Report by Danil Ginzburg

BS17-DS-01

Transforming images to the same size

The transformation is done by padding image to obtain square shape and then resizing it to particular state (typically it is 30 * 30). Padding is done via replication of neighbour pixels.

Here is an example of rectangular image (182, 67, 3) and its transformation to (30, 30, 3):

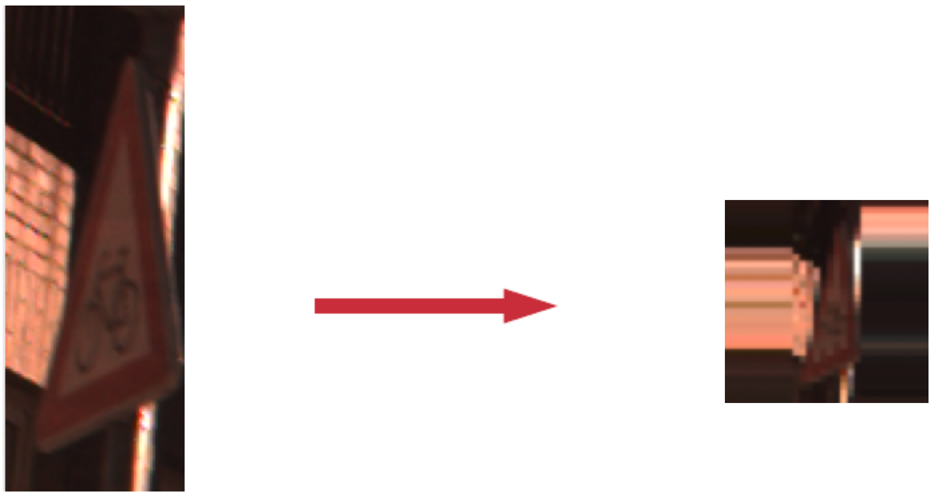
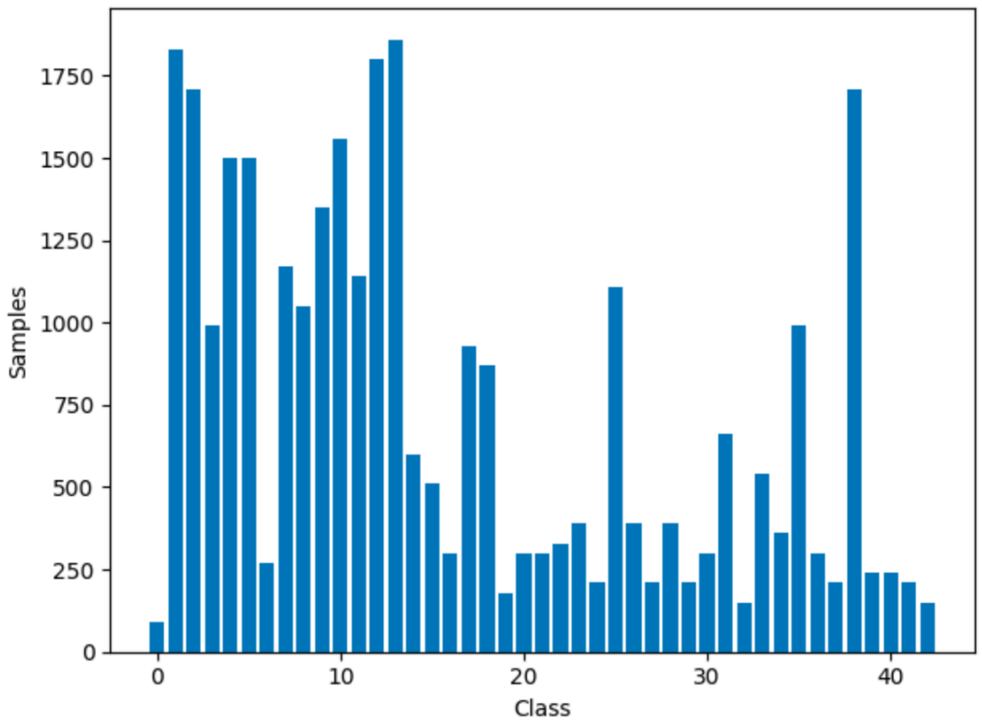


Image frequencies

After splitting data into training and validation sets, 80% of images went to training set and the rest went to validation set. Here is bar representing the number of samples for each class:

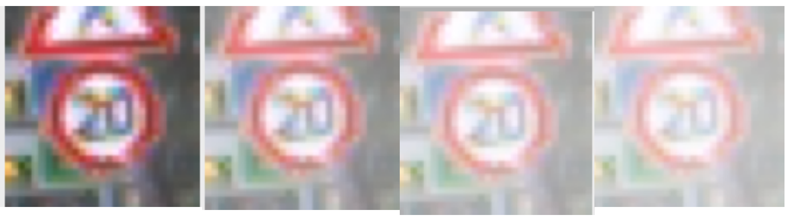


Augmentation

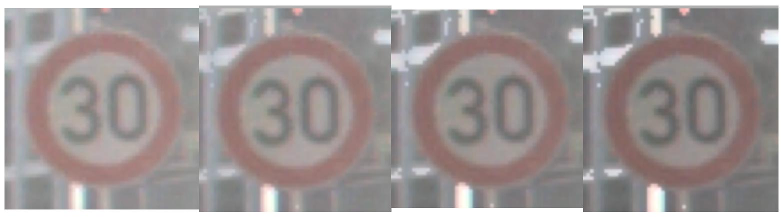
In order to equalise the number of samples per class we are using Augmentation. Here is two types of augmentation I applied for images: **Add light** and **Contrast**. Since all signs are already centralised and since a flipped sign is not a sign anymore, thus such techniques as rotate, flip, etc. are meaningless. Therefore, I find Add light and Contrast the most useful augmentation techniques.

