HUMPBACK WHALE IDENTIFICATION

Anoop Jakkala
Meng, Robotics Engineering
Electronics and Computer Dept.

Harseerat Sohal
Meng, Software Engineering
Electronics and Computer Dept.

Shivani Solanki
Meng, Software Engineering
Electronics and Computer Dept.

Abstract

This report describes about the current work done on whale photo surveillance system. We are developing a deep learning model using convolutional neural network to be used to identify the type of humpback whale. The whale species are identified by the light and dark pigmentation patches on their tails. The data set it taken from Kaggle competition.

Introduction

After centuries of intense whaling, recovering whale populations still have a hard time adapting to warming oceans and struggle to compete every day with the industrial fishing industry for food. To aid whale conservation efforts, scientists use photo surveillance systems to monitor ocean activity. They use the shape of whales' tails and unique markings found in footage to identify what species of whale they're analysing and meticulously log whale pod dynamics and movements. For the past 40 years, most of this work has been done manually by individual scientists, leaving a huge trove of data untapped and underutilized.

The challenge here is to build a machine learning model to detect and analyse the whale species in the HappyWhale database of 25000+ images. This project will help the people and the researchers to automatically identify the whale species as the images are taken and will also speed up the process of image identification.

Dataset Description

The training dataset contains 25361 images and the test set consists of 7960 images. The whales are classified into 5005 classes out of which one is 'new_whale'. The species of whales which are not identified yet are classified as new whale.

As seen in figure 1, the classification data is imbalance in the number of images available per class. For most of the classes there are only one or two images available in the data set. Whereas, around 9664 images from train set belongs to "new_whale" category. The graph shows that there are

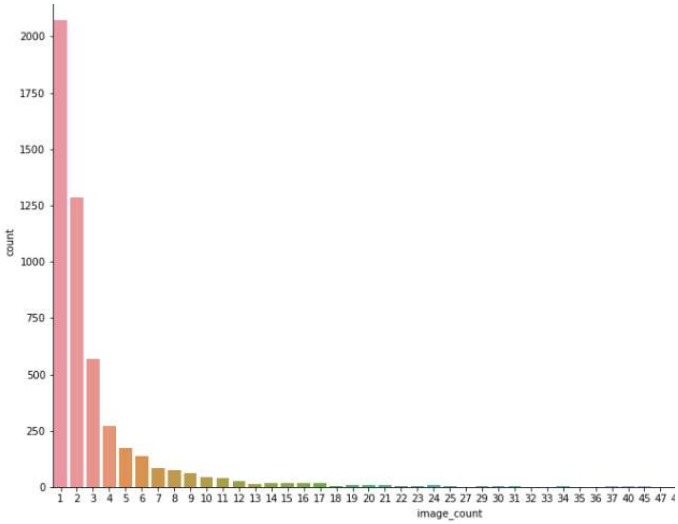

Figure 1

2000+ classes with just one training example and around 1300 classes with 2 training examples. It also shows that around 50 classes have more than 22 training examples. So, images distribution is like almost 30% comes from whales with 4 or less images, around 40% comes from 'new_whale' group and the rest

30% comes from whales with 5-73 images. This also causes problem in splitting the data into train set and validation set.

Data Pre-processing

In the train data set, some images are of RGB type while some are in greyscale. So, we changed all images to greyscale to reduce the computational complexity involved with RGB images. On later stages, this model can be extended to RGB images with minor changes.

Duplicate Image removal

To remove the duplicate images, we used perpetual hashing i.e pHash, which acts as an image fingerprint for each image, from imagehash library. Phash algorithm analyzes the images and represent it using 64-bit number fingerprint. Two images phash values are close to each other if the image contents are similar.

We compared two image's phash value and if the difference of the phash was less than 6 bit we considered those images similar.For the multiple images with close phash values and same id, we selected the image with highest resolution. We found 778 duplicate images. The errors associated with it were that the same image with the corresponding Id appeared multiple times. The same image which appeared with any other Id and as "new_whale", we deleted the "new_whale" entry. The images with close hash values and which appeared with different Ids, we deleted all of them due to ambiguous classification.

