媒体与认知课程实验：人脸姿态估计

无 35	无 36	无 36	无 37
陈馨瑶	李文硕	李思涵	刘家硕
2013011166	2013011177	2013011187	2013011212

2016 年 6 月 12 日

目录

1 实验背景与要求

研发一个人脸姿态估计应用程序，程序能根据人脸图像给出角度估计。最后的评测指标为估计的角度与真实角度的平均误差。

输入输出要求

1. 输入：待测图片路径文件 ImgList.txt，待测图片关键点位置注释文件所在路径。
2. 输出：在当前文件夹下，输出名为 FacePosition.txt 的文件，文件每行内容为一张待测图片的路径，空格，估计的角度 $(-90 \sim 90)$

提供的内容

1. 数据库：共包含 938 个人，每个人共 7 张不同角度的图像，其角度信息包含在文件名中。
2. 人脸检测程序
3. 人脸关键点定位程序

2 实验原理与思路

输入为图像及其坐标点，根据其给出连续的角度，是一个**回归**问题。因此，可以采用的方案有

- 最小二乘法
- 广义线性回归
- 高斯回归
- 神经网络
- .....

2.1 数值回归

2.1.1 基本模型

回归分析研究的是变量与变量间的关系，其中一个变量称为自变量 $x \in R^d$，另一个变量称为因变量 $y \in R$。假设两者存在如下关系

$$y = f(x) + e$$

其中 e 为表示误差的随机变量，$f(x)$ 称为回归函数。

回归函数好坏的评价，一般用在一组有别于训练数据的测试数据 $\tau = \{(x_{*,j}, y_{*,j})|j = 1, ..., m\}$ 上，计算出的均方误差 (Mean Squared Error, MSE) 来衡量。

$$MSE = \frac{1}{M}\sum_{j=1}^{m}(y_{*,j} - f(x_{*,j}))^2$$

测试数据上的均方误差越小，表明两变量的关系拟合的越好。

2.1.2 标准线性模型的最小二乘解

标准线性模型为 $f(x) = x^T\omega$，由于

$$MSE(\omega) = \frac{1}{m}\sum_{j=1}^{m}(y_j - x_j^T\omega)^2$$

令其对 ω 的偏导数为 0，则有

$$W = (XX^T)^{-1}Xy$$

则要预测的结果为

$$y_* = X_*^T W$$

2.1.3 多项式基函数的广义线性回归

对于带有高斯白噪声的线性模型，$y = x^T\omega + \varepsilon, \varepsilon \sim N(0, \sigma_n^2)$。此处的 σ_n 我们可以把线性模型的最小二乘方法得到的方差当作噪声。

在贝叶斯方法中, 我们假定拥有先验知识

$$\omega \sim N(0, \Sigma_p)$$

那么我们得到了一个高斯过程。根据学过的知识可以得到

$$p(y|X, \omega) = N(X^T\omega, \sigma_n^2 I)$$

再由贝叶斯公式

$$p(\omega|y, X) = \frac{p(y|X, \omega)p(\omega)}{p(y|X)}$$

可以得到

$$p(\omega|X, y) \sim N(\frac{1}{\sigma_n^2}A^{-1}Xy, A^{-1}),\ A = \sigma_n^{-2}XX^T + \Sigma_p^{-1}$$

计算出 ω 的概率后可以以其均值作为权重来进行预测。

以上方法都只能用于线性的情况进行预测, 为了对非线性情况预测我们可以引入一组基函数 $\phi(x) = (1, x, x^2, x^3, ...)^T$，将输出 x 映射到该基函数构成的空间中, 即可进行线性预测。则

$$f(x) = \phi(x)^T\omega$$

将标准线性模型中的 x 替换成 $\phi(x)$ 即可。

2.1.4 高斯过程回归分析

对于上文的推导, 其实我们可以给出更一般的形式。令 $K(C,D)=\Phi(C)^T\Sigma_p\Phi(D)$，则有

$$f_*|X,y,X_* \sim N(\bar{f}_*, cov(f_*))$$
$$\bar{f}_* = E[f_*|X,y,X_*] = K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}y$$
$$cov(f_*) = K(X_*,X_*) - K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}K(X,X_*)$$

这是高斯过程回归分析的基本形式, 其中 K 函数即协方差函数，其参数 (超参数，Hyperparamaters) 的选择对于回归分析的性能有很大影响。

2.1.5 超参数选择

我们进行回归分析的目的是根据已有的输入 X 尽可能好的预测输出结果 y，所以我们希望能有最大的边际似然函数 (Marginal Likelihood)$p(y|X)$，为了计算方便我们取其 log 形式。

