genetic programming.

2. Analysing the literature to specify which multi-class classification method will be used and which transfer learning methods will be used.

3. Adapting the multi-class data classification method and the transfer learning methods selected to enable the methods to integrate with each other.

4. Evaluating and comparing the performance of the different transfer learning methods selected to each other and to the baseline in order to determine which method, if any, improve the performance of GP on the problem domain of multi-class data classification.

3 Literature Study

3.1 Evolutionary Algorithms

Algorithms, which are modelled on natural processes, can provide computer algorithms, which are robust and direct [BS93]. One such group of algorithms are Evolutionary Algorithms. Evolutionary Algorithms have the main feature of using a process similar to Darwin's Theory of Biological Evolution to search a search space for solutions. Evolutionary Algorithms are, therefore, derived from a collective learning process within a population [BS93].

Elements in the population of an Evolutionary Algorithm are called chromosomes and are evaluated using a specific objective function to determine the chromosomes' fitness [BS93]. A variety of methods, known as the genetic operators, are used to select parents for different types of Evolutionary Algorithms, such as the deterministic, best chromosomes method [BS93]. There are three main genetic operators, namely mutation, recombination and selection

[BS93]. Evolutionary Algorithms are categorised by the solution given by the algorithm and the categories are genetic algorithm (GA), genetic programming (GP), differential evolution (DE), the evolution strategy (ES), and evolutionary programming (EP) [SK20]. This research proposal will focus on GP.

3.2 Genetic Programming (GP)

GP was initially introduced in 1994 by John Koza [Koz94] and is a relatively new Evolutionary Algorithm [SK20]. It is a specific form of GA which works on the program space [Koz94], for example having two variables' values and requiring a function, which illustrates the relationship between the two variables. GP can use arithmetic operations, mathematical functions, conditional logical operations and domain specific functions as a set of primitive functions [Koz94].

The chromosome of a GP commonly consists of one or more parse trees [Koz94]. Parse trees are composed of delete nodes, which are selected from the terminal set and the function set. A terminal set contains variables representing inputs, constants and zero argument operators [Koz94]. A function set consists of non-zero argument operators [Koz94]. Nodes in the terminal set are known as terminal nodes and node in the function set are known as function nodes.

The program space is explored through the use of generic operators over numerous iterations. The best performing chromosomes are selected to be evolved through the use of generic operators to create the next population. The performance of a chromosome is determined through the use of a fitness function. The search terminates, when the termination criterion has been met. A summary of this algorithm is given in *Algorithm 1.* The next paragraphs details each step in *Algorithm 1.*

Algorithm 1 GP Algorithm

```
Create initial population
while the termination criteria is not met do
    Evaluate population
    Select parents
    Apply generic operators to parents which creates the next population
end while
return best performing chromosome
```

Koza described the grow method, the full method and the ramped half-and-half method for initialisation of the population [Koz94]. The grow method starts by randomly selecting a root node from the function set and randomly selecting nodes that are used as inputs to the root node from the function set and terminal set. The inputs to the root node are then added to the tree. This process of selecting inputs for the function set nodes is repeated until all inputs are mapped to a terminal node. The full method, meanwhile, randomly selects nodes from the function set until a maximum depth is reached, after which

nodes are only selected from the terminal set. The full method creates more regular trees than the grow method. The two methods are combined to create the ramped half-and-half method, were half of the population is generated using the grow method and the other half is generated using the full method.

Various parent selection methods have been proposed, as illustrated by Helmuth and Abdelhady [HA20]. The tournament selection method, proposed by Koza [Koz94], is one of the simplest methods to use for parent selection. The tournament selection method randomly selects a subset of the population. The method evaluates the fitness of each chromosome in the randomly selected subset and selects the chromosome with the best fitness. The size of the subset, called the tournament size, is specified beforehand.

Koza uses crossover, mutation and reproduction for generic operators [Koz94]. Crossover uses the selection method to select two parents from the population, which are then combined by randomly selecting a subtree from each of the parents and then swapping the subtrees between the parents. Mutation works by selecting a parent from the population and replacing a random subtree in the parent with a newly generated subtree. It should be noted that the subtrees used in crossover and mutation must have the same return type as the subtree that it replaces. Reproduction is the simplest of the generic operators. It copies chromosomes from the previous generation to the new generation.

