第十届英特尔杯全国大学生软件创新大赛

《智能牙齿健康监测 caffe 应用详述》

项目名称:“Protectooth”

-牙齿的日常贴身小护士

参赛团队：SmartDentist 团队

目录

1. 概述

1.1 背景概述

Pt 智能牙齿监测系统由患者端，医生端和服务器三部分组成。系统允许用户采用专用外部设备从一定的角度拍摄口腔图片，并经由客户端上传至服务端，在服务端进行数据预处理，之后云端综合上传的图片的识别结果和调查问卷分析是否存在口腔问题，并提出针对性的建议，同时会为用户推荐附近的口腔医院。所有注册并审核成功的医生也可看到用户的自检照片和描述，提供更中肯的建议。同时所有信息会被存储在服务器，供医学预防科学部门和仲裁部门使用。

因为识别口腔内的环境、牙齿、牙龈以及进行颜色比对是一件较为困难的事情，所以我们一并开发了配套硬件，集成标准比色卡、300万像素口腔内窥镜摄像头、景深摄像头、闪关灯和开口器的一体化解决方案，来获取相对标准的图像并进行处理。我们将使用卷积神经网络模型作为分类器并进行训练，使之能够检测并识别出一些特定牙齿。并通过开口器上集成的标准比色卡和景深摄像头，辅助确定牙龈与黏膜颜色、牙齿位置以及牙体情况。

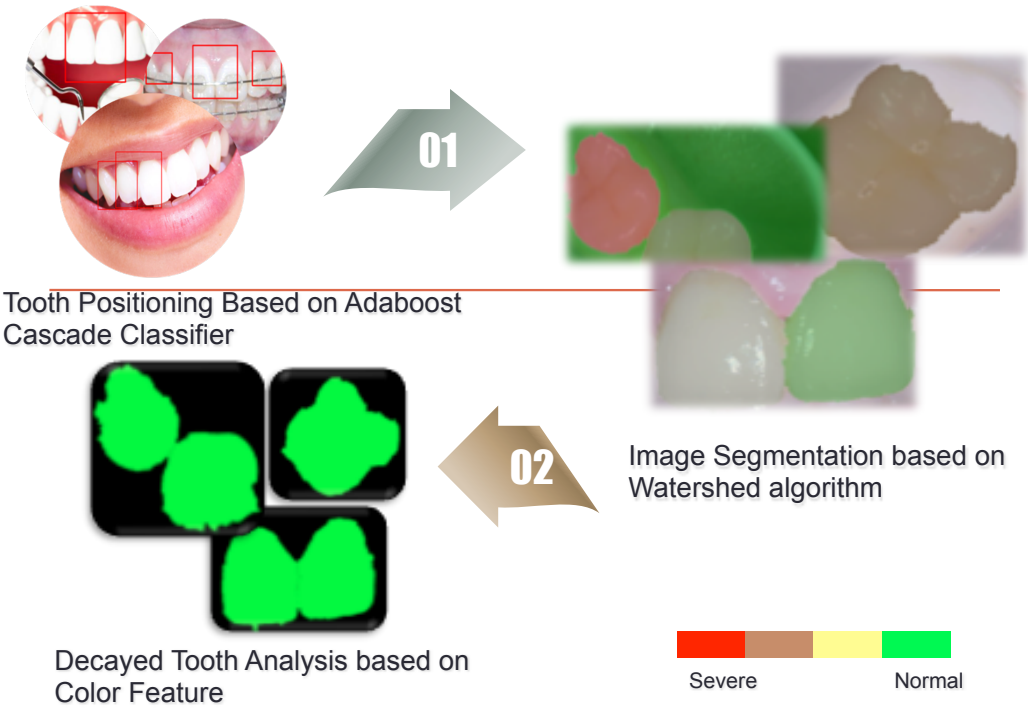

图 1.1.12 标准比色卡

1.2 算法选取

对于我们从摄像头获取的图片，必须经过一定的变换，才能符合以至于更好的为算法提供信息。由于本项目的要使用卷积神经网络模型进行分类器的训练，所以要求所有输入图像在大小方面必须一致，同时为优化运算速度可转换为灰度图像后再处理。之所以选择神经网络方法，是因为在医学图像领域，很多方法都是从计算机视觉领域借鉴过来的，而计算机视觉的许多方法又离不开机器学习和人工智能的基础。在典型的图像分类和识别问题中，通常有两个重要的步骤，一个是特征提取，常见的有 GLCM, HOG, LBP, Haar Wavelet, 一个是分类器， 例如 SVM, Random Forest, Neuron Network 等。特征提取过程中通常是人工选取某些特征，但由于本系统所接受的样本的

