2 动物识别系统相关理论及技术综述

本章对动物识别系统中使用到的理论和技术基础进行简要介绍。系统的卷积网络模型采用VGG16网络与RepVGG网络。模型的损失函数采用交叉熵损失函数与负对数似然损失函数。优化器采用Adam优化器与Momentum优化器。系统依托AI Studio深度学习开源平台线上BML codelab开发环境，采用支持python3.9的paddle paddle深度学习框架，完成对模型的训练，最后采用AI Studio的在线部署和预测功能，创建一个在线服务, 应用模型生成在线API。

2.1 AI Studio平台

AI Studio是基于[百度](https://baike.baidu.com/item/%E7%99%BE%E5%BA%A6/6699?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/AI%20Studio/_blank)深度学习开源平台[飞桨](https://baike.baidu.com/item/%E9%A3%9E%E6%A1%A8/23472642?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/AI%20Studio/_blank)的人工智能学习与实训社区，为开发者提供了功能强大的线上训练环境、免费[GPU](https://baike.baidu.com/item/GPU/105524?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/AI%20Studio/_blank)算力及存储资源，系统采用了AI Studio中的BML开发平台，paddle paddle深度框架，和AI Studio的在线部署和预测功能，下面将依次对这三个概念进行介绍。

2.1.1 BML开发平台

BML 全功能 AI 开发平台是一个面向企业和个人开发者的机器学习集成开发环境，为经典机器学习和深度学习提供了从数据处理、模型训练、模型管理到模型推理的全生命周期管理服务，其具有以下优势：

（1）提供高性能的算力支持，支持多机多卡分布式训练，提供丰富算力机器资源选型。

（2）具有强大的数据处理服务，是业界首个开放的数据采集和清洗服务，支持自动数据增强，助力模型效果有效提升。

（3）高精度训练效果，其预置百度超大规模视觉预训练模型，预置业界领先的 NLP 预训练模型[文心 ERNIE](https://ai.baidu.com/wenxin" \t "https://ai.baidu.com/ai-doc/BML/_blank)，支持自动超参搜索、不平衡数据处理等先进训练机制，支持视觉、文本、机器学习等丰富任务类型。

（4）具有丰富完善的开发方案，相比于 Jupyter Notebook，其提供适合 AI 初学者的脚本调参开发方式，快速上手。支持开放框架多语言的作业建模，高度灵活，提供本地客户端[BML CodeLab](https://cloud.baidu.com/solution/bml/codelab.html" \t "https://ai.baidu.com/ai-doc/BML/_blank)，预置海量丰富插件。

2.1.2 paddle paddle深度学习框架

 PaddlePaddle 是一个易用、高效、灵活、可扩展的深度学习框架，致力于让深度学习技术的创新与应用更简单。其语法简洁，API的设计干净清晰 丰富的模型库。借助于其丰富的模型库，可以非常容易的复现一些经典方法。具有全中文说明文档。运行速度快。其充分利用 GPU 集群的性能，可以为分布式环境的并行计算进行加速。

2.1.3 AI Studio的在线部署和预测功能

在线部署与预测为开发者提供训练模型向应用化API转换的功能. 开发者在AI Studio平台通过NoteBook项目完成模型训练后, 在Notebook详情页通过创建一个在线服务, 应用模型生成在线API, 使用该API可以直接检验模型效果或实际应用到开发者的私有项目中。目前, 该功能暂时仅对Notebook项目开放。

2.2 卷积网络

卷积神经网络（Convolutional Neural Network, CNN）是一种[前馈神经网络](https://zh.wikipedia.org/wiki/%E5%89%8D%E9%A6%88%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C" \o "前馈神经网络)，它的人工神经元可以响应一部分覆盖范围内的周围单元[[1]](https://zh.wikipedia.org/zh-my/%E5%8D%B7%E7%A7%AF%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C" \l "cite_note-deeplearning-1)，对于大型图像处理有出色表现。

