Deep Learning Based Fish Species Classification

Sainath Veerla

Department of Applied Data Science, San José State University

DATA 270 Data Analytics Processes

Dr. Eduardo Chan

December 8, 2023

Modeling

The studies conducted by taxonomists usually revolve around finding the distributions as well as resources of the numerous aquatic varieties. The project aims to develop a reliable classification model that will assist in those endeavors. To achieve this, two essential datasets: Fish4Knowledge collected close to Taiwan's coast and DeepFish collected close to the Austral coast. Our research will use these two open-source datasets available on the Internet.

This vast dataset is used for training our model of predicting various fish species, with an aim that it works effectively in reality. Our data set is constructed using videos from these places and every frame of the video serves as an individual image into the data set. To reduce the complexity of modeling, the frame exclusion process started with frames that had no fishes.

Median and bilateral filters are applied to enhance further this dataset by minimizing the noise and improving the pixel distribution. To achieve standardization, data in different formats were changed into uniform JPG format and were resized into 256 x 256 dimensions. Further modifications were also made to tackle the problem of underlit images and prepared the dataset perfectly and adequately well for model training.

Lastly, SVD was used with 10 components for capturing 95% of the variance. The choice optimizes efficiency and it is suitable for preset models to improve the class model's performance. SVD also adapts well with pre-trained models and provides better results by Valipour et al. (2022).

Model Proposals

Deep neural networks used in image classification must be capable of distinguishing small, fine-grain features specific to each image; recognising differences between them. For this

study vision transformers have been picked because they are more efficient than classifying low resolution, low image dataset and resilient in classification.

Vision Transformers

Vision Transformer is a modified form of Transformer which is generally used in NLP that was adapted to solve computer vision problems. These act through transforming the incoming images into patches and thereafter feature vectors thus feeding the traditional Transformer encoder – decoder model. The model has a self-attention mechanism that directs the focus within the Transformer on particular components of the above mentioned input sequences. The Vision Transformer model achieves fine details that are very critical in making accurate labeling by shifting attention all over the photos.

Sparse Transformers by Child et al. (2019) is a new technique that promises efficient generation of long sequences. The transformers are versatile in dealing with diverse data sets such as audio, text and image data. The mechanism utilizes attentional heads that are factored in a multi-head attention scheme in order to pay focused attention to input sequences. It optimizes on memory usage by recomputing the attention and feed-forward blocks in the backward pass. They combined the softmax operation, which is an integral part of the single-headed attention mechanism, and a single kernel. The innovation keeps pace with the nonlinearity so as to enhance computational efficiency. This led to the testing of the effectiveness of sparse transformers on various datasets. The CIRCAR -10 dataset was used for image data, resulting in impressive outcomes at 120 epochs and a learning rate of 0.00035.o The sequence generation was done at the rate of 2.8 bits per second in the model. Also for text data, EnWik8 dataset was utilized with 80 epochs, giving the generation speed of one point one three bits per byte.

Mao et al. (2021) suggested patch-aware scaling for robust ViT models by decreasing meaningless patches' attentional weights. It also uses patch-wise data augmentation of random noise, flipping, and cropping transformation for each patch. This two-stage approach strengthens the models' ability to handle noisy data. In order to test RVT, it was compared with ImageNet-1K reaching a competitive top-1 accuracy of 79.2%. The robustness assessment was completed using the ImageNet-C which are altered forms of the original ImageNet pictures in order to test the Resilience Vector Testing (RVT) algorithm. At ten noise patches, the RVT showed an accuracy of 78.6%, proving that it remains accurate under harsh conditions. Patch-aware scaling has an added advantage in that it focuses on discriminative local patches for both clean as well as corrupted images, thus providing better accuracy. However, the patch-level augmentations cause locality and translation invariance during this period. Instead of relying solely on intense augmentation, RVTs target robustness directly in the architecture, as it is an attempt to fill in the gap between vision and robot perception in order to catch up with humans. These innovations offer promising ways, deploying ViT towards robustness of real world image artifacts and application of fish recognition

Cheng et al. (2023) used the Vision Transformer model for femur classification from radiographs. Patients with proximal femur fracture between January 2013 and December 2020 data was collected by performing a study at a Level-I Trauma center. The initial 2645 images were manually labeled with the help of a senior trauma surgeon. Pre-processing was performed where images with less lighting and femur partially hidden were removed. Then the images were cropped to the femur region and resized to 224X224. For further effective detection and correction between left and right femurs YOLOv3 was used. The data has been further labeled

into different fracture types and modeled using ViT. The model was modified by incorporating GELU activations with a learning rate of 1-e^{-4}. The ViT model training occurred for 4 epochs on an NVIDIA Quadro RTX 6000 GPU with batches of 16 images.. The model was able to classify test data 83% with a precision of 0.77.