3.3 Multi-class classification in GP

GP can be used to solve multi-class classification. Multi-class classification is the machine learning problem of classifying data instances into more than two classes. GP can perform multi-classification tasks by using one of five different methods, namely Binary Decomposition, Static Range Selection, Dynamic Range Selection, Class Enumeration and Evidence Accumulation [LC01].

Standard GP methods can perform binary classification with a very high accuracy [Tac93], but has the problem of only classifying data instances into two classes. This problem can be solved by deconstructing the multi-class problem into multiple binary classification problems [GR94]. For example if the data set has three classes, a data instance will have its classification attribute removed and replaced with three new attributes, each representing a class and whether or not the data instance belongs to that class. A parse tree is then generated for each of these attributes. Loveard and Ciesielsk called this method Binary Decomposition in their paper [LC01].

A more efficient method for multi-class classification is Static Range Selection. Static Range Selection generates one parse tree, which returns a real value for each data instance. Ranges are assigned to each of the classes in the data set. The value gained from the parse tree, together with the rangers specifies, which class the data instance belongs to [4, 15].

The third method which can be used to solve multi-class classification is

Dynamic Range Selection, a variant of Static Range Selection. Dynamic Range Selection differs from Static Range Selection in that the ranges are evolved alongside the programs [LC01]. Smart and Zhang introduced two variations of Dynamic Range Selection, namely Centred Dynamic Range Selection (CDRS) and Slotted Dynamic Range Selection (SDRS) [SZ03]. It should be noted that SDRS is very similarly to Loveard and Ciesielsk Dynamic Range Selection method, since both uses slots to assign ranges [4, 15].

The Class Enumeration method changes the if operator in the function set to only return a class value or another if statement in order to improve the parse trees' expressive capabilities at the cost of enlarging the search space [LC01]. In the last method, Evidence Accumulation, the results of the GP consist of two parts, namely a certainty vector and a parse tree (the chromosome). The certainty vector contains a value between -1 and 1 for each class, where the highest value indicates the class of the data instance. The parse tree is a combination of an if statement and a new operator, which adds or subtracts a value at a specific position in the certainty vector [LC01].

3.4 Transfer Learning in GP

Transfer learning is a machine learning technique which aims to enhance the learning performance of machine learning algorithms by transferring knowledge from previously solved problems [HDCN15]. It therefore follows, that there needs to be at least two problems, known as the source task and the target task, in order to apply transfer learning. The source task is a machine learning task which is used to accumulate learned knowledge, while the target task is the problem that the algorithm aims to solve using the knowledge gained from the source task [WKW16]. In this research proposal, both of the tasks will be in the domain of multi-class classifiers and therefore the tasks will only differ in their complexity. Weiss, Khoshgoftaar and Wang, as well as Dinh, Chu and Nguyen, formally defined transfer learning as follows [HDCN15], [WKW16]:

"Given a source domain D_s and learning task T_s , a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function F_T (.) in D_T using the knowledge in D_s and T_s, where $D_s = D_T$, or $T_s = T_T$."

There are three areas, which need to be considered when implementing transfer learning. These areas are: which knowledge to transfer, how to transfer the knowledge and when to transfer the knowledge [HDCN15].

The first transfer learning methods introduced for GP were Fulltree, Sub-Tree and BestGen [WKW16]. All three of these methods address the area of how to transfer the knowledge by using the knowledge as a starting point for the GP, when starting with the problem task. The knowledge which the Fulltree method transfers is the specified top percentage of chromosomes from the final generation, which was generated by the GP that evolved from the source task [WKW16]. The Sub-Tree method differs from the Fulltree method, only in that

this method transfers random parts of the chromosome instead of the whole chromosome, i.e. if the chromosome is a parse tree, a random sub-tree will be selected for transfer [WKW16]. The BestGen method, meanwhile, selects a sample of chromosomes (with a prior specified sample size) from each generation [WKW16]. Haslam, Xue and Zhang expanded on this method by adding the newly generated chromosomes to the selected chromosomes and using two different selection criteria to select the chromosomes for the initial population used in the GP that will be trained to solve the problem task [HXZ16]. The two methods used to make the selection in this method are terminal selection and k-Throttle [HXZ16]. This method, however, does not significantly improve the performance of the algorithms [HXZ16].