系统将采用两种卷积神经网络模型，VGG16网络与RepVGG网络，下面将依次进行介绍。

2.2.1 VGG16

2014年，牛津大学计算机视觉组（Visual Geometry Group）和Google DeepMind公司的研究员一起研发出了新的深度卷积神经网络：VGGNet，并取得了ILSVRC2014比赛分类项目的第二名（第一名是GoogLeNet，也是同年提出的）和定位项目的第一名。  
VGGNet探索了卷积神经网络的深度与其性能之间的关系，成功地构筑了16层深的卷积神经网络VGG16，证明了增加网络的深度能够在一定程度上影响网络最终的性能，使错误率大幅下降，同时拓展性又很强，迁移到其它图片数据上的泛化性也非常好。到目前为止，VGG仍然被用来提取图像特征。  
VGG16可以看成是加深版本的AlexNet，都是由卷积层、全连接层两大部分构成。

2.2.1.1 VGG16的特点

（1）小卷积核和多卷积子层

VGG16卷积核的尺寸设置为(3x3)是因为这是捕获左/右，上/下，中心概念的最小尺寸。

VGG16使用1x1的卷积核目的是在不影响输入输出的维度情况下，对输入进行形变，再通过ReLU进行非线性处理，提高决策函数的非线性。1x1的卷积神经网络还可以用来替代全连接层。

VGG16使用小卷积核(3x3)和多卷积子层代替一个卷积核较大的卷积层的好处一是可以减少参数，二是相当于进行了更多的非线性映射，可以增加网络的拟合/表达能力。

相比于AlexNet中较大的卷积核尺寸（如11x11, 7x7），VGG通过降低卷积核的大小（3x3），增加卷积子层数来达到同样的性能（VGG：从1到4卷积子层，AlexNet：1子层）（图2-1）。

VGG的两个3x3的卷积堆叠获得的感受野大小，相当一个5x5的卷积；而3个3x3卷积的堆叠获取到的感受野相当于一个7x7的卷积。这样可以增加非线性映射，也能很好地减少参数（例如7x7的参数为49个，而3个3x3的参数为27）

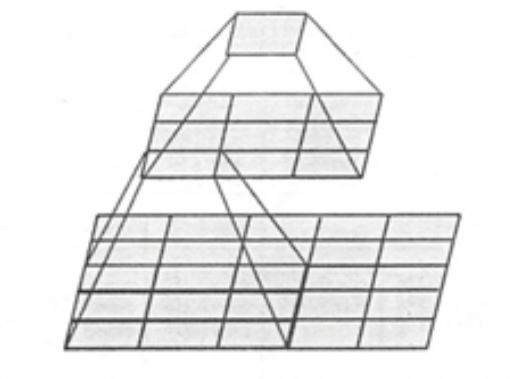


图2-1 两个串联的3x3的卷积层功能类似一个5x5的卷积层

（2）小池化核  
相比AlexNet的3x3的池化核，VGG全部采用2x2的池化核。

（3）通道数多  
VGG网络第一层的通道数为64，后面每层都进行了翻倍，最多到512个通道，通道数的增加，使得更多的信息可以被提取出来。

（4）层数更深、特征图更宽  
由于卷积核专注于扩大通道数、池化专注于缩小宽和高，使得模型架构上更深更宽的同时，控制了计算量的增加规模。

（5）全连接转卷积（测试阶段）  
在网络测试阶段将训练阶段的三个全连接替换为三个卷积，使得测试得到的全卷积网络因为没有全连接的限制，因而可以接收任意宽或高为的输入，这在测试阶段很重要。  
例如7x7x512的层要跟4096个神经元的层做全连接，则替换为对7x7x512的层作通道数为4096、卷积核为1x1的卷积（图2-2）。

