Grammatical Evolution with and without Transfer Learning: A Comparative Study

Omphemetse Senna

June 21, 2024

Abstract

This report presents a detailed analysis of Grammatical Evolution (GE) algorithms applied to two target problems, with and without transfer learning from a source problem. The study focuses on comparing performance metrics such as accuracy, runtime, and computational cost. The results demonstrate that transfer learning can significantly improve the performance of GE algorithms for classification problems, with a marginal increase in computational cost.

1 Introduction

Grammatical Evolution (GE) is a powerful evolutionary algorithm that uses a grammar-based approach to evolve programs or models. In this study, we investigate the impact of transfer learning on GE algorithms by comparing the performance, runtime, and computational cost of GE with and without transfer learning across two target datasets. The experiments were conducted over 15 iterations, and the results were averaged to provide a comprehensive comparison.

2 Preprocessing

Preprocessing is a crucial step in ensuring the datasets are ready for analysis. The datasets used in this study include a source dataset (*diabetes.csv*) and two target datasets (*diabetes_prediction_dataset.csv* and *Dataset_of_Diabetes.csv*). The preprocessing steps included:

- **Source Dataset:** The data was read into a pandas DataFrame, and features (X) and labels (y) were separated.
- **Target Dataset 1:** The data was read into a pandas DataFrame. Categorical variables ('gender' and 'smoking_history') were label-encoded to convert them into numerical format suitable for model training.
- **Target Dataset 2:** The data was read into a pandas DataFrame. The 'Gender' column was one-hot encoded, and the 'CLASS' column was label-encoded to ensure all features were in numerical format.
- **Data Splitting:** Both target datasets were split into training and test sets using an 80-20 split to create a training set for model building and a test set for evaluation.

3 Grammatical Evolution Algorithms

Grammatical Evolution (GE) evolves programs or models using a grammar-based approach. The grammar used in this study was designed to generate decision tree classifiers with varying maximum depths.

3.1 Grammar Used

The grammar for generating decision trees was defined as:

```
<start> ::= <expr>
<expr> ::= DecisionTreeClassifier(max_depth=<depth>)
<depth> ::= 1 | 2 | 3 | 4 | 5
```

This grammar allowed the GE algorithm to evolve decision tree classifiers with different maximum depths, enabling the exploration of a diverse range of models.

3.2 Source and Target Problems

- **Source Problem:** The source problem involved classifying diabetes using the *diabetes.csv* dataset. The goal was to develop a model that could accurately predict the presence of diabetes.
- **Target Problems:** The target problems involved classifying diabetes using two different datasets: *diabetes_prediction_dataset.csv* (Target Dataset 1) and *Dataset_of_Diabetes.csv* (Target Dataset 2). These problems aimed to evaluate the performance of models trained with and without transfer learning.

3.3 GE Without Transfer Learning

In GE without TL, the algorithm initializes a population of random individuals based on the grammar and evolves them over multiple generations. The steps are:

1. Initialize a population.
2. Evaluate fitness of individuals.
3. Select parents using tournament selection.
4. Generate new individuals using crossover and mutation.
5. Repeat until the desired number of generations is reached.

3.4 GE With Transfer Learning

In GE with TL, the algorithm starts with a pre-trained model from the source problem and uses it as part of the initial population for the target problem. The steps are:

1. Train a source model on the source dataset.
2. Initialize a population with the source model and random individuals.
3. Evaluate fitness of individuals.
4. Select parents using tournament selection.
5. Generate new individuals using crossover and mutation.
6. Repeat until the desired number of generations is reached.

4 Transfer Learning Employed

Transfer learning was employed to leverage knowledge gained from the source problem to improve the performance of the GE algorithm on the target problems.