Jaśkowski, Krawiec and Wieloch then created the Genetic Programming with Code Reuse (GPCR) method [JKW14]. The GPCR method transfers the full chromosome at the end of the GP run for the source task. The GPCR method then uses a new genetic operator, called the Crossbreeding operator, which works the same as the crossover operator, except that one chromosome is selected from the pool of source task chromosomes and the other chromosome is selected from the pool of target task chromosomes [JKW14].

Another transfer learning method introduced for GP, namely the Common Subtree from Related Problems (CSRP) method, was proposed by O'Neill, Al-Sahaf, Xue, and Zhang [OASXZ17]. The CSRP method requires two source tasks which are related. Two different GP are then trained on both of the source tasks. The best chromosomes are selected from the final generation of the GP and compared to obtain common subtrees in the chromosomes [OASXZ17]. The subtrees are then used in the function set when the GP is trained on the target task.

Mullar, Al-Sahaf, Xue, and Zhang improved the CSRP method by proposing a method, called the Primitives from Sub-trees (PST) method, which has a more restrictive selection process [MASXZ19]. The PST method only allows unique subtrees of a specified length and replaces the subtrees' leaf nodes with inputs to create functions [MASXZ19].

3.5 Critical Analysis

Sections 3.1 to *3.4* of this research proposal shows GP can solve the problem of multi-class data classification and that transfer learning is a valid method to improve the performance of a GP. However, most of the literature on transfer learning as a valid method to improve the performance of GP was performed on the problem domains of symbolic regression and image classification, not multi-class data classification. Therefore, research will be performed on which, if any, of the transfer learning methods improves the performance of GP for the problem domain of multi-class data classification.

As illustrated above, there are multiple methods of adapting GP to solve the multi-class data classification and transfer learning for GP. The next sub-

sections will discuss how the literature will be used in the proposed research, what the research can contribute to the field and potential future research, which can be done on this topic.

3.5.1 Overview of the GP

The SDRS method will be used in the proposed research. Therefore, *algorithm 1* will be modified as follows: an additional step will be added to the start of the algorithm to initialise the class ranges and a step will be added to the evolutionary loop, which applies the SDRS method. See *section 3.5.2* Multi-class Data Classification Method, for more detail on the SDRS method. The chromosomes of the GP will consist out of a parse tree and ranges. The implementation of the GP that is given in the next paragraphs does not take into account the changes that the various transfer learning methods requires.

The initial population generation will use the ramped half-and-half method as described in *section 3.2.* . The function set and terminal set will consist out of various logic and arithmetic operators. See *table 1* for the function set, which will be used and *table 2* for the terminal set, which will be used. The crossover, mutation and reproduction methods will be used; however the methods will only be applied to the parse tree and not to the whole chromosome, as discussed in *section 3.2.* .

The fitness function that will be used to evaluate the chromosomes, is the accuracy of the chromosome on the training data. That is to say the fitness of a chromosome will be the number of correctly classified instances divided by the total number of instances. Therefore, the fitness value will be a percentile, where a 100% means that all of the data instances were correctly classified.

The termination criteria for the GP will be when one of the following two criteria is met: the accuracy (fitness) of a chromosome is above a specified threshold or the number of generations is above a pre-defined threshold.

3.5.2 Multi-Class Data Classification Method

As mentioned previously, the proposed research will focus on the impact of different transfer learning methods for GP in the problem domain of multi-class classification. Therefore, only the original multi-class data classification methods were considered in the above literature survey. Out of the original multi-class data classification methods, the Dynamic Range Selection method was found to be the most accurate and consistent method [LC01]. However, the SDRS method is simpler to implement, but still based on the same principles as the Dynamic Range Selection Method and thus also assumed to be an accurate and consistent method. This research will therefore use the SDRS method. The SDRS method will have a hundred slots between the values of -25 and 25.