Image Cropping

In most of the images the whale fluke occupies only a tiny fraction of the picture. Downscaling first will result in loss of details and poor quality of cropping. So we used the approach of bounding box to zoom in the region of Interest in the picture which will provide the greater accuracy to a classification model. As the name suggest in bounding box we draw a box around the image of interest based on our requirements.

To automate the process of cropping images, we constructed Bounding Box model using Convolutional Neural networks(CNN) which was then used to crop the high resolution image. To created the bounding box we reffered to bounding box coordinates for 1200 images which was created by 'Martin Piotte's article' (mentioned in references) on humpback whale data set. Out of 1200 images we took 1000 images and augmented it 15 times to increase the size of training data. The images were converted to black&white, resized to 128x128, normalized to zero mean and unit variance. We used 4 layers in CNN with 2x2 convolution with stride 2, along with batch normalization and dropout. In end, we used max pooling on rows and columns separately. The model was trained on 8000 images created after augmentation and tested it on remaining 200 images. One such example of computed bounding box on the final trainset can be seen below in figure 2.

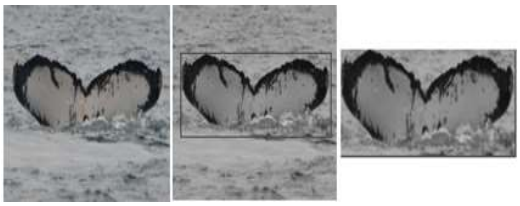

Figure 2

Data Augmentation

As mentioned above there was a problem of unbalaned classes in our dataset. The unbalanced classes created problems due to two main reasons.

a) We don't get optimised result for the imbalanced class as the model doesn't have sufficient data to learn about that class.
b) This also create a problem in spliting the data between training and validation sample as number of samples in few classes is only one.

To deal with the serious problem of unbalanced classes we used data augmentation techniques. We created 15 images in all the classes which has less than 15 samples with different orientation of the same image (of unbalanced class) and copied those back into the training dataset. We scaled the images of each class to min of 15 through data augmentation.

The data augmentaion techniques tht we used are:-

1. Rotation- rotated images at 10 degree angle.

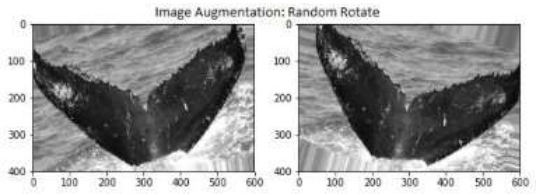

Figure 3

2. Horizontaly flipped the images on the middle asix.

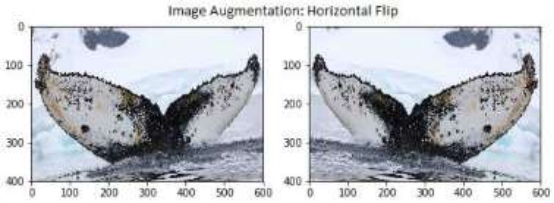

Figure 4

3. Zoomed in and out the images.

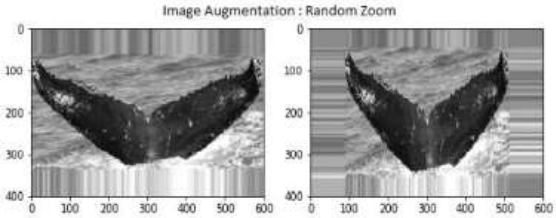

Figure 5

4. Shifting: giving random shifts to the images.

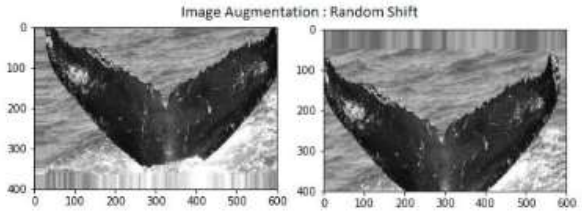

Figure 6

Proposed Machine Learning Models:

Model 1: Plain Convolutional Neural Nets (CNNs):

Convolutional Neural Network is a deep learning model which take input image, assign importance to every feature and can differentiate between the features. The input image is converted into an array of pixels. Each input image is passed through many convolutional layers with filters also called kernels, pooling layer, fully connected layers and apply softmax function to classify the images into probability values from 0 to 1. Generally coordinating the two images of their positions, these models will get exceptionally productive and stable by finding the image similarities than complete matrix matching schemes (Figure 7).