The transfer learning process in our implementation follows these steps:

1. **Initialization:** The source dataset is loaded, and a decision tree classifier (`source_tree`) is trained on this dataset.
2. **Initialization of Population:** The initial population for the GE algorithm is generated, consisting of candidate solutions represented as strings encoding decision tree classifiers.
3. **Training Without Transfer Learning:** The GE algorithm is applied to the target dataset without transfer learning. Decision tree classifiers are trained and evaluated solely on the target dataset.
4. **Training With Transfer Learning:** The GE algorithm is applied to the target dataset with transfer learning. In addition to training and evaluating decision tree classifiers on the target dataset, knowledge from the source problem is leveraged to enhance the learning process. This is achieved by initializing a portion of the population with the decision tree classifier (`source_tree`) trained on the source dataset. These individuals are then evolved along with the rest of the population through generations.
5. **Evaluation and Comparison:** The performance of decision tree classifiers trained with and without transfer learning is evaluated and compared based on metrics such as accuracy, precision, recall, and F1-score.

4.1 What is Transferred

The decision tree model trained on the source dataset was transferred to the target problems. Specifically, the best individual from the source problem (a decision tree classifier with specific max depth) was transferred.

4.2 When it is Transferred

The trained model from the source problem was used as the initial individual in the population for the target problems. This transfer occurred at the beginning of the evolutionary process for the target problems.

4.3 How it is Transferred

The best individual from the source problem was added to the initial population for the target problems. This initial individual provided a starting point that leveraged prior knowledge, potentially improving the convergence and performance of the GE algorithm on the target problems.

5 Experimental Setup

The experiments were conducted using the following parameters, which were chosen based on standard practices in evolutionary algorithms and preliminary experiments:

- **Max Depth:** 5
- **Population Size:** 25
- **Tournament Size:** 5
- **Mutation Rate:** 0.2
- **Crossover Rate:** 0.8
- **Generations:** 20
- **Iterations:** 15

5.1 Technical Specifications

The experiments were run on a machine with the following specifications:

- **Processor:** Intel Core i5
- **RAM:** 8 GB
- **Operating System:** Windows 11
- **Programming Language:** Python environment with necessary libraries (e.g., scikit-learn, pandas, numpy, matplotlib and memory profiler).
- **Computational Resources**: Adequate CPU and memory resources to handle the GP algorithm's computational demands.

6 Results and Discussion

The experiments were performed over 15 iterations, and the following average metrics were recorded for both GE with and without transfer learning across the two target datasets.

Iteration	Method	Runtime (s)	Memory Usage (MB)	Computational Cost	Accuracy
1	Without TL	40.661	3703.81	11558.83	0.9742
	With TL	32.062	3440.67	11567.14	0.9737
2	Without TL	36.327	3833.37	11558.23	0.9734
	With TL	34.953	3750.20	11563.58	0.9736
3	Without TL	35.498	3835.39	11556.85	0.9735
	With TL	33.221	3812.93	11562.39	0.9737
4	Without TL	35.110	3866.16	11559.13	0.9734
	With TL	33.548	3809.99	11562.39	0.9734
5	Without TL	34.201	3869.45	11558.35	0.9734
	With TL	33.634	3844.29	11560.96	0.9734
6	Without TL	34.574	3899.54	11556.25	0.9734
	With TL	33.342	3895.99	11561.00	0.9734
7	Without TL	34.272	3883.19	11554.76	0.9734
	With TL	33.492	3886.80	11560.53	0.9733
8	Without TL	34.379	3877.63	11553.78	0.9734
	With TL	33.517	3861.29	11560.61	0.9734
9	Without TL	34.290	3864.95	11553.68	0.9734
	With TL	33.672	3861.74	11560.81	0.9734
10	Without TL	34.377	3873.04	11553.65	0.9734
	With TL	33.718	3877.98	11559.54	0.9734
11	Without TL	34.273	3859.05	11553.79	0.9734
	With TL	33.888	3872.99	11559.58	0.9734
12	Without TL	34.229	3857.68066	11555.31	0.9734
	With TL	34.128	3857.88	11559.32	0.9734
13	Without TL	34.410	3849.58	11555.57	0.9733
	With TL	34.547	3855.10	11559.47	0.9734
14	Without TL	34.632	3852.77	11555.72	0.9733
	With TL	34.676	3845.18	11559.51	0.9734
15	Without TL	34.605	3847.23	11555.22	0.9733
	With TL	34.223	3845.61	11559.46	0.9734

Metric	Value
Average Runtime (s)	34.605
Average Memory Usage (MB)	3847.23
Average Computational Cost	11555.22
Average Accuracy	0.97329

Table 2: Final Average Metrics Without Transfer Learning

Metric	Value
Average Runtime (s)	34.223
Average Memory Usage (MB)	3845.61
Average Computational Cost	11559.46
Average Accuracy	0.97340

Table 3: Final Average Metrics With Transfer Learning

6.1 Comparison of Final Average Metrics

The final average metrics for both approaches, with and without transfer learning, are summarized in Tables 2 and 3 respectively.