Yu et al. (2023) introduced mixing attentive to vision transformers to address ultra fine grained visual categorization. It uses a self supervised model to predict if a token has been attentively substituted. This module is supervised by binary cross entropy loss which enhances feature representation by forcing the model to recognize various tokens. The model is pre-trained on ImageNet21K. Random cropping and random horizontal flipping is applied on the images before training and SGD optimizer is used to optimize the entire architecture. The learning rate for all the datasets was set to 5e-3 except for SoyGlobal wish 2e-2. The model is trained and tested in five different ultra-fine grain image datasets Cotton80, SoyLocal, SoyGene, SoyAging and SoyGlobal and provided an accuracy of 60.42%, 56.17%, 79.94, 76.3%, and 51% respectively. The possible reason for less accuracy is due to fewer samples per category but promisingly challenges ultra grain visual categorization.

Okolo et al. (2022) enhanced ViT by adding a CNN block which is concatenated between each transformer encoder layer such that the model remembers the full image until the end instead of storing the information in tokens at every stage. The CNN block comprises stacked 2D convolution layers and 1D global maximum pooling layer. The model was pre trained on ImageNet dataset and evaluated against chest X-ray datasets. The images were processed by performing label smoothing and data augmentation on a training dataset. For training the model was assigned with 0.0001 learning rate such that it can adapt the weight of pre trained data with

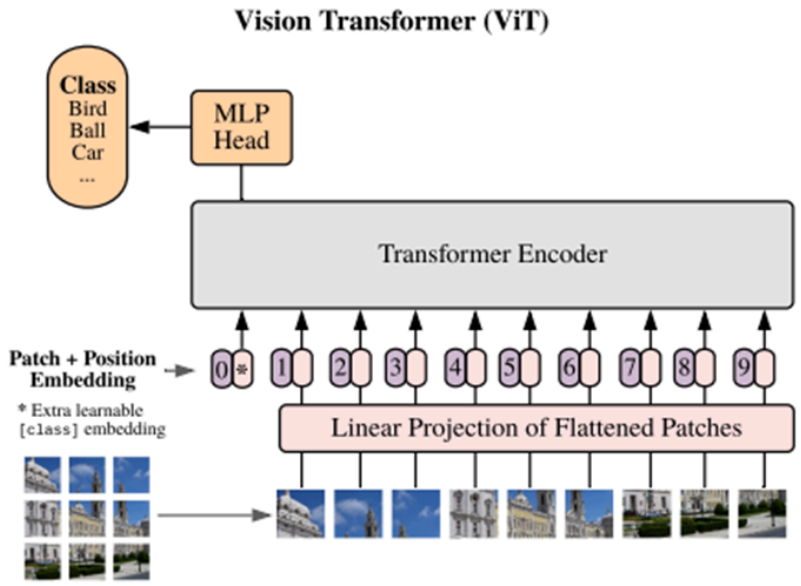
new data. The proposed model with patch size 32 provided promising results with an accuracy of 98.08% and 96.99 precision for patch size 16 when compared with the standard ViT model. The experiment revealed that the proposed model is not constrained to neither specific device nor specific settings making it powerful and easily accessible for assisting in diagnosis.

Algorithm and Equation

Vision Transformers (ViT) initially split the input image of resolution (H, W) into N flattened patches of size (P x P pixels), where N = HW/P^2. Each patch is then linearly projected to a D-dimensional vector called a token. Positional embeddings are added to the patches to retain spatial relationships between patches . Together, the sequence of N patch tokens and positional codes create the input representation for the transformer model. The figure 1 provided by Dosovitskiy et al. (2021) shows the entire architecture of the Vision Transformer model.

Figure 1

Architecture of ViT



The embeddings are sent to multi head attention in the form of Query (Q), Keys (K), and Values (V) by linearly projecting them along with token embeddings. Multi-head attention is a combination of multiple self-attention layers where each layer focuses weights at different parts of the image. These images with different weights are called heads which encapsulate different relationships in the image.The figure 2 provided by Khan et al.(2022) denote the equation for multi-head attention where W^0, W_i^Q, W_i^K, W_i^V are weights for linear projection.

