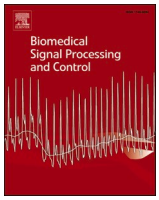
Biomedical Signal Processing and Control 109 (2025) 108035

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

journal homepage: www.elsevier.com/locate/bspc



# Classification of direct microscopic fungi images using optimized graph networks

Pooya Sajjadi [a], Farshid Hajati [b,c,*], Alireza Rezaee [a], Shahadat Uddin [d]

[a] Department of Mechatronics, School of Intelligent Systems, College of Interdisciplinary Science and Technology, University of Tehran, Tehran, Iran
[b] School of Science and Technology, Faculty of Science, Agriculture, Business and Law, University of New England, Armidale, NSW 2350, Australia
[c] College of Arts, Business, Law, Education and IT, Victoria University, Melbourne, VIC 3011, Australia
[d] School of Project Management, Faculty of Engineering, The University of Sydney, NSW 2050, Australia

A B S T R A C T

Fungal infections pose a significant threat to immunocompromised patients due to their variable antifungal susceptibility and the limitations of conventional identification methods. Current approaches relying on microscopy, histopathology, and culture demand specialized training, often leading to delays and increased computational costs. To overcome these limitations, we introduce a deep learning approach for identifying uncultured fungi patches. Our model leverages optimized graph-based neural networks to classify images of direct microscopic examination, addressing the challenges of low data availability and lengthy identification processes. To further enhance the model's performance, its hyperparameters are optimized using various swarm-based optimization methods. Our model can potentially eliminate the need for expert mycologists and lengthy biochemical identification processes. By employing few-shot learning and hyperparameter optimization, the model achieves remarkable accuracy in identifying direct, uncultured fungal images, reducing identification time from 10-14 days to hours. To the best of our knowledge, our approach achieves the highest accuracy on the DeFungi (93.1 %) and DIFaS (100 %) datasets, surpassing the closest benchmark methods, which report accuracies of 86.8 % on DeFungi and 93.9 % on DIFaS.

## 1. Introduction

Fungal and yeast infections significantly cause mortality in patients with weakened immune systems. Rapidly classifying fungal infections from clinical specimens is crucial due to their inherently variable antifungal susceptibility profile and the growing number of variations not covered by conventional commercial kits [1,2]. Various approaches have been used to date for fungal classification, such as potassium hydroxide (KOH) preparation, histopathological examination with periodic acid-Schiff stain (PAS), culture, and polymerase chain reaction (PCR) amplification [3,4]. Among these, PCR has shown the most accurate outcome [5,6].

However, the classification of fungal infections still heavily relies on direct microscopic examination of clinical specimens, histopathology, and culture, which require specialized mycology training. The visual similarity and the increasing number of fungi to be classified have prompted mycologists to develop new methods to identify fungi. Chemical and molecular methods and antigen detection as an alternative to culture in diagnosing fungal diseases are increasingly emphasized, leading to delays in diagnosis, prolonged recovery time, and increased costs [7].

To address these challenges, we propose a deep learning (DL) model that uses optimized graph-based neural networks to classify direct, uncultured fungal images. We optimized the model's hyperparameters using multiple swarm-based optimization methods such as Ant Colony Optimizer (ACO) [8], Bald Eagle Search (BES) [9], Grey Wolf Optimizer (GWO)[10], Harris Hawks Optimization (HHO) [11], Marine Predator Algorithm (MPA) [12], Particle Swarm Optimization (PSO) [13], Sail Fish Optimizer (SFO) [14], and Whale Optimization Algorithm (WOA) [15].

This model can eliminate the need for a mycology expert and further experiments on the cultured specimen, reducing the classification process's time and cost. The entire process consists of collecting test materials, such as swabs and urine samples, and processing a picture taken by a microscope equipped with a camera.

### 1.1. Related Works

There are various approaches to utilizing DL models to classify medical images. These include training a deep convolutional neural

* Corresponding author at: School of Science and Technology, Faculty of Science, Agriculture, Business and Law, University of New England, Armidale, NSW 2350, Australia. Also, at the College of Arts, Business, Law, Education and IT, Victoria University, Melbourne, VIC 3011, Australia
*E-mail addresses:* pooyasajjadi@ut.ac.ir (P. Sajjadi), fhajati@une.edu.au, farshid.hajati@vu.edu.au (F. Hajati), arrezaee@ut.ac.ir (A. Rezaee), shahadat.uddin@sydney.edu.au (S. Uddin).

https://doi.org/10.1016/j.bspc.2025.108035
Received 13 April 2024; Received in revised form 15 February 2025; Accepted 29 April 2025
Available online 3 May 2025

network (CNN) backbone from scratch, using a pre-trained CNN on an extensive image database such as ImageNet or transfer learning, and pretraining the CNN on natural or medical images in a manner of unsupervised learning, followed by fine-tuning on medical target images [16].

In [17], supervised transfer learning was conducted on a total of 176 images in 9 classes of cultured and gram stained fungi from the DIFaS dataset [17]. The performance of various well-known deep networks, such as pre-trained AlexNet, Inception V3, and ResNet, using a Support Vector Machine (SVM) as the classifier was measured, resulting in a maximum patch-based accuracy of 82.4 %. In [18], the same experiment was conducted on open-source conidial fungi data from the Department of Health-Philmech [18]. With 1,152 images from nine different classes of cultured fungi, the proposed method achieved an accuracy of 93.3 %, using a Multi-Layer Perceptron (MLP) as the classifier.

In another study, Quezada et. Al. [19] introduced RaViTT, a random patch sampling strategy for Vision Transformer (ViT) models that improves image classification accuracy in datasets like ImageNet-1 k and CIFAR-100, outperforming state-of-the-art augmentation techniques significantly. The proposed approach incorporates random patch sampling into existing ViTs, and they achieved a classification accuracy of 86.8 % on the DeFungi dataset. In the paper, larger images are randomly broken down to smaller random patches and fed to the vision transformers with the performance metrics (Top-1 accuracy) being calculated based on the classification of the whole image rather than individual patches.

In [20], the researchers utilized the DeFungi dataset, donated by a mycology laboratory, which was labelled and preprocessed with expert assistance. Images were cropped into 500x500 pixel patches, filtering out irrelevant sections, noise, and artifacts. Three CNN architectures—ResNet50, Inception V3, and VGG16—were trained and tested under two conditions: training from scratch and employing transfer learning based on pre-trained ImageNet models. The models' performance was evaluated using k-fold cross-validation, with the highest accuracy achieved by VGG16 at 85.04 % when using transfer learning. This study provides initial benchmarks for early-stage fungi classification through direct microscopic examination.

The success of DL models is driven by the increase in the computational power and availability of an extensive amount of labelled data [21]. Most medical imaging datasets lack the number of labelled samples. [22,23]. Few-shot learning aims to create a classifier to recognize unseen data classes from a few labelled examples. Although data augmentation and generalization algorithms help reduce over-fitting, they cannot eliminate it [24]. Inspired by the ability to generalize in the human brain and disappointed with the performance of CNNs in generalization, few and zero-shot learning has become vastly popular in recent years [22,23,24,25,26,27,28].

In [22], the Bayesian One-Shot algorithm was proposed to classify newly seen classes by taking advantage of previously learned categories with the help of the Bayesian probability distribution. Similarly, in [23], a generative model for one-shot learning was proposed to identify handwritten characters.

There are many ways to implement few-shot learning, such as *meta*-learning [29], metric learning [30] and, Graph neural network (GNN) [27]. Graph neural networks were mainly designed and used to process graph-structured data [31]. There is now a trend to utilize these powerful models to process other data types, such as images [32]. For instance, Few-Shot GNN is proposed in [25], which is a graph network where each node represents concatenated feature vectors (extracted by a CNN backbone) and corresponding class labels. Node features are updated with the propagation of label information through the graph network. Edge-labeling graph neural network (EGNN) is introduced in [26] which by updating the edge labels, enables the formation of explicit clustering with direct utilization of both intra-cluster similarity and inter-cluster dissimilarities.

In contrast with GNN and EGNN, Distribution Propagation Graph Network (DPGN) [27] employs a dual graph architecture where each node stands as an example. Equipped with a point graph and a distribution graph, DPGN represents instance-level and distribution-level similarities, respectively. Updating the graph is made possible by propagating label information from labeled to unlabeled examples.

Due to the complexity of medical imaging data, a complex model is required, but the limited availability of labelled samples may lead to overfitting. Deep neural networks have undoubtedly revolutionized various fields with their remarkable ability to learn complex patterns from data. However, their effectiveness hinges on carefully tuned hyperparameters, which govern the network architecture and optimization procedures, directly influencing training dynamics and model performance [33]. In the past, hyperparameter selection was often a manual and time-consuming process, as researchers had to rely on limited trial options. Nevertheless, with the advent of powerful computer clusters and GPU processors, conducting more extensive trials and exploring diverse optimization methods has become feasible. While conventional optimizers can effectively compute hyperparameters, the choice of which optimizer to employ remains an open question.

In [34], different algorithms such as Sequential Model-based Global Optimization, The Gaussian Process Approach, and Random Search for Hyper-Parameter Optimization in DBNs are presented. In [10], Grey Wolf Optimizer (GWO) is proposed as a *meta*-heuristic optimization technique inspired by the leadership and hunting mechanism of Canis Lupus in nature. The GWO has proven successful in different fields, including machine learning [35]. In [36], GWO is utilized for tuning sigma and gamma hyperparameters in SVM to classify intracranial electroencephalogram signals. In another paper [37], GWO is used to tune hyperparameters of the least square SVM (LSSVM). Their work compared GWO, grid search, and artificial bee colony (ABC) using cross-validation. Their experiments showed that the performance of the GWO-LSSVM was significantly higher compared to the other two approaches.

In [38], GWO is applied to optimize weights and biases of a single hidden MLP network, the most popular neural network type. The training approach was benchmarked against other evolutionary trainers, including particle swarm optimization (PSO), genetic algorithm (GA), ant colony optimization (ACO), and population-based incremental learning (PBIL). The experiments showed the advantage of using GWO.

