mathematics

MDPI

Article

Feature Optimization and Dropout in Genetic Programming for Data-Limited Image Classification

Chan Min Lee [1], Chang Wook Ahn [1,2,*] and Man-Je Kim [3,*]

[1] AI Graduate School, Gwangju Institute of Science and Technology, Gwangju 61005, Republic of Korea; mini0404@gm.gist.ac.kr
[2] GIST Institute for Artificial Intelligence, Gwangju Institute of Science and Technology, Gwangju 61005, Republic of Korea
[3] Convergence of AI, Chonnam National University, Gwangju 61186, Republic of Korea
* Correspondence: cwan@gist.ac.kr (C.W.A.); jaykim0104@jnu.ac.kr (M.-J.K.)

Abstract: Image classification in data-limited environments presents a significant challenge, as collecting and labeling large image datasets in real-world applications is often costly and time-consuming. This has led to increasing interest in developing models under data-constrained conditions. This paper introduces the Feature Optimization and Dropout in Genetic Programming (FOD-GP) framework, which addresses this issue by leveraging Genetic Programming (GP) to evolve models automatically. FOD-GP incorporates feature optimization and adaptive dropout techniques to improve overall performance. Experimental evaluations on benchmark datasets, including CIFAR10, FMNIST, and SVHN, demonstrate that FOD-GP improves training efficiency. In particular, FOD-GP achieves up to a 12% increase in classification accuracy over traditional methods. The effectiveness of the proposed framework is validated through statistical analysis, confirming its practicality for image classification. These findings establish a foundation for future advancements in data-limited and interpretable machine learning, offering a scalable solution for complex classification tasks.

Keywords: genetic programming (GP); feature optimization; dropout; image classification

MSC: 68T20

check for updates

Citation: Lee, C.M.; Ahn, C.W.; Kim, M.-J. Feature Optimization and Dropout in Genetic Programming for Data-Limited Image Classification. *Mathematics* **2024**, *12*, 3661. https://doi.org/10.3390/math12233661

Academic Editor: Vladimir Balan

Received: 24 October 2024
Revised: 20 November 2024
Accepted: 20 November 2024
Published: 22 November 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Image classification has garnered significant attention in recent years, especially in fields such as healthcare, medical diagnostics, and automated surveillance, where accurate and efficient systems are critical [1–4]. Advances in technology have enabled the development of advanced methods in image classification. Traditionally, research focused on designing algorithms to extract meaningful features from images to achieve accurate classification [5–7]. However, with the advent of deep learning, particularly Convolutional Neural Networks (CNNs), the process has become automated, allowing models to learn complex patterns directly from raw image data [8,9]. While CNNs have achieved state-of-the-art performance in many applications, they typically require large labeled datasets to train effectively [10]. In data-intensive fields such as medical diagnostics, generating large labeled datasets is often costly and time-consuming, requiring expert annotations [11].

To address this issue, there has been a growing interest in developing data-limited classification models [12]. These models aim to reduce their reliance on large datasets while maintaining high accuracy. Recent efforts, such as the VIPriors challenge [13], have highlighted the need for novel approaches that can perform well in data-limited scenarios. Specifically, researchers have developed various novel approaches for scenarios that prohibit the use of pre-trained models and transfer learning, compelling them to explore alternative strategies for enhancing data efficiency. Despite these advances, the

challenge of balancing data efficiency with model performance remains, as many data-limited models fail to generalize effectively to unseen data.

This study proposes the Feature Optimization and Dropout in Genetic Programming (FOD-GP), a novel approach that address the challenges of generalization and data efficiency in image classification. FOD-GP integrates feature optimization and adaptive dropout into the GP-based Evolutionary Deep Learning (GPEDL) framework, enabling models to evolve and handle data-limited scenarios more effectively. By incorporating feature optimization, we aim to enhance model efficiency by selecting the most relevant features and reducing redundancy, which is crucial in high-dimensional datasets. Additionally, adaptive dropout is employed as a regularization technique to improve performance during the evolutionary process. Our approach leverages the flexibility of GP to dynamically evolve models for data-limited scenarios. Our key contributions are as follows:

1. Feature optimization process: we introduce a feature optimization process within the GP framework to reduce redundant features and improve model performance.
2. Adaptive dropout techniques: we incorporate adaptive dropout techniques within GP nodes to enhance the generalization capabilities of the evolved models in data-limited scenarios.
3. Comprehensive evaluation on benchmark datasets: we conduct a rigorous evaluation of our approach on well-known benchmark datasets–including CIFAR10, FMNIST, and SVHN–comparing its performance with existing GPEDL method through statistical analysis.

We organize the remainder of this paper as follows: Section 2 introduces GPEDL framework and details the structure, terminals, and function of the proposed FOD-GP. Section 3 describes the experimental design and results comparing the performance of FOD-GP with the existing method. Finally, Section 4 summarizes the key contributions and the significance of the proposed approach.

2. Materials and Methods

In this section, we introduce the existing GPEDL approach and detail the structure, terminals, and function sets of the Feature Optimization and Dropout in Genetic Programming (FOD-GP) framework. Additionally, we provide a comprehensive introduction to the feature optimization layer and adaptive dropout techniques.

2.1. GPEDL: Genetic Programming-Based Evolutionary Deep Learning

Genetic Programming (GP) is an evolutionary algorithm that leverages the principles of natural selection to automatically generate complex solutions. GP offers flexibility by utilizing tree structures to evolve solutions with variable depth, enabling adaptation to different problems and data types [14]. Models in GP evolve over generations, refining individuals through operations such as selection, crossover, and mutation. Like deep learning, GP automates the learning of models using complex operators, which has led to increased interest in GP as a form of Evolutionary Deep Learning (EDL). GP-based EDL methods, which automatically evolve variable-length models, have demonstrated the potential to outperform traditional deep learning models that rely on fixed neural network architectures [15,16].

Research is actively exploring the application of GP-based EDL to image classification tasks [15–19]. A notable advancement in this field is the GP-based Evolutionary Deep Learning (GPEDL) framework, introduced by Bi et al. [15]. GPEDL integrates GP with deep learning-inspired techniques to evolve interpretable, flexible models that perform well in data-constrained environments. By constructing models that can be either shallow or deep, GPEDL can learn a variety of features from both color and grayscale images. It performs effective transformations on input data through internal nodes, allowing for robust feature extraction. GPEDL has demonstrated superior performance in several data-limited image classification tasks compared to traditional deep learning methods.

While GPEDL improves both interpretability and data efficiency, there are still opportunities to enhance its performance further. In this work, we build on the GPEDL framework by incorporating feature optimization and adaptive dropout techniques. Feature optimization enhances the model's efficiency, accuracy, and interpretability by selecting relevant features, which is crucial in high-dimensional data environments where irrelevant or redundant features can negatively impact model performance [20,21]. As the complexity of data increases, the need for robust feature optimization methods becomes even more critical. By integrating these techniques into the GP framework, we aim to improve the model's ability to effectively manage feature vectors, ensuring better performance and generalization in data-limited environments.

In addition to feature optimization, dropout is a widely used regularization technique in both neural networks and evolutionary algorithms that prevents overfitting by randomly omitting neurons during training. Dropout has proven powerful in improving model generalization and robustness [22]. For example, the Dropout Genetic Algorithm (DGA) combines genetic algorithms with dropout to minimize generalization errors, especially in small datasets [23]. This study demonstrated the effectiveness of DGA in improving model robustness and performance. To the best of our knowledge, dropout has not yet been applied to GP-based Evolutionary Deep Learning (EDL) frameworks. Building on the success of dropout in neural networks and evolutionary algorithms, we propose incorporating adaptive dropout techniques into the nodes within GP.

