移动互联网实践

课程设计

题　目：

基于视频流和计算机视觉的学生课堂学习状态实时分析

张景赫（2006040126）

白鹏冉（2009040114）

2023 年 2 月 25 日

目录

一、 功能简介

本软件通过手机拍摄课堂学习画面并上传至数据处理服务器，并获取数据处理服务器实时分析学生的整体学习状态结果，并通过可视化、标注画面等手段对服务器返回数据进行多种实时展示。

二、 需求分析

当前的教育模式下，授课学时紧张、听课学生众多，教师对学生实时听课状况无法整体把握，进而导致授课教师无法较好地把握讲课速度和授课内容深度，导致一部分学生无法和老师的授课进度同步，进而导致其学习效率地下、学习效果较差。

1. 功能需求

1) 拍摄课堂实时画面

本软件通过获取手机前置摄像头画面对学生学习状态进行实时拍摄，并将实时视频流推流至数据处理服务器;

2) 实时画面处理

数据处理服务器通过获取软件拍摄画面，运行计算机视觉识别程序，将识别结果和识别后的框取画面发送至移动软件;

3) 显示实时数据

从数据处理服务器接收分析数据，每隔数秒列出当前听课人数和

未听课人数的数据;

4) 可视化处理

将收到的数据以折线图形式表现在页面相应区域内，通过可视化展示向授课教师传达班级整体听课情况信息;

5) 标注画面展示

从数据处理服务器获取框取后的实时画面，并展示在页面的相应区域内，以直观向授课教师表现学生听课情况。

三、 详细设计

1. 功能设计

1) 拍摄权限申请

通过开源移动平台的全能多媒体开发框架 Vitamio 相关 API 实现摄像机的调用和画面实时获取;

通过配置 AndroidManifest.xml 和动态权限申请语句申请访问用户摄像机。

关键代码如下:

```
private void requestPermissions() {
RxPermissions rxPermission = new RxPermissions(MainActivity.this);
       rxPermission
               .request(permissionNames)
               .subscribe(new io.reactivex.functions.Consumer<Boole
an>() {
                   @Override
                   public void accept(Boolean aBoolean) throws Exce
ption {
```

```
                        if (aBoolean) {
                            initView();
                        } else {
                            Toast.makeText(MainActivity.this, "关闭的
权限可以在手机设置中打开。",
                                    Toast.LENGTH_SHORT).show();
                            initView();
                        }
                    }
                });
    }
```

2) 实时画面显示和推流

通过调用开源移动平台的全能多媒体开发框架 Vitamio 视频播放器，设置相关参数，将摄像头录制信息实时展示到主界面中，并将显示画面推流通过开源直播 SDK WLive 相关 API 实现摄像画面实时推流到目的服务器。

关键代码如下：

```
rtmpUrl ="rtmp://192.168.137.1:1935/live/home"
mLiveCameraView = (StreamLiveCameraView) findViewById(R.id.stream_
previewView);
rtmpUrl=
        //参数配置 start
       streamAVOption = new StreamAVOption();
       streamAVOption.streamUrl = rtmpUrl;
       //参数配置 end

        mLiveCameraView.init(this, streamAVOption);
       mLiveCameraView.addStreamStateListener(resConnectionListene
r);
       LinkedList<BaseHardVideoFilter> files = new LinkedList<>();
       files.add(new GPUImageCompatibleFilter(new GPUImageBeautyFi
lter()));
;
       mLiveCameraView.setHardVideoFilter(new HardVideoGroupFilter
(files));
```

3) 推流服务器搭建

通过开源服务器 Nginx 进行 rtmp 服务器部署，下载 rtmp 相关模块包后，开放 1935 端口以接收移动端推流画面，开放 1936 端口以推流分析后的框取画面。

由于在 windows 下未对 rtmp 模块进行编译，需要采用 nginx Gryphon 配合 nginx-rtmp-module 的方式搭建 rtmp 服务器

关键配置项如下：

```
rtmp {
    server {
        listen 1935;
        chunk_size 4000;
        application live {
             live on;
        }
    }
    server {
        listen 1936;
        chunk_size 4000;
        application live {
             live on;
        }
    }
}
```

运行如下语句后，启动 nginx 服务器：

```
nginx -c ./conf/nginx-win.conf
```

4) 实时画面处理

通过开源计算机视觉识别框架 yolov5 对 Nginx 服务器 1935 端口的视频流进行实时分析和推理，将分析后的实时听课情况数据写入本地 MySQL 数据库，并将分析后的框取视频流推流至 Nginx 服务器的