Here are the examples of them:

I. **Add light**. Images— Original; gamma=2; gamma=3; gamma=4.



II. **Contrast image**. Images — Original; contrast=10; contrast=20; contrast=30.



Evaluation

On the right side is the result of calling *classification_report* function from *sklearn.metrics*.

```
clf = RandomForestClassifier(n_estimators=30, max_depth=50)
clf.fit(X, y)
y_pred = clf.predict(X_test)
print(classification_report(y_test, y_pred))
```

On the left side we see the number of class (0-42) and its precision, recall, f1-score, etc.

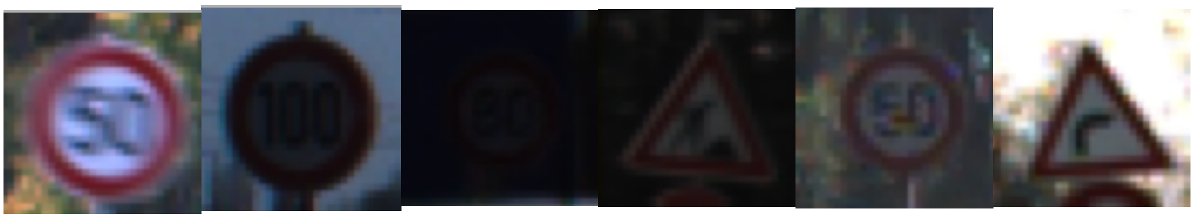
The overall accuracy is appeared to be **75%** with image shape = (30, 30) and on augmented data. Although it can be increased (see «Experiments» subtitle below).

```
              precision    recall  f1-score   support

           0       0.52      0.58      0.55        60
           1       0.67      0.64      0.66       450
          10       0.86      0.93      0.89       390
          11       0.86      0.82      0.84       360
          12       0.98      0.90      0.94       630
          13       0.92      0.83      0.87       360
          14       0.90      0.98      0.94       150
          15       0.66      0.78      0.72        60
          16       0.57      0.27      0.36        30
          17       0.97      0.98      0.98       120
          18       0.77      0.78      0.78       270
          19       0.87      0.66      0.75        90
           2       0.55      0.72      0.62       360
          20       0.43      0.65      0.52        60
          21       0.00      0.00      0.00         0
          22       0.79      0.92      0.85        90
          23       0.81      0.45      0.58       210
          24       0.48      0.50      0.49        30
          25       0.91      0.81      0.86       360
          26       0.46      0.52      0.48        60
          27       0.60      0.90      0.72        30
          28       0.87      0.78      0.82       150
          29       0.77      0.60      0.68       120
           3       0.65      0.63      0.64       390
          30       0.63      0.52      0.57       120
          31       0.50      0.70      0.58        90
          32       0.52      0.20      0.29        60
          33       0.97      0.99      0.98       149
          34       0.83      0.97      0.89        91
          35       0.98      0.91      0.95       239
          36       0.86      0.98      0.92        91
          37       1.00      0.92      0.96        89
          38       0.97      0.95      0.96       481
          39       0.00      0.00      0.00         0
           4       0.72      0.75      0.73       390
          40       0.73      1.00      0.85        30
          41       1.00      0.60      0.75        60
          42       0.00      0.00      0.00         0
           5       0.39      0.39      0.39       360
           6       0.54      0.93      0.68        90
           7       0.40      0.47      0.43       210
           8       0.66      0.62      0.64       390
           9       0.97      0.70      0.81       240

    accuracy                           0.75      8010
   macro avg       0.69      0.68      0.67      8010
weighted avg       0.78      0.75      0.76      8010
```

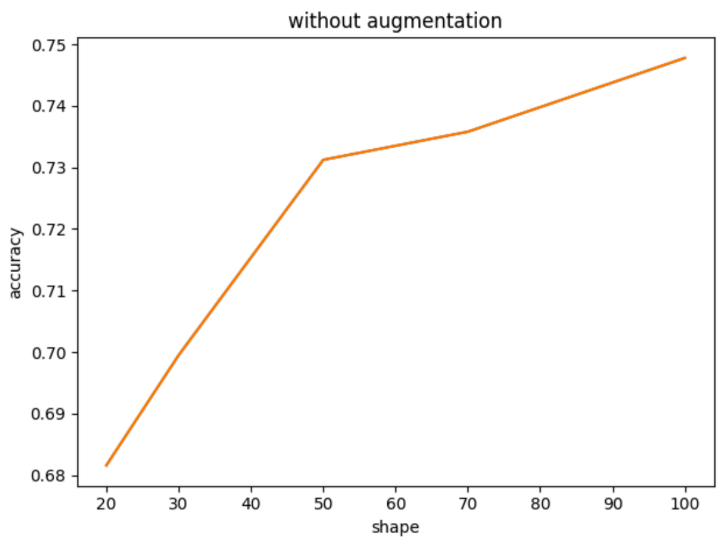
Some images has been misclassified and here are some examples of those:

