Report on Architecture

groveh

April 2019

1 Motivation

The orientation estimation problem is at heart a convolutional neural network problem. Depending on the dataset, the input image is expected to be small due to the nature of this specific problem where the animal in question is cropped from the original image. DenseNet exhibits advantages over other modern networks through feature reuse, high information flow, and narrow layers. These all benefit a smaller input size as well as eliminate the need for learning redundant feature maps that were previously computed in previous layers of a dense block. In order to preserve computational efficiency, the current implementation has been designed after the DenseNet-BC concept. This version uses a bottleneck and a compression factor to reduce the size as well as number of feature maps.

2 Architecture

This implementation follows a very similar design to the DenseNet-BC paper implementation. The depth of the network can be specified, but is set at 100 currently. The growth rate is a variable defining the size of each feature map, labeled growth rate because of the concatenation of feature maps therefore causing the overall features to "grow" by this value (default set to 12). All but four layers (the initial convolution, two transition, and terminating fully connected layers) are split up evenly between the three dense blocks. Each layer within dense blocks are either single layers or bottlenecks. Single layers involves a batch normalization, a ReLU activation, and a 3x3 convolution. The feature map generated from each single layer is concatenated to those generated in previous layers. When the bottleneck option is set, a 1x1 convolution can be introduced before each 3x3 single layer convolution in order to reduce the size of the feature maps, therefore instead of dense blocks comprising of solely single layers, a bottleneck replaces every other single layer. If the compression option is set to a value less than 1, it acts as a multiplier to reduce the number of feature maps during each transition layer. Both bottleneck and a compression value are used in this implementation, following the DenseNet-BC model. In between each dense block, a transition layer is applied performing a batch normalization, ReLU, and 1x1 convolution followed by an average pooling with a

stride of 2. The final portion of the network after the third dense block has an average pooling followed by a fully connected layer and it's output activation. The output activation is dependent on whether single output regression or multi-class classification are being run. Classification performs a softmax followed by a log function, known as log softmax. Regression uses the identity function to output the value of the linear fully connected layer.

3 Training & Augmentation

The training process is relatively standard, using a negative loss likelihood gradient function for classification and a custom L1 loss for regression. The regression gradient function uses traditional L1 loss followed by an operation that ensures the following:

$$360 - |y - \hat{y}| \text{ if } |y - \hat{y}| > 180 \text{ else } |y - \hat{y}|$$

to counteract the wrap around problem which occurs when comparing angles such as 1 and 359 that are logically 2 degrees away but numerically 358 units away.
Data augmentation in this implementation takes in some image with or without an annotated angle. If a dataset does not include annotated angles, it is assumed that these images are already at their true orientation. If angles are provided, these images are rotated by this angle to their natural orientation. From here, various transforms are applied such as crops and resizes along with a random rotation that is used as the label for the image during training. The random angle is within a range dependent on the model being run, mainly the two cases of $[0, 360)$ or $[-45, 45]$ degrees.

For the case of $[-45, 45]$ degrees, labels are wrapped around in order to avoid having a value less than zero. To do this, the range is actually $[0, 45]$ and $[315, 360]$ degrees. This is essentially equivalent to the 360 degree model with the middle 270 degrees cut out. The same L1 error can be used, and the results are more promising than the full range model.

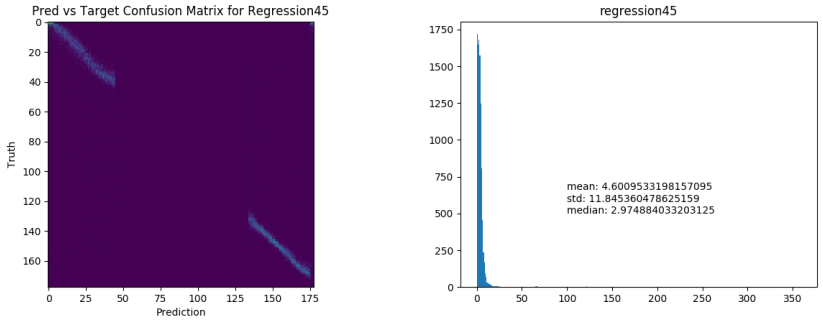
4 Results & Analysis

The results of the various networks were positive. The specifics will be discussed in each subsection. Each section contains graphs to visualize the error distribution, as well as a few examples of the predicted angle and what the image looks like when rotated by this prediction. The first figure is a confusion matrix of all truth and prediction pairs. A perfect guess runs along the negative diagonal of the graph, therefore the width of this line is important to see how far off the prediction was and in which way it went wrong. The second figure is a histogram of the errors between truth and prediction. On top of this graph are the mean, standard deviation, and median of the distribution.