To summarize, the contributions of this paper are as follows:

- Introduction of a DL approach for direct, uncultured pathogenic fungi identification and potentially eliminating the need for a mycology expert
- Utilization of few-shot learning techniques integrated with optimized graph neural networks to address overfitting and enhance generalization capabilities
- Optimization and benchmarking of model hyperparameters employing a variety of metaheuristic optimizers to further boost performance
- Contrast with previous research methodologies, exemplified by [17,18], which primarily focused on deep models and their application in the classification of cultured patches of fungi, whereas our study addresses directly sampled fungi
- Demonstrated superiority in accuracy compared to current state-of-the-art models, as evidenced by experimental evaluation to the best of our knowledge

## 2. Method

Fig. 1 illustrates the overall methodology. The proposed model comprises a deep feature extractor, a feature processing block, and a graph neural network classifier. To overcome the vanishing gradient problem, a deep CNN is used as the feature extractor block, which is comprised of 16 bottleneck residual blocks inspired by the ResNet-50 architecture [39]. Influenced by the VGG network [40], the depth of convolutional layers doubles after each block when the feature map is

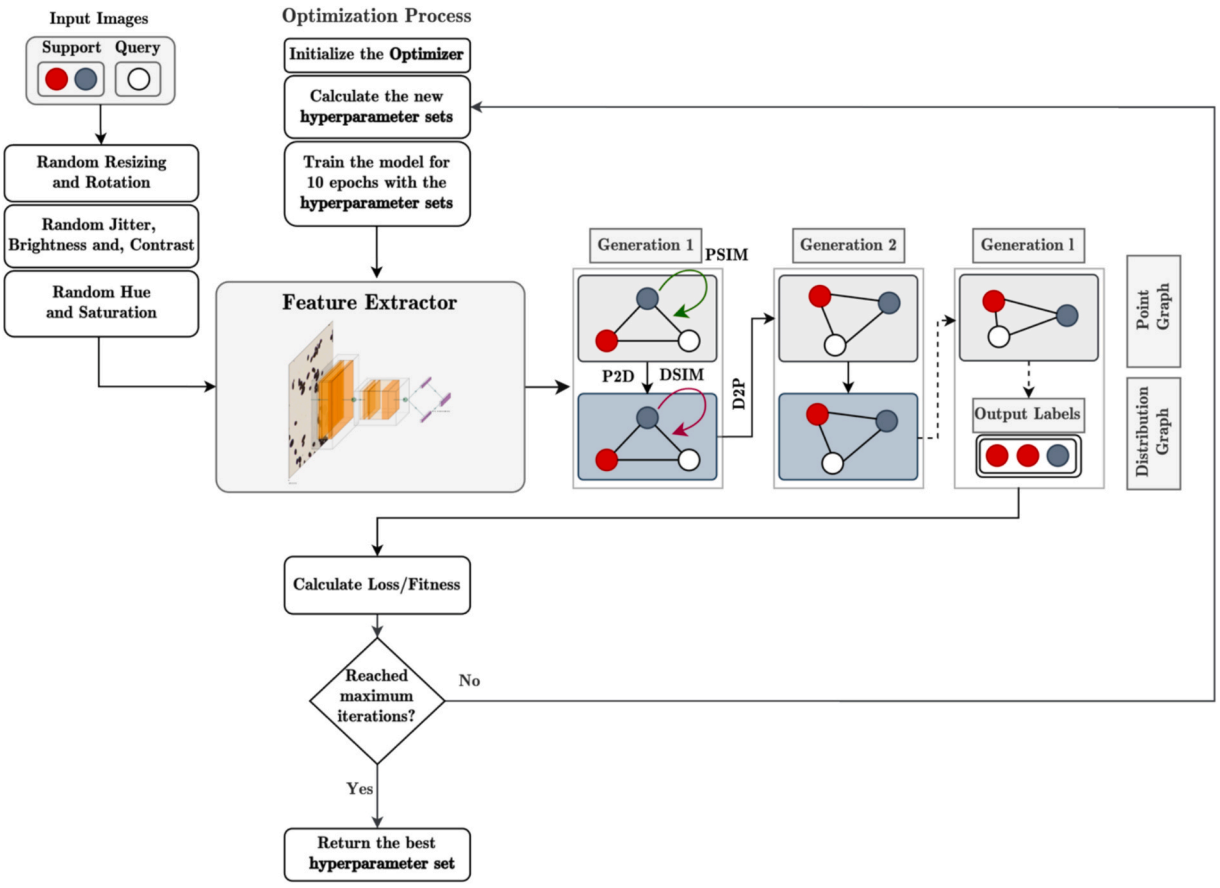



**Fig. 1.** The overall structure of the proposed model for a 2-way 1-shot episode.

halved to retain the complexity of each layer and also, the convolutional layers generally have 3 × 3 filters. After each convolutional layer, a batch normalization layer is inserted to increase regularization and decrease the sensitivity to network initialization.

Hyperparameter optimization in DL involves iteratively fine-tuning model configurations for enhanced performance. Beginning with random values within predefined bounds, a swarm-based optimizer constructs and trains the model using training data, updating weights to minimize loss. Validation data assesses model performance, with validation loss informing the optimizer's adjustments for generating new hyperparameters. The iterative process continues, refining model configurations to minimize validation loss and improve generalization. This adaptive approach explores hyperparameter space to discover optimal configurations for DL models.

In the inference phase, after preprocessing the input images, the model is fed by image patches in episodes. Each episode comprises a support set (S) and query set (Q), taken from the training and query data, respectively. The support set (S) contains N classes (ways) with K samples (shots) from each class (N-way K-shot), and the query set (Q) contains T samples. While in the training stage, labels are provided for both the support and query sets, in the inference phase, only the support set labels are fed to the trained classifier, which in turn accurately categorizes unlabeled query samples Q from test examples, with the aid of a few support samples (S) obtained from the train examples.

As the total amount of available data is low, the CNN is pre-trained on the ImageNet dataset [41] to tackle overfitting. Since fine-grained low-level features characterize the texture, no pooling was carried out between the convolutional layers to prevent loss of information [42]. Instead, a higher stride value is used to downsample the data [43].

The feature extractor block includes two parallel routes for better generalization; one route comprises a fully connected layer with batch normalization, while the other route consists of a convolutional layer with batch normalization, max pooling, leaky ReLU, and fully connected batch normalization layers. The processed feature tensors of each sample are the outputs of the feature processing block. The combination of the feature extractor and the feature processing block is depicted in Fig. 2.

The GNN classifier operates by propagating label information from labeled to unlabeled samples within several updates and can compute instance similarities using feature embeddings extracted by a CNN backbone. As shown in Figs. 3 and 4, the GNN classifier has l generations for each episode, where each generation contains two graphs: point graph $G_l^p = (V_l^p, E_l^p)$ and distribution graph $G_l^d = (V_l^d, E_l^d)$, inspired by DPGN. The instance similarities $E_l^p$ are calculated using the feature embeddings, and node features $V_l^d$ are computed by aggregating $E_l^p$ with the help of a point-to-distribution aggregation. Edge features $E_l^d$ are distribution similarities between $V_l^d$. To calculate the $V_l^p$, $E_l^d$ is returned to $G_l^p$ via a distribution-to-point aggregation. The operation is repeated for the next generation such that $E_l^p \rightarrow V_l^d \rightarrow E_l^d \rightarrow V_l^p \rightarrow E_{l+1}^p$, where l stands for the number of generations. Fig. 3 depicts the working process for a 2-way 1-shot problem. Here, the blue hatched rectangle is the query node, while the filled rectangle and crosshatched green rectangles are the support nodes. In each aggregation, the smaller and longer rectangles are the edges and vertices of the graphs, respectively. In this figure, dark and light grey background colors show the aggregation processes in the point graph and the distribution graph, respectively, where FC stands for Fully Connected Layers of CNNs.

The first generation of point node features $v_{0,i}^p$ is calculated using the output of the feature processing block. For the other generations, $v_{l,i}^p$ uses the aggregation based on the sum of the multiplication of the distribution edge features $e_{l,ij}^d$ and the last generations point node features $v_{l,i}^p$ and $v_{l,j}^p$ on the query set Q with the size T. This operation is depicted in