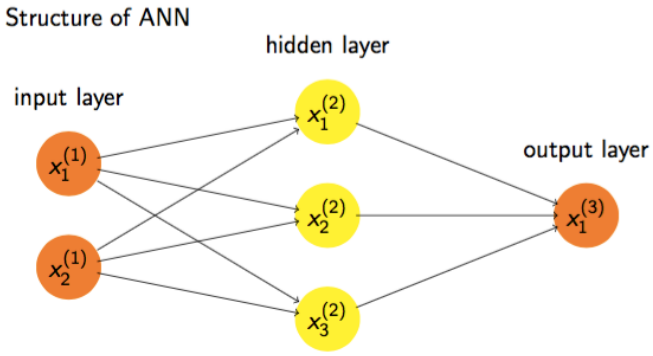
$$p(y|X) = \int p(y|f,X)p(f|X)\,df$$
$$\log p(y|X) = -0.5y^T(K+\sigma_n^2 I)^{-1}y - 0.5\log|K| - 0.5n\log 2\pi$$

对带有超参数 θ 的，使其边际似然函数最大，该问题为凸优化问题。可对其超参数 θ 求偏导, 用梯度下降法等凸优化方法寻找最优超参数。

2.2 人工神经网络 (ANN)

人工神经网络 (Artificial Neural Networks) 在一定程度上受到了生物学的启发，利用大脑神经元的特征构建出人工神经元来进行数学建模或者计算。神经网络是一种运算模型，由大量的结点和结点之间的连接组成, 它除了输入结点以外, 每个结点都代表着一种特殊的输出函数，称为激励函数。每两个节点间的连接都代表一个对于通过该连接信号的加权值，称之为权重 (weight)，这相当于人工神经网络的记忆。

ANN 的结构 (多层感知器) 如下：



2.2.1 BP 学习算法概述

多层前馈网络的反向传播 (BP) 学习算法，简称 BP 算法，是监督学习算法，它是梯度下降法在多层前馈网络中的应用。

BP 学习算法是已知输入/输出样本进行监督学习，由正向传播和反向传播组成:

- 正向传播是输入信号从输入层经隐含层，传向输出层，若输出层得到了期望的输出，则学习算法结束；否则，转至反向传播。
- 反向传播是将误差（样本输出与网络输出之差）按原联接通路反向计算，由梯度下降法调整各层节点的权值和阈值，使误差减小。

记算法的输入输出样本为 u_q, g_q, $q = 1, 2, ..., Q$。网络训练的目的，是使对每一个输入样本，调整网络参数，使输出均方误差 $J(\omega) = \frac{1}{2} \parallel g - y \parallel^2 = \frac{1}{2} \parallel e \parallel^2$ 最小。

考虑迭代算法，设有初始权值为 ω_0，k 时刻权值为 ω_k，则使用泰勒级数展开，有：

$$J(\omega_{k+1}) = J(\omega_k) + [\frac{dJ}{d\omega} \mid_{\omega=\omega_k}]^T \Delta\omega_k + ...$$

为了使 J 最小，最直接的办法就是选择 $\delta\omega_k = -\alpha \frac{dJ}{d\omega} \mid_{\omega=\omega_k}$ $\alpha > 0$。这样每一步都能保证 $J(\omega_{k+1}) \leq J(\omega_k)$。这就是梯度下降算法，也是 BP 学习算法的基本思想。

BP 学习算法步骤：

1. 设置初始权系数 ω_0 为较小的随机非零值
2. 给定输入/输出样本对，计算网络输出，完成前向传播
3. 计算目标函数 J。如 $J < \varepsilon$，训练成功，退出；否则转入步骤四
4. 反向传播计算：由输出层将误差反向传播，按梯度下降法逐层调整权值。

2.2.2 前向传播

对于当前样本，期望输出为 g，输入为 $u = (u_1, u_2, ..., u_j)^T$

隐含层输出：对于第 i 个神经元,

$$y_i^{(1)} = f((\omega_i^{(1)})^T u - \theta_i^{(1)}) = f(\sum_{j=0}^{n} \omega_{ij}^{(1)} u_j)$$

$$\omega_{i0}^{(1)} = -\theta_i^{(1)};\ u_0 = 1$$

$f = \frac{1}{1+e^{-x}}$，选取为 Log Sigmoid 函数

输出层：对于第 r 个神经元,

$$y_r = f((\omega_r^{(2)})^T y^{(1)} - \theta_r^{(2)}) = f(\sum_{t=0}^{h} \omega_{rt}^{(2)} \cdot y_t^{(1)})$$

$$\omega_{r0}^{(2)} = -\theta_r^{(2)};\ y_0^{(1)} = 1$$

2.2.3 误差反向传播

$\Delta\omega_k = -\alpha\frac{dJ}{d\omega}\mid_{\omega=\omega_k}$，因此需要求 $\frac{dJ}{d\omega}\mid_{\omega=\omega_k}$