2.2. FOD-GP Structure

The FOD-GP structure consists of six different layers, which are divided into core and adaptive. Figure 1 illustrates the structure of FOD-GP, offering insight into feature optimization.

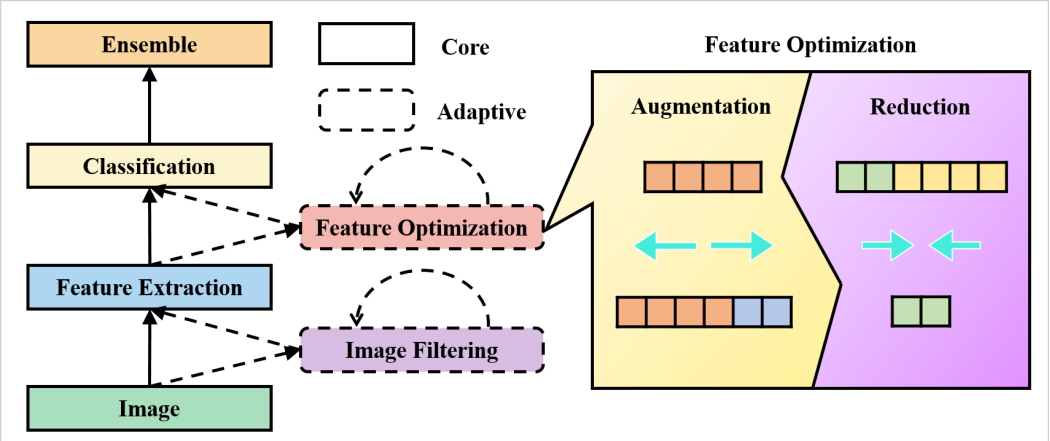


Figure 1. The structure of the Feature Optimization and Dropout in Genetic Programming (FOD-GP) framework, highlighting the distinction between core and adaptive layers. The right section illustrates the process of feature optimization utilizing feature augmentation and reduction.

- Core layers: The core layer is an essential component in FOD-GP, responsible for ensuring the transformation of data types during the classification of raw images. Within each individual, all four core layers must exist for each path from the terminal node to the root node representing the raw image.
  - Image layer: This initial core layer consists of terminal nodes, each representing a single channel of the image. It is responsible for selecting the channel of the image data and forwarding it to subsequent layers.
  - Feature extraction layer: The second core layer extracts meaningful features from the image data. It transforms complex pixel data into a simplified feature vector, which is essential for reducing data dimensions and facilitating subsequent steps.

  - Classification layer: The third core layer utilizes classifiers to perform the actual classification tasks. This crucial decision-making component takes the feature vector from the previous layer and assigns classification labels to each input image.
  - Ensemble layer: The final core layer utilizes the majority-voting technique to combine the outcomes of multiple classifiers, thereby enhancing the robustness and accuracy of the ultimate prediction.
- Adaptive layers: Adaptive layers provide flexibility and enhance data quality. These layers can be applied multiple times or omitted entirely as needed. Unlike core layers, adaptive layers are not essential components because they do not change the data type. Figure 2 illustrates an example tree of the FOD-GP framework, demonstrating the flexibility of adaptive layers. The tree has two image nodes with different paths. Both paths contain one core layer of each of the four types, but the number of adaptive layers varies. The blue path 1 skips the image filtering layer entirely and applies the feature optimization layer twice to iteratively enhance the features. In contrast, the green path 2 applies the image filtering layer twice to enhance the input image and applies the feature optimization layer once. This example highlights the flexible application of adaptive layers based on specific needs.
  - Image filtering layer: The image filtering layer executes various image processing techniques, such as normalization, resizing, and augmentation. These techniques facilitate better feature extraction in subsequent layers by transforming existing pixel values with filters that enhance important features and reduce noise in images.
  - Feature optimization layer: The feature optimization layer dynamically manages and enhances the feature set through augmentation or reduction of the feature vectors generated in the previous layer, as illustrated in Figure 1. Augmentation consists of two techniques. The first is concatenation, which enriches the data by merging multiple feature vectors. The second technique is cascading, which enhances the features by integrating the prediction results of the classifier with the feature vector [15]. Reduction controls the feature space using feature selection techniques. By removing less relevant features, it prevents indiscriminate expansion caused by augmentation and enhances data quality. The proper combination of augmentation and reduction allows the model to work with relevant data, improving accuracy and efficiency, and plays an important role in managing and optimizing feature vectors. As a result, this enhances the performance of the classification layer.

Generic operators: We use two genetic operators, subtree crossover and mutation, to generate new offspring from the parents. Subtree crossover selects two parents and determines which subtrees to exchange. Since there are six types of layers in our proposed method, we exchange subtrees by selecting two parent nodes from the same layer. Subtree mutation selects a single parent and replaces a subtree with an arbitrary subtree that has the same layer type as the parent node.

Fitness evaluation: Fitness evaluation is designed to improve classification accuracy. It is performed using 2-fold cross-validation. After calculating the classification accuracy for each test set, the average of these accuracies is used as the fitness value. This process is defined as shown in Figure 3.

$$fitness = \frac{Acc_1 + Acc_2}{2} \tag{1}$$

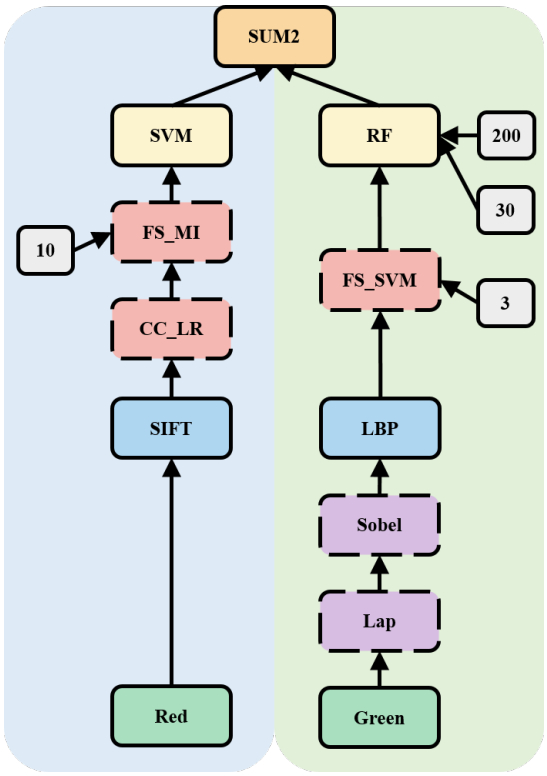


Figure 2. An example of a tree in the FOD-GP framework. The tree represents the flexibility of the adaptive layers through two different paths.

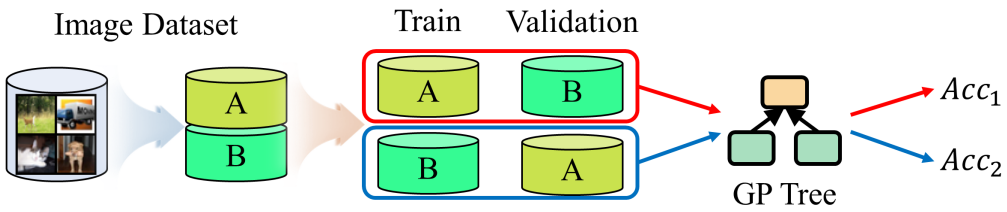


Figure 3. The process of fitness evaluation.