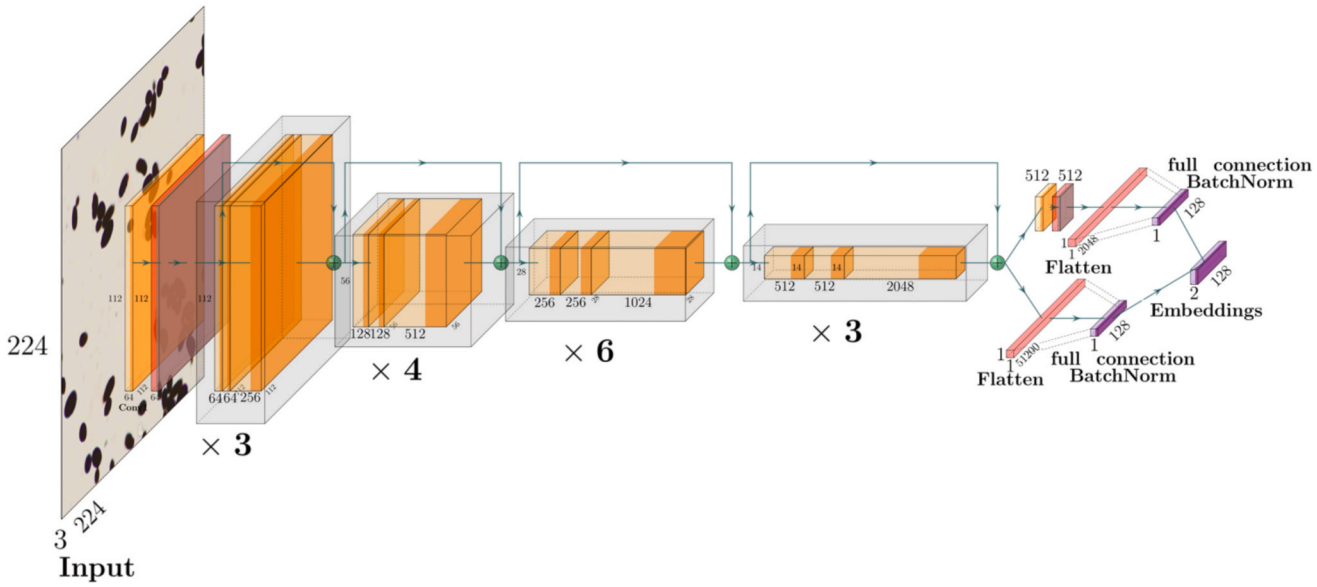



**Fig. 2.** Feature Extraction Block.

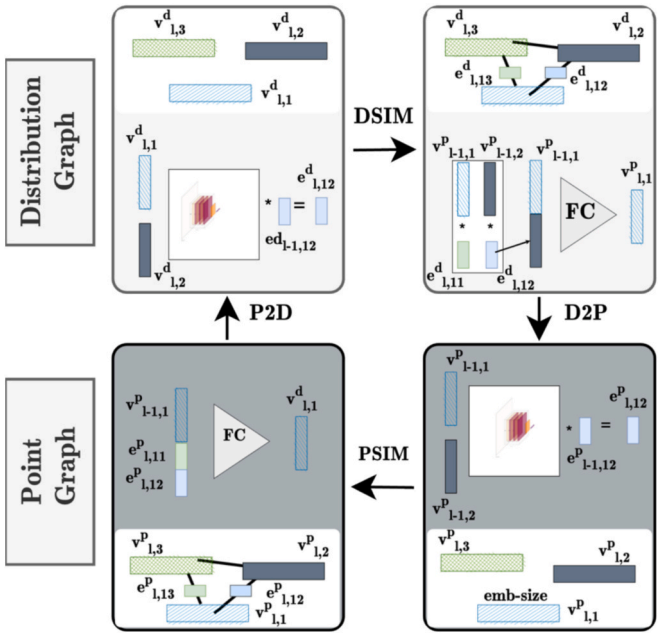



**Fig. 3.** Graph Network Aggregation Process Showing P2D, DSIM, D2P, and PSIM.

Fig. 3 as Distribution to Point Aggregation (D2P). In the following expression, $f_{D2P\_opt}(\cdot)$ stands for the optimized architecture of multiple Conv-BN-ReLu blocks, $\theta_{D2P\_opt}$ stands for its weight matrix, and the $||$ operator stands for the concatenation of input values.

$$v^p_{l,i} = \begin{cases} f_{D2P\_opt}\left(\theta_{D2P\_opt}, \left(\sum_{j=1}^{T} abs\left(e^d_{l,ij}.\, v^p_{l-1,j}\right) \| v^p_{l-1,i}\right)\right) l \neq 0 \\ f_{emb}\left(input_{image}\right) l = 0 \end{cases} \quad (1)$$

The first generation of the instance similarity $E^p_0$ is initialized with the multiplication of the feature processing block's outputs. The Point Similarity Aggregation (PSIM) calculates the $L_1$ distance between the inputs and consists of multiple convolutional, BN and ReLU (Conv-BN-ReLU) blocks, a convolutional and a sigmoid layer. As for the other generation, the output of the sigmoid layer is multiplied by the $E^p_{l-1}$ from the last generation. Here, $f_{PSIM\_opt}(\cdot)$ stands for optimized the CNN block described previously with $\theta_{PSIM\_opt}$ as its weights.

$$e^p_{l,ij} = \begin{cases} f_{PSIM\_opt}\left(\theta_{PSIM\_opt}, abs\left(v^p_{l-1,i} - v^{p\ T}_{l-1,j}\right)\right).e^p_{l-1,ij} l \neq 0 \\ f_{PSIM\_opt}\left(\theta_{PSIM\_opt}, abs\left(v^p_{l,i} - v^{p\ T}_{l,j}\right)\right) l = 0 \end{cases} \quad (2)$$

Distribution node features, mark the distribution level relation of input instances, accordingly each $v^d_{l,i}$ is a tensor of size NK where the realtion of $x_i$ and $x_j$ is represented in the jth element. To initialize the distribution node features of the first generation, equation (3) is used where $\delta(\cdot)$ is the Kronecker delta function where the output is one if both inputs are equal and zero otherwise, and $y_i$ is the $x_i$'s label if it is known.

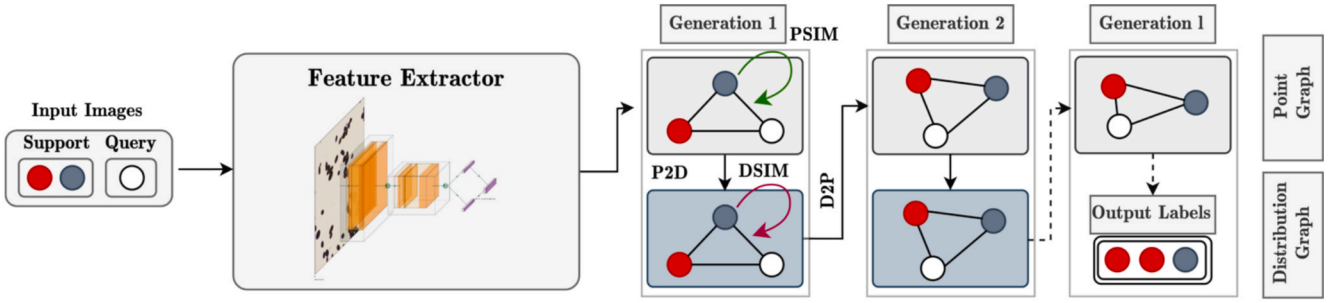



**Fig. 4.** The Inference process for a 2-way 1-shot episode.

$$v_{0,i}^{d} = \begin{cases} \left[\delta\left(y_i, y_j\right) for\, j\, in\, NK\right] y_i is known \\ \left[\frac{1}{NK} for\, j\, in\, NK\right] y_i is unknown \end{cases} \quad (3)$$

To calculate the distribution node features of the other generations, in the following expression, $f_{P2D\_opt}(\cdot)$ stands for the optimized architecture of multilayer perceptron, $\theta_{P2D\_opt}$ stands for its weight matrix and, || operator stands for the concatenation of input values. This step is shown in Fig. 3 as Point to Distribution aggregation (P2D).

$$v_{l,i}^{d} = f_{P2D\_opt}\left(\theta_{P2D\_opt}, \left(\left[e_{l,ij}^{p} for\, j\, in\, NK\right] \| \, v_{l-1,i}^{d}\right)\right) \quad (4)$$

Similar to the point similarity, the distribution similarity block forwards the $L_1$ distance of the distribution node features through multiple optimized Conv-BN-ReLU blocks and Conv-sigmoid layers denoted by $f_{DSIM\_opt}(\cdot)$ in the following expression with $\theta_{DSIM\_opt}$ as its weights. The first generation is initialized using the distribution node features of the same generation. This aggregation is marked as Distribution Similarity Aggregation (DSIM) in Fig. 3.

$$e_{l,ij}^{d} = \begin{cases} f_{DSIM\_opt}\left(\theta_{DSIM\_opt}, abs\left(v_{l-1,i}^{d} - v_{l-1,j}^{d\,T}\right)\right). e_{l-1,ij}^{d}\, l \neq 0 \\ f_{DSIM\_opt}\left(\theta_{DSIM\_opt}, abs\left(v_{l,i}^{d} - v_{l,j}^{d\,T}\right)\right) l = 0 \end{cases} \quad (5)$$

To calculate the output labels, the node edge features of the last generation $E_l^p$ are fed into a softmax function, which in turn calculates the probability of an input having the label y.

As there are two graphs involved in the classification of input images, two types of losses are implemented: Point graph loss and distribution graph loss. Both loss functions are computed by calculating the cross entropy loss of softmax of the last generation of the corresponding edge.

$$Loss = CE\left(Softmax\left(\sum_{j=1}^{NK} e_{l,ij}.onehot\left(y_j\right)\right), y_j\right) \quad (6)$$

The rationale for using DPGN to classify images of uncultured fungi lies in the advantages of its dual-graph structure within few-shot learning contexts. The DPGN combines a point graph and a distribution graph to represent two levels of relationships among samples. The point graph captures instance-level relations by modelling pairwise similarities between specific images, whereas the distribution graph captures broader distribution-level relations by aggregating each sample's similarity with all other samples. This dual representation allows DPGN to capture both fine-grained details of each instance and contextual similarities across the dataset. Such a combination is useful when dealing with small-data and high-variation tasks like fungal classification, where each graph's role complements the other to refine both instance and distribution features effectively.

The DPGN approach assumes that combining instance and distribution-level features provides a richer feature space, which is especially effective when the training set is sparse. By cyclically updating the point graph and distribution graph, DPGN propagates label information in a way that strengthens intra-class similarity while amplifying inter-class dissimilarity to improve its ability to differentiate visually subtle classes within fungi. This approach enhances classification performance even with minimal labelled samples, as DPGN has demonstrated superior label propagation capabilities and accuracy on few-shot learning benchmarks. In this setting, DPGN's design not only addresses the challenge of limited training data but also optimally exploits contextual relations, making it well-suited for accurately classifying complex, under-sampled fungal images.

The network architecture hyperparameters encompass a range of critical decisions, including the number of layers, neurons per layer, activation functions, and dropout rates, among others. Each choice impacts the network's capacity to learn and generalize from data. Inadequate hyperparameter settings can lead to issues such as underfitting or overfitting, severely hindering model performance. Consequently, identifying the optimal network architecture requires careful experimentation, often in combination with architectural innovations and transfer learning techniques.

Similarly, the optimization hyperparameters, such as the learning rate, momentum, batch size, and weight decay, influence how the model updates its parameters during training. The learning rate, in particular, plays a crucial role in determining the step size of parameter updates and can significantly affect the convergence speed and final performance. A poorly chosen learning rate may cause training instability, oscillations, or convergence to suboptimal solutions. Therefore, selecting suitable optimization hyperparameters is equally vital in attaining successful training outcomes.

While common optimizers provide means to calculate hyperparameters, the challenge of selecting the most suitable optimizer persists. In this research, we conducted a comprehensive benchmarking study to evaluate the efficacy of eight different swarm-based optimizers for hyperparameter optimization. Swarm-based optimizers are a class of metaheuristic algorithms inspired by the collective behavior of social organisms, such as birds flocking and fish schooling. These algorithms mimic the interactions between individuals in a swarm to find optimal solutions in complex search spaces. By leveraging the principles of swarm intelligence, these optimizers can effectively navigate high-dimensional and non-convex hyperparameter spaces, making them attractive candidates for hyperparameter optimization tasks in DL. In this work, several architecture-related hyperparameters, such as the number of layers in aggregations and the number of generations, are optimized.

Choosing the learning rate can be challenging because a value too small may result in a lengthy training process, whereas a value too large can lead to learning an unideal set of weights too quickly or an unstable training process. The learning rate may be the most critical hyperparameter when configuring neural networks. The step decay function calculates the learning rate in each iteration as seen in equation (7), where $\lfloor \cdot \rfloor$ stands for the floor operation.

$$lr_{new} = lr_{last} * \left(decayFactor^{\left\lfloor \frac{iteration}{stepSize} \right\rfloor}\right) \quad (7)$$

This work uses swam based optimizers to optimize the initial learning rate, decay factor, and step size of the step decay function. Additionally, the optimizers, calculate the dropout ratio and weight decay to avoid overfitting. The entire optimization process is shown in Fig. 1.

