DOI: 10.14421/ijid.2023.4038 IJID (International Journal on Informatics for Development), *e-ISSN: 2549-7448*
Vol. 12, No. 2, December 2023, Pp. 350-361

Optimisation of Residual Network Using Data Augmentation and Ensemble Deep Learning for Butterfly Image Classification

Diniati Ruaika[1]*
Departement of Informatics
UIN Sunan Kalijaga
Yogyakarta, Indonesia
[1*]ruaikadiniati@gmail.com

Shofwatul 'Uyun[2]
Department of Informatics
UIN Sunan Kalijaga
Yogyakarta, Indonesia
[2]shofwatul.uyun@uin-suka.ac.id

Article History
Received June 1st, 2023
Revised December 25th, 2023
Accepted December 31st, 2023
Published January, 2024

Abstract*— Image classification is a fundamental task in vision recognition that aims to understand and categorize an image under a specific label. Image classification needs to produce a quick, economical, and reliable result. Convolutional Neural Networks (CNN) have proven effective for image analysis. However, problems arise due to factors such as the model's quality, unbalanced training data, overfitting, and layers' complexity. ResNet50 is a transfer learning-based convolutional neural network model frequently used in many areas, including Lepidopterology. Studies have shown that ResNet50 performs with lower accuracy than other models for classifying butterflies. Therefore, this study aims to optimise the accuracy of ResNet50 using an augmentation approach and ensemble deep learning for butterfly image classification. This study used a public dataset of butterflies from Kaggle. The dataset contains 75 different butterfly species, 9.285 training images, 375 testing images, and 375 validation images. A sequence of transformation functions was applied. The ensemble deep learning was constructed by incorporating ResNet50 with CNN. To measure ResNet50 optimisation, the experimental results of the original dataset and the proposed methods were compared and analysed using evaluation metrics. The research revealed that the proposed method provided better performance with an accuracy of 95%.

Keywords*— Butterfly; CNN; data augmentation; ensemble deep learning; ResNet50

1 INTRODUCTION

Along with moths, butterflies are insects belonging to the order of Lepidoptera. The world population of butterfly species is approximately 21,000 [1]. The existence of butterflies is crucial for maintaining ecosystem balance and enriching biodiversity [2]. Butterflies are not only natural pollinators for plants but also bioindicators of environmental change [3]–[5]. The extinction of butterflies in nature may indicate polluted environmental conditions [6], [7]. The wing pattern of a butterfly, including colour, wing shape, and eyespot size may evolve independently because of environmental cues [8]. This underscores the importance of studying butterflies as a means to understand and monitor the environment. However, the current classification of butterfly species is still based on the traditional method, in which butterflies are caught using insect nets [3], [9], [10]. This approach, unfortunately, grapples with challenges such as low accuracy and slow recognition due to the vast diversity of species, their striking similarities, and the lack of evident characteristics for clear differentiation [11]. Other weaknesses for traditional method are extended observation duration, low accuracy because many butterflies have similar or identical shapes that are sometimes difficult to distinguish, high cost, limited availability of taxonomists, and the risk of butterfly extinction [11], [12]. The other challenges to study computer vision algorithms for butterfly identification species are entomologists' difficulty in gathering the butterfly dataset, prolonged process of butterfly identification, and the incomplete dataset in terms of the number of butterfly species included. Moreover, the photographs of butterflies utilized for training were all pattern images with clear morphological characteristics, and they did not include any images of butterflies in their natural habitat. There are clear variations between the two images, making it challenging to combine manufacturing and research resulting in low identification accuracy [11]. With all these shortages, deep learning can help scientists or taxonomists analyse the butterfly images faster and more efficiently, automatically, and inexpensively. It allows for optimal identification accuracy without annihilating the population of butterflies' cause of capturing.

One of the most popular methods and a breakthrough in the field of image classification is Convolutional Neural Network (CNN) [13]. It is a multi-layer neural network model or deep learning architecture inspired by how the human brain works. The primary benefit of CNNs over earlier techniques is their capacity to automatically extract the most pertinent characteristics of the patterns of our selected pictures, eliminating the need for human feature extraction [14]. Typically, there are four layers making up the CNN architecture: the input layer, convolution layer, pooling layer, and fully connected layer [15]. Although CNN is the most widely used method in image classification, in its implementation, there are common problems such as imbalanced training data, overfitting, and the complexity of the layers, which lead to a long duration of the model training. To solve these problems, a simple CNN architecture is required [16]. Other options include the use of data augmentation approach and transfer learning techniques to solve the problems at CNN. Data augmentation allows the size of training datasets to become more complex by adding additional parameters such as cropping, horizontal flipping, rescaling, shear range, fill mode, rotation range, height shift, and width shift [1], [17]. Transfer learning is a method that uses a CNN model that has already been trained, also known as a pre-trained model [18]. In addition to the three techniques mentioned above (simple CNN, data augmentation, and transfer learning), there is a combination of several models called ensemble deep learning [19]. It can create better predictive models than just one model [20].

Transfer learning architectures have been used in several previous butterfly classification studies: the Inception V3, VGG19, VGG16, Xception, ResNet50, GoogLeNet, MobileNet, and LeNet architectures [1], [19]–[22]. Research [21] showed an accuracy of 79.5% for VGG16, 77.2% for VGG19, and 70.2% for ResNet50. The experiments did not involve any data augmentation. Other research using a data augmentation approach found 94.66% accuracy for InceptionV3, 92% for VGG19, 86.66% for VGG16, 87.99% for Xception, 81.33% for MobileNet, and 43.99% for ResNet50. Another study [20] found an accuracy of 97.5% for GoogleNet architecture. In addition, the data augmentation approach was also used by Prudhivi et al. [22]. The study using the VGG16, ResNet50, Inception V3, and MobileNet architectures resulted in consecutive accuracies of 86%, 53%, 95%, and 93%, respectively. Research [23] also used the VGG16 architecture and showed an accuracy of 93% and 67% for LeNet. The above research shows that data augmentation impacts accuracy. It was also found that the ResNet50 accuracy in butterfly classification research is the lowest compared with other architectures.

Based on the above issues, this research aims to improve the low accuracy of the ResNet50 architecture in butterfly classification. This study attempts to use two scenarios to perform ResNet50 optimisation of butterfly classification. The first scenario is to change the parameters in the data augmentation using the image data generator on the Keras framework for the following functions: image rotation, epsilon-zca brightening, fill mode, shear range, zoom range, horizontal rotation, width and height shift range, and last channel shift range. The second plan is to combine the ResNet50 architecture with a simple CNN architecture called Ensemble Deep Learning. Previous studies have shown that ensemble deep learning can achieve higher accuracy in image classification than a single model [20], [24]. Therefore, this research attempts to adapt it.

2 METHOD

2.1. Methodology Workflow

This part briefly explains the tool specifications and research flow. Fig. 1 depicts the methodology workflow. The workflow starts with collecting butterfly image datasets and then follows two scenarios: pre-processing (with data augmentation) and without pre-processing (without data augmentation). Both data are then applied to train the proposed models for training and validation. These are Resnet50, CNN, and the ensemble model. The trained model's performance was then examined further using accuracy scores and evaluation measures. The tool used to train the proposed model was the same device; therefore, no hardware factors affected the outcome of the study. The specifications of the device are listed in Table 1.

