**Background**

A picture is worth a thousand words. A lot of information can be absorbed by looking at an image or scene closely and inspecting what is present. It’s the reason a person can call themselves a “visual-learner” and why it is often helpful to accompany a data heavy argument with an infographic that conveys the basis of what a person is saying. The human brain can identify an image that’s been seen for as little as 13 milliseconds (2). While humans are gifted at recognizing and naming images, computers do not have the same innate talent. However, with the right tools, a computer can detect patterns within images with greater speed than any person could hope to achieve. This task is referred as image classification: the labeling of images into one of a number of predefined classes (1). Image classification has a great amount of utility in a modern world, such as detecting cancer from an x-ray, facial recognition, and even autonomous driving (3).

**Purpose**

This project will develop a model that will be able to view a photo of a fish and determine the type of fish from a predetermined list. This will be accomplished by using a convolutional neural network to analyze imaging data from photos of fish that can then be used to classify the fish. This model will serve as an example of a use case of image classification within a defined space. In the context of a real-world situation, this model could be used to automatically separate fish into their distinct groups by using images taken while the fish are transported along an assembly line. Other aspects of this project will serve as the steps that are necessary to process images into a usable data format, such as amalgamating data into array that can be used to store and manipulate data and augment a small amount of data through the creation of slightly modified images.

**Data Investigation**

The data for this project consists of 430 images from 9 types of fish. The list of fish are as follows: black sea sprat, gilt head bream, horse mackerel, red mullet, red sea bream, sea bass, shrimp, striped, red mullet, and trout. There were 50 images for each type of fish, apart from trout which only had 30 images. The resolutions for these images were split between 2832 x 2128 and 1024 x 768, giving the images an aspect ratio of 4:3. The images themselves were all taken from a similar viewpoint, with the camera viewing the fish from above as it was placed on a blue mat, with the fish running somewhat parallel to the wider dimension of the photo. All photos were in color.

**Data Cleaning** **and Preprocessing**

First, all the photos were imported into Python can converted into numpy arrays for easy manipulation. To address the differing resolutions of the images, as well as the high dimensionality of the images in general, all the images were resized to 590 x 445, which had the benefits of both reducing the dimensionality of the images and maintaining the aspect ratio for all images. The labels for all the images were one hot encoded for use in the algorithms and to prevent the model from assigning higher importance on any one of the labels due to the size of an integer. The image data and labels were then split into training and testing sets to measure the success of the model at the final stage. Finally, an ImageDataGenerator object was created to creating additional images to be used in model training by slightly altering existing images to overcome the issues of the dataset containing a small number of images.

**Model Development and Performance**

The model used to classify the images in this project was a neural network. The foundation of this neural network focused on convolutional layers paired with pooling layers and a final fully connected layer that would serve as the final output of the model. The alternating convolutional and pooling layers had the benefit of letting the model look at how a pixel compares to its neighbors, which allowed for useful patterns and features to be detected within the images. After the convolutional and pooling layers, the data was flattened so it could be used in the final layer to make the prediction. Due to the introduction of generated images, a dropout layer was added to the model to reduce the risk of overfit within the model.

The model was then run for 50 epochs, which was hypothesized to provide the model sufficient time to create a useful model. The accuracy of the final layer of this model was 97.95%, which shows promising initial results for the model. The validation accuracy for this same timepoint was 86.27%, which shows that the data may have been overfit slightly but held that the model was still effective. A plot showing the model training over time can be seen in Figure 1.

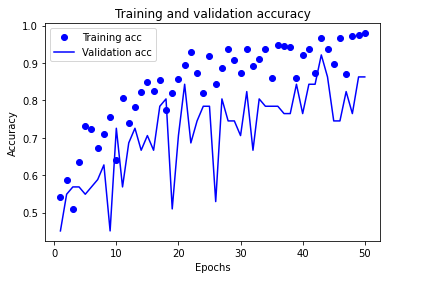


Figure - Visual of Model Performance.

**Conclusion**

The high accuracy of the preliminary model is a good indication that a convolutional neural network was a good choice to solve this task. While the initial results are promising, improvements to the model can still be made. Tweaking of the model to use additional convolutional and pooling layer may assist with model accuracy even further. In addition, further image processing methods (edge detection, ground truth, etc.) will be explored to determine if they may be effective in increasing model performance. The testing data remains untouched and will only be used once the final model has been determined. A full classification report will be reported alongside accuracy to evaluate the success of the final model.

**Appendix A**

**A.1 References (will properly format them in final draft)**

1. <https://iq.opengenus.org/basics-of-machine-learning-image-classification-techniques/>
2. <https://www.seyens.com/humans-are-visual-creatures/>
3. <https://iq.opengenus.org/basics-of-machine-learning-image-classification-techniques/>

**A.2 Q&A**

1. Would it have been possible to set the resolution for all images to 1024 x 768 to incorporate more data in the model?
   1. While that is certainly an option, I believe the amount of data that exists within a 590 x 445 resolution will be enough to accomplish this task with satisfactory results. A larger resolution may increase the time need to make predictions as well, which could slow the process in which the model is involved.
2. Were there other algorithms suited for this task?
   1. An artificial neural network could have been used but would not have had the same benefit as a convolutional network, as the positions of pixels in relation to one another would not have been able to be considered.
3. What concerns are there with the model according to the graph in Figure 1?
   1. The spikes of validation accuracy mean that there are more improvements that can be made in the model and that it may also be necessary to increase the amount of data that is dropped before the final classification is made.
4. How many images are being made using ImageDataGenerator?
   1. The number of images is not precisely calculated. Instead, the parameters of the new images are defined (rotation, flipping, resizing, etc.), and then new images are created in tandem with model training.
5. Is there a way to see some exploratory analysis of the data beyond simply aggregating them into arrays?
   1. I am hoping to get an average image for each fish, which would sort of look like each image of each fish laid on top of another until a fuzzy image remains showing the general outline of what is to be expected from each image. I have been having difficulties getting it to work, but I hope it will be ready by the final submission date.
6. Have the features been standardized before being used in model training?
   1. Not yet. The features are standardized within the ImageDataGenerator, but I need to rescale the images by resizing them by a factor of 1/255 (making each input exist on a scale of 0 to 1).
7. What difficulties did you have with the project?
   1. Getting the images into the arrays in a way that let them be fed into the models. Originally, each image was extracted into an array and placed within a pandas dataframe. When later accessed as a group, pandas put them into another array, nesting the arrays of each image and preventing the algorithms from seeing the true shape of the data, which threw up errors. The fix I found is to convert each inner array into a list, which has been time consuming.
8. Is this project applicable to more than just the use case you mentioned in the beginning?
   1. Yes, for instance if you wanted to scan each fish at time of purpose to automatically ring up the customer with the price, that should be possible to do with the algorithm as well. In a more general use sense, this type of model can be used anytime you want the computer to figure out what it is looking at.
9. What other visuals will be prepared for the final presentation?
   1. The classification report and the average images that I mentioned earlier are the things I am focusing on when it comes to creating visuals for the project.
10. What limitations exists with the project right now?
    1. Time and lack of data. While image augmentation is helping the problem, more images in general would be useful in helping to train the model. Not much I can do about the time aspect, however.