Instructions for experimentation

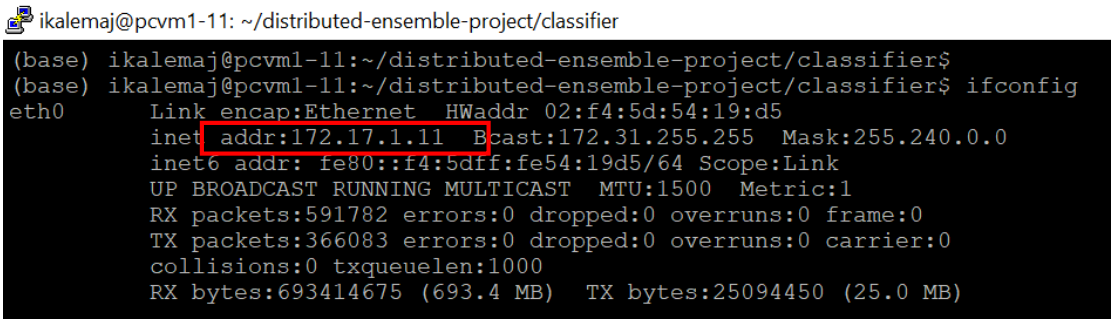
After having downloaded the correct environment and scripts, follow the instructions below to run your own experiments with the system.

Logging in

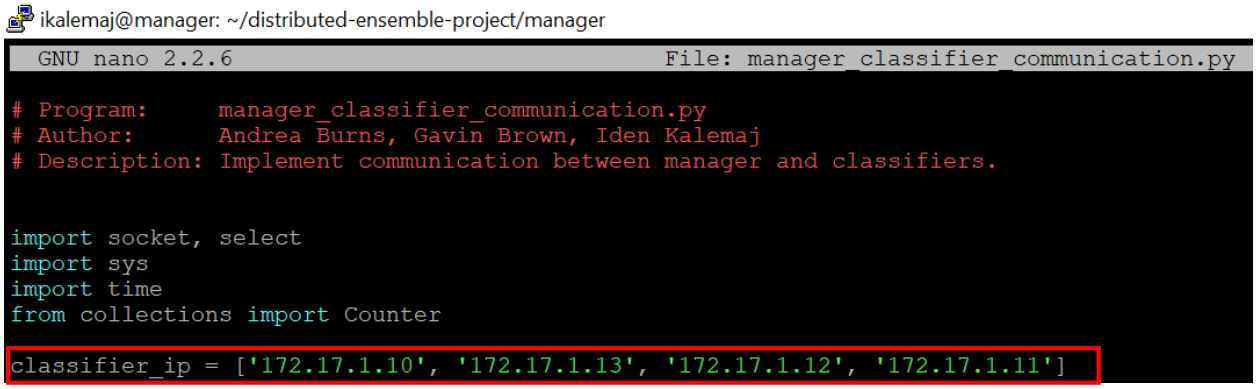
- SSH to the client node, manager node, and all the classifiers.
- For each node, switch to the project directory by typing “cd distributed-ensemble-project”
- For each node, switch to the respective folder, typing “cd client”, “cd manager”, “cd classifier” in the client, manager, and classifiers respectively.

Determining IP addresses:

- For each classifier and manager determine the IP address. You can do so by running ifconfig on each node.



- Change the IP addresses accordingly in the scripts:
    - In the manager, open “manager_classifier_communication.py” and place the correct classifier IP addresses in the list **classifier_ip**.



    - In the client, open “client_image.py” and change the value of the **HOST** variable to the correct IP address of the manager.

Values for corruption and delays

- Decide on values of corruption probability, probabilistic delay mean, and fixed delays experienced by the classifiers.
- In the classifier, open "classifier_main.py" and place these values in the lists *error_probabilities, exp_delays, fixed_delays* situated at the top of the file.

```
error_probabilities = [0.0, 0.1, 0.2, 0.3]
exp_delays = [0.0, 0.1, 0.2, 0.3, 0.4]
fixed_delays = [0.1, 0.2]
```

- Decide on combinations of corruption and delays you want to test. Decide as well on the way you sample the sequence of images sent from the client to the manager.
- In the client, edit the file named experiments.csv.