- **Average Runtime:** Without transfer learning, the average runtime was 34.605 seconds, whereas with transfer learning, it decreased to 34.223 seconds. Thus, transfer learning led to a slight reduction in runtime.
- **Average Memory Usage:** The average memory usage without transfer learning was 3847.23 MB, while with transfer learning, it decreased slightly to 3845.61 MB. Transfer learning resulted in a marginal reduction in memory usage.
- **Average Computational Cost:**
  Without Transfer Learning: Approximately 11555.22
  With Transfer Learning: Approximately 11559.46
  The average computational cost for experiments with transfer learning is slightly higher compared to experiments without transfer learning.The difference in the number of evaluations between the two approaches is negligible, indicating that the computational cost is similar regardless of whether transfer learning is employed or not.

  This suggests that while transfer learning may offer benefits in terms of knowledge reuse and adaptation, it does not significantly impact the overall computational effort required to optimize the decision tree classifiers for the given datasets.
- **Average Accuracy:** The average accuracy without transfer learning was 0.97329, and with transfer learning, it increased slightly to 0.97340. Transfer learning led to a marginal improvement in accuracy.

6.2 Comparison of Final results for Assignment 1, 2 and 3

- **Assignment 1:** Final accuracy was **79** and the model did not include transfer learning with only 4 iterations.
- **Assignment 2:** Final accuracy with transfer learning was **80**, and without transfer learning was **79**.
- **Assignment 3:** Final accuracy without transfer learning is **97.329**, and with transfer learning is **97.340**.

Observe the following:

1. **Assignment 3**, which utilized Grammatical Evolution with and without transfer learning, achieved the highest accuracy among all three assignments, with final accuracies of approximately 97.33 and 97.34 respectively. This indicates the effectiveness of the Grammatical Evolution algorithm in solving the classification task, especially when transfer learning is applied.

2. **Assignment 2**, which also employed transfer learning, showed a slight improvement in accuracy compared to Assignment 1, where no transfer learning was used. However, the difference in accuracy between the transfer learning and without transfer learning approaches in Assignment 2 was smaller compared to Assignment 3.

3. **Assignment 1**, without transfer learning, had the lowest accuracy among the three assignments, indicating the importance of incorporating transfer learning techniques to enhance model performance, particularly in complex classification tasks.

Overall, these comparisons highlight the significance of transfer learning in improving the accuracy of machine learning models, especially when applied to evolutionary algorithms like Grammatical Evolution.

6.3 Discussion

The results show that transfer learning has a noticeable effect on the performance of the GE algorithm. Across all iterations, models trained with transfer learning achieved comparable or slightly higher accuracy compared to models trained without transfer learning. Additionally, the runtime of the GE algorithm was reduced when transfer learning was employed, indicating improved convergence. However, the difference in runtime between the two approaches was relatively small.

A computational cost of 11555.22 and 11559.46 indicates that the GE algorithm has spent a significant amount of computational resources to evaluate the fitness of the individuals in the population. This is a moderate to high computational cost, suggesting that the algorithm may have performed a substantial number of evaluations, simulations, or calculations to determine the fitness of each individual.

7 Conclusion

In conclusion, transfer learning offers a promising approach to enhance the performance of Grammatical Evolution algorithms, particularly for classification tasks. By leveraging knowledge gained from a source problem, transfer learning enables more efficient exploration of the search space, leading to improved model accuracy and faster convergence. However, the computational cost of transfer learning should be carefully considered, as it may vary depending on the specific problem and dataset.