首先考虑输出层第 r 个神经元与隐含层第 t 个神经元之间的权系数 $\omega_{rt}^{(2)}$，根据链式求导法则：

$$\frac{\partial J}{\partial \omega_{rt}^{(2)}} = [\frac{\partial J}{\partial e}]^T \frac{\partial e}{\partial \omega_{rt}^{(2)}}$$

由于

$$J(\omega) == \frac{1}{2} \parallel g - y \parallel^2 = \frac{1}{2} \parallel e \parallel^2$$
$$\frac{\partial J}{\partial e} = e$$

注意到 $\omega_{rt}^{(2)}$ 仅和 $y_r(e_r)$ 有关，因此

$$\frac{\partial J}{\partial \omega_{rt}^{(2)}} = -e_r\frac{\partial y_r}{\partial \omega_{rt}^{(2)}} = -e_r y_r(1-y_r)\cdot y_t^{(1)}$$
$$\Delta\omega_{rt}^{(2)}(k) = -\alpha\frac{dJ}{d\omega_{rt}^{(2)}}\mid_{\omega_{rt}^{(2)}=\omega_{rt}^{(2)}(k)} = \alpha y_r(1-y_r)\cdot y_t^{(1)} e_r$$

对隐含层，注意到 $\omega_{ij}^{(1)}$ 仅和 $y_i^{(1)}$ 有关

$$\frac{\partial J}{\partial \omega_{ij}^{(1)}} = [[\frac{\partial J}{\partial e}]^T \frac{\partial e}{\partial y_i^{(1)}}]\frac{\partial y_i^{(1)}}{\partial \omega_{ij}^{(1)}}$$
$$\frac{\partial J}{\partial e} = e$$
$$\frac{\partial e}{\partial y_i^{(1)}} = -\frac{\partial y}{\partial y_i^{(1)}} = -[\frac{\partial y_1}{\partial y_i^{(1)}}, ..., \frac{y_m}{y_i^{(1)}}]^T$$

可以推出

$$[\frac{\partial J}{\partial e}]^T \frac{\partial e}{\partial y_i^{(1)}} = -\sum_{r=1} m y_r(1-y_r)\cdot\omega_{ri}^{(2)} e_r$$
$$\frac{\partial y_i^{(1)}}{\partial \omega_{ij}^{(1)}} = y_i^{(1)}(1-y_i^{(1)})u_j$$

综合上述结果：

$$\Delta\omega_{ij}^{(1)}(k) = -\alpha\frac{dJ}{d\omega_{ij}^{(1)}}\mid_{\omega_{ij}^{(1)}=\omega_{ij}^{(1)}(k)} = \alpha[\sum_{r=1}^{m} y_r(1-y_r)\cdot\omega_{ri}^{(2)} e_r]\cdot y_i^{(1)}(1-y_i^{(1)})\cdot u_j$$