Optimizing the hyperparameters of a DL model involves a systematic process to fine-tune the model's configurations for enhanced performance. Initially, the swarm-based optimizer is initialized with random values within predefined boundaries for each hyperparameter. Subsequently, a DL model is constructed using the newly generated hyperparameter set, and the model is trained using the processed images from the training dataset for a specific number of epochs. During this training phase, the model iteratively updates its weights to minimize the loss function, capturing the patterns and representations within the training data.

After completing the training process, the model's performance is assessed on a separate validation dataset. This computed validation loss is then relayed back to the swarm-based optimizer as the fitness value. Leveraging the fitness value, the optimizer performs an optimization routine to generate a new set of hyperparameters. This optimization step is driven by the goal of finding hyperparameter values that minimize the validation loss, leading to improved generalization and performance on unseen data. The process iterates as the optimizer continuously updates the hyperparameter set based on the fitness value, resulting in a refined model configuration. Through this iterative and adaptive approach, the swarm-based optimizer effectively explores the vast hyperparameter

space, facilitating the discovery of optimal configurations for DL models.

## 3. Experiments

Here, we present the experimental setup and results of the proposed model for classifying microscopic fungi images. The experiments were designed to validate the effectiveness of the proposed method compared to the performance of existing techniques in the field.

### 3.1. Datasets

#### 3.1.1. DeFungi

To evaluate the proposed model, we have used two publicly available fungi datasets: DeFungi [20] and DIFas [17]. DeFungi dataset contains the direct image patches of 5 common fungi types: Tortuous septate hyaline hyphae (TSH), Beadedarthroconidial septate hyaline hyphae (BASH), Groups or mosaics of arthroconidia (GMA), Septate hyaline hyphae with chlamydioconidia (SHC), and Broad brown hyphae (BBH). In total, there are 660 directly examined microscopic images with no preprocessing, varying in size from 640 × 480 pixels up to 5152 × 3864 pixels.

As illustrated in Fig. 5, the images encompass a wide range of lighting conditions, magnification levels, and color correction schemes, thereby amplifying both intraclass dissimilarities and interclass similarities within the dataset. Notably, these images have not undergone any preprocessing procedures such as gram staining or culture, posing a significant challenge for identification tasks. Among these classes, namely TSH, BASH, and GMA, there exists a striking resemblance in both color and shape, further complicating the classification process.

#### 3.1.2. DIFaS

To further examine the model's performance, Digital Images of Fungus Species database (DIFaS) [17] dataset was also used. As shown in Fig. 6, DIFaS consists of images of cultured and gram stained patches from five yeast-like fungal strains, two yeast strains, and two strains of Basidiomycetes. Image classes are Candida Albicans ATCC 10231 (CA), Candida Glabrata ATCC 15545 (CG), Candida Tropicalis ATCC 1369 (CT), Candida Parapsilosis ATCC 34136 (CP), and Candida Lustianiae ATCC 42720 (CL); Saccharomyces Cerevisiae ATCC 4098 (SC) and Saccharomyces Boulardii ATCC 74012 (SB); Maalasezia Furfur ATCC 14521 (MF) and Cryptococcus Neoformans ATCC 204092 (CN) respectively. Altogether, there are 176 high resolution (3600 × 5760 pixels) 16-bit color images of 9 class strains [17].

Culturing and employing Gram staining techniques substantially amplify the distinctions among fungal species, thereby facilitating their visual identification. Nonetheless, these methodologies entail considerable time and financial resources, rendering them impractical in certain scenarios. Furthermore, a notable challenge within the dataset pertains to the limited quantity of available data. With a total of 180 image patches distributed across 20 classes, the dataset poses a significant challenge for analysis and classification.

### 3.2. Preprocessing

In microscopic fungi images, the majority of the background tends to have no objects, making selecting random patches from input images challenging. The background of these images often exhibits high-intensity levels, further complicating the preprocessing task. To address these challenges, a specialized automatic preprocessing pipeline is employed. Initially, random patches are selected from the images. Then, all patches are converted to black and white and the contrast of the patches is enhanced by a factor of 150 %. Subsequently, only patches with average intensity value under a certain threshold are retained for further processing. By eliminating patches with high-intensity levels (mainly regions without fungi), the preprocessing pipeline ensures that meaningful image regions are selected, effectively filtering out the predominantly blank regions. This pipeline is shown in Fig. 7.

To increase the data size and avoid overfitting the model [39], we implement different image augmentation algorithms such as random crop of 142 × 142 pixels, random rotation of at most 30 degrees, random

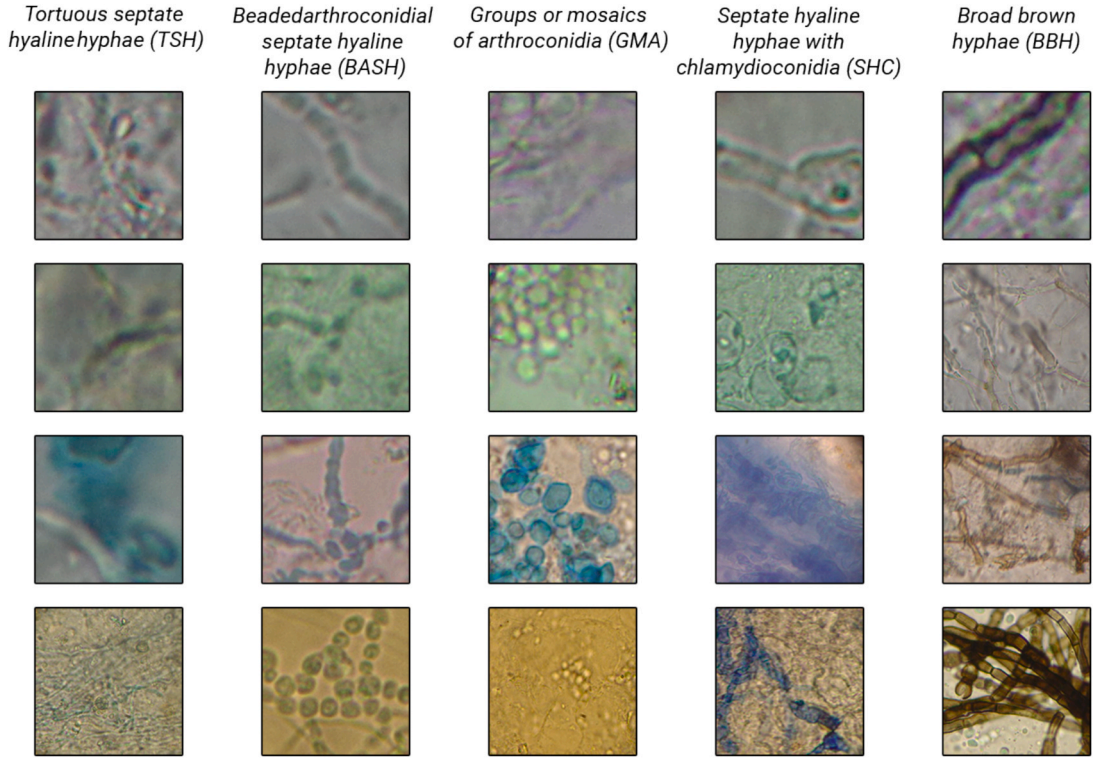



**Fig. 5.** Raw image samples from DeFungi database [34].

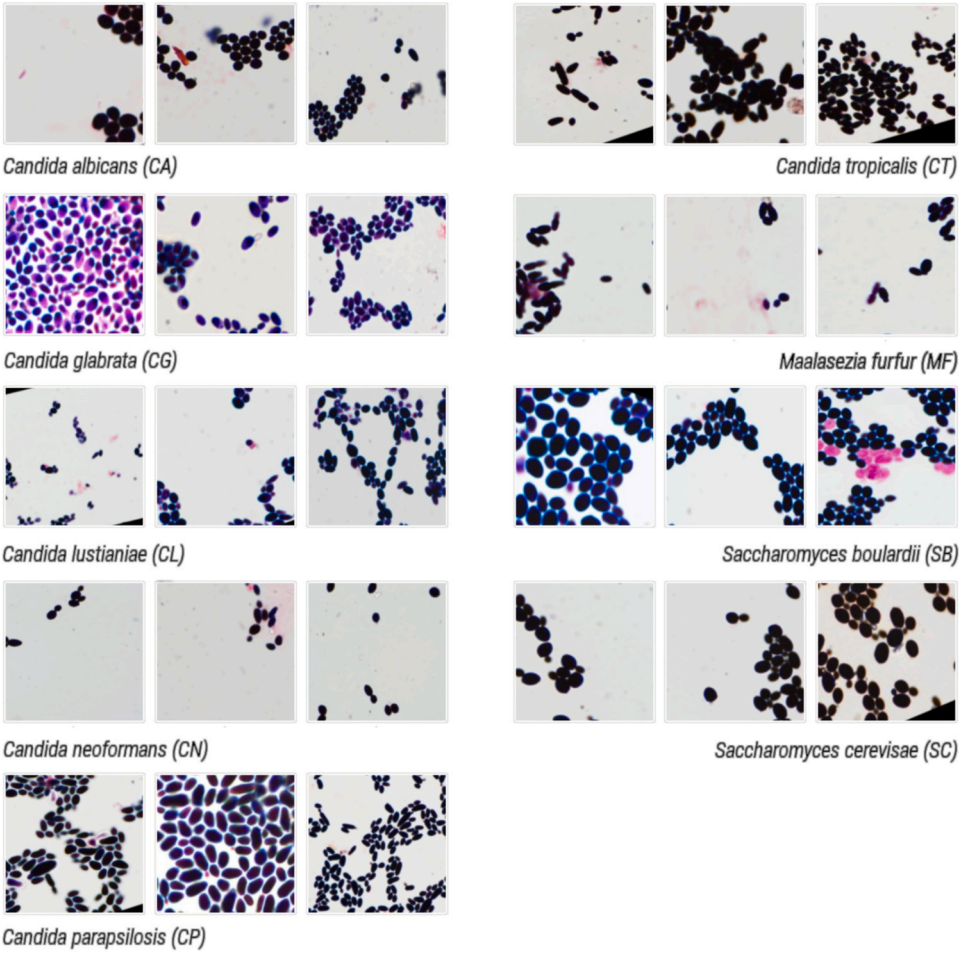



**Fig. 6.** Image patches from the DIFaS database [17].

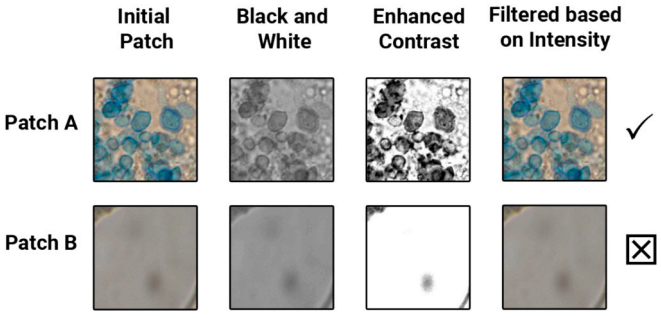



**Fig. 7.** Automatic patch selection pipeline. After the preprocessing, patch B has been eliminated as the average intensity level is higher than the selection threshold.

