Fish Challenge - AI@UNICT2024 - Challenge 1

Asad Aslam aslam1996asad@gmail.com

1. Introduction

The task of classifying different species of fish using deep learning models presents several significant challenges. The primary objective of this challenge is to design and validate a deep learning model capable of accurately classifying fish species. The dataset provided consists of a small labeled training dataset of annotated images, a test dataset, and an auxiliary dataset of unannotated images. The small size of the labeled training dataset and the large volume of unlabeled data introduce unique difficulties and opportunities in model training and validation. One of the main challenges is the limited size of the labeled training dataset. With only few annotated images, training a deep learning model from scratch is impractical due to the high risk of overfitting and poor generalization to unseen data. Additionally, the dataset contains 15 different species, each exhibiting significant inter-species variation and intra-species similarity, making the classification task even more complex. To address these challenges, our proposed solution leverages a combination of transfer learning and semi-supervised learning techniques, specifically pseudo-labeling, to effectively utilize both the small labeled dataset and the large auxiliary dataset of unannotated images. The proposed solution effectively leverages transfer learning and pseudo-labeling to overcome the challenges posed by the small labeled training dataset and the large unlabeled auxiliary dataset. By combining the strengths of pre-trained models with the augmentation benefits of semi-supervised learning, our approach enhances the model's performance and ensures it generalizes well to new data. The key innovations lie in the strategic integration of pre-trained networks, confidence-based pseudo-labeling, and iterative retraining, providing a robust solution for the fish species classification task.

2. Model Description

The proposed solution leverages multiple pre-trained convolutional neural networks (CNNs) to classify fish species, taking advantage of transfer learning to overcome the challenge of limited labeled data. The networks used include AlexNet, ResNet18, ResNet50, VGG16, and DenseNet121. Among these, the best results were obtained using DenseNet121. This section provides a comprehensive description of the architectures, activation functions, regularization techniques, optimization algorithms, and loss functions used.

Pre-trained Networks

AlexNet

- Architecture: AlexNet consists of 5 convolutional layers followed by 3 fully connected layers. The convolutional layers use ReLU activation functions and max-pooling layers.

- Activation Functions: ReLU (Rectified Linear Unit) for all layers.
- Regularization: Dropout layers after the first two fully connected layers to prevent overfitting.

ResNet18 and ResNet50

- Architecture: ResNet (Residual Networks) use skip connections (or shortcuts) to jump over some layers. ResNet18 has 18 layers, and ResNet50 has 50 layers.
- Activation Functions: ReLU for all convolutional layers.
- Regularization: Batch normalization is applied after each convolution layer to stabilize the learning process and improve convergence.

VGG16

- Architecture: VGG16 consists of 13 convolutional layers followed by 3 fully connected layers. The convolutional layers use small receptive fields (3x3) and max-pooling layers.
- Activation Functions: ReLU for all layers.
- Regularization: Dropout layers after the fully connected layers.

DenseNet121

- Architecture: DenseNet (Densely Connected Convolutional Networks) consists of dense blocks where each layer is connected to every other layer in a feed-forward fashion. DenseNet121 has 121 layers.
- Activation Functions: ReLU for all layers.
- Regularization: Batch normalization is applied after each convolution layer.

Among these architectures, DenseNet121 yielded the best performance. A detailed description of the modifications and training process for DenseNet121 are, the DenseNet121 model is initialized with pre-trained weights from training on a large dataset (ImageNet). The final fully connected layer (classifier) is replaced to adapt the model to our specific classification task with 15 fish species. The number of input features to the classifier is set to match the number of output classes (15). Furthermore, Stochastic Gradient Descent (SGD) is used for training the model as optimizer and Cross-Entropy Loss is used, which is suitable for multi-class classification problems.