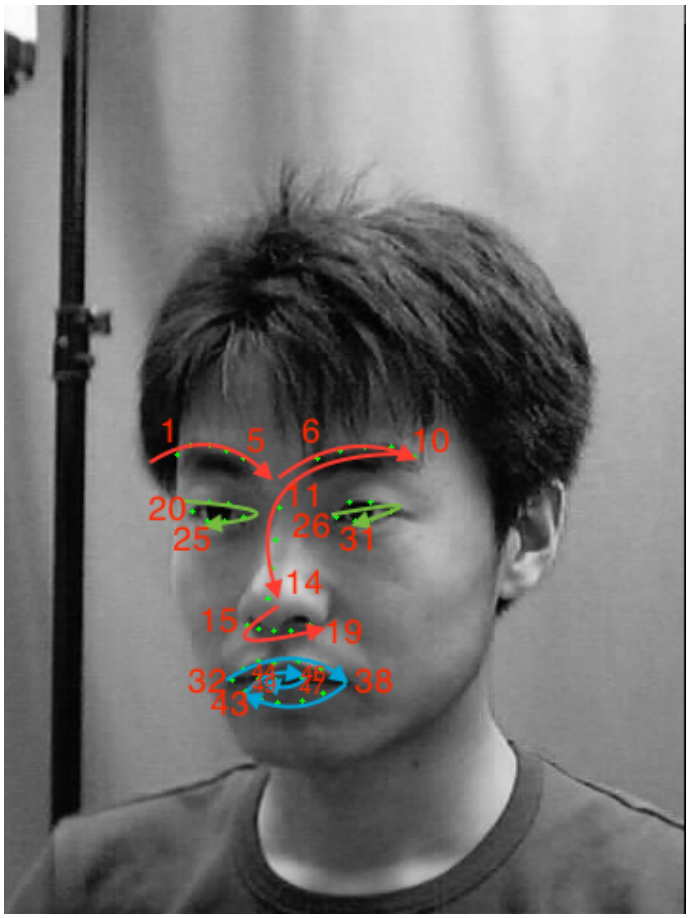
当隐含层多于 1 层，可依此类推。

3 实验设计与步骤

实验中，我们分别采取了两类方法来估计人脸姿态：数值回归和 ANN 神经网络。具体说明如下：

3.1 数值回归与 ANN

人脸关键点检测程序检查出的关键点及其顺序如下：



对数值回归，我们设计了两种不同方式：第一种是直接用检测出的 49 个关键点的横纵坐标（即共 98 个值）进行回归，另一种是先分析图片提取特征，利用提取出的特征进行回归。

对 ANN 神经网络，输入值为：49 个关键点的横纵坐标和人脸双眼中心的横纵坐标共 102 个值（可能经过预处理），隐藏层数目为 2，输出的结果为人脸的角度。

3.2 特征提取算法

分析不同角度图片的特点，可以看到十分明显的一点就是左右两侧脸颊的比例不同。据此，我们共提取了五种不同特征以用于分析，如下（x_i 表示第 i 个关键点的横坐标，y_i 作为第 i 个点的纵坐标，f_i 表示第 i 个特征）：

1. 眉长比：以一条眉毛的最左和最右侧点的横坐标的差值作为眉长，从而
$$f_1 = \frac{x_5 - x_1}{x_{10} - x_6}$$
2. 眼长比：和眉长比类似，以一个眼睛的最左和最右侧点的横坐标的差值作为眼长，从而
$$f_2 = \frac{x_{23} - x_{20}}{x_{29} - x_{26}}$$
3. 鼻子夹角：鼻子夹角指鼻子的斜率对应的夹角。选取第 11 和第 14 个点连成的直线与水平方向的夹角作为鼻子夹角特征，计算方式如下：
$$f_3 = arctan\frac{y_{14} - y_{11}}{x_{14} - x_{11}}$$
4. 鼻长比：鼻长比的计算利用的是第 15 到第 19 个关键点，同样也只考虑了横坐标的信息，如下：
$$f_4 = \frac{x_{17} - x15}{x_{19} - x_{17}}$$
5. 嘴长比：由于嘴的中点并未直接给出，故而我们首先用点 35、45、48、41 的横坐标取平均值来作为嘴的中线横坐标，不妨记为 x，再利用 x 和嘴两侧的点的横坐标值计算嘴长比，如下：
$$x = \frac{x_{35} + x_{45} + x_{48} + x_{41}}{4}$$
$$f_5 = \frac{x - x_{32}}{x - x_{38}}$$

3.3 预处理算法

对采用全部关键点坐标的方法，除了直接输入不作处理，我们还尝试了对关键点坐标进行一定预处理后再回归出角度的方式。采用的预处理方式有如下几种：

1. 高度归一化：即将不同图片的人脸高度进行归一化。以点 47 和 49 纵坐标的均值，和点 3 和 8 纵坐标的均值的差作为人脸高度，即人脸高度 $= \frac{|y_3 - y_8|}{2} - \frac{|y_{49} - y_{47}|}{2}$，对其进行归一化。
2. 去重心：分别求 x 和 y 方向的均值，再用关键点的横坐标和纵坐标分别减去各自方向上的均值作为神经网络的输入。
3. 去重心且高度归一化：即对图片同时采用以上两种方法预处理。

4 实验结果与分析

4.1 数值回归

4.1.1 用关键点做回归

直接采用 49 个关键点的横纵坐标（即 98 个坐标值）进行回归。对高斯回归，由于其计算复杂度较高且可能出现奇异值，于是并未采用。具体的结果对比如下：

	最小二乘法	广义线性回归（五阶）	高斯回归 (PCA)
MSE	9.0260	3.0153	-
平均误差（单位：度）	3.004	1.736	-

结果的误差还是很小的，且有准确度广义线性回归（五阶）> 最小二乘法。

4.1.2 用特征做回归

用提取出的特征进行数值回归的结果对比如下：

	最小二乘法	广义线性回归（五阶）	高斯回归
MSE	99.4958	18.1758	16.2148
平均误差（单位：度）	9.975	4.263	4.027

其中需要进一步说明的是高斯回归，通过调整，我们得出目前为止最优的参数为：

$$l = 0.5,\ f = 1250,\ n = 0.28$$

此外，以上所有算法的运行结果都与样本的选取存在一定相关。以高斯回归为例，如果取前 5000 个样本训练，剩下的测试，最好的 MSE 为 16.2148；如果随机选取，那么最优的 MSE 为 18.2436，算下来平均误差约为 4.27°。但无论如何，在采用特征进行数值回归的三种方法中，都有高斯回归准确率 > 广义线性回归（五阶）> 最小二乘法的关系。根据最终算出的权重，在特征中，对结果贡献最大的是第三和第四个特征。