horizontal flip, and color jitter with random variations of 10 % in brightness, contrast, saturation and hue of images. As the original resolution of images is in a diverse range, we sampled all input images to 1280 × 800 pixels (1 Megapixel) and 8-bit color depth. After the sampling, all images are cropped to 224 × 224-pixel patches. Before feeding the cropped images to the data augmentation algorithm, we normalize all channel levels (red, green, and blue). Our final image patches, fed into the model, are 224 × 224 pixels with 3 channels and an 8-bit depth, irrespective of the dataset, zoom level, or applied image augmentation techniques. Images were split into train and test categories, each with 80 % and 20 % of the data, respectively.

The experiments were conducted using DIFaS and DeFungi datasets, which consists of microscopic images of various fungal species in a cultured and uncultured manner. To ensure a fair comparison, all models were trained and tested on the same data splits. Each dataset was divided into 80 % for training and 20 % for testing, with 10-fold cross-validation applied. All experimental results were cross-validated using k-fold cross-validation with ten folds. For benchmarking, all operations were performed on Google Collaboratory with Tesla K80 GPU and Intel (R) Xeon(R) CPU giving us 12 GB GPU and 12 GB RAM. The whole process, from image preprocessing to performance measurement, was written in Python programming language.[1] We used PyTorch to implement the model in python because of its ability to use GPU parallel processing capabilities. To optimize the model, the mealpy python library was used.

### 3.3. Hyperparameter optimization

In this study, the optimization of hyperparameters, including the model's architecture and training parameters, was performed. As the optimization results stabilized and showed slight variation in the later iterations, the optimization steps were set at 5. Each iteration consisted of a population size of 30. Consequently, the model went through a total of 150 training and testing sessions across all optimizers. Table 1 presents the optimized hyperparameters along with their respective minimum and maximum ranges.

[1] Codes are hosted on GitHub at https://github.com/pooyasa/gnn-fungi-classification

The proposed model was subjected to a comprehensive training process involving various hyperparameter optimizers. Each optimizer returned a distinct set of hyperparameters, and the model was trained for ten epochs with a batch size of 16. This iterative training process was performed individually for each optimizer, and upon completion, the model with the best overall performance across all optimizers was selected for further refinement. The chosen model went through an additional 10 epochs of training to enhance the predictive capabilities.

To evaluate the model's performance, classification accuracy was measured using two different methods: patch-based and aggregation scan-based. The classification accuracy was determined by comparing the model's predicted classes with the ground truth labels in the testing dataset. To ensure robustness, the evaluation was carried out using a five-fold cross-validation approach, and the results were averaged across the folds.

For patch-based classification accuracy, the model's accuracy was calculated based on the correct classification of individual patches within each image. On the other hand, the aggregation scan-based classification accuracy utilized a voting mechanism, where the label of a test sample was determined by the inferred label of the majority of classified patches within the image. An example of this voting mechanism is illustrated in Fig. 8. In Fig. 8, out of the 9 image patches, two are incorrectly classified which in turn leads to a patch-based classification accuracy of 77.8 % and an aggregation scan-based accuracy of 100 %.

## 4. Results

The box plot displayed in Fig. 9 showcases the validation set accuracy of the model at each iteration of the hyperparameter optimization process. The figure provides compelling evidence of the convergence of validation accuracy, indicating the effectiveness of the applied hyperparameter optimization techniques. Initially, all optimizers exhibit similar accuracy distributions, gradually improving as the optimization process unfolds. However, among the evaluated optimizers, Ant Colony, Particle Swarm, and Gray Wolf optimizers demonstrate the best overall performance. Notably, the Gray Wolf optimizer emerges as particularly proficient in identifying the optimal hyperparameters, ultimately yielding the highest overall accuracy among all evaluated optimizers.

### 4.1. DeFungi dataset

Our proposed graph neural network approach achieved an accuracy of 88.7 % on patch-based classification and 93.1 % on aggregation scan-based classification of microscopic fungi images. These results demonstrate the effectiveness of our method in identifying and classifying fungal species from microscopic images.

To show the advantages of the proposed model, we compared the performance to several existing methods in the field. Table 2 shows the classification accuracies of our method and other state-of-the-art techniques on the DIFaS dataset. As can be seen, the proposed model outperforms other methods in both patch-based and aggregation scan-based classification tasks. A comparison of our accuracy with RaViTT and the DeFungi paper yields p-values of 1.8e-5 and 0.0098, respectively, indicating statistically significant results. It is worth noting that due to differences in benchmarking and experimental setups between the methodologies introduced by Mohamed et al. [44], Singh et al. [45], and Asadi Amiri et al. [46] and our approach, we ensured a fair comparison by replicating their methods, including all preprocessing steps and hyperparameters. We then implemented these methods on our dataset using our setup and compared the results with our approach. Fig. 10 shows the learning curve and the confusion matrix for the test set.

### 4.2. DIFaS dataset

To assess the effectiveness of our model, we conducted additional tests using the DIFaS database. Evaluation of this independent dataset yielded superior results, indicating the model's capability for generalization. The enhanced performance of the model on the DeFungi database, as presented in Table 3, underscores its adaptability and reliability across diverse datasets. The p-values comparing the patch-based accuracy of our work with AlexNet FV SVM and AlexNet SA-AbMILP are 7.1e-5 and 0.0025, respectively, demonstrating statistically significant results for this task. The learning curve and confusion matrix pertaining to this database are further delineated in Fig. 11.

### 4.3. Ablation study

In machine learning, an ablation study involves systematically disabling or removing specific parts of a model, such as layers, features, or parameters, to evaluate their contributions to the model's performance. The main objective of this type of study is to gain a deeper understanding of the model's functioning, determine which components are crucial, and measure each component's impact on overall performance.

In this work, we performed an ablation analysis to evaluate the significance of various components within our graph model. Instead of fully removing each component, we chose to partially reduce its influence to avoid disrupting the information flow between layers, which is critical for our analysis. Specifically, we removed 25 % of the parameters within the trainable layers of each component, following the approach recommended in [48]. This involved manually setting certain weights and bias values to zero. As the number of neurons and filters varies across different components, the quantity of neurons and filters eliminated also varied in each experiment.

Fig. 12 provides an overview of the experiment setup, illustrating seven distinct experiments conducted across seven model components. These components are as follows: (a) the feature extraction block, (b) feature processing block I, (c) feature processing block II, (d) point similarity aggregation, (e) point-to-distribution graph aggregation, (f) distribution similarity aggregation, and lastly, (g) the distribution-to-point graph aggregation.

Following the reduction in the effect of each component within the main model, the damaged model is evaluated on the DeFungi dataset. The results, displayed in Table 4, are compared against the performance of the original model. This comparison allows us to clearly understand the contribution of each component to the overall accuracy in classifying Fungi images.

As shown in Table 4, components (f) distribution similarity aggregation and (g) distribution-to-point graph aggregation have the greatest impact on final accuracy, as their damage leads to the lowest accuracy

**Table 1**
Optimized parameters and their ranges.

| Optimized Parameter | Learning Rate | Weight Decay | Dropout | Learning Rate Adjustment | Step Size | Number of Generations | Conv Layers in Each Aggregation | Conv Channel Count | MLP Layers in aggregations | Feature Embeddings Size |
|---|---|---|---|---|---|---|---|---|---|---|
| Minimum Value | 1e-4 | 1e-6 | 0.01 | 0.01 | 10,000 | 3 | 1 | 64 | 1 | 64 |
| Maximum Value | 1e-2 | 1e-4 | 0.5 | 0.3 | 20,000 | 10 | 5 | 256 | 5 | 512 |

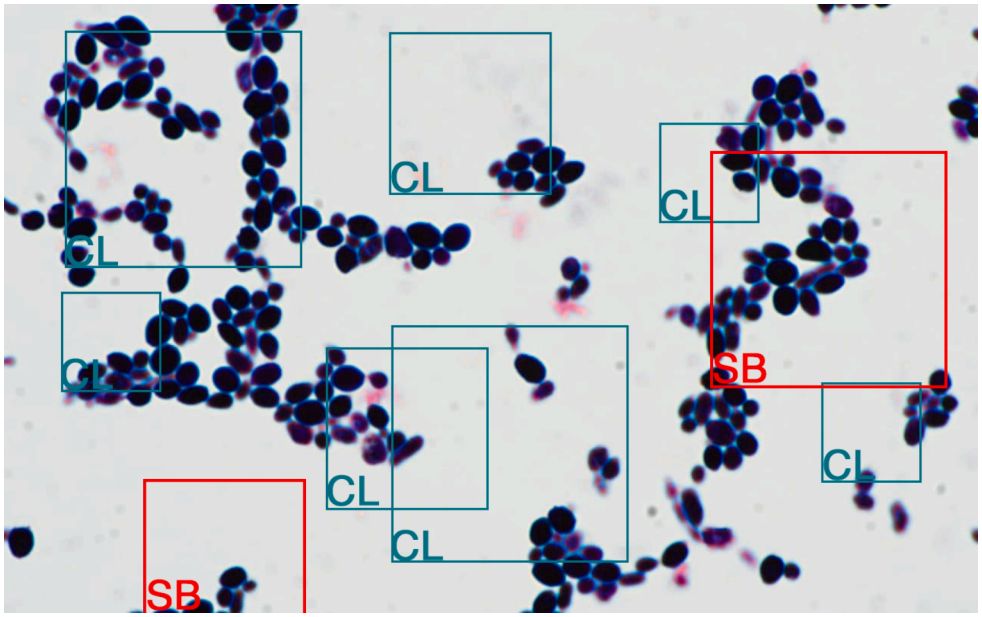



**Fig. 8.** Patch-based and Aggregation scan-based classification.

scores 69.85 % and 67.36 %, respectively. In contrast, diminishing the impact of components (d) point similarity aggregation and (e) point-to-distribution graph aggregation results in fewer changes to the model's accuracy, with the scores 72.09 % and 72.16 %, suggesting these components are less critical to the model's overall performance.