2.3. Adaptive Dropout Implementation in FOD-GP

In addition, we investigated the possibility of applying dropout techniques, traditionally used in neural networks, to improve generalization in GP. Dropout prevents the model from relying excessively on specific patterns by randomly deactivating nodes during training to enhance generalization [24]. However, in FOD-GP, the standard dropout method interfere with the essential structures due to the unique layered design.

Our proposed adaptive dropout techniques are specifically designed for FOD-GP. Within this framework, core layers are essential for preserving the model's stability. Implementing dropout in these core layers may cause instability. To address this issue, adaptive dropout is applied only to adaptive layers, as described in Algorithm 1. During training, if a random value for an adaptive layer is below the threshold, the node is deactivated, allowing the input data to be transferred without being changed to subsequent layers. In all other cases, nodes operate normally, thereby improving the model's adaptability and generalization ability.

Algorithm 1 Adaptive dropout algorithm

Require: f ▷ Function to be applied
Require: X ▷ Input data for f
Require: $r' \leftarrow 0.2$ ▷ Dropout rate

```
if training process then
    if f is part of adaptive layer then
        r ← random value from [0, 1]
        if r < r' then
            return X
        end if
    end if
end if
return f(X)
```

We selected a dropout rate of 0.2 after testing various rates on the CIFAR10(8) to evaluate its impact. The tests, repeated five times for each rate (0.1, 0.2, 0.3, and 0.4), showed the following results: a rate of 0.1 achieved a maximum accuracy of 31.60% (mean: 30.27%), 0.2 achieved a maximum of 34.15% (mean: 31.09%), 0.3 achieved a maximum of 33.33% (mean: 29.75%), and 0.4 achieved a maximum of 31.38% (mean: 28.59%). Based on these results, a dropout rate of 0.2 was selected as it provided the best performance.

Figure 4 shows an example of how adaptive dropout works. This model contains five adaptive layers, to which dropout is applied selectively. In four of these layers, the random values exceed the thresh hold, enabling them to operate normally. However, the random value for the *Sobel* node, r4, is below the threshold, leading to the deactivation of *Sobel*. As a result, the output from the *Lab* node bypasses *Sobel* and is transmitted directly to *LBP*. This mechanism mitigates the model's dependence on the nodes within the adaptive layer, thereby improving its ability to generalize in data-limited environments.

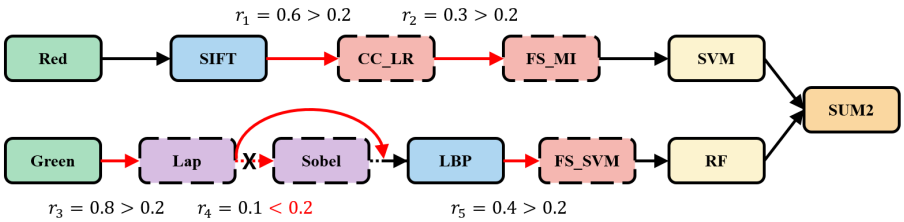


Figure 4. Example of adaptive dropout application in the FOD-GP framework. In this instance, there are five adaptive layers, each with a corresponding random value (r). The *Sobel* node is deactivated as its random value ($r_4 = 0.1$) is below the dropout threshold of 0.2. This results in the direct transmission of *Lap*'s output to *LBP*, bypassing *Sobel*.

2.4. Terminal

The terminal nodes used in the GP trees for our proposed approach are listed in Table 1. The CIFAR10 and SVHN image datasets consist of three distinct channels: red, green, and blue. The color channels represent the basic elements of colors in an image, where each pixel's color is composed of varying ratios of these primary colors. In addition to this scheme, we use a grayscale channel. As a result, the *channel* terminal has four values in total—three original channels and the grayscale channel. Each *channel* terminal node independently selects one of these four channels as input, allowing each individual to extract features from multiple channels across multiple terminal nodes. Conversely, in grayscale image dataset, *channel* nodes only take grayscale channel as input. The *num* terminal is newly added in this work and refers to the desired output size when using the feature selection function. It defines how many features are to be retained in the output after the feature selection process. This terminal helps manage the dimensionality of the data, ensuring that the most relevant features are preserved while reducing noise and

redundancy. The values for this terminal are set to 1, 3, 5, 10, and 30, providing flexibility in feature optimization. The σ terminal represents the standard deviation used in various Gaussian-based functions such as *Gau*, *GauD*, *Gau_Con*, and *GauD_Con*. It defines the spread of the Gaussian function and can vary within the range of 1 to 3. This parameter is crucial for controlling the extent of smoothing or blurring applied to the image data. The o_1 and o_2 terminals represent the order of each axis for the *GauD* and *GauD_Con* functions, ranging from 0 to 2. These orders specify the degree of the derivative applied along each axis, influencing edge detection and texture analysis processes. The θ terminal specifies the orientation for the *Gabor* and *Gabor_Con* functions. The values range from 0 to 7, with each value multiplied by $\pi/8$, enabling different orientations of the Gabor filters to capture various directional features in the image. The f terminal indicates the spatial frequency for the *Gabor* and *Gabor_Con* functions. The values range from 2 to 6, each representing an exponent of $1/\sqrt{2}$. This terminal helps detect patterns and textures at various scales within the image. The n terminal represents the number of trees in the *Random Forest (RF)* and *Extremely Randomized Trees (ERF)* functions. The values range from 50 to 1000 in increments of 50, determining the ensemble size for these tree-based models, which impacts their robustness and generalization capabilities. The d terminal specifies the maximum depth of trees in the *RF* and *ERF* functions. The values range from 10 to 100 in increments of 10, controlling the complexity and potential overfitting of the trees.

Table 1. Terminals.

Terminal	Value	Description
channel	red, green, blue, gray	Channels of the input image
num	1, 3, 5, 10, 30	Desired output size when using the feature selection function
σ	range [1, 3]	Standard deviation for *Gau*, *GauD*, *Gau_Con*, *GauD_Con*
o_1, o_2	range [0, 2]	Order of each axis for *GauD*, *GauD_Con*
θ	range [0, 7]	Orientation for *Gabor*, *Gabor_Con* filters, values multiplied by $\pi/8$
f	range [2, 6]	Spatial frequency for *Gabor*, *Gabor_Con*, values as exponents of $1/\sqrt{2}$
n	range [50, 1000, 50]	Number of trees in *RF*, *ERF* functions
d	range [10, 100, 10]	Maximum depth of trees in *RF*, *ERF* functions

2.5. Function

The set of functions is defined differently, depending on the purpose of each layer. The overall set is summarized in Table 2. The functions of the layers, except for feature optimization, are referenced from existing GPEDL research [15]. The set of image filtering functions consists of *Mean*, *Median*, *Min*, *Max*, *Gau*, *GauD*, *Lap*, *LoG1*, *LoG2*, *Sobel*, *Gabor*, *LBP*, *HOG*, *Sqrt*, *ReLU*, *Add_MaxP* and *Sub_MaxP*. For the *Mean*, *Median*, *Min* and *Max*, 3x3 filters are used. The *Gau* and *GauD* are Gaussian filters that require the standard deviation σ as a parameter. The *GauD* also has the σ_1 and σ_2 parameters for the order along the x/y-axis. The *Lap* is a Laplacian filter that computes the second derivative of an image. The *LoG1* and *LoG2* are Laplacian filters that use Gaussian second derivatives with standard deviations of 1 and 2, respectively. The *Sobel* is used for edge detection. The *LBP* and *HOG* are widely used filters for object detection. The *Sqrt* and *ReLU* apply their operations to each pixel value. The *Add_MaxP* and *Sub_MaxP* functions take two images as input, add or subtract them after applying 2x2 max pooling.