4.1.3 两种数值回归方法的对比

将两种方法的结果都列入下表中 (MSE)：

	最小二乘法	广义线性回归（五阶）	高斯回归
关键点回归	9.0260	3.0153	-
特征回归	99.4958	18.1758	16.2148

从上表中可以看出：总体来看，利用关键点进行回归的准确率要远高于利用特征进行回归，这说明我们所提取的特征还并不能完全反映人脸角度的变化，尚具有改进的空间。

4.2 神经网络

首先需要说明的是我们的训练集、验证集 (Validation) 和测试集的选取比重，分别是：训练集 70%，验证集 15%，测试集 15%。

不采用和分别采用不同方式的预处理得到的结果如下表所示：

	不作处理	高度归一化	去重心	去重心 + 高度归一化
平均误差（单位：度）	1.014	2.612	3.134	3.588

上表的结果似乎与我们的预期不符，以为随着加入预处理，结果却越来越差。实际上这是因为数据本身含有系统信息，即角度基本固定在 7 个值，且他们对应的高度不同，因此坐标中也蕴含了一定的信息。所以虽然以上的过程实际上增大了程序的普适性，但对这个数据集本身的表现变差了。

不同处理方式作出的拟合曲线如下：

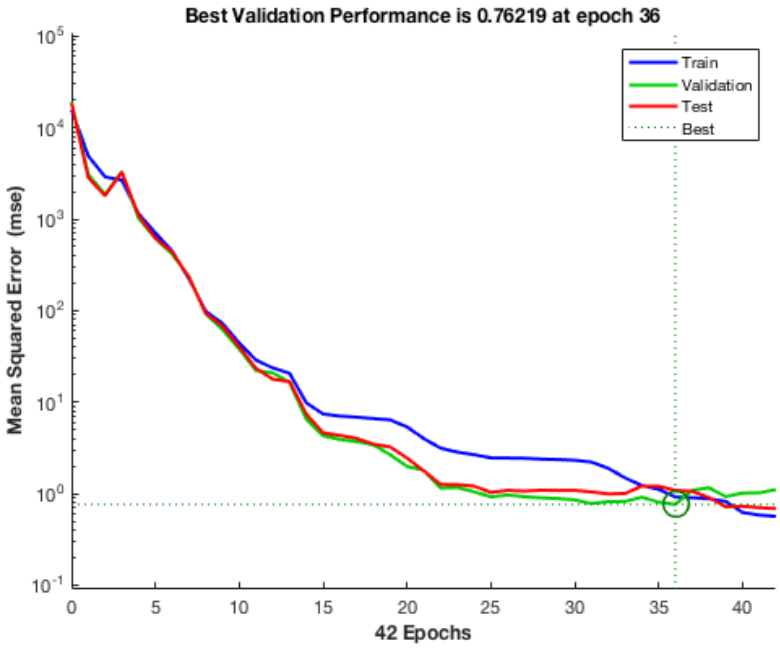


图 1: 原始结果（未采取任何预处理方式）

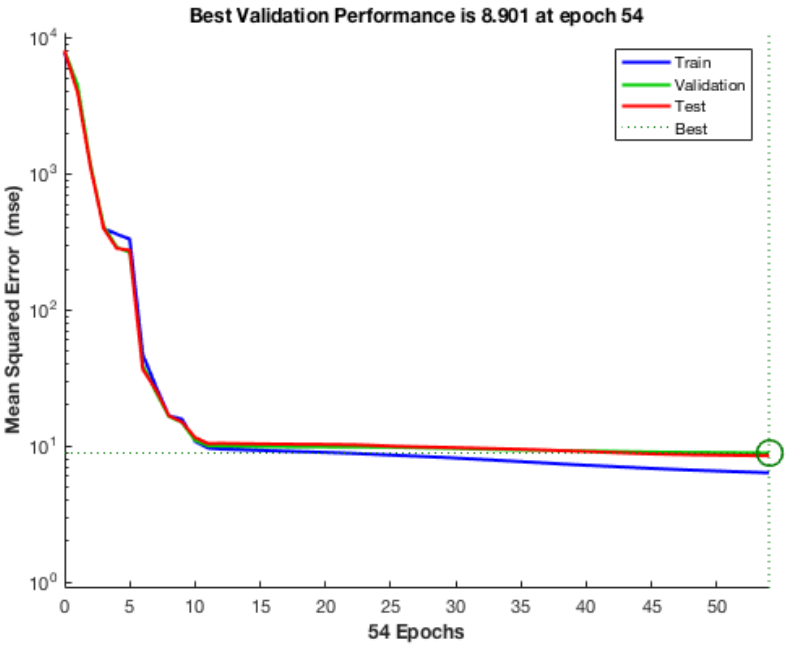


图 2: 高度归一化后的结果

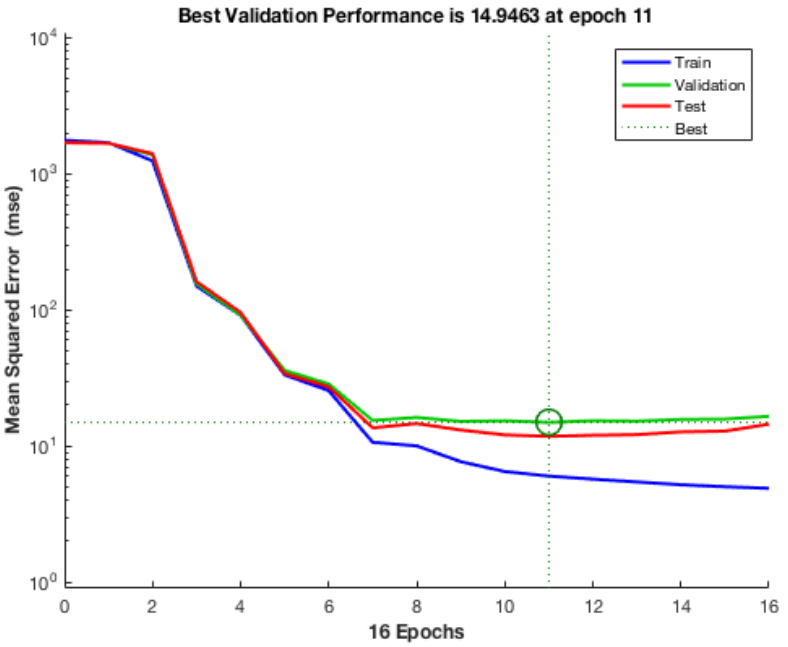


图 3: 高度归一化 + 去重心后的结果

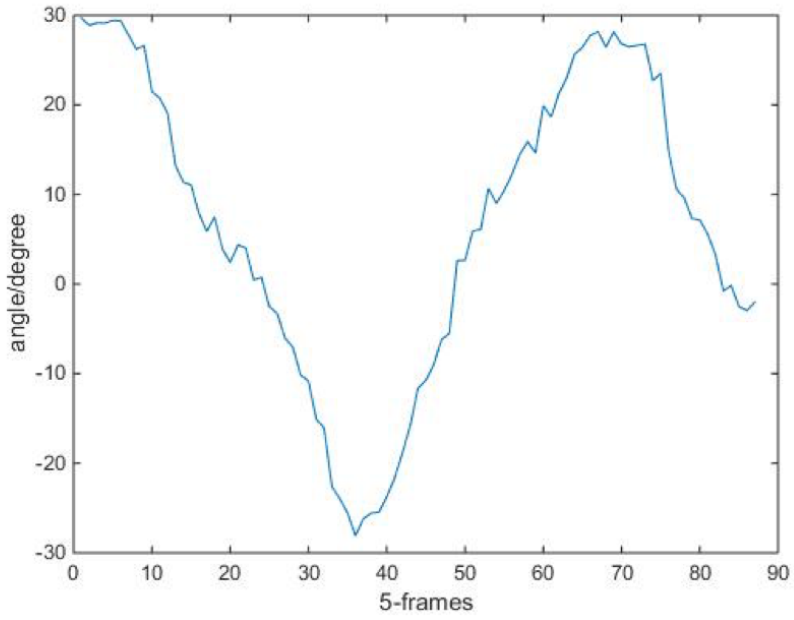
从上面三幅图的对比中可以清楚地看出：原始结果更好，但存在明显的**过拟合**情况。据我们分析可能的原因如下：

