基于 WIFI 探针的商业大数据分析技术

Commercial data analysis technology based on WIFI probe

软件设计说明书

参赛学校：河海大学常州校区

组　　名：Super Super Handsome

指导老师：陈慧萍

队　　长：魏臻江

队　　员：丁翰雯、陶宇

目录

1.引言 4
1.1 编写目的 4
1.2 背景 4
1.3 定义 4
1.4 参考资料 5
2.总体设计 5
2.1 需求规定 5
2.2 运行环境 6
2.2.1 硬件环境 6
2.2.2 软件环境 6
2.3 设计原则与思想 6
2.3.1 设计原则 6
2.3.2 设计思想 7
2.4 系统设计方案 7
2.4.1 系统技术架构 7
2.4.2 处理流程 9
2.4.2.1 系统总流程 9
2.4.2.1 分模块设计流程 11
2.5 系统部署方案 17
2.5.1 总体部署说明 17
2.5.2 环境搭建流程 18
2.5.2.1 Hadoop2.2 环境搭建 18
2.5.2.2 Spark2.1.0 环境搭建 19
2.5.2.3 Docker 镜像配置 21
2.5.2.4 Tomcat 集群配置 21
2.6 系统界面设计 24
3.接口设计 31
3.1 用户接口 31

3.2 外部接口 32
3.3 内部接口 32
4.运行设计 33
4.1 运行模块组合 33
4.2 运行控制 33
4.3 运行时间 34
5.数据库设计 34
5.1 数据库表列表 34
5.2 逻辑结构设计 35
5.3 物理结构设计 35
6.系统出错处理设计 39
6.1 出错信息 39
6.2 补救措施 39
7.系统维护设计 40

软件设计说明书

1.引言

1.1 编写目的

本文将明确说明系统各功能的实现方式，确定软件的全部需求和软件组成模块，并且确定各模块的功能和用户接口，以此作为详细设计的依据和基础。

预期读者为：软件开发的人员，项目评审人员，及软件测试人员。

1.2 背景

随着科学技术的高速发展，我们已步入数字化、网络化的时代。网上购物愈益流行，然而，受到产品质量检验和实际体验感的限制，线下商店当然是不可替代的，很多顾客仍然希望进入实体店亲身试用挑选。

首先开发探针设备能够采集客户唯一的定位标识，比如 MAC 地址，通过数据分析技术，采用离线计算和实时计算结合的方式，为商业环境提供科学的、全面的数据决策依据。不仅对营销能力的评估，也可以对管理上进行优化。利用探针数据的客流分析打破模式束缚，不仅仅只是提供可信的客流数据分析，同时还

还可以通过本系统随时查看店铺内的客流量情况，并根据客流高峰时段，对店内工作人员进行合理分配，提高人力资源利用率，并在一定程度上降低经营成本。

1.3 定义

WIFI 探针：WIFI 探针技术是指基于 WIFI 探测技术来识别 AP(无线访问接入点)附近已开启 WIFI 的智能手机或者 WIFI 终端（笔记本，平板电脑等），无需用户接入 WIFI，WIFI 探针就能够识别用户的信息。

大数据：指无法在一定时间范围内用常规软件工具进行捕捉、管理和处理的数据集合，是需要新处理模式才能具有更强的决策力、洞察发现力和流程优化能力的海量、高增长率和多样化的信息资产。

1.4 参考资料

[1]维克托·迈尔·舍恩伯格，肯尼思·库克耶，周涛. 《大数据时代》[J]. 教育科学论坛, 2013, 8(7):27-31.

[2]余阳，汤庸. 《软件工程》实例化教学探索[J]. 逻辑学研究, 2003, 23(5):8-13.

[3]殷人昆,郑人杰,马素霞,白晓颖. 实用软件工程(第三版)[J]. 计算机教育, 2010(24):95.

[4]《猫酷室内行为采集系统》 http://www.mallcoo.cn/action.html

2.总体设计

2.1 需求规定

(1) WIFI 探针设备：通过探针设备采集可监测范围内的手机 MAC 地址、地理位置、与探针距离、时间信息（特别针对 ANDROID6.0 和 IOS10 版本后的移动终端设备进行测试需能采集到 MAC 地址）。

(2) 数据传送：服务端接收探针定时发送的数据，将数据保存到数据分析平台待用。接收数据不能有数据丢失或者数据失真，探针每 3 秒发送一次数据，数据采集并发量即单台服务器的数据吞吐量，不得低于 1000 台设备；

(3) 数据存储：实时数据量获取后及需及时结合历史数据分析，需将已采集生成的数据快速并定期处理后，存储并生成相应的文件，方便以查阅及处理。

(4) 数据处理：利用大数据技术 3s 之内完成各门店客流量统计，顾客来访次数和来访时长，与顾客活跃度。

(5) 数据可视化：探针采集的原始数据要经过比较处理后以图表或者其他直观的方式实时展现出。

(6) 活跃度趋势预测：根据当前采集到的数据，分析并预测顾客的来访活跃度。

(7) 营销方案热度判断：系统会定期向管理层提交营销方案的热度判断报告。

(8) 数据更新：系统定期更新历史数据及其模型。

2.2 运行环境

2.2.1 硬件环境

(1) CPU：Intel CoreI5 1.8GHz 及以上

(2) 内存：2G×3 及以上

(3) 硬盘：60G 及以上

(4) 探针：双核 探测距离半径>100 米频率 2.4GHz-2.5GHz

2.2.2 软件环境

(1) 服务器： Tomcat / IIS（tomcat 和 IIS 需启动 CGI 支持）

(2) 操作系统：Ubuntu 16.04 LTS

(3) 数据库：HBASE 1.1.2（分布式数据库）

(4) 基本配置：Spark 2.1.0（Built for Hadoop 2.2.0）分布式环境、JDK 1.7 及以上、Scala 2.12.1 及以上

(5) 开发工具：IntelliJ IDEA 2017.1.1 及以上、Eclipse 3.6 及以上

(6) PC 端：IE6.0 及以上版本；IE 内核的其它浏览器；Chrome21.0 等

(7) 手机端：自带浏览器即可

2.3 设计原则与思想

2.3.1 设计原则

基于 WIFI 探针的商业大数据分析系统遵循以下原则：

(1) 快速性，探针每三秒采集一次数据，在三秒内系统分析处理完成探针采回的实时数据；

(2) 准确性，每一次的数据分析能正确的得到各个店铺的数据；

(3) 并发性，支持 1300 台的 WIFI 探针并发；

(4) 对比性，客户端显示历史记录，可以分小时、天、周、月查看；

(5) 易控性，系统实时监测探针运行状态，并能远程控制探针的开启和关闭；

(6) 简洁性，用户操作界面、简单易用、信息展现明了；

(7) 易用性，客户端跨平台、自适应；

2.3.2 设计思想

分析处理大数据处理是分布式的特点，使用 Spark 框架其事务运行在内存中能保证在三秒内处理完探针采回的实时数据，同时我们使用分布式数据库 HBASE 比传统的关系型数据库有更快的响应速度，能保证分析的实时数据在较短时间内全部存入数据库，保证了数据的准确性，使得出错率降低到了 1%以下。

为了处理系统的并发性，在分布式事务中开启 1000 个线程进行 1000 台 WIFI 探针采回实时数据的并发接收。

本系统除了实时分析以外，还支持历史分析，通过存储每一次的分析结果能够快速的响应用户需要的历史数据对比和环比。

为了使不同分辨率和不同浏览器的客户端拥有一致的显示效果和用户体验。本系统采用了响应式布局和 bootstrap 的框架，从而可以为不同终端的用户提供更加舒适的界面和更好的用户体验。如此，也就实现了高度的跨平台性。

2.4 系统设计方案

2.4.1 系统技术架构

基于 WIFI 探针的商业大数据分析系统主要采用 Spark+HBase 框架。

选用 HDFS，是因为 HDFS 作为 Hadoop 生态技术圈底层的关键技术之一，被设计成适合运行在通用硬件上的分布式文件系统。但 HDFS 是一个高度容错性的系统，适合部署在连接的机器上。HDFS 能够提供高吞吐量的数据访问，非常适合大规模数据集上的应用，从而适合处理高并发。

选用 Spark 主要实现和编程接口基于 Scala，Spark 的核心思路就是将数据集缓存在内存中加快读取速度，同时用 lineage 关联的 RDD 以较小的性能代价保证数据的鲁棒性。RDD 是弹性分布式数据也是 spark 的核心，完全弹性的，如果数据丢失一部分还可以重建。RDD 有自动容错、位置感知调度和可伸缩性，通过数据检查点和记录数据更新金象容错性检查。通过 SparkContext.textFile()

加载文件变成 RDD，然后通过 transformation 构建新的 RDD，通过 action 将 RDD 存储到外部系统。适用领域正如其目标作用域，Spark 适用于需要多次操作特定数据集的应用场合。需要反复操作的次数越多，所需读取的数据量越大，受益越大；这也正是我们选用 Spark 的主要原因。

采用分布式系统的可靠协调系统 ZooKeeper，因为其目标就是封装好复杂易出错的关键服务，将简单易用的接口和性能高效、功能稳定的系统提供给用户。

选用 HBASE，HBASE 是运行在 Hadoop 上的 NoSQL 数据库，它是一个分布式的并可扩展的大数据仓库，也就是说 HBASE 能够利用 HDFS 的分布式处理模式，这意味着在一组商业硬件上存储许多具有数十亿行和上百万列的大表。除去 Hadoop 的优势，HBASE 本身就是十分强大的数据库，它能够融合 key/value(键值)存储模式带来实时查询的能力，以及通过 Spark 进行离线处理或者批处理的能力。总的来说，HBASE 能够让你在大量的数据中查询记录。这对于商业大数据的分析具有较大的益处，也能为历史比较，环比提供有效的支持。

