Multi Image Classification using CNN Model

Saurab Kharel

October 5, 2024

Abstract

Traditional approaches for identifying or classifying objects, animals, species, or images are often time-consuming due to extensive data processing requirements. These methods are generally effective only in controlled scenarios, where data has already been processed, tagged, and stored in a database. However, they struggle to identify new or unseen images quickly, as they require specific, hand-crafted features to recognize known patterns.

To address these limitations, Convolutional Neural Networks (CNNs) have been developed, enabling more efficient image classification and identification. CNNs can automatically learn and extract features from raw images, often achieving high accuracy with minimal preprocessing. The accuracy of CNN-based models depends heavily on the chosen pretrained architecture and its limitations. Popular architectures include ResNet-18, ResNet-34, ImageNet, AlexNet, VGG19, and ViT, each offering unique trade-offs in terms of accuracy, speed, and computational efficiency.

The main objective of this research is to analyze pre-trained dataset using CNN model to find the high accuracy and low loss during the training and validation data. Different techniques have been used to improve generalization and speed up training. To gain the highest level of accuracy and evaluate each model, the assessment was conducted using different training models to get the accuracy, precision, recall and F1 score as evaluation metrics.

1 Introduction

Image identification and classification by family or species have traditionally been challenging tasks in data science, requiring manual feature extraction and often yielding limited accuracy. Convolutional Neural Networks (CNNs) have transformed this area by automating feature extraction, utilizing convolution operations and linear algebra principles to identify intricate patterns within image data. CNNs analyze pixel relationships and spatial hierarchies, making them highly effective for tasks requiring precise visual identification. Today, CNNs are used in medical image analysis, facial recognition, and autonomous driving, where accurate classification is crucial.

Traditional methods like color histograms allow for basic color-based comparisons but lack spatial context, meaning images with different contents may appear similar if their color distributions align. In contrast, CNNs capture complex image details such as color, shape, and texture, enabling more accurate classification.

This project leverages CNNs to classify butterfly species based on features like size and color, aiming for high accuracy. Key techniques include data augmentation, which prevents overfitting by enhancing model generalization, and transfer learning, which uses pre-trained models to streamline the training process. Pre-trained CNNs, such as ResNet-18, EfficientNet-B0, and ViT-Base, are adapted for this task, each offering unique strengths in classification accuracy and computational efficiency.

The project's primary objective is to maximize classification accuracy while minimizing training and validation loss. Performance is rigorously assessed using metrics like accuracy, precision, recall, and F1 score. Experimental outcomes demonstrate CNNs' effectiveness in butterfly species identification, showcasing a powerful and structured approach that outperforms traditional methods in handling complex image data.

2 Additional Research Related Works

Before deep learning, researchers used traditional techniques like the gray-level co-occurrence matrix (GLCM) and local binary patterns (LBP) for butterfly classification. These approaches required manual feature extraction and were limited by dataset size and controlled environments, often reaching accuracy rates below 70%, although they could exceed 90% in ideal conditions. However, these methods lacked scalability and were less effective when applied to diverse, real-world datasets.

The rise of artificial intelligence and deep learning has dramatically improved classification accuracy, now often surpassing 97%. Convolutional Neural Networks (CNNs) have been especially transformative, allowing automatic feature learning for complex classification tasks. CNNs excel at distinguishing fine-grained differences between butterfly species, automating the extraction of patterns such as color, texture, and shape.

These advancements, combined with the computational power of GPUs, have enabled researchers to process large datasets efficiently and with greater accuracy. Modern AI-driven methods eliminate the need for manual feature selection, providing robust, scalable classification solutions. This shift has made butterfly classification more accessible and effective, enhancing image recognition capabilities across various applications. Today, CNNs have become the cornerstone of image classification, offering powerful tools for researchers tackling fine-grained species identification and other challenging visual tasks.

In 2005, H. Zhang, Y. Wang, C. Zhang and his team explored the possibility of classifying the butterfly using Fourier transform to analyze wing pattern frequency. Their main objective was to corelate the frequency-domain analysis and predict the butterfly species based on wing texture and their pattern. With the implementation of Fourier transform, the accuracy was 60-70% predictable but with the limitation for species with overlapping frequency characteristics.

