COL 341: Assignment 3

Notes:

- This assignment is purely competitive and you may use any machine learning method (including deep learning, convolutional neural networks etc.) and publicly available code to get the best possible performance.
- This assignment will have four submissions on **28/10/2021; 4/11/2012; 11/11/2021** and **26/11/2021** before 9:00pm. For each submission you can improve your results from the previous submission. However, if you are unable to improve your model performance in a week, you can resubmit the code of the previous week.
- Include a detailed report which should contain a detailed description explaining what you did in every week. Include any observations and/or results for your intermediate results in your report.
- You should use Python for all your programming solutions.
- Your assignments will be auto-graded on Kaggle and then on IIT servers, make sure you test your programs before submitting. We will use your code to train the model on training data and predict on test set.
- Input/output format, submission format and other details are included. Your programs should be modular enough to accept specified parameters.
- You should submit work of your own team only. You should cite all the public domain sources used by you in your report. If you use any external code/resource without suitably citing it in your report, it will be considered plagiarism and you will be awarded D or F grade along with disciplinary action if needed.
- If it is seen that someone is cheating/copying others work, or hard-coding the labels in the notebooks, it will be considered a violation of the honor code you will be awarded D or F grade along with disciplinary action if needed.

1. **Yoga pose image classification (Due date: 26th November, 2021)**
   Human activity recognition is one of important problems in computer vision. It includes accurate identification of activity being performed by a person or group of people. In this assignment, we will be classifying yoga poses into different classes. We have around 19 types of Asanas in our dataset and 29K images for training a machine learning model. Yoga pose estimation has multiple applications such as in creating a mobile application for yoga trainer.
   This assignment will be performed on Kaggle. This is the link to the competition.

   About Competition:

   (a) The competition is to be done in **groups of maximum two students**. You are expected to create a group on kaggle. Fill this google form, where you will have to report your team name on kaggle, and the team members. No team with more than 2 members, will be considered at the time of final evaluation.

   (b) Each team will be evaluated weekly. Thus, you are expected to save your kaggle notebook before each submission deadline. For more info on saving your notebook check out this link.

   (c) You can use any **publicly available** framework/software/library. Make sure that either the library is present in kaggle or you put the necessary commands to install the library in your notebook only.

Submission instructions:

(a) Make sure you save your notebook's latest version. It should be named as **TEAMNAME_COL341_A3.ipynb**. On submission, your notebook is run automatically by kaggle. Thus you are expected to output *submission.csv* on Kaggle and save your model on Moodle. At the end of each week, we will publish everyone's relative rank on that hidden set in this google sheet.

(b) At the end of competition you are expected to write a report which includes your approach to tackle the problem. Mention all hyperparameters. If you have used any publicly available code, do mention that too. **Although in no case, you can use any other student's code**. This report has to be submitted on gradscope at the end of competition.

(c) At the end of each sub-deadline, you are expected to submit train_entrynumer1_entrynumber2.py, test_entrynumer1_entrynumber2.py and model object file on Moodle.
Format for training script shall be:

```
python train_entrynumber1_entrynumber2.py
--trainfolder <path to training file>
--outputfolder <path to save model weights to>
```

For example:

```
python train_2017CS50411_2020MCS2475.py
--trainfolder ./train.csv
--outputfolder ./checkpoints/
```

(d) We will be running your notebook against a hidden test set and use your model weights which was saved on kaggle. Thus you are required to **submit a testing script on moodle** (any of the team member can do). The script should be named *test_entrynumber1_entrynumber2.py* and it should take model weights path as input and create a file named *submission.csv*. Thus,

```
python test_entrynumer1_entrynumber2.py
--modelpath <path to weights folder>
--testinput <path to testfile>
--testoutput <path to test output>
```

would be the format for running the testing script. For ex,

```
python test_2017CS50411_2020MCS2475.py
--modelpath ./checkpoints
--testinput ./test.csv
--testoutput ./submission.csv
```

Summarizing, each team is expected to submit 3 things:

(a) Submit report.pdf on Gradescope.

(b) Save best version of Kaggle notebook before each deadline.

(c) Submit train script, testing script and model object folder on Moodle.

Kaggle specific instructions:

(a) For evaluating your submission, you can submit your predictions for public learderboard, 10 times a day. To avoid any plagiarism, we will re-run your testing script with model weights on our hidden test case.

(b) On Kaggle, when you want to submit any prediction, your notebook is run automatically and it is expected that you write a file named *submission.csv* in the home directory. If there are any errors on running your submission, you wont be able to see any score.

(c) The submission notebook should be strictly made private. Making it public will give result in a penalty.

(d) For any doubts regarding this competition, you can use the discussions forum on kaggle itself.

Baseline Approaches:

(a) One approach could be to firstly use a pose estimation algorithm, such as Alphapose, Detectron2, Facebook AI, or MoveNet and then use those key points as an input to classifier such as Random Forests. You can use pretrained model weights as well. You can also refer to the tutorial notebooks mentioned on their repository/paper.

(b) Another approach can be to just use any CNN model (such as ResNet, DenseNet, InceptionV3 etc) to classify the images directly into the classes.

Tutorials and Extra Readings:

(a) Transfer Learning and fine tuning tensorflow models

(b) PoseNet paper

(c) Blog post on MoveNet

(d) Papers with Code - Pose estimation

(e) How to use Detectron2

(f) 3D Human pose estimation in Vietnamese traditional martial art videos