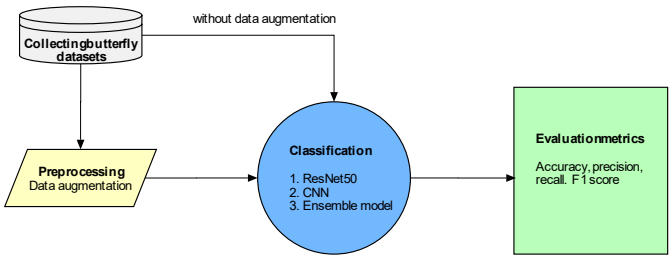

Figure 1. The methodology workflow

Table 1. Tools Details

Type	Detail type
Hardware	Processor Intel(R) Core (TM) i7-7500U with CPU @2.70GHz 2.90 GHz and RAM 8 GB NVIDIA GeForce MX150
Software	Windows 10 Home with version of Single Language 22H2, and Google Collaboratory
Language	Python 3

2.2. Dataset Description

The datasets of butterfly images in this research were obtained from the Kaggle public dataset. The URL was https://www.kaggle.com/datasets/gpiosenka/butterflyimages 40-species. The number of butterfly datasets in Kaggle is constantly being updated. The dataset obtained contains 75 butterfly species, with 9,285 images used for training, 375 images for testing, and 375 images for validation (see Table 2). The dimensions of all butterfly images are the same: 224 x 224 pixels. The images in the butterfly dataset include various features such as image orientation, camera angle, wing length, background complexity, and occlusion. Some examples of butterfly species listed in Figure 2 are Adonis, Banded Orange Butterfly, Cairns Birdwing, and Monarch.

Table 2. Butterfly Species

	Species	Train	Validation	Testing
1	Adonis	127	5	5
2	African Giant Swallowtail	107	5	5
3	American Snoot	105	5	5
4	An 88	121	5	5
5	Appollo	128	5	5
6	Atala	143	5	5
7	Banded Orange Heliconian	139	5	5
8	Banded Peacock	119	5	5
9	Beckers White	116	5	5
10	Black Hairstreak	121	5	5
11	Blue Morpho	107	5	5
12	Blue Spotted Crow	123	5	5
13	Brown Siproeta	141	5	5
14	Cabbage White	128	5	5
15	Cairns Birdwing	118	5	5
16	Checquered Skipper	136	5	5
17	Chestnut	122	5	5
18	Cleopatra	133	5	5
19	Clodius Parnassian	124	5	5
20	Clouded Sulphur	131	5	5
21	Common Banded Awl	125	5	5
22	Common Wood-Nymph	128	5	5
23	Copper Tail	134	5	5
24	Crecent	138	5	5
25	Crimson Patch	103	5	5
26	Danaid Eggfly	135	5	5
27	Eastern Coma	133	5	5
28	Eastern Dapple White	135	5	5
29	Eastern Pine Elfin	136	5	5
30	Elbowed Pierrot	117	5	5
31	Gold Banded	104	5	5
32	Great Eggfly	111	5	5
33	Great Jay	135	5	5
34	Green Celled Cattleheart	126	5	5
35	Grey Hairstreak	123	5	5
36	Indra Swallow	115	5	5
37	Iphiclus Sister	136	5	5
38	Julia	115	5	5
39	Large Marble	116	5	5
40	Malachite	104	5	5
41	Mangrove Skipper	125	5	5
42	Mestra	123	5	5
43	Metalmark	108	5	5
44	Milberts Tortoiseshell	137	5	5
45	Monarch	129	5	5
46	Mourning Cloak	187	5	5
47	Orange Oakleaf	125	5	5
48	Orange Tip	137	5	5
49	Orchard Swallow	109	5	5
50	Painted Lady	112	5	5
51	Paper Kite	129	5	5
52	Peacock	120	5	5
53	Pine White	123	5	5
54	Pipevine Swallow	120	5	5
55	Popinjay	121	5	5
56	Purple Hairstreak	113	5	5
57	Purplish Copper	132	5	5
58	Question Mark	110	5	5
59	Red Admiral	117	5	5
60	Red Cracker	137	5	5
61	Red Postman	127	5	5
62	Red Spotted Purple	123	5	5
63	Scarce Swallow	139	5	5
64	Silver Spot Skipper	119	5	5
65	Sleepy Orange	153	5	5
66	Sootywing	128	5	5
67	Southern Dogface	125	5	5
68	Straited Queen	124	5	5
69	Tropical Leafwing	118	5	5
70	Two Barred Flasher	109	5	5
71	Ulyses	120	5	5
72	Viceroy	115	5	5
73	Wood Satyr	102	5	5
74	Yellow Swallow Tail	107	5	5
75	Zebra Long Wing	108	5	5
	Total	**9285**	**375**	**375**

2.3 Data Augmentation

Data augmentation includes techniques that can increase the quality and size of butterfly training data [1]. All original images of butterflies were transformed in each epoch with different transformation functions using the Keras framework image data generator. Epsilon-ZCA whitening is the first in a series of consecutive operations carried out by data augmentation. The following are image rotation, width and height shift range, shear range, zoom range, channel shift range, blending mode, and horizontal flipping. As a result, the recently created images will have different variations of the same image. Fig. 2 shows the examples shown in Table 3 of transformed stack images and transformation functions used in this data augmentation study.

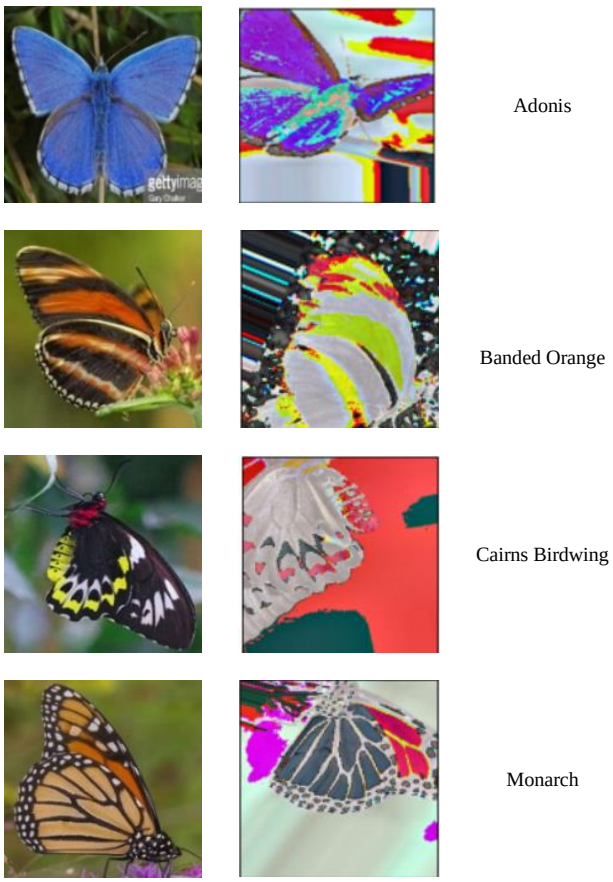

Figure 2. Data augmentation process

2.4 Deep Neural Networks

Deep learning has recently emerged as a significant working discipline for artificial intelligence applications. Deep learning's adoption rate may be credited to substantial successes in a variety of fields, including linked speech recognition [25], natural language processing [26], and human-computer interaction [27], object detection, pose estimation, face recognition, eye movement analysis, scene labelling, action recognition, object tracking [28]–[30]. Deep learning processes are based on artificial neural networks that are altered by neurons in the human brain that schematize [11]. By automatically removing the distinguishing characteristics from the input data, deep learning approaches can address the issues of feature extraction and selection. Deep learning, however, requires more detailed data than traditional neural networks.