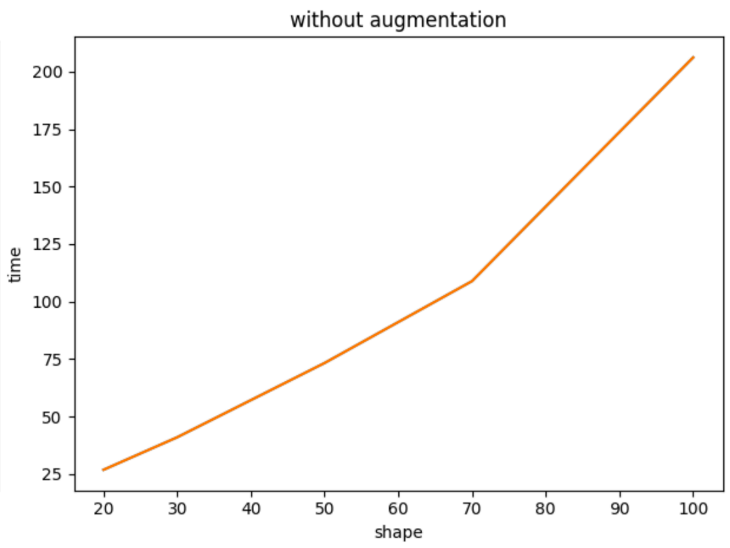


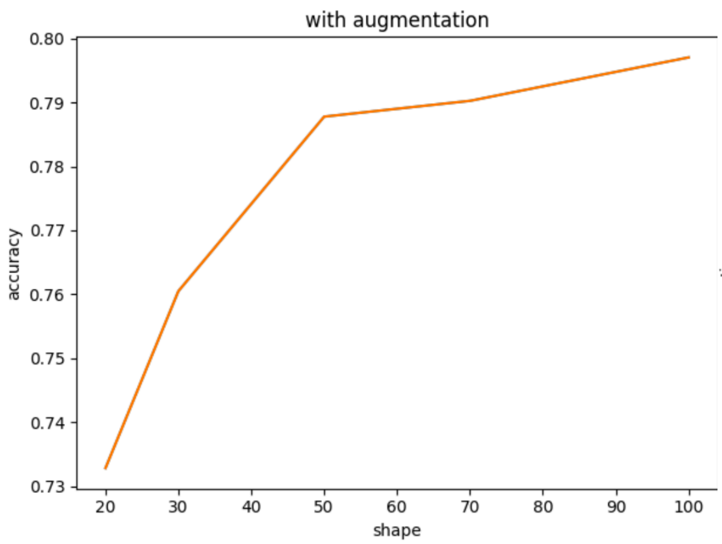
The obvious reason they have been misclassified is because they are blurred or too dark.

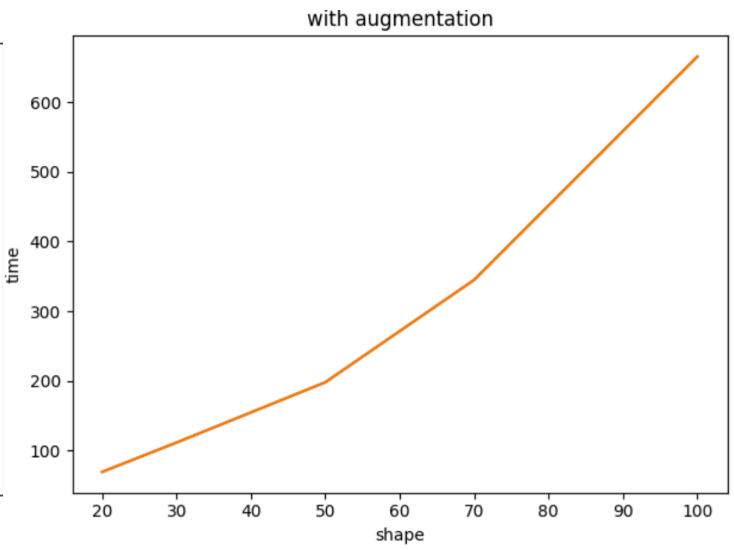
Experiments

Here is result plots of doing training on **non augmented** data with different shapes - (20, 20); (30, 30); (50, 50); (70, 70); (100, 100):









And here is result plots of doing the same training, but on **augmented** data:
As we observe, the larger the size, the better the accuracy. So choosing appropriate size is only about tradeoff between time to train and accuracy.

Augmentation takes time itself, and training on augmentation data is much longer (in this case 3 times longer in general). However, it definitely worth it, since we managed to increase accuracy by 5% in general.