Fish Family Classification with Computer Vision

Callan Hoskins and Michael Basili
Stanford University
{chosk, mbasili}@stanford.edu

Abstract

We use a proprietary dataset scraped ourselves to train a computer vision model to classify the taxonomic family of fish. Applying transfer learning, our best model achieves state-of-the-art accuracy of 73% and top-three accuracy of 92%. This represents a major breakthrough in the area of fish classification, in which many systems rely on domain knowledge and hand-selected features.

1. Introduction

Fish classification (FC) is a task that has been studied extensively by computer scientists and biologists alike. Given an image of a fish, a FC system classifies the fish by family, genus, species, or danger level. Relevant stakeholders in an effective FC system include marine scientists, biologists, fisheries managers, and industrial fishery operators (canneries and other fish factories)[5].

In this paper, we use a proprietary dataset scraped ourselves to train a computer vision model to classify the taxonomic family of fish. Inputting a 256x256x3 image of a fish into a CNN, we output the predicted family of the illustrated fish. Using several industry-standard models, we compare the accuracy of the models on this dataset to see which architecture is most suited for detecting the minute variations required in fish classification. This model has applications in basic science, the global tropical fish trade, and fishery management.

After training three different industry-standard models (ResNet 50, VGGNet-19, and Inception-V3) on two datasets, we found that the ResNet was most effective in classifying fish, with surprisingly high levels of accuracy on the smaller dataset (92% accuracy over the top 3 predictions). Both VGGNet and Inception-V3 were quite effective as well, coming in around 85% accuracy over the top 3 predictions in the shorter dataset, but did not perform quite as well as ResNet 50 over all metrics.

2. Related Work

2.1. Fish Classification (FC)

Our CNN-based fish classification systems are novel in two ways: a) it relies solely on deep learning, and b) it is trained and evaluated on a far greater number of fish families than other systems have been.

Though FC is extensively studied, it is widely recognized as quite a challenging task. As Al Smadi et al. put it, "all types of solutions for automatic FC should consider several elements such as the orientation of fish and arbitrary size; variability of features; changes in the environment; poor image quality; segmentation failures; imaging conditions; and physical shaping"[5]. As opposed to a classifier that discerns between varied species such as cats vs. dogs vs. and birds, a FC system must be able to capitalize on minute differences between otherwise very similar-looking fish to succeed.

Most early FC systems relied heavily on manual geometric feature extraction. In 2000, Zion et al.'s real-time FC system based on geometric features and moment-invariants became one of the first effective FC systems, classifying three species of fish with over 91% accuracy[14]. Lee et al. used geometric descriptors of fish like line segments, polygon approximation, Fourier descriptors, and CF analysis obtained from contour representation to obtain pretty good classification results[9]. However, with most geometric-based systems the same problem arises: automated systems cannot always deduce the geometric attributes from the picture and human intervention is often necessary.

Only in the past few years have researchers begun applying convolutional neural networks to FC in earnest. Though some have applied CNNs to tasks like fish freshness rating[12] and fish *detection*[7], the only use of CNNs in fish classification was in 2020 by Chhabra et al.[6]. Chhabra et al. used a pre-trained VGG16 model to extract features from their fish images, then fed the resulting features through traditional ML methods (kNN, SVM, random forest).

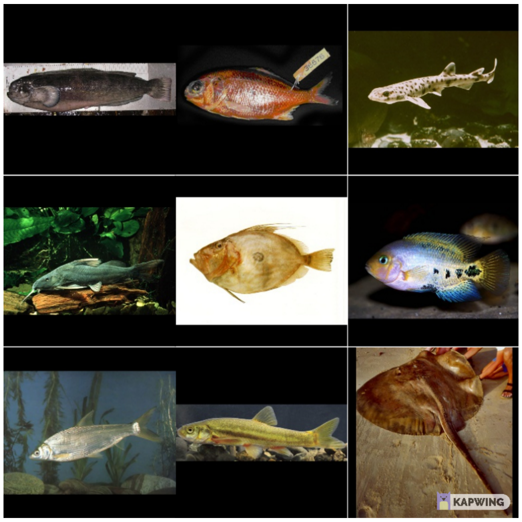


Figure 1. Diverse sample of images from the dataset.

