50.039 - Theory and Practice of Deep Learning PASCAL VOC Classification Project

Lee Kwan Meng, Tan Oon Tong

March 21, 2019

1 Introduction

The PASCAL VOC 2012 [1] dataset provides multiple binary classification of class objects, in which images have one or more of these class labels (repetition of images across classes). Having made an attempt at this multilabel classification task, this report aims to describe the methods taken and document the process for reproducability.

2 Methods

In this section we describe the problem setup, our model design and training phases in detail.

2.1 Data

Images and labels from PascalVOC are used in setting up the Pytorch `Dataset` and `Dataloader`, with the aid of the parser script `vocparseclslabels.py` provided.

Dataset For our custom `Dataset`, images examples and ground truth labels are stored in a dictionary mapping. In this mapping, keys are the paths to the image within the image directory `/jpg` and values as their respective class labels. Class labels of each class are stored in multi-hot label encodings, allowing image examples with more than one class to have the appropriate ground truth representation. Actual images are only retrieved from `__getitem__`, where the extracted PIL image is then transformed to `Tensor` and returned.

Dataloader and Augmentation Separate transforms are applied to training and validation set. Images for the training set receive 2 image augmentations, namely random (0.5 probability) resize and crop, followed by a random horizontal flip. It is with these data augmentation techniques that improve the model's ability to generalize, while not destroying the image's natural semantic structure (such as by using a vertical flip). Images for the validation set have fixed resizing and crop to ensure consistency in validation scores. Five crop is also used as a form of averaging prediction scores over a certain image, without

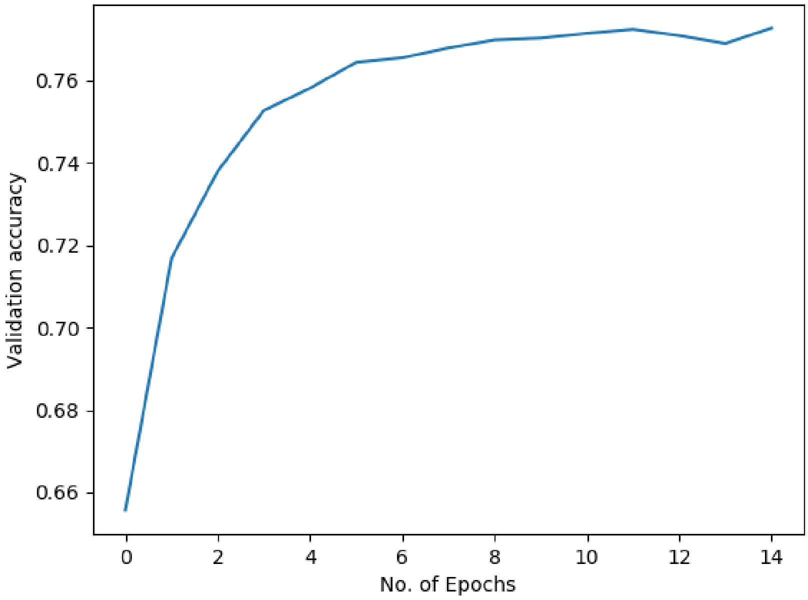

Figure 1: Validation Accuracy over Epochs

introducing randomness (such as of a random crop) at validation time. Following which, images in both sets are converted to tensors and normalized with respect to normalization values from the ImageNet dataset (since we are using a model pretrained on ImageNet).

2.2 Model

The model used was a ResNet18 [2], with pretrained weights on ImageNet dataset. We changed the output size of the last fully conncected layer of the model to fit the 20 classes. All layers of the model were unfrozen, allowing fine-tuning of lower layers too.

2.3 Training Details

Hyperparameters We used a batch size of 32 images, with gradient updates using Adam optimizer, initialized with a learning rate of 0.001. Adam optimizer was chosen over standard SGD with momementum as we hypothesized that weight updates in the lower layers of our ResNet model may be small. As the step size of Adam update rule is invariant to the size of the gradient, it should aid in going through such points with small gradients.

Loss Function Given that the predictions are made class-wise, it seems natural to use binary cross entropy loss, where logits from the model's last layer are inputs into the softmax function. The negative log of probablities for each

Aeroplance	96.4
Bicycle	81.1
Bird	94.5
Boat	86.8
Bottle	61.4
Bus	92.0
Car	78.0
Cat	95.2
Chair	71.6
Cow	79.7
Dining Table	64.2
Dog	90.6
Horse	88.4
Motorbike	87.1
Person	94.9
Potted Plant	53.4
Sheep	86.3
Sofa	58.1
Train	93.8
TV Monitor	84.0

Table 1: Average Precision of Classes (%)

class are then averaged out as the model's loss value. Training of the model was carried out over 15 epochs as the validation accuracy does not seem to improve past this point, as seen in Figure 1. At every epoch the weights that produces the best validation loss are updated as the best weights and saved.

2.4 Results

Average Precision For a more informative measure of the performance of our model on the validation set, average precision (AP) is used, where it sums up the area under the precision-recall curve across all examples, for each class in the validation set. AP is preferred over accuracy in this classification task as it takes into account false negatives, such as not detecting classes found in the image (”playing it safe”). This is especially crucial for multi-class detection, where a model which misses many labels would then be rightly penalized. AP of each class can be seen in Table, with a mean AP of 81.9%, which has a relatively decent position in the public leaderboard.

Tail Accuracy Analysis on how varying threshold values for classification on tail accuracies was also carried out. For instance, a threshold value $t = 0.5$ would take into consideration all predictions with accuracies greater than 0.5, forming the upper tail of the threshold bound. The mean accuracy is then taken over this tail for a given class, and then averaged over all classes for $TailAcc$. As can be seen in Figure 2, tail accuracy increases at greater t values.

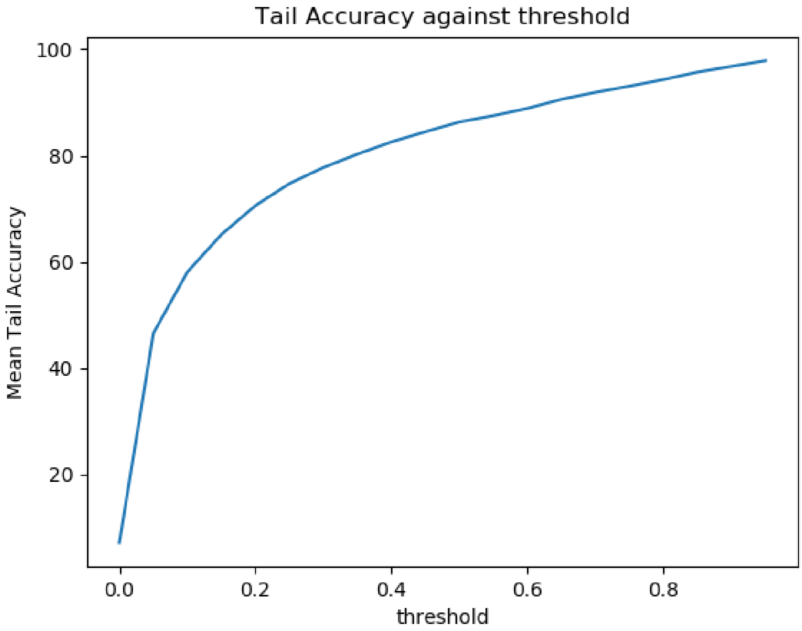

Figure 2: Tail Accuracy over Varying Threshold Values

References

[1] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International journal of computer vision, 111(1):98–136, 2015.

[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, IEEE, 6 2016.