1936 端口。

写入数据库关键代码如下：

```
db = pymysql.connect(**config)
cursor = db.cursor()
sql_mk_empty = "delete from temp;"
print(sql_mk_empty)
cursor.execute(sql_mk_empty)
db.commit()
......
sql = "INSERT INTO temp(data_nl,data_l) VALUES("
if len(det):
    # Rescale boxes from img_size to im0 size
    det[:, :4] = scale_coords(img.shape[2:], det[:, :4],
im0.shape).round()
    sql_cir = sql
    # Print results
    for c in det[:, -1].unique():
        n = (det[:, -1] == c).sum()  # detections per class
        s += f"{n} {names[int(c)]}{'s' * (n > 1)}, "  # add to
string
        if len(det[:, -1].unique()) == 1:
            if names[int(c)] == "not-listening":
                sql_cir += str(n.item())
                sql_cir += ",0);"
            if names[int(c)] == "listening":
                sql_cir += "0,"
                sql_cir += str(n.item())
                sql_cir += ");"
        if len(det[:, -1].unique()) == 2:
            if names[int(c)] == "not-listening":
                sql_cir += str(n.item())
                sql_cir += ","
            if names[int(c)] == "listening":
                sql_cir += str(n.item())
                sql_cir += ");"
    print(circle)
    if circle % skip_cir == 0:
        print(sql_cir)
        cursor.execute(sql_cir)
        db.commit()
    circle = circle + 1
    ......
else:
```

```
    if circle % skip_cir == 0:
        sql_null = sql
        sql_null += "0,0);"
        print(sql_null)
        cursor.execute(sql_null)
        db.commit()
    circle = circle + 1
```

数据处理服务器对框取画面进行推流的关键代码如下：

```
pipe = subprocess.Popen(command, shell=True,
stdin=subprocess.PIPE)
......
pipe.stdin.write(im0.tobytes())
```

5) 移动应用获取分析画面

移动应用通过 SDK WSLive 开发框架的相关 API 获取 Nginx 服务器的 1936 端口的实时框取画面视频流，将画面通过多线程展示到页面相应区域。

关键代码如下：

```
rtmpUrl ="rtmp://192.168.137.1:1935/live/home"
mVideoView.setVideoPath(rtmpUrl_rec);
       mVideoView.setMediaController(new MediaController(this));
       mVideoView.requestFocus();
       mVideoView.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
           @Override
           public void onPrepared(MediaPlayer mp) {
               mp.setPlaybackSpeed(1.0f);
           }
       });
surface_text.setVisibility(v.GONE);
mVideoView.setVisibility(v.VISIBLE);
```

6) 分析数据传输

移动应用通过对数据处理服务器的 MySQL 服务(即 3306 端口)进行相应查询操作，通过多线程展示到页面相应区域。

Java 中访问数据库部分关键代码如下，核心为 jdbc 的相关 API 的实现。

```
public static int[] search() throws Exception {
    //1.加载驱动程序
    Class.forName("com.mysql.jdbc.Driver");
    //2. 获得数据库连接
    Connection conn = DriverManager.getConnection(URL, USER,
PASSWORD);
    //3.操作数据库，实现增删改查
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery("select * from temp where id
= (select MAX(id) from temp);");
    //如果有数据，rs.next()返回true
    int arr[] = new int[5];
    int i = 0;
    while(rs.next()){
        arr[i++] = rs.getInt("data_1");
        arr[i++] = rs.getInt("data_n1");
    }
    for (int j = 0; j < i; j++){
        System.out.println(arr[j]);
    }
    return arr;
}
```

数据库查询内容通过多线程更新到页面对应区域关键代码如下，通过多线程的 arg1 和 arg2 参数传递到 handler 中更新显示的内容。

```
final Handler handler2 = new Handler(){
    public void handleMessage(Message msg2) {
        super.handleMessage(msg2);
        data_n1.setText("未听课人数："+String.valueOf(msg2.arg1));
        data_1.setText("听课中人数："+String.valueOf(msg2.arg2));
    }
};
```

```
......
new Thread(){
    public void run(){
        while(true){
            Message msg2 = Message.obtain();
            DbUtil db = new DbUtil();
            int[] res_arr;
            try {
                res_arr = db.search();
            } catch (Exception e) {
                throw new RuntimeException(e);
            }

            msg2.arg1 = res_arr[1];
            msg2.arg2 = res_arr[0];
            handler2.sendMessage(msg2);
            try {
                sleep(3000);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}.start();
```