复杂性、样本总数限制、样本光照角度等不一致以及单张样本图片的信噪比不平衡等等的原因，决定充分利用 CAFFE 实现卷积神经网络模型对几何变换、形变、光照具有一定程度的不变性的优点，此外训练过的卷积神经网络可以用较小的计算代价扫描整幅待检测图像，因此本系统在服务端采用 caffe 深度学习框架实现卷积神经网络模型并对用户牙齿图片进行分类。

2.数据预处理

2.1 样本数据获取

用户牙齿样本数据主要由两个渠道获取：华西口腔医院数据库中样本记录以及网络图片。从华西口腔医院获取龋齿症状高清照片共 200 张，图片清晰但数量不足，考虑到每张图片信噪比、以该数据集训练出的卷积神经网络的分类的准确性（仅仅 23.4%的准确率）以及测试数据集的需求，于是需要获取更多的数据资源。本系统其他训练样本来源于互联网，使用网络爬虫：

```
url = r'http://image.baidu.com/search/index?tn=baiduimage&ipn=r&ct=201326592&cl=2
dirpath = r'F:\img'

html = requests.get(url).text
urls = re.findall(r'"objURL":"(.*?)"', html)

if not os.path.isdir(dirpath):
    os.mkdir(dirpath)

index = 1
for url in urls:
    print("Downloading:", url)
    try:
        res = requests.get(url)
        if str(res.status_code)[0] == "4":
            print("未下载成功：", url)
            continue
    except Exception as e:
        print("未下载成功：", url)
    filename = os.path.join(dirpath, str(index) + ".jpg")
    with open(filename, 'wb') as f:
        f.write(res.content)
        index += 1

print("下载结束，一共 %s 张图片" % index)
```

图 2.1.1 网络爬虫代码片段

从百度图片上搜索关键词：龋齿、牙齿等，通过爬虫脚本下载到本地，获取正常牙齿图片和龋齿图片共 1014 张左右，通过人工辨识筛去不符合要求的图片共 600 张左右，因此共有样本 600 张，龋齿图片 400 张，正常牙齿图片 200 张，分别保存于 0、1 两个文件夹中。