Table 2. Functions.

Layer	Functions
Image filtering	*Mean, Median, Min, Max, Gau, GauD, Lap, LoG1, LoG2, Sobel, Gabor, LBP, HOG, Sqrt, ReLU, Add_MaxP, Sub_MaxP*
Feature extraction	*Con, Hist, HOG, LBP, SIFT, LBP_1d, HOG_1d, Sobel_1d, Gabor_1d, Gau_1d, GauD_1d*
Feature Optimization	*Conca2, Conca3, Conca4, CC_LR, CC_SVM, CC_RF, CC_ERF, FS_MI, FS_Chi, FS_LR, FS_SVM, FS_RF, FS_ERF*
Classification	*LR, SVM, RF, ERF*
Ensemble	*SUM2, SUM3, SUM4*

In the feature extraction layer, the *LBP* and *HOG* functions are designed to extract features associated with LBP and HOG. The *LBP_1d*, *HOG_1d*, *Sobel_1d*, *Gabor_1d*, *Gau_1d*, and *GauD_1d* perform their respective filtering operations and then combine the results into a one-dimensional feature vector. The *Con* function flattens two images into one feature vector. The *Hist* and *SIFT* functions extract histogram and SIFT features from an image, respectively.

In the feature optimization layer, functions are divided into two types. The first type aims to augment the feature vector: *Conca2, Conca3, Conca4, CC_LR, CC_SVM, CC_RF*, and *CC_ERF*. The *Conca2*, *Conca3*, and *Conca4* concatenate 2, 3, and 4 feature vectors into one vector, respectively. The *CC_LR* (cascade logistic regression), *CC_SVM* (cascade support vector machine), *CC_RF* (cascade random forest), and *CC_ERF* (cascade extra trees classifier) are cascade functions that concatenate the results of the classifier with the input data [15]. The second type includes functions with feature selection methods: *FS_MI* (feature selection based on mutual information), *Fs_Chi* (feature selection based on Chi-square), *FS_LR, FS_SVM, FS_RF*, and *FS_ERF*. These functions measure relevance based on each evaluation metric and select the top *num* features. A detailed description of this layer is covered in Algorithm 2. The classification layer includes four classifiers: LR, SVM, RF, and ERF, which provide the predictive probabilities for each class. The ensemble layer utilizes majority voting by combining the results of 2, 3, or 4 classifiers to determine the class with the highest probability.

Algorithm 2 Functions of feature optimization

procedure CONCA$N(X_1, \ldots, X_N)$
  return $Concatenate(X_1, \ldots, X_N)$
end procedure
procedure CC(f, X_{train}, Y_{train}, X_{test})
  Train f with X_{train}, Y_{train}
  $C \leftarrow$ Probability values predicted for X_{test} using the trained f
  return $Conca2(X_{train}, C)$
end procedure
procedure FS(X, Y, *metric*, *num*)
  if *metric* is *MI* or *Chi* **then**
    Calculate the *metric* of X
  else ▷ LR, SVM, RF, ERF
    Train the *metric* with X, Y
    Calculate the feature importance of X
  end if
  $\mathbf{X}' \leftarrow$ Keep only top *num* features depending on the calculated values
  return $\mathbf{X}'$ ▷ The shape of $\mathbf{X} : (N, D) \rightarrow \mathbf{X}' : (N, num)$
end procedure

3. Results and Discussion

3.1. Experiment Design

In this experiment, we evaluated the proposed method on three image classification datasets: CIFAR10 [25], Fashion MNIST (FMNIST) [26], and Street View House Numbers (SVHN) [27]. The CIFAR10 dataset consists of 32 × 32 RGB images divided into 10 classes, with a training set of 50,000 images and a test set of 10,000 images. FMNIST contains 28 × 28 grayscale images divided into 10 classes, totaling 60,000 training images and 10,000 test images. SVHN is a color dataset of 32 × 32 images, containing 73,257 training images and 26,032 test images, distributed across 10 classes. Preprocessing included normalizing pixel values by dividing by 255 to scale them to the range [0,1].

To achieve data-limited image classification, we employed a downsampling algorithm to generate limited data subsets, as illustrated in Algorithm 3. The datasets referred to as CIFAR10(N), FMNIST(N), and SVHN(N) were used for training with sets of 10N images by randomly selecting N samples from each class of the original dataset. Conversely, all test images were used in the evaluation process. The values of N were assigned as 2, 4, 8, 16, and 32. All experiments were conducted 30 times with different random seeds to ensure robust results. For each seed, the evolutionary process explored the best individuals to evaluating their performance. Finally, the best individuals were applied to the test images, yielding the final classification results. This process ensured a robust evaluation of the method under data-limited conditions. The hyperparameters utilized in the proposed method are listed in Table 3.

Algorithm 3 Downsampling algorithm for creating limited data subsets

Require: OriginalDataset, ClassLabels, N
Initialize DownsampledDataset as an empty list
for each label in ClassLabels **do**
    Filter OriginalDataset to get InstancesLabel where instance.label == label
    Shuffle InstancesLabel randomly
    Select N SampledInstances from InstancesLabel
    Add SampledInstances to DownsampledDataset
end for
Shuffle DownsampledDataset randomly
return DownsampledDataset

Table 3. Hyperparameter values for FOD-GP.

Hyperparameter	Value
Generation size	30
Population size	100
Crossover rate	0.5
Mutation rate	0.49
Elitism rate	0.01
Initial depth of tree	2–10
Maximum depth of tree	10

3.2. Results

This section provides a comprehensive evaluation of the existing GPEDL method and the proposed FOD-GP method. Detailed results are summarized in Table 4, which presents both the maximum accuracy and mean accuracy across 30 random seeds for each dataset and sample size. Across all datasets and sample sizes, FOD-GP generally outperforms GPEDL in terms of both maximum and mean accuracy.

Table 4. The classification accuracy results for the GPEDL and the proposed FOD-GP across different sample sizes N (2, 4, 8, 16, 32 per class) for CIFAR10, FMNIST, and SVHN datasets. (**a**) CIFAR10(N); (**b**) FMNIST(N); (**c**) SVHN(N).

(**a**)

		Max (%)					Mean ± SD (%)				
Method	N	2	4	8	16	32	2	4	8	16	32
GPEDL		24.05	28.41	34.71	40.72	44.74	17.41 ±3.01	23.00 ±3.38	28.39 ±4.00	35.46 ±3.74	42.11 ±1.40
FOD-GP		24.59	28.73	36.62	42.06	46.64	17.73 ±2.96	23.22 ±2.99	29.44 ±2.77	36.89 ±3.06	43.29 ±1.69

(**b**)

		Max (%)					Mean ± SD (%)				
Method	N	2	4	8	16	32	2	4	8	16	32
GPEDL		65.40	69.88	74.90	79.21	81.06	54.58 ±5.87	64.39 ±3.11	71.11 ±2.47	76.48 ±1.90	79.65 ±1.12
FOD-GP		67.03	70.96	75.21	79.71	82.11	55.80 ±5.14	64.57 ±3.60	71.66 ±2.58	76.64 ±1.85	80.58 ±0.87

(**c**)

		Max (%)					Mean ± SD (%)				
Method	N	2	4	8	16	32	2	4	8	16	32
GPEDL		31.98	39.04	54.91	63.15	70.95	20.80 ±5.56	31.49 ±6.16	47.58 ±4.67	58.63 ±2.89	66.82 ±1.65
FOD-GP		**37.91**	**42.36**	**58.01**	**71.00**	72.65	**23.31** ±5.62	32.20 ±6.65	**49.58** ±5.89	**61.56** ±3.10	**68.85** ±1.77

On the CIFAR10, both methods show similar performance, with FOD-GP achieving slightly higher accuracies. Specifically, for CIFAR10(8), FOD-GP achieves a maximum accuracy that is 1.91% higher and a mean accuracy 1.05% higher. For CIFAR10(16), the maximum accuracy is 1.34% higher, and the mean accuracy is 1.43% higher. For CIFAR10(32), FOD-GP achieves a maximum accuracy 1.90% higher and a mean accuracy 1.18% higher than GPEDL. Overall, FOD-GP consistently demonstrates good and stable performance across various N values of CIFAR10.

