Google ML Bootcamp Competition 2023 for Ukrainians

Yoga poses detector

by Dmytro Kainara

01 Introduction

General information

Yoga has gained popularity in the past couple of decades and because of its success in making people physically and mentally fit, it has become widely popular all over the world. Especially in the last 2 years after the pandemic hit the world, people have been spending most of their time in their homes which opens up more suitable conditions and possibilities of practicing yoga.
However, it is very important to stretch the body correctly in every asana as each yoga pose targets a specific muscle of your body and the problem with yoga is that, just like any other exercise, it is of utmost importance to practice it correctly as any incorrect posture during a yoga session can be unproductive and possibly detrimental.
Human pose estimation is a well-known problem in computer vision to locate joint positions. The application of pose estimation for yoga is challenging as it involves a complex configuration of postures. Furthermore, some state-of-the-art methods fail to perform well when the asana involves horizontal body posture or when both the legs overlap each other or any similar complex pose.

Main Links

[Kaggle competition]

[Jupyter notebook]

[Data information]

02 Data information

General information about dataset

Numbers of images

2360 pictures

Number of classes

6

General information about dataset

Information about

Yoga poses on photos

Source of data

Links

Classes of images (poses)

Class 0. Standing

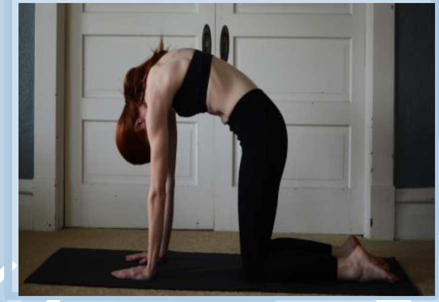
Class 1. Sitting

Class 2. Balancing poses

Class 3. Inverted poses

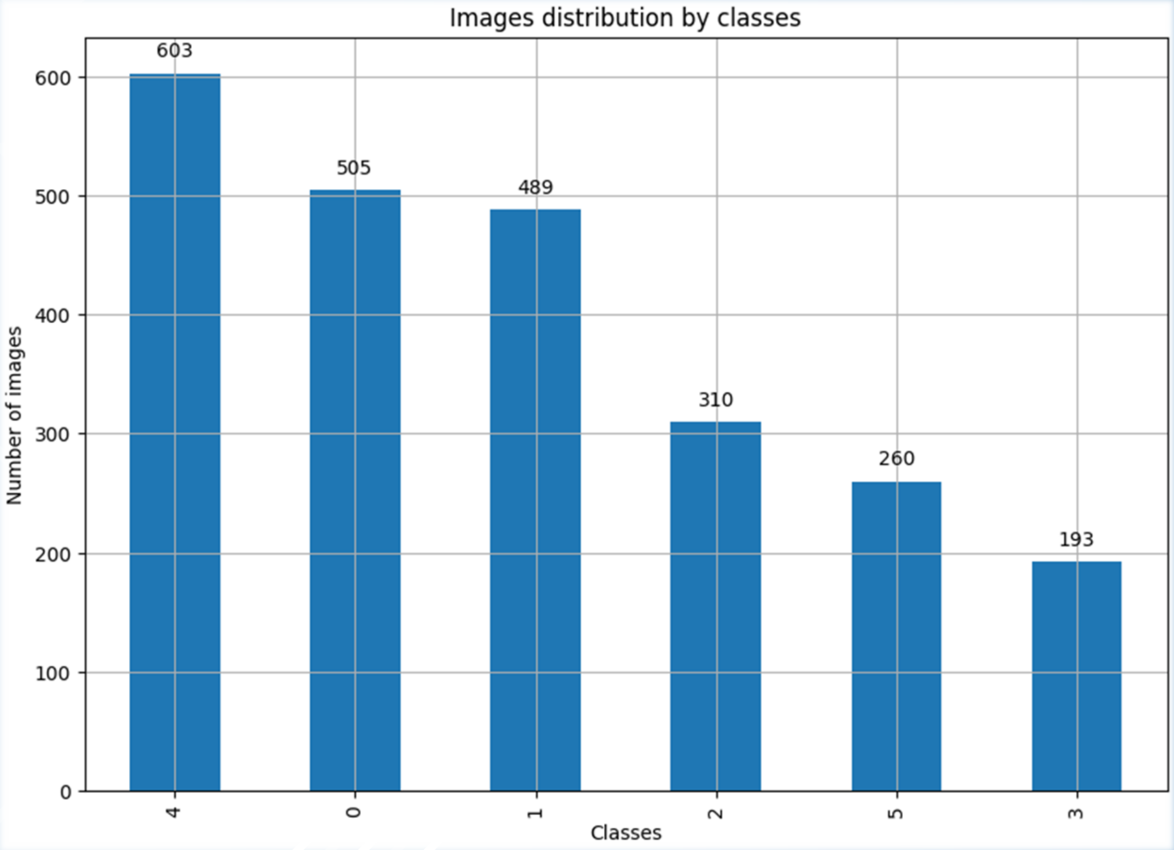
Class 4. reclining (cobra, dog, plank poses)

