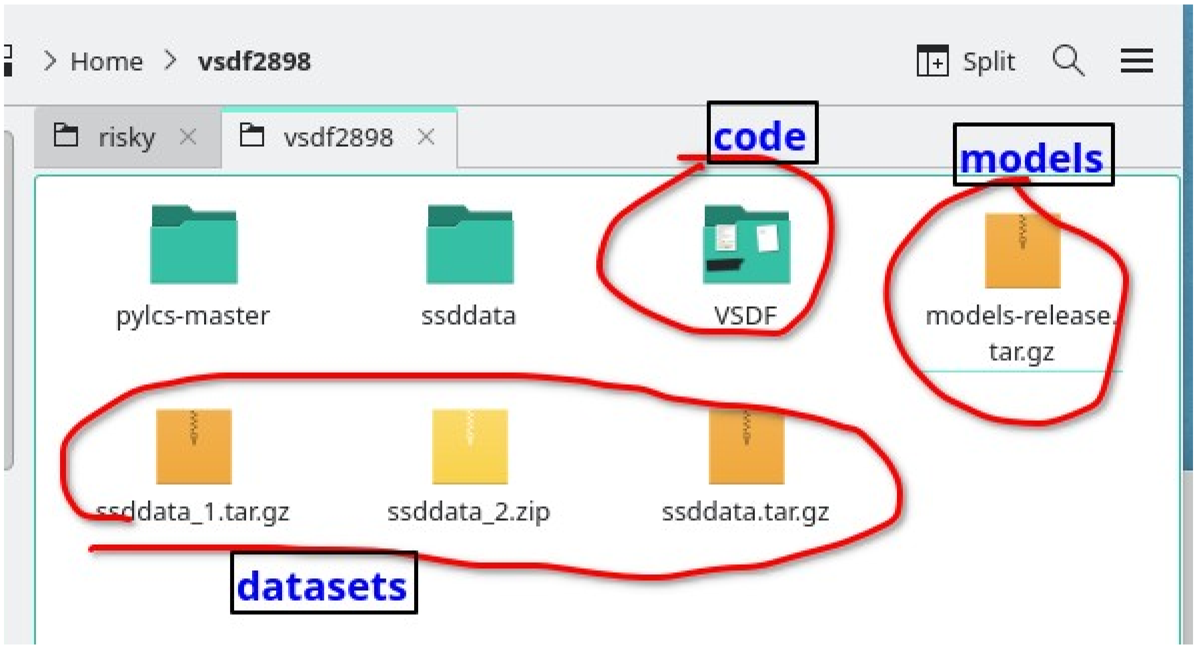
Manual

I. **Setting up environments.**

Step 1. Install a clean Manjaro Linux (Archlinux should do as well) and download code and data. You should have 3 data folders and 1 model folder from Kaggle (here we use the 4 archives we uploaded to Kaggle to save time) and a source code folder from Github.



Step 2. Configure mirror, update the system, install a building requirements and reboot via

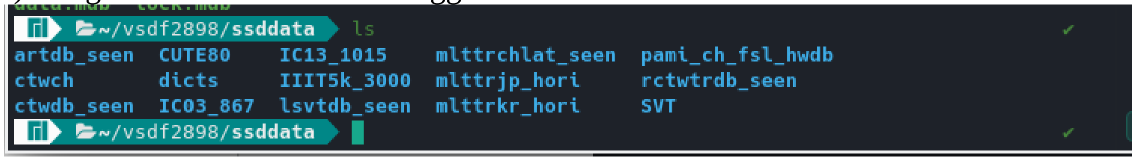
```
sudo pacman -Syyu pybind11; reboot
```

Step3A. Install PyTorch, Pycharm

```
sudo pacman -S pycharm-community-edition python-pytorch-cuda
```

Step3B. While waiting, grab the code, data, and the modified pylcs.

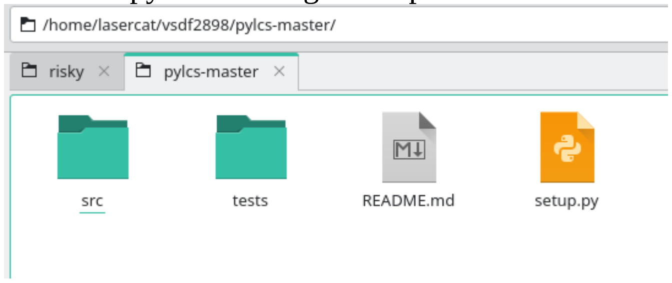
1) Merge the 3 data folders on Kaggle to the ssddata folder.



```
~/vsdf2898/ssddata  ls
artdb_seen  CUTE80    IC13_1015   mlttrchlat_seen  pami_ch_fsl_hwdb
ctwch       dicts     IIIT5k_3000 mlttrjp_hori     rctwtrdb_seen
ctwdb_seen  IC03_867  lsvtdb_seen mlttrkr_hori     SVT
~/vsdf2898/ssddata
```

2) While waiting, install the modded pylcs shipped in the repo.

a.Extract pylcs-master-getlcs.zip

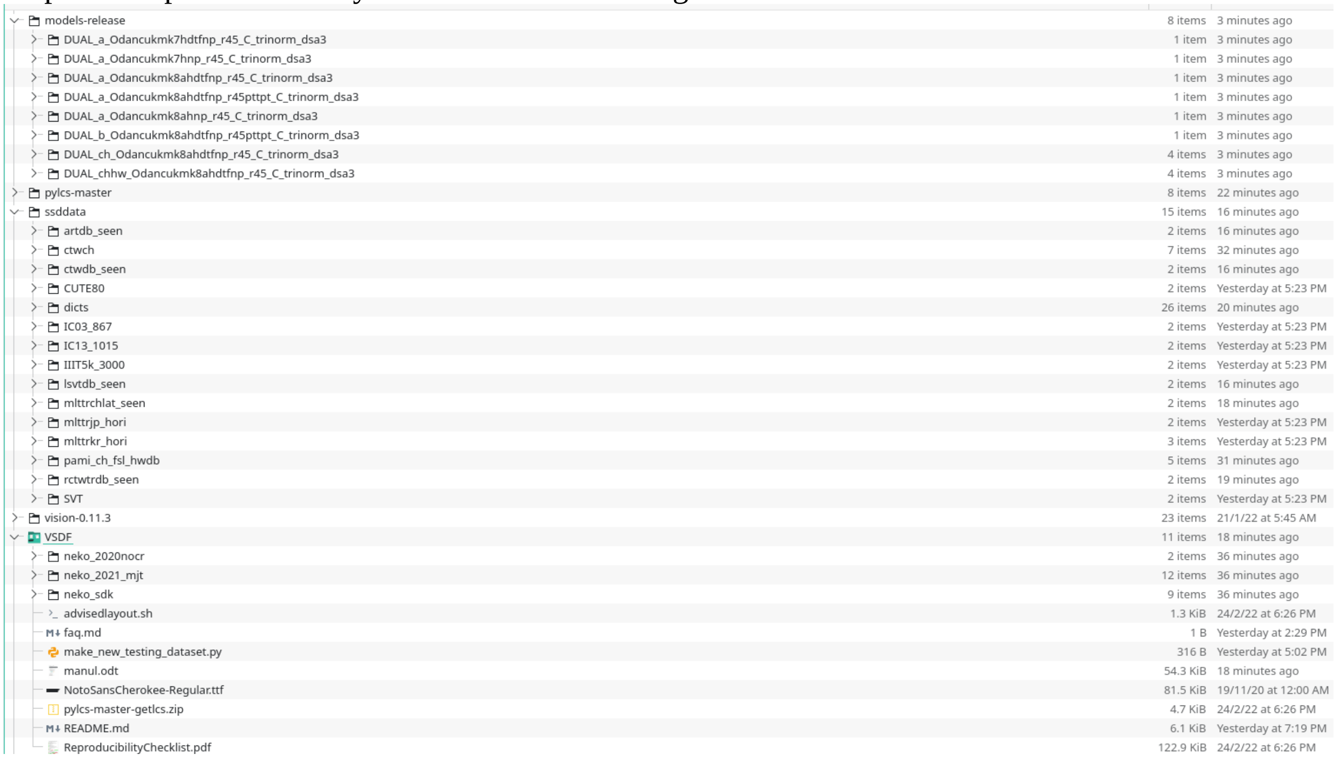


b. Run setup.py

```
python setup.py install --user
```

3) Extract released models into the models-release folder (Skip if it's already there)

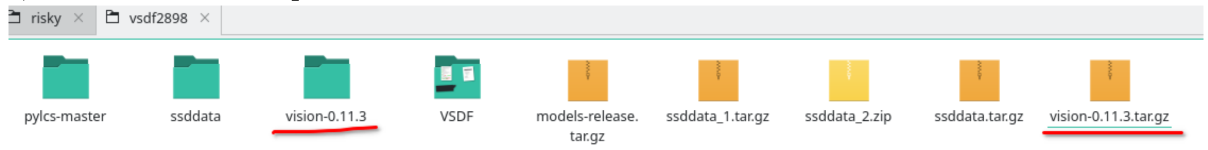
Step 3C. Stop and check if you have all the following folders



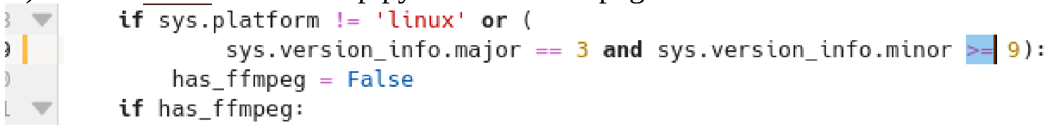
```
models-release                                          8 items   3 minutes ago
  DUAL_a_Odancukmk7hdtfnp_r45_C_trinorm_dsa3             1 item    3 minutes ago
  DUAL_a_Odancukmk7hnp_r45_C_trinorm_dsa3                1 item    3 minutes ago
  DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3            1 item    3 minutes ago
  DUAL_a_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3       1 item    3 minutes ago
  DUAL_a_Odancukmk8ahnp_r45_C_trinorm_dsa3               1 item    3 minutes ago
  DUAL_b_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3       1 item    3 minutes ago
  DUAL_ch_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3           4 items   3 minutes ago
  DUAL_chhw_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3         4 items   3 minutes ago
pylcs-master                                            8 items   22 minutes ago
ssddata                                                 15 items  16 minutes ago
  artdb_seen                                             2 items   16 minutes ago
  ctwch                                                  7 items   32 minutes ago
  ctwdb_seen                                             2 items   16 minutes ago
  CUTE80                                                 2 items   Yesterday at 5:23 PM
  dicts                                                  26 items  20 minutes ago
  IC03_867                                               2 items   Yesterday at 5:23 PM
  IC13_1015                                              2 items   Yesterday at 5:23 PM
  IIIT5k_3000                                            2 items   Yesterday at 5:23 PM
  lsvtdb_seen                                            2 items   16 minutes ago
  mlttrchlat_seen                                        2 items   18 minutes ago
  mlttrjp_hori                                           2 items   Yesterday at 5:23 PM
  mlttrkr_hori                                           3 items   Yesterday at 5:23 PM
  pami_ch_fsl_hwdb                                       5 items   31 minutes ago
  rctwtrdb_seen                                          2 items   19 minutes ago
  SVT                                                    2 items   Yesterday at 5:23 PM
vision-0.11.3                                           23 items  21/1/22 at 5:45 AM
VSDF                                                    11 items  18 minutes ago
  neko_2020nocr                                          2 items   36 minutes ago
  neko_2021_mjt                                          12 items  36 minutes ago
  neko_sdk                                               9 items   36 minutes ago
  advisedlayout.sh                                       1.3 KiB   24/2/22 at 6:26 PM
  faq.md                                                 1 B       Yesterday at 2:29 PM
  make_new_testing_dataset.py                            316 B     Yesterday at 5:02 PM
  manul.odt                                              54.3 KiB  18 minutes ago
  NotoSansCherokee-Regular.ttf                           81.5 KiB  19/11/20 at 12:00 AM
  pylcs-master-getlcs.zip                                4.7 KiB   24/2/22 at 6:26 PM
  README.md                                              6.1 KiB   Yesterday at 7:19 PM
  ReproducibilityChecklist.pdf                           122.9 KiB 24/2/22 at 6:26 PM
```