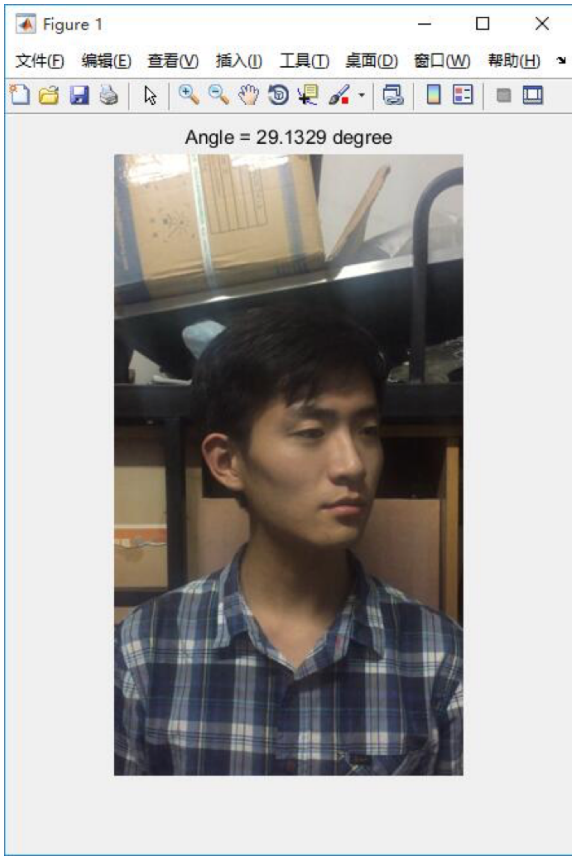
如上面分析所说，如果不做任何处理，数据集本身可能包含系统信息，这一点在训练集、验证集和测试集上是相同的，因此越在该数据集上迭代训练，在该数据集上的表现就会越好，但相应拿到其他数据集上的效果就会变差。采用预处理后，过拟合现象得到了明显改进，这说明算法的普适性也在提高。

5 算法应用：视频流的读取和处理

为了检测结果的可实践性，我们将我们的算法应用到视频流的读取和处理中，对每一帧分析并显示人脸的角度。忽略人脸关键点检测程序的弹窗，在人脸缓慢旋转的前提下，基本可以达到实时的效果。

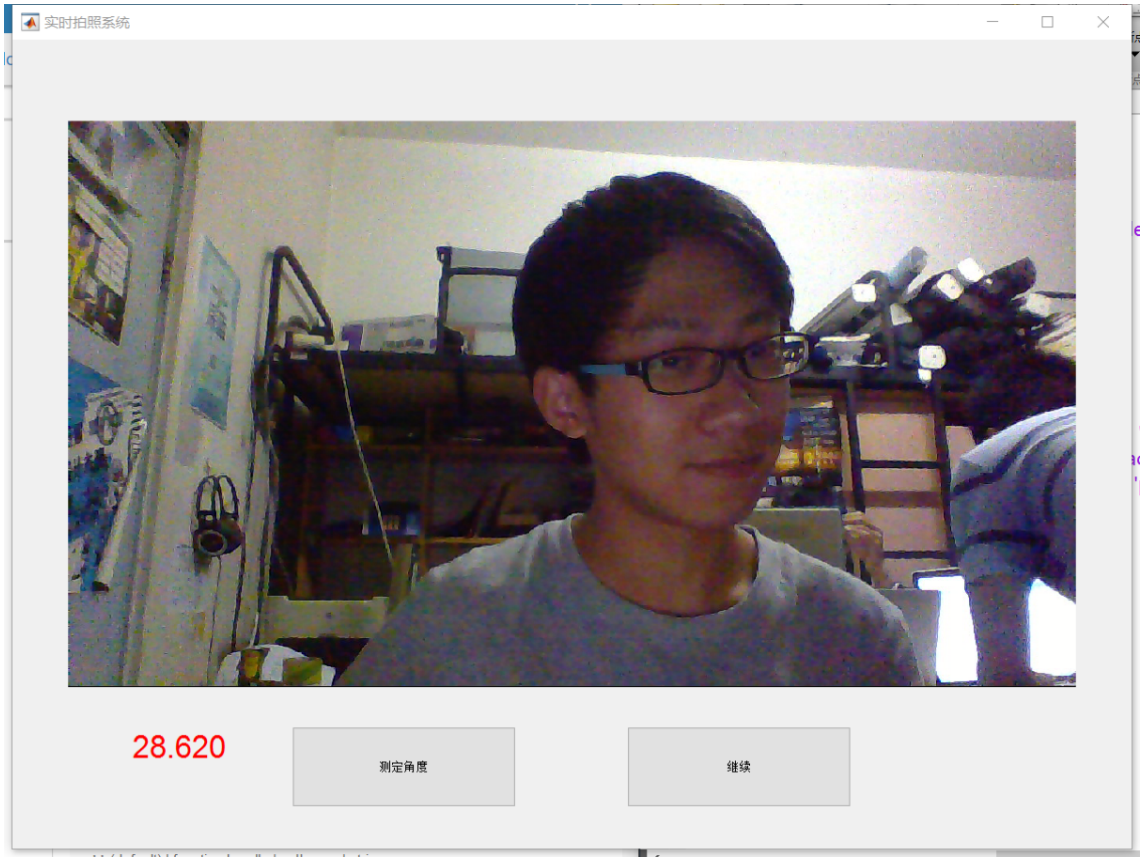
录制一段视频，视频中的人脸在 $-30°$ 到 $30°$ 的范围内缓慢旋转，每 5 帧检测并输出一次结果，得到的输出曲线如下：