而采用机器学习中的 Spark 机器学习中的二项 logistic 回归模型对历史数据结合实时数据进行预测,不断训练模型，可以为商家提供有效的参考信息。

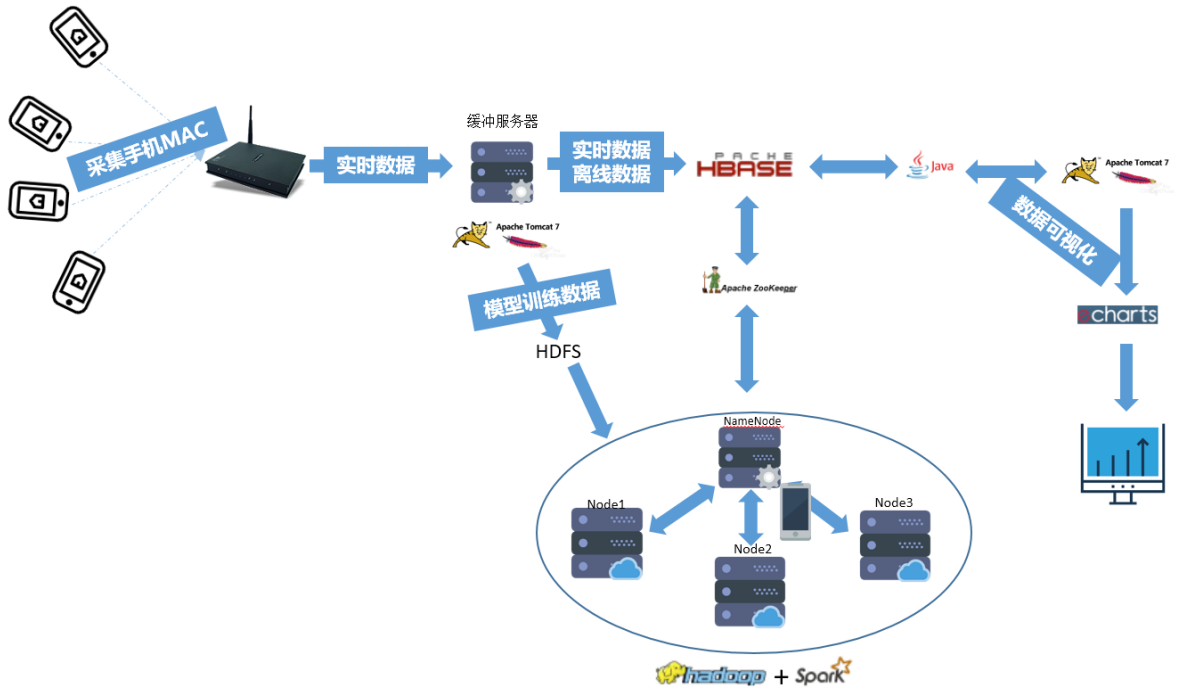

图 1 系统结构图

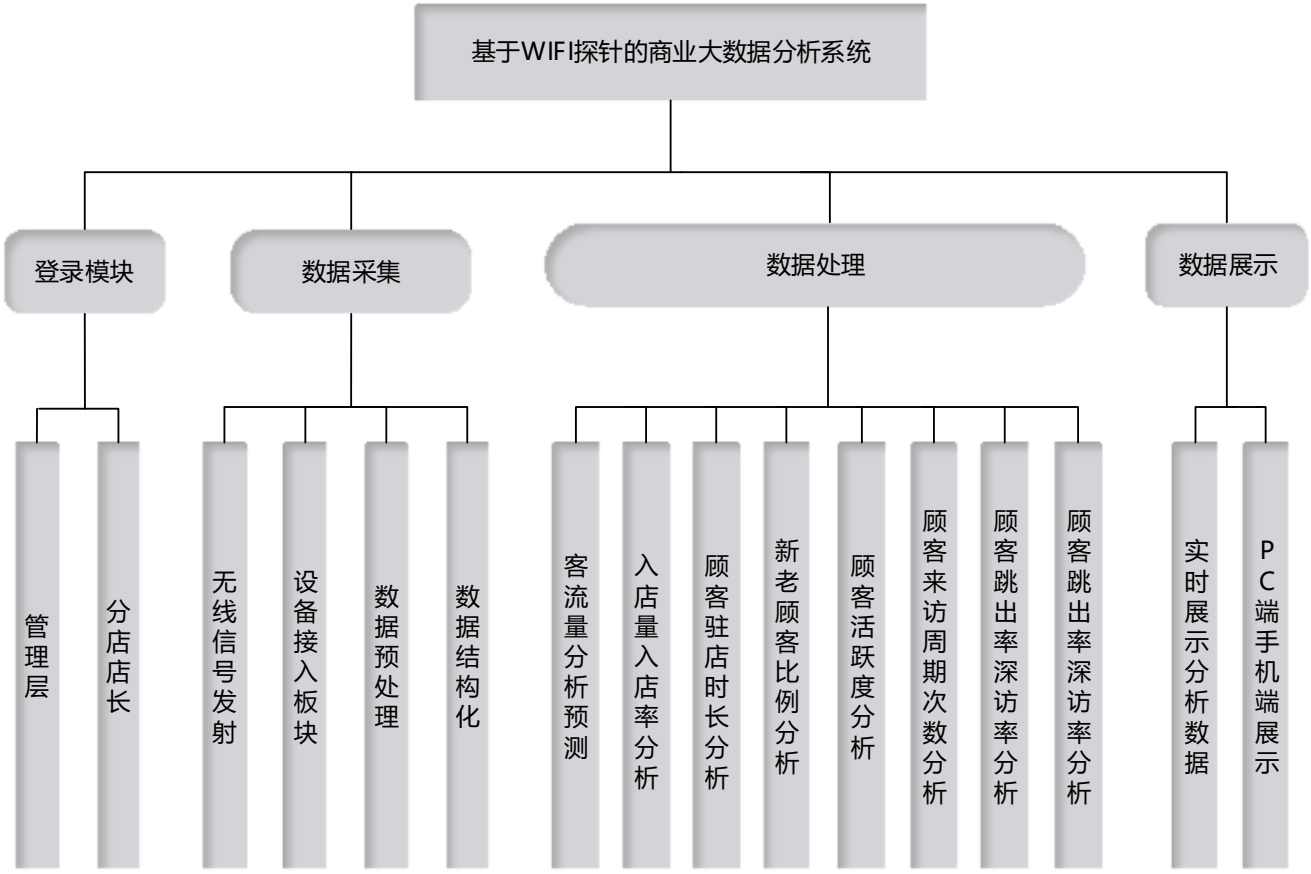

图 2 处理流程图

2.4.2 处理流程

2.4.2.1 系统总流程

注：数据的分析结果包括结构化数据历史比较以及环比结果

(1) 管理层：管理员登录系统；

(2) 分店店长：分店店长登录系统；

(3) 无线信号发射：探针发射 WIFI 信号；

(4) 设备接入模块：用户手持设备接收 WIFI 信号，并返回接入信息；

(5) 设备状态监控及设置：后台对所有接入 WIF 探针设备进行实时监控；

(6) 数据预处理：数据清洗、数据集成、数据变换和数据规约等一系列处理；

(7) 数据结构化：将数据存入数据库；

(8) 数据可视化：将数据库中的数据以图表的形式呈现给用户；

(9) 客流量分析：分析历史客流量及当前客流量，以图表形式实时展现；

(10)客流量的预测：根据上一时段的客流量数据预测下一时段以及未来几个时段的客流量；以图表形式展现；

(11)入店量、入店率分析：获取顾客实时入店量并以图表形式实时展示并进行环比与历史比较，呈现进入店铺或区域的客流及趋势。

(12)活跃度预测：根据顾客历史来访数据，计算顾客活跃度动态归类实时以图表形式分别展示高活跃度、中活跃度、低活跃度、沉睡活跃度的人数以及比例，预测顾客未来访店时间；

(13)驻店时长分析：根据顾客进店出店行为快速分析进入店铺的顾客在店内的停留时长进行动态归类以图表形式展示以及实时展示。

(14)来访周期分析：比照历史数据中距离上次来店的不同间隔对顾客群动态归类快速分析得出顾客来访周期，以图表形式实时展示；

(15)跳出率分析：根据已经分析出的驻店时长结构化数据来判断进入店铺后很快离店的顾客及占比(占总体客流)，并实现实时展示与小时、日、周、月多维度环比以及历史比较

(16)深访率分析：根据已经分析出的驻店时间长度结果根据顾客停留时长判定计算进入店铺深度访问的顾客及占比(占总体客流)即顾客深访率并予以实时展示与小时、日、周、月多维度环比以及历史比较

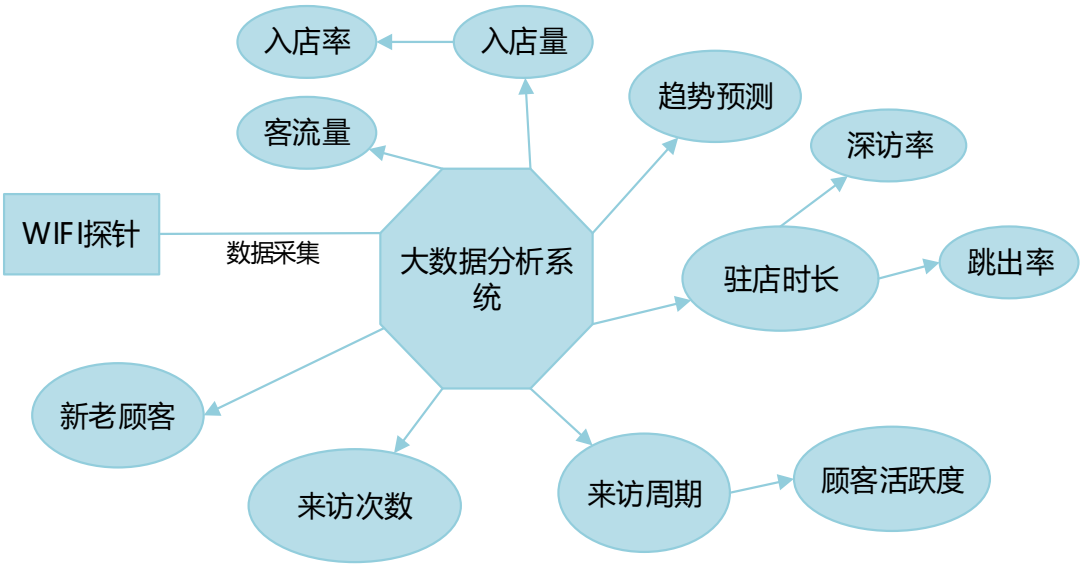

图 3 系统总流程图

2.4.2.1 分模块设计流程

基于 WIFI 探针的商业大数据分析系统功能需求包括 WIFI 探针设备、数据采集、数据分析（十一个指标）。

(1) WIFI 探针设备：WIFI 探针设备，可以进行服务器的相关配置（服务器 IP、端口、路径、发送时间间隔），能够采集 MAC 地址、地理位置信息、与探针的大概距离、采集时间等信息（特别针对 ANDROID6.0 和 IOS10 版本后的移动终端设备进行测试，要求能采集到 MAC 地址）。每三秒发送一次数据到接收服务器。

探针状态监控说明：

- 正常情况下通过探针监控页面关闭或打开探针不会使探针产生异常
- 探针当前状态如果数据库中卫 OFF，则恒显示为状态良好，如果为 ON 但通过数据库比对发现距离上一次收到数据超过五分钟，则为非正常
- 当服务器在接受探针发送过来的数据后检测发送数据的探针在数据库中状态，如果为 OFF 则修改为 ON