Figure 2

Equations representing Multi-head attention mechanism

$$MultiHead(Q,K,V) = Concat(head_1,\ldots,head_h)\,W^0$$
$$where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Note. Equation denotes the representation of multi head attention mechanism

A single self-attention layer allows the model to incorporate global context by enabling each token to attend to all other tokens Vaswani et al. (2017). Attention scores are computed between queries and keys via a scaled dot product in this layer as shown in figure 3 was provided from Vaswani et al. (2017) where d_k is dimension of key vectors:

Figure 3

Self Attention mechanism formulae

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

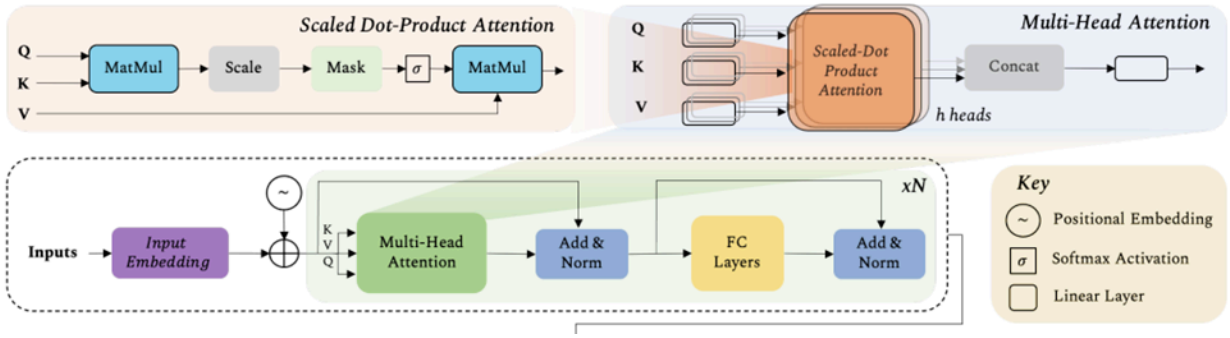
Note. Equation denotes the representation of self attention mechanism

The division by $\sqrt{d_k}$ allows smooth gradient flow during training. The softmax normalization then turns the scores into attention weights over the values. The model can

effectively understand details and context to inform image classification. Multiple heads allow exponentially more relationships to be modeled. Figure 4 provided by Kha n et al.(2022).shows the full architecture of the transformer encoder, multi head attention and self attention mechanism.

Figure 4

Transformer Encoder Architecture



Note. Detailed view layers in transformer encoder

The output of the Multi head attention layer is passed on to a Feed Forward network which comprises a simple 2-layer network with ReLU activation. This layer captures complex patterns and relationships and enriches patch-level representations individually Then layer normalization is applied and labels are predicted based on values. The figure 5 provided by Vaswani et al. (2017) denotes the equation for Feed Forward Network where W_1 *and* W_2 are linear transformations with dimensions, b_1 *and* b_2 are bias terms.

Figure 5

Equation of Multi layer Perceptron

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Note. Equation denotes the representation of feed forward network

The token embedding stores all the information that is related to the image. It gathers all details the model has extracted and stores it. The final layer of MLP uses this token embedding and creates a probability for all the classes. The classes with highest probability is assigned to the image.

Optimization

Vision transformers consist of parameters that can be optimized such that the model can be tuned so as to meet the requirement of the project. These parameters include patch length, learning rate, loss function and optimizer used.

Using ViT as a baseline, Ibrahimovic (2023) has carried out experiments in which different hyperparameters were tuned to see how they affect some selected metrics or dimensions. The CIFAR-100 dataset comprises 60000 color 32×32 images spanning over one hundred classes; they formed the basis of the experiments. The three main factors examined were the layers of the transformer, and the size of the image patches being considered as inputs. Increasing the number of transformer layers resulted in marked increase in accuracy for a given patch size at the expense of very long training time because of increased complexity. However, the improvement reduced after the parameter's threshold saturation. It was due to more granular representation that reducing the patch size provided benefits in terms of accuracy and convergence rate; however, those improvements ceased to improve at a resolution smaller than 8×8. Generally, in this instance the study showed that tweaking core ViT parameter settings increases accuracy, time, and customisation of a model suitable for image classification for an application context. These insights will help create optimized ViT architecture by providing ways for balancing parameter tuning and practical deployment of a model.

Based on literature review attention has to be given to hyperparameters to gauge the importance of model performance. The goal is to identify configurations that yield optimal results while minimizing training time-a critical factor in deploying efficient machine learning models. This iterative process enables a comprehensive understanding of how the model responds to different hyperparameter settings and facilitates the discovery of a well-balanced configuration that aligns with the desired metrics and time constraints.

Model Supports

All the collected images from open-access websites have been organized into folders and stored in Google Buckets for easy access by team members. These images have been pre-processed, transformed, and evaluated in the Vertex AI environment such that additional GPU can be added if it is required for the model.