Step 4. Prepare and mod the torchvision code (as of 2022 03 09, it refuses to compile without disabling ffmpeg)

a) Download and unzip the torchvision



b) Mod line 359 of the setup.py to disable ffmpeg



```
    if sys.platform != 'linux' or (
            sys.version_info.major == 3 and sys.version_info.minor >= 9):
        has_ffmpeg = False
    if has_ffmpeg:
```

c) Build and install ffmpeg

python setup.py install --user

Step 5. Reboot your PC and Install other dependencies:

a) Via pacman

sudo pacman -S python-lmdb scipy python-opencv python-regex python-matplotlib

sudo pacman -S python-editdistance

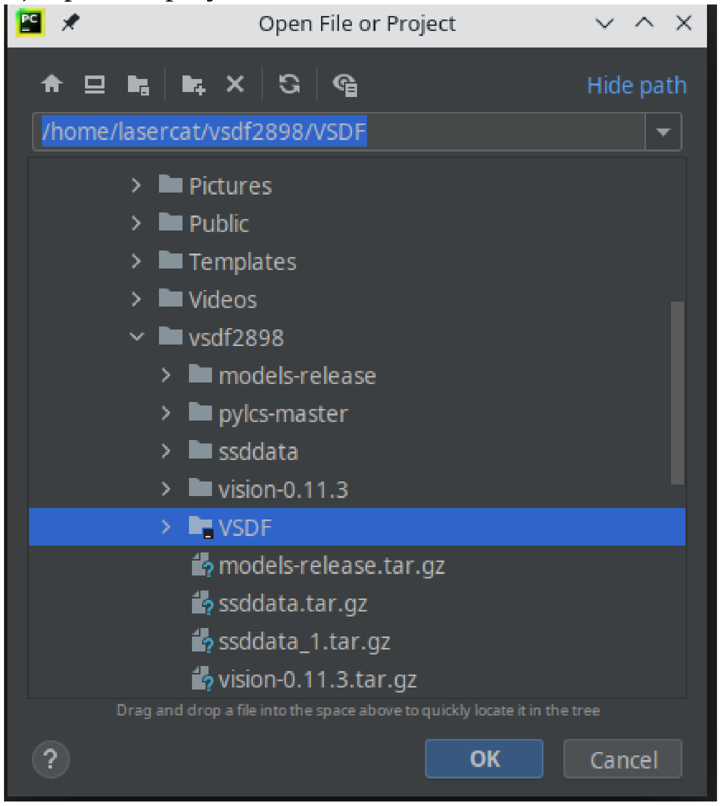
b) Via Pip

pip install torch-scatter

Note that you may have trouble building CUDA support of torch-scatter if you did not reboot

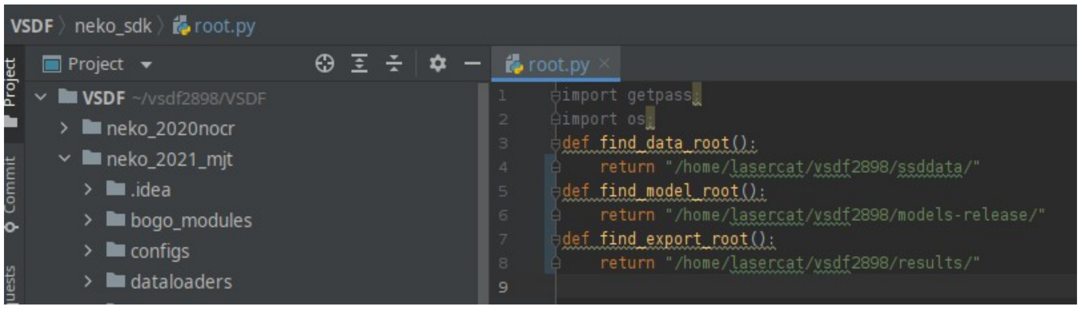
Step 6. Open the project in pycharm and configure pycharm for running.

a) Open the project at its root folder



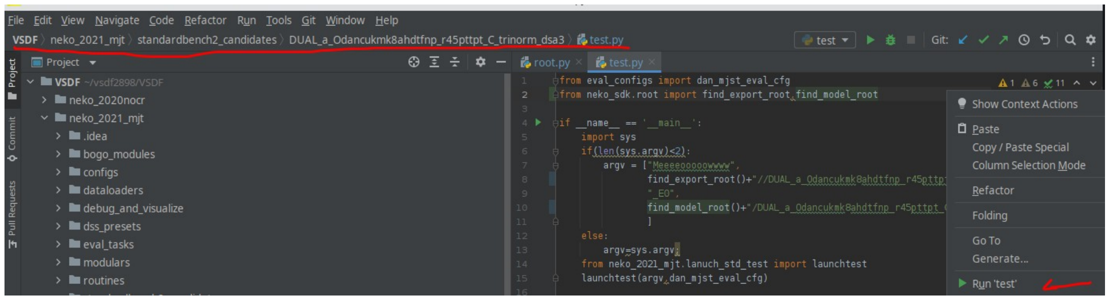
/home/lasercat/vsdf2898/VSDF

b) Set the paths in the $PROJ_ROOT/neko_sdk/roots.py



```python
import getpass
import os
def find_data_root():
    return "/home/lasercat/vsdf2898/ssddata/"
def find_model_root():
    return "/home/lasercat/vsdf2898/models-release/"
def find_export_root():
    return "/home/lasercat/vsdf2898/results/"
```

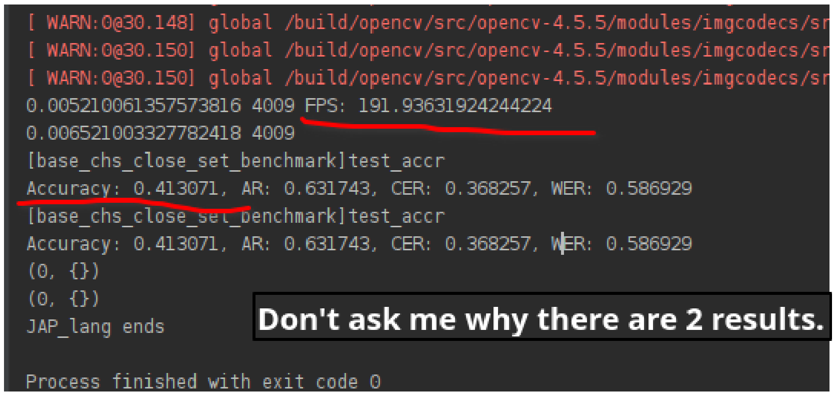
c) Run $PROJ_ROOT/neko_2021_mjt/standardbench2_candidates/DUAL_a_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3/test.py



```python
from eval_configs import dan_mjst_eval_cfg
from neko_sdk.root import find_export_root,find_model_root

if __name__ == '__main__':
    import sys
    if(len(sys.argv)<2):
        argv = ["Meeeeooooowww",
                find_export_root()+"//DUAL_a_Odancukmk8ahdtfnp_r45pttp
                "_E0",
                find_model_root()+"/DUAL_a_Odancukmk8ahdtfnp_r45pttpt_
                ]
    else:
        argv=sys.argv;
    from neko_2021_mjt.lanuch_std_test import launchtest
    launchtest(argv,dan_mjst_eval_cfg)
```

d) At the end of a bunch of warnings, you will get the Accuracy and FPS,

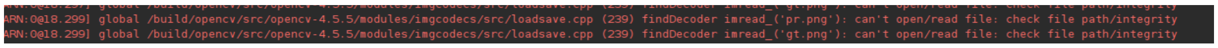


```
[ WARN:0@30.148] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
[ WARN:0@30.150] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
[ WARN:0@30.150] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
0.005210061357573816 4009 FPS: 191.93631924244224
0.006521003327782418 4009
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.413071, AR: 0.631743, CER: 0.368257, WER: 0.586929
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.413071, AR: 0.631743, CER: 0.368257, WER: 0.586929
(0, {})
(0, {})
JAP_lang ends

Process finished with exit code 0
```

The results will be dumped into your result path as well.

Note you should ignore these warnings ^_^



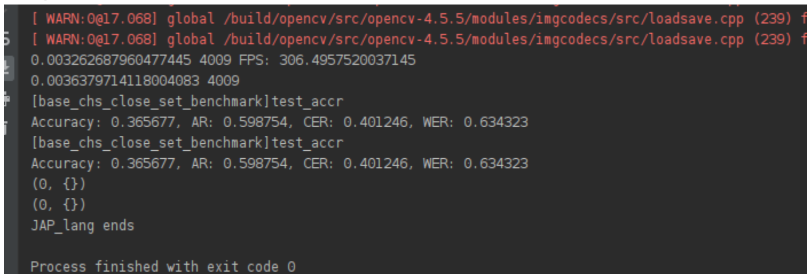
```
ARN:0@18.299] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('pr.png'): can't open/read file: check file path/integrity
ARN:0@18.299] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('gt.png'): can't open/read file: check file path/integrity
```

And if you get the accuracies correct, the environment is all good for ya.

II. **Evaluate downloaded models.**

Performance Benchmarks:
With the environments properly set, you are now able to reproduce most cells in the paper. The steps to run each experiment are shown in Section I and this section explains which experiment one should run to reproduce each individual result.

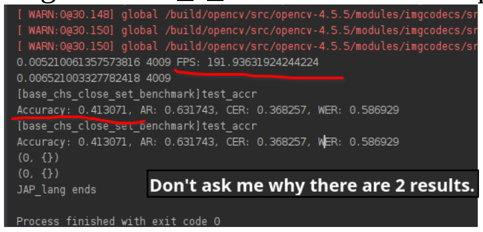
The open-set text recognition.
Regular=DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3



```
[ WARN:0@17.068] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) f
[ WARN:0@17.068] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) f
0.003262687960477445 4009 FPS: 306.4957520037145
0.0036379714118004083 4009
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.365677, AR: 0.598754, CER: 0.401246, WER: 0.634323
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.365677, AR: 0.598754, CER: 0.401246, WER: 0.634323
(0, {})
(0, {})
JAP_lang ends

Process finished with exit code 0
```

Large=DUAL_a_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3*



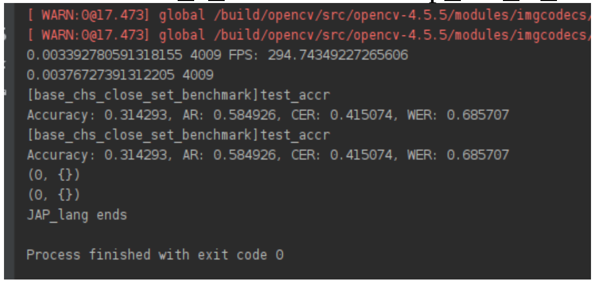
```
[ WARN:0@30.148] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
[ WARN:0@30.150] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
[ WARN:0@30.150] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/sr
0.005210061357573816 4009 FPS: 191.93631924244224
0.006521003327782418 4009
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.413071, AR: 0.631743, CER: 0.368257, WER: 0.586929
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.413071, AR: 0.631743, CER: 0.368257, WER: 0.586929
(0, {})
(0, {})
JAP_lang ends

Process finished with exit code 0
```

Don't ask me why there are 2 results.