Similarly, K. Watanabe, M. Hori, and Y. Takeuchi and his team used geometric morphometric data of butterfly wings to identify the butterfly by extracting the landmark from wing images. The method focused on tallying specific wing shapes and vein position across species allowing for quantitative shape comparison with the accuracy of 75% for clearly identifiable shapes but wasn't effective in case of minor shape variation.

The above two examples illustrate some of the research methods used prior to the introduction of pre-CNN identification approaches with a lot of limitations.

With the advent of deep learning models, particularly Convolutional Neural Networks (CNNs), identification accuracy has significantly improved. CNNs enable models to learn complex data patterns and features, making them highly effective in analyzing and identifying similarities among butterfly species.

Wen et al.'s Innovative Insect Classification Approach built a complex model of identifying the morphological diversity of insect species. The model represents a significant advancement in automated insect classification, demonstrating the power of multi-classifier approaches in handling the complex morphological diversity of insect species using three primary classifiers like KNN (K-Nearest Neighbor), Decision Tree (DT), Nearest Mean Classifier (NMC) with the accuracy of 86.6% on the real world.

Hernandez et al. developed a multi-species classification model achieving 91% accuracy across 740 species. Their approach integrated geometrical, texture, and morphological features to enhance species identification. By leveraging diverse datasets and advanced descriptors like Area, Perimeter, Entropy, and Hu1 morphological indicators, they demonstrated remarkable performance in biological specimen classification. The research underscores the potential of comprehensive feature extraction for robust species recognition.

Zhu et al.'s Innovative Butterfly Classification Method focused on the two powerful computational strategies Region Matching, Dual-Complex Discrete Wavelet Transform (DCDWT) to classify the complex insects. It focused mainly on critical morphological features with the emphasis on the key areas like Wing patterns, edge characteristics, specific spots etc. With this approach, the team were able to achieve Achieved 84.47% accuracy on a provided dataset. It clearly denotes the big achievement in computational taxonomy offering the best approach to identify and classify the insects.

3 Methodologies Adopted:

3.1. Dataset description: The dataset contains approximately **4500 images of 50 species of butterflies** around the world. The images were extracted from Kaggle and are multi-classifications of butterflies and are resized to a uniform resolution of 128x128 pixels which is different from their original resolutions.

3.2. Data Reinforcement The data reinforcement has been applied via transforms.compose function in the transform object used by the JPEGImageDataset. The original size has been reduced to **128x128 pixels** and normalization has been done using **mean std=[0.230, 0.226, 0.224]**): and **standard deviation std=[0.230, 0.226, 0.224]**).

It is one of the important ways to enhance the size and diversity of training datasets with the help of transformation like rotation, flipping, scaling and color adjustment. With this approach, it helps to model the data very preciously because of its classification mechanism.

3.3 Batch Normalisation:

The formula was developed by Sergey Ioffe in 2015 to stabilize layer input during the process with the method to accelerate deep neural network with the below mentioned formula:

The Batch Normalization formula is:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot \gamma + \beta$$

Where:

- x is the input to the BatchNorm layer
- μ is the mean of the mini-batch
- σ^2 is the variance of the mini-batch
- γ is the scale parameter
- β is the shift parameter
- ϵ is a small constant

It is basically used to normalize each feature individually and calculate the mean and variance per mini batch, allowing the network to learn an identify function and helps to reduce internal covariate shift.

4 Convolutional Neural Network Model CNN

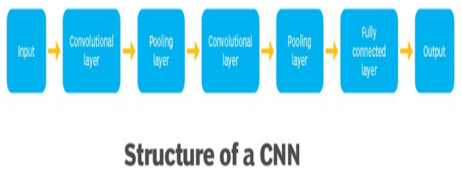


Figure 1: CNN Structure Diagram

Convolutional Neural Networks (CNNs) are a powerful architecture in machine learning, particularly well-suited for analyzing two-dimensional data. They are highly effective at capturing hierarchical patterns and spatial dependencies within images, making them ideal for applications where the structure of data in space is important. CNNs have gained widespread use in various fields, including image classification, object detection, facial recognition, and medical image analysis.

