

To download the data:

Navigate to the get_data folder.

create a virtual environment: `python -m venv get_data_venv`

Activate the get_data_venv: `source get_data_venv/bin/activate`

Install the requirements: `pip install -r requirements.txt`

Run the get_meta_data.py: `python3 get_collection_metadata.py`

Run the tcia_download_script.py:

For CBIS-DDSM

`python3 tcia_download_script.py --start`

`1.3.6.1.4.1.9590.100.1.2.117041576511324414842508325652101471266 --out /data/bl70/CBIS-DDSM CBIS-DDSM`

For CMMD:

`python3 tcia_download_script.py CMMD --out /data/bl70/CMMD --start`

`1.3.6.1.4.1.14519.5.2.1.1239.1759.332861114452774771680160736748`

deactivate the environment: `deactivate`

To process the data:

Navigate to data_processing folder

Note – to process different datasets, comment out one datasets path eg cmmd_path or cbis_ddsm_path at line 166

and the corresponding host_output_dir at line 176.

```
#cbis_ddsm_path = "/data/bl70/CBIS-DDSM/CBIS-DDSM"
```

```
cmmd_path = "/data/bl70/CMMD/CMMD"
```

```
# verify_path(cbis_ddsm_path)
```

```
verify_path(cmmd_path)
noise_levels = [0.01, 0.1, 1.0]
clip_limits = [2.0, 3.0, 5.0]
tile_grid_sizes = [(8, 8), (16, 16)]

#host_output_dir_base = os.path.expanduser("/data/bl70/CBIS-DDSM/CBIS-DDSM/processed")
host_output_dir_base = os.path.expanduser("/data/bl70/CMMD/CMMD/processed")
#processed_list_path = os.path.expanduser("/data/bl70/CBIS-DDSM/CBIS-DDSM/processed/processed.txt")
processed_list_path = os.path.expanduser("/data/bl70/CMMD/CMMD/processed/processed.txt")
```

Ensure to save.

create the environment: `python -m data_processing_venv`

Activate the venv: `source data_processing_venv/bin/activate`

Install the requirements: `pip install -r requirements.txt`

Run the data_processing.py: `python3 data_processing.py`

Run data_augmentation.py: `python3 data_augmentation.py`

Again, you can comment or uncomment out the paths at line 119 to augment the desired dataset:

```
# base_path = "/data/bl70/CBIS-DDSM/CBIS-DDSM/processed"
base_path = "/data/bl70/CMMD/CMMD/processed"
# output_base_dir = "/data/bl70/CBIS-DDSM/CBIS-DDSM/augmented"
output_base_dir = "/data/bl70/CMMD/CMMD/augmented"
```

Deactivate

To Split the Data:

Navigate to data_prep

run split_ddsm: `python3 split_ddsm.py`

you will need to check the data split:

To find the number of images run:

`find /data/bl70/CBIS-DDSM/Training/Benign -type f | wc -l`

`find /data/bl70/CBIS-DDSM/Training/Malignant -type f | wc -l`

To run the duplication script:

set the `base_dir` to the minority class path on line 5

set the `target_count` to that of the majority class on line 6

Save

Run the `duplicate_minority_class.py`: `python3 duplicate_minority_class.py`

To run the final ShuffleNet and AlexNet hybrid load processed and augmented CBIS-DDSM data into a directory, navigate to the `shuffle_alex` folder, change the directories to suit your placements.

build the docker image:

`docker build -t breast-cancer-classifier:newversion .`

to run the docker image adjust the paths to suit your paths:

```
docker run --rm --gpus all \
```

```
--shm-size=5g \
```

```
-v $(pwd):/workspace \
```

```
-v /data/bl70/CBIS-DDSM/Training:/data/bl70/CBIS-DDSM/Training \
```

```
-v /data/bl70/CBIS-DDSM/Testing:/data/bl70/CBIS-DDSM/Testing \
```

```
-v /data/bl70/CBIS-DDSM/Data:/data/bl70/CBIS-DDSM/Data \
```

```
-v /data/bl70/CBIS-DDSM/Models:/data/bl70/CBIS-DDSM/Models \
```

```
-p 6669:6669 \
```

```
breast-cancer-classifier:newversion \
```

```
python3 /workspace/manual_optuna_Shuffle.py
```

To run inception_resnet_trial.py:

Navigate to inception_resnet_trials

update the paths

save

create the environment: `python -m venv inception_resnet_venv`

activate the venv: `source inception_resnet_venv`

install the requirements: `pip install -r requirements.txt`

Set the train directory to parent of the Benign and Malignant directory.

Set the output directories as you wish.

run the script: `nohup python3 inception_resnet_trial.py > trials.log 2>&1 &`

To run the `resnet_mob_inc_trials.py`:

navigate to `resnet_mob_inc_trials` folder

update the paths

save

create the environment: `python -m venv resnet_mob_inc_venv`

activate the environment: `source resnet_mob_inc_venv/bin/activate`

install the requirements: `pip install -r requirements.txt`

run the script: `nohup python3 resnet_mob_inc_trials.py > trials.log 2>&1 &`

To run `shuffle_resnet_trials.py`:

navigate to the `shuffle_resnet18_trials` folder

update the paths

save

create the environment: `python -m venv shuffle_resnet_venv`

activate the environment: `source shuffle_resnet_venv/bin/activate`

install the requirements: `pip install -r requirements.txt`

run: `nohup python3 shuffle_resnet18_trials.py > trials.log 2>&1 &`

To run `vision_model.py`:

navigate to `vision_model`

update the paths

save

create the environment: `python -m venv vision_model_venv`

activate the environment: `source vision_model_venv/bin/activate`

install the requirements: `pip install -r requirements.txt`

run: `nohup python3 vision_model.py > trials.log 2>&1 &`

to run `testing_loss.py`

navigate to testing_loss

update the paths

save

create the environment: `python -m venv testing_loss_venv`

activate the environment: `source testing_loss_venv/bin/activate`

run the script: `nohup python3 testing_loss.py > testing_loss.log 2>&1 &`

****Note** - Also tests batch sizes and seeds.

to run shuffle_alex_trials.py:

navigate to shuffle_alex_trials

update the paths

save

create the environment: `python -m venv shuffle_alex_venv`

activate the environment : `source shuffle_alex_venv/bin/activate`

install the requirements: `pip install -r requirements.txt`

run the script: `nohup python3 shuffle_alex_trials.py > trials.log 2>&1 &`