ikalemaj@client: ~/distributed-ensemble-project/client

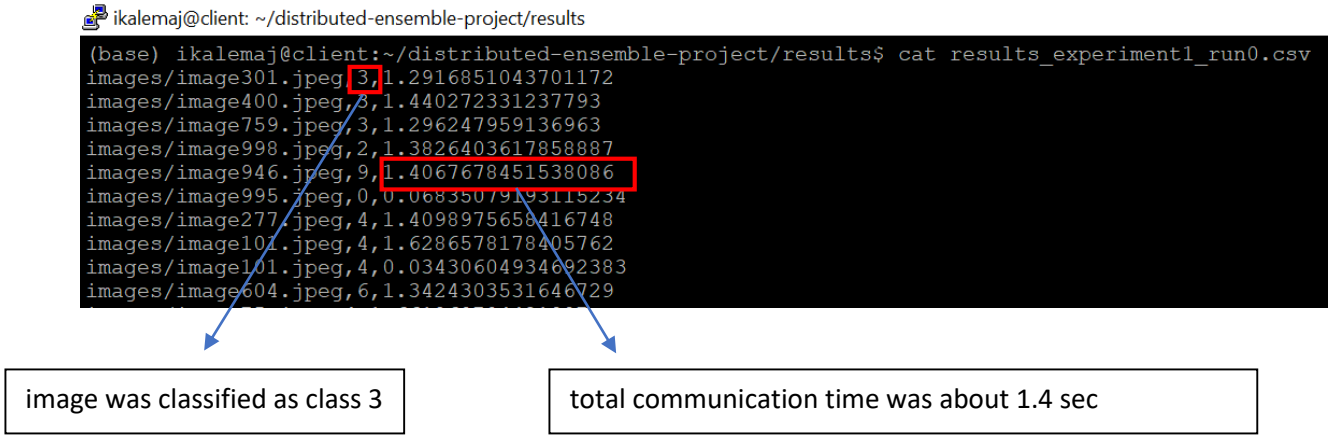
```
(base) ikalemaj@client:~/distributed-ensemble-project/client$ cat experiments.csv
error,delay,cache_size,sampling
0,0,30,power
1,0,30,power
2,0,30,power
3,0,30,power
0,0,30,uniform
1,0,30,uniform
2,0,30,uniform
3,0,30,uniform
```

Each line in the file corresponds to an experiment you want to run. The experiment parameters are as follows:

- The first item in the "error" column gives the corruption probability. Rather than the actual value, just indicate the index of this corruption probability in the list *error_probabilities* located in the classifier. E.g. error = 2 indicates that we are using the error probability at index 2, which, form the previous figure should be 0.2.
- The second item gives the mean of the exponential delay experienced by the classifier. Again, rather than indicating the actual value, indicate the index of that value in the exp_delays list. E.g. delay = 1 indicates that we are using the delay mean at index 1 which, form the previous figure should be 0.1.
- The third item indicates the cache size you are using. This is only for recordkeeping and analysis and will not actually affect the cache size – which has to be changed manually.
- The fourth item indicates the method by which the client samples the images to send to the manager from the "client/images" folder. This method is either "uniform" for uniform sampling, or "power" for sampling according to a power law, where some images are more likely than other to be sampled.

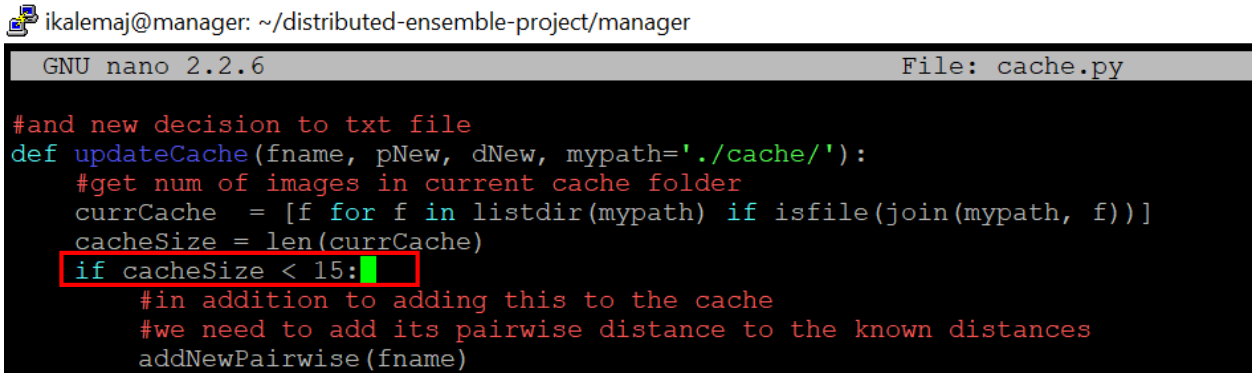
- Each experiment will have 3 runs and for each run the client sends 200 images selected according to the distribution you indicate.
- At the end of each run, a .csv file with results gets saved in the “results” folder. The file name contains an experiment number, which is the order of the experiment in the experiments.csv file, and a run number.

  Each line corresponds to an image sent, and indicates the name of the image sent, the classification received, and the total time it took from when communication with the manager was established until the classification was received.



Cache size

If you want to experiment with different cache sizes, in the manager, open “cache.py” and in the function updateCache, change the value in the line: if cacheSize < 15.



Running the experiments

You are now set to run the experiments. It is crucial to run these steps in the order indicated below, since communication channels need to be opened before communication takes place.

- For each classifier run: python classifier_main.py 8888
- In the manager run: python manager_image.py 8888
- In the client run: python client_image.py 8888

The port number is arbitrary but needs to be the same amongst all nodes.

As the experiments are running you will see messages printed in all the nodes, to indicate the communication flow and the messages being communicated.

```
(base) ikalemaj@client:~/distributed-ensemble-project/client$ python client_image.py 8889
answer = GOT SIZE
Sending ID: 0.0.0
answer = GOT ID
Sending image...
answer = Image is of class 3
answer = GOT SIZE
Sending ID: 0.0.0
answer = GOT ID
Sending image...
answer = Image is of class 0
answer = GOT SIZE
Sending ID: 0.0.0
answer = GOT ID
Sending image...
answer = Image is of class 3
```