2.2. Generic Classification

In our approach to the FC problem, we make use of several vision classification models that were originally designed for objects besides fish. Transfer learning has proved a powerful tool for almost all deep learning tasks[13]. In this paper, we leverage insights from ResNet 50[8], VGGNet[10], and Inception-V3[11] architectures, all of which were once (and still are close to) state-of-the-art models in image classification.

We were able to use pretrained models from the Torchvision library[4](code example from [3]), sparing us the monetary and computational cost of training them from scratch.

3. Dataset

For this project, we scraped a proprietary dataset that includes 324 taxonomic families of fish, from sharks to seahorses to parrotfish.

Fishbase.com is a website where amateur photographers, aquarists, and others interested in fish can post photos of fish, identified either by the original photographer or by the community. FishBase is the largest website of its kind, boasting more than 61,000 pictures.

Using Selenium and other open-source Python tools, we were able to collect more than 31,000 pictures comprising 324 taxonomic families from FishBase's crowd-sourced collection. Because the photos were not taken in a systematic way, they do not have a cohesive style, dimensions, or orientation. Though imperfect, our method of fixing this is to zero-pad the images to make them square, then resizing them to be 256x256 pixels.

Since the dataset is crowd-sourced, there is not an equal number of images per family; several families have over 1,000 images but the majority of families have less than 100. We anticipate that this could bring difficulties in training our model. For this reason we created a second dataset including only the classes which included more than 400 images. This second dataset with 17 fish families is a more evenly distributed subset of our original dataset which resulted in much higher prediction accuracies, as expected.

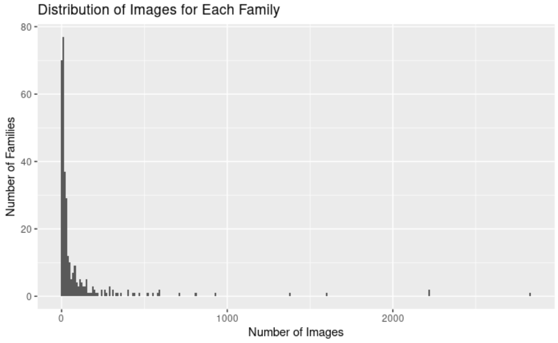


Figure 2. Distribution of Number of Images Per Family

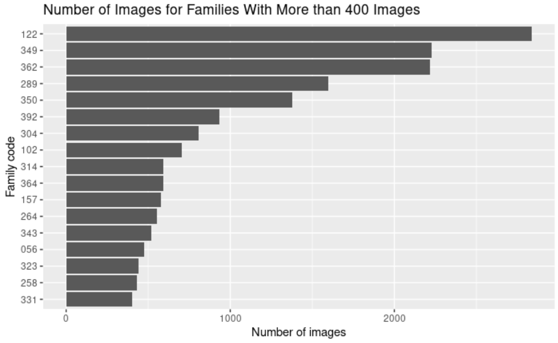


Figure 3. Families with More than 400 Images

4. Methods

To solve this classification problem we utilize convolutional neural networks, which take as input a $x \in \mathbb{N}^{256x256x3}$ image of a fish and output the predicted class of the fish. We draw upon previously tested architectures in our approach, comparing the results of models such as ResNet-, Inception-, and VGG-based networks. For reference, we include an experiment with a basic fully-connected neural net. We trained all models for thirty epochs, a length that allowed most all to reach convergence without overfitting; later in the paper we will detail our hyperparameter tuning (using varied optimizers, learning rates, etc).

4.1. Loss Function

Our task was a classic multi-class classification problem. Therefore, we decided to use multi-class cross-entropy loss,

$$L_{\text{cross-entrooy}} = -\frac{1}{N}\sum_{i}^{N} \log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

where f_j is the j-th element of the class score vector f[2]. This is a pretty standard loss function for this type of task.

4.2. Baseline Models

4.2.1 Fully Connected Neural Network

Our baseline attempts to perform family-level classification using a Multilayered Perceptron (MLP), also called a fully-connected neural network.

Our network features two hidden layers, each connected by an affine transformation and a ReLU nonlinearity. Our final scores are calculated via a final affine transformation and softmax loss function.