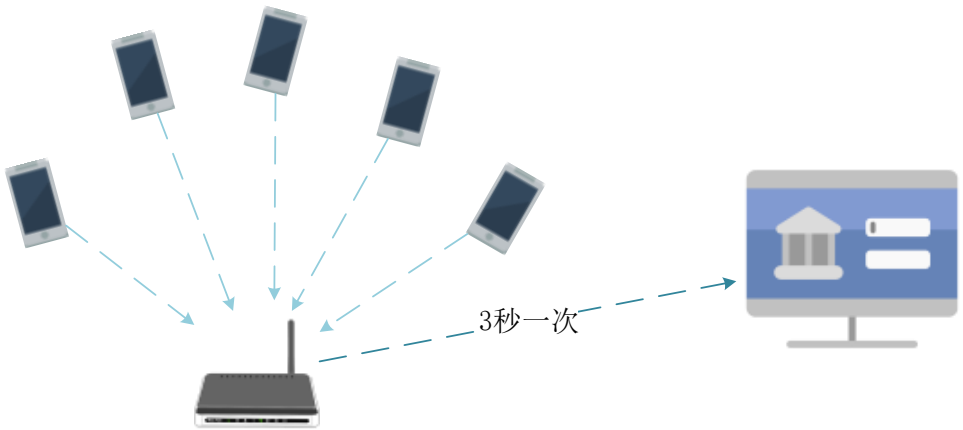

图 4　WIFI 探针流程图

(2) 数据采集：在本系统中，单独写一个服务器，用来接收探针定时发送的数据，将数据保存到数据分析平台待用，文件系统使用 HDFS 分布式文件系统，保存在 HBASE 数据库中，设置探针每三秒发送一次数据，采集的原始数据为 JSON 结构。

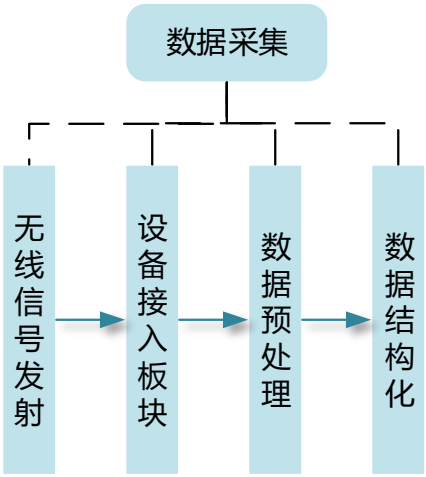

图 5 数据采集流程图

(3) 客流量：当前某店铺或区域整体客流及趋势，包括探针有效范围内检测到的全部 MAC 数。

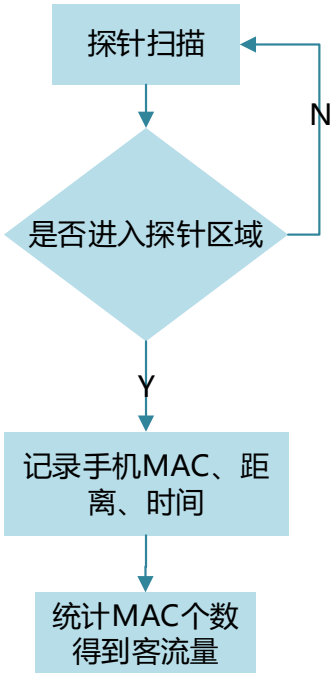

图 6 客流量分析流程图

(4) 入店量：当前进入店铺或区域的客流及趋势，几个探针同时定位一个 MAC 的位置，判断是否入店。

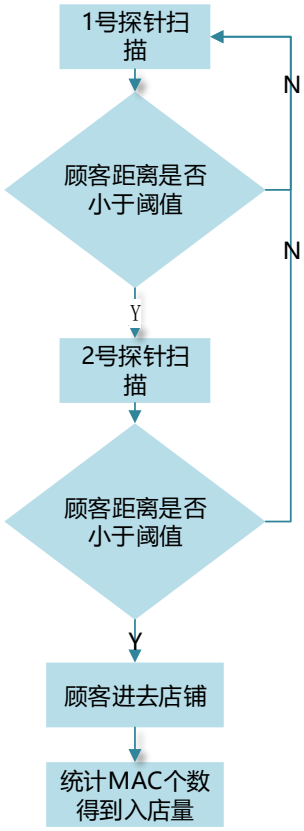

图 7 入店量分析流程图

(5) 入店率：进入店铺或区域的客流占全部客流的比例及趋势。入店率=入店量 / 客流量。

(6) 来访周期：进入店铺或区域的顾客距离上次来店的间隔，通过将到店 MAC 与历史到过店的 MAC 记录表中查找匹配，若存在则减去上次到店时间得到来访时间间隔。