Class 5. Wheel



Google

Images distribution



Conclusion

The images distribution is **imbalanced**.

Therefore, we should:
- Use F1-score to evaluate the quality of the model
- Use stratification for balancing model training

Conclusion about quality of the dataset

01 Imbalanced dataset

02 Small dataset (only 2360 images)

03 Random arrangement of objects in the image

04 Presence of false objects, e.g.

05 The presence of several people in the picture

06 There are many subclasses of poses within a single class, e.g., image class 4

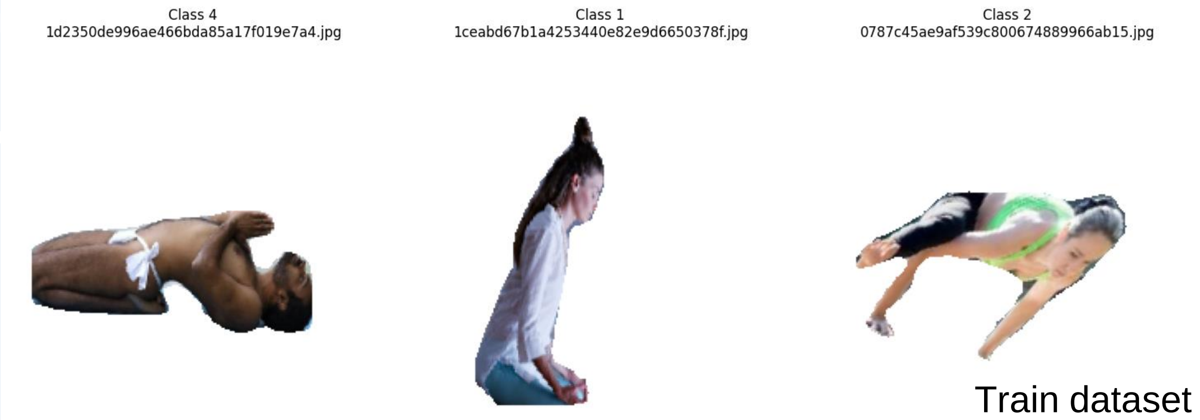
03 Data preprocessing

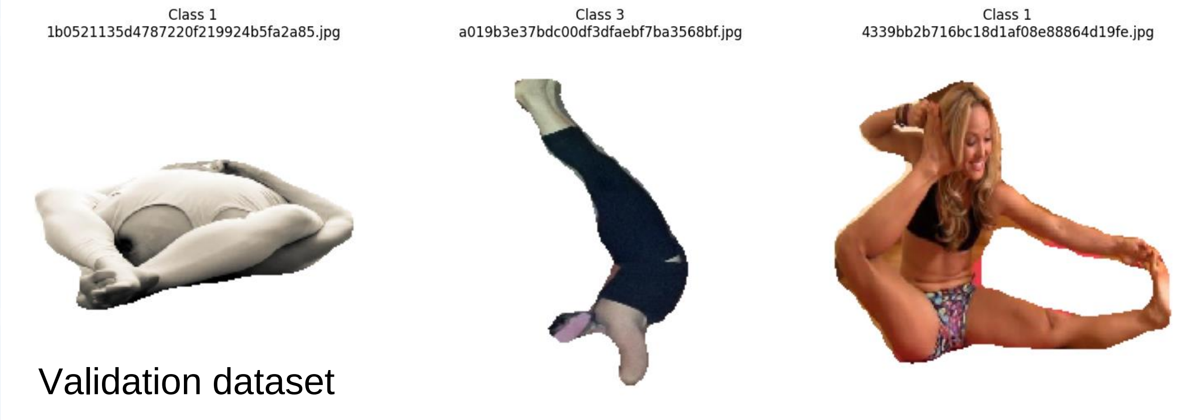
Conclusion about quality of the dataset

Resolve the problem with dataset

- Stratification
  - Imbalanced dataset
- Use the mask Detectron2
  - The presence of several people in the picture
  - Subclasses of poses within a single class
  - Presence of false objects
- Saturn is not solid like Earth, but is a gas giant
  - Random arrangement of objects in the image

The result of preprocessing stage





Conclusion

The Detectron2 and Augmentation work correct. The images change in the right way

04 Model

Google

The structure of the model

The dataset is too small for training our own model (only 2360 images). Therefore, it's necessary to use pretrained models.
The idea of the model is to create an ensemble of models from EfficientNet, MobileNet_v3, ResNet, and DenseNet201.

The results of training

```
Epoch 1/60
15/15 [==============================] - 197s 13s/step - loss: 0.8352 - f1_score: 0.7160 - val_loss: 0.9185 - val_f1_score: 0.6820
Epoch 2/60
