

Springboard ML Course Capstone Project Deployment

Our face recognition model is based on Dlib and KNN. A set of face images were used to train the KNN model. The training program is `face_model_train.py`. The program can be executed as below:

```
(face1)$ python face_model_train.py
```

A trained model `face_model_file_frg` will be saved in current folder by using pickle

Deployment Process

1 Create a flask app

The `load_model()` loads a trained ML model

The `get_prediction()` receives JSON data. A face encoding 128D data is treated as a string in the JSON data. This function will extract the string and convert it to a numpy array. The 128 x 1 array is sent to `model.predict()` to get a prediction (name)

The program is saved as `face_app.py`

```

In [ ]: 1 #ShengpingJiang- Face recognition model as a flask application
2
3 import pickle
4 import numpy as np
5 from flask import Flask, request
6
7 #model = None
8 app = Flask(__name__)
9
10
11 def load_model():
12     global model
13     # model variable refers to the global variable
14     with open('face_model_file_frg', 'rb') as f:
15         model = pickle.load(f)
16
17
18 @app.route('/')
19 def home_endpoint():
20     return 'Hello World!'
21
22
23 @app.route('/predict', methods=['GET', 'POST'])
24 def get_prediction():
25     dist_threshold = 0.4
26     name = ''
27     # Works only for a single sample
28     if request.method == 'POST':
29         data = request.get_json() # Get data posted as a json
30         #data[0] means 1st {} in the JSON data [{..},{..}]. data[0]
31         #the value of key 'encoding' in data[0]
32         #print(type(data[0]['encoding']))
33         #print(data[0]['encoding'])
34         #The value of the key 'encoding' is a string '[-0.17077433
35         str1 = data[0]['encoding']
36         # str1[1:-1] from '[-0.17077433 0.086519...]' to '-0.170774
37         # np.fromstring changes a string '-0.17077433 0.086519...'
38         # [-0.17077433 0.086519...]
39         encoding = np.fromstring(str1[1:-1], dtype=float, sep=' ')
40         #print("ecoding type:", type(encoding))
41         #print(encoding)
42
43         # reshape(1,-1) change [-0.17077433 0.086519...] to [[-0.17
44         xt = encoding.reshape(1,-1)
45         #print('xt:', xt)
46         closest_distance = model.kneighbors(xt, n_neighbors=1, return
47         #print("closest_distance[0][0][0]:",closest_distance[0][0][0]
48         if closest_distance[0][0][0] <= dist_threshold :
49             # model.predict(xt) returns a string list ['name']
50             # model.predict(xt)[0] returns 'name'
51             name = model.predict(xt)[0]
52             print('name:', name)
53         else:
54             name = "Unknown"
55     elif request.method == 'GET':
56         print("Shengping")

```

```

57
58     return name
59
60
61 if __name__ == '__main__':
62     load_model() # load model at the beginning once only
63     app.run(host='0.0.0.0', port=5000)
64