The efficiency of a CNN on picture categorization tasks can be evaluated using a variety of criteria.

Among the most popular metrics are:

Accuracy

Accuracy is the proportion of correctly classified images, defined as:

$$\text{Accuracy} = \frac{\text{Correctly Predicted Images}}{\text{Total Images}} \times 100\%$$

Precision

Precision measures the exactness of predictions for a specific class:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Recall

Recall measures the completeness of predictions for a specific class:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

F1 Score

The F1 Score is the harmonic mean of precision and recall:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where:

$$\begin{aligned} \text{TP} &= \text{True Positive} \\ \text{TN} &= \text{True Negative} \\ \text{FP} &= \text{False Positive} \\ \text{FN} &= \text{False Negative} \end{aligned}$$

2D Convolution Operation

The 2D convolution operation is defined mathematically as:

$$(I * K)_{(i,j)} = \sum_{m} \sum_{n} I(i+m, j+n) \cdot K(m,n)$$

Where:

- I is the input image
- K is the filter/kernel
- $*$ represents the convolution operation
- (i, j) represents the current position in the output

Output Volume Size Formula

For an input of size $W \times W \times D$:

- F = filter spatial size
- S = stride
- P = padding

The output volume size is calculated as:

$$W_{\text{out}} = \frac{W - F + 2P}{S} + 1$$

Practical Convolution Example

Consider a 3×3 filter on a 5×5 input image:

- Slide the filter across the image
- Perform element-wise multiplication
- Sum the results to produce output values

4 Convolutional Neural Network Architecture

4.1 Convolutional Layers

Convolutional operations are applied to input images using filters (kernels) to detect edges, textures, and more complex patterns.

4.2 Pooling Layers

Pooling layers reduce the dimensions of feature maps by down-sampling, decreasing computational complexity.

4.3 Activation Functions

Activation functions are crucial in neural networks because they enable the model to learn complex patterns and relationships in data through non-linearity. Rectified Linear Unit (ReLU) is commonly used to capture intricate patterns.

4.4 Fully Connected Layers

Fully connected layers analyze the output of previous layers, connecting each neuron to all neurons in the next layer to integrate all features and make the final classification decision.

4.5 Transfer Learning

Transfer learning involves using pre-trained models for new tasks within a similar domain, such as ResNet or VGG.

5 Adopted Architecture

5.1 ResNet-18: Revolutionary Deep Learning Architecture

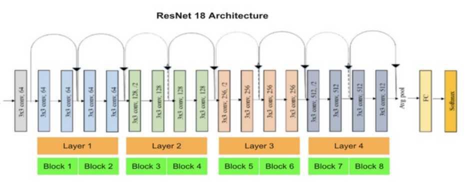


Figure 2: Resnet-18

Kaiming He and his team introduced the ResNet (Residual Network) concept to skip the convolutional layer blocks and ability to learn residual functions to train the extremely deep neural networks. Here, the residual blocks are referred to as "bottleneck" and implement the residual connections between CNN layers bypassing layers through ReLU activation layer and maintain the gradient flow i.e. the weights learned from earlier layers won't vanish during backpropagation and finally helps to identify functions more easily. Resnet-18 has 18 layers and an initial lightweight design with 11.7 million parameters.

5.2 EfficientNet

It is basically designed for high accuracy and efficiency using few parameters and less computational

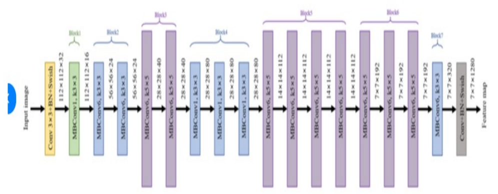


Figure 3: Effnet$_{d}iagram$

resources with respect to other models. It uses a scaling technique which always makes sure that the network depth, width and resolutions remain balanced. It is designed by Google family and uses inverted residual blocks with squeeze-and-excitation (SE) layers for better feature extraction. It is basically used where the challenges with the resource constraints.