15/15 [==============================] - 184s 12s/step - loss: 0.8422 - f1_score: 0.7076 - val_loss: 0.8741 - val_f1_score: 0.6941
Epoch 3/60
15/15 [==============================] - 192s 13s/step - loss: 0.7990 - f1_score: 0.7224 - val_loss: 0.9369 - val_f1_score: 0.6861
Epoch 4/60
15/15 [==============================] - 183s 12s/step - loss: 0.8150 - f1_score: 0.7115 - val_loss: 1.0138 - val_f1_score: 0.6539
Epoch 5/60
15/15 [==============================] - 182s 12s/step - loss: 0.8114 - f1_score: 0.7220 - val_loss: 1.0000 - val_f1_score: 0.6552
Epoch 6/60
15/15 [==============================] - 192s 13s/step - loss: 0.8049 - f1_score: 0.7373 - val_loss: 0.9186 - val_f1_score: 0.6754
Epoch 7/60
15/15 [==============================] - 182s 12s/step - loss: 0.7650 - f1_score: 0.7401 - val_loss: 1.0086 - val_f1_score: 0.6648
Epoch 8/60
15/15 [==============================] - 182s 12s/step - loss: 0.7239 - f1_score: 0.7672 - val_loss: 0.9376 - val_f1_score: 0.6667
Epoch 9/60
15/15 [==============================] - 194s 13s/step - loss: 0.7410 - f1_score: 0.7514 - val_loss: 0.9184 - val_f1_score: 0.6896
Epoch 10/60
15/15 [==============================] - 193s 13s/step - loss: 0.7494 - f1_score: 0.7502 - val_loss: 0.9402 - val_f1_score: 0.6674
Epoch 11/60
15/15 [==============================] - ETA: 0s - loss: 0.7550 - f1_score: 0.7439 Restoring model weights from the end of the best epoch: 1.
15/15 [==============================] - 184s 12s/step - loss: 0.7550 - f1_score: 0.7439 - val_loss: 0.8850 - val_f1_score: 0.6923
Epoch 11: early stopping
```

Google

Thank You for your attention!

D. Kainara

Ph.D. Economics, Data scientist