In the case of FMNIST, although the maximum and mean accuracies have improved with FOD-GP, the differences are less pronounced than those observed in CIFAR10. FOD-GP demonstrated a 1.63% enhancement in maximum accuracy and a 1.22% improvement in mean accuracy for FMNIST(2). Furthermore, the maximum accuracy for FMNIST(4) and FMNIST(32) increases by 1.08% and 1.05%, respectively. These results indicate that while the improvements for FMNIST are modest, FOD-GP still demonstrates consistent enhancements over GPEDL.

For SVHN, FOD-GP demonstrates the most substantial improvements in both maximum and mean accuracy compared to GPEDL; a maximum accuracy 5.93% higher (37.91% vs. 31.98%) and a mean accuracy 2.51% higher (23.31% vs. 20.80%) in SVHN(2), a maximum accuracy 3.32% higher (42.36% vs. 39.04%) and a mean accuracy 0.71% higher (32.20% vs. 31.49%) in SVHN(4), a maximum accuracy 3.10% higher (58.01% vs. 54.91%) and a mean accuracy 2.00% higher (49.58% vs. 47.58%) in SVHN(8), a maximum accuracy 7.85% higher (71.00% vs. 63.15%) and a mean accuracy 2.93% higher (61.56% vs. 58.63%) in SVHN(16), a maximum accuracy 1.70% higher (72.65% vs. 70.95%) and a mean accuracy 2.03% higher (68.85% vs. 66.82%) in SVHN(32). FOD-GP shows superior performance across all SVHN(N), indicating significant improvements over GPEDL. In summary, the proposed FOD-GP method consistently outperforms the existing GPEDL method across all

datasets and sample sizes. The improvements are modest on CIFAR10 and FMNIST but become substantial on SVHN.

To statistically validate the differences in performance between FOD-GP and EDLGP, we used a series of statistical tests. The Shapiro–Wilk test was initially used to assess the normality of the distribution [28]. If the data followed a normal distribution, statistical tests such as the paired *t*-test or Welch's test were used to compare the methods. The purpose of the paired *t*-test is to evaluate whether there is a significant difference between the means of two groups. Welch's test is a modified version of the *t*-test that is suitable for comparing two samples with unequal variances [29]. Before applying these tests, the Levene test was used to examine the equality of variances [30]. If it was determined that the variances were equal, the standard paired *t*-test was utilized; otherwise, Welch's test, which does not assume equal variances, was employed. In cases where the assumption of normality was violated, the Wilcoxon signed-rank test was used. The Wilcoxon signed-rank test assesses the differences between two paired groups when the data are not normally distributed. Figure 5 illustrates the statistical analysis procedure employed in this study.

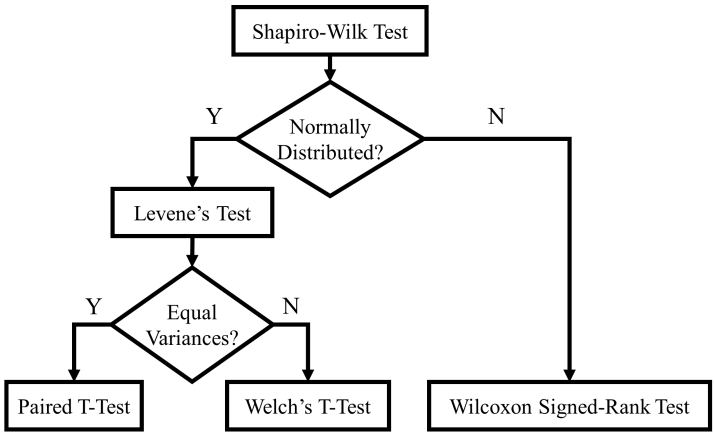


Figure 5. Statistical analysis process used to validate performance differences.

The results of these statistical tests are summarized in Table 5. For both CIFAR10(N) and FMNIST(N), the Shapiro–Wilk *p*-values exceed 0.05 for all X, indicating that the data are normally distributed. In the case of SVHN(N), the *p*-values for SVHN(2), SVHN(4), SVHN(8), and SVHN(16) exceed 0.05, indicating normality. However, for SVHN(32), the *p*-value is 0.033, indicating a departure from normality. Consequently, only SVHN(32) was excluded from Levene's test. In Levene's test, the *p*-value was also compared against 0.05. If the *p*-value is greater than 0.05, the paired *t*-test is applied because the assumption of equal variance is established. If the *p*-value is less than 0.05, Welch's test is applied as it accounts for unequal variance. For SVHN(32), the final *p*-value was calculated using the Wilcoxon signed-rank test due to the violation of normality.

The analysis of p-values for CIFAR10(N) shows that, regardless of N, there is no statistically significant difference between FOD-GP and EDLGP. Similarly, the results for FMNIST(N) indicate no significant difference. However, for SVHN(N), the *p*-values are less than 0.05 for SVHN(4), SVHN(8), SVHN(16), and SVHN(32), indicating that FOD-GP performs significantly better than GPEDL for these sample sizes. This enhanced performance on SVHN may be related to the dataset's unique characteristics. SVHN images contain multiple digits and complex backgrounds, which can make it challenging for models to identify the essential features. The feature optimization and adaptive dropout techniques of FOD-GP can mitigate this issue by allowing the model to prioritize relevant features. These results suggest that FOD-GP provide advantages in handling visual structures like SVHN, but further testing on similar datasets would be necessary to determine if these advantages are consistent across different data conditions.

Table 5. Statistical analysis results comparing the proposed FOD-GP with the GPEDL across different datasets and sample sizes.

Dataset	N	Shapiro–Wilk *p*-Value	Levene's *p*-Value	Test Used	*p*-Value	Significance ($p < 0.05$)
CIFAR10(N)	2	0.078	0.000	Welch	0.685	No
	4	0.910	0.000	Welch	0.403	No
	8	0.404	0.000	Welch	0.092	No
	16	0.465	0.087	Paired	0.797	No
	32	0.893	0.632	Paired	0.835	No
FMNIST(N)	2	0.182	0.002	Welch	0.674	No
	4	0.898	0.001	Welch	0.251	No
	8	0.092	0.016	Welch	0.407	No
	16	0.810	0.193	Paired	0.157	No
	32	0.321	0.315	Paired	0.117	No
SVHN(N)	2	0.084	0.000	Welch	0.742	No
	4	0.745	0.000	Welch	**0.000**	**Yes**
	8	0.693	0.006	Welch	**0.005**	**Yes**
	16	0.189	0.002	Welch	**0.001**	**Yes**
	32	0.033	-	Wilcoxon	**0.000**	**Yes**

To further evaluate FOD-GP's robustness, we also performed comparisons with various deep learning models, including ResNet [31], CNTK with 5, 8, 11, and 14 layers [32], and Harmonic Networks [33], to demonstrate the robustness of FOD-GP. Table 6 presents the accuracy results on CIFAR10(N), with results for ResNet, CNTK, and Harmonic Networks referenced from [15]. This comparison aims to highlight FOD-GP's ability to perform competitively with deep learning models in data-limited scenarios. FOD-GP consistently outperforms the other models across all subsets except CIFAR10(32), demonstrating its adaptability and robustness in data-limited scenarios. This performance is achieved by FOD-GP's feature optimization and adaptive dropout techniques, enabling it to effectively capture critical patterns. In contrast to deep learning models, which encounter difficulties with generalization, FOD-GP facilitates adaptable model structures that are effective in data-limited scenarios.