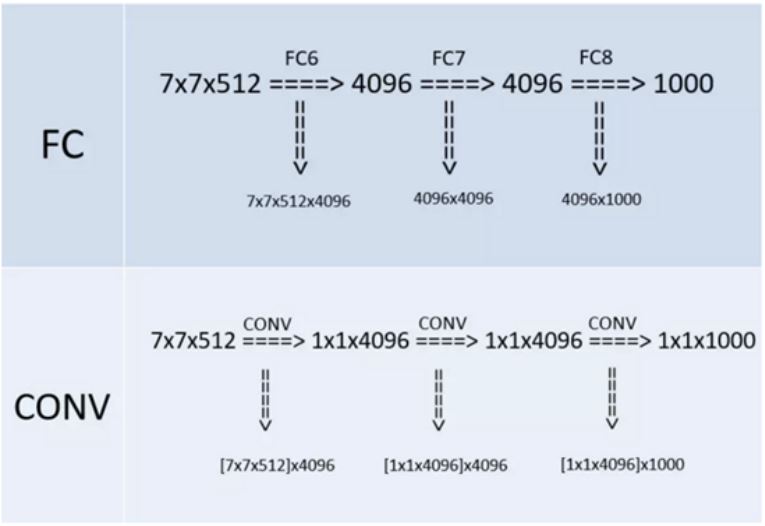


图2-2全连接层转卷积

（6）用简单模型初始化复杂模型  
VGGNet在训练的时候先训级别A的简单模型，再复用A网络的权重来初始化后面的几个复杂模型，这样收敛速度更快

（7）采用了Multi-Scale的方法来训练和预测  
可以增加训练的数据量，防止模型过拟合，提升预测准确率。

2.2.1.2 VGG16网络结构

VGG16包含了16个隐藏层（13个卷积层和3个全连接层，被分为5个block，卷积核都为3x3大小，随模型变深每一个卷积层的卷积核个数都翻倍（图2-3）。

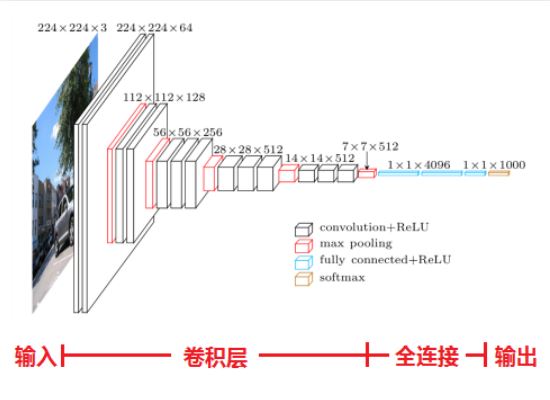


图2-3VGG网络结构图

VGG16处理过程如下：  
 1、输入224x224x3的图片，经64个3x3的卷积核作两次卷积+ReLU，卷积后的尺寸变为224x224x64  
 2、作max pooling（最大化池化），池化单元尺寸为2x2（效果为图像尺寸减半），池化后的尺寸变为112x112x64  
 3、经128个3x3的卷积核作两次卷积+ReLU，尺寸变为112x112x128  
 4、作2x2的max pooling池化，尺寸变为56x56x128  
 5、经256个3x3的卷积核作三次卷积+ReLU，尺寸变为56x56x256  
 6、作2x2的max pooling池化，尺寸变为28x28x256  
 7、经512个3x3的卷积核作三次卷积+ReLU，尺寸变为28x28x512  
 8、作2x2的max pooling池化，尺寸变为14x14x512  
 9、经512个3x3的卷积核作三次卷积+ReLU，尺寸变为14x14x512  
 10、作2x2的max pooling池化，尺寸变为7x7x512  
 11、与两层1x1x4096，一层1x1x1000进行全连接+ReLU（共三层）  
 12、通过softmax输出1000个预测结果

2.2.2 RepVgg

RepVGG是一个分类网路，该网络是在VGG网络的基础上进行改进，主要的改进点包括：（1）在VGG网络的Block块中加入了Identity和残差分支；（2）模型推理阶段，通过Op融合策略将所有的网络层都转换为Conv3\*3，便于模型的部署与加速。

## 2.2.2.1 RepVGG网络架构

图2-4展示了部分RepVGG网络，图A表示的是原始的ResNet网络，该网络中包含着Conv1\*1的残差结构和Identity的残差结构，正是这些残差结构的存在解决了深层网路中的梯度消失问题，使得网络更加易于收敛。图B表示的是训练阶段的RepVGG网络架构，整个网络的主体结构和ResNet网络类似，两个网络中都包含残差结构。两个网络中的主要差异如下所述：