可以看到，所作出的曲线的趋势和人脸的运动趋势是一致的，不足之处是曲线存在毛刺，不够平滑。

此外，我们还为这个可以实时读取并判断人脸角度的程序写了一个 Matlab GUI 界面，通过调用系统摄像头，在每一次按键时都可以估计出当前电脑前的人的脸部角度，运行截图如下：



6 总结

在本次实验中，我们讨论了多种不同算法在人脸姿态估计方面的应用，并比较了它们的结果，且根据实际问题进行了一定优化，最终结果的误差也是较小的。但同时，在本次实验的过程中依然存在以下不足之处：

- 在数值回归方面，高斯回归的参数是凭经验设定和手动调整的，因此可能不是最优。此外我们所选取的特征并未经过论证，实际上贡献较大的两个特征都是鼻子的特征，其他特征权值较小，可能需要调整。
- 在神经网络方面，我们虽然进行了一定的处理，但图片间可能还有一定的相关性，所以可能存在较严重的过拟合。此外神经网络的节点数等参数也只能根据经验设定，可能无法达到最优。
- 此外，在实验设计层面，现在用的方法都是基于关键点的，但老师给出的关键点程序容易崩溃且没有源码。进一步也可以考虑不带关键点直接使用 CNN 的方法。

综上所述，在人脸姿态估计这一问题上，很难找出一种“最优”的解法，我们只能在大量实验的基础上一步步优化，力图让我们的算法逼近最优的结果。

7 附：程序运行说明与文件清单

7.1 程序运行说明

7.1.1 命令行调用

main <command> [<args>]
main train [img_list [label_dir [method [model_file [unify]]]]]
main test [img_list [label_dir [method [model_file [unify]]]]]

- img_list: 图片路径文件，默认为 ImgList.txt。
- label_dir: 关键点文件所在路径，默认为 label。
- method: 采用的方法，可能的取值有
  - ANN (默认)
  - GPR
  - poly
  - LS
- model_file: 输入/输出模型文件名，默认与 method 相同。若没有后缀名，则会自动补上”.mat” 后缀。
- unify: 是否对坐标去均值 & 高度归一化，可能的取值有：
  - false (默认)
  - true

main video [filename]
进行视频监测

- filename: 视频文件路径，默认为 detect/speak.m4v。

7.1.2 cam 模式及神经网络训练的说明

在用 MATLAB 调用摄像头时需要进入 cam 模式，关于 cam 模式所需安装包的说明如下：

1. use MATLAB Support Package Installer to install DCAM Hardware.(login required)
2. use MATLAB Support Package Installer to install OS Generic Video Interface.(login required)

如果想调用摄像头实时检测程序，请在安装好上述安装包的前提下，cd 到 detect 文件夹并运行 camera.m。注意需要用 imaqhwinfo 函数查看自己的摄像头名称并修改 camera.m 里对应的名称。

此外，MATLAB 不支持神经网络工具箱 nntools 中和训练有关的函数的编译，因此不能用 exe 文件训练网络，但测试过程可以顺利进行。如果想复现训练过程，需要在确保安装了 nntools 工具箱的前提下，在 main.m 的目录下在 MATLAB 的 command line 里运行 main train xxx xxx ANN。

7.2 文件清单及接口

- data
  - test: 测试集数据，其中包含关键点数据
  - test.txt: 测试集图片列表
  - train: 训练集数据
    - points: 训练集关键点
  - train.txt: 测试集图片列表
- src
  - ANN
    - estimate.m: ANN 预测
    - train.m: ANN 训练
  - GPR
    - SEKernel.m: 高斯核函数
    - estimate.m: GPR 预测
    - train.m: GPR 训练
  - LS

    * estimate.m: LS 预测
    * train.m: LS 训练
- poly
    * estimate.m: 多项式预测
    * polybase.m: 多项式基函数
    * train.m: 多项式训练
- ANN.mat: 训练得到的 ANN 模型
- GPR.mat: 训练得到的 GPR 模型
- LS.mat: 训练得到的 LS 模型
- poly.mat: 训练得到的多项式模型
- FacePosition.txt: 使用 ANN 模型在测试集上测试的结果
- detect
    * IntraFaceDetector.exe: 人脸关键点标记程序及相关文件
    * ANN.mat: 训练得到的 ANN 模型
    * camera.m: 实时摄像头标记
    * speak.m4v: 样例视频
- load_data.m: 读取数据
- load_testset.m: 读取测试集
- load_trainset.m: 读取训练集
- main.m: 主程序