Note a TPT module from OSOCR is loaded, but it is not used by the network due to … an implementation blunder – since it has nothing to do with the contribution or ablative studies of the paper, we left it be.

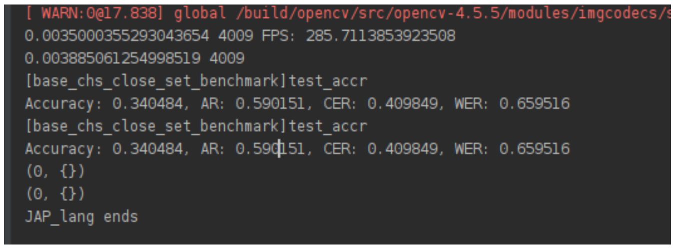
Ablative studies
Base=DUAL_a_Odancukmk7hnp_r45_C_trinorm_dsa3



```
[ WARN:0@17.473] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/
[ WARN:0@17.473] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/
0.003392780591318155 4009 FPS: 294.74349227265606
0.0037672739131 2205 4009
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.314293, AR: 0.584926, CER: 0.415074, WER: 0.685707
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.314293, AR: 0.584926, CER: 0.415074, WER: 0.685707
(0, {})
(0, {})
JAP_lang ends

Process finished with exit code 0
```

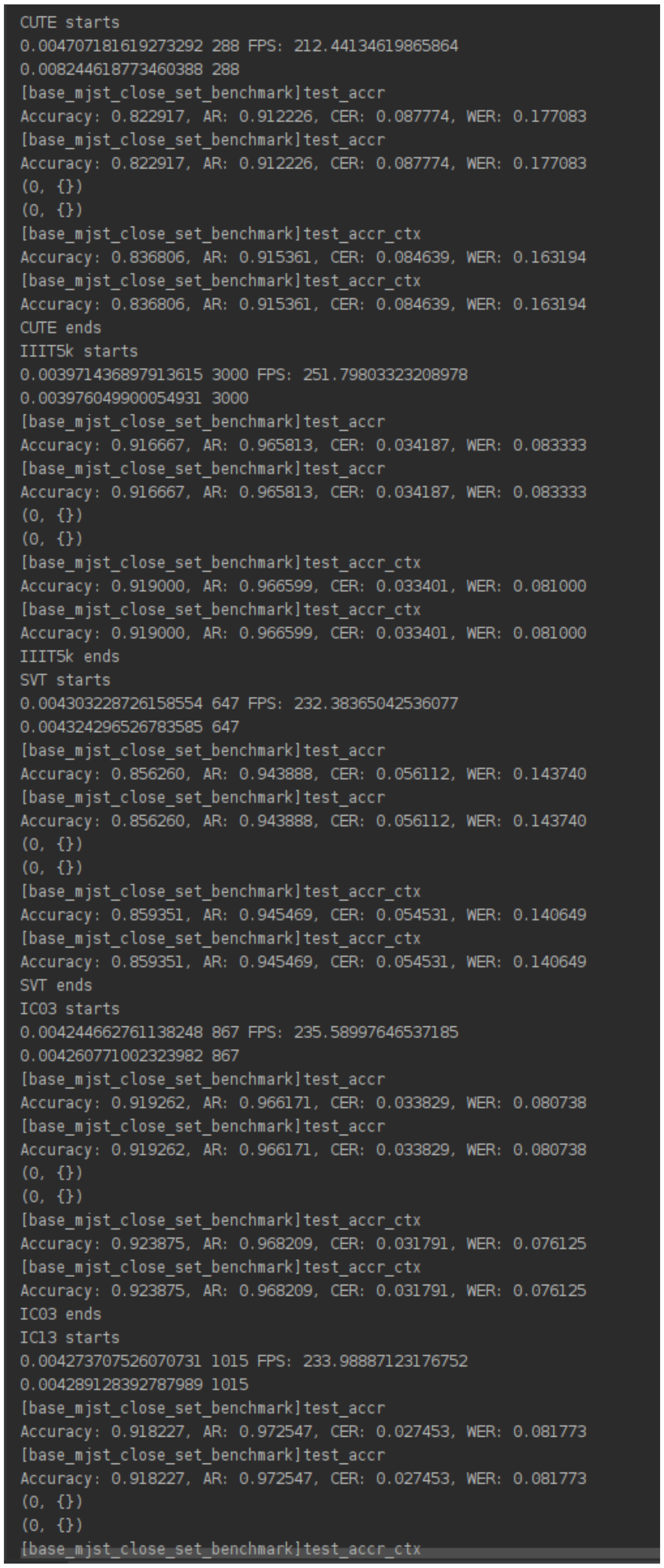
DTA=DUAL_a_Odancukmk7hdtfnp_r45_C_trinorm_dsa3



```
[ WARN:0@17.838] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/s
0.0035000355293043654 4009 FPS: 285.7113853923508
0.003885061254998519 4009
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.340484, AR: 0.590151, CER: 0.409849, WER: 0.659516
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.340484, AR: 0.590151, CER: 0.409849, WER: 0.659516
(0, {})
(0, {})
JAP_lang ends
```

FULL=DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3
See the Regular Fig. above

The close-set benchmarks.
Ours-Large=DUAL_b_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3



```
CUTE starts
0.004707181619273292 288 FPS: 212.44134619865864
0.008244618773460388 288
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.822917, AR: 0.912226, CER: 0.087774, WER: 0.177083
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.822917, AR: 0.912226, CER: 0.087774, WER: 0.177083
(0, {})
(0, {})
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.836806, AR: 0.915361, CER: 0.084639, WER: 0.163194
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.836806, AR: 0.915361, CER: 0.084639, WER: 0.163194
CUTE ends
IIIT5k starts
0.003971436897913615 3000 FPS: 251.79803323208978
0.003976049900054931 3000
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.916667, AR: 0.965813, CER: 0.034187, WER: 0.083333
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.916667, AR: 0.965813, CER: 0.034187, WER: 0.083333
(0, {})
(0, {})
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.919000, AR: 0.966599, CER: 0.033401, WER: 0.081000
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.919000, AR: 0.966599, CER: 0.033401, WER: 0.081000
IIIT5k ends
SVT starts
0.004303228726158554 647 FPS: 232.38365042536077
0.004324296526783585 647
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.856260, AR: 0.943888, CER: 0.056112, WER: 0.143740
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.856260, AR: 0.943888, CER: 0.056112, WER: 0.143740
(0, {})
(0, {})
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.859351, AR: 0.945469, CER: 0.054531, WER: 0.140649
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.859351, AR: 0.945469, CER: 0.054531, WER: 0.140649
SVT ends
IC03 starts
0.004244662761138248 867 FPS: 235.58997646537185
0.004260771002323982 867
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.919262, AR: 0.966171, CER: 0.033829, WER: 0.080738
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.919262, AR: 0.966171, CER: 0.033829, WER: 0.080738
(0, {})
(0, {})
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.923875, AR: 0.968209, CER: 0.031791, WER: 0.076125
[base_mjst_close_set_benchmark]test_accr_ctx
Accuracy: 0.923875, AR: 0.968209, CER: 0.031791, WER: 0.076125
IC03 ends
IC13 starts
0.004273707526070731 1015 FPS: 233.98887123176752
0.004289128392787989 1015
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.918227, AR: 0.972547, CER: 0.027453, WER: 0.081773
[base_mjst_close_set_benchmark]test_accr
Accuracy: 0.918227, AR: 0.972547, CER: 0.027453, WER: 0.081773
(0, {})
(0, {})
[base_mjst_close_set_benchmark]test_accr_ctx
```

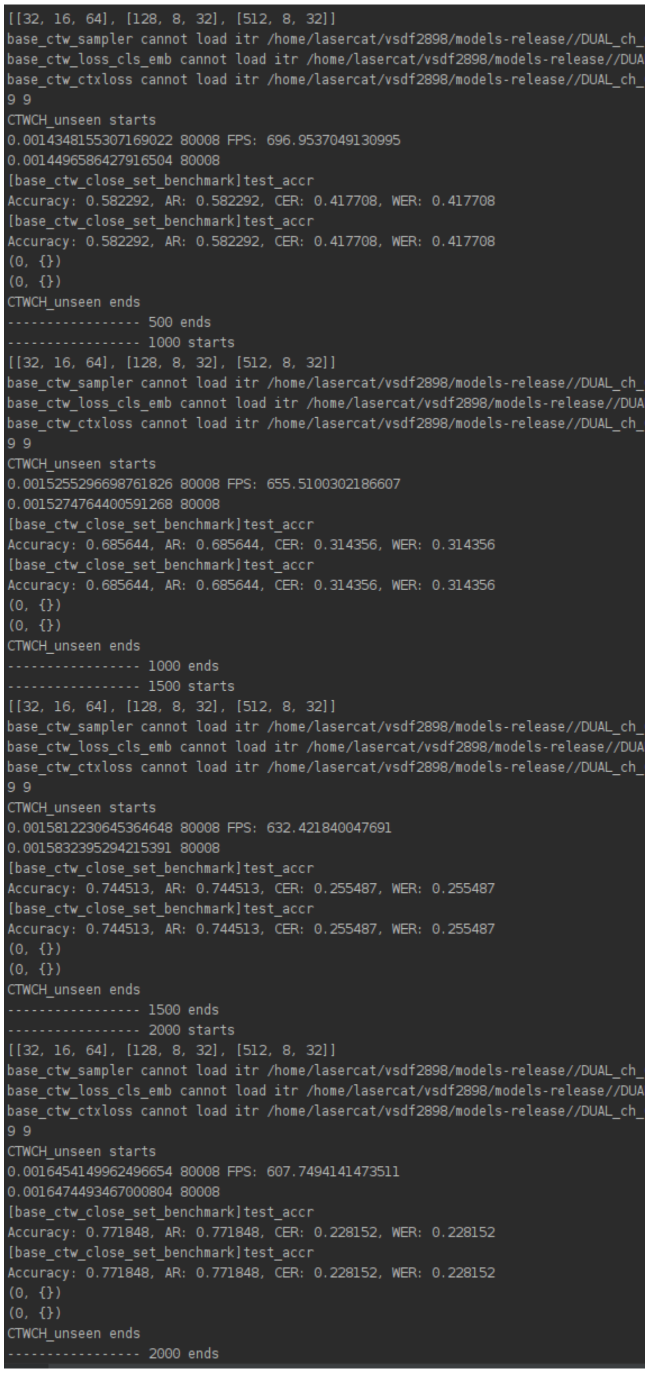
Well if you do want to reproduce the dictionary-based close-set experiments, pls open a Github issue or hack it thru. The codes and the models are there but a few quirks may need ironing out as we updated the interface of image-based testing to allow users to add their own languages.