7) 分析数据可视化

通过移动端开源图表框架 MPAndroidChart 的相关 API 接口，将从数据处理服务器的 MySQL 服务（即 3306 端口）查询的结果通过多线程展示到页面相应区域。

通过 MPAndroidChart 初始化图表的函数关键代码如下，主要内容为 MPAndroidChart 的 API 实现。

```
public void make_chart(LineChart chart, List<Entry> valsComp1,
List<Entry> valsComp2){
    List<Entry> entries = new ArrayList<Entry>();
    chart.setBackgroundColor(Color.BLACK);

    Legend legend = chart.getLegend();
    legend.setTextColor(Color.WHITE);
```

```
    YAxis leftAxis = chart.getAxisLeft();
    leftAxis.setTextColor(Color.WHITE);

    YAxis rightAxis = chart.getAxisRight();
    rightAxis.setTextColor(Color.WHITE);


    XAxis upAxis = chart.getXAxis();
    upAxis.setTextColor(Color.WHITE);

    chart_update(chart, 0, 0, 0);

    LineDataSet setComp1 = new LineDataSet(valsComp1, "听课人数");
    setComp1.setAxisDependency(YAxis.AxisDependency.LEFT);
    setComp1.setColor(Color.YELLOW);
    LineDataSet setComp2 = new LineDataSet(valsComp2, "未听课人数
");
    setComp2.setAxisDependency(YAxis.AxisDependency.LEFT);
    setComp2.setColor(Color.LTGRAY);


    List<ILineDataSet> dataSets = new ArrayList<ILineDataSet>();
    dataSets.add(setComp1);
    dataSets.add(setComp2);

    LineData data = new LineData(dataSets);
    chart.setData(data);

    chart.invalidate();
}
```

通过 MPAndroidChart 更新图表的函数关键代码如下，主要内容为 MPAndroidChart 的 API 实现。

```
public void chart_update(LineChart chart, int order, int
data_l_input, int data_nl_input){
    Entry data_l = new Entry(order, data_l_input); // 0 == quarter
1
    Entry data_nl = new Entry(order, data_nl_input); // 0 ==
quarter 1
    System.out.println(data_l);
    if(order >= 1){
```

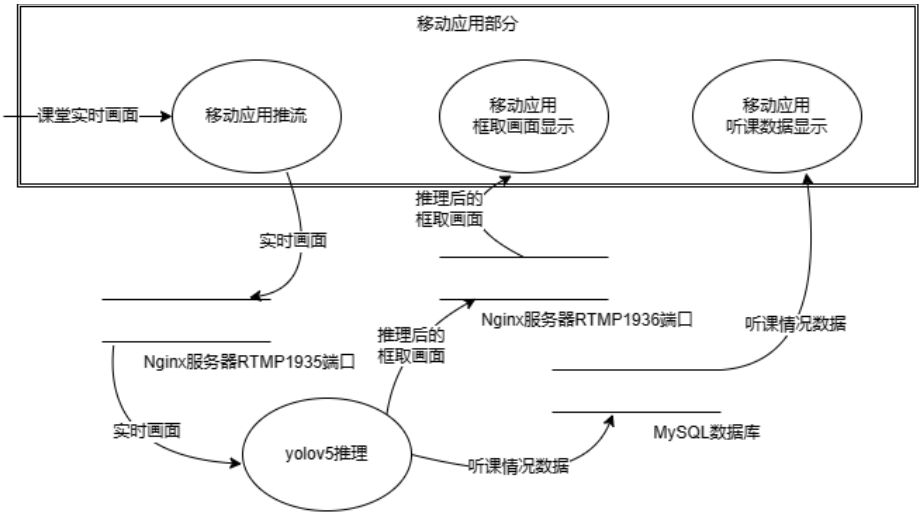
```
        LineData data = chart.getData();
        data.addEntry(data_nl, 1);
//            System.out.println(set);
        chart.notifyDataSetChanged();
        chart.invalidate();

        LineData data2 = chart.getData();
        data2.addEntry(data_l, 0);
        chart.notifyDataSetChanged();
        chart.invalidate();
    }
}
```