4.3. CNN Models

4.3.1 ResNet-50

We use a pre-trained ResNet 50 model to assess how the industry standard performs on our datasets. The ResNet architecture employs skip layers which allows the network to learn the identity mapping, effectively skipping convolutional layers. This key addition is what differentiates ResNets from other CNNs, and is the reason ResNet has became one of the most popular architectures in various computer vision tasks.

Although the model is pretrained on the 1000-class Imagenet dataset [4], we use feature extraction to update the final affine layers of the network from which we derive predictions. This method allowed us to use the pre-trained network as a fixed feature extractor, while shaping the final layer to fit the intricacies of our fish classification task. The basic architecture of ResNet 50 is illustrated in Figure 4. [8]

We further modified the architecture by inserting a small fully-connected net at the end of the network, which included two layers of dropout, an affine connection, and a ReLU nonlinearity. We discovered that inserting these nonlinearities and dropout connections improved the accuracy of the model when compared to using only a single affine layer.

4.3.2 VGG-19 with Batch Normalization

Our third model we used employs the VGG-19 architecture, a CNN which is 19 layers deep and which uses batch normalization. VGG models became quite popular after AlexNet for their reduction in the number of parameters; this greatly reduces training time. The reduction is achieved by using smaller filters and stacking them on top of each other, which allows the filters to have the same receptive field as larger kernels but with fewer parameters and more opportunity for nonlinearity. [10]

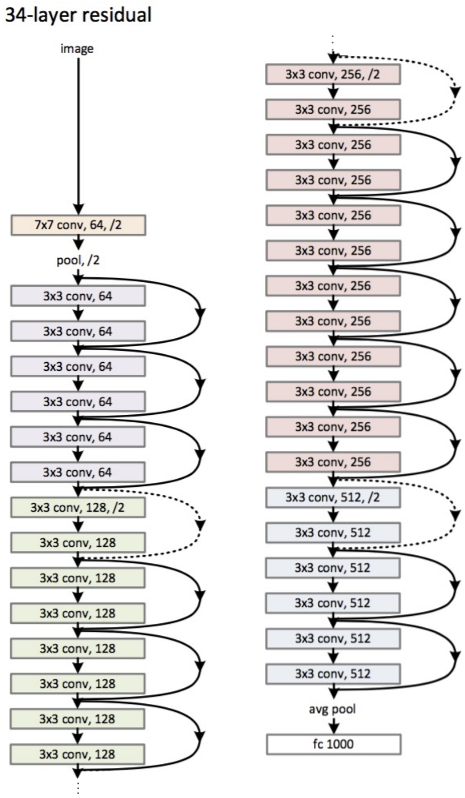


Figure 4. ResNet-34 architecture. Radial arrows illustrate a residual connection between layers. Our model is similar, but deeper.

Using the batchnorm variant of VGG-19, we insert batch normalization at each layer. This normalizes the activations throughout the network, which helps reduce the chance of vanishing or exploding gradients, and helps speed up training. This normalization is performed by normalizing the output of a previous activation layer; it is calculated by subtracting the batch mean and dividing by the batch standard deviation. We furthermore add two trainable parameters to each layer: γ which allows us to adjust the standard deviation, and β which allows us to shift the bias. Mathematically, batch normalization transforms the input as follows:

$$\mu_\beta = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_\beta^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\beta)^2$$

$$\hat{x}_i = \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}$$

$$y_i = \gamma\hat{x}_i + \beta$$

We used the VGG-19 with batch normalization model pretrained on the Imagenet dataset [4], again modifying the final affine transformation. After trying various architectures, we found that a single connected layer proved to be most effective.

4.3.3 Inception-V3

The final model we implemented was the Inception-V3 architecture. This model acknowledges the fact that there can exist variable-sized features, and to account for this the architecture includes multiple kernels of different sizes within the same layer. Larger kernels can detect more global features while smaller kernels are better for smaller area-specific features within the image. Thus, the Inception model expands horizontally rather than vertically, constructing several modules as illustrated in Figure 5. One of the main advantages of this method is that inception increases the network space from which the best model can be found via training. It is therefore more robust to variations in feature sizes throughout the images, and allows the training process to choose which features hold the most weight.