The zero-shot Chinese character recognitions

CTW=DUAL_ch_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3

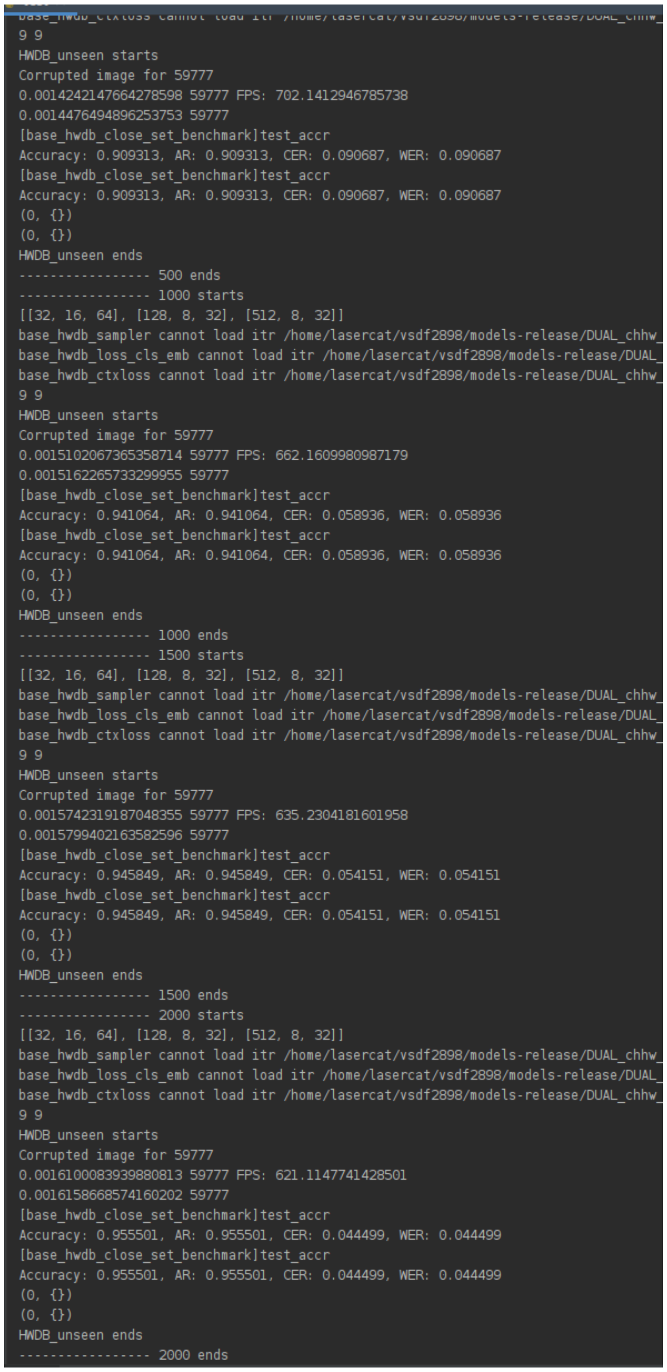
HWDB=DUAL_chhw_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3

CTW



```
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_ctw_sampler cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
base_ctw_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release//DUA
base_ctw_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
9 9
CTWCH_unseen starts
0.0014348155307169022 80008 FPS: 696.9537049130995
0.0014496586427916504 80008
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.582292, AR: 0.582292, CER: 0.417708, WER: 0.417708
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.582292, AR: 0.582292, CER: 0.417708, WER: 0.417708
(0, {})
(0, {})
CTWCH_unseen ends
---------------- 500 ends
---------------- 1000 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_ctw_sampler cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
base_ctw_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release//DUA
base_ctw_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
9 9
CTWCH_unseen starts
0.0015255296698761826 80008 FPS: 655.5100302186607
0.0015274764400591268 80008
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.685644, AR: 0.685644, CER: 0.314356, WER: 0.314356
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.685644, AR: 0.685644, CER: 0.314356, WER: 0.314356
(0, {})
(0, {})
CTWCH_unseen ends
---------------- 1000 ends
---------------- 1500 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_ctw_sampler cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
base_ctw_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release//DUA
base_ctw_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
9 9
CTWCH_unseen starts
0.0015812230645364648 80008 FPS: 632.421840047691
0.0015832395294215391 80008
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.744513, AR: 0.744513, CER: 0.255487, WER: 0.255487
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.744513, AR: 0.744513, CER: 0.255487, WER: 0.255487
(0, {})
(0, {})
CTWCH_unseen ends
---------------- 1500 ends
---------------- 2000 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_ctw_sampler cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
base_ctw_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release//DUA
base_ctw_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release//DUAL_ch_
9 9
CTWCH_unseen starts
0.0016454149962496654 80008 FPS: 607.7494141473511
0.0016474493467000804 80008
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.771848, AR: 0.771848, CER: 0.228152, WER: 0.228152
[base_ctw_close_set_benchmark]test_accr
Accuracy: 0.771848, AR: 0.771848, CER: 0.228152, WER: 0.228152
(0, {})
(0, {})
CTWCH_unseen ends
---------------- 2000 ends
```

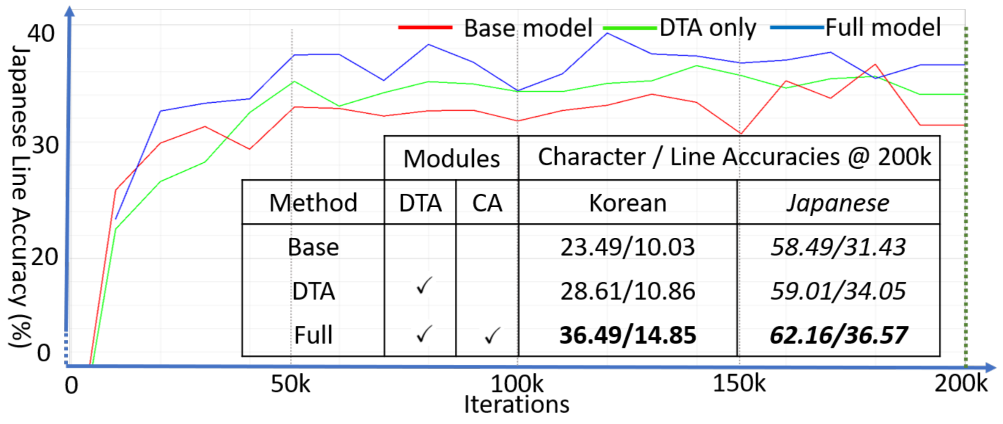
HWDB



```
base_hwdb_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
9 9
HWDB_unseen starts
Corrupted image for 59777
0.0014242147664278598 59777 FPS: 702.1412946785738
0.0014476494896253753 59777
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.909313, AR: 0.909313, CER: 0.090687, WER: 0.090687
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.909313, AR: 0.909313, CER: 0.090687, WER: 0.090687
(0, {})
(0, {})
HWDB_unseen ends
---------------- 500 ends
---------------- 1000 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_hwdb_sampler cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
base_hwdb_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_
base_hwdb_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
9 9
HWDB_unseen starts
Corrupted image for 59777
0.0015102067365358714 59777 FPS: 662.1609980987179
0.0015162265733299955 59777
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.941064, AR: 0.941064, CER: 0.058936, WER: 0.058936
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.941064, AR: 0.941064, CER: 0.058936, WER: 0.058936
(0, {})
(0, {})
HWDB_unseen ends
---------------- 1000 ends
---------------- 1500 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_hwdb_sampler cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
base_hwdb_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_
base_hwdb_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
9 9
HWDB_unseen starts
Corrupted image for 59777
0.0015742319187048355 59777 FPS: 635.2304181601958
0.0015799402163582596 59777
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.945849, AR: 0.945849, CER: 0.054151, WER: 0.054151
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.945849, AR: 0.945849, CER: 0.054151, WER: 0.054151
(0, {})
(0, {})
HWDB_unseen ends
---------------- 1500 ends
---------------- 2000 starts
[[32, 16, 64], [128, 8, 32], [512, 8, 32]]
base_hwdb_sampler cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
base_hwdb_loss_cls_emb cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_
base_hwdb_ctxloss cannot load itr /home/lasercat/vsdf2898/models-release/DUAL_chhw_
9 9
HWDB_unseen starts
Corrupted image for 59777
0.0016100083939880813 59777 FPS: 621.1147741428501
0.0016158668574160202 59777
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.955501, AR: 0.955501, CER: 0.044499, WER: 0.044499
[base_hwdb_close_set_benchmark]test_accr
Accuracy: 0.955501, AR: 0.955501, CER: 0.044499, WER: 0.044499
(0, {})
(0, {})
HWDB_unseen ends
---------------- 2000 ends
```

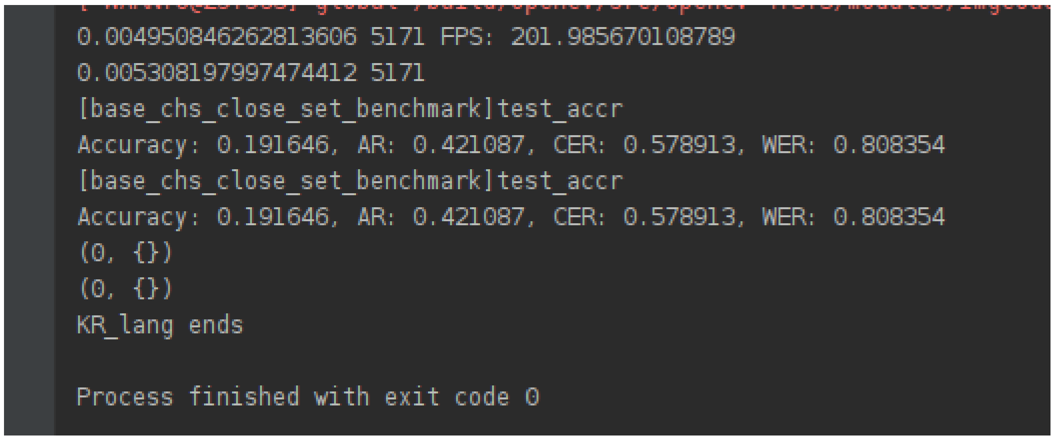
Extra Benchmarks:
Now perhaps you want to see the results on completely unseen language. Well, you can actually have it by running the test_kr.py scripts in *_a_* folders.

Despite we had run that for you on regular models, you may want to rerun it and see the individual images.



	Modules		Character / Line Accuracies @ 200k	
Method	DTA	CA	Korean	*Japanese*
Base			23.49/10.03	*58.49/31.43*
DTA	✓		28.61/10.86	*59.01/34.05*
Full	✓	✓	**36.49/14.85**	***62.16/36.57***

And the large model :



```
0.004950846262813606 5171 FPS: 201.985670108789
0.005308197997474412 5171
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.191646, AR: 0.421087, CER: 0.578913, WER: 0.808354
[base_chs_close_set_benchmark]test_accr
Accuracy: 0.191646, AR: 0.421087, CER: 0.578913, WER: 0.808354
(0, {})
(0, {})
KR_lang ends

Process finished with exit code 0
```

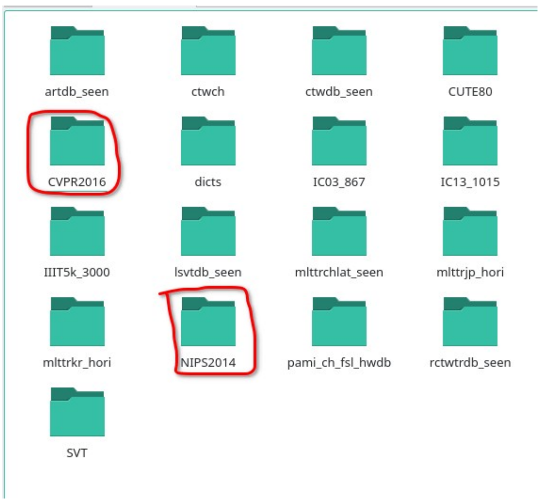
III. **Training with downloaded datasets**

Now you may want to see if this framework works with your novel MaGicNeT and here is where to start. Before that, you may want to train the model as is to get a baseline on your machine and confirm if everything you need is in place.