Convolution Layer:

This layer is the 1st layer of (convolution neural networks) CNN to extract the features of the image. It preserved the relationship between pixels by learning the features using small squares of input by applying mathematical operations on the image matrix and filter. Convolution of image with different filters perform different operations like edge detection, blur and sharpening of the image.

Pooling:

This technique takes expansive pictures and shrink them down while protecting the most pertinent data in those images. We used Max pooling for our model. Under this technique, it selects a small window of the array and finds the maximum value that must be assign to the

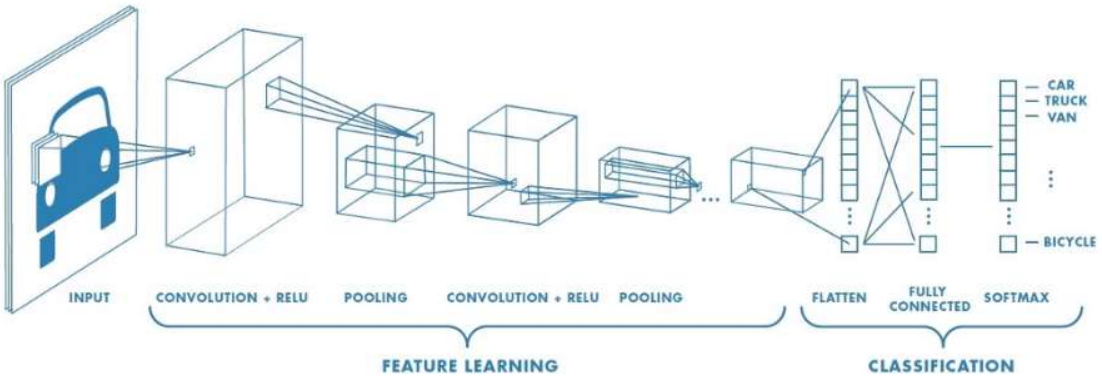

Figure 7 (CNN)

output. Mostly, the window size is to be in 2-3 pixels.

Fully Connected Layer:
The matrix from the pooling layer is flattened into vector and fed to the fully connected layer. The feature map matrix from the pooling layer are combined to create a model. The output from the fully connected layer is applied to activation function to classify the images.

Activation Functions:
ReLU:
The activation function utilized in this model is ReLU. ReLu stands for rectified linear unit function. Here, this layer reassigns a serial number. So, the values measures are match with the model. For a -ve range it assigns 'zero' and +ve range it assigns 'One'. This helps the CNN model to stay learning values and weights among the desired vary, instead of obtaining blown to time.

Softmax:
This activation function is generally applied to the last layer of the machine learning models. This function returns the chances of each class, for any input file. The output is within the style of float array values between zero and one.

Dropout:
It is used to prevent overfitting in neural networks. This layer selects some neurons indiscriminately and quickly deactivates the neuron for training phase. Their contribution in training is temporarily stopped for the aerial, therefore, no weight update is finished for those neurons during the backward pass.

Cross-entropy loss function:
Cross-entropy loss measures the performance of a classification model whose output may be a chance worth between zero and One. Cross-entropy loss will increase because the expected chance diverges from the actual label.

Layers Used:

We used 3*3 filters with 4 CNN layers with stride 2 and 3 Maxpooling layers and 1 Average pooling layer each of size 3 x 3 for Down sampling with CNN layers with stride 2. The Padding was kept as Same with 2 Batch Normalization layer. The ouput of above convolution layer were further given as input to Fully Connected layer with hyperparameters as softmax activation, Adam Optimizer for learning rate optimization and categorical cross-entropy loss function for multiclass Classification.

Training Parameters:

The trainset was split into Train and Validation set with ratio 70:30. The model was trained on trainset with 100 epochs and batchsize if 500 , it was then validated of validationset.

Model 2: Residual Network (ResNet) CNN:

Residual Neural Network, ResNet are advanced version of Convolutional neural network. Neural Networks are universal

fuction approximators and accuracy increases with increase in number of layers. But when the network grows deep, the gradient where the loss fucntion is calculated shrinks to zero. This results in the weights never updating its value and the model doesn't learn anything further. ResNet solves the problem of vanishing gradient. To do so it uses the concept of Skip connections or Residual connections. In skip connections the original input of one convolution layer is added to the output of the convolutional block. It also allows the model to learn the identity function to ensure that the higher layer will perform as well as the lower layer.