Hardware and Software Configurations

The hardware requirements include a system with 8GB RAM, 222 GB hard disk with intel i5 processor. As the project makes use of deep learning models the system should have a GPU that helps in faster computation therefore Intel HD Graphics 520 has been used. Table 1 gives details on different hardware requirements required for the project.

Table 1

Hardware Configuration

Hardware	Configuration
CPU	Intel(R) Core(TM) i5-6200U
GPU	Intel HD Graphics 520
Frequency	2.3GHz
RAM size	8GB
Hard disk	222 GB

The research and development of the model utilized the Windows Operating System along with the Colab Enterprise IDE. Colab is google powered IDE to perform various operations and execute large models seamlessly on cloud giving it an easy access. Table 2 gives an overview of softwares used.

Table 2

Software Requirements

Software	Configuration
Operating System	Windows 10 Enterprise
IDE	Colab 3.10

Tools and Libraries

To meet the image classification project requirements, we leveraged various Python libraries for the data pipeline and modeling. The PIL packages enabled loading images and transforming them to meet model input specifications in terms of size, channels, pixel values, etc. We relied on the Transformers and PyTorch ecosystems for implementing the deep-learning Vision Transformer (ViT) model. Pre-trained ViT architectures were fine-tuned on dataset composed of images spanning different fish species. The NumPy library supplemented array manipulations coupled with conversions between pixel representations and tensors. After training and testing, Matplotlib plots were used to visualize model metrics such as accuracy and loss values across the epochs. These practices helped seamlessly move between stages of processing and inferences regarding classifying visually similar fish images. Table 3 shows the complete details of libraries and their use case in the project.

Table 3

Tools and Libraries used for Research

	Library	Method	Usage
Transformers	ViTForImageClassification		For image classification tasks
	TrainingArguments		Define arguments such epochs, learning rate parameters for the model
	Trainer		Define train, test and validation datasets for learning
	ViTImageProcessor	from_pretrained	Loads a pre-trained Vision Transformer model
Datasets	Load_dataset	with_transform	Loads processed datasets
	Load_metric	Accuracy, recall, F1 score, Precision	Metrics for evaluation
Tensorflow	Keras, reshape, images	Layers	Keras layers and image-related operations
Torch	Nn, stack, tensor		Neural network module, tensor values
Numpy	Argmax, array, shape, concatenate		Array operations and manipulation
PIL	ImageFilter	MedianFilter	Preprocessing on images
Matplotlib	pyplot	Bar, heatmap	Create visualizations

Model Architecture and Data Flow

Upon preprocessing the dataset, the images are saved into another separate folder that will be used for further modeling. The flow in figure 6 represents a sophisticated structure of data flow for image classification which was used in this research with an emphasis on fish species' identification. To begin with, the images are divided into small patches consisting of 16x16 pixels. These patches are shown visually with positional embeddings maintaining important spatial relations between patches for visualizing their understanding. Token embedding are also created along with positional embedding to store the image information.