调用如上两函数的关键代码如下，在 onCreate 主线程执行图表创建函数，通过多线程更新图表内容。

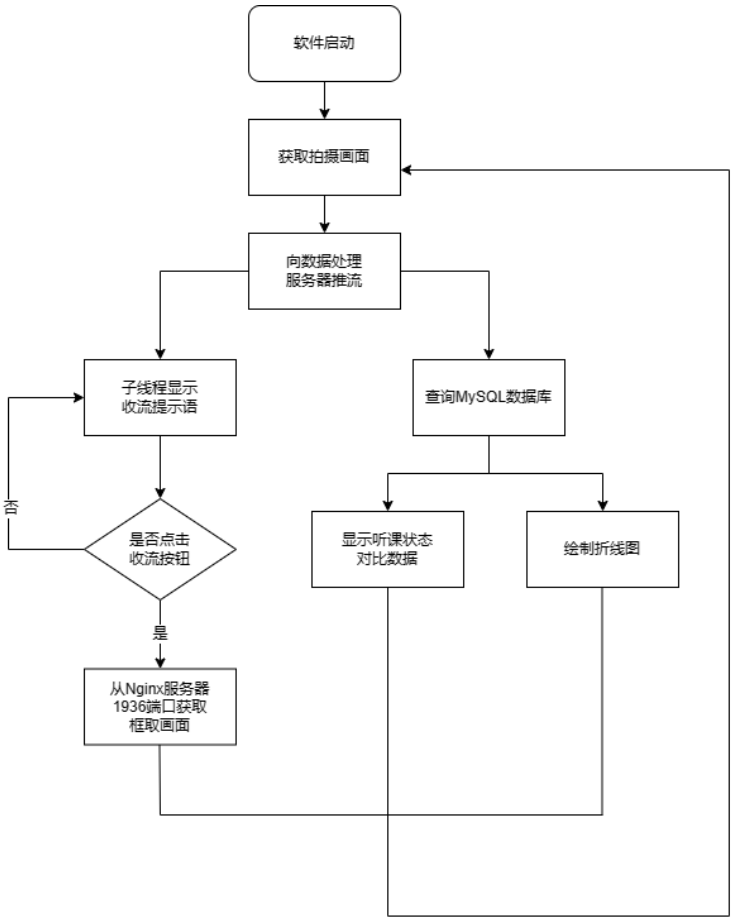
```
final List<Entry> valsComp1 = new ArrayList<Entry>();
final List<Entry> valsComp2 = new ArrayList<Entry>();
make_chart(chart, valsComp1, valsComp2);
......
new Thread(){
    int order = 1;
    public void run(){
        DbUtil db = new DbUtil();
        int[] res_arr;
        while(true){
            try {
                res_arr = db.search();
            } catch (Exception e) {
                throw new RuntimeException(e);
            }
            chart_update(chart, order, res_arr[0], res_arr[1]);
            order = order+1;
            try {
                sleep(3000);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}.start();
```