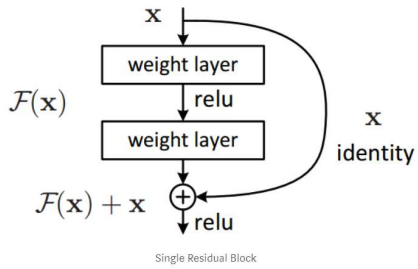

Figure 8

Two main types of blocks are used in ResNet depending on weather the dimensions of the input/output are same or different.

a) Identity Block: It is a standard block used in ResNet and is used when the input activation has the same dimension as the output activation
b) Convolutional Block: When the input and output dimensions are different the convolutional layer is added in the shortcut path.

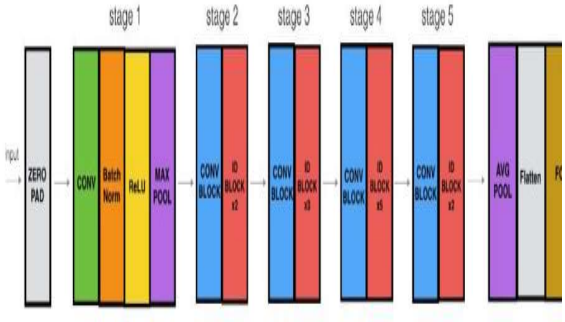

Figure 9

We created the below ResNet Architecture:

- Zero Padding layer
- Stage 1 – Convolution layer with 64 filters having (7,7) kernel and (2,2) stride. BatchNorm is applied to the channels axis of the input. MaxPooling uses a (3,3) window and a (2,2) stride.
- Stage 2: 1 covolutional block and 3 identity block with both having 3 filters of size (128,128,512)
- Stage 3: 1 convolutional block and 2 identiy block both having 3 filters of size (256,256,2048)
- The 2D Average Pooling uses a window of shape (2,2).
- Flatten layer
- Fully connected layer with softmax activation function.

Evaluation:

We used accuracy and Mean Average Precision of top 5 predictions for evaluating our model. Accuray is calculated by the below formula:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

As we had an object detection problem, Mean average precision(map) is the best evalutation metric. Precision is the percentage of positive instances out of the total predicted positive instances. We took the mean of first five predicted classes and used the below formula to calculated the map.

$$MAP@5 = \frac{1}{U}\sum_{u=1}^{U}\sum_{k=1}^{min(n,5)} P(k) \times rel(k)$$

Where, U is the number of images, P(k) is the precision at cutoff k, n is the number predictions per image, and rel(k) is an indicator function equaling 1 if the item at rank k is a relevant (correct) label, zero otherwise.

Results:

For trainset the plain CNN model gave 65% accouracy(Figure 10) and 50% for validation set (Figure 11). The Kaggle score in terms of

MAP metrics for testset was 0.41. These observations clearly indicate overfitting and under-generalization.

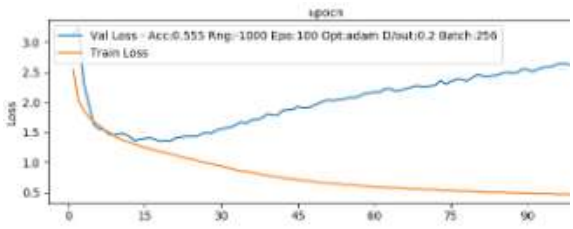

Figure 10

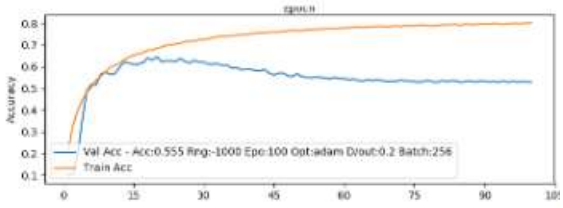

Figure 11

We then applied ResNet which deals with the various drawbacks of the CNN as explained above.