Again, we use a pre-trained Inception-V3 model[4], and use feature extraction to customize the final affine layer. After experimenting with various fully connected architectures, we found that a single affine connection proved to be most successful.

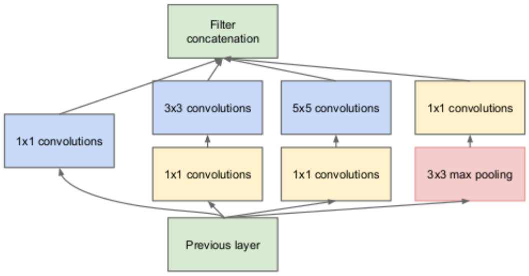


Figure 5. Inception module architecture

5. Experiments and Results

5.1. Metrics

Given the high number of fish species used in training and testing, we expected our classifier's accuracy to be modest at best on our large dataset. We therefore used two measures of success. The first is the accuracy ("acc") of our model in classifying the fish species correctly - mathematically, the number of correct classifications divided by the number of classifications performed.

$$\text{acc} = \frac{\text{Number of images correctly classified}}{\text{Number of images in sample}}$$

The second method is the accuracy over the top three predictions ("acc@3"). This metric considers an example correctly classified if it was one of the top 3 most likely classes outputted by the model. The mathematical formulation for this accuracy is the number of correct classifications over the top three predictions divided by the number of classifications performed.

$$\text{acc@3} = \frac{\text{Number of images w/ correct class in top three}}{\text{Number of images in sample}}$$

5.2. Datasets / Subsets

We also assessed our models on two datasets. Given the high variance in the original dataset in terms of the number of examples for each fish family, we created a second dataset that includes only the families with more than four hundred images: almost twenty families of fish. Though performance on the original dataset is still important to measure, we believed evaluating on this was a better method for measuring algorithms' peformances on this task.

5.3. Hyperparameter Search

After a hyperparameter search, all models used a batch size of 64 images (balancing the need for computational speed and the regularizing effect of a smaller batch size), were trained for thirty epochs, and utilized the Adam optimizer for training. Our Fully-Connected Baseline used a learning rate of 0.0001, whereas the other models were trained with an annealing learning rate starting at 0.0001 and halving every five epochs. When training our pretrained ResNet 50 model, we noticed overfitting (training acc@3 reached 99%) so we added a dropout layer before both fully-connected layers with a drop probability of 0.3; this fixed the overfitting issue.

5.4. Results

Below are the results of evaluating our models on the test set:

Test Set Results			
Model	Acc	Acc@3	Dataset
Baseline FC Net	51.84	72.13	Top 20
Baseline FC Net	27.67	41.75	All
Pretrain InceptionV3	67.38	88.13	Top 20
Pretrain InceptionV3	19.10	31.41	All
Pretrained ResNet 50	**73.32**	**91.80**	**Top 20**
Pretrained ResNet 50	**47.73**	**67.30**	**All**
Pretrained VGGNet	60.19	84.77	Top 20
Pretrained VGGNet	32.54	50.66	All