Table 3. Transformation functions

Transformation functions	Description
epsilon zca whitening = 1e-06	apply ZCA whitening to Epsilon, which determines the degree of image correlation reduction. The higher the epsilon value, the smaller the correlation reduction
image rotation = 30	sets the degree range for random rotation.
width shift range = 0.2	the picture is shifted to the left or right (horizontal shift).
height shift range = 0.2	the tasks are the same as for the latitudinal shift range, but shear intensity is shifted up or down (vertically)
shear range = 0.2	shear intensity, i.e., anti-clockwise shear angle setting in degrees
zoom range = 0.3	randomizes the range for zooming
channel shift range = 0.	changes the saturation level of the colour (for example: bright red/dark red) of pixels by changing the channel [R, G, B] of the input image.
fill mode = 'nearest'	replaces the nearest pixel area. During image rotation, many pixels shift out of the image, generating unoccupied areas to be filled. The fill mode's default setting is nearest.
horizontal flip = true	flips the image horizontally

The first proposed model is CNN. CNN is an essential deep learning model that uses convolution rather than generic matrix multiplication [20]. The deep learning model of CNN is mainly utilized in image classification, object detection, and similarity detection [31], [32]. The typical CNN structure comprises numerous sequential layers of convolution and pooling, then fully linked layers, and lastly the output layer, or SoftMax layer, to classify the images. Figure 3 depicts the fundamental construction of CNN. Fig. 4 displays the precise layout and layered architecture of the CNN model employed in this study [33]. Input, convolution, pooling, and fully linked layers make up the foundation of the CNN architecture.

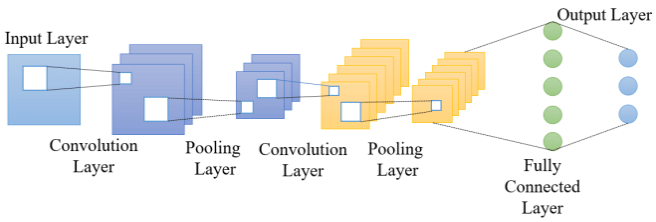

Figure 3. The structure of basic CNN

The steps used in the model are input image butterfly images with a size of 224x244 pixels; convolution layer; pooling layer; and fully connected layer. The success and resource requirements of the specified model heavily depend on the organization of the layer. The input data may need to have certain preparation techniques performed to it, such as noise reduction and scaling. Low-resolution photos as an input may result in a decline in the network's depth and functionality [11]. The input data undergoes a convolution process using feature selection filters in the convolution layer, also referred to as the transformation layer. Both randomly generated and pre-set filters are available. Data are transformed into a feature map as a result of the convolution process. For the subsequent convolution layer, the pooling layer serves to minimize the size of the input matrix [11]. The two most popular methods in the pooling layer are max pooling and average pooling. Then convolution and pooling layers are followed by the fully linked layer. The data from

the pooling layer is condensed into a single dimension in the fully connected layer. A neuron is referred to as fully linked because every neuron is connected. The process of classification is done at this layer, where activation functions like Sigmoid and ReLU are employed. The model's score values are used in the output layer to create the probability-based loss value using the Softmax function. In addition, two dense layers and two dropouts with a dropout area of 0.2 and Softmax activation are added to the fully connected layer. The dropout layer, which is employed when the model exhibits over-fitting (memorization), facilitates the process of removing certain connections that lead to overlearning. Consequently, some neurons in the fully linked layers are randomly removed by the model to stop the network from memorizing. Concisely, the proposed CNN model in this study created four 2D convolution layers (2DConv), three 2x2 max pooling layers, four activation layers (ReLU), four batch normalization layers, and global pooling with 2D global average pooling as the flattening. The loss function of the CNN model is a sparse categorical cross-entropy with the Adam optimiser. The model is then trained with 50 epochs.

The second proposed model is ResNet50. The residual Neural Network 50 (ResNet50) is a 50 layers convolutional neural network. This architecture was developed in 2015 by a team from Microsoft Research. The ResNet design was created in response to a surprise discovery in deep learning research: adding additional layers to a neural network did not necessarily improve the results. ResNet can have a very deep network of up to 152 layers by inserting a skip connection (also known as a shortcut connection) to fit the input from the previous layer to the next layer without altering the input [33]. By combining the output of an earlier layer with the output of a later layer, skip connections are created (See Fig. 5). These connections enabled the network to develop improved representations of the input data by preserving information from previous levels. ResNet-50 is made up of 50 layers separated into 5 blocks, each of which contains a collection of residual blocks. The residual blocks enable the network to develop better representations of the input data by preserving information from previous levels. The capacity to train enormously deep networks with hundreds of layers is one of ResNet's primary features. This is made feasible by the use of residual blocks and skip connections, which allow information from preceding levels to be preserved. Another benefit of ResNet-50 is its capacity to produce cutting-edge results in a variety of image-related tasks such as object identification, image classification, and picture segmentation. This architecture generally consists of three unique features. These include link skipping, stack normalization, and the elimination of fully connected layers. Convolution is performed on the input picture by the network's convolutional layer, which is its first layer. A max-pooling layer that down samples the convolutional layer's output comes next. Following the max-pooling layer, a number of residual blocks are applied to the output. A rectified linear unit (ReLU) activation function and a batch normalization layer come after each of the two convolutional layers that make up each residual block. Subsequently, the output of the second convolutional layer is combined with the input of the residual block, and it undergoes yet another ReLU activation function. The subsequent block receives the output of the residual block. The output of the last residual block is mapped to the output classes by the fully connected layer, which is the last layer of the network. The number of output classes is equal to the number of neurons in the fully linked layer. The hopping connection strategy in ResNet is then referred to as residual learning. The vanishing gradient problem that plagued prior CNN designs was substantially eliminated by ResNet's approach. Furthermore, ResNet is less complex than smaller networks such as VGGNet (19 layers). As a result, ResNet outperformed all ConvNets in the 2015 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition, using 152 layers and achieving at best-five error rate of 3.6% [34]. ResNet has emerged as the finest CNN architecture and practice standard since 2016 [30]. An illustration of the difference between the classic or basic CNN architecture and ResNet is shown in Fig. 6.