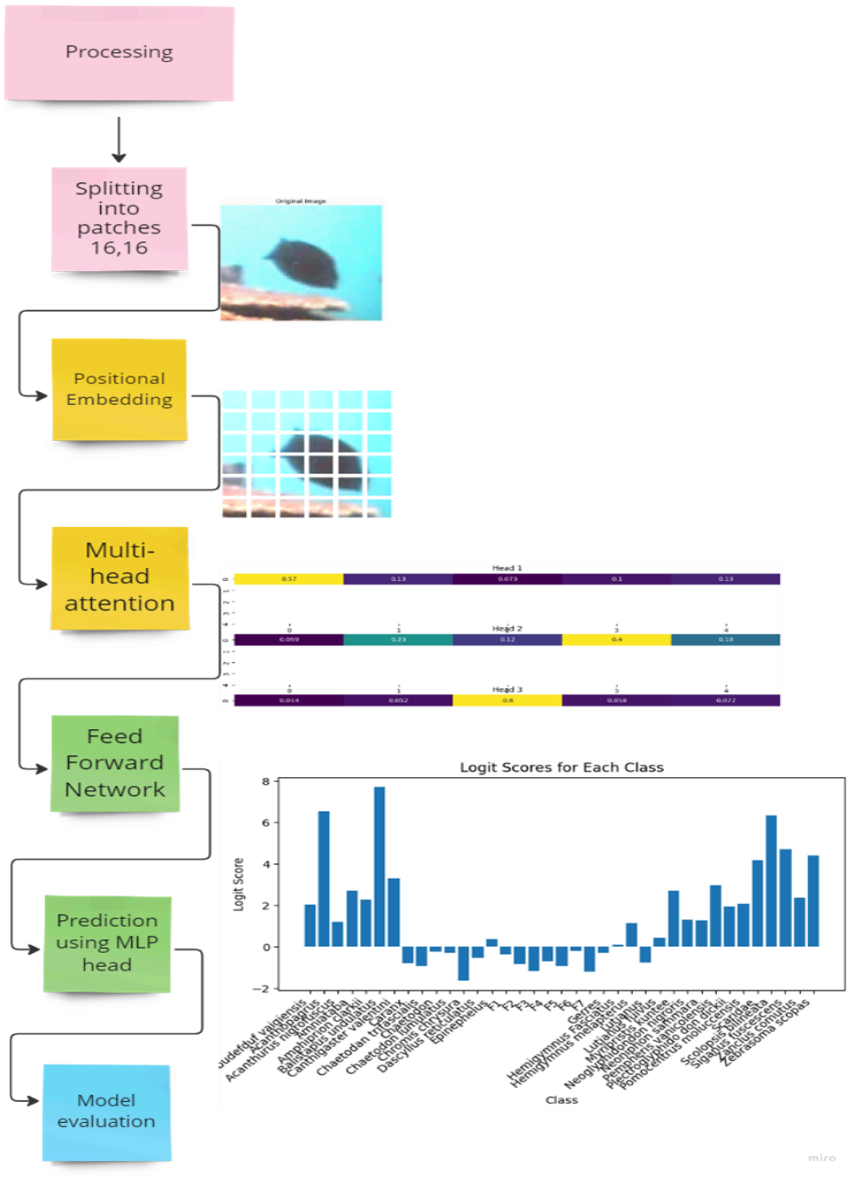
These patches are taken as input by the multi head attention where weights are applied on different parts of the image to understand the image and find in depth details. Multi-head attention is at the core of this network. Multi head attention is a combination of multiple self-attention mechanisms where attention is applied as a serial order Using the self-attention mechanism, these patches are redistributed across multiple heads. Attention maps arising from these images demonstrate differences in focused attention which is manifested in vivid yellows for particular areas of interest. Thus minute details that are critical for correct identification, and among various fishes species are captured by varying attention on patches.

As soon as the process of self-attention is complete, the pictures are fed forward into the network. This is when the model uses a MLP to predict and merge the features extracted by the Self-attention mechanism. This is where non-linearity takes place so that it can discover complex shapes and spatial relationships between visual elements. Through a synergetic approach where self-attention is coupled with the feedforward network operations, the models learn strong and dissimilar representations. The MLP layer has the information of the image as a whole not only as patches thereby allowing it to understand all the information collected in the previous step.

After collecting all the information from the token embedding and MLP layer, the model makes a prediction on the class of the input image at the final stage. This last prediction step is illustrated through the accompanying graph provided in the figure 6 and shows the output probabilities of the model for several fish species.

Figure 6

Data flow of Model



Note. Detailed overview of data flow for the research

Lastly, the model is applied on the testing data set and evaluated based on accuracy, recall, precision, confusion matrix to assess the efficiency of the model. By passing through every phase, the acquired and computed information enables the model to forecast correctly as it reflects the capability of vision transformers in classifying images with different kinds of fishes.

Model Comparison and Justification

ViTs use pure transformers with self-attention while Inception-ResNet has both CNN and residual blocks Both handle image data but ViTs need more data to train effectively. Inception-ResNet generalizes better with transfer learning. Residual connections make Inception-ResNet more robust. ViTs have higher compute needs than Inception-ResNets. Self-attention weights consume more memory for ViTs. Inception-ResNet trains faster than vision transformers ViTs capture global contexts, Inception-ResNet extracts multi-level features. ViTs struggle with large images, Inception-ResNet worse at complex patterns. More in depth analysis between ViT and InceptionResNet has been listed in Table 4. Although the computational time for training is very high when compared with InceptionResNet, ViT is able to find patterns and hidden details in images even with less epoch. ViT produced an accuracy of 92% for 8 epochs while InceptionResNet had an accuracy of 86% despite the huge difference in training time of the model.