## 5. Discussion

In this study, we introduced a DL approach to directly identify pathogenic fungi without culturing, potentially eliminating reliance on mycology experts. Our methodology integrated few-shot learning techniques with optimized graph neural networks to address overfitting and enhance generalization capabilities. Through extensive experimentation and benchmarking of model hyperparameters using various meta-heuristic optimizers, we demonstrated the effectiveness of our approach compared to previous methodologies. Our graph neural network classifier leveraged the learned feature embeddings from the CNN backbone to propagate label information from labeled to unlabeled samples. Using two types of graphs – point graph and distribution graph – enabled the model to capture local and global relationships within the data, contributing to its performance in classification tasks.

Hyperparameter optimization played a crucial role in fine-tuning our model for optimal performance. By iteratively adjusting hyperparameters such as learning rate, dropout rate, and architectural hyperparameters, we enhanced the model's convergence speed and final accuracy. Careful selection of these optimization hyperparameters ensures stable training dynamics and prevents convergence to suboptimal solutions.

We conducted multiple hyperparameter optimization procedures using several swarm-based optimizers to explore various configurations. We observed that different optimizers converged on distinct sets of hyperparameters, which aligns with the exploration strategies employed by each optimizer. For instance, some optimizers, like GWO and PSO, tend to balance exploration and exploitation effectively and often lead to a more diverse set of solutions, while in our experiments, methods like Bald Eagle Search and Harris Hawks Optimization converged more quickly to specific values which shows signs of getting stuck in local minimum. This variation in the final hyperparameter sets shows each optimizer's way of navigating the hyperparameter space, which may influence performance outcomes.

The distinct sets of hyperparameters obtained through various optimizers provide insight into which parameters may be more sensitive for this model and datasets. For instance, we noticed certain hyperparameters, such as the number of convolutional layers and MLP layers in aggregations, showed more variation across optimizers which suggests these could be more influential for optimizing performance. This variability could offer insight into the inner workings of the final model, showing that architectural hyperparameters relating to the GNN classifier have the most impact on the final performance.

While our main goal was to achieve high classification accuracy, we observed certain patterns in selected hyperparameters that align with the characteristics of the datasets. Specifically, for the DeFungi dataset, which exhibits high interclass similarity, optimizers often favored moderate to high dropout rates and higher learning rate adjustments, which could be interpreted as settings to overcome overfitting, though they increased training time compared to the DIFaS dataset. These trends indicate that hyperparameters might be adjusted further based on specific dataset characteristics, which we intend to explore in future work.

Moving on to the experimental setup, we conducted comprehensive experiments to evaluate the performance of our proposed model on two publicly available fungi datasets: DeFungi and DIFaS. The DeFungi dataset comprises direct image patches of five common fungi types, totaling 660 microscopic images with varying sizes and no preprocessing. These images exhibit a wide range of lighting conditions, magnification levels, and color correction schemes, posing challenges for identification tasks. Notably, the lack of preprocessing procedures such as gram staining or culturing further complicates the classification process, especially given the striking resemblance among certain fungal classes.

From the confusion matrix analysis, it becomes evident that distinguishing between the TSH, BASH, and GMA classes poses a significant challenge for the model. This difficulty can primarily be attributed to the visual similarities observed among these classes and the inherent variations present within each class, such as differences in patch color, magnification levels, and image quality, as illustrated in Fig. 5. Furthermore, beyond the visual similarities and intra-class variations, the shape and size of each fungal type further contribute to the complexity of classification. For instance, some patches exhibit bead-like shapes within the BASH or GMA classes, while others display a string-like morphology. In contrast, the SHC and BBH classes demonstrate more consistent shapes across different samples.

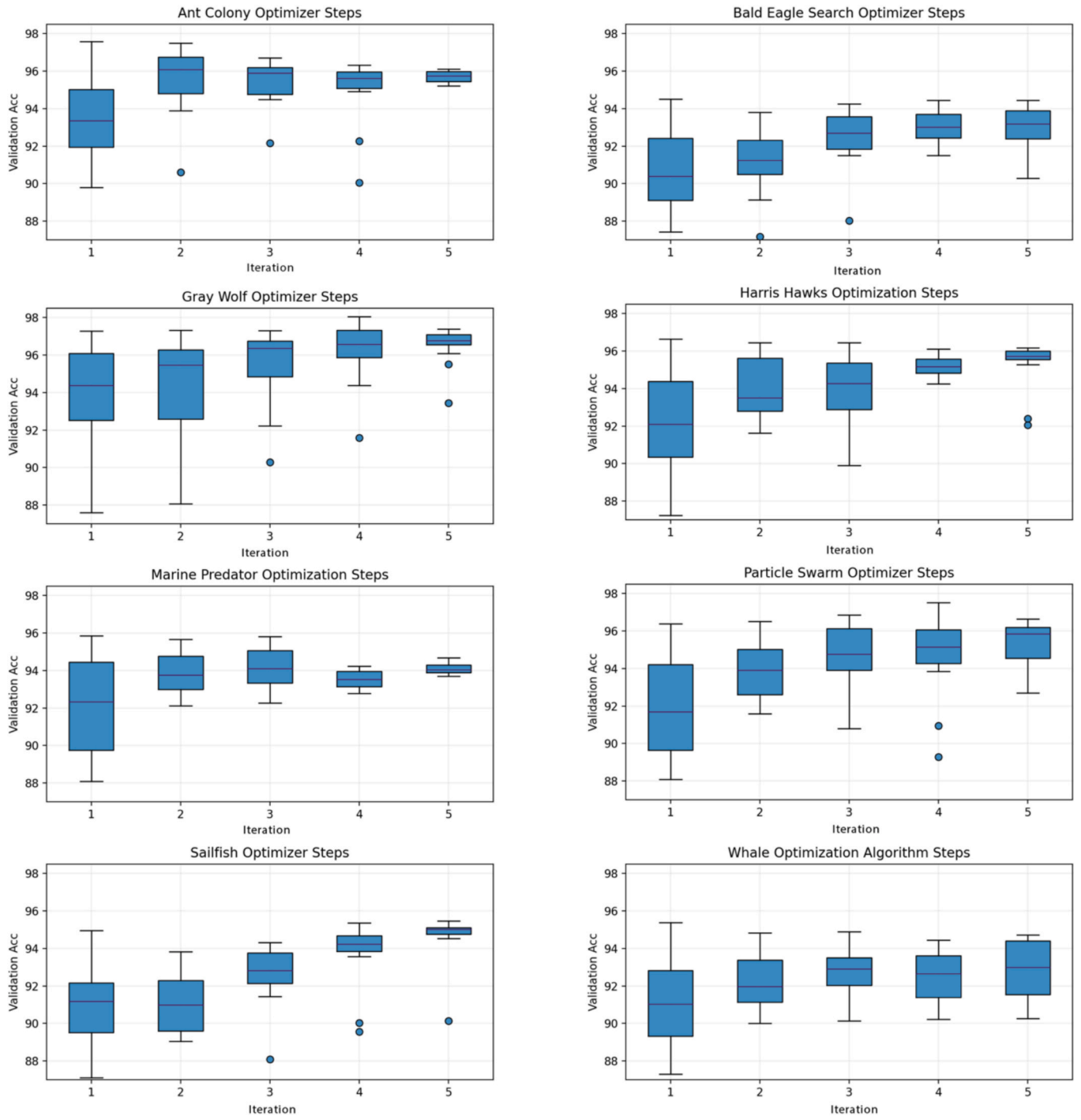



**Fig. 9.** Validation accuracy of each iteration of the optimization process for different optimizers.

The model achieved a patch-based accuracy of 91.4 % and an aggregation accuracy of 100 % on the DIFaS dataset, indicating strong performance. Notably, since the species in this dataset are cultured and gram-stained, the distinctions among classes are amplified, facilitating easier differentiation for the model. However, despite the overall high accuracy, the confusion matrix and visual examination of the strains reveal challenges distinguishing between MF, CN, and CA. These strains exhibit similarities in visual aspects such as density, roundness, and color, leading to some ambiguity for the model in classification.

We have compared our model with multiple state-of-the-art approaches to the identification of pathogenic fungi; RaViTT [19], original DeFungi paper [20], AlexNet FV SVM [17], and AlexNet SA-AbMILP [47]. The rationale for comparing our work with these baseline methods lies in their similarity of methodology, relevance to the same topic of microscopic fungal detection, and the use of the DeFungi and DIFaS datasets. These methods are designed for similar problems, characterized by small dataset sizes and high pixel importance in microscopic imagery.

RaViTT [19] reported an accuracy of 86.8 % on the aggregation-based classification of the DeFungi dataset using visual transformers. In the original DeFungi paper [20], the authors reported a maximum accuracy of 85.0 % using pre-trained VGG16 CNN in conjunction with a neural network classifier on the dataset. As for the DIFaS dataset, the original authors of the dataset attained a classification accuracy of 82.4

**Table 2**
Comparison of classification accuracies of our method and other state-of-the-art techniques on the DeFungi dataset from 10-fold cross-validation.

| Method | Patch-based Accuracy | Aggregation Scan-based Accuracy |
|---|---|---|
| Our Graph Neural Network (GNN) | 88.7 ± 3.4 | 93.1 ± 2.2 |
| RaViTT [19] | – | 86.8 |
| DeFungi Paper [20] | 85.0 ± 2.0 | – |
| DenseNet121 [44] | 85.2 | – |
| MobileNetV3 [45] | 79.9 | – |
| VGG19 [46] | 81.6 | – |

% using pre-trained AlexNet and SVM classifiers. Rymarczyk et al. [47] reported a patch-based accuracy of 86.0 % on the same dataset using self-attention attention-based MIL Pooling in conjunction with pre-trained AlexNet. The methodology proposed in the paper introduces Self-Attention Attention-based MIL Pooling (SA-AbMILP), a model designed for weakly supervised image classification under multiple instance learning (MIL). This approach uses self-attention mechanisms which allows it to capture dependencies between instances within a bag. The SA-AbMILP model first processes individual instances through a CNN to extract feature representations. It then applies a self-attention module, which uses a variety of kernel functions to model inter-instance dependencies. These feature representations are subsequently aggregated by the AbMILP operator to produce a fixed-size vector for each bag, suitable for classification.