```

2 Test face_app.py in faceprod virtualenv

2.1 Create a virtual env faceprod and install packages

mkvirtualenv faceprod -p python3

(faceprod) pip install numpy

(faceprod) pip install flask

(faceprod) pip install pickle-mixin

(faceprod) pip install sklearn

(faceprod) pip freeze > faceprod_list.txt

2.2 Launch the flask app face_app.py

(faceprod)\$ python face_app.py

2.3 Open another terminal. Send test data (dlib face encoding 128D vector) to web

0.0.0.0:5000/predict, and test the model

```

$ curl -X POST 0.0.0.0:5000/predict -H 'Content-Type: application/json' -d '{"encoding": "[
-0.17077433 0.086519 0.04608656 0.02226515 -0.10071052 0.0246949 -0.09879136
-0.08271502 0.15330137 -0.1101086 0.2084657 0.0172283 -0.18812549 0.00964276
-0.06756912 0.11148367 -0.11918792 -0.07723383 -0.05200598 -0.01760992 0.0567386
0.04599836 0.03339319 0.04884979 -0.10915887 -0.33869374 -0.10735007 -0.11223182
0.08643846 -0.07478593 -0.05546422 -0.08678006 -0.11504613 0.01475477 0.01169325
0.15265159 -0.02465688 -0.06824835 0.21678171 -0.03042633 -0.19874264 -0.01212559
-0.02762683 0.26414317 0.13703299 0.0334272 0.01637992 -0.10932572 0.09580361
-0.21135658 0.11234938 0.1291863 0.0340074 0.03284376 0.09014399 -0.17272305
0.01153929 0.14709072 -0.14064969 0.02695761 0.03161349 0.01307983 -0.0100578
-0.05213601 0.20376676 0.14580815 -0.11039062 -0.15493403 0.11541102 -0.2119666
0.0013991 0.08922509 -0.11429761 -0.22043382 -0.28854343 0.04549009 0.44805536
0.20364918 -0.16662233 0.02062135 -0.00946902 -0.02268174 0.16432424 0.10247331
-0.08463222 0.0589206 -0.11151487 0.04075154 0.17744561 0.00353054 -0.0321093
0.19991624 0.01635876 0.06169297 0.05581587 0.04786064 -0.07188784 -0.04009981
-0.1177263 -0.01570286 0.08082893 -0.0241716 0.03095182 0.11278267 -0.16012146
0.1034444 -0.01475013 -0.01811141 0.03154366 0.02885633 -0.14979976 -0.0449345
0.21942021 -0.22967488 0.15503235 0.15902625 0.02446658 0.15540583 0.12920454
0.0752509 -0.01832712 -0.00534262 -0.19305748 -0.00229457 0.01291393 -0.05213701
0.07341617 0.01301993]"]}'

```

Note: above command is one line. No return is in the line

3 Create a Dockerfile

Use text editor to create Dockerfile and put in lines below:

FROM python:3.6-slim

```
..
COPY ./face_app.py /deploy/
COPY ./faceprod_list.txt /deploy/
COPY ./face_model_file_frg /deploy/
COPY ./LICENSE /deploy/
COPY ./README.md /deploy/

WORKDIR /deploy/
RUN pip install -r faceprod_list.txt
EXPOSE 5000
ENTRYPOINT ["python", "face_app.py"]
```

```
1 4 Create Docker image<br>
2 Get out the virtual env faceprod. Check docker is running<br>
3 $ docker run hello-world<br>
4 Got permission denied...<br>
5 $ sudo chmod 666 /var/run/docker.sock #this command fix above
   error<br>
6 $ docker run hello-world<br>
7 Hello from Docker!<br>
8
9 Create docker image<br>
10 ~/faceprod$ docker build -t faceprod .
11
```

```
FT 08:35
simon@jspace: ~/faceprod
File Edit View Search Terminal Help
https://docs.docker.com/get-started/
simon@jspace:~/faceprod$ docker build -t faceprod .
Sending build context to Docker daemon 52.02MB
Step 1/10 : FROM python:3.6-slim
----> c36a97a24d09
Step 2/10 : COPY ./face_app.py /deploy/
----> d11c05055c32
Step 3/10 : COPY ./faceprod_list.txt /deploy/
----> 5a4d6dc70ca6
Step 4/10 : COPY ./face_model_file_frg /deploy/
----> 07b150989be1
Step 5/10 : COPY ./LICENSE /deploy/
----> 8f2de98dc2dc
Step 6/10 : COPY ./README.md /deploy/
----> 698f12915bc1
Step 7/10 : WORKDIR /deploy/
----> Running in 7124f593b2ac
Removing intermediate container 7124f593b2ac
----> 175d857eca47
Step 8/10 : RUN pip install -r faceprod_list.txt
----> Running in 16fd1c8b5397
Collecting click==7.1.2
  Downloading click-7.1.2-py2.py3-none-any.whl (82 kB)
Collecting Flask==1.1.2
  Downloading Flask-1.1.2-py2.py3-none-any.whl (94 kB)
Collecting itsdangerous==1.1.0
  Downloading itsdangerous-1.1.0-py2.py3-none-any.whl (16 kB)
Collecting Jinja2==2.11.2
  Downloading Jinja2-2.11.2-py2.py3-none-any.whl (125 kB)
Collecting joblib==0.17.0
  Downloading joblib-0.17.0-py3-none-any.whl (301 kB)
Collecting MarkupSafe==1.1.1
  Downloading MarkupSafe-1.1.1-cp36-cp36m-manylinux1_x86_64.whl (27 kB)
```

```

Download MarkupSafe-1.1.1-cp36-cp36m-manylinux1_x86_64.whl (27 kB)
Collecting numpy==1.19.2
  Downloading numpy-1.19.2-cp36-cp36m-manylinux2010_x86_64.whl (14.5 MB)
Collecting pickle-mixin==1.0.2
  Downloading pickle-mixin-1.0.2.tar.gz (5.1 kB)
Collecting scikit-learn==0.23.2
  Downloading scikit_learn-0.23.2-cp36-cp36m-manylinux1_x86_64.whl (6.8 MB)
Collecting scipy==1.5.2
  Downloading scipy-1.5.2-cp36-cp36m-manylinux1_x86_64.whl (25.9 MB)
Collecting sklearn==0.0
  Downloading sklearn-0.0.tar.gz (1.1 kB)
Collecting threadpoolctl==2.1.0
  Downloading threadpoolctl-2.1.0-py3-none-any.whl (12 kB)
Collecting Werkzeug==1.0.1
  Downloading Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
Building wheels for collected packages: pickle-mixin, sklearn
  Building wheel for pickle-mixin (setup.py): started
  Building wheel for pickle-mixin (setup.py): finished with status 'done'
  Created wheel for pickle-mixin: filename=pickle_mixin-1.0.2-py3-none-any.whl size=5997 sha256=6ab4334f59bd83b
ac33fb7bca5afd3d57db3036a669fcafd4d6ff013d2d112a
  Stored in directory: /root/.cache/pip/wheels/82/53/b0/6f80da2d461fa5f582eb274b0158ce81d01b977cbb59a2ae6a
  Building wheel for sklearn (setup.py): started
  Building wheel for sklearn (setup.py): finished with status 'done'
  Created wheel for sklearn: filename=sklearn-0.0-py2.py3-none-any.whl size=1316 sha256=f37e5e4b88be01920eb4112
b376ca04f4f26548dcff4750285d02c846907a7bc
  Stored in directory: /root/.cache/pip/wheels/23/9d/42/5ec745cbbb17517000a53cecc49d6a865450d1f5cb16dc8a9c
Successfully built pickle-mixin sklearn
Installing collected packages: click, Werkzeug, MarkupSafe, Jinja2, itsdangerous, Flask, joblib, numpy, pickle-
mixin, scipy, threadpoolctl, scikit-learn, sklearn
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 itsdangerous-1.1.0
joblib-0.17.0 numpy-1.19.2 pickle-mixin-1.0.2 scikit-learn-0.23.2 scipy-1.5.2 sklearn-0.0 threadpoolctl-2.1.0
Removing intermediate container 16fd1c8b5397
--> e4e07270196a
Step 9/10 : EXPOSE 80
--> Running in 7007bd2197ed
Removing intermediate container 7007bd2197ed
--> 0b4f546e7158
Step 10/10 : ENTRYPOINT ["python", "face_app.py"]
--> Running in eaec9aae78a4
Removing intermediate container eaec9aae78a4
--> 8cd00e033003
Successfully built 8cd00e033003
Successfully tagged faceprod:latest
simon@jsspace:~/faceprod$ docker run -p 5000:80 faceprod .

```

5 Launch and test docker image

Run docker image. 1st 5000 is local machine port. 2nd 5000 is the port assigned in face_app.py (it is inside docker image)

~/faceprod\$ docker run -p 5000:5000 faceprod .

- Serving Flask app "face_app" (lazy loading)
- Environment: production
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
- Debug mode: off
- Running on <http://0.0.0.0:5000/> (<http://0.0.0.0:5000/>) (Press CTRL+C to quit)


```

--> 8cd00e033003
Successfully built 8cd00e033003
Successfully tagged faceprod:latest
simon@jspace:~/faceprod$ docker run -p 5000:80 faceprod .
* Serving Flask app "face_app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
^Csimon@jspace:~/faceprod$ docker run -p 5000:5000 faceprod .
* Serving Flask app "face_app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
172.17.0.1 - - [16/Oct/2020 02:20:30] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [16/Oct/2020 02:20:55] "POST /predict HTTP/1.1" 200 -
172.17.0.1 - - [16/Oct/2020 15:32:13] "GET /predict HTTP/1.1" 200 -

