Udacity – Machine Learning Engineer Capstone Project Proposal – Matthew Burt

Domain

Image classification, in the computer vision field, is the process of classifying images into different categories based on the images content. It is a popular field that has seen massive improvements over the past few years, especially with the application of convolutional neural network (CNN) algorithms to the task. My project proposal involves using computer vision algorithms to correctly classify pictures of fish into the correct species category. This will enable users real time information of the size and bag limits of the fish they have caught.

Problem and Proposed Solution

The problem I am trying to solve is one of image classification, namely classifying fish into the correct species. There is currently a website that serves the purpose of giving fishers information on size and catch limits, but it requires users to either, know roughly the species or click into a lot of links to get the information.

I plan to use SageMakers image classification algorithm, a deep convolutional neural network (ResNet) model, trained on the SageMaker platform, to classify images of fish into the correct species in order to return this information. This model will be deployed on AWS and I will create an endpoint which will be used to test the model's accuracy. Once I have assessed and finetuned the model to an acceptable level I will convert it to ONNX format for use in my web app. This will accept a user supplied image and return a classification. Upon returning a classification the web application will load an image of the predicted fish species along with other information I will load such as size and catch limits.

Performance Metrics and Benchmark Model

I will use accuracy as my model performance metric, defined as the correct predictions /divided by total predictions. Accuracy is chosen as this is the metric that needs to be optimised. The final model will be identifying and returning one of a certain number of classes. Because the user is most concerned about receiving information about the correct fish, it is most appropriate that the model is accurately returning the correct class.

For a Benchmark Model I will train a Convolutional Neural Network in PyTorch from scratch. This will be done in Jupyter and the accuracy of this model will be used for the benchmark.

This model will be defined with 4 convolutional layers, each with ReLu activations and Batch Normalisation. I will also use a max pool layer and a final fully connected linear layer to output the fish species classes. Details of the benchmark model structure can be found appended to this proposal.

Dataset

This is a supervised learning problem so will require data labelled with the corresponding category classes. There are some freely available datasets of classified fish images, but none that are specific to the state of Victoria's fish species. As part of my project I will create a labelled dataset of images to represent the fish controlled by the Victorian Fisheries Authority. This is a large task and will only be partly completed for this project to include a limited number of species.

My data is scraped from Google images and is in various sizes and contexts. Because of the types of images people take of fish the images generally fall into three categories, fisherman posing with fish,

fish out of water or fish swimming in the ocean. I have chosen four species of fish to begin with. Albacore, Australian Herring, Flounder and Snapper. The distribution of these images is below, as well as the assigned categorical classes (in brackets): Albacore (0): 300, Australian Herring (1): 87, Flounder (3): 210, Snapper (4): 350. I will split the data randomly into a training and testing set with a ratio of 70:30. For the validation set, I will use a separate set of images sourced from free images repositories, excluding google images, with 10 images for each class totalling 40 images.

Proposed Workflow

1. I will create a list of species to focus on using the size and bag limit species of scaled marine and estuary fish from the VFA website.
2. I will use Fatkun Batch Download Image tool to batch download images of each species from Google. https://chrome.google.com/webstore/detail/fatkun-batch-download-ima/nnjjahlikiabnchcpehcpkdeckfgnohf?hl=en
3. I will manually clean the images and create a labelled dataset of each fish species; I will sort these into sub folders named as the class name. This will be zipped and uploaded to drop-box for easy download into my SageMaker notebook instance.
4. I will locally train a PyTorch classifier as my baseline model.
5. Next, I will train my SageMaker CNN model using transfer learning. The data needs to be in a specific format to use in the training of the model. This includes resizing the images, converting them into RecordIO files and uploading them to my S3 bucket that the SageMaker session is using, in this processing I will also split the test and train datasets.
6. I will upload the dataset to AWS and create a trained image classification model in SageMaker.
7. I will build a web application using React.js that uses the trained ONNX model to classify images supplied by the user.

Deliverables

The deliverables I will produce for this project are:

1. A labelled dataset of fish species images for Victorian fisheries species.
2. A complete Jupyter notebook with data summary, transformation and exploration.
3. A trained CNN classification model in ONNX format.
4. A web application using the trained model.

References

https://github.com/gabehollombe-aws/sagemaker-image-classifier-to-onnx-in-browser ONNX deployment of image classifier on a web app

https://docs.aws.amazon.com/sagemaker/latest/dg/image-classification.html AWS Image Classification resources.

https://arxiv.org/abs/1512.03385 Kaiming He, et al., 2016 IEEE Conference on Computer Vision and Pattern Recognition

https://vfa.vic.gov.au/recreational-fishing/recreational-fishing-guide/catch-limits-and-closed-seasons/types-of-fish/marine-and-estuarine-scale-fish VFA fish species data

https://github.com/MorvanZhou/PyTorch-Tutorial resource for PyTorch baseline model.

Benchmark model architecture

```
CNN(
  (conv1): Sequential(
    (0): Conv2d(1, 8, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(8, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (conv2): Sequential(
    (0): Conv2d(8, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (conv3): Sequential(
    (0): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (conv4): Sequential(
    (0): Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (out): Linear(in_features=6272, out_features=10, bias=True)
)
```