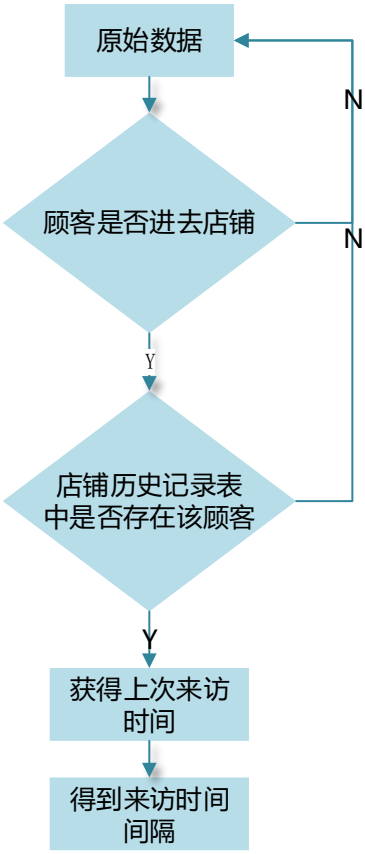

图 8 来访周期分析流程图

(7) 新老顾客：一定时间段内首次/两次以上进入店铺的顾客，通过记录到店 MAC 与历史到过店的 MAC 记录表中对比，若存在则说明是老顾客，否则为新顾客。

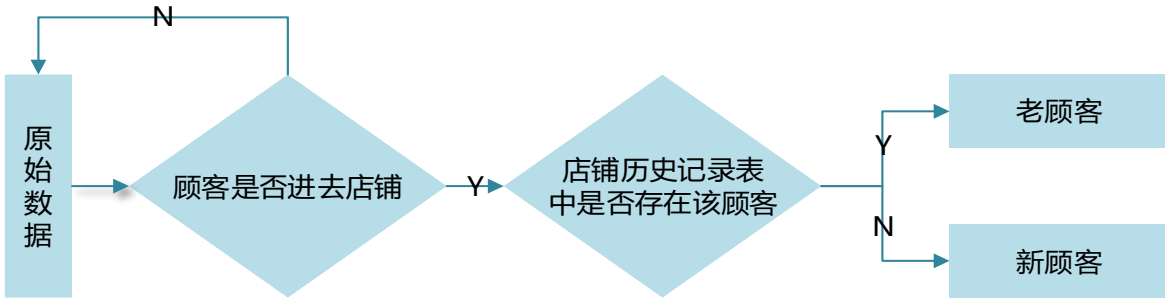

图 9 新老顾客分析流程图

(8) 顾客活跃度：按顾客距离上次来访间隔,划分为不同活跃度（高活跃度、中活跃度、低活跃度、沉睡活跃度）

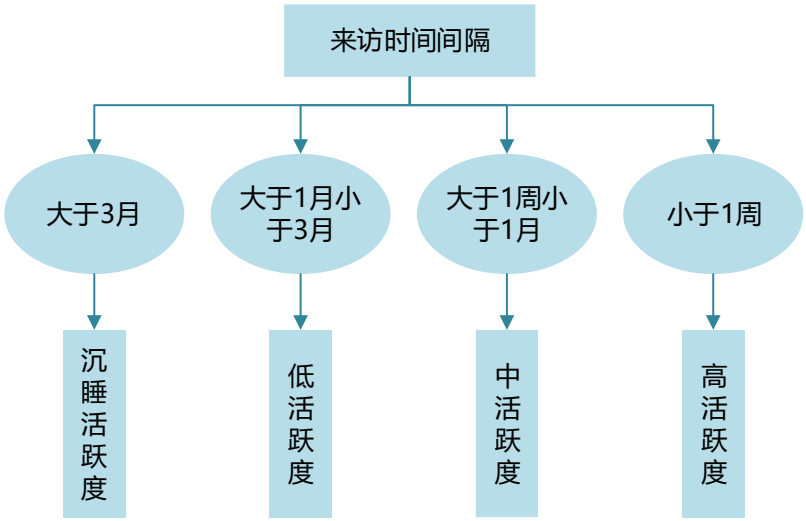

图 10 顾客活跃度分析流程图

(9) 驻店时长：进入店铺的顾客在店内的停留时长，通过记录顾客刚到店的时间戳，再判断该顾客已经出店的时间戳，记录时间和当前时间作差，得到驻店时长。

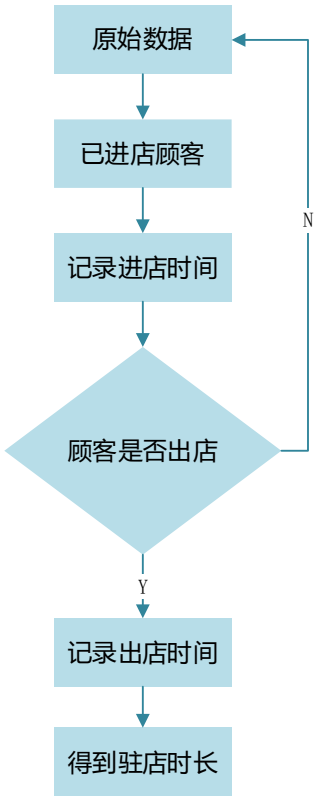

图 11 驻店时长分析流程图

(10) 跳出率：进入店铺后很快离店的顾客及占比(占总体客流)，跳出率=跳出数 / 客流量。

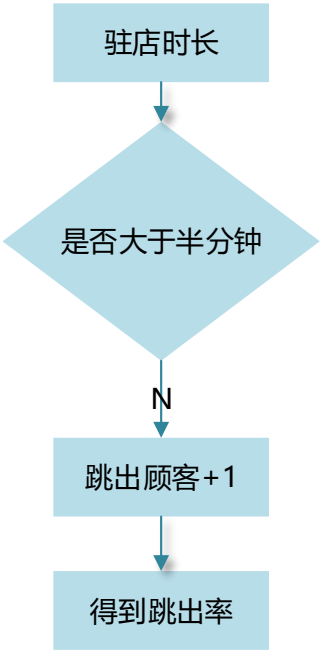

图 12 跳出率分析流程图

(11) 深访率：进入店铺深度访问的顾客及占比(占总体客流)，深访率 =深访数 / 客流量。

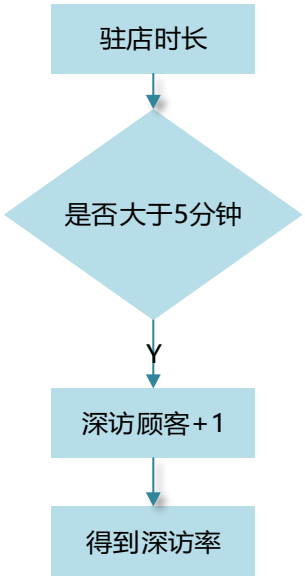

图 13 深访率分析流程图

(12) 访问次数：一段时间内顾客来访的次数，在其记录表中+1。

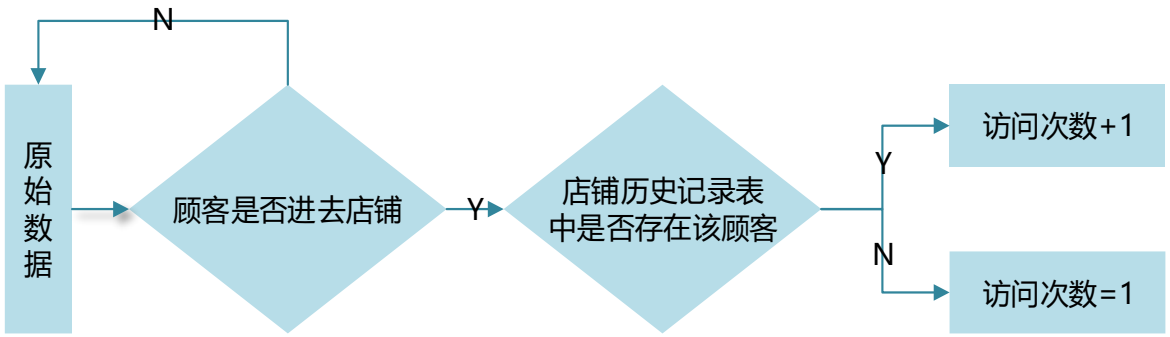

图 14 访问次数分析流程图

(13) 趋势预测：根据历史数据预测之后一段时间的客流量、入店量等，通过训练模型，来预测未来时间的流量情况。

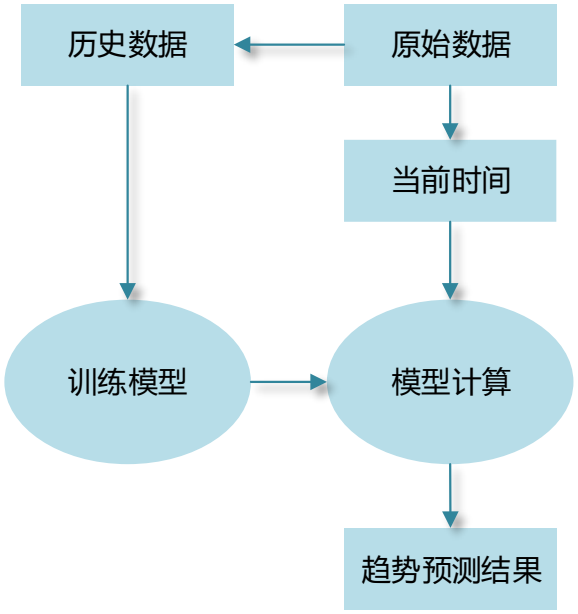

图 15 趋势预测分析流程图

2.5 系统部署方案

2.5.1 总体部署说明

基于 WIFI 探针的商业大数据分析系统的基础构架是由 WIFI 探针，接收服务器，分布式系统，分布式文件系统，分布式数据库，FTP 服务器，以及之间的相连网络和防火墙构成。

用户使用个人计算机，访问外部网络，并从外部的服务器中获取 WIFI 探针分析处理后的数据，并进行远程操作。个人计算机通过带有防火墙安全设置的网

络连接到应用服务器，并向应用服务器发送数据和操作请求。应用服务器与 FTP 服务器和数据库服务器直接相连，其根据个人计算机发送的请求，返回来自数据库服务器和 FTP 服务器的内容，或对数据库服务器和 FTP 服务器上的数据信息进行读写。

2.5.2 环境搭建流程

2.5.2.1 Hadoop2.2 环境搭建

我们用一台 Ubuntu 主机系统做 Clond1（Master），一台 Ubuntu 主机系做 Clond2（slave1），一台 Ubuntu 主机系做 Clond3（slave2），三台主机机器处于同一局域网下。

- 配置 SSH 无密码登录本机和访问集群机器

安装 openssh-server,并生成 ssh 公钥。

```
sudo apt-get openssh-server
ssh-keygen -t rsa -P ""
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
```

- JDK 和 Hadoop 安装配置

编辑~/.bashrc 文件，添加如下内容：

```
export JAVA_HOME=/usr/lib/jvm/default-java
export HADOOP_HOME=/usr/lib/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

- Hadoop 集群配置

修改 core-site.xml

```
<configuration>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/usr/lib/hadoop/tmp</value>
        <description>Abase for other temporary directories.</description>
    </property>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://clond1:9000</value>
```

```
        </property>
    </configuration>
```

修改 hdfs-site.xml：

```
    <configuration>
      <property>
            <name>dfs.replication</name>
            <value>3</value>
      </property>
      </configuration>
```

修改 mapred-site.xml(复制 mapred-site.xml.template,再修改文件名)

```
    <configuration>
      <property>
            <name>mapreduce.framework.name</name>
            <value>yarn</value>
      </property>
    </configuration>
```

修改 yarn-site.xml

```
 <configuration>
  <!-- Site specific YARN configuration properties -->
        <property>
              <name>yarn.nodemanager.aux-services</name>
              <value>mapreduce_shuffle</value>
        </property>
        <property>
              <name>yarn.resourcemanager.hostname</name>
              <value>clond1</value>
        </property>
    </configuration>
```

2.5.2.2 Spark2.1.0 环境搭建

Spark 分布式集群的安装环境，需要事先配置好 Hadoop 的分布式集群环境。

```
export  SPARK_HOME=/usr/lib/spark
export  PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
```

配置 spark-env.sh 文件

将 spark-env.sh.template 拷贝到 spark-env.sh

```
export SPARK_DIST_CLASSPATH=$(/usr/lib/hadoop/bin/hadoop classpath)
export HADOOP_CONF_DIR=/usr/lib/hadoop/etc/hadoop
export SPARK_MASTER_IP=clond1
```

(1) HBase 环境搭建

- 安装 HBase
- 配置 HBase

将 hadoop 添加到环境变量中

```
exportJAVA_HOME=/home/hadoop001/thirdparty/jdk1.7.0_51
export PATH=$HADOOP_HOME/bin:$PATH
```

编辑 hbase-site.xml ， 添加配置文件

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>hdfs://clond1:8020/hbase</value>
    </property>
    <property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
    </property>
    <property>
        <name>hbase.zookeeper.quorum</name>
        <value>clond1,clond2,clond3</value>
    </property>
    <property>
        <name>hbase.zookeeper.property.dataDir</name>
        <value>/root/zookeeper/data</value>
    </property>
</configuration>
```

编辑配置目录下面的文件 regionservers

```
Clond2
Clond3
```

(2) Docker 部署

我们选择配置好后上传镜像，主要过程与上面一样，区别在与，运行时的不同。

2.5.2.3 Docker 镜像配置

```
sudo docker run --name clond1 -h clond1 -it ubuntu:spark3.0
sudo docker run --name clond2 -h clond2 -it ubuntu:spark3.0
sudo docker run --name clond3 -h clond3 -it ubuntu:spark3.0
```

2.5.2.4 Tomcat 集群配置

- 环境要求
  a) 系统系统:linux
  b) 安装 xampp-linux-x64-1.8.3-5-installer.run
  c) 安装 jdk1.8
- Apache 配置
  a) 打开 xampp 文件目录，找到 opt/lamp/etc/httpd.conf 文件，修改需要获得管理员权限，由于不同版本的 apache 可能加载 module 文件可能不同，为防止少加载或重复加载，下面是我们加载部分

```
LoadModule authn_file_module modules/mod_authn_file.so
LoadModule authn_dbm_module modules/mod_authn_dbm.so
LoadModule authn_anon_module modules/mod_authn_anon.so
LoadModule authn_dbd_module modules/mod_authn_dbd.so
LoadModule authn_socache_module modules/mod_authn_socache.so
LoadModule authn_core_module modules/mod_authn_core.so
LoadModule authz_host_module modules/mod_authz_host.so
LoadModule authz_groupfile_module modules/mod_authz_groupfile.so
LoadModule authz_user_module modules/mod_authz_user.so
LoadModule authz_dbm_module modules/mod_authz_dbm.so
LoadModule authz_owner_module modules/mod_authz_owner.so
LoadModule authz_dbd_module modules/mod_authz_dbd.so
LoadModule authz_core_module modules/mod_authz_core.so
LoadModule authnz_ldap_module modules/mod_authnz_ldap.so
LoadModule access_compat_module modules/mod_access_compat.so
LoadModule auth_basic_module modules/mod_auth_basic.so
LoadModule auth_form_module modules/mod_auth_form.so
LoadModule auth_digest_module modules/mod_auth_digest.so
LoadModule allowmethods_module modules/mod_allowmethods.so
LoadModule file_cache_module modules/mod_file_cache.so
LoadModule cache_module modules/mod_cache.so
```

```
LoadModule cache_disk_module modules/mod_cache_disk.so
LoadModule socache_shmcb_module modules/mod_socache_shmcb.so
LoadModule socache_dbm_module modules/mod_socache_dbm.so
LoadModule socache_memcache_module modules/mod_socache_memcache.so
LoadModule dbd_module modules/mod_dbd.so
LoadModule bucketeer_module modules/mod_bucketeer.so
LoadModule dumpio_module modules/mod_dumpio.so
LoadModule echo_module modules/mod_echo.so
```

```
LoadModule case_filter_module modules/mod_case_filter.so
LoadModule case_filter_in_module modules/mod_case_filter_in.so
LoadModule buffer_module modules/mod_buffer.so
LoadModule ratelimit_module modules/mod_ratelimit.so
LoadModule reqtimeout_module modules/mod_reqtimeout.so
LoadModule ext_filter_module modules/mod_ext_filter.so
LoadModule request_module modules/mod_request.so
LoadModule include_module modules/mod_include.so
LoadModule filter_module modules/mod_filter.so
LoadModule substitute_module modules/mod_substitute.so
LoadModule sed_module modules/mod_sed.so
LoadModule charset_lite_module modules/mod_charset_lite.so
LoadModule deflate_module modules/mod_deflate.so
LoadModule mime_module modules/mod_mime.so
LoadModule ldap_module modules/mod_ldap.so
LoadModule log_config_module modules/mod_log_config.so
LoadModule log_debug_module modules/mod_log_debug.so
LoadModule logio_module modules/mod_logio.so
LoadModule env_module modules/mod_env.so
LoadModule mime_magic_module modules/mod_mime_magic.so
LoadModule cern_meta_module modules/mod_cern_meta.so
LoadModule expires_module modules/mod_expires.so
LoadModule headers_module modules/mod_headers.so
LoadModule usertrack_module modules/mod_usertrack.so
LoadModule unique_id_module modules/mod_unique_id.so
LoadModule setenvif_module modules/mod_setenvif.so
LoadModule version_module modules/mod_version.so
LoadModule remoteip_module modules/mod_remoteip.so
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_fcgi_module modules/mod_proxy_fcgi.so
LoadModule proxy_scgi_module modules/mod_proxy_scgi.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_express_module modules/mod_proxy_express.so
LoadModule session_module modules/mod_session.so
LoadModule session_cookie_module modules/mod_session_cookie.so
```

```
LoadModule session_dbd_module modules/mod_session_dbd.so
LoadModule slotmem_shm_module modules/mod_slotmem_shm.so
LoadModule ssl_module modules/mod_ssl.so
LoadModule lbmethod_byrequests_module modules/mod_lbmethod_byrequests.so
LoadModule lbmethod_bytraffic_module modules/mod_lbmethod_bytraffic.so
LoadModule lbmethod_bybusyness_module modules/mod_lbmethod_bybusyness.so
LoadModule lbmethod_heartbeat_module modules/mod_lbmethod_heartbeat.so
LoadModule unixd_module modules/mod_unixd.so
LoadModule dav_module modules/mod_dav.so
LoadModule status_module modules/mod_status.so
LoadModule autoindex_module modules/mod_autoindex.so
LoadModule info_module modules/mod_info.so
LoadModule suexec_module modules/mod_suexec.so
LoadModule cgi_module modules/mod_cgi.so
LoadModule cgid_module modules/mod_cgid.so
LoadModule dav_fs_module modules/mod_dav_fs.so
LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule dir_module modules/mod_dir.so
LoadModule actions_module modules/mod_actions.so
LoadModule speling_module modules/mod_speling.so
LoadModule userdir_module modules/mod_userdir.so
LoadModule alias_module modules/mod_alias.so
LoadModule rewrite_module modules/mod_rewrite.so
```

