riskserver-debugging-svm-with-queue-multi-model-min specification

Cyrus

October 10, 2017

1 Deploy

1 install django and mysql

2 modified the DATABASE in riskserver/settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',

        # modified below
        'NAME': 'test',
        'USER': 'root',
        'PASSWORD': 'root',
        # modified above

        'HOST': '',
        'PORT': '',
    }
}
```

3 migrate and create administrator
Enter to the project file, and run cmds below

```
python manage.py makemigrations
python manage.py migrate
python manage.py createsuperuser
```

4 run the server
Enter to the project file, and run cmds below

```
./runServer.py
./runWorker.py
```

2 Applications

This Multi-model-based system is much different from before.

A model box is a kind of posture. For every group of files, all the box will be asked and give responses (accuracy). Then decision will be made if the accuracy locates in a given range.

Models are used to store useful information in every application. Access to http://localhost:8000/admin to manage these data.

2.1 trainapp

The trainapp is divided into three parts, upload, dispatch and train.

The path is http://localhost:8000/train.

Upload. After checking the state(sit or walk), all the uploaded files will be grouped and stored in to the database. The group id will be assigned, which is very useful in next phases. Change TRAIN_FILE _GROUP_SIZE to alter the size of group. You can find this moduel in trainapp/view.py.

Dispatch. This will be triggered as soon as there is enough files for a group using the django-rq. These files will be dispatched into a model box (a kind of posture) existing or new. You can set BOX_IN_ACCRUCY and _UPPER BOX_IN_ACCURACY_LOWER to control this behavior to avoid excessive boxes or none. And also, udpate TRAINING_TIME to limit the numbers of models in a box(every model trained will not be deleted except bad performing one). Finaly, the train phase will be called. Relative models will be updated.You can find this moduel in trainapp/tasks.py.

Train. Semi-supervised system is supported. Geting all valid files in a box with files picked up from others, we seperated them into two parts, one for training and the other for testing. RATION and TRAIN_PROPORTION are configurable. By checking the performance between the new model and the old one, we make the decision. Relative models will be updated. You can find this moduel in trainapp/tasks.py.

2.2 testapp

After training, the test will begin when a file uploaded. All the things this application to do is comparing. Get all latest models and find a better one. All the middle results will be stored into database. The applicaiton contrains two moduels, one is in testapp/view.py and the other is in testapp/tasks.py, which are similar to Upload and Dispatch.

The path is http://localhost:8000/test.

2.3 query

You can get the max accuracy querying by a version.

The path is http://localhost:8000/query.

3 Admin Interface

There is a powerful admin interface in Django. Access to http://localhost:8000/admin, and have a try.

By the way, you can also manage djang-rq here.