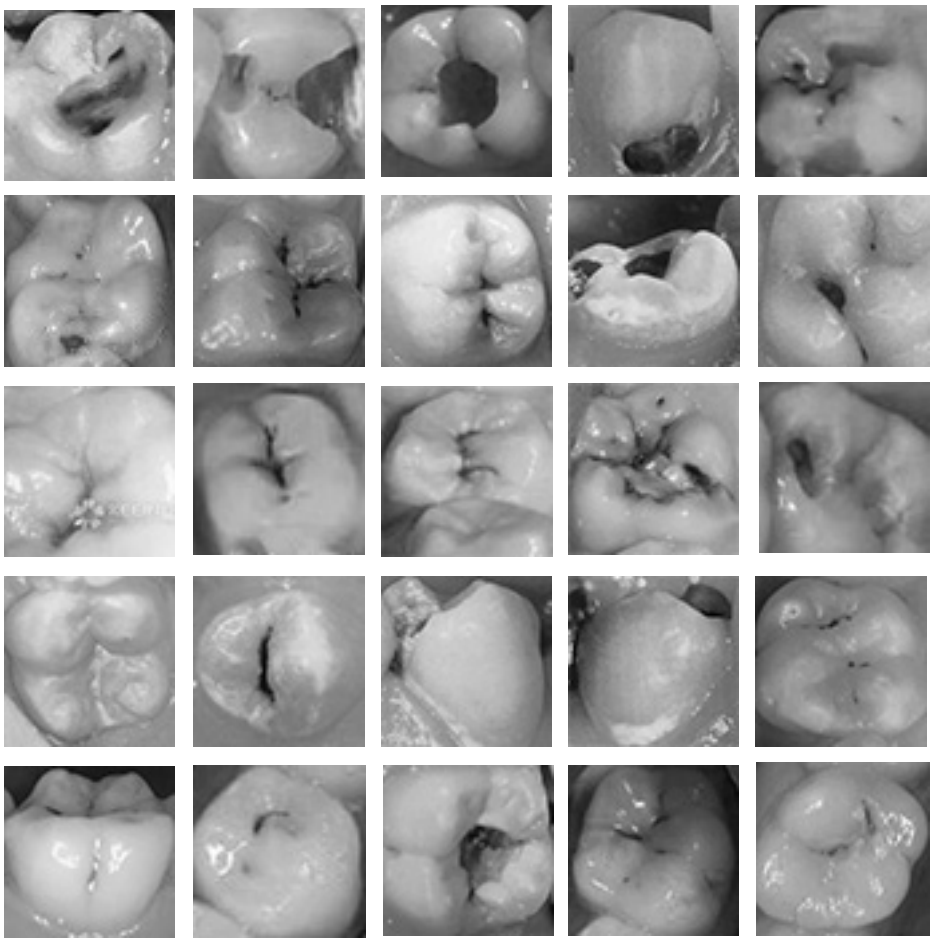

图 2.1.2 获取的口腔图片样本

2.2 图像数据处理

这个数量的样本仍不足在 caffe 框架内实现较好的识别准确率，因此在输入 caffe 框架前，本系统利用 tensorflow 自带的图像预处理功能将样本图片进行平移、旋转、反转等随机操作并保存，以生成更多的样本图片。

```
with tf.Session() as sess:
    img_data = tf.image.decode_jpeg(image_raw_data)
    plt.imshow(img_data.eval())
    plt.show()

    # 上下翻转
    flipped1 = tf.image.flip_up_down(img_data)
    plt.imshow(flipped1.eval())
    plt.show()
    # 左右翻转
    flipped2 = tf.image.flip_left_right(img_data)
    plt.imshow(flipped2.eval())
    plt.show()
    #对角线翻转
    transposed = tf.image.transpose_image(img_data)
    plt.imshow(transposed.eval())
    plt.show()

    # 以一定概率上下翻转图片。
    flipped = tf.image.random_flip_up_down(img_data)
    # 以一定概率左右翻转图片。
    flipped = tf.image.random_flip_left_right(img_data)
```

图 2.2.1 图像预处理代码片段

因此最终训练图片 800 张，测试图片 200 张，共 2 类。我将图片放在 caffe 根目录下的 data 文件夹下面。即训练图片目录：data/re/train/ ,测试图片目录: data/re/test/

2.3 数据格式对接 caffe 框架

原始的样本数据是图片文件，包括 jpg ,jpeg ,png ,tif 等格式，此外图片的大小还不一致。而在 caffe 中经常使用的数据类型是 lmdb 或 leveldb，因此需要将图片格式转换。 在 caffe 中，作者为我们提供了这样一个文件：convert_imageset.cpp，存放在根目录下的 tools 文件夹下。编译之后，生成对应的可执行文件放在 buile/tools/ 下面，这

个文件的作用就是用于将图片文件转换成 caffe 框架中能直接使用的 db 文件。

由于图片已经下载到本地电脑上了，那么需要创建一个图片列表清单，保存为 txt。图片目录是 data/re/train/，创建一个 sh 脚本文件，调用 linux 命令来生成图片清单：

```
# sudo vi examples/images/create_filelist.sh
```

图 2.3.1 生成脚本文件命令

```
# /usr/bin/env sh
DATA=examples/images
echo "Create train.txt..."
rm -rf $DATA/train.txt
find $DATA -name *cat.jpg | cut -d '/' -f3 | sed "s/$/ 1/">>$DATA/train.txt
find $DATA -name *bike.jpg | cut -d '/' -f3 | sed "s/$/ 2/">>$DATA/tmp.txt
cat $DATA/tmp.txt>>$DATA/train.txt
rm -rf $DATA/tmp.txt
echo "Done.."
```

图 2.3.2 图片列表生成代码片段示例

最终生成包含所有图片名称的 test.txt 文件以及 val.txt 文件，作为图像清单。

生成的这个 train.txt 文件，就可以作为第三个参数，直接使用了。

之后调用命令来生成最终的 lmdb 格式数据。首先创建 sh 脚本文件：

```
# sudo vi examples/images/create_lmdb.sh
```