b) 去除 Include conf/extra/httpd-vhosts.conf 的注释

c) 在/opt/lamp/modules 中加入文件 mod_jk.so(版本需与 apache 匹配)

d) 在文末加上下面的语句：

```
ProxyPass / balancer://cluster/ stickysession=JSESSIONID
ProxyPassReverse / balancer://cluster/
 <proxy balancer://cluster>
            BalancerMember ajp://127.0.0.1:8009 loadfactor=1 route=tomcat1
            BalancerMember ajp://127.0.0.1:9009 loadfactor=1 route=tomcat2
 </proxy>
```

其中 proxy 中 ajp 是集群部署的 tomcat 占用的的 ajp 端口

- 配置 tomcat

  将 XAMPP 中的 tomcat 复制两份命名为 tomcat1 与 tomcat2，

  修改 tomcat1 中的 server.xml 文件，仅仅只需要修改一下部分

<Engine name="Catalina" defaultHost="localhost" jvmRoute="tomcat1">

jvmRoute 与上述 route1 对应；去除<Cluster className=

"org.apache.catalina.ha.tcp.SimpleTcpCluster"/>的注释；修改 tomcat2 中的

server.xml 文件，不进需要完成上述修改(注意 jvmRoute 与 route2 对应)，还需要将一下的端口修改:

```
<Server port="9005" shutdown="SHUTDOWN">
<Connector port="9080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
<Connector port="9009" protocol="AJP/1.3" redirectPort="8443" />
```

- 由于集群主要是为了处理探针数据接收，因而没有配置 session 共享

2.6 系统界面设计

说明：界面总体采用 BootStrap 框架搭建，可实现自适应展示，PC 端、手机端可自由切换。ECharts3.0 图表的使用，使得图表的展示更加简洁与可视化增强，交互性增强。