The next set of figures shows a few of the results of the network. In each figure, the top row of numbers is the ground truth angle, while the second row is the angle inferred by the network during testing. The images below correspond to the images when rotated by the predicted angle.

4.1 Regression $\pm 45^\circ$

The first network is a single output regression network that assumes the input images is not rotated more than 45 degrees in either direction. Angles are kept positive during training and therefore preserve the negative diagonal correct prediction in the confusion matrix, but they are wrapped around at 180 because this network does not represent the full 360 angle spectrum.



(a) Regression ± 45 Confusion Matrix (b) Regression ± 45 Error Histogram

Figure 1: Error Distribution Visuals

Both graphs show good results. The confusion matrix follows the diagonal with a small width indicating small error in the prediction. The histogram shows that the distribution of error lies predominantly around zero. A large standard deviation indicates a few instances of predictions far from their expected angle.

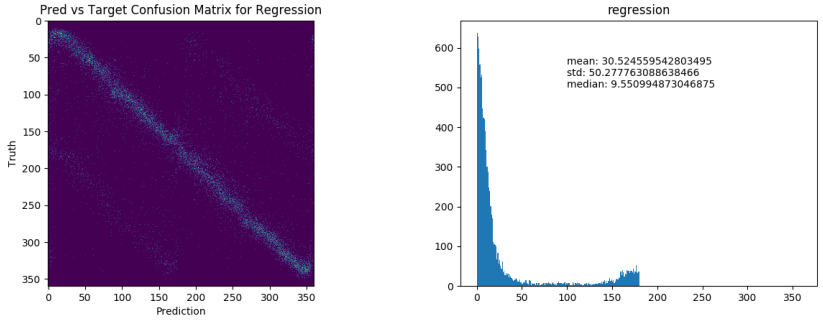
(a) Results Example 1 (b) Results Example 2

Figure 2: Results

As shown above, the predicted angles are relatively accurate and rotate the image very close to their true orientation.

4.2 Regression 360°

Next is the 360 degree regression network. This is similar to the previous network, however it does not assume that the image is within 45 degrees of its true orientation and therefore estimates angles within the whole 360 degree spectrum. Angles are positive utilizing the gradient function described earlier.



(a) Regression 360 Confusion Matrix (b) Regression 360 Error Histogram

Figure 3: Error Distribution Visuals

The confusion matrix above shows that the predictions are slightly more scattered than the 45 degree network, and this is also evident in the histogram. This is expected due to the much large set of possible angles, as well as various cases when the image is rotated past 45 degrees in either direction. These

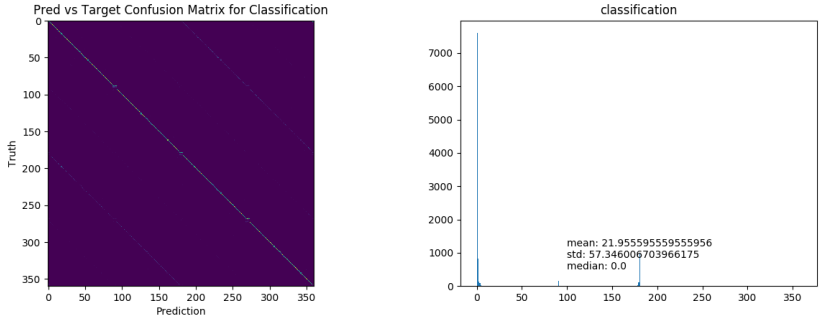
cases include symmetrical features that easily confuse the network, especially at multiples of 90 degrees.

(a) Results Example 1 (b) Results Example 2

Figure 4: Results

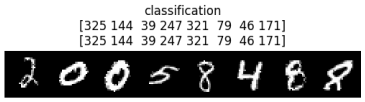
4.3 **Classification** 360°

Below are the full spectrum classification results as a point of comparison.

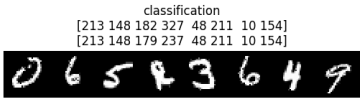


(a) Regression 360 Confusion Matrix (b) Regression 360 Error Histogram

Figure 5: Error Distribution Visuals



(a) Results Example 1



(b) Results Example 2

Figure 6: Results