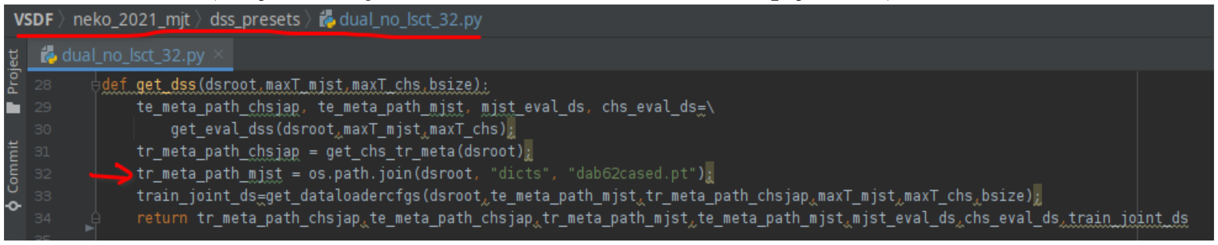
Training the model as is

Step 1: Grab NIPS14 and CVPR16 datasets from Kaggle our repo.

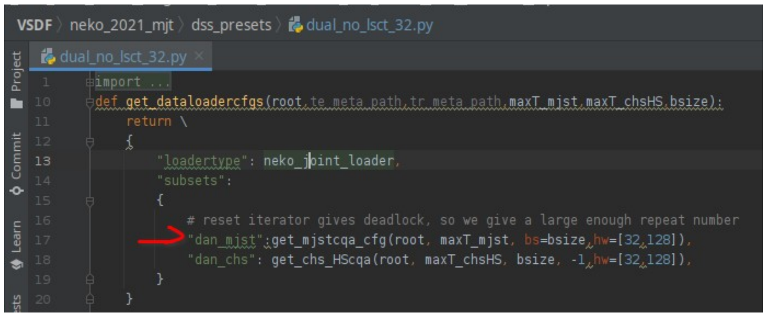
Alternatively, you can download them from DAN (AAAI 20) and put them in the ssddata folder. https://github.com/Wang-Tianwei/Decoupled-attention-network



For open-set benchmarks, they are loaded but not used for coding convenience. You may hack the following dataloader configurations if you think you don't want to load them when you don't use them. (Or you can just fake these datasets with empty ones.)

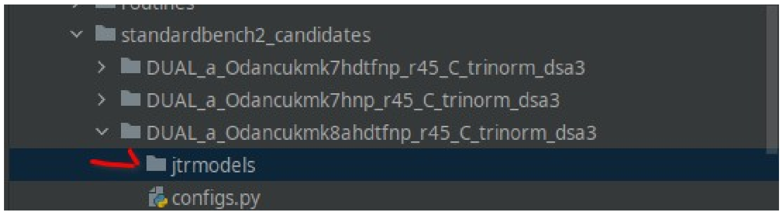


```
def get_dss(dsroot,maxT_mjst,maxT_chs,bsize):
    te_meta_path_chsjap, te_meta_path_mjst, mjst_eval_ds, chs_eval_ds=\
        get_eval_dss(dsroot,maxT_mjst,maxT_chs);
    tr_meta_path_chsjap = get_chs_tr_meta(dsroot);
    tr_meta_path_mjst = os.path.join(dsroot, "dicts", "dab62cased.pt");
    train_joint_ds=get_dataloadercfgs(dsroot,te_meta_path_mjst,tr_meta_path_chsjap,maxT_mjst,maxT_chs,bsize);
    return tr_meta_path_chsjap,te_meta_path_chsjap,tr_meta_path_mjst,te_meta_path_mjst,mjst_eval_ds,chs_eval_ds,train_joint_ds
```



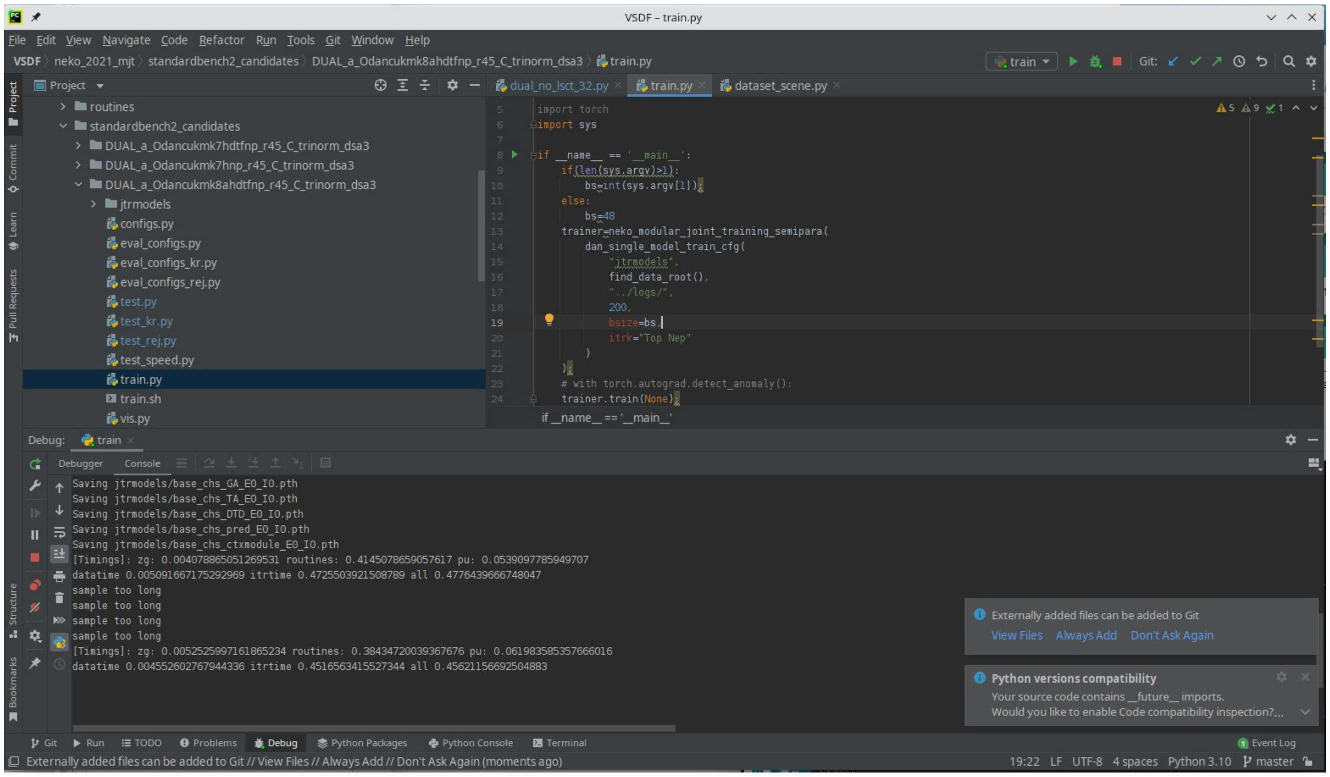
```
import ...
def get_dataloadercfgs(root,te_meta_path,tr_meta_path,maxT_mjst,maxT_chsHS,bsize):
    return \
    {
        "loadertype": neko_joint_loader,
        "subsets":
        {
            # reset iterator gives deadlock, so we give a large enough repeat number
            "dan_mjst":get_mjstcqa_cfg(root, maxT_mjst, bs=bsize,hw=[32,128]),
            "dan_chs": get_chs_HScqa(root, maxT_chsHS, bsize, -1,hw=[32,128]),
        }
    }
```

Step 2: Now that everything is in place, make a directory named jtrmodels in the baseline folder:



```
routines
standardbench2_candidates
    DUAL_a_Odancukmk7hdtfnp_r45_C_trinorm_dsa3
    DUAL_a_Odancukmk7hnp_r45_C_trinorm_dsa3
    DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3
        jtrmodels
        configs.py
```

Step 3: Fire up train.py in pycharm



```
import torch
import sys

if __name__ == '__main__':
    if(len(sys.argv)>1):
        bs=int(sys.argv[1]);
    else:
        bs=48
    trainer=neko_modular_joint_training_semipara(
        dan_single_model_train_cfg(
            "jtrmodels",
            find_data_root(),
            "../logs/",
            200,
            bsize=bs,
            itrk="Top Nep"
        )
    );
    # with torch.autograd.detect_anomaly():
    trainer.train(None);
```

```
Saving jtrmodels/base_chs_GA_E0_I0.pth
Saving jtrmodels/base_chs_TA_E0_I0.pth
Saving jtrmodels/base_chs_DTD_E0_I0.pth
Saving jtrmodels/base_chs_pred_E0_I0.pth
Saving jtrmodels/base_chs_ctxmodule_E0_I0.pth
[Timings]: zg: 0.004078865051269531 routines: 0.4145078659057617 pu: 0.0539097785949707
datatime 0.005091667175292969 itrtime 0.4725503921508789 all 0.4776439666748047
sample too long
sample too long
sample too long
sample too long
[Timings]: zg: 0.005252597161865234 routines: 0.38434720039367676 pu: 0.06198358535766016
datatime 0.004552602767944336 itrtime 0.4516563415527344 all 0.45621156692504883
```

VSDF – train.py
File Edit View Navigate Code Refactor Run Tools Git Window Help
VSDF › neko_2021_mjt › standardbench2_candidates › DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3 › train.py
Project
routines
standardbench2_candidates
DUAL_a_Odancukmk7hdtfnp_r45_C_trinorm_dsa3
DUAL_a_Odancukmk7hnp_r45_C_trinorm_dsa3
DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3
jtrmodels
configs.py
eval_configs.py
eval_configs_kr.py
eval_configs_rej.py
test.py
test_kr.py
test_rej.py
test_speed.py
train.py
train.sh
vis.py
dual_no_lsct_32.py
train.py
dataset_scene.py
Debug: train
Debugger
Console
Externally added files can be added to Git
View Files Always Add Don't Ask Again
Python versions compatibility
Your source code contains __future__ imports.
Would you like to enable Code compatibility inspection?...
Git Run TODO Problems Debug Python Packages Python Console Terminal Event Log
Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (moments ago)
19:22 LF UTF-8 4 spaces Python 3.10 master

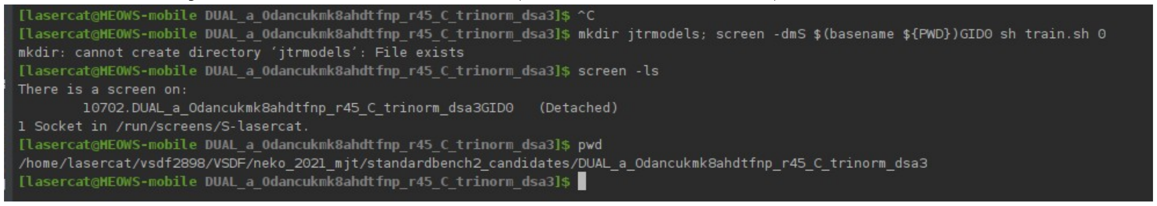
If it trains, it's good.