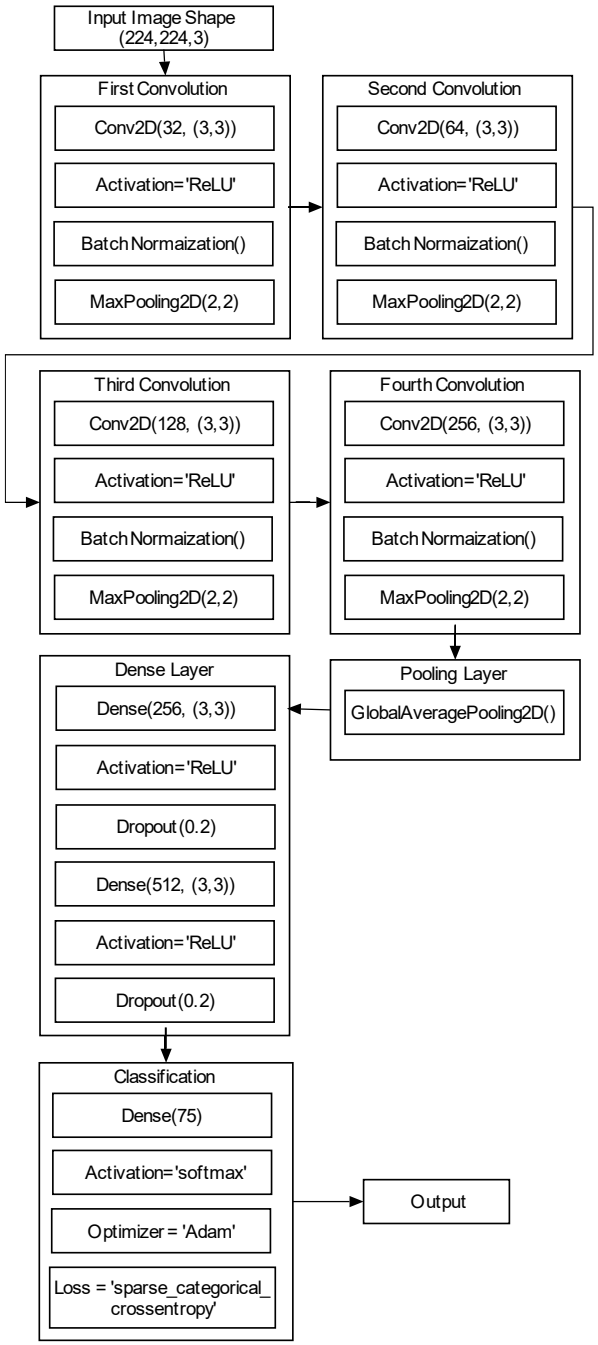

Figure 4. Structure and architecture of CNN

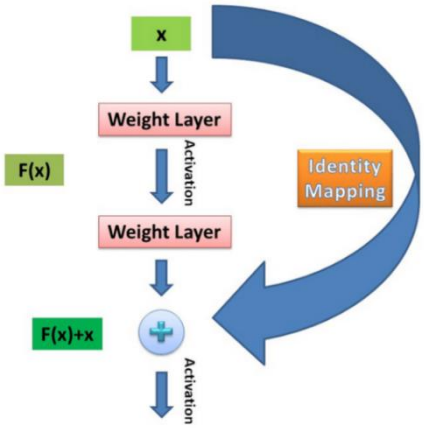

Figure 5. Skip connections of ResNet

Figure 7 depicts the ResNet50 architecture employed in the study. The procedures are outlined below. The input image is convolved on the convolution layer with a 7x7 filter and a 2-step size. The convolution method generates a feature map that has been batch normalized. The normalizing findings are then fed into the ReLU activation layer. Furthermore, before entering the second stage of convolution, the output value of the ReLU activation is lowered on the max-pooling layer. Between the second and fifth convolution stages, the next process is carried out using a combination of convolution blocks and identity blocks. The next step is to move to the fully linked layer phase to carry out the classification process by adding Softmax.

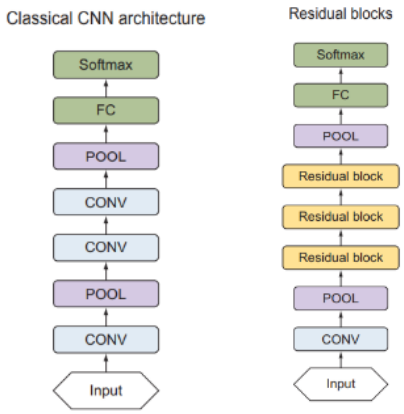

Figure 6. Differences between CNN and ResNet

The ensemble model is the third proposed model. Multiple models are trained on the same data set, and predictions are created by combining the forecasts in some fashion to provide the final prediction. [24]. The ensemble model is an advanced learning process that refines performance overall by integrating conclusions from numerous models [35]. In this study, an ensemble model is created by combining the ResNet50 architecture and the CNN architecture. Fig. 8 shows the process of ensemble learning and the value of the parameters.

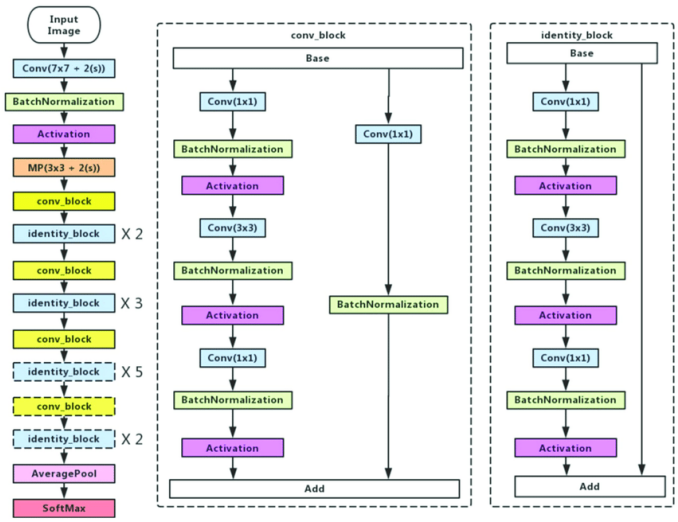

Figure 7. Structure and architecture of ResNet50

2.5 *Performance evaluation*

It is critical to understand how effectively the deep learning models function. Some analysis parameters are employed for this purpose. The models may categorize the data into four groups: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP represents how accurate the predicted positive cases are, whereas TN classifies them accurately as negative cases. Forward, FP, and FN are all mistakenly labelled as positive and negative. As performance evaluation measures in this study, accuracy, precision, recall, and F1 score values were used. The number of correctly categorized predictions per model divided by the total number of predicted cases is used to calculate accuracy [18]. Equation (1) depicts the equation for computing the accuracy value. Precision has been defined as the proportion of the model's total positive predictions that were correctly predicted [20]. The precision value is calculated using Equation (2). The number of true positive predictions, for instance, accurately mapped to the positive class, is recall. Equation (3) is used to calculate recall. The model's approximate performance, based on average accuracy and recall, is then determined using the F1 score. The F1 score is calculated using Equation (4).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

$$\text{F1} - \text{score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} \times \text{recall}} \quad (4)$$