Symbol	Return Type	Number of Arguments	Arguments Type	Description
+	Double	2	Double, Double	Addition
-	Double	2	Double, Double	Subtraction
*	Double	2	Double, Double	Multiplication
/	Double	2	Double, Double	Protected division (If arg 2 is 0 return 0)
If	Double	3	Boolean, Double, Double	Conditional, if arg 1 return arg 2 else return arg 3
>=	Boolean	2	Double, Double	Return true if arg1 is greater than or equal to arg2
<=	Boolean	2	Double, Double	Return true if arg1 is less than or equal to arg2
==	Boolean	2	Double, Double	Return true if arg1 equal to arg2
<>	Boolean	3	Double, Double, Double	Return true if arg1is between arg2 and arg3

Table 1: Function set

Symbol	Type	Description
Random(-1,1)	Double	Returns a random value between -1 and 1
Attribute x	Double	Returns the value of attribute x

Table 2: Terminal set

The first step of the SDRS method is to evaluate each chromosome of the population in order to get the chromosome's output (called O) for each training data instance and the fitness of the chromosome [SZ03]. In the second step, the SDRS method assigns values to the slots for each class based on the relative contribution of the chromosome (called R) [SZ03]. The R is calculated by adding 0.5 to the fitness of the chromosome [SZ03]. The slots index (called I) are obtained by rounding O to the closed slot. The last step is where the SDRS method determines the ranges dynamically, by selecting the class with the largest value for that specific slot [SZ03]. If a slot does not have any values, it will be assigned the class of its nearest neighbouring slot. *Algorithm 2* indicates the above steps in pseudocode.

It should however be noted that the SDRS method is only applied every five generations, in order to ensure that the parse trees are being evolved, and that the initial ranges are randomly assigned classes, even if the initial chromosome where transferred.

3.5.3 Transfer Learning Methods

The Transfer Learning Methods used in the proposed research will be selected based on two criteria, namely how the transfer learning method performs and whether the transfer learning method is the most improved method. The five methods, which will be used in the proposed research are: BestGen, Fulltree, Sub-Tree, GPCR and PST. Although the methods introduced by Haslam, Xue and Zhang are an expansion of the BestGen, Fulltree and Sub-Tree methods, were not chosen because they did not improve the original methods in terms of performance [HXZ16]. The CSPR method was also not chosen, because the PST method is an improved version of the CSPR method [MASXZ19]. All of the selected methods either transfer the full parse trees or part of the parse tree, not the whole chromosome. See *Section 3.4.* for an in-depth explanation of the above mentioned methods.

Algorithm 2 SDRS method

```
for each slot and each class do
    Set Array[slot][class] to zero
end for
for each data instance (called d) do
    for each chromosome (called ch) do
        O = execute ch with d as input
        I = round O to the closest slot
        Array[I][class of d] += R
    end for
end for
for each slot (called s1) do
    if all values in Array[s1] are zero then
        Ranges[s1] = -1
    else
        c = the class with the largest Array[slot][class] value
        Ranges[s1] = c
    end if
end for
for each slot (called s2) do
    if Ranges[s2] == -1 then
        Ranges[s2] = to the value of the closest slot in Ranges where the values
does not equal -1
    end if
end for
return best performing chromosome
```

3.5.4 Contributions

The original paper, which proposes methods for multi-class data classification using GP found that GP preforms poorly against other Machine Learning techniques in the problem domain of multi-class data classification [LC01]. New multi-class data classification methods for GP, such as random binary decomposition based GP, improves the accuracy of GP, but at the expense of longer runtimes [LPI21]. Transfer learning can improve the accuracy of GP and reduce the runtimes of GP, which will make GP a more viable option for solving the problem domain of multi-class data classification in a wide array of use cases and industries.

3.5.5 Potential Future Research

As discussed above, the proposed system to solve the problem of multi-class classification consists out of 3 parts, namely the underlying evolutionary algorithm used, the multi-class data classification method used and the transfer learning method used. The proposed research only focuses on comparing transfer learning methods for GP with a specific multi-classification method. Therefore, future research can be performed on how different evolutionary algorithms perform with transfer learning on the multi-class data classification problem. Future research can also be performed on how different multi-class data classification methods impact the performance of GP with transfer learning.