Our experimental results demonstrated the robustness and effectiveness of our proposed model across both datasets. By achieving higher accuracy in classifying diverse fungal species under challenging conditions, our model outperformed existing techniques in the field. The ability to accurately identify pathogenic fungi directly from microscopic images without the need for culturing or staining represents a significant advancement in medical diagnostics, potentially saving time and resources in clinical settings.

These results highlight the effectiveness of our proposed methodology in accurately identifying pathogenic fungi from microscopic images, showcasing its potential for improving medical diagnostics. Despite the challenges of certain fungal classes, our model demonstrates remarkable performance across diverse datasets, outperforming existing techniques in the field.

Despite the promising results, there are several avenues for future research. Further exploration of transfer learning techniques could enhance the model's adaptability to specific fungal species, while validation on more extensive and diverse datasets would increase its real-world applicability. Additionally, ongoing efforts to refine optimization strategies and explore alternative model architectures could lead to further improvements in performance and scalability.

Our study presents a new DL approach for direct pathogenic fungi identification, leveraging optimized graph neural networks and swarm-based optimization techniques. Our methodology offers a promising solution for improving medical diagnostics and personalized healthcare by addressing critical challenges in fungal classification.

## 6. Conclusion

The experimental results demonstrate that our proposed graph neural network approach is capable of accurately identifying and classifying microscopic fungi images. Our model achieved 100 % accuracy on the DIFaS dataset and 93.1 % accuracy on the DeFungi dataset, which, to the best of our knowledge, is the highest accuracy achieved on these benchmark datasets. The high accuracy obtained on both patch-based and aggregation scan-based classification tasks indicates that our method can effectively eliminate the need for additional examinations in the diagnosis of fungal infections. In addition, the comparison with existing techniques highlights the superiority of our approach in addressing this challenging problem.

The success of our method can be attributed to the integration of the graph neural network with a deep feature extractor pretrained on the ImageNet dataset. This combination allows for the effective extraction of discriminative features from the microscopic images, which in turn enables accurate classification of the fungal species. Furthermore, the use of data augmentation techniques and the few-shot learning setting help to mitigate the risk of overfitting and improve the generalization performance of the model.

**Table 3**
Comparison of classification accuracies of the proposed model and state-of-the-art techniques on the DIFaS dataset.

| Method | Patch-based Accuracy | Aggregation Scan-based Accuracy |
|---|---|---|
| Our Graph Neural Network (GNN) | 91.4 ± 3.8 | 100 |
| AlexNet FV SVM [17] | 82.4 ± 0.2 | 93.9 ± 3.9 |
| AlexNet SA-AbMILP [47] | 86.0 ± 1.0 | – |

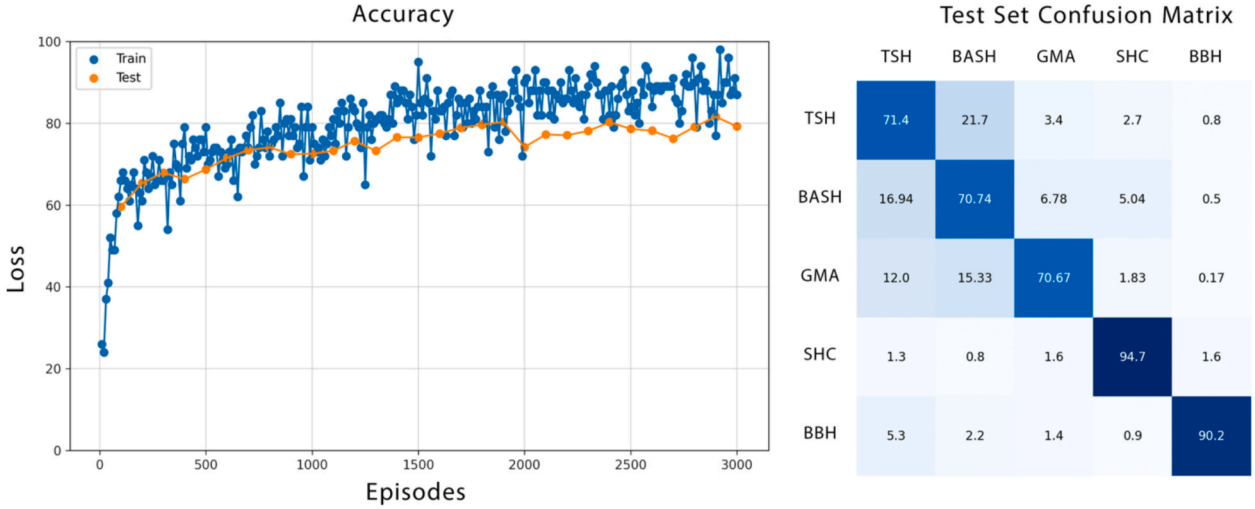



**Fig. 10.** Learning curve and confusion matrix for Defungi database [20].

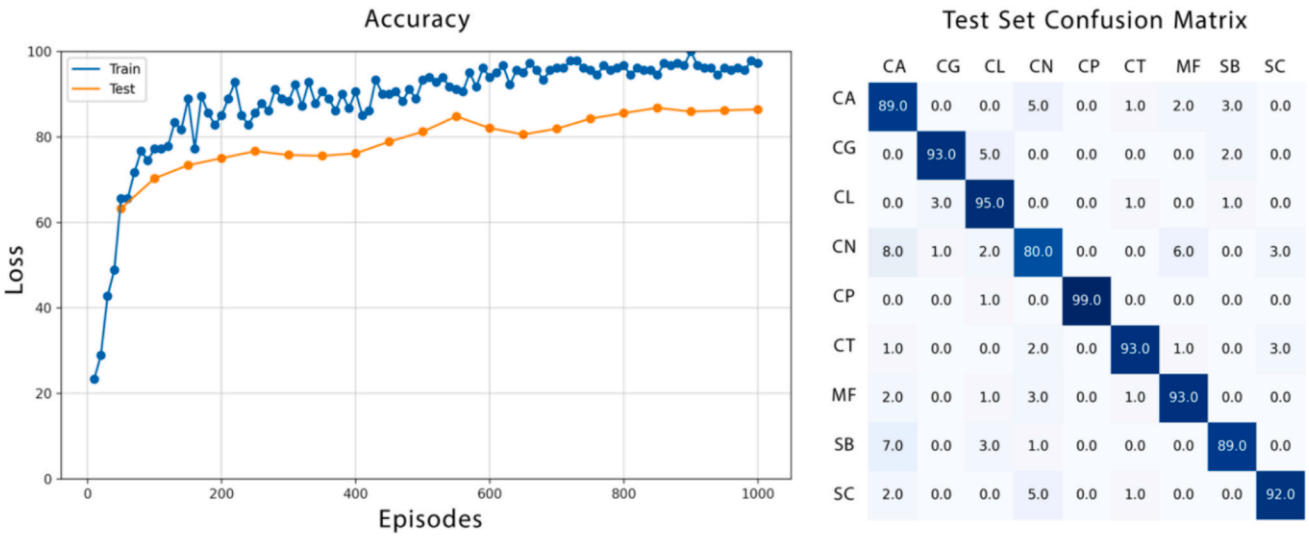



**Fig. 11.** Learning curve and confusion matrix for DIFaS database [17].

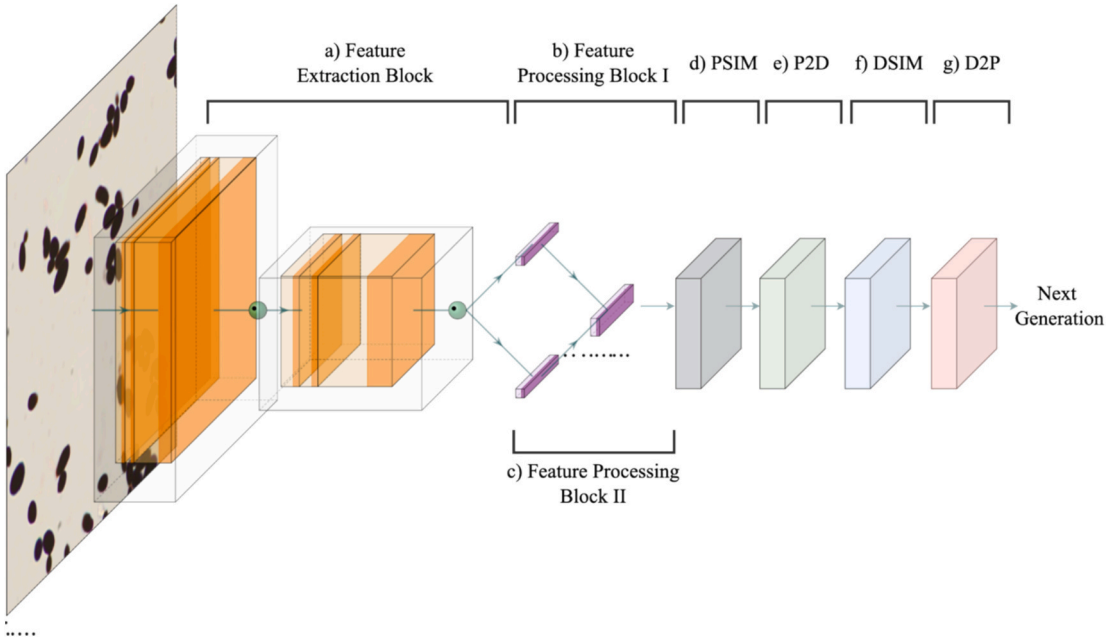



**Fig. 12.** The structure of the main model and damaged components in each ablation experiment.

**Table 4**
Performance comparison of various models on the DeFungi dataset in the ablation study.

| Method | Average Patch-based Accuracy % |
|---|---|
| Main Model | 88.7 |
| Damaged Model (a) | 72.92 |
| Damaged Model (b) | 70.88 |
| Damaged Model (c) | 73.80 |
| Damaged Model (d) | 72.09 |
| Damaged Model (e) | 72.16 |
| Damaged Model (f) | 69.85 |
| Damaged Model (g) | 67.36 |

## CRediT authorship contribution statement

**Pooya Sajjadi:** Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis. **Farshid Hajati:** Writing – review & editing, Supervision, Methodology, Data curation. **Alireza Rezaee:** Writing – review & editing, Supervision. **Shahadat Uddin:** Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

The used data is pubically available.