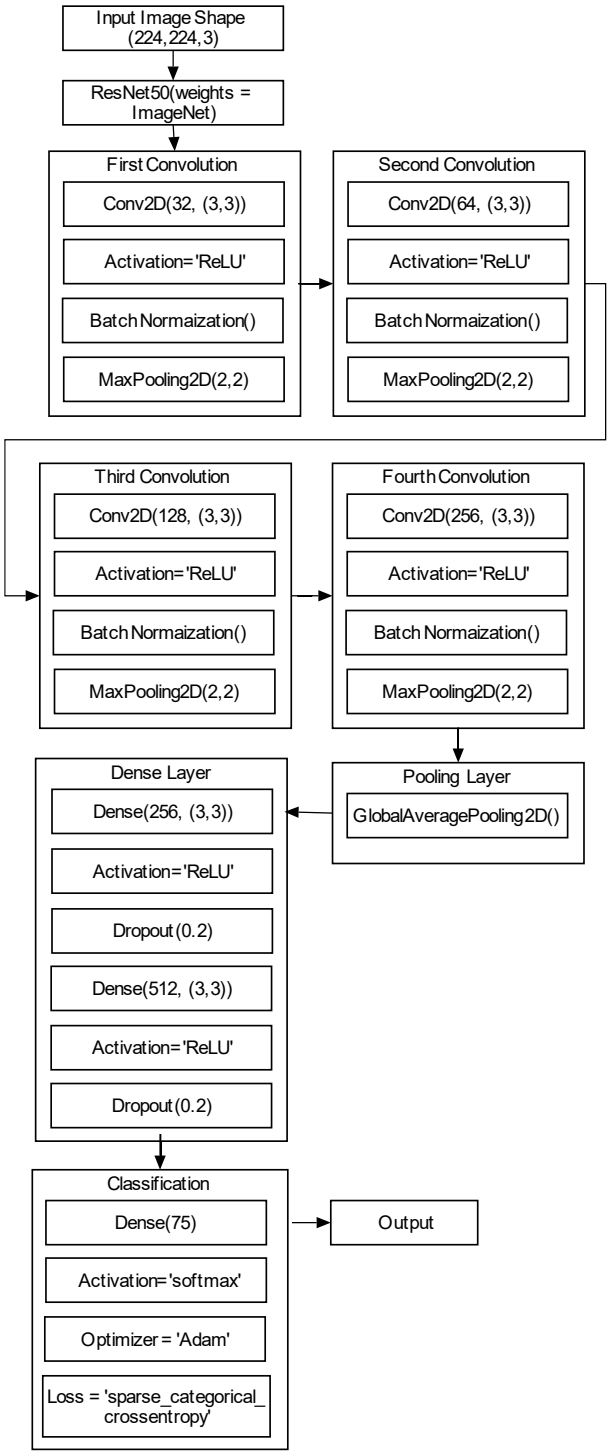

Figure 8. Ensemble model architecture

3 RESULT AND DISCUSSION

TensorFlow was used in the Google Collaboratory to run deep learning experiments. Three transfer learning models were presented using the Python 3 programming language and the Keras package. The models were ResNet50, CNN, and the Ensemble model (ResNet50 with CNN). The architecture of ResNet50 in this study was adopted from Ji et al. [33], while CNN's architecture and ensemble model were built independently. Table 4 lists the model parameters that were used in the models. All models were equipped with 50 epochs and 128 batch sizes. Sparse categorical cross-entropy was employed to boost the learning rate and save memory time [36]. The Adam optimiser was used for all models.

The accuracy and loss curves were acquired when the deep learning models were launched. Table 5 displays the training and validation accuracy at the conclusion of 50 epochs. With no data augmentation, Resnet50 had the lowest accuracy of any model, with a training accuracy of 88% and a validation accuracy of 69%. CNN achieved 91% training accuracy and 86% validation accuracy. Subsequently, the ensemble model achieved 100% training accuracy and 95% validation accuracy. Validation accuracy is more meaningful because it gives an indication of how successful the deep learning models are at correctly classifying the data they haven't seen before [37]. Therefore, particular importance was attached to validation accuracy. As the data expanded, each model improved. Resnet50 validation accuracy had increased to 85%, CNN to 87%, and the ensemble model to 97%. A similar improvement in ResNet50 can also be seen in validation loss accuracy. The data expansion reduced the accuracy of the ResNet50 validation loss from 1.43% to 0.61%. In contrast, the ensemble model achieved the lowest validation loss accuracy at 0.113%. Figure 9-20 depicts the accuracy and loss curves for each model. According to the graphs, data augmentation enhanced image training accuracy.

Table 4. Model Parameters

Parameters names	**Value**
Input sizes	224 x 224
Epoch	50
Batch size	128
Number of classes	75
Number of training images	9.285
Number of test images	375
Optimiser	Adam
Loss function	Sparse categorical cross entropy

Table 5. Training and Validation Accuracy

Models		**ResNet50**	**CNN**	**Ensemble**
Without Data Augmentation	Train acc.	0.88	0.91	1.00
	Valid acc.	0.69	0.86	0.95
	Train loss	0.58	0.48	0.0015
	Valid loss	1.43	0.58	0.33
With Data Augmentation	Train acc.	0.90	0.87	0.97
	Valid acc.	0.85	0.87	0.97
	Train loss	0.31	0.43	0.084
	Valid loss	0.61	0.42	0.113

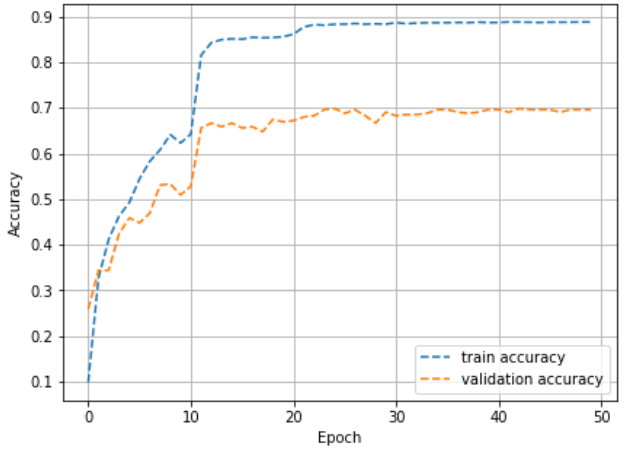

Figure 9. Resnet50 accuracy (without data augmentation)

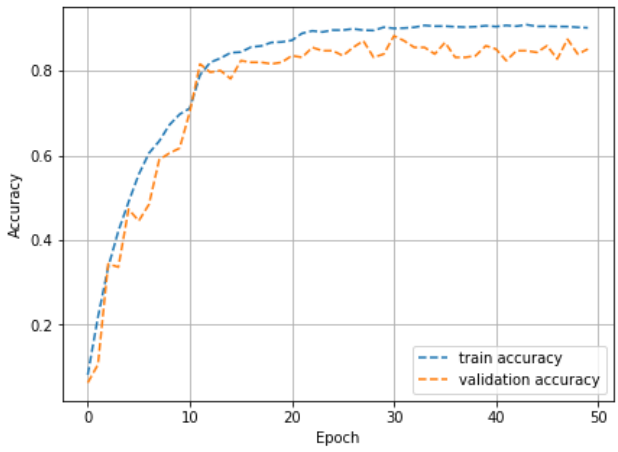

Figure 10. Resnet50 accuracy (with data augmentation)

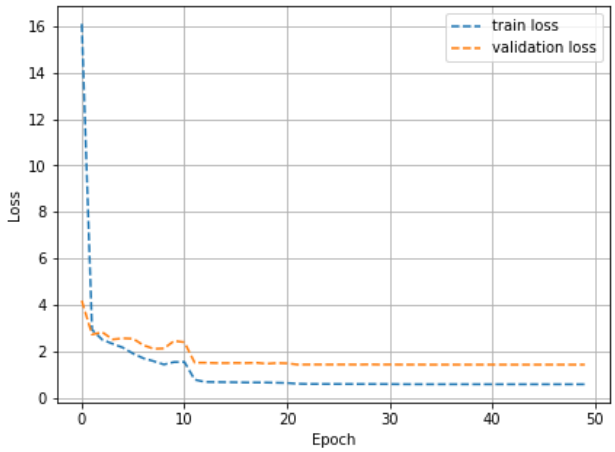

Figure 11. Resnet50 loss (without data augmentation)

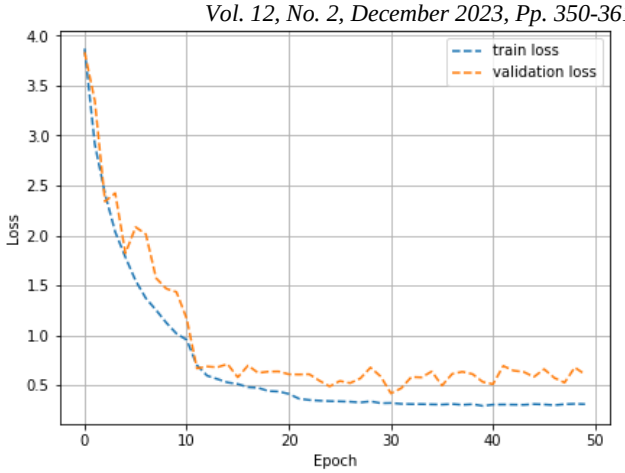

Figure 12. Resnet50 loss (with data augmentation)