（1） 用户登录界面：

必须在数据库中设置成为系统用户之后，方可使用系统，正确输入 ID 和密码即可进入系统。

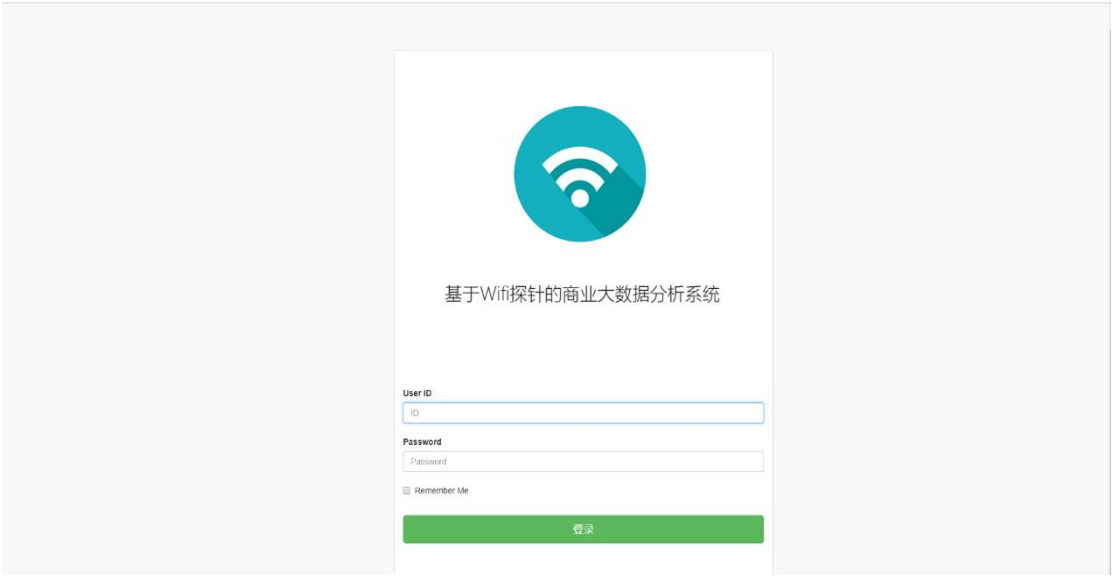

图 16 登录界面

（2）客流量展示界面

用户登陆成功之后进入客流量分析展示界面，分三个模块展示，客流量历史比较、客流量预测、实时客流量以及变化趋势展示、和客流量环比多维度展示。

其中环比展示模块中添加过去一天的客流量分布展示，便于商家对营业信息有一个全面的了解增强了信息的反馈力度。

此外客流量预测数据展示便于商家对现阶段的营销方案进行及时调整。

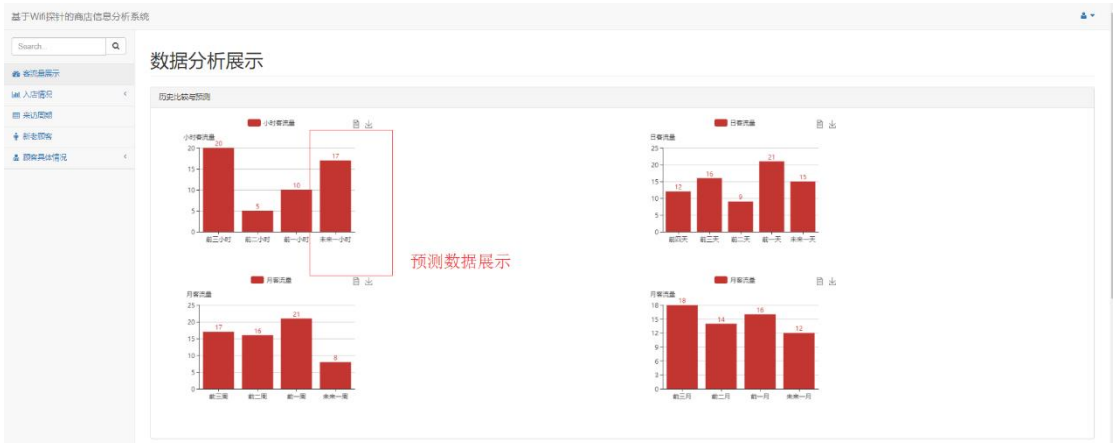

图 17 客流量历史比较与预测

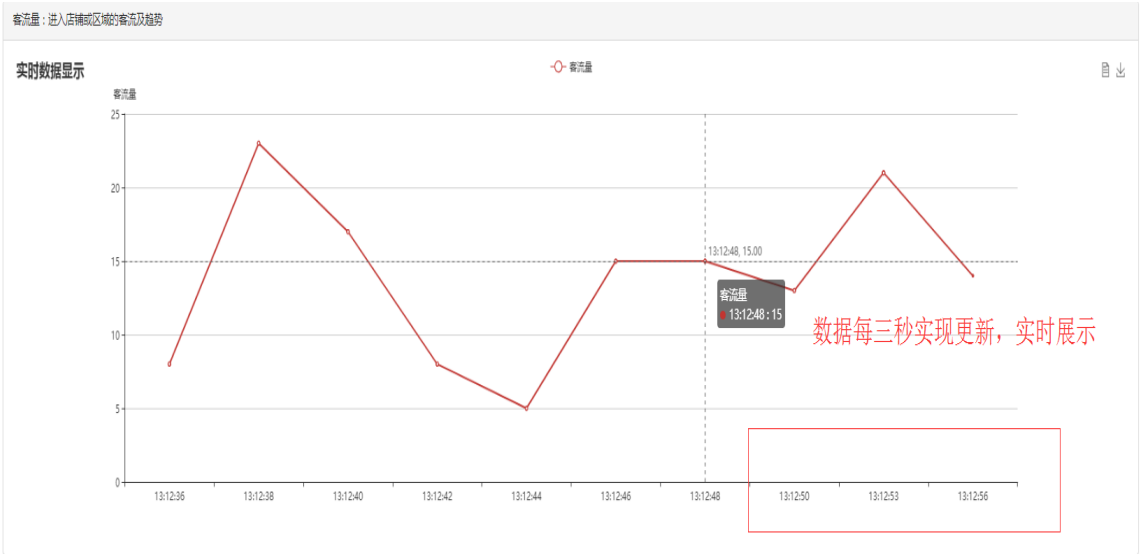

图 18 实时客流量及变化趋势图

对数据具有鲜明特点的周客流量数据进行单独分析与预测

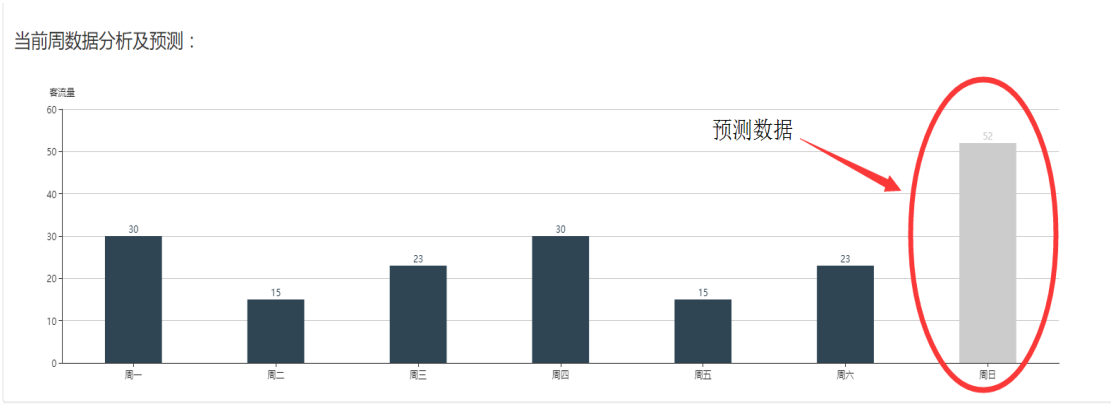

图 19 周预测

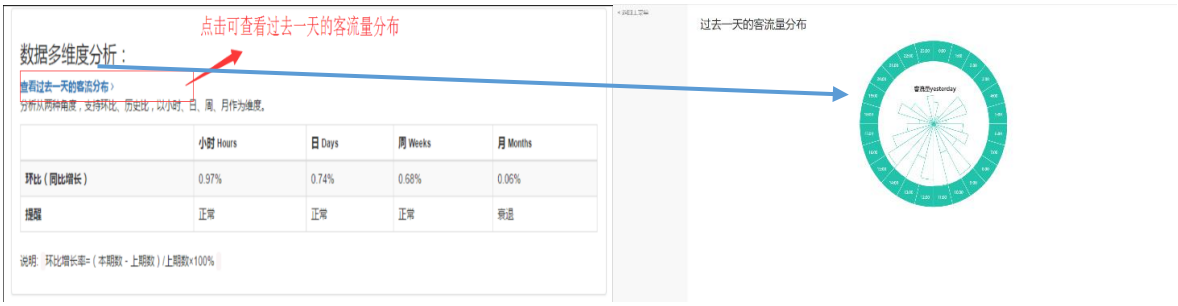

图 20 数据多维度环比以及过去一天的客流量分布图

（3）入店量、入店率分析展示页面：

点击系统左侧导航栏第二个选项入店情况，下拉菜单中第一个选项则为入店量分析展示界面，第二个选项则为入店率分析展示界面，二者展示形式同客流量展示，侧重历史比较、预测、实时展示、以及环比、多维度（小时，日，周，月）分析的展示。界面集中反映了各项指标，动态展示，便于用户及时，快速获取信息。

可视化形式多样，有数据视图展示，并可对实时数据进行一键截图，便于分析数据。

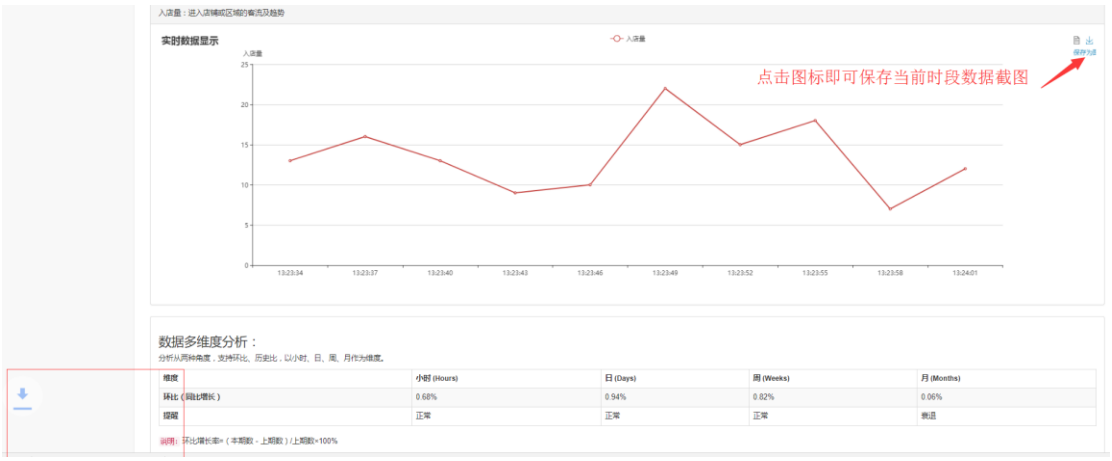

图 21 入店率-实时分析展示界面

（2） 来访周期界面：

顾客来访周期界面展示，以柱状图形式展示不同来访时间间隔的顾客人数以饼状图形式展示不同类型顾客所占比例。

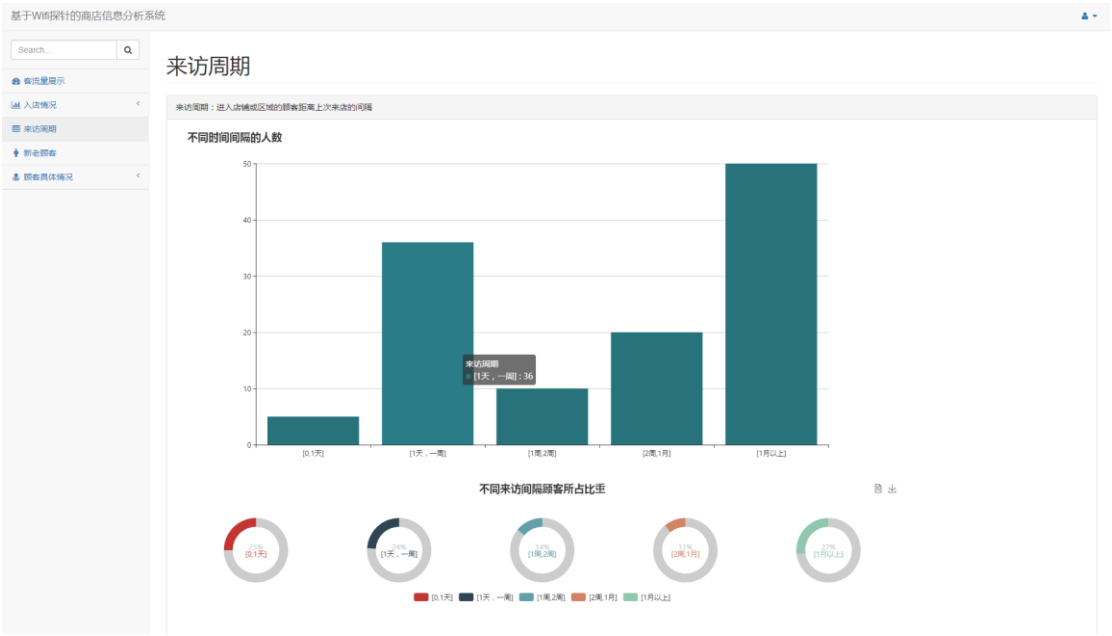

图 22 来访周期界面

（3） 新老顾客界面

注重新老顾客比例的展示，此外对新老顾客的实时人数予以展示，具有完成的分析内容展示，如历史比较、预测，环比分析，多维度分析，数据异步加载实时更新。

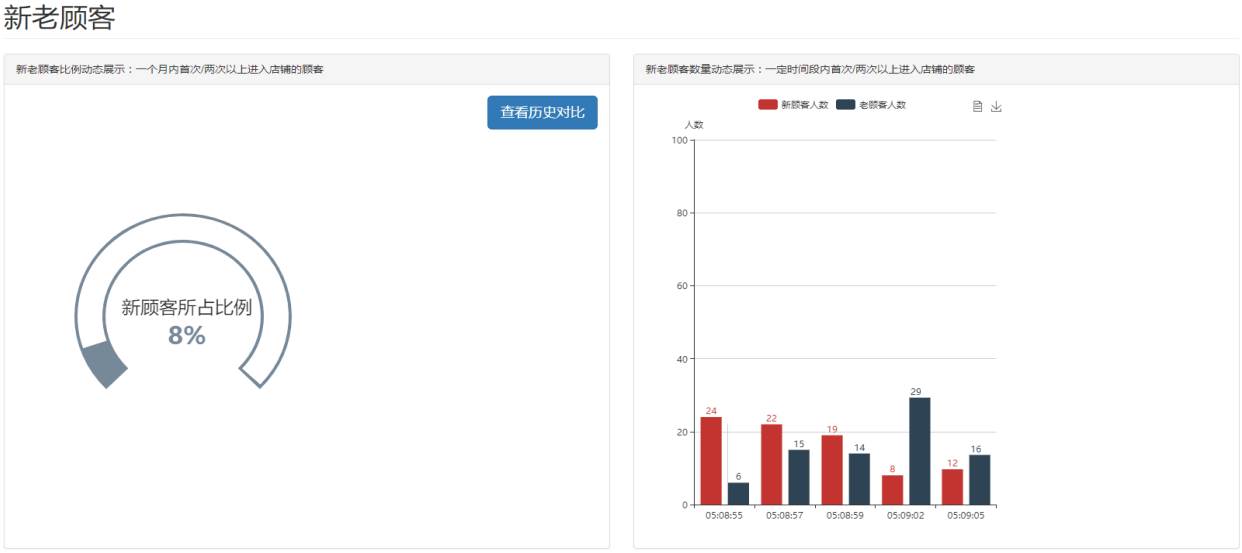

图 23 新老顾客实时比例展示界面

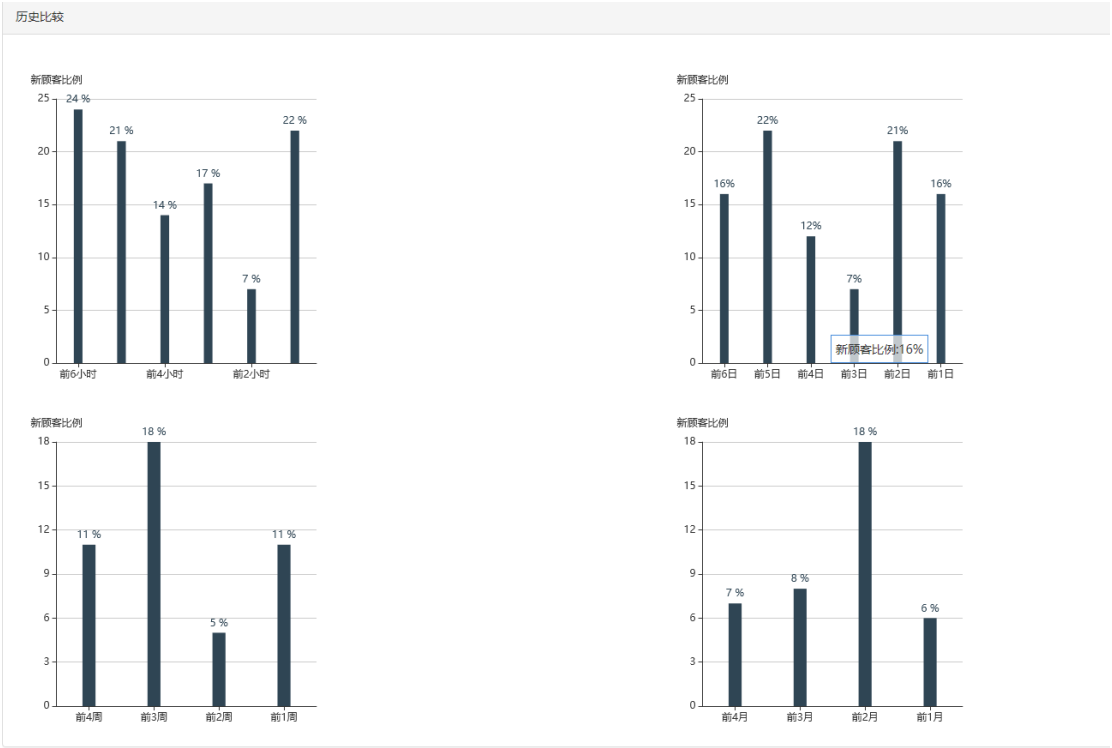

图 24 新老顾客分析历史比较与预测界面

（4） 顾客活跃度界面：

顾客活跃度，实时展示后台分析数据，根据顾客在一段时间内入店频率判定顾客活跃度，分四个层次分别为高活跃度、中活跃度、低活跃度、沉睡活跃度，以南丁格尔图形式展示根据色系冷暖以及文字说明区别活跃度种类，数据展示清晰，界面较友好。