```

In another terminal, send test data and get response

```

$ curl -X POST 0.0.0.0:5000/predict -H 'Content-Type: application/json' -d '{"encoding": "[
-0.17077433 0.086519 0.04608656 0.02226515 -0.10071052 0.0246949 -0.09879136
-0.08271502 0.15330137 -0.1101086 0.2084657 0.0172283 -0.18812549 0.00964276
-0.06756912 0.11148367 -0.11918792 -0.07723383 -0.05200598 -0.01760992 0.0567386
0.04599836 0.03339319 0.04884979 -0.10915887 -0.33869374 -0.10735007 -0.11223182
0.08643846 -0.07478593 -0.05546422 -0.08678006 -0.11504613 0.01475477 0.01169325
0.15265159 -0.02465688 -0.06824835 0.21678171 -0.03042633 -0.19874264 -0.01212559
-0.02762683 0.26414317 0.13703299 0.0334272 0.01637992 -0.10932572 0.09580361
-0.21135658 0.11234938 0.1291863 0.0340074 0.03284376 0.09014399 -0.17272305
0.01153929 0.14709072 -0.14064969 0.02695761 0.03161349 0.01307983 -0.0100578
-0.05213601 0.20376676 0.14580815 -0.11039062 -0.15493403 0.11541102 -0.2119666
0.0013991 0.08922509 -0.11429761 -0.22043382 -0.28854343 0.04549009 0.44805536
0.20364918 -0.16662233 0.02062135 -0.00946902 -0.02268174 0.16432424 0.10247331
-0.08463222 0.0589206 -0.11151487 0.04075154 0.17744561 0.00353054 -0.0321093
0.19991624 0.01635876 0.06169297 0.05581587 0.04786064 -0.07188784 -0.04009981
-0.1177263 -0.01570286 0.08082893 -0.0241716 0.03095182 0.11278267 -0.16012146
0.1034444 -0.01475013 -0.01811141 0.03154366 0.02885633 -0.14979976 -0.0449345
0.21942021 -0.22967488 0.15503235 0.15902625 0.02446658 0.15540583 0.12920454
0.0752509 -0.01832712 -0.00534262 -0.19305748 -0.00229457 0.01291393 -0.05213701
0.07341617 0.01301993]"]'

```

An answer from the flask app:

004郭坚

Screenshot for testing docker image and get an answer:

```
File Edit View Search Terminal Help
simon@jspacer:~$ curl -X POST 0.0.0.0:5000/predict -H 'Content-Type: application
/json' -d '[{"encoding": "[-0.17077433 0.086519 0.04608656 0.02226515 -0.10
071052 0.0246949 -0.09879136 -0.08271502 0.15330137 -0.1101086 0.2084657 0
.0172283 -0.18812549 0.00964276 -0.06756912 0.11148367 -0.11918792 -0.07723383
-0.05200598 -0.01760992 0.0567386 0.04599836 0.03339319 0.04884979 -0.1091
5887 -0.33869374 -0.10735007 -0.11223182 0.08643846 -0.07478593 -0.05546422 -0.
08678006 -0.11504613 0.01475477 0.01169325 0.15265159 -0.02465688 -0.06824835
0.21678171 -0.03042633 -0.19874264 -0.01212559 -0.02762683 0.26414317 0.1370
3299 0.0334272 0.01637992 -0.10932572 0.09580361 -0.21135658 0.11234938 0.
1291863 0.0340074 0.03284376 0.09014399 -0.17272305 0.01153929 0.14709072
-0.14064969 0.02695761 0.03161349 0.01307983 -0.0100578 -0.05213601 0.203766
76 0.14580815 -0.11039062 -0.15493403 0.11541102 -0.2119666 0.0013991 0.08
922509 -0.11429761 -0.22043382 -0.28854343 0.04549009 0.44805536 0.20364918 -
0.16662233 0.02062135 -0.00946902 -0.02268174 0.16432424 0.10247331 -0.084632
22 0.0589206 -0.11151487 0.04075154 0.17744561 0.00353054 -0.0321093 0.19
991624 0.01635876 0.06169297 0.05581587 0.04786064 -0.07188784 -0.04009981 -
0.1177263 -0.01570286 0.08082893 -0.0241716 0.03095182 0.11278267 -0.16012146
0.1034444 -0.01475013 -0.01811141 0.03154366 0.02885633 -0.14979976 -0.04493
45 0.21942021 -0.22967488 0.15503235 0.15902625 0.02446658 0.15540583 0.12
920454 0.0752509 -0.01832712 -0.00534262 -0.19305748 -0.00229457 0.01291393 -0
.05213701 0.07341617 0.01301993]}]']
004郭坚simon@jspacer:~$
```