Table 4

Comparison of ViT and InceptionResNet

Characteristic	Vision Transformers (ViTs)	InceptionResNet
Basic Architecture	Transformer-based architecture with self-attention mechanism	Hybrid architecture combining Inception blocks

		and ResNet residual connections
Data Types Processed	Primarily designed for image data	Image data
Performance on Small Data	Generally requires large datasets for optimal performance	Efficient on smaller datasets, especially with transfer learning
Overfitting/Underfitting	Susceptible to overfitting, especially with limited data	Robust against overfitting, especially with residual connections
Complexity	Moderate preprocessing, requires image patching and positional encoding Longer training times compared to some traditional CNNs	Moderate preprocessing, multi-level feature extraction Relatively faster training times than ViTs
Space Complexity	Requires more memory due to the self-attention mechanism	Moderate memory requirements
Computational Complexity	Computationally intensive, benefits from GPU acceleration	Efficient GPU utilization, faster inference
Strengths	Effective at capturing global dependencies in images Transferable across diverse tasks	Excellent feature extraction with multiple pathways Strong performance on various image tasks
Limitations	Resource-intensive, especially for large images Interpretability challenges	May not capture long-range dependencies well

	Not as effective for certain complex image patterns

Model Evaluation Methods

Once the model is trained it is necessary to evaluate its performance using various metrics. The project tries to identify the fishes based on image making it an image classification model therefore the model should be tested with metrics that focus on accurate and false predictions. For evaluation accuracy, precision, recall, F1-score and confusion matrix have been used to determine the strength of the model.

Accuracy

Accuracy refers to the ratio of correct predictions made to the total number of predictions. It gives an intuitive sense of how many test samples were classified correctly. However, it can be misleading for imbalanced classes. The formula is seen in (1).

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

Recall

Recall calculates the fraction of all relevant samples that were identified correctly. Easy to interpret independently and identify bias. The formula is seen in (2).

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

Precision

Precision calculates the ratio of positive identifications that were actually correct. For imbalanced classes, understanding precision for each class provides insight beyond just accuracy. The formula is seen in (3).

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

F1-score

F1 Score is the harmonic mean of precision and recall to provide balance between the two. Provides a singular consolidated metric for model performance. The formula is seen in (4).

$$F1 - score = \frac{2(Recall * Precision)}{Recall + Precision} \quad (4)$$

Confusion Matrix

A confusion matrix provides a breakdown of predictions into a table with actual classes as rows and predicted classes as columns. Key metrics like true positives and false negatives can be directly read off the matrix.

The matrix layout reveals insights about errors, that is if there is a chance they skew towards certain classes. The magnitude of values in the diagonal versus off-diagonal elements indicates prediction accuracy.

Model Validation and Evaluation

There are different metrics upon which a classification model can be tested such as ROC which gives a graph of true positive rate against false positive rate and Cross validation. ROC is better for binary classification but the combined dataset has 37 classes. Also there are 37 images in validation and test dataset in each class. This takes a lot of computational time for cross validation to get executed. Therefore accuracy, recall, precision and F1 score has been chosen. The metrics of different models have been briefed in table 5.

The model before hypertuning had an accuracy of .92, recall 0.92, precision 0.92 and F1-score .91. The model was successfully able to find the differences between images and classes with 8 epochs but trained for 3 hours although the epochs were less.

Based on the literature review few parameters were hypertuned such as image patch was changed to 32 allowing to cover more area of images and lesser patches than the standard model. Optimizer has also been modified to Adamw such that weights can be modified on the basis of stochastic gradient descent. The learning rate has also been improved from 2e-4 to 1e-3 but the hypertuned model provided an accuracy of .89, precision 0.87, recall 0.88 and F1-score 0.87. For these changes the model trained for around 5 hours. Figure 7 and 8 represent the confusion matrix before and after tuning the model. The standard ViT model has a high intensity of blue across the diagonal of the matrix. The confusion matrix also gives details like upon which classes did the model make a wrong classification which is very less when compared with a hypertuned model.