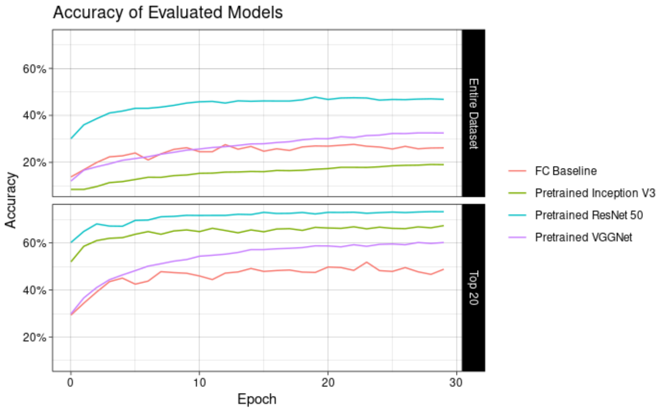


Figure 6. Test set accuracy by epoch

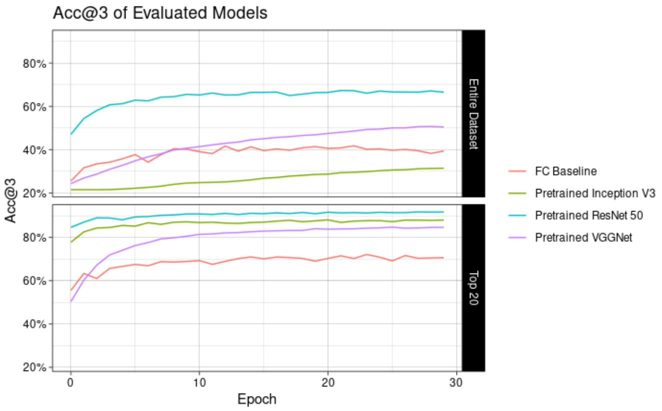


Figure 7. Test set acc@3 by epoch

5.5. Discussion

5.6. Model accuracies

Our model built on top of a pretrained ResNet 50 model outperformed all other models on both evaluation metrics and on both datasets. Because residual networks are able to learn the identity function for certain layers, they are able to choose an effective number of convolutional layers to train. Thus, we could train a large ResNet without too much overfitting (although we also added dropout to our network to ensure that this would be the case). Therefore, it is probable that our pretrained ResNet 50 had learned better features than the other deep networks we used; using these, our final two-layer fully-connected network was able to discriminate between our fish species more effectively than our additions to VGGNet and Inception-V3 were.

Surprisingly, the model built on pretrained Inception-V3 features performed worse than our fully connected baseline on the large dataset for both top-1 and top-3 accuracy. One possible explanation for this could be that the pretrained Inception-V3 model had already selected features ill-suited for our fish classification task. The main benefit of using Inception models is that the architecture finds features of arbitrary size, since it uses multiple kernels of different sizes in each convolutional layer. However, in our dataset the fish pictures are all relatively evenly sized, and thus it is possible that the pretrained model had learned to extract features of a different size than is optimal for this, our fish classification problem. However, this problem does not occur with our top-20 dataset; in fact, our Inception-V3 model performed second-best behind ResNet-50 for both measures of accuracy. Therefore, it seems that feature extraction is not an issue for all examples in our dataset, but rather that our Inception-V3 model performs noticeably worse on the classes with fewer examples. This could either be a result of the data itself - the classes with fewer examples could contain images ill-suited for our Inception model - or it could reflect poorly on the pretrained model's ability to discriminate between classes when there are few training examples. Future work could determine whether this result stays present with a custom-trained Inception-V3 model; if not, then the issue is likely a result of our dataset being ill-suited for Inception-V3 models trained on the ImageNet dataset.

Our pretrained VGG-19 model performed slightly better than the baseline across all metrics. This is likely a product of it being a convolutional neural network, which allows the model to learn more intricacies leading to better performance. However, it was the worst of the three CNNs (excluding Inception-V3 on the full dataset), likely because of its basic architecture and use of fewer layers.

5.6.1 Qualitative analysis

Visualizing the predictions from our models can provide insight into the predictive mechanisms underlying our algorithms. In particular, we can visualize the accuracy of our models across the different classes to determine which classes were easiest and which were most difficult to classify. We used confusion matrices[1] to visualize accuracy by class; Figure 9 illustrates one such confusion matrix for our VGG-19 model trained on the top-20 dataset. The faint vertical lines present represent classes that were predicted more frequently across many different inputs. Though we

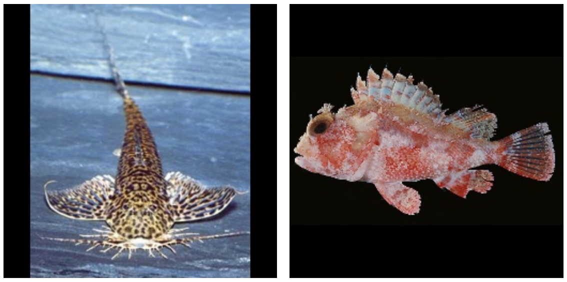


Figure 8. Examples from class 157 (left) and class 264 (right).