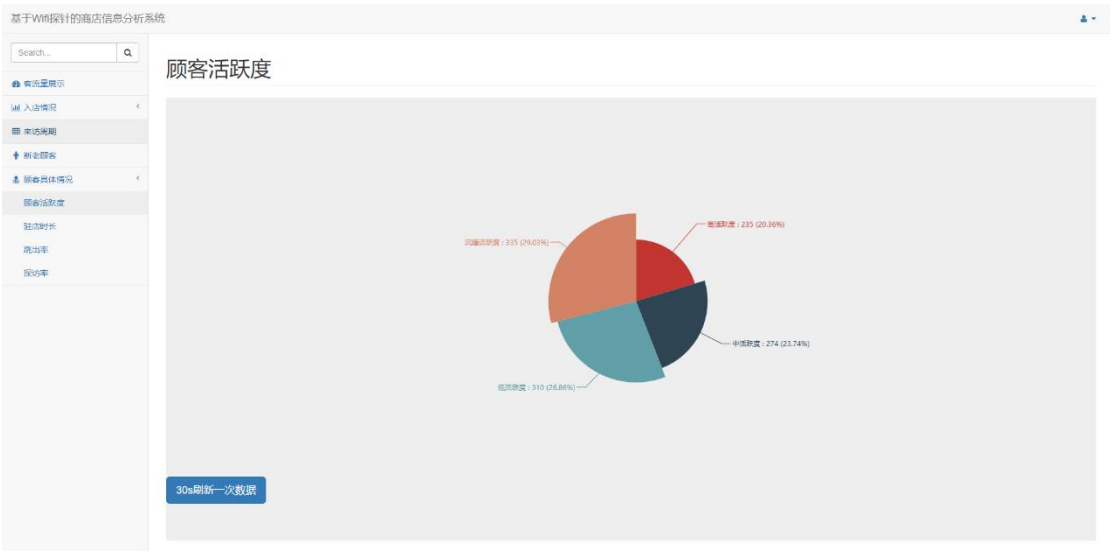

图 25 活跃度-实时分析展示界面

（5） 顾客驻店时长界面

采用 ECHARTS 图表中异步加载数据方式展示数据，以四种不同颜色展示不同在店内的停留时长的顾客人数，搭配有数据列表视图。界面对每三秒异步加载一次后的数据进行实时展示。

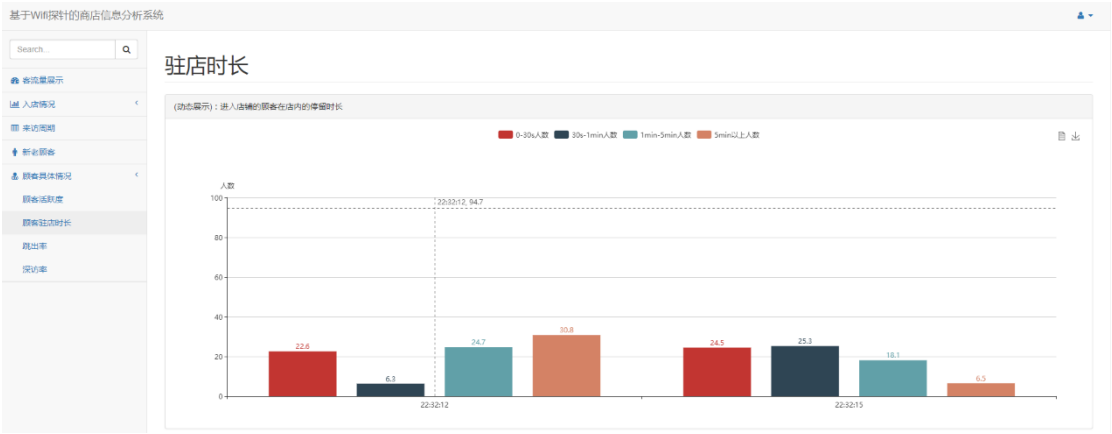

图 26 顾客驻店时长-实时分析展示界面

（6） 深度访问率

对入店顾客进行分析，根据驻留时长大于规定的阈值的顾客（视作深入访问）以比例仪表盘式图表的形式予以动态实时展示。

界面提供历史比较查看入口，与环比图表展示。

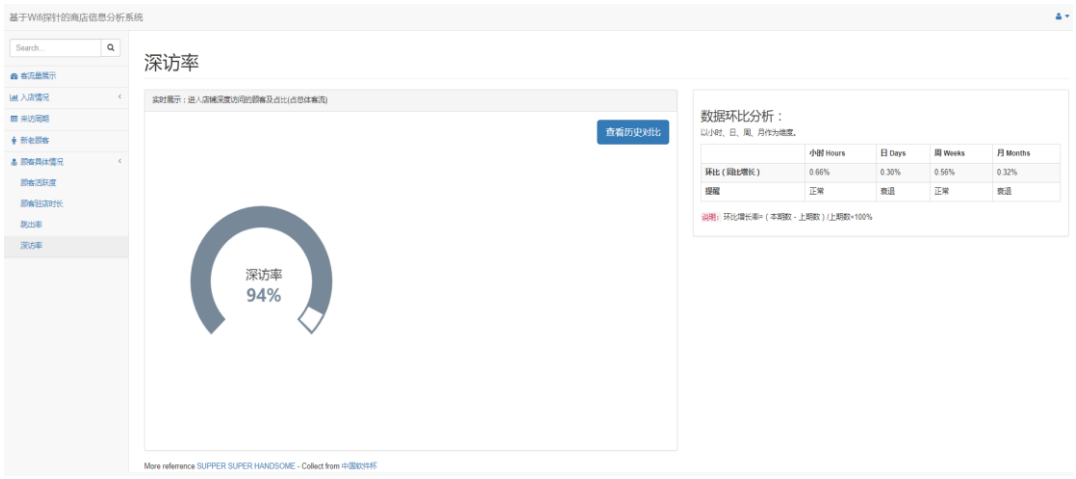

图 27 深入访问率实时分析界面

（7） 跳出率界面

对入店顾客进行分析，根据驻留时长小于规定阈值的顾客（视作跳出访问）以比例仪表盘式图表的形式予以动态实时展示。界面提供历史比较查看入口，与环比图表展示。

图 28 跳出率实时分析界面

（8） 探针状态监控与设置界面

系统主界面右上角提供探针状态及设置页面的入口，可实时查看探针状态，并且对状态进行远程设置，但设置探针状态需要经过验证，保证了系统的稳定行与安全性。

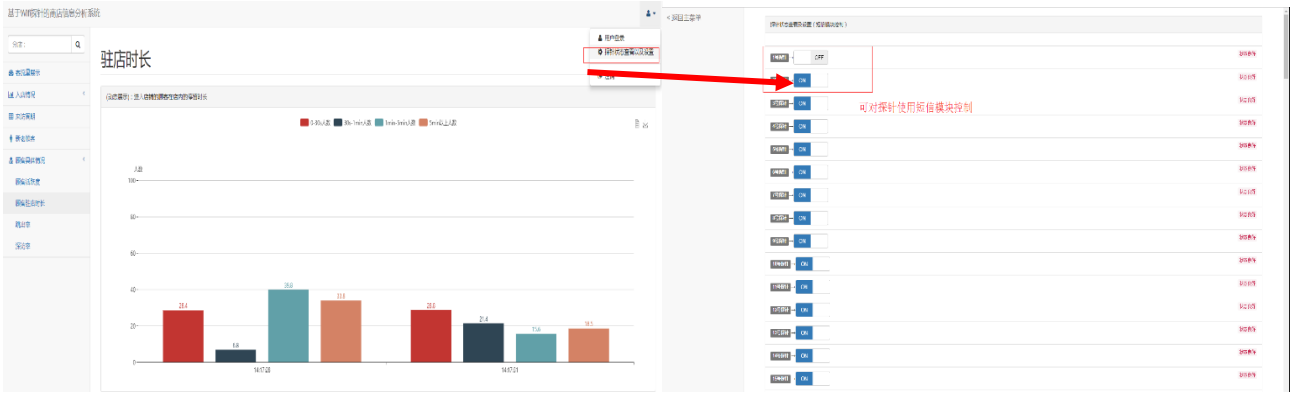

图 28 探针状态监控及设置界面

3.接口设计

3.1 用户接口

管理层和各门店负责人的账号和密码是直接从后台输入并分配到相应人员的。当用户登录进入操作界面之前，我们设置了用户名和密码，只有当用户名和密码相匹配时才能进入该用户所允许的权限操作界面，否则就只能看到登录界面。管理员有权利行使所有的管理功能，门店用户只能进行一般的查询。系统用户管理保证了只有授权的用户才能进入系统进行数据操作，而且对一些重要数据，系统设置为只有更高权限的人员方可读取或是操作。系统安全保密性较高。

在使用系统的过程中，管理员进行分店管理时，需要输入要管理的分店编号，需要系统进行按编号进行查询店内信息。

使用鼠标、键盘、触摸屏等外部构件进行功能选择及输入。

其余均为可视化界面，用户可以根据系统提示进行功能选择。

3.2 外部接口

本系统采用 B/S 结构，需要三台电脑配置分布式系统，用户通过客户端与系统交互，获取系统服务。

类别	方向	名称	描述
硬件接口	输出	WIFI 探针	与外部网络之间进行连接，发散无线网络
	输入	WIFI 探针	接收用户连接 WIFI 信息
	输出	终端显示设备	录入初始化系统使用基本信息
	输入	键盘鼠标	提供探针历史数据的高速读取
	输入输出	WIFI 探针设置	实时监控 WIFI 探针状态并使用短信模块进行设置
软件接口	输入输出	NOSQL 数据库	提供当日采集后结构化数据的存储
	输入输出	模型训练运行	提供模型初始化的训练，以及对输入数据进行分析和预测的服务
	输入输出	数据可视化	将输入数据进行可视化处理，并将其展示在显示终端

