## 一、背景综述

人脸表情识别（FER,face expression recognition），主要是基于静态图像的人脸，表情包括7大类：生气、害怕、厌恶、开心、悲伤、惊讶和正常表情。

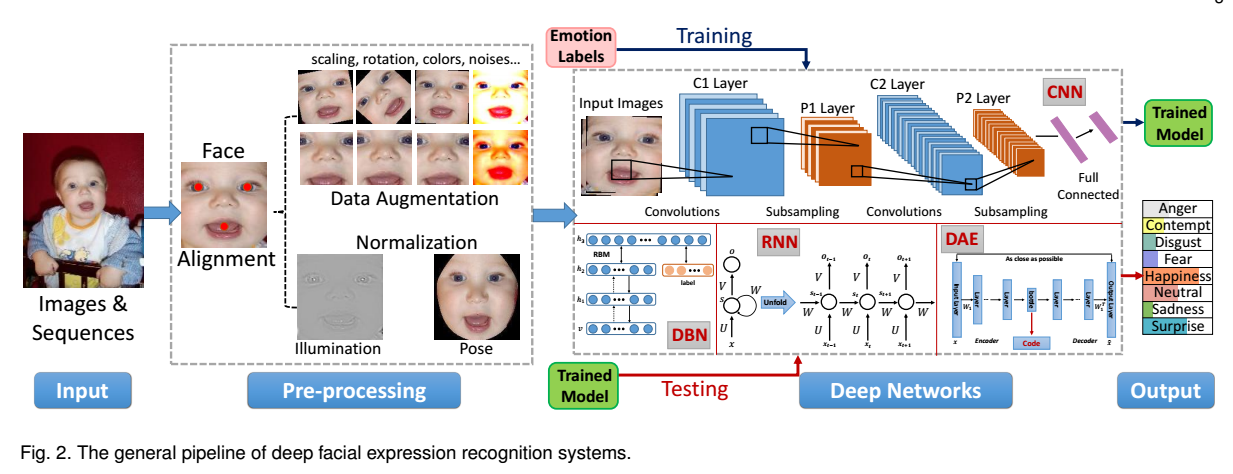
目前主流的方式主要有：

（1）传统的手工特征（LBP, LBP-TOP等）

（2）浅层学习（SVM、Adaboost等）

（3）深度学习（CNN，DBN，RNN等）

其中，传统的手工特征和浅层学习特征的抗干扰能力不是很好，例如光照变换，不同头部姿态和面部遮挡等因素的影响。最近的state-of-art研究基本都是基于深度学习技术。



基于Deep learning的FER，主要分3个过程：

### （1）预处理

提取表情特征前，排除掉跟脸无关的干扰，一般处理有：人脸检测，人脸对齐，数据增强和人脸归一化。

* **人脸检测**有很多方法，不做赘述。
* **人脸对齐**，主要是指根据人脸，检测出人脸定位点landmarks，主流算法有：IntraFace的SDM算法，49个点，其他如Mot, DRMF, Dlib, MTCNN, TinyFaces。
* **数据增强**

主流的离线数据增强，包含随机扰动，变换（旋转，平移，翻转，缩放，对齐），噪声添加如椒盐噪声，斑点噪声，以及亮度，饱和度变化，以及在眼睛之间添加2维高斯随机分布的噪声。同时，还有其他的如GAN生成脸，3DCNN辅助AUs生成表情。但是，GAN生成的脸，是否对网路模型有性能提升，还没有证实。

在线数据增强，包含Crop,水平翻转等，主要的意思是指，在预测时，可以一次性对测试数据进行Crop，翻转等操作生成多张类似的测试图，然后每张测试图预测出的输出做均值，这主要是基于随机扰动训练的模型，需要在测试时计算均值的原因而定的。

* **人脸归一化**

主要是指亮度归一化和姿态归一化（就是人脸对齐拉正）。

**亮度归一化**，不仅包含亮度调整，也包含了对比度调整，常见的对比度调整方式 有直方图归一化，DCT归一化，DoG归一化，相关论文证明直方图归一化效果最稳定，适应各种网络模型，也有论文证明，全局对比度归一化，局部归一化，直方图归一化，三种方法中，全局对比度归一化和直方图归一化效果最好。所以，建议直方图归一化联合亮度归一化一起。