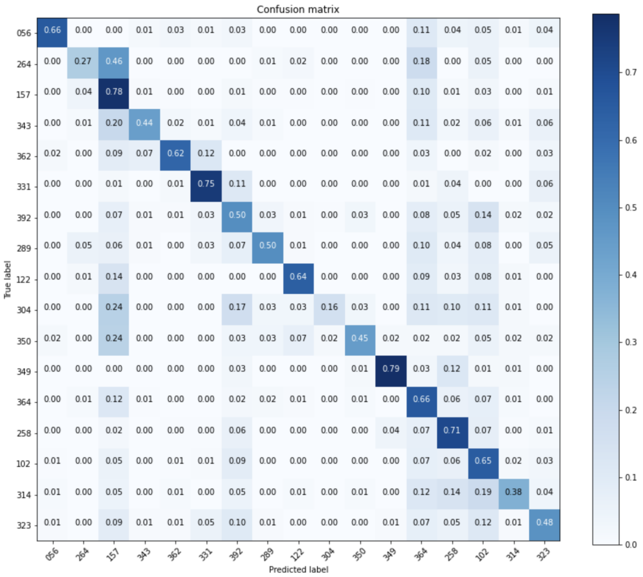
first hypothesized that these may be the result of more populous classes, leading our models to assign larger weights to these fish families, after cross referencing the size of each class in our top-20 dataset we found this theory to be incorrect. Furthermore, these vertical lines are only present in the VGG confusion matrix; the ResNet-50 confusion matrix (illustrated in Figure 10) demonstrates a superior ability to discriminate between classes, confirming our findings that the ResNet model was most accurate on our dataset. There are no visible patterns in incorrect predictions, indicating that the disposition to predict certain classes was unique to our VGG model.

One interesting observation that we noted was the number of incorrect predictions from examples of class 264, specifically the fact that the majority of predictions were labeled as class 157. All three of our models struggled with these two classes. After looking through various examples from each class, we discovered that though the fish had similar structural features, their main differentiator was color: fish from class 264 tended to have a more orangish color while fish from class 157 were black (examples in Figure 8). One possible theory we entertained from this observation was that our classifiers relied more on fish shape than color. This idea is supported by the wide variety of lighting conditions and fish colors within a given class in our dataset, suggesting that our models learned to weight color variations less than structural distinctions.

6. Conclusion

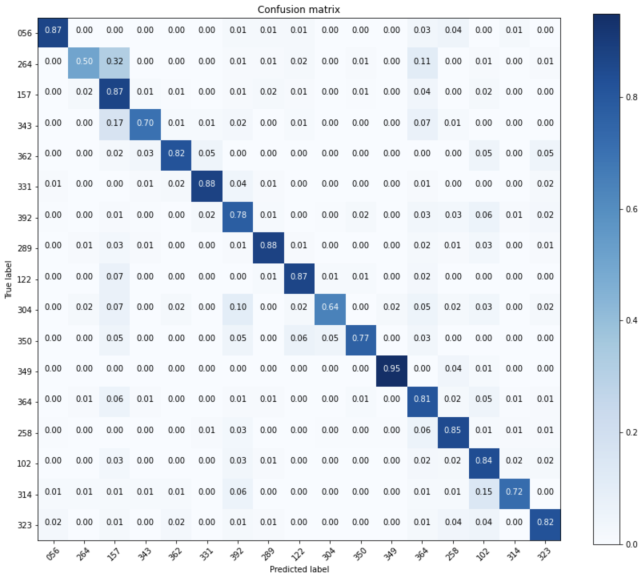
Using transfer learning, we were able to achieve state-of-the-art performance in fish classification, with our highest model scoring 92% acc@3 and 73% acc@1. Additionally, we introduced two novel labeled datasets: one composed of seventeen families of fish, each with more than four hundred images; and the other composed of 324 families of fish, totaling more than 31,000 images.