Figure 7

Confusion Matrix of standard ViT model

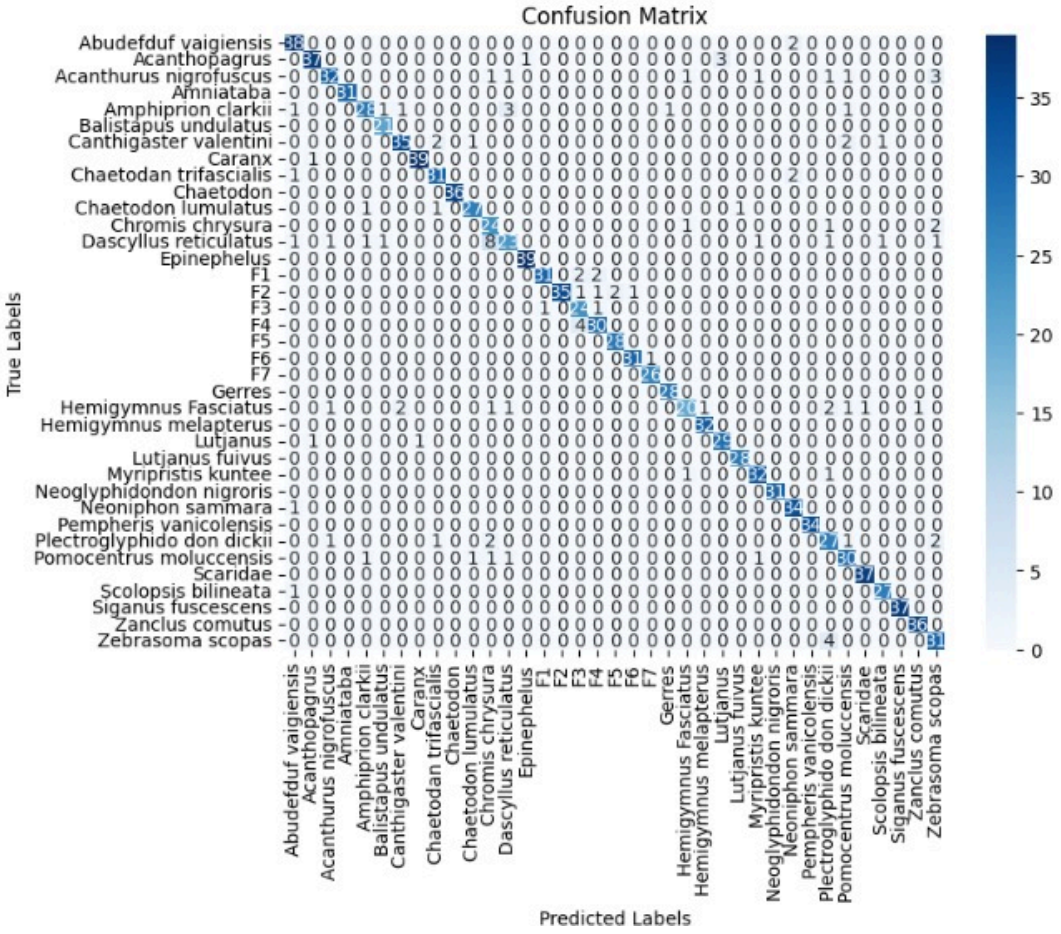


Figure 8

Confusion matrix of hypertuned ViT model

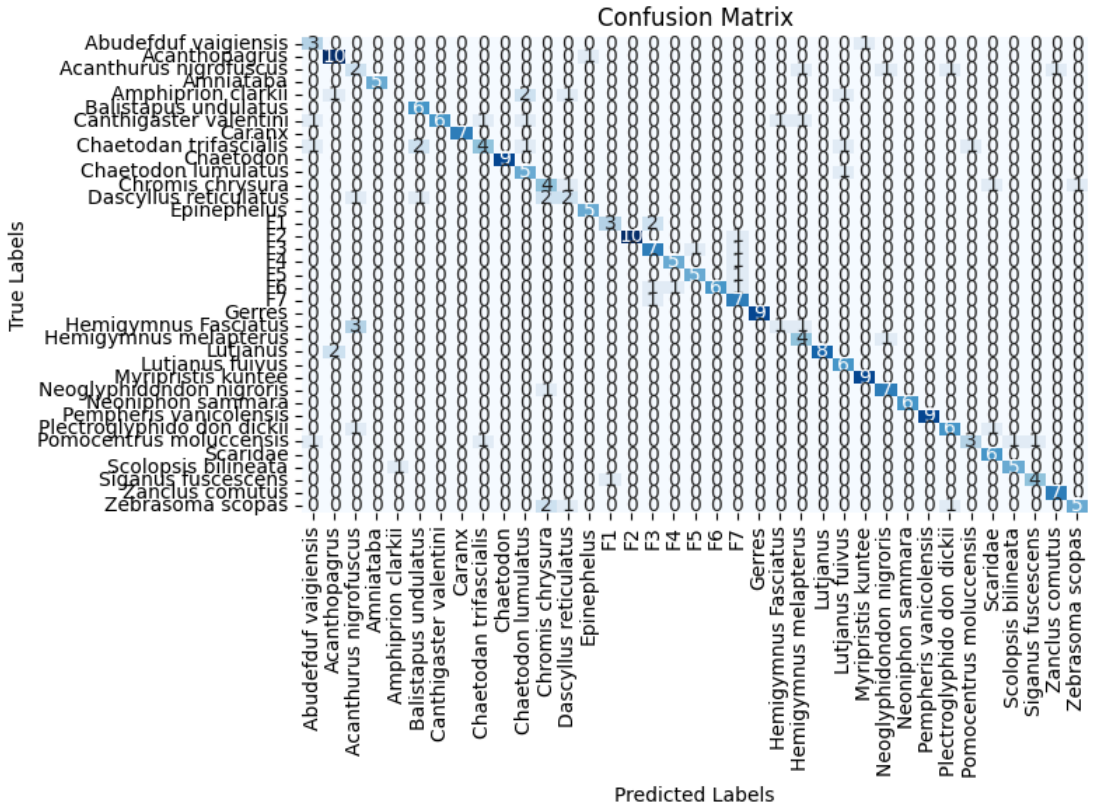


Table 5

Metrics of models

Model	Accuracy	Precision	Recall	F1-score
InceptionResNet	0.93	0.95	0.94	0.94
VisionTransformer	0.92	0.92	0.92	0.91
Hypertuned Vision Transformer	0.89	0.87	0.88	0.87

Conclusion

The model produced good results and was able to find intricate details to distinguish between images with less number of epochs. For a hypertuned model with 32 patch length it took longer computation time to get executed. The model is able to classify 37 species of species

accurately despite the differences in datasets and classes. ViT is a developing model and new modifications are being introduced according to the use case. The model provides promising results for both large datasets and outperformed the hypertuned model. ViT performs well even with low vision images which has been the main reason for choosing the model.

Limitations

ViTs require substantial computational resources, specifically GPU capabilities, for efficient model execution. This can pose challenges, particularly for users with limited access to high-performance computing infrastructure. Additionally, ViTs have longer training times, which can be a bottleneck in scenarios where rapid model deployment is essential. Another noteworthy limitation is that ViTs may struggle when faced with predicting on completely new or unseen fish species that were not part of the initial training data. For optimal performance, the model would need consistent retraining with updated datasets containing new fish species, emphasizing the need for a robust and adaptive training pipeline. Furthermore, the interpretability of the attention mechanisms in ViTs, especially in the context of understanding intricate features in fish images, can be a challenge. Addressing these limitations is crucial for ensuring the practical applicability and accuracy of ViTs in the specific domain of fish species classification.

Future Scope