3. Dataset

- Data Source: Provided by the professor.
- Data Size: 150 annotated samples for training (15 samples per class), about 38k unannotated images and 148 test images
- Preprocessing: Preprocessing is a crucial step in preparing the data for training and testing deep learning models. It ensures that the input data is in a suitable format and enhances the model's ability to generalize from the training data. The preprocessing steps undertaken in this project involve data normalization and augmentation techniques, which are implemented through the torchvision.transforms module. The specific transformations applied to the training and test datasets are detailed below. The preprocessing pipeline is defined using a set of transformations for both the training and test datasets. These transformations are applied to each image to ensure consistency and improve the model's performance. RandomResizedCrop, RandomHorizontalFlip, ToTensor, Normalize, Resize, and CenterCrop has been used to transform the data.

4. Training details and procedure

1. **Evaluation Metrics**
   - Metrics: Accuracy and loss.
2. **Training Phase**
   - Epochs: 100.
   - Batch Size: 32.
   - Validation Strategy: Validation during training. At the end of each epoch, the model's performance is evaluated on the validation subset by calculating the accuracy. This allows for tracking the model's learning progress and identifying issues such as overfitting or underfitting.
   - Optimizer: SGD and its parameters (learning rate = 0.001, momentum - 0.9).
   - Initialization: The weights for the convolutional and other intermediate layers are initialized using pre-trained weights from models trained on the ImageNet dataset. This provides a robust starting point, leveraging the extensive feature extraction capabilities of these pre-trained models. The final fully connected layer, adapted to the target task with 15 classes, is newly initialized.
3. **Pseudo Labeling**
   - After initial training, the models were used to predict labels for the unlabeled dataset.
4. **Re-Training Phase**
   - Epochs: 500.

- Batch Size: 64.
- Optimizer: SGD and its parameters (learning rate = 0.001, momentum - 0.9).

5. Experimental Results

- Training Curves:

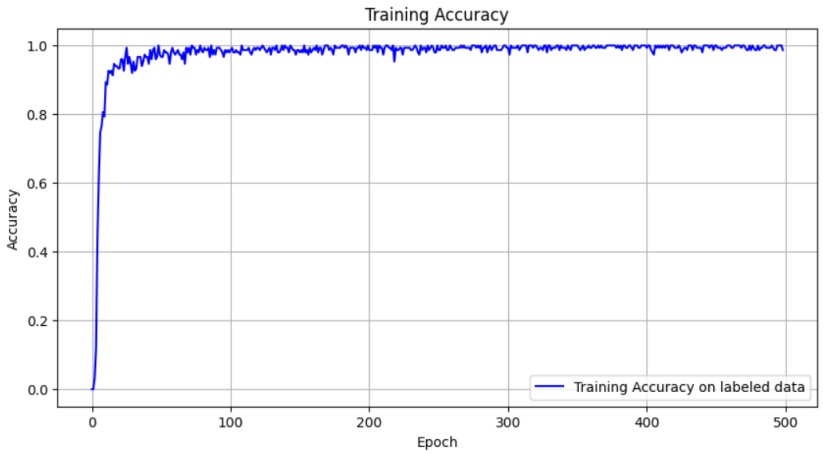

Figure 1: Training Phase Accuracy

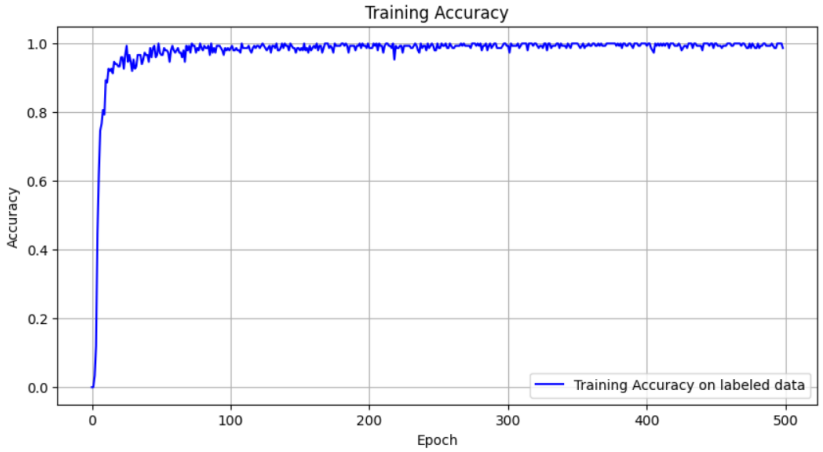

Figure 2: Re-Training Phase Accuracy

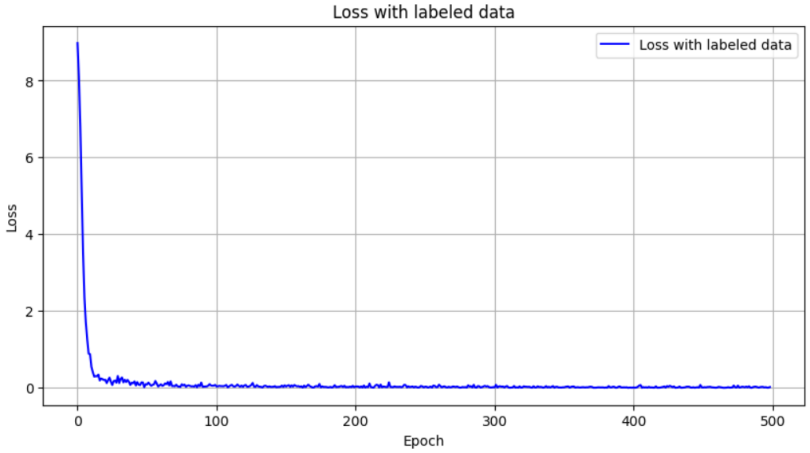

Figure 3: Training Loss

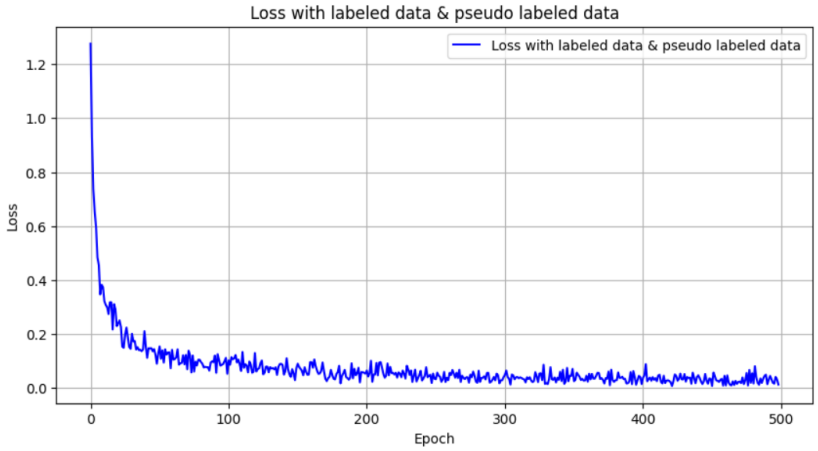

Figure 4: Re-Training Loss

- Performance and Comparison: Report final performance metrics on the test set.