This level of accuracy is high enough to build useful applications from our model, enabling easier ecological surveys, business operations, and fisheries management. Though we do not yet know with what accuracy a trained fish expert would be able to classify the fish in our dataset,



	056	264	157	343	362	331	392	289	122	304	350	349	364	258	102	314	323
056	0.66	0.00	0.00	0.01	0.03	0.01	0.03	0.00	0.00	0.00	0.00	0.00	0.11	0.04	0.05	0.01	0.04
264	0.00	0.27	0.46	0.00	0.00	0.00	0.00	0.01	0.02	0.00	0.00	0.00	0.18	0.00	0.05	0.00	0.00
157	0.00	0.04	0.78	0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.10	0.01	0.03	0.00	0.01
343	0.00	0.01	0.20	0.44	0.02	0.01	0.04	0.01	0.00	0.00	0.00	0.00	0.11	0.02	0.06	0.01	0.06
362	0.02	0.00	0.09	0.07	0.62	0.12	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.02	0.00	0.03
331	0.00	0.00	0.01	0.00	0.01	0.75	0.11	0.00	0.00	0.00	0.00	0.00	0.01	0.04	0.00	0.00	0.06
392	0.00	0.00	0.07	0.01	0.01	0.03	0.50	0.03	0.01	0.00	0.03	0.00	0.08	0.05	0.14	0.02	0.02
289	0.00	0.05	0.06	0.01	0.00	0.03	0.07	0.50	0.01	0.00	0.00	0.00	0.10	0.04	0.08	0.00	0.05
122	0.00	0.01	0.14	0.00	0.00	0.00	0.00	0.00	0.64	0.00	0.00	0.00	0.09	0.03	0.08	0.01	0.00
304	0.00	0.00	0.24	0.00	0.00	0.00	0.17	0.03	0.03	0.16	0.03	0.00	0.11	0.10	0.11	0.01	0.00
350	0.02	0.00	0.24	0.00	0.00	0.00	0.03	0.03	0.07	0.02	0.45	0.02	0.02	0.02	0.05	0.02	0.02
349	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.01	0.79	0.03	0.12	0.01	0.01	0.00
364	0.00	0.01	0.12	0.01	0.00	0.00	0.02	0.02	0.01	0.00	0.01	0.00	0.66	0.06	0.07	0.01	0.00
258	0.00	0.00	0.02	0.00	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.04	0.07	0.71	0.07	0.00	0.01
102	0.01	0.00	0.05	0.00	0.01	0.01	0.09	0.00	0.00	0.00	0.00	0.00	0.07	0.06	0.65	0.02	0.03
314	0.01	0.00	0.05	0.00	0.01	0.00	0.05	0.00	0.01	0.00	0.01	0.00	0.12	0.14	0.19	0.38	0.04
323	0.01	0.00	0.09	0.01	0.01	0.05	0.10	0.01	0.00	0.00	0.00	0.01	0.07	0.05	0.12	0.01	0.48

Figure 9. Normalized confusion matrix for our VGG-19 model. The graph compares predictions to correct labels, such that the diagonal entries indicate correct predictions. The darker squares correspond to a higher percentage of predictions, meaning the classes with dark diagonals were easiest to predict.



	056	264	157	343	362	331	392	289	122	304	350	349	364	258	102	314	323
056	0.87	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.00	0.00	0.00	0.03	0.04	0.00	0.01	0.00
264	0.00	0.50	0.32	0.00	0.00	0.00	0.01	0.01	0.02	0.00	0.01	0.00	0.11	0.00	0.01	0.00	0.01
157	0.00	0.02	0.87	0.01	0.01	0.00	0.01	0.02	0.01	0.00	0.01	0.00	0.04	0.00	0.02	0.00	0.00
343	0.00	0.00	0.17	0.70	0.01	0.01	0.02	0.00	0.01	0.00	0.00	0.00	0.07	0.01	0.00	0.00	0.00
362	0.00	0.00	0.02	0.03	0.82	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.05
331	0.01	0.00	0.00	0.01	0.02	0.88	0.04	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.02
392	0.00	0.00	0.01	0.00	0.00	0.02	0.78	0.01	0.00	0.00	0.02	0.00	0.03	0.03	0.06	0.01	0.02
289	0.00	0.01	0.03	0.01	0.00	0.00	0.01	0.88	0.01	0.00	0.00	0.00	0.02	0.01	0.03	0.00	0.01
122	0.00	0.00	0.07	0.00	0.00	0.00	0.00	0.01	0.87	0.01	0.01	0.00	0.02	0.00	0.00	0.00	0.00
304	0.00	0.02	0.07	0.00	0.02	0.00	0.10	0.00	0.02	0.64	0.00	0.02	0.05	0.02	0.03	0.00	0.02
350	0.00	0.00	0.05	0.00	0.00	0.00	0.05	0.00	0.06	0.05	0.77	0.00	0.03	0.00	0.00	0.00	0.00
349	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.95	0.00	0.04	0.01	0.00	0.00
364	0.00	0.01	0.06	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.00	0.81	0.02	0.05	0.01	0.01
258	0.00	0.00	0.00	0.00	0.00	0.01	0.03	0.00	0.00	0.00	0.00	0.00	0.06	0.85	0.01	0.01	0.01
102	0.00	0.00	0.03	0.00	0.00	0.00	0.03	0.01	0.00	0.00	0.00	0.00	0.02	0.02	0.84	0.02	0.02
314	0.01	0.01	0.01	0.01	0.01	0.00	0.06	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.15	0.72	0.00
323	0.02	0.00	0.01	0.00	0.02	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.04	0.04	0.00	0.82