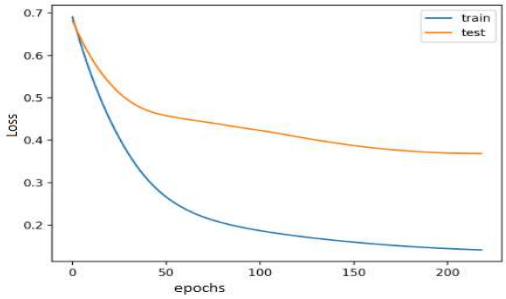

Figure 12

As seen in the plot above, the losses decreased significantly compared to last model. The Training accuracy and validation accuracy achieved were 72% and 59% respectively. The MA boosted to 0.52 through ResNet.

Conclusions:

The allocation of resources was challenging as GPU was required to handle such huge dataset, we used google colab for these models. Due to high data imbalance, data preparation required much efforts as with data augmentation only, the total image count in the trainset reached over 0.2 million images.

We tested two models plain CNN and ResNet. The finding fo ROI and key points was performed by Bounding box model on the amassed collection of whale images. Accuracy could have been improved by excluding images where whales were partially visible. For reason like this, we believe there is still room for improvement in this challenging task.

References:

Robert Bogucki, Marek Cygan, Christin Brangwynne Khan, Maciej Klimek (Applying deep learning to right whale photo identification

Shubrashankh Chatterjee (Towards data scirnce: Deep learning unbalanced training data)

Martin Piotte (https://sites.google.com/site/pragmatictheory)

https://www.kaggle.com/c/whale-categorization-playground/overview

Vladislav Shakhray (https://towardsdatascience.com/a-gold-winning-solution-review-of-kaggle-humpback-whale-identification-challenge-53b0e3ba1e84)

https://www.slideshare.net/ren4yu/humpback-whale-identification-challenge

ImageHash https://pypi.org/project/ImageHash/

Perceptual hash algorithm http://www.hackerfactor.com/blog/index.php?/archives/432-Looks-Like-It.html

Pytorch data loading tutorial https://pytorch.org/tutorials/beginner/data_loading_tutorial.html

Pytorch transfer learning tutorial https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33

https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8

https://towardsdatascience.com/image-detection-from-scratch-in-keras-f314872006c9?gi=aa77e56696be

https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

https://www.analyticsvidhya.com/blog/2019/01/build-image-classification-model-10-minutes/

https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

https://www.analyticsvidhya.com/blog/2019/01/build-image-classification-model-10-minutes/

https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d?gi=7c78d217912b

https://cloudinary.com/blog/how_to_automatically_identify_similar_images_using_phash

Appendix:

The project work was divided equally amongst the team members. Mr. Anoop handled the part related to data exploration. He analysed the data and visualised the same in plots to find out the total no. of unique classes, image count of individual whale class, image distribution in the trainset, the resolution of the images, mode and size of the images. This exploratory analysis suggested a severe imbalanced classes problem with largest class by a longshot is the “new_whale” label. Also with his obersvations we found out that the images did not have same color spectrum with respect to RGB but mixture of different colour schemes, after which we decided to convert all images to greyscale. He prepared the CSV files for the final trainset, helped in model training and made the repositories of the code on Bitbucket.

Ms Harseerat and Ms Shivani worked together on the datapreprocessing, model building and training and evaluation as the dataset was huge. Ms Harseerat worked on image cropping by building the bounding box model using CNN and the image coordinates CSV file. She handled the data augmentaion part as well for both the bounding box model and the final models (CNN and ResNet). She also contributed to CNN and ResNet model selection and building, deciding the hyperparameters. She trained the models with different combinations to optimise the models. She prepared and uploaded the final trainset on the google drive to be used directly in googlecolab. She prepared the starter codes for training in googlecolab. She coordinated and compiled all the codes from all group members as final for submission.

Ms Shivani worked on data preprocessing based on the observations by Mr. Anoop. She handled the part for detection of duplicate images and removal the one with ambigous classes and to select the one with best resolution against each ID. She used the concept of perpetual hash for image segregation of similar images. She contributed to the design of CNN and ResNet and trained the models on googlecolab with different combinations of hyperparameter to find out the best optimized model. She also worked on dumping the model and further on the model evaluation and results using testset using the evaluating metrics specified in the report. She prepared the presentation and final report for the project.