Eventually, for the future a dynamic pipeline that can handle photos that aren't classed and are kept in a central pool. This approach's versatility makes it possible to name photos later on, which makes iterative modeling easier. In contrast to the existing emphasis on classifying distinct fish species in marine life, the proposed pipeline seeks to create a more adaptable model that can classify a wide range of marine life forms. By using this approach, the modeling phase

may be repeated on a regular basis and the model becomes flexible enough to learn from fresh data. The main goal is to enable the model to identify and categorize a wider variety of aquatic creatures, expanding its applicability beyond particular fish species. This flexible methodology guarantees that the model stays applicable and efficient in the ever-changing field of marine life categorization.

References

Cheng, Y., Zhao, Z., Wang, Z., & Huang, D. (2023). Rethinking vision transformer through human–object interaction detection. *Engineering Applications of Artificial Intelligence*, *122*, 106123. https://doi.org/10.1016/j.engappai.2023.106123

Child, R., Gray, S., Radford, A., & Sutskever, I. (2019). Generating Long Sequences with Sparse Transformers. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1904.10509

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations*. https://openreview.net/pdf?id=YicbFdNTTy

Ibrahimovic, E. (2023). Optimizing Vision Transformer Performance with Customizable Parameters. *2023 46th MIPRO ICT and Electronics Convention (MIPRO)*. https://doi.org/10.23919/mipro57284.2023.10159761

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in Vision: a survey. *ACM Computing Surveys*, *54*(10s), 1–41. https://doi.org/10.1145/3505244

Okolo, G. I., Katsigiannis, S., & Ramzan, N. (2022). IEViT: An enhanced vision transformer architecture for chest X-ray image classification. *Computer Methods and Programs in Biomedicine*, *226*, 107141. https://doi.org/10.1016/j.cmpb.2022.107141

Valipour, M., Rezagholizadeh, M., Kobyzev, I., & Ghodsi, A. (2022). DyLoRA: Parameter Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2210.07558

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All you Need. *arXiv (Cornell University)*, *30*, 5998–6008. https://arxiv.org/pdf/1706.03762v5

Yu, X., Wang, J., Zhao, Y., & Gao, Y. (2023). Mix-ViT: Mixing attentive vision transformer for ultra-fine-grained visual categorization. *Pattern Recognition*, *135*, 109131. https://doi.org/10.1016/j.patcog.2022.109131