4 Methodology

The Proof by Demonstration research methodology, which is based on iterative improvement, will be followed [Joh06]. Therefore, an artefact will be built and iterated in order to improve its performance. For the proposed research, the artefact will consist out of an implementation for each of the five transfer learning methods selected above and the chosen baseline. Each selected transfer learning method will be developed individually. The following subsections will discuss the major parts of the proposed research:

4.1 Data sets and data pre-processing

Five data sets will be selected form the UC Irvine Machine Learning Repository (`https://archive.ics.uci.edu/ml/index.php`). The winequality-white data set will be used as the source task for each transfer learning method. Note that the winequality-red data set will also be used for the PST method, since the PST method needs two data sets to operate. The iris, seeds, mushroom and dry beans data sets will be used as the target tasks, because these data sets differs in sizes and have missing values. See *table 3* for more details. The categorical data in the mushroom and the iris data sets need to be mapped to integers in order to work with the function set. The missing values in the mushroom data set will be assigned to -1 for the reason given above.

4.2 Implementations of the artefact

The initial artefact will be implemented as discussed in *Section 3.5. Algorithm 3* below, shows the full algorithm of the GP with the SDRS method. All of the transfer learning methods will extend *Algorithm 3* as discussed in *Section 3.4.* and *Section 3.5.3.* .

Data set	**Number of Instances**	**Number of Attributes**	**Attributes Types**	**Has missing values**
Wine quality (winequality-white)	4898	12	Real	No
Wine quality (winequality-red)	1599	12	Real	No
Seeds	210	7	Real	No
Mushroom	8124	22	Categorical	Yes
Iris	150	4	Real and Categorical	No
Dry Bean	13611	17	Real	No

Table 3: Data set

Algorithm 3 GP Algorithm with the SDRS method

```
Create initial population
Create initial ranges
while the termination criteria is not met do
    Evaluate population
    Select parents
    Apply generic operators to parents which creates the next population
    if the number of generations are a factor of 5 then
        Apply the SDRS method
    end if
end while
return best performing chromosome
```

4.3 Comparison of transfer learning methods

Each of the five selected transfer learning methods and the baseline, will be evaluated and then compared to each other. Each of the selected transfer learning methods and the baseline will be run twenty times for each of the data sets. The transfer learning methods and the baseline will be evaluated on each of the following statistical measurements:

1. Average run time – the average time it takes the GP to meet the termination criteria.
2. Average accuracy – the average accuracy of the solution given by the GP.
3. Standard deviation – the standard deviation of the accuracy form each of the solutions given by the GP for each run.

4.4 Artefact improvement

The Proof of Demonstration research methodology specifies that the artefact is improved by iterative improvements. The proposed artefact will be iteratively improved by adjusting the different GP parameters, until either no more improvement is seen in the statically measures specified above, which will be used to evaluate the transfer learning methods or the research deadline is met. The GP parameters which will be adjusted are:

1. Population size – the number of chromosomes in a population.
2. Percentage of generic operators used – the percentage of crossover, mutation and reproduction used.
3. Transfer learning methods parameters – the parameters needed for each transfer learning method, for example the number of chromosomes transferred from one GP to another. (The parameters are dependent on the transfer learning method and therefore will not be listed here.)
4. Termination criteria – consist out of the maximum number of generations and the accuracy threshold.
5. Tournament size – The number of chromosomes that are evaluated in the tournament selection method.

4.5 Technical Specifications

The artefact will be coded in Java. Java was chosen, because of its powerful parallel operators. The experiments will be performed on a desktop PC. This has an Intel Core i7-9700 CPU and NVIDIA GeForce RTX 2060 Super GPU. If the experiments runtimes are exceptionally long, the experiments will be adapted to run on the CHPC (Centre for High Performance Computing), to reduce runtimes.

4.6 Final Evaluation

Statistical testing will be used to test the viability of the final artefact. More specifically the null hypothesis testing will be used. The setup of the testing is:

$$h_0 : \mu_B >= \mu_T$$

$$h_a : \mu_B < \mu_T$$

$$\alpha : 0.05$$

Where μ_B is the average accuracy of the baseline, across all of the datasets and μ_T is the average accuracy of the transfer learning methods, across all of the datasets. Although transfer learning methods might improve the run time of GP, the testing is focused on the accuracy, because the proposed research aims to improve the performance of GP in the problem domain of multi-class data classification. Thus the alternative hypothesis (h_a) is accepted when a GP with a transfer learning method outperforms a GP without transfer learning. The research will be a success when the alternative hypothesis is accepted with a confidence level of 95%.