**姿态归一化**，其实这个是人脸项目中影响最大的了。目前大多数还是在小角度内，2D 的landmark对齐，比较可靠的方向是3D landmark,有通过图像和相机参数估计出来的，也有通过深度传感器测量然后计算出来的。比较新的估计模型有：FF-GAN,TP-GAN,DR-GAN等。

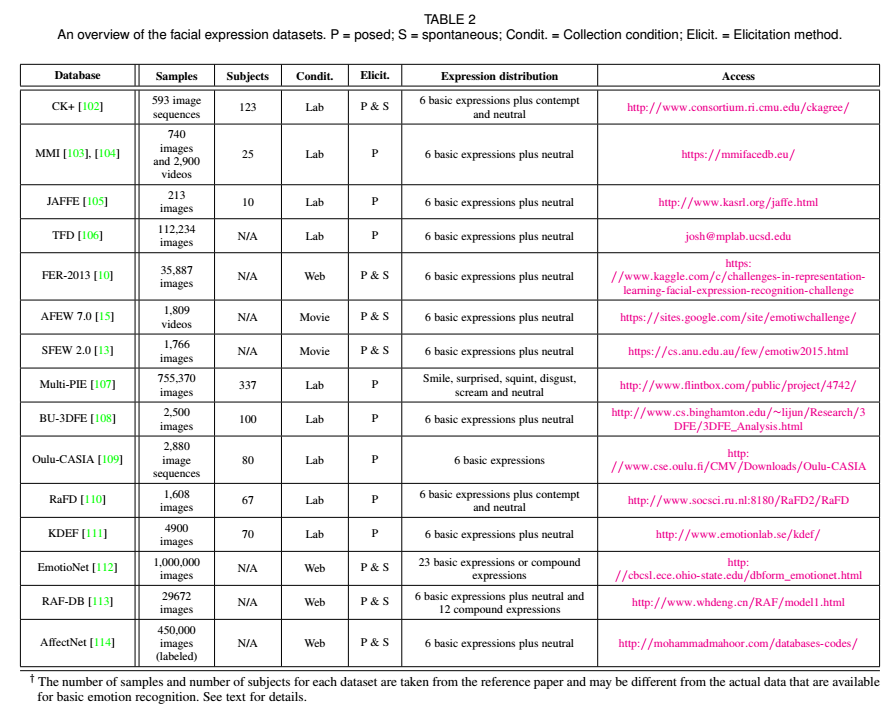
### （2）深度特征学习

主要是基于CNN的网络模型，具有非常高性能的特征表达能力。特别适合基于图像的分类问题，另外一个基于DBN网络的特征提取，以及DAE转门用作特征提取。最后，还有一部分是基于序列建模的RNNs，如LSTM等。

### （3）面部表示分类

基于深度学习，直接学习特征，预测概率，也可以把学习的深度特征，用SVM等浅层分类器进行分类。

### （4）FER涉及到数据集。



## 二、算法选择

使用CVPR 2018的论文《A Compact Deep Learning Model for Robust Facial Expression Recognition》提出的CNN简化模型来训练，训练数据库使用CK+，框架采用tensorflow，工程应用使用Opencv的DNN模块加载训练好的tensorflow网络模型进行表情分类。

### （1）数据

* **CK+数据库有4个文件夹**

cohn-kanade-images：人脸表情灰度图像，**大小为640x490**，每种类别都是由正常到夸张的表情程度。

Emotion：表情类别，**0-中性 1-愤怒 2-蔑视 3-厌恶 4-恐惧 5-高兴 6-悲伤 7-惊讶**。有一些图像没有给出类别是因为该图像的表情难以人为分类。

FACS：for each sequence (593) there is only 1 FACS file, which is the last frame (the peak frame). Each line of the file corresponds to a specific Action Unit and then the intensity。

Landmarks：每个人脸图像的68个landmarks坐标。

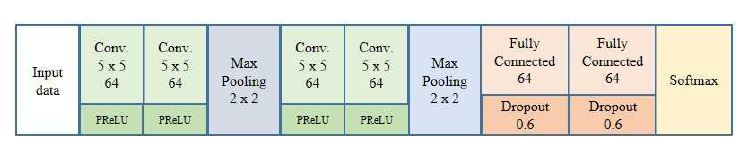
在我们的需求中，只需要表情图和对应的表情类别作为训练数据。

* **预处理**

根据CK+给出的landmarks截取人脸区域，计算crop\_img的大小L = a x max(dv, dh)，其中dv是水平反向的landmarks最大距离，dh是垂直方向landmarks最大距离，a设置为1.05，以鼻子为中心crop出图像，再resize为120x120大小。