Figure 10. Normalized confusion matrix for our ResNet-50 model. Notice the higher accuracy across all classes when compared to the VGG confusion matrix.

our model certainly performs better than either of the authors of this paper would be able to themselves.

Future researchers may be interested in trying to improve the performance of deep learning models on our "large" dataset, which includes many classes with few example images. This dataset could easily be augmented with standard methods to provide more training examples; because we didn't perform augmentation, our accuracy was notably lower on the large dataset.

In the field of fish classification, many of the highest-performing models rely solely or largely on hand-selected features often chosen by experts in the field. Our experiments have shown that models taking as input features from pretrained, deep neural networks can perform at par or better than systems relying on hand-selection and domain knowledge.

7. Contributions

Major thanks to Google Colaboratory for providing a space to develop our models; to the FishBase community for compiling the dataset over the years; and to the Pytorch community for providing us with a framework and pretrained models to build on. In this project, Michael developed the Fully-Connected Baseline and VGGNet models, and Callan made the ResNet 50 and Inception-V3 ones. We both collaborated equally on the milestone report and final project report.

8. Appendix

Github repo: https://github.com/callanhoskins/fishClassification.

References

[1] Cnn confusion matrix with pytorch - neural network programming https://deeplizard.com/learn/video/0lhis6yu2qq. 5

[2] Cs231n notes: Linear classification. 3

[3] Finetuning torchvision models - https://pytorch.org/tutorials/beginner/finetuning-torchvision-models-tutorial.html. 2

[4] torchvision.models - https://pytorch.org/vision/stable/models.html. 2, 3, 4

[5] M. K. Alsmadi and I. Almarashdeh. A survey on fish classification techniques. *Journal of King Saud University - Computer and Information Sciences*, 2020. 1

[6] H. S. Chhabra, A. K. Srivastava, and R. Nijhawan. A hybrid deep learning approach for automatic fish classification. In *Proceedings of ICETIT 2019*, pages 427–436. Springer, 2020. 1

[7] S. Cui, Y. Zhou, Y. Wang, and L. Zhai. Fish detection using deep learning. *Applied Computational Intelligence and Soft Computing*, 2020, 2020. 1

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. 2, 3

[9] D.-J. Lee, R. B. Schoenberger, D. Shiozawa, X. Xu, and P. Zhan. Contour matching for a fish recognition and migration-monitoring system. In *Two-and Three-Dimensional Vision Systems for Inspection, Control, and Metrology II*, volume 5606, pages 37–48. International Society for Optics and Photonics, 2004. 1

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 2, 3

[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. 2

[12] A. Taheri-Garavand, A. Nasiri, A. Banan, and Y.-D. Zhang. Smart deep learning-based approach for non-destructive freshness diagnosis of common carp fish. *Journal of Food Engineering*, 278:109930, 2020. 1

[13] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019. 2

[14] B. Zion, A. Shklyar, and I. Karplus. In-vivo fish sorting by computer vision. *Aquacultural Engineering*, 22(3):165–179, 2000. 1