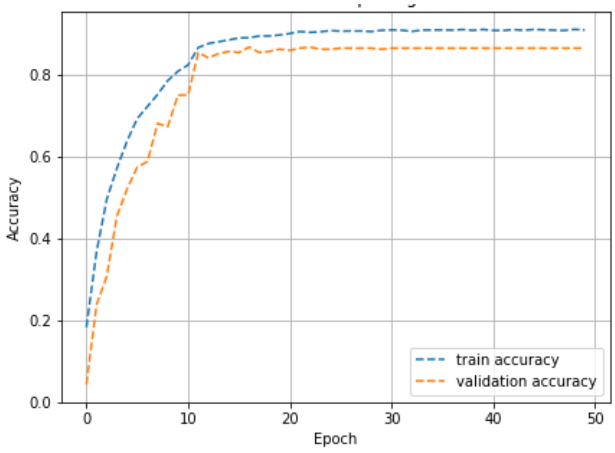

Figure 13. CNN accuracy (without data augmentation)

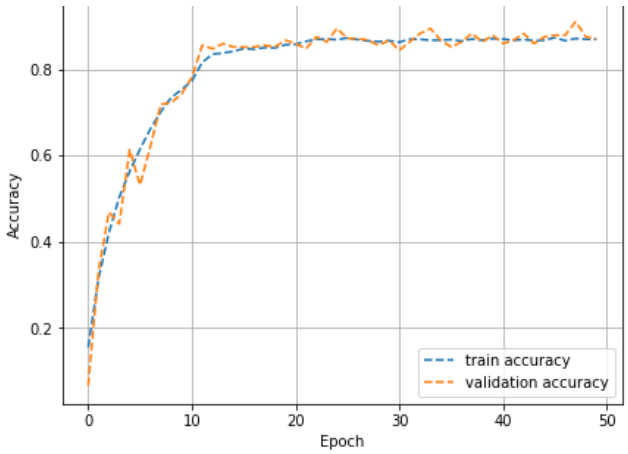

Figure 14. CNN accuracy (without data augmentation)

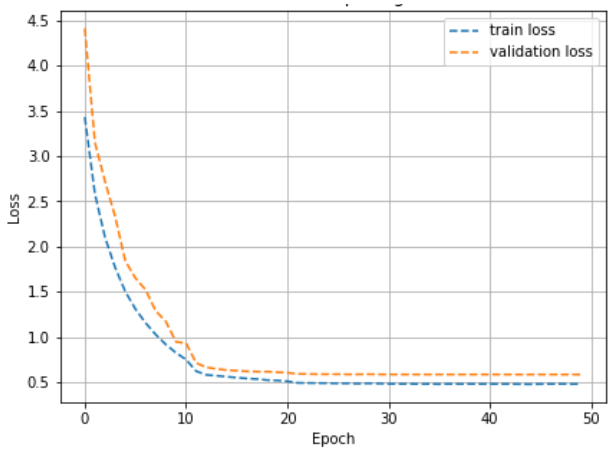

Figure 15. CNN loss (without data augmentation)

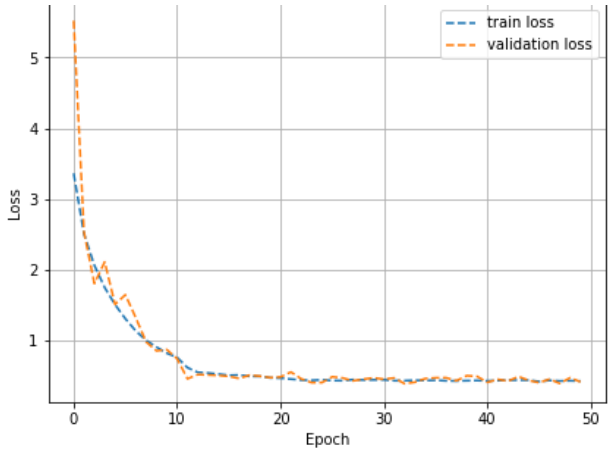

Figure 16. CNN loss (with data augmentation)

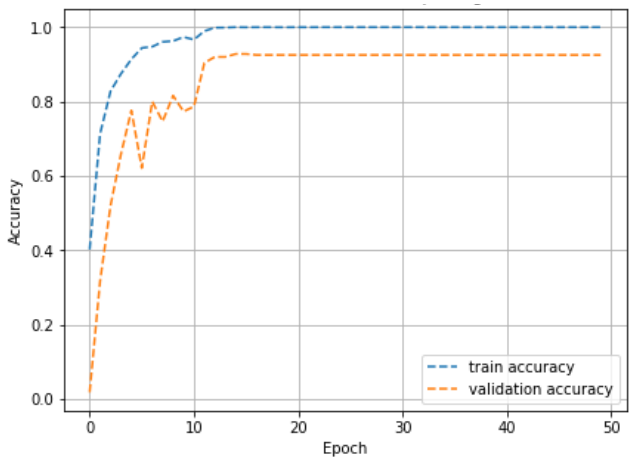

Figure 17. Ensemble model accuracy (without data augmentation)

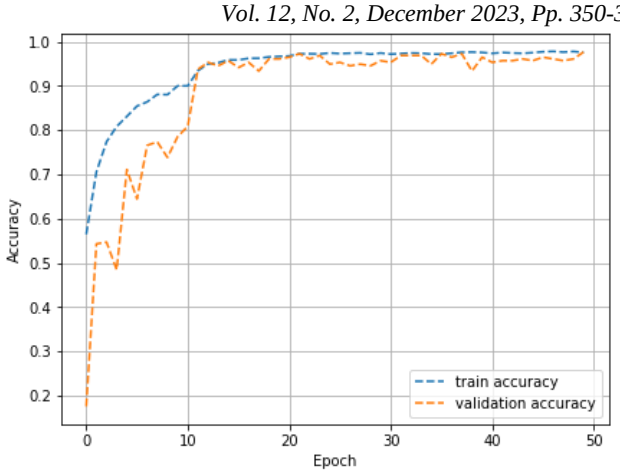

Figure 18. Ensemble model accuracy (with data augmentation)

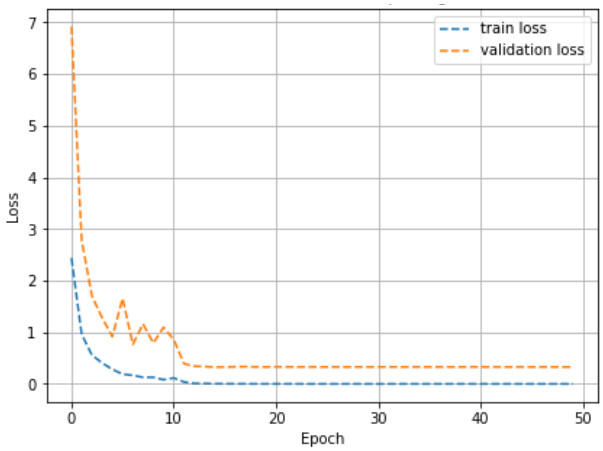

Figure 19. Ensemble model loss (without data augmentation)