2. 数据流图和功能流程图

1) 数据流图



2) 功能流程图

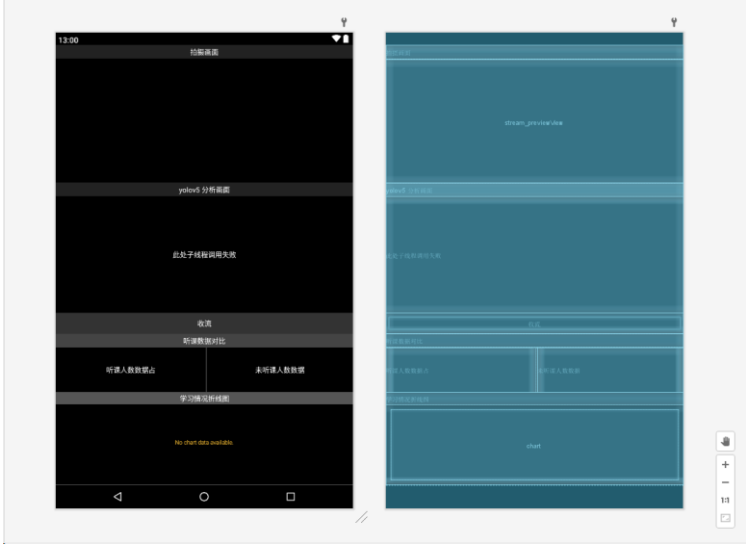


3. 页面设计

1) 整体布局

通过 AndroidStudio 的 LinearLayout 规范，根据权重对页面进行整体划分，可以保证界面对各种设备的适配，不会出现界面留白、错乱的等情况。

通过字体居中设计和黑色背景，使得标题和数据信息更显眼，方便用户理解。



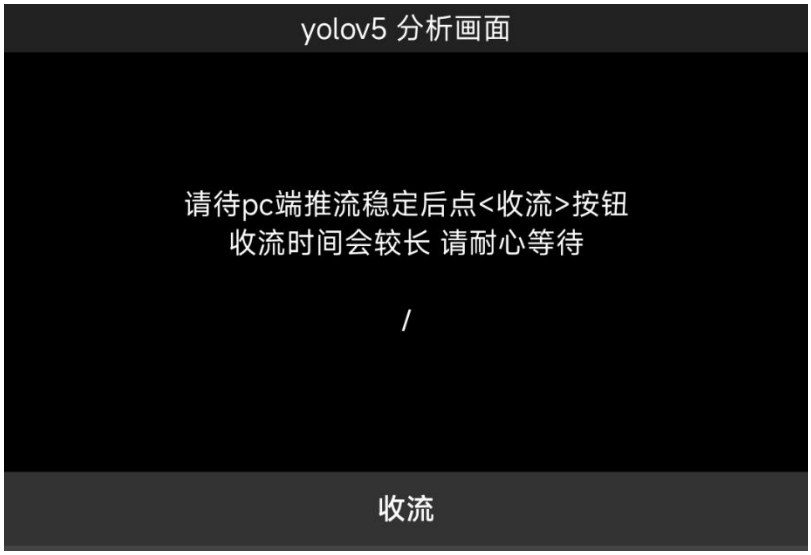
2) 运行效果

本软件的实际运行效果如下所示：

3) yolov5 分析画面部分设计介绍

由于本处采用线程方式，默认文字为“此处线程调用失败”，由于

本软件的收流需要在数据处理服务器进行稳定的视频流推流之后进行，故当线程运行后会改变此处文字为以下提示信息：



当用户点击收留按钮后，会通过安卓的 Toast 规范进行提示：

收流成功后，画面会在黑色区域显示，如下所示：

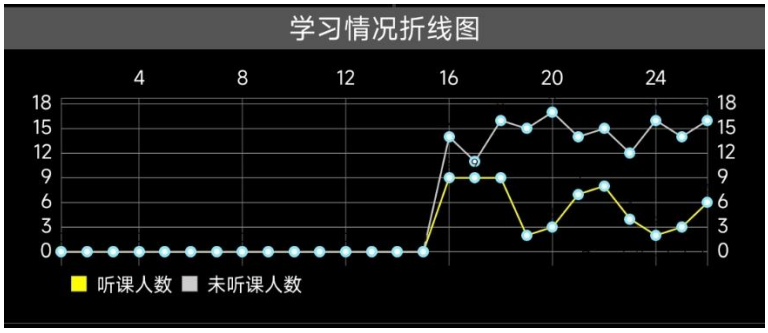
4) 听课数据对比设计介绍

通过两部分区域进行数据对比展示，展示形式简单易懂，用户操

作难度低。

5) 学习情况折线图设计介绍

通过折线图直观展示课堂的听课趋势变化，使得授课教师更快获得信息，其中听课人数用黄色标注，更加显眼。



4. 数据库设计

由于本软件涉及实体较少，数据库相对较简单，仅包含一张表，共 3 列，各列及其属性如下：

列名	列属性	操作	含义
ID	INT PRIMARY_KEY	SELF_INCREMENT	ID 序号
DATA_L	INT		当前时间听课人数
DATA_NL	INT		当前时间未听课人数

软件操作数据库相关内容见详细设计部分的功能设计部分。

四、 总结

通过本次移动互联网实践课程的学习，我们学到了很多有用的知识，在老师的指导下，完整地实现了从开始的简单的页面布局、简单控件等功能，直至整个 app 系统的完善。

在本次课程设计中，在老师的指导下，我们完成了学生学习状态实时分析软件。起初，我们对推流都不熟悉。幸好，在寒假假期中有足够的时间来学习和完善相关的知识体系，我们系统学习了使用 Vitamio 定义视频框架，采用 WSLive 进行推流，使用 Nginx 来搭建推流服务器，使用 yolo5 进行视频流的画面处理，使用 Mysql 来存储学习状态的相关信息，使用 MPAndroidChart 来完成学习状态相关图表的绘制。 过程中，我们分为两个大部分来完成，分为 app 端的设计，以及后端用 python 来实现 yolo 的应用。中间通过搭建 nginx 服务器来完成两部分的连接。最终，我们成功完成了 app 的设计。

总体来说相对比较完善，但是，还有一些不太完善的地方，比如在收流后显示的视频流并不稳定，yolo 标注时占用的位置太大，导致观测结果不清晰等问题，需要后期进一步地完善。

本次课程设计中，在老师的帮助下，我们完成了本以为很难完成的工作，学会了很多理论知识，并将其付诸实践，并学会调用第三方开发库，大大提高了完成项目的能力。