Step 4. Install screen, stop the pycharm, and train it with screen.

1) Install screen via pacman -S screen

2) Change into the directory and run the training speed with your GPUID (0 here).

mkdir jtrmodels; screen -dmS $(basename ${PWD})GID0 sh train.sh 0



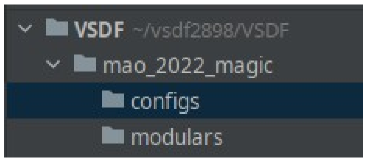
```
[lasercat@MEOWS-mobile DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3]$ ^C
[lasercat@MEOWS-mobile DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3]$ mkdir jtrmodels; screen -dmS $(basename ${PWD})GID0 sh train.sh 0
mkdir: cannot create directory 'jtrmodels': File exists
[lasercat@MEOWS-mobile DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3]$ screen -ls
There is a screen on:
        10702.DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3GID0   (Detached)
1 Socket in /run/screens/S-lasercat.
[lasercat@MEOWS-mobile DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3]$ pwd
/home/lasercat/vsdf2898/VSDF/neko_2021_mjt/standardbench2_candidates/DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3
[lasercat@MEOWS-mobile DUAL_a_Odancukmk8ahdtfnp_r45_C_trinorm_dsa3]$
```

You can check on the PLAYDAN.log to check on the training process. Expect some delay due to buffering

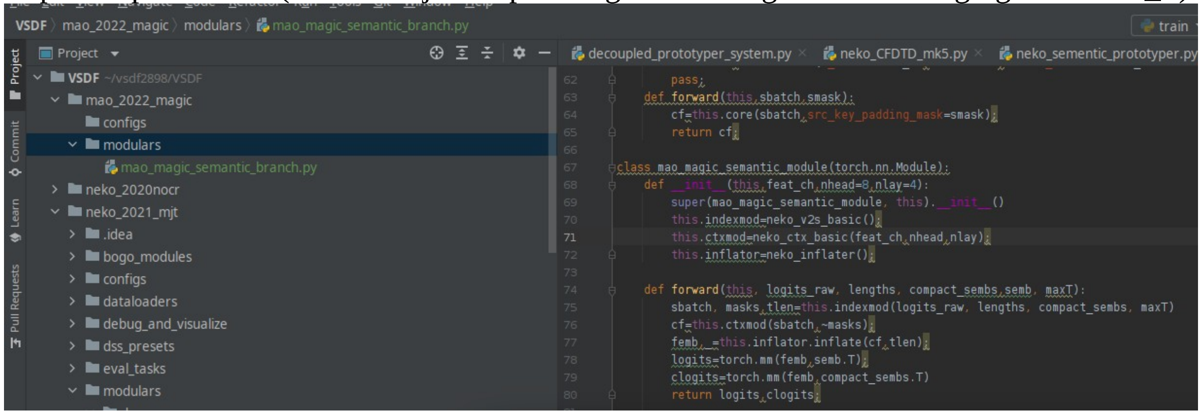
Adding the modules

The framework uses a configure file for each module to achieve complex model sharing, and this design also makes controlling variables simpler. Here are the steps to add your own modular. Let's say you want to have something better than the transformer for the CA.

Step 1. make a working directory, say mao_2022_magic, and add a modular subfolder and a config subfolder.



Step 2. Implement it! (Well I am just duplicating the existing one and changing its name=_=)

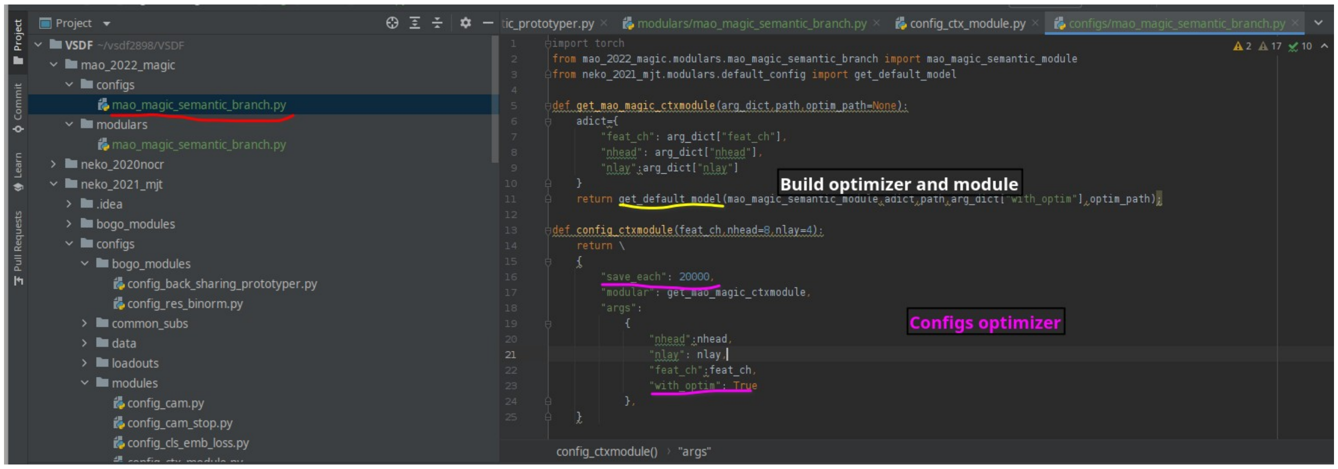


```python
        pass;
    def forward(this,sbatch,smask):
        cf=this.core(sbatch,src_key_padding_mask=smask);
        return cf;

class mao_magic_semantic_module(torch.nn.Module):
    def __init__(this,feat_ch,nhead=8,nlay=4):
        super(mao_magic_semantic_module, this).__init__()
        this.indexmod=neko_v2s_basic();
        this.ctxmod=neko_ctx_basic(feat_ch,nhead,nlay);
        this.inflator=neko_inflater();

    def forward(this, logits_raw, lengths, compact_sembs,semb, maxT):
        sbatch, masks,tlen=this.indexmod(logits_raw, lengths, compact_sembs, maxT)
        cf=this.ctxmod(sbatch,~masks);
        femb,_=this.inflator.inflate(cf,tlen);
        logits=torch.mm(femb,semb.T);
        clogits=torch.mm(femb,compact_sembs.T)
        return logits,clogits;
```

Step 3. Write a config file, specifying if it needs optimizing and if it needs saving.

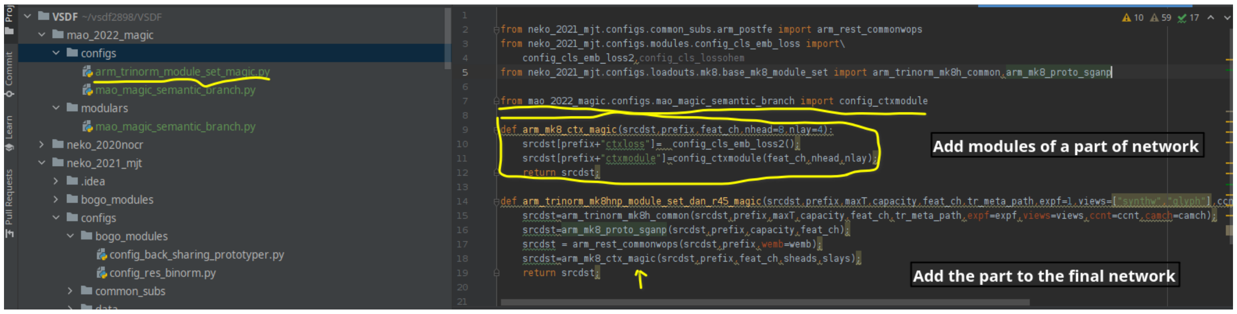


```python
import torch
from mao_2022_magic.modulars.mao_magic_semantic_branch import mao_magic_semantic_module
from neko_2021_mjt.modulars.default_config import get_default_model

def get_mao_magic_ctxmodule(arg_dict,path,optim_path=None):
    adict={
        "feat_ch": arg_dict["feat_ch"],
        "nhead": arg_dict["nhead"],
        "nlay":arg_dict["nlay"]
    }
    return get_default_model(mao_magic_semantic_module,adict,path,arg_dict["with_optim"],optim_path);

def config_ctxmodule(feat_ch,nhead=8,nlay=4):
    return \
    {
        "save_each": 20000,
        "modular": get_mao_magic_ctxmodule,
        "args":
            {
                "nhead":nhead,
                "nlay": nlay,
                "feat_ch":feat_ch,
                "with_optim": True
            },
    }
```

Step 4. Write a network config file including the module. Now we only changed the context module (CA), so we can reuse most of the old network.


Add modules of a part of network
Add the part to the final network

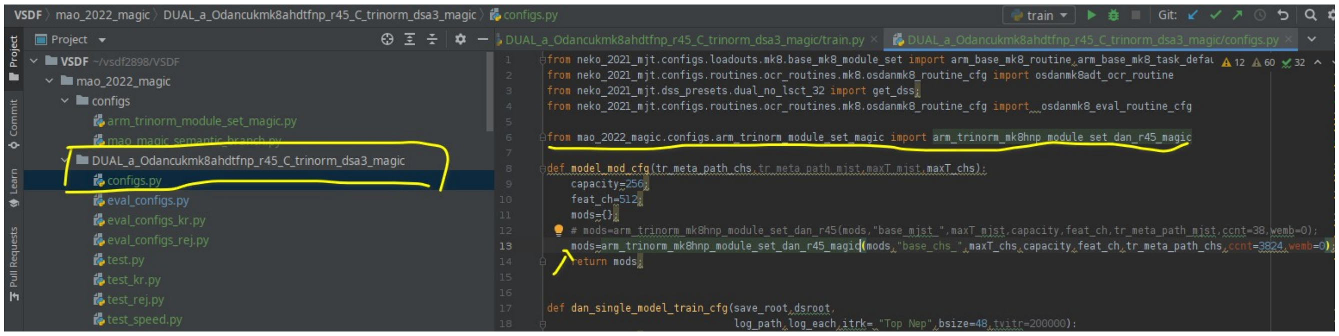
```python
from neko_2021_mjt.configs.common_subs.arm_postfe import arm_rest_commonwops
from neko_2021_mjt.configs.modules.config_cls_emb_loss import\
    config_cls_emb_loss2,config_cls_lossohem
from neko_2021_mjt.configs.loadouts.mk8.base_mk8_module_set import arm_trinorm_mk8h_common,arm_mk8_proto_sganp

from mao_2022_magic.configs.mao_magic_semantic_branch import config_ctxmodule

def arm_mk8_ctx_magic(srcdst,prefix,feat_ch,nhead=8,nlay=4):
    srcdst[prefix+"ctxloss"]= config_cls_emb_loss2();
    srcdst[prefix+"ctxmodule"]=config_ctxmodule(feat_ch,nhead,nlay);
    return srcdst;

def arm_trinorm_mk8hnp_module_set_dan_r45_magic(srcdst,prefix,maxT,capacity,feat_ch,tr_meta_path,expf=1,views=["synthw","glyph"],cc
    srcdst=arm_trinorm_mk8h_common(srcdst,prefix,maxT,capacity,feat_ch,tr_meta_path,expf=expf,views=views,ccnt=ccnt,camch=camch);
    srcdst=arm_mk8_proto_sganp(srcdst,prefix,capacity,feat_ch);
    srcdst = arm_rest_commonwops(srcdst,prefix,wemb=wemb);
    srcdst=arm_mk8_ctx_magic(srcdst,prefix,feat_ch,sheads,slays);
    return srcdst;
```