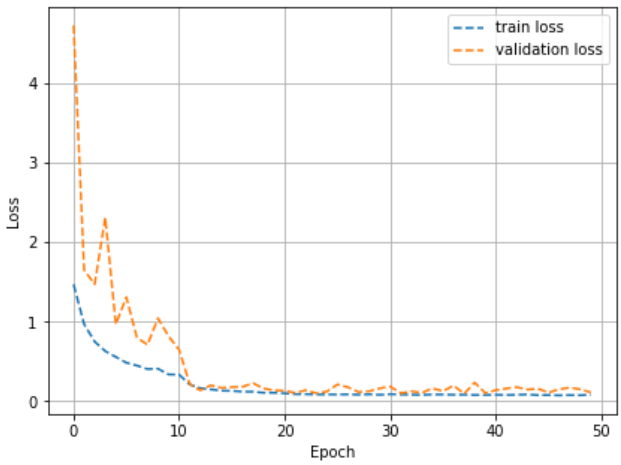

Figure 20. Ensemble model loss (with data augmentation)

Figs. 9 and 10 display the training and validation accuracy curves of ResNet50 using data augmentation and no data augmentation. After 10 epochs, the training accuracy grew dramatically and stabilized after 20 epochs. The validation accuracy curve showed a similar pattern, albeit with lower accuracy, and there was a gap between the curves. The accuracy curves of CNN training and validation were then shown in Figures 13 and 14. At the beginning of the 10 epochs, there were some points of variation in validation accuracy, with the gap being smaller than ResNet50. After data augmentation, the curves were crowded together, with some points going up and down. In Figs. 15 and 16, the loss curves show better performance than ResNet50, although there were some ups and downs in the first 10 epochs. Overall, the data augmentation improved accuracy, and the loss of CNN was not as significant as with ResNet50. Subsequently, the ensemble model's accuracy and loss curves are depicted in Figs. 17, 18, 19, and 20. As with the other deep learning models, there were some deviations in the first ten epochs. With data augmentation, the pattern still appeared in the first ten epochs. Therefore, the training images were slightly improved by the data augmentation.

In addition to the accuracy and loss curves, evaluation metrics were also used as parameters to determine model performance. The evaluation metrics' test results are shown in Table 6. ResNet50 outperformed other models in terms of accuracy after data augmentation. This model previously achieved 69% accuracy, 72% precision, 69% recall, and a 69% F1 score. After data augmentation, ResNet50 then achieved higher accuracy of 86%, 87% precision, 86% recall, and an 85% F-1 score. These results indicate that data augmentation could improve ResNet50 performance. CNN came to the same conclusion. Without data augmentation, the model obtained 86% accuracy, 88% precision, 86% recall, and an 86% F1 score. Then data augmentation improved to 88%, 89% precision, 88% recall, and an 88% F1 score. For the ensemble model, the mixture of ResNet50 and CNN attained higher accuracy than ResNet50. The accuracy increased to 93%, 93% of precision, 93% of recall, and 92% of F1 score. With data augmentation, the ensemble model attained better accuracy at 95% precision, 96% recall, and 95% F1 score. This finding indicated that the ensemble model could improve the performance of ResNet50.

Table 6. Evaluation Metrics

Models		ResNet50	CNN	Ensemble
Without Data Augmentation	Accuracy	0.69	0.86	0.93
	Precision	0.72	0.88	0.93
	Recall	0.69	0.86	0.93
	F-1 Score	0.69	0.86	0.92
With Data Augmentation	Accuracy	0.86	0.88	0.95
	Precision	0.87	0.89	0.96
	Recall	0.86	0.88	0.95
	F-1 Score	0.85	0.88	0.95
	Support	375	375	375

Table 7. Performance Comparison with Different Approaches

Ref.	Dataset	Pre-processing	Techniques	Optimiser	Accuracy (%)
[11]	10 species (900 images)	No data augmentation	VGG16, VGG19 ResNet50	AdaDelta	79,5 77,2 70,2
[1]	15 species (1125 images)	cropping, rescaling, horizontal flips, fill mode, shear range, width shift, height shit, dan rotation range	VGG16 VGG19 MobileNet Xception ResNet50 Inception V3	SGD Adam SGD SGD SGD RMSProp	86,6 92 81,3 87,9 43,9 94,6
[22]	1000 images	flipping, rotation, translation, cropping, geometric transformation dan color space	VGG16, ResNet50 InceptionV3 MobileNet	RMSProp	86 53 95 93
This study	75 species (10035 images)	epsilon zca whitening, image rotation, width shift range, height shift range, shear range, zoom range, channel shift range, fill mode horizontal flip, dan pre-processing function.	ResNet50 CNN ResNet50 + CNN	Adam	86 88 95

A performance comparison between the suggested technique and current state-of-the-art approaches was done to demonstrate the importance of the suggested strategy. The comparative results, particularly in relation to ResNet50, are presented in Table 7, aligning with previous studies on butterfly image recognition. By combining CNN with ResNet50, the present method outperforms the selected studies, achieving an impressive 95% accuracy in butterfly image recognition, as indicated by the performance comparison.

4 CONCLUSION

The proposed research aimed to optimise ResNet50 accuracy in classifying butterfly species from visual images. Data augmentation was utilized to improve the ResNet50 accuracy and thus the dataset's quality. Several transformation functions were sequentially implemented. The result of the experiment revealed that applied data augmentation might improve ResNet50 accuracy by up to 85%. The ensemble model has also been used to ResNes50 with CNN. With data augmentation, the ensemble model of ResNet50 attained an accuracy of up to 95%. These results indicate that ResNet50 accuracy can be optimised by applying data augmentation and ensemble deep learning. This research can be further extended by combining ResNet50 with other deep-learning models to find the best performance.

AUTHOR'S CONTRIBUTION

Diniati Ruaika is the first author who primarily conducted the research, wrote the paper, and managed the technical issues. The research was supervised by the second author, Shofwatul Uyun, who provided suggestions in how to analyse the data and writing method.

DECLARATION OF COMPETING INTERESTS

Conforming to the publication ethics of IJID journal, the authors, Diniati Ruaika and Shofwatul Uyun declared that the article is no potential conflict and competing of interest.

REFERENCES

[1] R. P. P. Fathimathul *et al.*, "A Novel Method for the Classification of Butterfly Species Using Pre-Trained CNN Models," *Electron.*, vol. 11, no. 13, pp. 1–20, 2022.

[2] F. Rohman, M. A. Efendi, and L. R. Andrini, *BIOEKOLOGI*, 1st ed. Malang: FMIPA UNM, 2019.

[3] J. An and S.-W. Choi, "Butterflies as an indicator group of riparian ecosystem assessment," *J. Asia. Pac. Entomol.*, vol. 24, 2020.

[4] N. Ismail, A. Awg Abdul Rahman, M. Mohamed, M. F. Abu Bakar, and L. Tokiman, "Butterfly as bioindicator for development of conservation areas in Bukit Reban Kambing, Bukit Belading and Bukit Tukau, Johor, Malaysia," *Biodiversitas J. Biol. Divers.*, vol. 21, pp. 334–344, 2020.

[5] S. Chowdhury *et al.*, "Insects as bioindicator: A hidden gem for environmental monitoring," *Front. Environ. Sci.*, vol. 11, 2023.

[6] A. M. Shephard, A. M. Zambre, and E. C. Snell-Rood, "Evaluating costs of heavy metal tolerance in a widely distributed, invasive butterfly," *Evol. Appl.*, vol. 14, no. 5, pp. 1390–1402, 2021.