图 2.3.3 创建格式转换脚本文件命令

```
#!/usr/bin/en sh
DATA=examples/images
rm -rf $DATA/img_train_lmdb
build/tools/convert_imageset --shuffle \
--resize_height=256 --resize_width=256 \
/home/xxx/caffe/examples/images/ $DATA/train.txt  $DATA/img_train_lmdb
```

图 2.3.4 图片格式转换代码片段示例

设置参数-shuffle,打乱图片顺序。此外设置参数-resize_height 和-resize_width 将所有图片尺寸都变为 256*256，方便之后 caffe 框架运行图像识别。/home/xxx/caffe/examples/images/ 为图片保存的绝对路径。最后输入 sudo sh examples/images/create_lmdb.sh 运行该脚本，就会在 examples/images/ 目录下生成一个名为 img_train_lmdb 的文件夹，里面的文件就是项目所需要的需要的 db 文件。

3.卷积神经网络模型搭建

3.1 理论概述

卷积神经网络（Constitutional Neural Networks, CNN）是在多层神经网络的基础上发展起来的针对图像分类和识别而特别设计的一种深度学习方法。

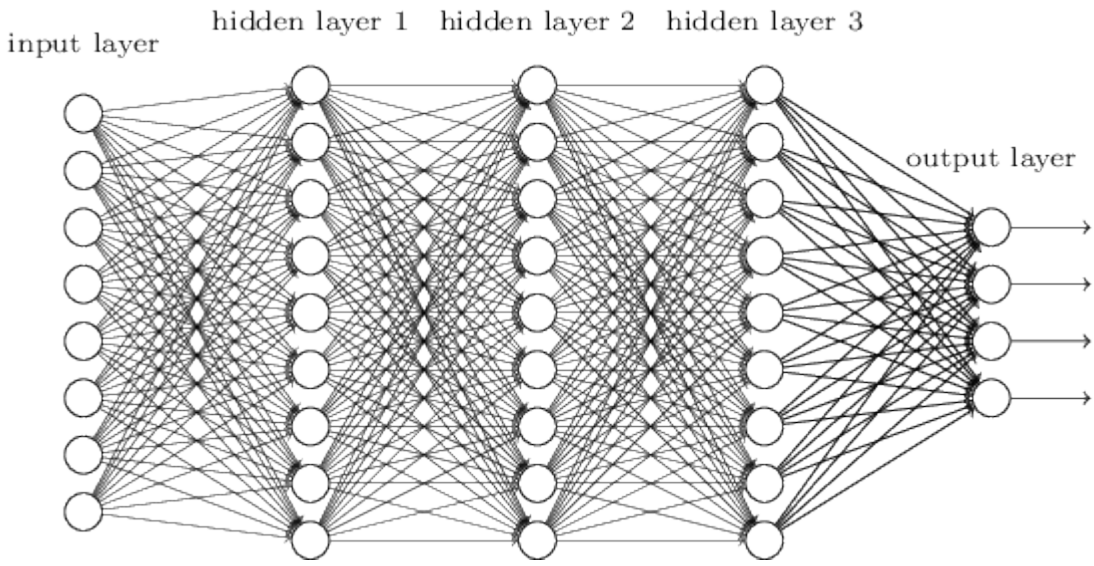

图 3.1.1 多层神经网络简图

多层神经网络包括一个输入层和一个输出层，中间有多个隐藏层。每一层有若干个神经元，相邻的两层之间的后一层的每一个神经元都分别与前一层的每一个神经元连接。在一般的识别问题中，输入层代表特征向量，输入层的每一个神经元代表一个特征值。

在 Pt 智能牙齿监测软件图像识别问题中，输入层的每一个神经元代表一个像素的灰度值（或 RGB 值）。但这种神经网络用于图像识别有几个问题，一是没有考虑图像的空间结构，识别性能会受到限制；二是每相邻两层的神经元都是全相连，参数太多，训练速度受到限制。而卷积神经网络就可以解决这些问题。卷积神经网络使用了针对图像识别的特殊结构，可以快速训练。因为速度快，使得采用多层神经网络变得容易，而多层结构在识别准确率上又很大优势。