Step 5. Duplicate a model folder, replace the modules and run it



```python
from neko_2021_mjt.configs.loadouts.mk8.base_mk8_module_set import arm_base_mk8_routine,arm_base_mk8_task_defau
from neko_2021_mjt.configs.routines.ocr_routines.mk8.osdanmk8_routine_cfg import osdanmk8adt_ocr_routine
from neko_2021_mjt.dss_presets.dual_no_lsct_32 import get_dss;
from neko_2021_mjt.configs.routines.ocr_routines.mk8.osdanmk8_routine_cfg import  osdanmk8_eval_routine_cfg

from mao_2022_magic.configs.arm_trinorm_module_set_magic import arm_trinorm_mk8hnp_module_set_dan_r45_magic

def model_mod_cfg(tr_meta_path_chs,tr_meta_path_mjst,maxT_mjst,maxT_chs):
    capacity=256;
    feat_ch=512;
    mods={};
    # mods=arm_trinorm_mk8hnp_module_set_dan_r45(mods,"base_mjst_",maxT_mjst,capacity,feat_ch,tr_meta_path_mjst,ccnt=38,wemb=0);
    mods=arm_trinorm_mk8hnp_module_set_dan_r45_magic(mods,"base_chs_",maxT_chs,capacity,feat_ch,tr_meta_path_chs,ccnt=3824,wemb=0);
    return mods;


def dan_single_model_train_cfg(save_root,dsroot,
                               log_path,log_each,itrk= "Top Nep",bsize=48,tvitr=200000):
```

Other modding

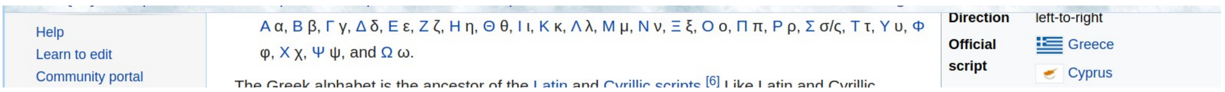
You may have noticed the DTA is implemented via routines rather than modules (`osdanmk8adt_ocr_routine, osdanmk7dt_ocr_routine`), these moddings are somewhat different, however, will not be included in this manual due to the amount of effort to document all the details. Feel free to open an issue if you encounter any trouble building your own routines.

IV. **Add a new language for evaluation**

Say just having CJK is boring and you want to run some languages at your own preference. Here we give an example of how this may or may not work out with the Greek language.

Step 1. Make the charset

a) Find charset on Wikipedia



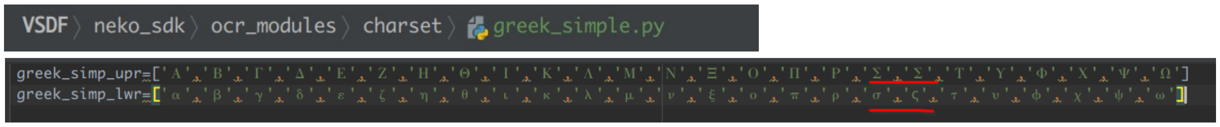
Help
Learn to edit
Community portal

Α α, Β β, Γ γ, Δ δ, Ε ε, Ζ ζ, Η η, Θ θ, Ι ι, Κ κ, Λ λ, Μ μ, Ν ν, Ξ ξ, Ο ο, Π π, Ρ ρ, Σ σ/ς, Τ τ, Υ υ, Φ φ, Χ χ, Ψ ψ, and Ω ω.

The Greek alphabet is the ancestor of the Latin and Cyrillic scripts.[6] Like Latin and Cyrillic

Direction	left-to-right
Official script	Greece Cyprus

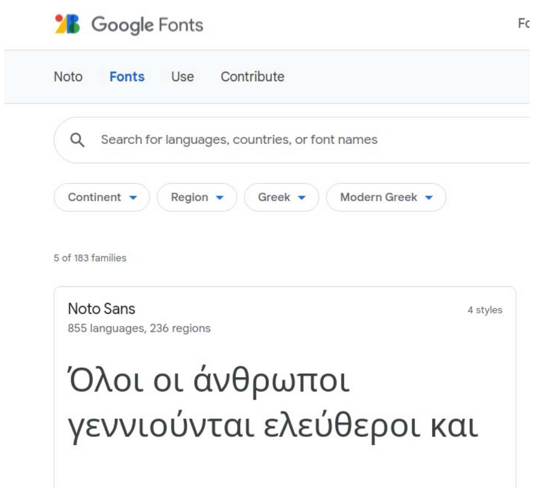
b) Make a list of upper cases and lower cases, note how they are corresponded.



VSDF 〉 neko_sdk 〉 ocr_modules 〉 charset 〉 greek_simple.py

```
greek_simp_upr=['Α','Β','Γ','Δ','Ε','Ζ','Η','Θ','Ι','Κ','Λ','Μ','Ν','Ξ','Ο','Π','Ρ','Σ','Σ','Τ','Υ','Φ','Χ','Ψ','Ω']
greek_simp_lwr=['α','β','γ','δ','ε','ζ','η','θ','ι','κ','λ','μ','ν','ξ','ο','π','ρ','σ','ς','τ','υ','φ','χ','ψ','ω']
```

c) Find noto-sans on greeks.

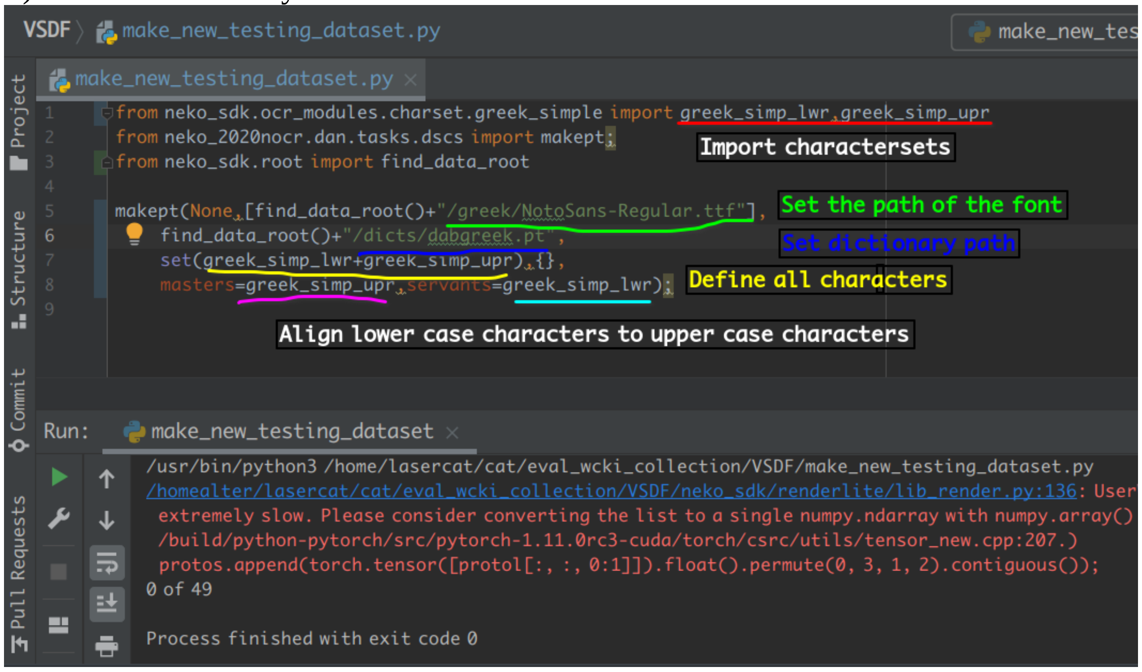


5 of 183 families

Noto Sans
855 languages, 236 regions
4 styles

Όλοι οι άνθρωποι γεννιούνται ελεύθεροι και

d) Build the dictionary file.



VSDF 〉 make_new_testing_dataset.py

```
from neko_sdk.ocr_modules.charset.greek_simple import greek_simp_lwr,greek_simp_upr
from neko_2020nocr.dan.tasks.dscs import makept;
from neko_sdk.root import find_data_root

makept(None,[find_data_root()+"/greek/NotoSans-Regular.ttf"],
    find_data_root()+"/dicts/dabgreek.pt",
    set(greek_simp_lwr+greek_simp_upr),{},
    masters=greek_simp_upr,servants=greek_simp_lwr);
```

Import charactersets

Set the path of the font

Set dictionary path

Define all characters

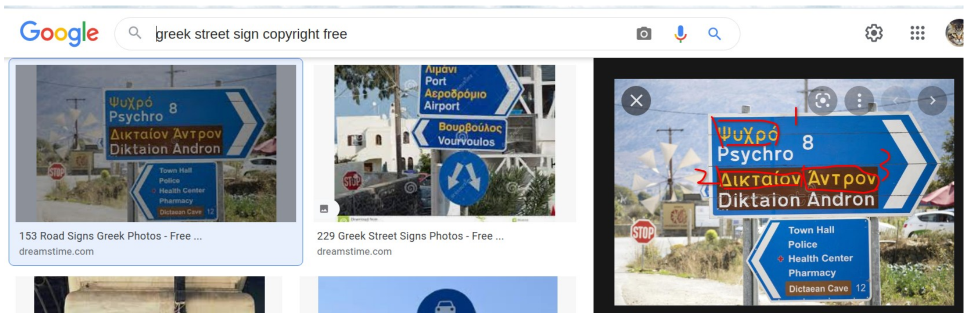
Align lower case characters to upper case characters

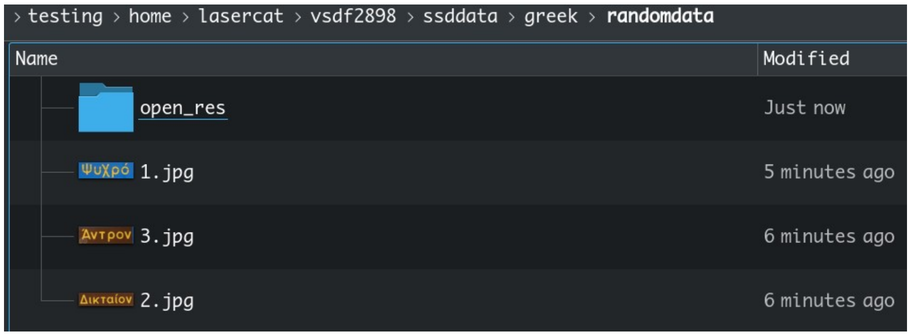
Run: make_new_testing_dataset

```
/usr/bin/python3 /home/lasercat/cat/eval_wcki_collection/VSDF/make_new_testing_dataset.py
/homealter/lasercat/cat/eval_wcki_collection/VSDF/neko_sdk/renderlite/lib_render.py:136: User
 extremely slow. Please consider converting the list to a single numpy.ndarray with numpy.array()
 /build/python-pytorch/src/pytorch-1.11.0rc3-cuda/torch/csrc/utils/tensor_new.cpp:207.)
 protos.append(torch.tensor([protol[:, :, 0:1]]).float().permute(0, 3, 1, 2).contiguous());
0 of 49

Process finished with exit code 0
```

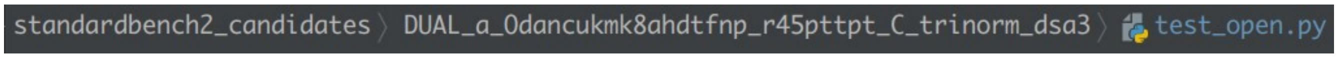
Step 2. Find some data and make a result dir.