[7] N. N. Kamaron Arzar, N. Sabri, N. F. Mohd Johari, A. Amilah Shari, M. R. Mohd Noordin, and S. Ibrahim, "Butterfly Species Identification Using Convolutional Neural Network (CNN)," *2019 IEEE Int. Conf. Autom. Control Intell. Syst. I2CACIS 2019 - Proc.*, no. June, pp. 221–224, 2019.

[8] K. R. L. Van Der Burg and R. D. Reed, "ScienceDirect Seasonal plasticity : how do butterfly wing pattern traits evolve environmental responsiveness ?," *Curr. Opin. Genet. Dev.*, vol. 69, no. Figure 1, pp. 82–87, 2021.

[9] A. Ghosh, M. S. Abedin, A. J. Howlader, and M. M. Hossain, "Molecular identification and phylogenetic relationships of seven Satyrinae butterflies in Bangladesh using Cytochrome c oxidase subunit I gene," *Jahangirnagar Univ. J. Biol. Sci.*, vol. 8, no. 1, pp. 67–74, 2019.

[10] A. Paramita, M. Syazali, and M. Erfan, "Identifikasi Spesies Kupu-Kupu di Taman Narmada Lombok Barat," *J. Sci. Educ.*, vol. 02, no. 1, pp. 22–25, 2022.

[11] A. S. Almryad and H. Kutucu, "Automatic identification for field butterflies by convolutional neural networks," *Eng. Sci. Technol. an Int. J.*, vol. 23, no. 1, pp. 189–195, 2020.

[12] H. M. Geyle *et al.*, "Butterflies on the brink: identifying the Australian butterflies (Lepidoptera) most at risk of extinction," *Austral Entomol.*, vol. 60, no. 1, pp. 98–110, 2021.

[13] M. Chikkamath, D. Dwivedi, R. B. Hirekurubar, and R. Thimmappa, "Benchmarking of Novel Convolutional Neural Network Models for Automatic Butterfly Identification," in *Computer Vision and Robotics*, 2023, pp. 351–364.

[14] C. Cunha, H. Narotamo, A. Monteiro, and M. Silveira, "Detection and measurement of butterfly eyespot and spot patterns using convolutional neural networks," *PLoS One*, vol. 18, no. 2, pp. 1–15, 2023.

[15] A. Ghosh, A. Sufian, F. Sultana, A. Chakrabarti, and D. De, *Fundamental concepts of convolutional neural network*, vol. 172, no. June. 2020.

[16] D. Prasetyawan and S. 'Uyun, "Penentuan Emosi pada Video dengan Convolutional Neural Network," *JISKA (Jurnal Inform. Sunan Kalijaga)*, vol. 5, no. 1, pp. 23–35, 2020.

[17] E. Lashgari, D. Liang, and U. Maoz, "Data Augmentation for Deep-Learning-Based Electroencephalography," *J. Neurosci. Methods*, vol. 346, pp. 1–55, 2020.

[18] S. T. Krishna and H. K. Kalluri, "Deep Learning and Transfer Learning Approaches for Image Classification," *Int. J. Recent Technol. Eng.*, vol. 7, no. 5, pp. 427–432, 2019.

[19] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan, "Ensemble deep learning: A review," *Eng. Appl. Artif. Intell.*, vol. 115, 2022.

[20] M. Mujahid, F. Rustam, R. Álvarez, J. Luis Vidal Mazón, I. de la T. Díez, and I. Ashraf, "Pneumonia Classification from X-ray Images with Inception-V3 and Convolutional Neural Network," *Diagnostics*, vol. 12, no. 5, pp. 1–16, 2022.

[21] S. H. Amrullah, Hilda, and R. F. Rusli, "Identifikasi lebah dan kupu polinator di Hutan Billa Battang Kota Palopo," *J. Din.*, vol. 09, no. 2, pp. 1–12, 2018.

[22] L. Prudhivi, M. Narayana, C. Subrahmanyam, and M. Gopi Krishna, "Animal species image classification," *Mater. Today Proc.*, no. xxxx, 2021.

[23] Micheal and E. Hartati, "Klasifikasi Spesies Kupu Kupu Menggunakan Metode Convolutional Neural Network," *MDP Student Conf. 2022*, pp. 569–577, 2022.

[24] T. Zhou, H. Lu, Z. Yang, S. Qiu, B. Huo, and Y. Dong, "The ensemble deep learning model for novel COVID-19 on CT images.," *Appl. Soft Comput.*, vol. 98, p. 106885, Jan. 2021.

[25] P. Gurunath Shivakumar and P. Georgiou, "Transfer learning from adult to children for speech recognition: Evaluation, analysis and recommendations," *Comput. Speech Lang.*, vol. 63, p. 101077, 2020.

[26] S. Ruder, M. E. Peters, S. Swayamdipta, and T. Wolf, "Transfer Learning in Natural Language Processing," in *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Tutorials*, 2019, pp. 15–18.

[27] Z. Lv, F. Poiesi, Q. Dong, J. Lloret, and H. Song, "Deep Learning for Intelligent Human–Computer Interaction," *Appl. Sci.*, vol. 12, no. 22, 2022.

[28] C. Stoean *et al.*, "Unsupervised Learning as a Complement to Convolutional Neural Network Classification in the Analysis of Saccadic Eye Movement in Spino-Cerebellar Ataxia Type 2," 2019, pp. 26–37.

[29] R. Stoean, C. Stoean, A. Samide, and G. Joya, "Convolutional Neural Network Learning Versus Traditional Segmentation for the Approximation of the Degree of Defective Surface in Titanium for Implantable Medical Devices," 2019, pp. 871–882.

[30] A. Samide, C. Stoean, and R. Stoean, "Surface study of inhibitor films formed by polyvinyl alcohol and silver nanoparticles on stainless steel in hydrochloric acid solution using Convolutional Neural Networks," *Appl. Surf. Sci.*, vol. 475, 2018.

[31] O. Kembuan, G. Caren Rorimpandey, and S. Milian Tompunu Tengker, "Convolutional Neural Network (CNN) for Image Classification of Indonesia Sign Language Using Tensorflow," *2020 2nd Int. Conf. Cybern. Intell. Syst. ICORIS 2020*, no. 26, 2020.

[32] D. Bhatt *et al.*, "CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope,"

Electronics, vol. 10, no. 20, 2021.

[33] Q. Ji, J. Huang, W. He, and Y. Sun, "Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images," *Algorithms*, vol. 12, no. 3, pp. 1–12, 2019.

[34] Z. Zahisham, C. P. Lee, and K. M. Lim, "Food Recognition with ResNet-50," in *2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, 2020, pp. 1–5.

[35] V. Rupapara, F. Rustam, H. F. Shahzad, A. Mehmood, I. Ashraf, and G. S. Choi, "Impact of SMOTE on Imbalanced Text Features for Toxic Comments Classification Using RVVC Model," *IEEE Access*, vol. 9, pp. 78621–78634, 2021.

[36] J. Kakarla, B. V. Isunuri, K. S. Doppalapudi, and K. S. R. Bylapudi, "Three-class classification of brain magnetic resonance images using average-pooling convolutional neural network," *Int. J. Imaging Syst. Technol.*, vol. 31, no. 3, pp. 1731–1740, 2021.

[37] D. Theckedath and R. R. Sedamkar, "Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks," *SN Comput. Sci.*, vol. 1, no. 2, p. 79, 2020.