## References


[1] M. Ruhnke, "Epidemiology of Candida albicans Infections and Role of Non-Candidaalbicans Yeasts," *Curr Drug Targets*, vol. 7, no. 4, Apr. 2006, doi: 10.2174/138945006776359421.

[2] M. Nucci and K. A. Marr, "Emerging Fungal Diseases," *Clinical Infectious Diseases*, vol. 41, no. 4, Aug. 2005, doi: 10.1086/432060.

[3] C.E. Litz, R.Z. Cavagnolo, Polymerase chain reaction in the diagnosis of onychomycosis: a large, single-institute study, *Br. J. Dermatol.* 163 (3) (2010) Sep, https://doi.org/10.1111/j.1365-2133.2010.09852.x.

[4] Mm. Shenoy, S. Teerthanath, V. Karnaker, B. Girisha, M. Krishna Prasad, and J. Pinto, "Comparison of potassium hydroxide mount and mycological culture with histopathologic examination using periodic acid-Schiff staining of the nail clippings in the diagnosis of onychomycosis," *Indian J Dermatol Venereol Leprol*, vol. 74, no. 3, 2008, doi: 10.4103/0378-6323.39584.

[5] K. A. Haynes, T. J. Westerneng, J. W. Fell, and W. Moens, "Rapid detection and identification of pathogenic fungi by polymerase chain reaction amplification of large subunit ribosomal DNA," *Med Mycol*, vol. 33, no. 5, Jan. 1995, doi: 10.1080/02681219580000641.

[6] K. Makimura, S. Y. Murayama, and H. Yamaguchi, "Detection of a wide range of medically important fungi by the polymerase chain reaction," *J Med Microbiol*, vol. 40, no. 5, 1994, doi: 10.1099/00222615-40-5-358.

[7] T. R. Kozel and B. Wickes, "Fungal diagnostics," *Cold Spring Harb Perspect Med*, vol. 4, no. 4, 2014, doi: 10.1101/cshperspect.a019299.

[8] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, 1996, doi: 10.1109/3477.484436.

[9] H. A. Alsattar, A. A. Zaidan, and B. B. Zaidan, "Novel meta-heuristic bald eagle search optimisation algorithm," *Artif Intell Rev*, vol. 53, no. 3, 2020, doi: 10.1007/s10462-019-09732-5.

[10] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey Wolf Optimizer, *Adv. Eng. Softw.* 69 (2014), https://doi.org/10.1016/j.advengsoft.2013.12.007.

[11] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, *Futur. Gener. Comput. Syst.* 97 (2019), https://doi.org/10.1016/j.future.2019.02.028.

[12] A. Faramarzi, M. Heidarinejad, S. Mirjalili, A.H. Gandomi, Marine Predators Algorithm: A nature-inspired metaheuristic, *Expert Syst Appl* 152 (2020), https://doi.org/10.1016/j.eswa.2020.113377.

[13] J. Kennedy, R. Eberhart, Particle swarm optimization, *IEEE International Conference on Neural Networks - Conference Proceedings* (1995), https://doi.org/10.4018/ijmfmp.2015010104.

[14] S. Shadravan, H.R. Naji, V.K. Bardsiri, The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems, *Eng Appl Artif Intell* 80 (2019), https://doi.org/10.1016/j.engappai.2019.01.001.

[15] S. Mirjalili, A. Lewis, The Whale Optimization Algorithm, *Adv. Eng. Softw.* 95 (2016), https://doi.org/10.1016/j.advengsoft.2016.01.008.

[16] H. C. Shin *et al.*, "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning," *IEEE Trans Med Imaging*, vol. 35, no. 5, 2016, doi: 10.1109/TMI.2016.2528162.

[17] B. Zieliski, A. Sroka-Oleksiak, D. Rymarczyk, A. Piekarczyk, and M. Brzychczy-Woch, "Deep learning approach to describe and classify fungi microscopic images," *PLoS One*, vol. 15, no. 6 june, 2020, doi: 10.1371/journal.pone.0234806.

[18] M.E. Mital, et al., "Transfer learning approach for the classification of conidial fungi (genus aspergillus) thru pre-trained deep learning models," in *IEEE Region 10 Annual International Conference, Proceedings/tencon* (2020), https://doi.org/10.1109/TENCON50793.2020.9293803.

[19] Felipe A. Quezada, Carlos F. Navarro, and Cristian Muñoz, "RaViTT: Random Vision Transformer Tokens," *arxiv open access*, Jun. 2023.

[20] M. A. V. Álvarez, L. Sopó, C. J. P. Sopo, F. Hajati, and S. Gheisari, "P456 Defungi: direct mycological examination of microscopic fungi images," *Med Mycol*, vol. 60, no. Supplement_1, 2022, doi: 10.1093/mmy/myac072.p456.

[21] C. Sun, A. Shrivastava, S. Singh, A. Gupta, Revisiting Unreasonable Effectiveness of Data in Deep Learning Era, in: In *Proceedings of the IEEE International Conference on Computer Vision*, 2017, https://doi.org/10.1109/ICCV.2017.97.

[22] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans Pattern Anal Mach Intell*, vol. 28, no. 4, 2006, doi: 10.1109/TPAMI.2006.79.

[23] B.M. Lake, R. Salakhutdinov, J. Gross, J.B. Tenenbaum, One shot learning of simple visual concepts. In *in {proceedings of the 33rd Annual Conference of the Cognitive Science Society}*, 2011.

[24] F. Sung, Y. Yang, L. Zhang, T. Xiang, P.H.S. Torr, T.M. Hospedales, Learning to Compare: Relation Network for Few-Shot Learning, in: In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, https://doi.org/10.1109/CVPR.2018.00131.

[25] V. Garcia, J. Bruna, Few-shot learning with graph neural networks. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[26] J. Kim, T. Kim, S. Kim, C.D. Yoo, Edge-labeling graph neural network for few-shot learning, in: In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019, https://doi.org/10.1109/CVPR.2019.00010.

[27] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, Y. Liu, DPGN: Distribution propagation graph network for few-shot learning, in: In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020, https://doi.org/10.1109/CVPR42600.2020.01340.

[28] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-Based Classification for Zero-Shot Visual Object Categorization," *IEEE Trans Pattern Anal Mach Intell*, vol. 36, no. 3, Mar. 2014, doi: 10.1109/TPAMI.2013.140.

[29] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks. In *34th International Conference on Machine Learning*, 2017, 2017..

[30] O. Vinyals, C. Blundell, T. Lillicrap, koray kavukcuoglu, and D. Wierstra, "Matching Networks for One Shot Learning," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf.

[31] Q. Xu, M. Zhang, Z. Gu, G. Pan, Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs, *Neurocomputing* 328 (Feb. 2019), https://doi.org/10.1016/j.neucom.2018.03.080.

[32] P. W. Battaglia *et al.*, "Relational inductive biases, deep learning, and graph networks," Jun. 2018, [Online]. Available: http://arxiv.org/abs/1806.01261.

[33] O. Y. Bakhteev and V. V. Strijov, "Comprehensive analysis of gradient-based hyperparameter optimization algorithms," *Ann Oper Res*, vol. 289, no. 1, 2020, doi: 10.1007/s10479-019-03286-z.

[34] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for Hyper-Parameter Optimization.".

[35] H. Faris, I. Aljarah, M. A. Al-Betar, and S. Mirjalili, "Grey wolf optimizer: a review of recent variants and applications," *Neural Comput Appl*, vol. 30, no. 2, Jul. 2018, doi: 10.1007/s00521-017-3272-5.

[36] S. Eswaramoorthy, N. Sivakumaran, and S. Sekaran, "Grey Wolf optimization based parameter selection for support vector machines," *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 35, no. 5, 2016, doi: 10.1108/COMPEL-09-2015-0337.

[37] Z. Mustaffa, M.H. Sulaiman, M.N.M. Kahar, Training LSSVM with GWO for price forecasting, in: In *2015 4th International Conference on Informatics, Electronics and Vision, ICIEV*, 2015, 2015., https://doi.org/10.1109/ICIEV.2015.7334054.

[38] S. Mirjalili, How effective is the Grey Wolf optimizer in training multi-layer perceptrons, *Appl. Intell.* 43 (1) (Jul. 2015) 150–161, https://doi.org/10.1007/s10489-014-0645-7.

[39] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, https://doi.org/10.1109/CVPR.2016.90.

[40] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2010. doi: 10.1109/cvpr.2009.5206848.

[42] M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, and S. Mougiakakou, "Lung Pattern Classification for Interstitial Lung Diseases Using a Deep Convolutional Neural Network," *IEEE Trans Med Imaging*, vol. 35, no. 5, 2016, doi: 10.1109/TMI.2016.2535865.

[43] J. B. Epstein, "Antifungal therapy in oropharyngeal mycotic infections," *Oral Surgery, Oral Medicine, Oral Pathology*, vol. 69, no. 1, 1990, doi: 10.1016/0030-4220(90)90265-T.

[44] H. Mohamed, A. E. Nagib, and M. Hany, "Comparative Study of Microscopic Fungii Classification Using Transfer Learning Models," in *2024 Intelligent Methods, Systems, and Applications (IMSA)*, IEEE, Jul. 2024, pp. 87–92. doi: 10.1109/IMSA61967.2024.10652826.

[45] G. Singh, K. Guleria, and S. Sharma, "DeepFungusDet: MobileNetV3 Model in Medical Imaging for Fungal Disease Detection," in *2024 3rd International Conference for Innovation in Technology (INOCON)*, IEEE, Mar. 2024, pp. 1–6. doi: 10.1109/INOCON60754.2024.10511726.

[46] Sekine Asadi Amiri and Fatemeh Mohammady, VGG19-DeFungi: A Novel Approach for Direct Fungal Infection Detection Using VGG19 and Microscopic Images, *Journal of AI and Data Mining* 12 (2) (Apr. 2024).

[47] D. Rymarczyk, A. Borowa, J. Tabor, and B. Zielinski, "Kernel self-attention for weakly-supervised image classification using deep multiple instance learning," in *Proceedings - 2021 IEEE Winter Conference on Applications of Computer Vision, WACV 2021*, 2021. doi: 10.1109/WACV48630.2021.00176.

[48] R. Meyes, M. Lu, C. W. de Puiseau, and T. Meisen, "Ablation Studies in Artificial Neural Networks," Jan. 2019, [Online]. Available: http://arxiv.org/abs/1901.08644.