- Learning rate (η): 0.20
- Maximum iterations: 200
- Stratify: Ensures balanced class distribution
- Random state: 3 for reproducibility

5.3 Vision Transfer

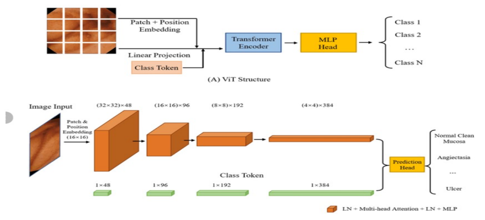


Figure 4: Vit-10-Diagram.jpg

ViT-10 processes images in a different way than the conventional CNNs by using transformers to model relationship between patches. It helps realize the effective image classification by using global context across patches. Here, each patch has been acted as a "token" and are flattened into vectors then linearly projected to create a sequence of embeddings for each patch. ViT-10 consists of 10 transformer encoder layers with two components.

- Multi-Head Self-Attention
- Feed-Forward Network (FFN)

It is considerably useful when there is limited data but accelerates training and enhances accuracy is required without building a model in a large dataset.

5.4 Optimizer

It is an algorithm or process used to reduce the losses after making some changes in the weights and learning rate of neural network to provide the best possible results. It helps to minimize the loss function with an adjustment of model's parameters. The use of algorithms precisely depends on the dataset size and the complexity of models. Some of the most commonly available optimizers are mentioned below.

- Stochastic Gradient Descent
- Madam
- RMSprop

Each optimizer has its own pros and cons so the choosing of an optimizer purely depends on the requirements of model. So, we must properly understand the model architecture before we make final decisions to implement the optimizer.

Here in this case, AdamW has been used because of better regularization, helps in learning rate of each parameter and balances the speed and accuracy.

5.5 Loss Function: Cross-Entropy Loss

Mathematical Formulation

The multi-class Cross-Entropy Loss is defined as:

$$L(y, \hat{y}) = -\frac{1}{M} \sum_{j=0}^{M} \sum_{i=0}^{N} (y_{ij} \cdot \log(\hat{y}_{ij}))$$

Where:

- $\hat{y}$ represents the predicted probability
- y is the actual probability
- M is the number of mini-batches
- N represents the number of butterfly species

Implementation Details

- **Loss Function:** `nn.CrossEntropyLoss()`
- **Purpose:** Measure the difference between predicted and actual probability distributions
- **Suitability:** Optimal for multi-class classification problems

Key Characteristics

- Penalizes confident incorrect predictions
- Lower loss indicates better model performance
- Guides model learning by minimizing prediction errors

Objective: Minimize the cross-entropy loss to improve classification accuracy across butterfly species.

6. Result of Trained Model

6.1 Requirements Evaluation Metrics

The PyTorch framework has been used to train the datasets and normalization process with channel-wise normalization has been used. Image resizing to defined pixel instead of original to ensure that the input dimension remains same. 80-20 rule has been implemented for validating and testing purpose with the batch size of 32 and epoch of 20 training. AdamW optimizer has been used with learning rate of $1e^{-4}$ and weight decay of 0.01 along with cross-entropy loss which is suitable for multi-class classification.

6.2 Computational Infrastructure

- **Hardware:** GPU-supported system
- **RAM:** 64GB
- **Purpose:** Optimize dataset training and model performance

Objective: Comprehensive performance evaluation using multi-dimensional metrics to assess model classification effectiveness.

7. Result Explanation

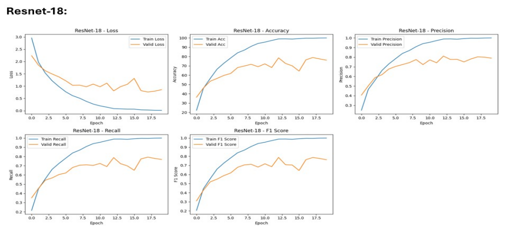


Figure 5: Resnet-18-result.jpg

ResNet-18 demonstrated exceptional performance in butterfly species classification, achieving 99.84% accuracy with a low 0.0153 loss. Its efficient feature extraction and residual connections enabled superior model fitting, yielding remarkable precision, recall, and F1 scores of 0.9994 across the dataset.