Table 6. Accuracy comparison of FOD-GP with various deep learning models (ResNet, CNTK with 5, 8, 11, 14 layers, and Harmonic Networks) on CIFAR10(N).

Method	CIFAR10(2)	CIFAR10(4)	CIFAR10(8)	CIFAR10(16)	CIFAR10(32)
ResNet	17.50	19.52	23.32	28.30	33.15
5-layer CNTK	18.03	20.83	24.82	29.63	35.26
8-layer CNTK	18.50	21.07	25.18	30.17	36.05
11-layer CNTK	18.69	21.23	25.40	30.46	36.44
14-layer CNTK	18.79	21.34	25.48	30.48	36.57
Harmonic Networks	22.24	26.28	34.94	40.47	**49.59**
GPEDL(Max)	24.05	28.41	34.71	40.72	44.74
FOD-GP(Max)	**24.59**	**28.73**	**36.62**	**42.06**	46.64

For CIFAR10(32), however, the Harmonic Networks model achieves the highest accuracy (49.59%), exceeding FOD-GP's performance of 46.64%. This result highlights the potential of deep learning models when more training data are available. However, given the significant improvement compared to CIFAR10(16), this may reflect a temporary benefit specific to dataset size. Further testing on multiple datasets will be needed to evaluate the performance.

To extend the evaluation, we conducted experiments on an additional dataset, ORL(2). The ORL [34] dataset is well-known benchmark for face classification, consisting of 400 images distributed across 40 classes. Due to its relatively small size, the dataset has been widely used in data-limited studies. This dataset provides an ideal test case for FOD-GP, enabling

the evaluation of the model's generalization ability in data-limited scenarios. Over five iterations, FOD-GP achieved a mean accuracy of 88.00% ± 4.60%, with a maximum accuracy of 94.69%. In contrast, GPEDL achieved a mean accuracy of 86.62% ± 3.98%, with a maximum accuracy of 93.13%. Furthermore, we compared KCNN [35], a deep metric learning-based model integrating CNN and k-nearest neighbor, which achieved 72.8% ± 3.1%, as reported in [15]. The results demonstrate that FOD-GP effectively handles small datasets such as ORL, outperforming both GPEDL and KCNN.

Finally, to evaluate the structural complexity of FOD-GP compared to GPEDL, we analyzed the number of nodes in the best individuals on CIFAR10(N), FMNIST(N), and SVHN(N). Figure 6 illustrates a boxplot summarizing the distribution of node counts for the two methods. The results indicate that FOD-GP generates trees with higher node counts than GPEDL. While GPEDL achieves structural simplicity, FOD-GP explores deeper and more complex tree solutions. We present two example trees of the best individuals extracted from the SVHN(16) in Figure 7. In these examples, GPEDL achieves its optimal performance without utilizing adaptive layers, while FOD-GP extensively incorporates adaptive layers, enhancing exploration through adaptive dropout, resulting in a performance improvement (63.15% for GPEDL vs. 71.00% for FOD-GP). The structural complexity of FOD-GP, though higher than that of GPEDL, leads to better generalization and performance. This comparative analysis of number of nodes and example trees highlights the advantages of FOD-GP's flexibility. The higher number of nodes in FOD-GP trees suggests that FOD-GP is capable of generating more complex representations, leveraging adaptive layers to capture a broader range of features effectively.

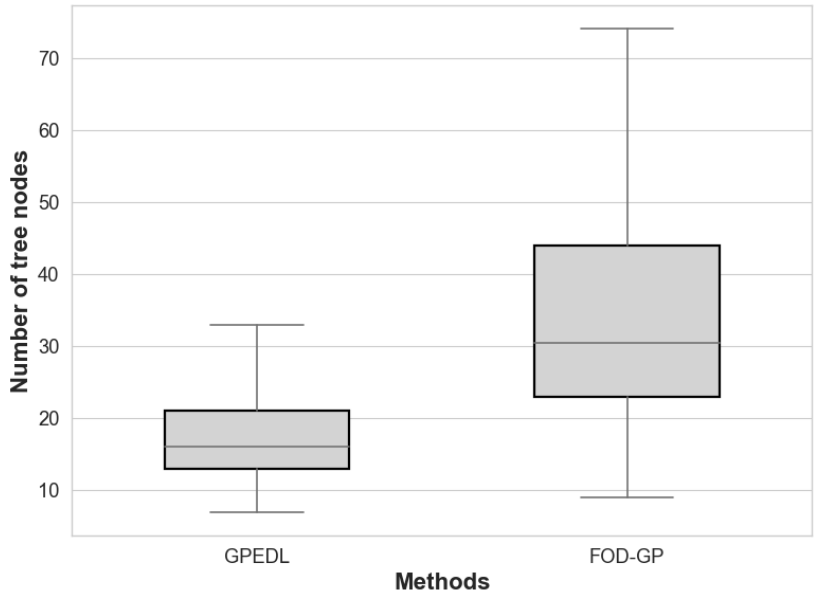


Figure 6. Comparing the number of nodes in trees generated by GPEDL and FOD-GP, based on the best individuals across CIFAR10(N), FMNIST(N), and SVHN(N).

In conclusion, comprehensive evaluations across CIFAR10, FMNIST, and SVHN consistently demonstrate the advantages of FOD-GP compared to the existing GPEDL method. While improvements on CIFAR10 and FMNIST are modest, FOD-GP achieves significant performance improvements on SVHN. The statistical and structural complexity analyses underscore the benefits of FOD-GP's adaptive dropout and feature optimization techniques, enabling it to generate more complex and effective tree structures than GPEDL. These results highlight FOD-GP's generalization ability and robustness, especially in challenging classification tasks with limited data.

