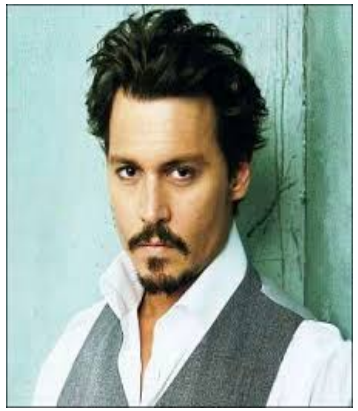
Building face recognition application

Fan Yang

Project summary

- Who are they?



- Who are really involved in these movies?


Image pre-processing

- Images were downloaded from Google Image, presenting 6 actors and 20 pieces per actor.
- For each actor, 15 images were assigned as training set and 5 images for test set
- Images were read by OpenCV
- Faces were detected based on Haar Cascade of Classifiers method (Viola and Jones, 2001, Rapid object detection using a boosted cascade of simple features)
- Face images were converted into grayscale mode and be resized to 28x28

Classification accuracy from traditional machine learning models are less than 0.48

method 1: Logistic regression

```
from sklearn.linear_model import LogisticRegression as LR

model = LR()
model.fit(X_train, Y_train)

print("Logistic regression: ",  "Training Accuracy=", model.score(X_train, Y_train),
      "Test Accuracy=", model.score(X_test, Y_test))
```

```
Logistic regression:  Training Accuracy= 1.0 Test Accuracy= 0.48
```

method 2: Naive bayes

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, Y_train)
print("Naive_bayes: ", "Training Accuracy=", model.score(X_train, Y_train),
      "Test Accuracy=", model.score(X_test, Y_test))
```

```
Naive_bayes:  Training Accuracy= 0.72 Test Accuracy= 0.4
```

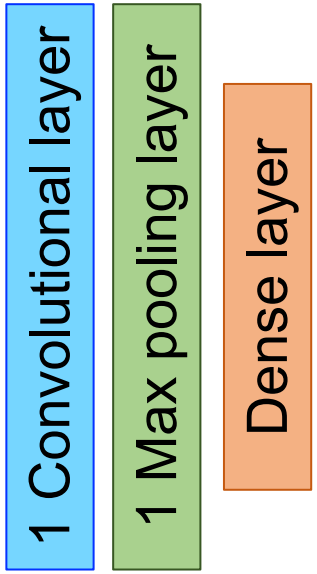
method 3: Support vector machine

```
from sklearn.svm import SVC
model = SVC()
model.fit(X_train, Y_train)
print("Support vector machine: ", "Training Accuracy=", model.score(X_train, Y_train),
      "Test Accuracy=", model.score(X_test, Y_test))
```

```
Support vector machine:  Training Accuracy= 0.56 Test Accuracy= 0.36
```

Convolutional neural network (CNN) with simple structure produce classification accuracy at 0.72

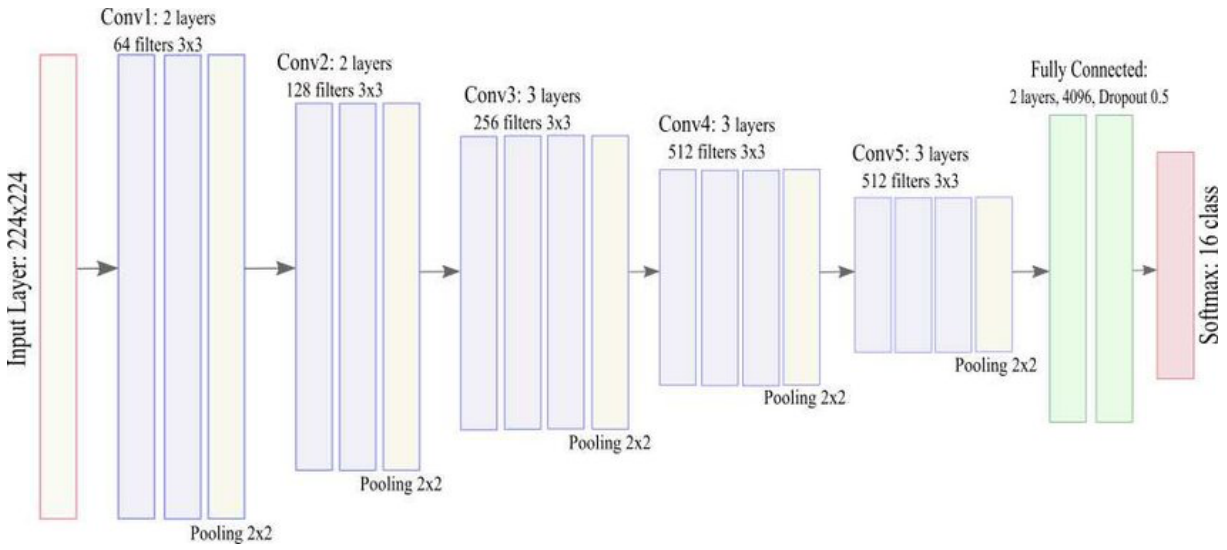
- CNN architecture