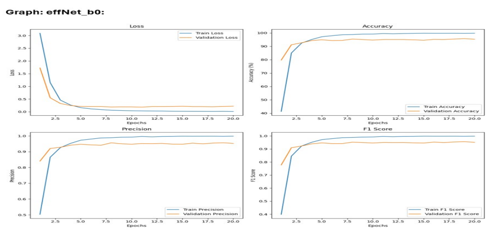


Figure 6: effNet-10.jpg

EfficientNet-B0 demonstrated robust performance with 95.3% accuracy, 0.2198 loss, 0.9520 precision, 0.9532 recall, and 0.9505 F1 score, consistently maintaining high evaluation metrics across different assessments.

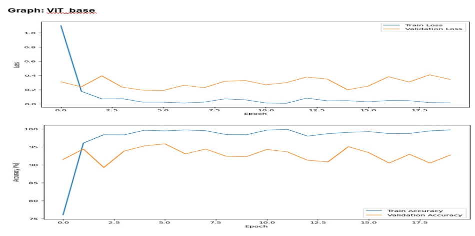
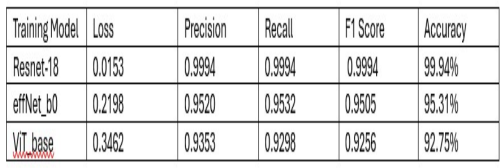


Figure 7: ViT-10.jpg

In the case of ViT Base, the score achieved the lowest in comparison of the above two whereas it was able to maintain at 92.75% with a loss value of 0.3462 followed by precision 0.9353, recall 0.9298 and F1 score 0.9256 respectively.

8. Conclusion Recommendations



Training Model	Loss	Precision	Recall	F1 Score	Accuracy
Resnet-18	0.0153	0.9994	0.9994	0.9994	99.94%
effNet_b0	0.2198	0.9520	0.9532	0.9505	95.31%
ViT_base	0.3462	0.9353	0.9298	0.9256	92.75%

Figure 8: Output-Result.jpg

ResNet-18 emerges as the top performer in butterfly species classification, surpassing alternative models across evaluation metrics. Recommended strategies include dataset expansion, ensemble method exploration, and advanced data augmentation techniques to enhance model robustness.

Leveraging GPU resources can optimize processing efficiency, ultimately improving butterfly species identification accuracy through comprehensive training and testing approaches.

References

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2009.

[2] IBM Developer, *Machine Learning and Deep Learning Architectures*, available at: https://developer.ibm.com/articles/cc-machine-learning-deep-learning-architectures/, accessed: November 9, 2024.

[3] Fast.ai, *Fast.ai Course*, available at: https://course.fast.ai/, accessed: November 9, 2024.

[4] Arbaz Khan, *Image Classification Using CNN (94% Accuracy)*, available at: https://www.kaggle.com/code/arbazkhan971/image-classification-using-cnn-94-accuracy, accessed: November 9, 2024.

[5] ScienceDirect, *Research Article on Convolutional Neural Networks*, available at: https://www.sciencedirect.com/science/article/abs/pii/S1226861520307597, accessed: November 9, 2024.

[6] GeeksforGeeks, *Convolutional Neural Network (CNN) in Machine Learning*, available at: https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/, accessed: November 9, 2024.

[7] TechTarget, *Convolutional Neural Network (CNN)*, available at: https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network, accessed: November 9, 2024.

[8] Viso.ai, *Vision Transformer (ViT): A Detailed Guide*, available at: https://viso.ai/deep-learning/vision-transformer-vit/, accessed: November 9, 2024.

[9] Pankaj Gupta, *Butterfly Classification Dataset*, available at: https://www.kaggle.com/datasets/pnkjgpt/butterfly-classification-dataset, accessed: November 9, 2024.

[10] Keras, *EfficientNet API Documentation*, available at: https://keras.io/api/applications/efficientnet/, accessed: November 9, 2024.

[11] MathWorks, *ResNet-18 Documentation*, available at: https://www.mathworks.com/help/deeplearning/ref/resnet18.html, accessed: November 9, 2024.

[12] Hugging Face, *Vision Transformer (ViT) Model Documentation*, available at: https://huggingface.co/docs/transformers/en/model$_doc/vit, accessed: November 9, 2024.$

Github - Link : https://github.com/Saurabkharel1/CNN-Assignment

Please be noted that the code has been prepared with the help of AI as well.