Face Recognition through Deep Neural Network

Yang Song

Contents

- Face Recognition, Identification and Verification
- ConvNet Layers
- Implementation of VGG16
- Data augmentation

FaceID
Face Identification:

- **Face Identification**
  - Face ID: Taylor Swift
  - Query Image
  - Training set

Face Verification:

- **Face Verification**
  - Whether they are the same person?
  - Query Images
  - Training set

Face Recognition = Face Identification + Face Verification

A face recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a face database.
Traditional Method

Given Image  
Face Detection  
Feature Extraction  
Face Identification/Verification

- e.g., Landmark detection
- Physical features: the relative position, size, shape of eyes, nose, jaw and etc.
- Skin color
- SIFT or HOG features
- e.g., SVM or a bayesian model
- Classifier/Model

Limitations?

In the real application, there are large variation with face pose, background, illumination and occlusion. It is hard to design a feature extraction method to be robust and discriminative.

Why our human brain can figure it out?

A CNN Network

- Given Image  
- Face Detection  
- Feature Extraction & Face Identification/Verification  
- output

- A CNN Network
Layers used to build ConvNets

- Convolutional Layer
- Pooling Layer
- Fully Connected Layers
- Normalization Layers (e.g., Batch Normalization)
- Activation Function Layers (e.g., RELU Layer)
Convolutional Layer -- Padding

No padding, stride=2
Zero padding, stride=2
Zero padding, stride=1

Convolutional Layer -- Quick Test

Padding | Stride | Width | Height | Depth
--- | --- | --- | --- | ---
Same | 1 | 28 | 28 | 16
Valid | 1 | 26 | 26 | 16
Valid | 2 | 13 | 13 | 16
Same | 2 | 14 | 14 | 16

Output size = ceil(w - k + 2p)/s + 1
VGG Face Network

VGG16 Tensorflow Implementation

https://www.cs.toronto.edu/~frossard/vgg16/vgg16.py

VGG16 Tensorflow Implementation

https://www.cs.toronto.edu/~frossard/vgg16/vgg16.py
Another lightweight implementation of VGG16

TF-Slim is a lightweight library for defining, training and evaluating complex models in TensorFlow.

```python
with slim.arg_scope([slim.conv2d], weights_initializer=tf.truncated_normal_initializer(stddev=0.1), biases_initializer=tf.zeros_initializer):
    net = slim.conv2d(images, [6, 3, 3], padding='SAME', scope='conv1_1', 
                      weights_initializer=tf.glorot_normal_initializer(0), 
                      biases_initializer=tf.zeros_initializer)
```

Another lightweight implementation of VGG16

TF-Slim is a lightweight library for defining, training and evaluating complex models in TensorFlow.

```python
with slim.arg_scope([slim.conv2d], weights_initializer=tf.glorot_normal_initializer(0), biases_initializer=tf.zeros_initializer):
    net = slim.conv2d(images, [6, 3, 3], padding='SAME', scope='conv1_1', 
                      weights_initializer=tf.glorot_normal_initializer(0), 
                      biases_initializer=tf.zeros_initializer)
```
Data Augmentation

Translation
Flipping
Rotation
Random Cropping
Compression

Avoid overfitting!

VGG16 by TF-Slim

net = slim.conv2d(net, 4896, [7,7], padding='VALID', scope='fc6')