Report for the Screw Classification Task

Majd Wardeh

February 2022

0.1 Data Preprocessing and Augmentation

By taking a closer look at the provided dataset, we can observe that the size of the images is not equal. We calculated the average dimensions of the images and it was equal to (h=94, w=334), where h and w are the height and width of the image, respectively. We considered (h=90, w=335) to be the size of the input images to our classifier. Also, since the shape of the screws is vital for the classification task, we decided to resize the images while keeping their aspect ratio unchanged. Therefore, we had to write our own resizing code that pads the images with either zeros or with the nearest pixels. Also, since the size of our training dataset is 1271 images only, we increased the number of images by writing a data generator that augments the images by applying:

1. Random horizontal and vertical flip with a probability of 0.5.
2. Random horizontal and vertical shift with the nearest pixel padding with a probability of 0.4.
3. Random zoom out and in with a probability of 1, and 5% zoom in and out.
4. Random shifting in the image brightness, which is represented by a Gaussian of zero mean and 0.12 standard deviation.

Applying random vertical flipping was the most valuable randomization technique for our dataset since we observed that some images have screws with different illumination/brightness values which were correlated with the left or right orientation. The horizontal flipping was also applied to generate images of screws with small rotations.

0.2 Network Architecture

We used Resnet50 [1] as a feature extractor for our network. We kept the convolution layers only and dropped the dense layers. The output features of the Resnet were flattened and fed as input to a fully-connected dense network. The dense network consists of 2 hidden layers of 1000 and 500 neurons with a ReLu activation function respectively. The two hidden layers were preceded with dropout layers of rate 0.3. The last hidden layer is followed by the output layer which has a single neuron with a Sigmoid activation function. We froze the weights of the Resnet layers and optimized the weights of the dense layers only using Adam optimizer [2].

0.3 Training and Results

We split the provided training set into training and validation sets with a ratio of 80 and 20 percent respectively. We trained for 3000 epochs to make sure we encountered most of the random samples from the data generator.

We noted that there is a small imbalance in the provided training set. The number of samples for class 1 is 592, compared with 679 for class 2. We considered this imbalance in samples by assigning a weight of $679/(592+697)$ for class 1 and $592/(592+697)$ for class 2 during training.

The achieved accuracies across the datasets are (rounded to 4 decimal places):
Train set: 0.9970
Validation set: 1.0000
Test set: 0.9906

0.4 Running the code

Please refer to the provided README file.

Bibliography

[1] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[2] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).