与常规神经网络一样，输入层的神经元需要和隐藏层的神经元连接。但是这里不是将每一个输入神经元都与每一个隐藏神经元连接，而是仅仅在一个图像的局部区域创建连接。在本项目中，所有样本图

像被统一处理到了 256 X 256 像素的尺寸，设置第一个隐藏层的神经

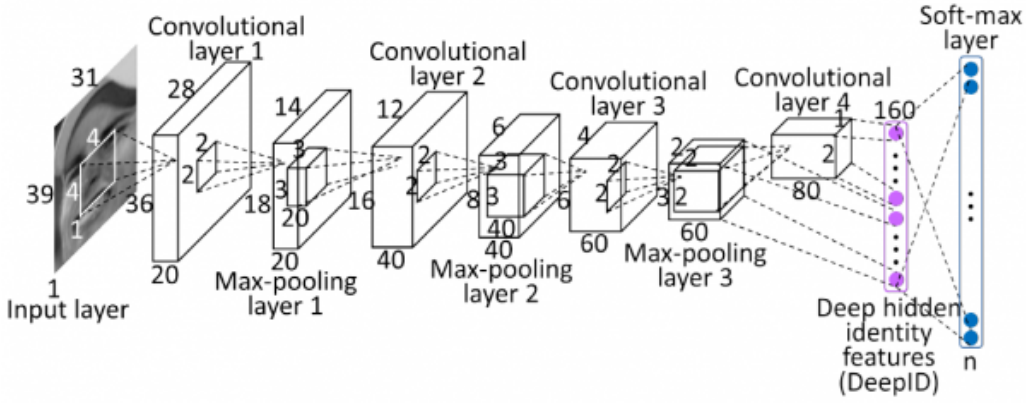

图 3.1.1 卷积神经网络结构

元与输入层的一个 5X5 的区域连接，这个 5X5 的区域就叫做局部感知域，也叫做卷积核。该局部感知域的 25 个神经元与第一个隐藏层的同一个神经元连接，每个连接上有一个权重参数，因此局部感知域共有 5X5 个权重。如果将局部感知域沿着从左往右，从上往下的顺序滑动，就会得对应隐藏层中不同的神经元，因此应用该模型训练效果更好且高效。

3.2 caffe 框架应用

卷积神经网络的模型使用 caffe 内部的 caffenet 模型，位置在 models/bvlc_reference_caffenet/文件夹下。

```
# sudo cp models/bvlc_reference_caffenet/solver.prototxt examples/myfile/
# sudo cp models/bvlc_reference_caffenet/train_val.prototxt examples/myfile/
```

图 3.2.1 拷贝配置文件命令

之后修改其中的 solver.prototxt 配置文件，设置每次处理一批的图像数量为 50，设置迭代次数，以及学习速率（0.1 ~ 0.5 为佳，兼顾速度和精度),设置步长等等其他参数，并设置使用 GPU 进行模型计算。并设置学习速率随着迭代的进行不断地减小，以达到最大化精度的目

的。

```
net: "examples/myfile/train_val.prototxt"
test_iter: 2
test_interval: 50
base_lr: 0.001
lr_policy: "step"
gamma: 0.1
stepsize: 100
display: 20
max_iter: 500
momentum: 0.9
weight_decay: 0.005
solver_mode: GPU
```

图 3.2.2 配置文件

然后设置卷积神经网络使用 CAFFE 框架的各项参数。我们采用 3 层卷积层，按照卷积层-池化层-激活函数这样的顺序链接各个层，最后链接全连接层计算输出结果，配置输入数据的格式为 LMDB，测试阶段每批 50 个图像数据，训练阶段每批 256 个图像数据等等参数。

```
name: "CaffeNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    mirror: true
    crop_size: 227
    mean_file: "examples/myfile/mean.binaryproto"
  }
  data_param {
    source: "examples/myfile/img_train_lmdb"
    batch_size: 256
    backend: LMDB
  }
}
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TEST
  }
  transform_param {
    mirror: false
    crop_size: 227
    mean_file: "examples/myfile/mean.binaryproto"
  }
  data_param {
    source: "examples/myfile/img_test_lmdb"
    batch_size: 50
    backend: LMDB
  }
}
```