```
Step 1800, Training Loss= 699.7183, Training Accuracy= 0.933
Step 1800, Test Loss= 6284.0625, Test Accuracy= 0.640
Step 2000, Training Loss= 537.0546, Training Accuracy= 0.973
Step 2000, Test Loss= 5821.0723, Test Accuracy= 0.680
Step 2200, Training Loss= 479.0638, Training Accuracy= 0.973
Step 2200, Test Loss= 5608.5762, Test Accuracy= 0.720
Step 2400, Training Loss= 412.6966, Training Accuracy= 0.973
Step 2400, Test Loss= 6006.1226, Test Accuracy= 0.720
Optimization Finished! Step:  2400
Testing Accuracy: 0.72
```

transfer learning based on VGG16 achieved classification accuracy at 1.0

- VGG16 architecture




```
from sklearn.linear_model import LogisticRegression as LR

model = LR()
model.fit(X_train_embedding, Y_train)
print('Logistic regression: ')
print('training accuracy: ', model.score(X_train_embedding, Y_train))
print('test accuracy: ', model.score(X_test_embedding, Y_test))
```

```
Logistic regression:
training accuracy:  0.9866666666666667
test accuracy:  1.0
```

```
from sklearn.naive_bayes import GaussianNB

model = GaussianNB()
model.fit(X_train_embedding, Y_train)
print('Naive bayes: ')
print('training accuracy: ', model.score(X_train_embedding, Y_train))
print('test accuracy: ', model.score(X_test_embedding, Y_test))
```

```
Naive bayes:
training accuracy:  1.0
test accuracy:  1.0
```

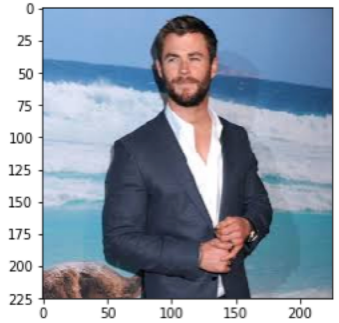
```
from sklearn.svm import SVC

model = SVC()
model.fit(X_train_embedding, Y_train)
print('Support vector machine: ')
print('training accuracy: ', model.score(X_train_embedding, Y_train))
print('test accuracy: ', model.score(X_test_embedding, Y_test))
```

```
Support vector machine:
training accuracy:  0.9733333333333334
test accuracy:  1.0
```

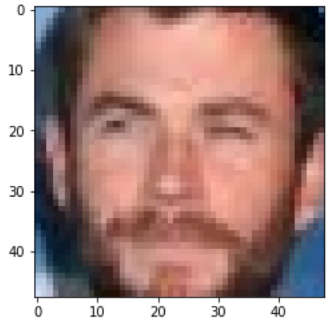
My face recognition application says “Thor really look like Aquaman Arthur”

```
loading image...
```



```
image loaded

Face has been detected
```



```
This face has been recognized as: JasonMomoa with prob 0.261966
Identity: JasonMomoa, Prob 0.261966
Identity: JohnnyDepp, Prob 0.151715
Identity: PatrickWilson, Prob 0.155469
Identity: RyanReynolds, Prob 0.257547
Identity: WillemDafoe, Prob 0.173303
```

```
Now search for the most similar Faces
displaying the top 3 ...
```