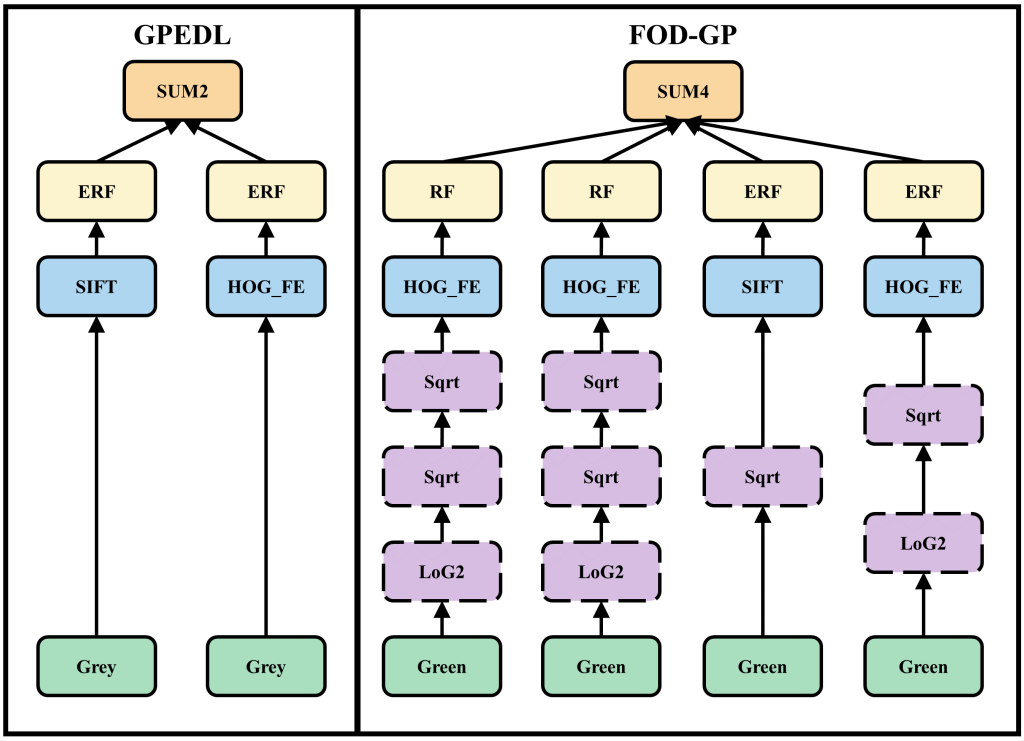


Figure 7. Example trees of the best individuals on SVHN(16) for GPEDL and FOD-GP.

3.3. Discussion

The experimental results indicate that the proposed FOD-GP method generally outperforms the existing GPEDL method. Although only 4 out of 15 cases show statistically significant improvements, these results are still meaningful. The significant improvements observed on the SVHN dataset suggest that FOD-GP is particularly well-suited to certain datasets, offering valuable insights for future research. Even in cases where the differences are not statistically significant, FOD-GP consistently improves both the maximum and mean accuracy across all datasets and sample sizes. This consistency highlights the robustness of FOD-GP and suggests that even slight increases in accuracy can provide meaningful benefits in real-world applications, particularly in data-limited environments.

We present a detailed analysis of the training accuracy comparison between FOD-GP and GPEDL. Table 7 summarizes the training accuracies for both methods. FOD-GP consistently outperforms GPEDL in terms of training accuracy across all CIFAR10(N), FMNIST(N), and SVHN(N). This indicates that FOD-GP is more effective during training because of the application of feature optimization and adaptive dropout techniques. Although the training accuracy difference between the two methods decreases as the sample size (N) increases, FOD-GP still consistently performs better. However, the improvement in test accuracy of FOD-GP (Table 4) is not as significant as the improvement in training accuracy. While FOD-GP demonstrates superior learning capabilities on training data, its ability to generalize to unseen data are only slightly better than GPEDL. Both methods show that the gap between training and test accuracy decreases with an increasing sample size, indicating that a larger dataset offers improved learning opportunities and reduces overfitting.

These results demonstrate that FOD-GP is exceptionally efficient during training, indicating the necessity for further enhancements to improve its generalization capability. Addressing this challenge will be crucial for enhancing the model's ability to generalize. Further research will focus on refining the feature optimization process and incorporating additional regularization techniques to mitigate overfitting. Overall, the trend of increasing accuracy with larger sample sizes and decreasing overfitting reemphasizes the importance of having sufficient data.

Table 7. Training accuracies for GPEDL and FOD-GP across CIFAR10, FMNIST, and SVHN datasets. (**a**) CIFAR10(N); (**b**) FMNIST(N); (**c**) SVHN(N).

(**a**)						
		Training Accuracy (%)				
Method	N	2	4	8	16	32
GPEDL		41.17 ± 7.49	36.92 ± 5.50	35.17 ± 5.53	38.42 ± 4.71	41.73 ± 3.20
FOD-GP		**47.83** ± 9.28	**41.17** ± 6.32	**39.29** ± 5.23	**41.29** ± 4.48	**45.04** ± 2.95
(**b**)						
		Training Accuracy (%)				
Method	N	2	4	8	16	32
GPEDL		66.00 ± 12.07	73.67 ± 6.79	74.46 ± 5.76	77.94 ± 3.62	81.06 ± 2.45
FOD-GP		**73.00** ± 12.36	**77.58** ± 7.68	**77.67** ± 5.37	**79.71** ± 3.15	**82.40** ± 2.51
(**c**)						
		Training Accuracy (%)				
Method	N	2	4	8	16	32
GPEDL		40.50 ± 10.90	41.00 ± 7.21	45.29 ± 6.09	53.63 ± 4.66	61.28 ± 2.72
FOD-GP		**49.00** ± 11.21	**44.50** ± 7.26	**50.83** ± 5.47	**58.33** ± 5.24	**64.95** ± 2.80

These results demonstrate that FOD-GP is exceptionally efficient during training, indicating the necessity for further enhancements to improve its generalization capability. Addressing this challenge will be crucial for enhancing the model's ability to generalize. Further research will focus on refining the feature optimization process and incorporating additional regularization techniques to mitigate overfitting. Overall, the trend of increasing accuracy with larger sample sizes and decreasing overfitting reemphasizes the importance of having sufficient data.

4. Conclusions

The purpose of this study is to introduce the FOD-GP framework, designed to address the challenges of image classification with limited data. FOD-GP leverages the flexibility of Genetic Programming (GP) to automatically develop models, while integrating feature optimization and adaptive dropout to address the issues of poor performance and inefficient feature usage in GP-based EDL. The experimental results on the CIFAR10, FMNIST, and SVHN datasets consistently demonstrate that FOD-GP outperforms existing methods, particularly in terms of training accuracy in data-limited scenarios. Significant improvements in classification accuracy for certain cases highlight FOD-GP's potential for achieving better generalization, making it an effective approach in real-world applications where efficient data utilization and model interpretability are essential.

The comparative structural complexity analysis further highlights FOD-GP's adaptability, demonstrating that it generates more efficient tree structures than GPEDL by effectively utilizing adaptive layers. However, the modest improvement in classification accuracy compared to the considerably enhanced training accuracy indicates that overfitting remains a concern. Furthermore, FOD-GP has only been tested on a limited number of datasets, emphasizing the need for further validation in various datasets. Future research will focus on refining the feature optimization process and exploring additional normalization methods to improve the generalization capabilities of FOD-GP on unseen data. In conclusion, FOD-GP combines the flexibility of GP with feature optimization and adaptive dropout to make significant advancements in data-limited image classification. This approach effectively addresses the issue of image classification in situations where there is a limited amount of data available and sets the foundation for future advancements in data-limited and interpretable machine learning.

Author Contributions: Conceptualization, C.M.L.; methodology, C.M.L.; software, C.M.L.; validation, C.M.L., C.W.A. and M.-J.K.; formal analysis, C.W.A.; investigation, M.-J.K.; resources, C.W.A.; data curation, C.M.L.; writing—original draft preparation, C.M.L.; writing—review and editing, C.W.A. and M.-J.K.; visualization, C.M.L.; supervision, C.W.A. and M.-J.K.; project administration, C.W.A.; funding acquisition, C.W.A. and M.-J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIT, Ministry of Science and ICT) (RS-2024-00347902) and the Ministry of Education (RS-2023-00247900), and the Institute of Information & communications Technology Planning & Evaluation (IITP) under the Artificial Intelligence Convergence Innovation Human Resources Development (IITP-2023-RS-2023-00256629) grant funded by the Korea government (MSIT) and ITRC (Information Technology Research Center) support program (IITP-2024-RS-2024-00437718).