图 3.2.3 CNN 模型配置参数代码片段

3.3 caffe 分类器训练

配置过 caffe 环境和以上所有卷积神经网络的参数后，执行命令 ：

sudo build/tools/caffe train -solver examples/myfile/solver.prototxt

即可运行卷积神经网络训练程序。在本地 PC 机器上配置过 gpu+cudnn

```
I1228 18:08:46.524875  4075 solver.cpp:321] Iteration 500, loss = 0.112465
I1228 18:08:46.524926  4075 solver.cpp:341] Iteration 500, Testing net (#0)
I1228 18:08:46.656059  4075 solver.cpp:409]     Test net output #0: accuracy
I1228 18:08:46.656110  4075 solver.cpp:409]     Test net output #1: loss = 0.
I1228 18:08:46.656121  4075 solver.cpp:326] Optimization Done.
I1228 18:08:46.656128  4075 caffe.cpp:215] Optimization Done.
```

图 3.3.1CNN 训练结果之一截图

环境后采用 GPU 运算效率很高，对总量达 1000 的图像数据进行 10 次迭代训练测试仅使用 10 分钟左右时间。经过 500 次左右迭代后在测试集上该神经网络模型准确率达到了最高 73.5%。

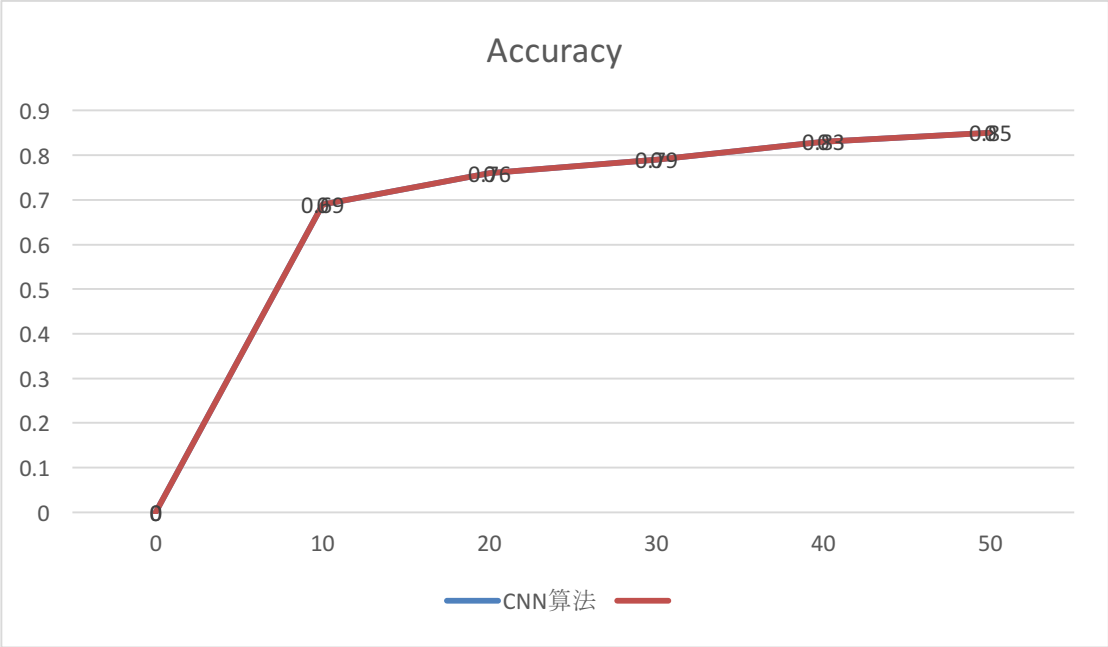

图 3.3.1 CNN 模型训练精度(epoch = 50)

3.4 在线图像识别

为实现用户将口腔图片上传后，在服务端在线自动解析、快速返回用户口腔分析结果，本系统先在本地对模型进行大量训练和测试，当模型在数据集上达到 82.3%左右准确率时已经难以再提高，之后将训练好的模型上传至服务器，在服务器配置了 caffe+cpu 深度学习框

架环境，并结合 python 接口，使用 python 的 django 框架搭建服务端。

图 3.4.1 服务端返回的 caffe 计算结果

对接用户对服务端的请求和服务端 caffe 框架运行反馈结果。此外，本系统拟对服务端接收的用户数据进行分析，扩充模型训练集，在服务器投入运行的过程中，定时更新模型，以达到更高的准确率。