### 训练

由于CK+有效样本数才1000多，为了防止过拟合，通过镜像翻转、左右旋转8度、左右旋转14度，图形直方图均衡这4个操作将样本扩展为20000多，其中16000用做训练，4400用做验证，1000用于测试。注意事项：训练样本中要包含全部类别的样本。训练输入数据是以鼻子为中心截取96x96的大小。训练网络模型图：



### （3）工程应用

* h5转pb：h5是Keras接口保存的模型文件格式，转成pb才能给opencv的dnn读取

问题1：opencv dnn在readNetFromTensorflow会报错Parse GraphDef。

解决：调用tf.train.write\_graph()时，必须设置参数as\_text=Fasle，不然

问题2：InputLayer无法识别

解决：使用Conv2D中加input\_shape=()和batch\_size。

* opencv dnn模块调用pb

问题1：OpenCV Error: Unspecified error (Unknown layer type StridedSlice in op strided\_slice) in populateNet。

问题2：(Unknown layer type PlaceholderWithDefault in op dropout/keras\_learning\_phase)

问题3：const\_layers.insert(std::make\_pair(name, li)).second) in addConstNodes

解决步骤：

1. 训练生成fer\_model.h5后，调用h5topb.py转成fer\_model.pb

2、调用tensorflow优化脚本对pb文件做优化：

|  |
| --- |
| python ~/tensorflow/tensorflow/python/tools/optimize\_for\_inference.py --input=fer\_model.pb --output=fer\_model\_opt.pb --input\_names=conv2d\_input --output\_names=dense\_2/Softmax |

3、生成模型图描述文件fer\_model\_opt.pbtxt，将Const nodes清除掉，解决问题3：

|  |
| --- |
| import tensorflow as tf  # Read the graph.  with tf.gfile.FastGFile('opt\_model.pb') as f:  graph\_def = tf.GraphDef()  graph\_def.ParseFromString(f.read())  # Remove Const nodes.  for i in reversed(range(len(graph\_def.node))):  if graph\_def.node[i].op == 'Const':  del graph\_def.node[i]  for attr in ['T', 'data\_format', 'Tshape', 'N', 'Tidx', 'Tdim', 'use\_cudnn\_on\_gpu', 'Index', 'Tperm', 'is\_training', 'Tpaddings']:  if attr in graph\_def.node[i].attr:  del graph\_def.node[i].attr[attr]  # Save as text.  tf.train.write\_graph(graph\_def, "", "model.pbtxt", as\_text=True) |

4、编辑fer\_model\_opt.pbtxt文件，将节点类型为

flatten/Shape，

flatten/strided\_slice，

flatten/Prod，

flatten/stack清除掉，替换节点flatten/Reshape如下，解决问题1：

|  |
| --- |
| 将节点  node {  name: "flatten/Reshape"  op: "Reshape"  input: "max\_pooling2d\_1/MaxPool"  input: "flatten/stack"  }  改为  node {  name: "flatten/Reshape"  op: "Flatten"  input: "max\_pooling2d\_1/MaxPool"  } |

5、编辑fer\_model\_opt.pbtxt文件，将节点类型为

dropout\_1/keras\_learning\_phase,

dropout\_1/cond/Switch,

dropout\_1/cond/mul/Switch,

dropout\_1/cond/mul,

dropout\_1/cond/dropout/Shape,

dropout\_1/cond/dropout/random\_uniform/RandomUniform,

dropout\_1/cond/dropout/random\_uniform/sub,

dropout\_1/cond/dropout/random\_uniform/mul,

dropout\_1/cond/dropout/random\_uniform,

dropout\_1/cond/dropout/add,

dropout\_1/cond/dropout/Floor,

dropout\_1/cond/dropout/div,

dropout\_1/cond/dropout/mul,

dropout\_1/cond/Switch\_1,

dropout\_1/cond/Merge清除掉，替换节点input: "dropout..."为input:"dense\_1/BiasAdd"，解决问题2

【Reference】:

<https://blog.csdn.net/App_12062011/article/details/80504960>

<http://answers.opencv.org/question/183682/opencv-dnn-import-dropout-layer-error-after-finetuning-keras-vgg16/>

<http://answers.opencv.org/question/183507/opencv-dnn-import-error-for-keras-pretrained-vgg16-model/>