5 Planning

Figure 1 shows the project plan for the academic and non-academic work for the research. The time period between columns is two weeks.

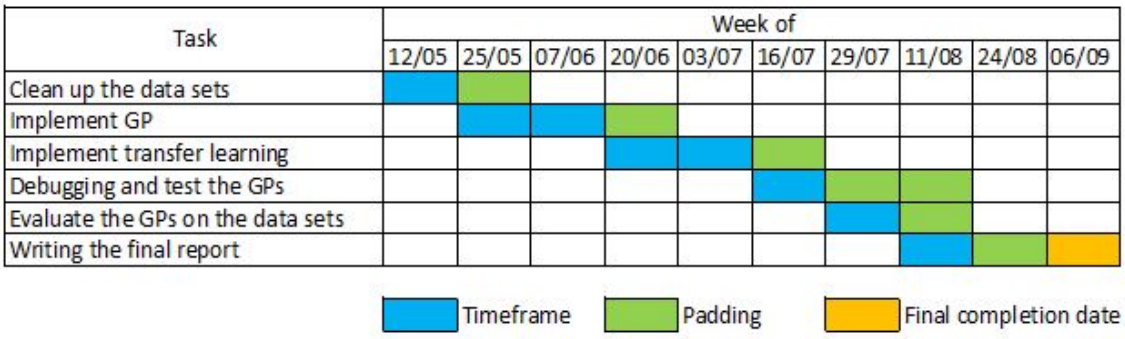

Figure 1: An image of a galaxy

References

[BS93] Thomas Bäck and Hans Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, March 1993.

[GR94] Chris Gathercole and Peter Ross. Dynamic training subset selection for supervised learning in genetic programming. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer, editors, *Parallel Problem Solving from Nature — PPSN III*, pages 312–321, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[HA20] Thomas Helmuth and Amr Abdelhady. Benchmarking parent selection for program synthesis by genetic programming. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 237–238. Association for Computing Machinery, New York, NY, USA, 2020.

[HDCN15] Thi Thu Huong Dinh, Thi Huong Chu, and Quang Uy Nguyen. Transfer learning in genetic programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1145–1151, 2015.

[HXZ16] Edward Haslam, Bing Xue, and Mengjie Zhang. Further investigation on genetic programming with transfer learning for symbolic regression. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 3598–3605, 2016.

[JKW14] Wojciech Jaśkowski, Krzysztof Krawiec, and Bartosz Wieloch. Cross-Task code reuse in genetic programming applied to visual learning. *International Journal of Applied Mathematics and Computer Science*, 24:183–197, March 2014.

[Joh06] Chris Johnson. What is research in computing science? Technical report, 2006.

[Koz94] John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4:87–112, June 1994.

[LC01] T Loveard and V Ciesielski. Representing classification problems in genetic programming. *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, 2:1070–1077, May 2001.

[LPI21] Lushen Liao, Adam Kotaro Pindur, and Hitoshi Iba. Genetic programming with random binary decomposition for Multi-Class classification problems. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 564–571, 2021.

[MASXZ19] Brandon Muller, Harith Al-Sahaf, Bing Xue, and Mengjie Zhang. Transfer learning: A building block selection mechanism in genetic programming for symbolic regression. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO

'19, pages 350–351, New York, NY, USA, 2019. Association for Computing Machinery.

[OASXZ17] Damien O'Neill, Harith Al-Sahaf, Bing Xue, and Mengjie Zhang. Common subtrees in related problems: A novel transfer learning approach for genetic programming. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1287–1294, 2017.

[SK20] Adam Slowik and Halina Kwasnicka. Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 32:12363–12379, March 2020.

[SZ03] Will Smart and Mengjie Zhang. Classification strategies for image classification in genetic programming. *Proceeding of Image and Vision Computing Conference*, 2003.

[Tac93] Walter Alden Tackett. Genetic programming for feature discovery and image discrimination. In *ICGA*, 1993.

[WKW16] Karl Weiss, Taghi M Khoshgoftaar, and Dingding Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, May 2016.