Data Availability Statement: The data presented in this study are openly available in [25–27].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Huang, S.C.; Pareek, A.; Jensen, M.; Lungren, M.P.; Yeung, S.; Chaudhari, A.S. Self-supervised learning for medical image classification: A systematic review and implementation guidelines. *NPJ Digit. Med.* **2023**, *6*, 74. [CrossRef] [PubMed]
2. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
3. Chen, L.; Li, S.; Bai, Q.; Yang, J.; Jiang, S.; Miao, Y. Review of image classification algorithms based on convolutional neural networks. *Remote Sens.* **2021**, *13*, 4712. [CrossRef]
4. Irfan, M.; Hameed, I.A. Deep learning based classification for healthcare data analysis system. In Proceedings of the 2017 International Conference on Behavioral, Economic, Socio-cultural Computing (BESC), Krakow, Poland, 16–18 October 2017; pp. 1–6.
5. Wang, W.; Liang, D.; Chen, Q.; Iwamoto, Y.; Han, X.H.; Zhang, Q.; Hu, H.; Lin, L.; Chen, Y.W. Medical image classification using deep learning. In *Deep Learning in Healthcare: Paradigms and Applications*; Springer Nature: New York, NY, USA, 2020; pp. 33–51.
6. Chan, K.Y.; Lam, H.K.; Jiang, H. A genetic programming-based convolutional neural network for image quality evaluations. *Neural Comput. Appl.* **2022**, *34*, 15409–15427. [CrossRef]
7. Rasi, D.; Deepa, S. Hybrid optimization enabled deep learning model for colour image segmentation and classification. *Neural Comput. Appl.* **2022**, *34*, 21335–21352. [CrossRef]
8. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef]
9. Razzak, M.I.; Naz, S.; Zaib, A. Deep learning for medical image processing: Overview, challenges and the future. In *Classification in BioApps: Automation of Decision Making*; Springer: Cham, Switzerland, 2018; pp. 323–350.
10. Salehi, A.W.; Khan, S.; Gupta, G.; Alabduallah, B.I.; Almjally, A.; Alsolai, H.; Siddiqui, T.; Mellit, A. A study of CNN and transfer learning in medical imaging: Advantages, challenges, future scope. *Sustainability* **2023**, *15*, 5930. [CrossRef]
11. Tajbakhsh, N.; Roth, H.; Terzopoulos, D.; Liang, J. Guest editorial annotation-efficient deep learning: The holy grail of medical imaging. *IEEE Trans. Med Imaging* **2021**, *40*, 2526–2533. [CrossRef]
12. Brigato, L.; Barz, B.; Iocchi, L.; Denzler, J. Image classification with small datasets: Overview and benchmark. *IEEE Access* **2022**, *10*, 49233–49250. [CrossRef]
13. Bruintjes, R.J.; Lengyel, A.; Rios, M.B.; Kayhan, O.S.; Zambrano, D.; Tomen, N.; van Gemert, J. VIPriors 4: Visual Inductive Priors for Data-Efficient Deep Learning Challenges. *arXiv* **2024**, arXiv:2406.18176.
14. Espejo, P.G.; Ventura, S.; Herrera, F. A survey on the application of genetic programming to classification. *IEEE Trans. Syst. Man, Cybern. Part C Appl. Rev.* **2009**, *40*, 121–144. [CrossRef]
15. Bi, Y.; Xue, B.; Zhang, M. Genetic programming-based evolutionary deep learning for data-efficient image classification. *IEEE Trans. Evol. Comput.* **2022**, *28*, 307–322. [CrossRef]
16. Evans, B.; Al-Sahaf, H.; Xue, B.; Zhang, M. Evolutionary deep learning: A genetic programming approach to image classification. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–6.
17. Fan, Q.; Bi, Y.; Xue, B.; Zhang, M. Evolving effective ensembles for image classification using multi-objective multi-tree genetic programming. In *Australasian Joint Conference on Artificial Intelligence*; Springer: Cham, Switzerland, 2022; pp. 294–307.
18. Fan, Q.; Bi, Y.; Xue, B.; Zhang, M. A genetic programming-based method for image classification with small training data. *Knowl.-Based Syst.* **2024**, *283*, 111188. [CrossRef]
19. Bi, Y.; Liang, J.; Xue, B.; Zhang, M. A genetic programming approach with building block evolving and reusing to image classification. *IEEE Trans. Evol. Comput.* **2023**, *28*, 1366–1380. [CrossRef]

20. Ahadzadeh, B.; Abdar, M.; Safara, F.; Khosravi, A.; Menhaj, M.B.; Suganthan, P.N. Sfe: A simple, fast, and efficient feature selection algorithm for high-dimensional data. *IEEE Trans. Evol. Comput.* **2023**, *27*, 1896–1911. [CrossRef]
21. Tan, P.; Wang, X.; Wang, Y. Dimensionality reduction in evolutionary algorithms-based feature selection for motor imagery brain-computer interface. *Swarm Evol. Comput.* **2020**, *52*, 100597. [CrossRef]
22. Garbin, C.; Zhu, X.; Marques, O. Dropout vs. batch normalization: An empirical study of their impact to deep learning. *Multimed. Tools Appl.* **2020**, *79*, 12777–12815. [CrossRef]
23. Park, J.G.; Choi, E.S.; Kang, M.S.; Jung, Y.G. Dropout Genetic Algorithm Analysis for Deep Learning Generalization Error Minimization. *Int. J. Adv. Cult. Technol.* **2017**, *5*, 74–81.
24. Omar, A.; Abd El-Hafeez, T. Optimizing epileptic seizure recognition performance with feature scaling and dropout layers. *Neural Comput. Appl.* **2024**, *36*, 2835–2852. [CrossRef]
25. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. 2009. Available online: https://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf (accessed on 25 October 2024).
26. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.
27. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. Reading digits in natural images with unsupervised feature learning. In Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, 12–17 December 2011; Volume 2011, p. 4.
28. Shapiro, S.S.; Wilk, M.B. An analysis of variance test for normality (complete samples). *Biometrika* **1965**, *52*, 591–611. [CrossRef]
29. Welch, B.L. The generalization of 'STUDENT'S'problem when several different population varlances are involved. *Biometrika* **1947**, *34*, 28–35. [PubMed]
30. Brown, M.B.; Forsythe, A.B. Robust tests for the equality of variances. *J. Am. Stat. Assoc.* **1974**, *69*, 364–367. [CrossRef]
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
32. Arora, S.; Du, S.S.; Li, Z.; Salakhutdinov, R.; Wang, R.; Yu, D. Harnessing the power of infinitely wide deep nets on small-data tasks. *arXiv* **2019**, arXiv:1910.01663.
33. Ulicny, M.; Krylov, V.A.; Dahyot, R. Harmonic networks with limited training samples. In Proceedings of the 2019 27th European Signal Processing Conference (EUSIPCO), Coruña, Spain, 2–6 September 2019; pp. 1–5.
34. Samaria, F.S.; Harter, A.C. Parameterisation of a stochastic model for human face identification. In Proceedings of the 1994 IEEE Workshop on Applications of Computer Vision, Sarasota, FL, USA, 5–7 December 1994; pp. 138–142.
35. Liao, T.; Lei, Z.; Zhu, T.; Zeng, S.; Li, Y.; Yuan, C. Deep metric learning for k nearest neighbor classification. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 264–275.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.