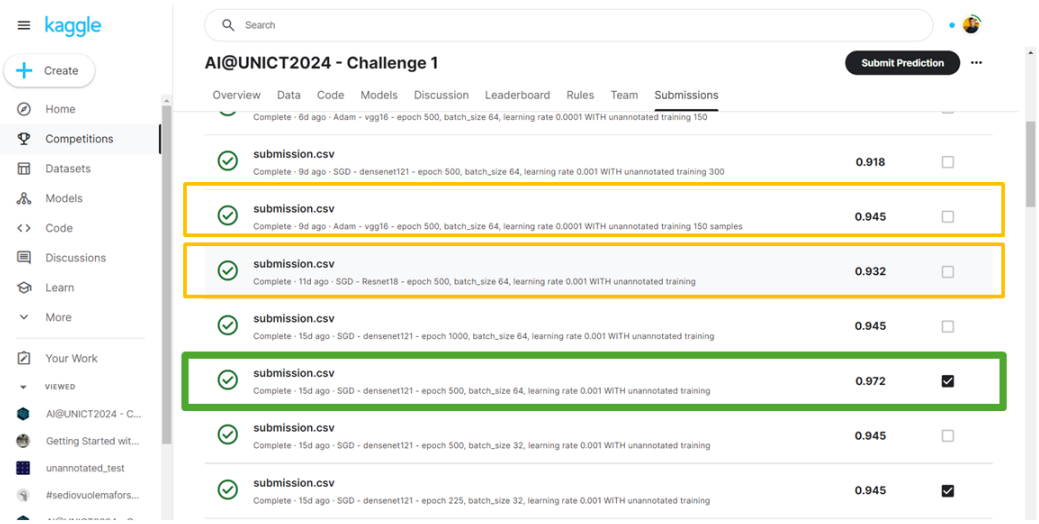

Figure 5: Comparison with different models

	Submission No	Model	batch	lr	epochs	acc	optimizer	unannotated sample size
26	25	resnet18	32	0.0001	75	0.932	Adam	0
27	26	resnet18	64	0.0001	50	0.918	Adam	0
28	27	vgg16	32	0.0001	50	0.959	Adam	0
29	28	vgg16	64	0.0001	50	0.918	Adam	0
30	29	vgg16	32	0.00001	50	0.905	Adam	0
31	30	vgg16	32	0.00001	75	0.891	Adam	0
32	31	vgg16	32	0.0001	50	0.82	Adam	0
33	32	densenet121	32	0.0001	50	0.9	SGD	0
34	33	densenet121	32	0.0001	100	0.9	SGD	0
35	34	densenet121	32	0.001	100	0.972	SGD	0
36	35	densenet121	32	0.001	100	0.918	Adam	0
37	36	densenet121	32	0.0001	100	0.918	Adam	0
38	37	densenet121	32	0.001	100	0.9	SGD	0
39	38	densenet121	32	0.001	150	0.932	SGD	150
40	39	densenet121	32	0.001	225	0.945	SGD	150
41	40	densenet121	32	0.001	500	0.945	SGD	150
42	41	densenet121	64	0.001	500	0.972	SGD	150
43	42	densenet121	64	0.001	1000	0.945	SGD	150
44	43	densenet121	128	0.001	500	0.945	SGD	150
45	44	resnet18	64	0.001	500	0.932	SGD	150
46	45	vgg16	64	0.0001	500	0.945	Adam	150
47	46	densenet121	64	0.001	500	0.918	SGD	300

Figure 6: Detailed comparison with multiple hyper-parameters

6. Conclusion

In this project, we aimed to design and validate a deep learning model for the classification of 15 fish species, leveraging both a small annotated training dataset and a large unannotated dataset. By employing multiple pre-trained models and experimenting with various combinations of hyperparameters, we were able to identify the most effective approach for this task. Among the tested models, DenseNet121 emerged as the most successful, achieving an impressive accuracy of 0.972 on the test data. The key hyperparameters contributing to this performance included a learning rate of 0.01, a batch size of 64, and training over 500 epochs. In comparison, VGG16 and ResNet18 also performed well, with accuracies of 0.945 and 0.932, respectively. These results highlight the robustness and effectiveness of transfer learning techniques, especially when dealing with small annotated datasets. My approach demonstrates that careful selection and fine-tuning of pre-trained models can lead to significant improvements in classification accuracy, even with limited labeled data. This study not only contributes a reliable solution for fish species classification but also underscores the potential of leveraging large unannotated datasets to enhance model performance through methods like pseudo-labeling.