3.3 内部接口

本系统分为三个大模块分别为用户模块、管理层模块，通过面向对象语言设计类，类实现封装，模块和模块之间信息交流使用类、函数等进行相互调用。

模块 A	模块 B	方式

原始数据	\	返回值
NOSQL 数据库	\	返回值
模型训练与运行	\	函数调用
数据可视化	\	函数调用
模型训练与运行	NOSQL 数据库	参数传递

4.运行设计

4.1 运行模块组合

客户机程序在有数据输入时启动数据接收模块，调用各模块之间数据，自行读入并对输入数据进行格式化。当数据接受模块获得充足的数据时，系统将调用网络传输模块，通过网络将数据发送到服务器，并等待接收服务器的返回的信息。接收到返回信息后随即调用数据输出模块，对信息进行处理并输出相应的数据报告。

服务器程序的数据接受模块必须时刻处于活动状态。当服务器接收到数据后，会调用数据处理/查询模块对数据库进行访问，完成后调用网络发送模块，将信息返回给客户机。

这样就完成了数据的读取，输入，格式化，接收，传送，处理等一系列操作。

4.2 运行控制

运行控制将严格按照各模块间函数调用关系来实现。系统需各模块中心机制进行正确的判断并选择正确的运行路径

网络方面，客户机在向服务器发送数据后，将等待其反馈的确认收到信号；如收到后，将再次等待服务器反馈的回答数据，之后对数据进行确认。在服务器

接收到数据后并发送确认信号，经过对数据处理、访问数据库后，将返回信息送回客户机，并等待确认。

4.3 运行时间

为了满足客户对本系统运行时间的要求,我们采用高速的 ATM 网络并且提高服务器的性能,以尽量减少数据库的访问时间的长短。其次硬件的运行速度对系统整体的运行速度也较大,我们将采用较好的硬件设施。结合以上两点的改进,以确保将该系统的运行时间控制在 2.5s 以内。

5.数据库设计

本系统采用分布式数据库 HBASE，构建在 HDFS 上的分布式列存储系统，从逻辑上讲，HBASE 将数据按照表、行、列逻辑进行存储。

5.1 数据库表列表

序号	名称	描述
1	‘WIFI 探针名’	存储该探针采集到的实时数据
2	'temp_log'	存储当天每个时段的客流量数据
3	'store_temp'	存储该店每个时段店内人数
4	'store_log'	存储当天以及过去日、周、月的的入店量、入店率记录
5	'store_time'	存储根据驻店时长长短分类的各类人数划分阈值为[30s,30s-1min,1min-5min,5min 以上]
6	'new_or_old_history'	存储根据顾客上次来访时间判别的新老顾客（间隔一个月为阈值）的人数及比例
7	'human_times'	存储顾客在一段时间内访问次数和每次访问的时间

8	'human_history'	存储顾客上次访问时间和上一次的访问时间间隔
9	'count_human_times'	存储店铺在一段时间内的被访问次数频率阈值为 [1，2，3，4，5，5 以上]
10	'shop_times'	存储店铺在一段时间内顾客的驻店时间频率阈值为 [1min,5min,10min,15min,30min,60min,60min 以上]
11	'interview_times'	存储店铺在一段时间内顾客访问时间间隔频率阈值为 [12h，24h，36h，48h，60h，72h，72h 以上]
12	'jump_log'	存储根据驻店时长划分，标准为低于 30s 视为跳出顾客来判定的人数
13	'deep_log'	存储根据驻店时长划分，标准为高于 30min 视为深访顾客来判定的人数
14	'activity_log'	存储根据顾客访问次数得出的活跃度及历史数据

5.2 逻辑结构设计

5.3 物理结构设计

(1) 客流量表（‘WIFI 探针名’）

行键	列族	列标识	时间戳	说明
‘MAC’	‘range’	1	64 位整型	距离
	‘time’	1	64 位整型	时间

(2) 客流量历史表（'temp_log'）

行键	列族	列标识	时间戳	说明
‘time’	‘count’	1	64 位整型	客流量

(3) 当前店内人数表（'store_temp'）

行键	列族	列标识	时间戳	说明
‘MAC’	‘time’	1	64 位整型	当前店内人数

(4) 入店量、入店率历史表（'store_log'）

行键	列族	列标识	时间戳	说明
‘time’	‘count’	1	64 位整型	入店量
	‘rate’	1	64 位整型	入店率

(5) 顾客驻留时间表（'store_time'）

行键	列族	列标识	时间戳	说明
‘MAC’	‘in_time’	1	64 位整型	进店时间
	‘out_time’	1	64 位整型	出店时间
	'subtract'	1	64 位整型	时间间隔

(6) 新老顾客人数比例的历史表（'new_or_old_history'）

行键	列族	列标识	时间戳	说明
‘time’	‘new’	1	64 位整型	新顾客人数
	‘old’	1	64 位整型	老顾客人数
	'rate'	1	64 位整型	新老顾客人数比例

(7) 顾客访问次数及时间表（'humam_times'）

行键	列族	列标识	时间戳	说明
‘MAC’	‘time’	1…values	64 位整型	每次访问时间
	‘values’	1	64 位整型	来访次数

(8) 顾客访问时间间隔表（'humam_history'）

行键	列族	列标识	时间戳	说明
‘MAC’	‘last_time’	1	64 位整型	上上次来访时间
	‘this_values’	1	64 位整型	上次来访时间
	'subtract’	1	64 位整型	最近一次来访时间间隔

(9) 顾客访问次数历史表（'count_human_times'）

行键	列族	列标识	时间戳	说明
‘time’	‘1’	1	64 位整型	来访次数 1
	‘2’	1	64 位整型	来访次数 2
	‘3’	1	64 位整型	来访次数 3
	‘4’	1	64 位整型	来访次数 4
	‘5’	1	64 位整型	来访次数 5
	‘other’	1	64 位整型	来访次数 5 以上

(10) 顾客驻店时间历史表（'shop_times'）

行键	列族	列标识	时间戳	说明
‘time’	‘1’	1	64 位整型	驻店时间小于 1min
	‘5’	1	64 位整型	驻店时间小于 5min
	‘10’	1	64 位整型	驻店时间小于 10min
	‘15’	1	64 位整型	驻店时间小于 15min
	‘30’	1	64 位整型	驻店时间小于 30
	‘60’	1	64 位整型	驻店时间小于 60min
	‘other’	1	64 位整型	驻店时间大于 60min

(11) 顾客访问周期历史表（'interview_times'）

行键	列族	列标识	时间戳	说明
‘time’	‘12’	1	64 位整型	12h 内来访次数大于等于 2 次
	‘24’	1	64 位整型	24h 内来访次数大于等于 2 次
	‘36’	1	64 位整型	36h 内来访次数大于等于 2 次
	‘48’	1	64 位整型	48h 内来访次数大于等于 2 次
	‘60’	1	64 位整型	60h 内来访次数大于等于 2 次
	‘72’	1	64 位整型	72h 内来访次数大于等于 2 次
	‘other’	1	64 位整型	大于 72h 来访次数大于等于 2 次

(12) 顾客跳出率历史表（'jump_log'）

行键	列族	列标识	时间戳	说明
‘time’	‘jump_human’	1	64 位整型	跳出顾客数
	‘all_human’	1	64 位整型	客流量
	‘rate’	1	64 位整型	跳出率

(13) 顾客深访率历史表（'deep_log'）

行键	列族	列标识	时间戳	说明
‘time’	‘num’	1	64 位整型	深访顾客数
	‘all	1	64 位整型	客流量
	‘rate’	1	64 位整型	深访率

(14) 活跃度历史表（'activity_log'）

行键	列族	列标识	时间戳	说明
‘time’	‘high’	1	64 位整型	高活跃度
	‘medium’	1	64 位整型	中活跃度
	‘low’	1	64 位整型	低活跃度

	‘sleep’	1	64 位整型	沉睡活跃度

6.系统出错处理设计

6.1 出错信息

使用一览表的方式说明每种可能的出错或故障情况出现，系统输出信息的形式、含意及处理方法。

出错的位置	出错的原因	提示信息
管理员账号	输入数据库中不存在该账号	用户名错误
管理员密码	输入数据库中不存在该账号	密码错误
管理层账号	输入数据库中不存在该账号	用户名错误
管理层密码	输入的密码不正确	密码错误
门店负责人账号	输入数据库中不存在该账号	用户名错误
门店负责人密码	输入的密码不正确	密码错误
顾客账号	输入数据库中不存在该账号	用户名错误
顾客密码	输入的密码不正确	密码错误
查询客户信息	输入查询信息不正确	没有符合条件的记录
添加信息	添加信息不完整	填写完整信息
更新历史数据	未查找到相关历史数据	更新失败
功能使用	该功能已超出权限	该功能已超出权限

6.2 补救措施

(1) 定期进行数据备份，每天进行增量转储，每月一次进行海量转储，可使用静态转储也可使用动态转储；

(2) 若发生服务器崩溃的致命性错误，需要进行手工错作病人共记录下所需要记录的数据，当系统恢复正常工作时，再把在此期间记录的数据重新放入系统中；

(3) 使用具有检查点的恢复技术，检查点记录的内容包括：建立检查点时刻所有正在执行的事务清单；这些事务最近一个日志记录的地址。利用系统故障恢复的方法（BDR），同时建立检查点，保存数据库状态，具体步骤为：

1．将当前日志缓冲中的所有日志记录写入磁盘的日志文件上；

2．在日志文件中写入一个检查点记录；

3．将当前数据缓冲的所有数据记录写入磁盘的数据库中；

4．把检查点记录在日志文件中的地址写入一个重新开始文件。系统出现故障时恢复子系统将根据事务的不同状态采取不同的恢复策略。

(4) 后备技术，当万一原始系统数据丢失时所启用的副本的建立和启动技术，我们将定期将磁盘上的信息存储记录到磁带上，以供备用；

(5) 降效技术，使用另一个效率稍低的系统或方法来求得所需结果的某些部分，我们将采用；

(6) 手工操作和人工记录一些核心数据；

(7) 恢复及再启动技术，使软件从故障点恢复执行或使软件从头开始重新运行。

7.系统维护设计

(1) 该系统的维护主要包括硬件维护和软件维护两大方面，而软件维护又包括数据库维护和软件功能的维护。

(2) 对于硬件维护，我们会定期对系统的硬件设施进行查修保养，做到早发现早修正，以免出现大面积的损伤。

(3) 对于数据库维护，本系统主要由管理员对数据库基本结构进行管理维护。

(4) 对于软件维护，本系统采用的是模块化的设计方法，各模块之间相互独立性较高，这对后期操作人员维护系统带来了很大的方便。如需修改某个单独功能，只需修改相关页面即可；如需添加某功能，只需再添加页面选项的内容即可。