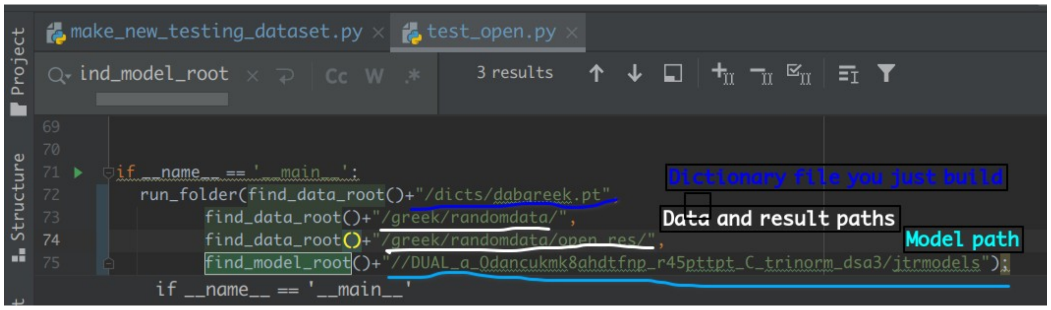



› testing › home › lasercat › vsdf2898 › ssddata › greek › **randomdata**

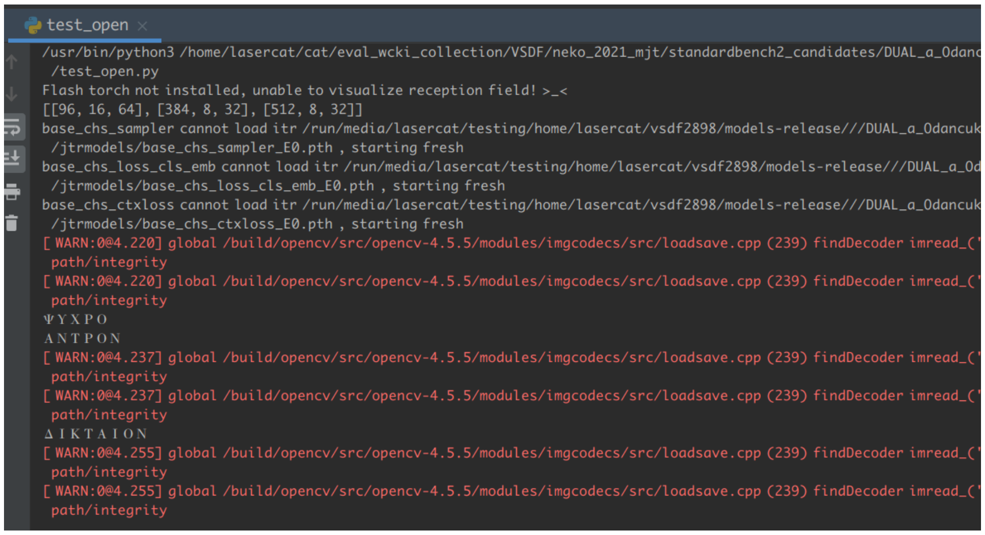
Name	Modified
open_res	Just now
1.jpg	5 minutes ago
3.jpg	6 minutes ago
2.jpg	6 minutes ago

Step 3. Setup paths and run it~


standardbench2_candidates 〉 DUAL_a_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3 〉 test_open.py



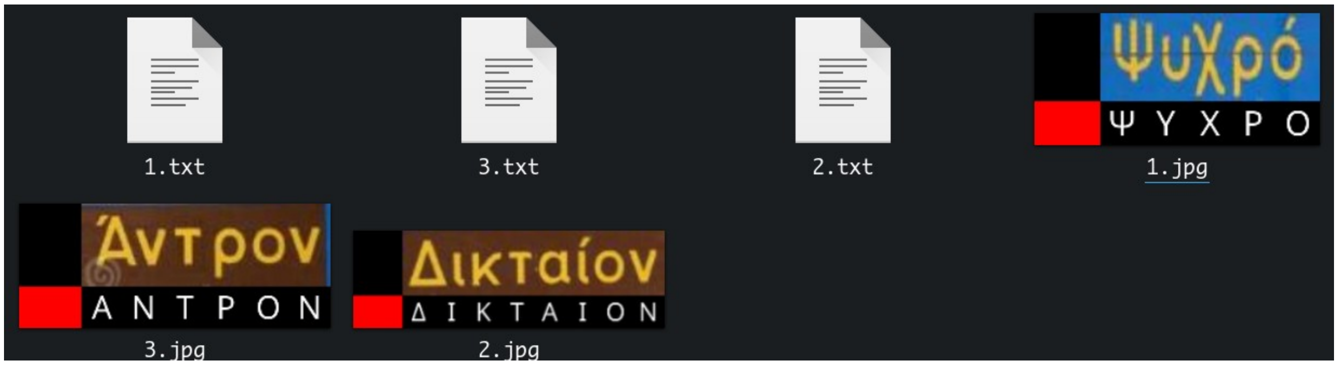
```python
if __name__ == '__main__':
    run_folder(find_data_root()+"/dicts/dabgreek.pt",
            find_data_root()+"/greek/randomdata/",
            find_data_root()+"/greek/randomdata/open_res/",
            find_model_root()+"//DUAL_a_Odancukmk8ahdtfnp_r45pttpt_C_trinorm_dsa3/jtrmodels");
    if __name__ == '__main__'
```



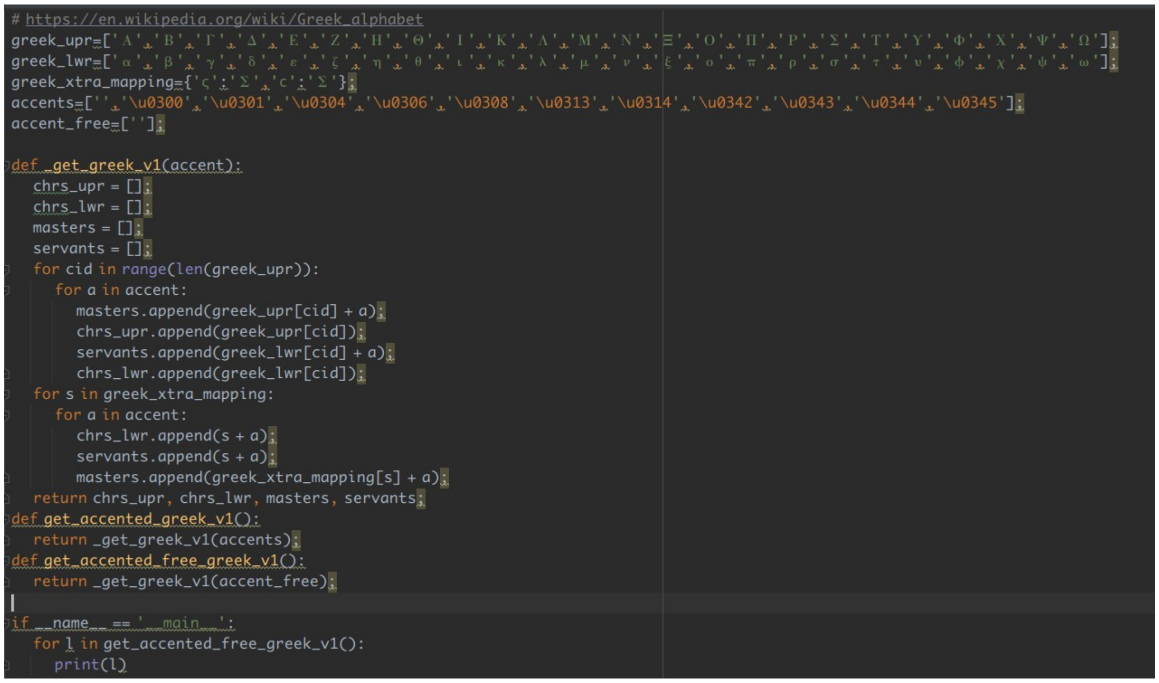
```
/usr/bin/python3 /home/lasercat/cat/eval_wcki_collection/VSDF/neko_2021_mjt/standardbench2_candidates/DUAL_a_Odanc
 /test_open.py
Flash torch not installed, unable to visualize reception field! >_<
[[96, 16, 64], [384, 8, 32], [512, 8, 32]]
base_chs_sampler cannot load itr /run/media/lasercat/testing/home/lasercat/vsdf2898/models-release///DUAL_a_Odancuk
 /jtrmodels/base_chs_sampler_E0.pth , starting fresh
base_chs_loss_cls_emb cannot load itr /run/media/lasercat/testing/home/lasercat/vsdf2898/models-release///DUAL_a_Od
 /jtrmodels/base_chs_loss_cls_emb_E0.pth , starting fresh
base_chs_ctxloss cannot load itr /run/media/lasercat/testing/home/lasercat/vsdf2898/models-release///DUAL_a_Odancuk
 /jtrmodels/base_chs_ctxloss_E0.pth , starting fresh
[ WARN:0@4.220] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
[ WARN:0@4.220] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
ΨΥΧΡΟ
ΑΝΤΡΟΝ
[ WARN:0@4.237] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
[ WARN:0@4.237] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
ΔΙΚΤΑΙΟΝ
[ WARN:0@4.255] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
[ WARN:0@4.255] global /build/opencv/src/opencv-4.5.5/modules/imgcodecs/src/loadsave.cpp (239) findDecoder imread_('
 path/integrity
```

Now why v is recognized as n I wonder? (Because 'N'-'ν') Well, it's not bad eh.

Step 4. Harvest text results and prediction in the result folder.



Step 6, you may actually try some thing more complex like this:



```
# https://en.wikipedia.org/wiki/Greek_alphabet
greek_upr=['Α','Β','Γ','Δ','Ε','Ζ','Η','Θ','Ι','Κ','Λ','Μ','Ν','Ξ','Ο','Π','Ρ','Σ','Τ','Υ','Φ','Χ','Ψ','Ω'];
greek_lwr=['α','β','γ','δ','ε','ζ','η','θ','ι','κ','λ','μ','ν','ξ','ο','π','ρ','σ','τ','υ','φ','χ','ψ','ω'];
greek_xtra_mapping={'ς':'Σ','ϲ':'Σ'};
accents=['','\u0300','\u0301','\u0304','\u0306','\u0308','\u0313','\u0314','\u0342','\u0343','\u0344','\u0345'];
accent_free=[''];

def _get_greek_v1(accent):
    chrs_upr = [];
    chrs_lwr = [];
    masters = [];
    servants = [];
    for cid in range(len(greek_upr)):
        for a in accent:
            masters.append(greek_upr[cid] + a);
            chrs_upr.append(greek_upr[cid]);
            servants.append(greek_lwr[cid] + a);
            chrs_lwr.append(greek_lwr[cid]);
    for s in greek_xtra_mapping:
        for a in accent:
            chrs_lwr.append(s + a);
            servants.append(s + a);
            masters.append(greek_xtra_mapping[s] + a);
    return chrs_upr, chrs_lwr, masters, servants;
def get_accented_greek_v1():
    return _get_greek_v1(accents);
def get_accented_free_greek_v1():
    return _get_greek_v1(accent_free);

if __name__ == '__main__':
    for l in get_accented_free_greek_v1():
        print(l)
```

We won’t demo it here, but it is definitly on our schedule for near future.