1. RepVGG网络中的残差块并没有跨层，如图中的绿框所示；
2. 整个网络包含2种残差结构，如图中的绿框和红框所示，绿框中的残差结构仅仅包含Conv1\*1残差分支；红框中不仅包含Conv1\*1的残差结构，而且包含Identity残差结构。

（3）模型的初始阶段使用了简单的残差结构，随着模型的加深，使用了复杂的残差结构，这样不仅仅能够在网络的深层获得更健壮的特征表示，而且可以更好的处理网络深层的梯度消失问题。图C表示的是推理阶段的RepVGG网络，该网络的结构非常简单，整个网络均是由Conv3\*3+Relu堆叠而成，易于模型的推理和加速。

  这种架构的主要优势包括：

1. 当前大多数推理引擎都对Conv3x3做了特定的加速，假设整个网络中的每一个Conv3x3都能节省3ms，如果一个网络中包含30个卷积层，那么整个网络就可以节省3\*30=90ms的时间，这还是初略的估算。

（2）对于残差节点而言，需要当所有的残差分支都计算出对应的结果之后，才能获得最终的结果，这些残差分支的中间结果都会保存在设备的内存中，会对推理设备的内存具有较大的要求，来回的内存操作会降低整个网络的推理速度。而推理阶段首先在线下将模型转换为单分支结构，在设备推理阶段就能更好的提升设备的内存利用率，从而提升模型的推理速度（图2-5）。总而言之，模型推理阶段的网络结构越简单越能起到模型加速的效果。

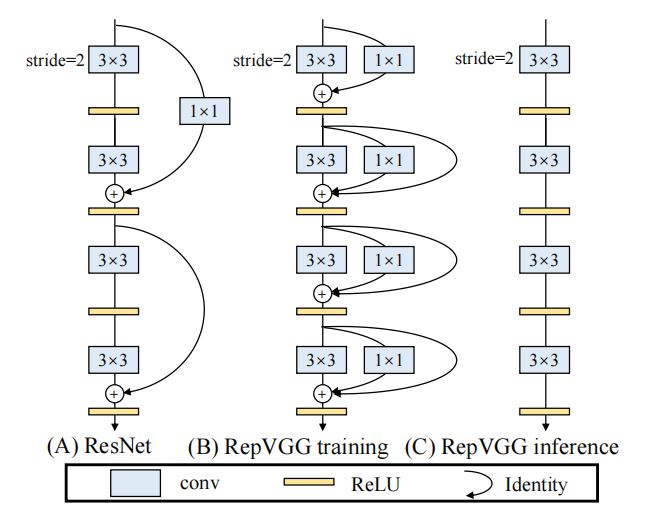
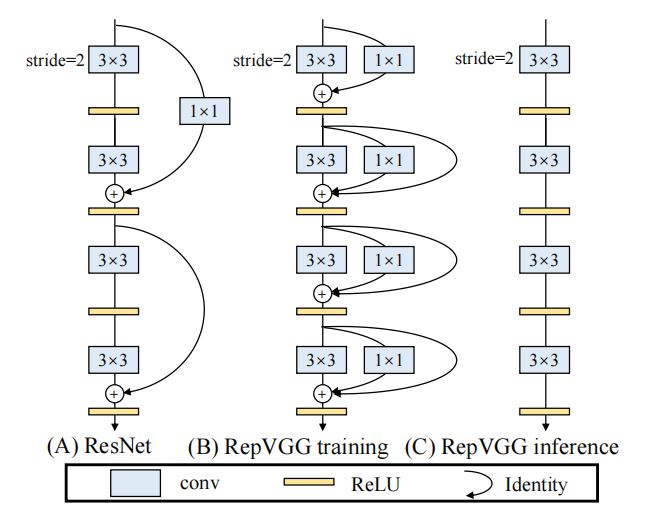


图2-4 RepVGG架构的草图，RepVGG有

5 个阶段，并在阶段开始时通过步长为2的卷积进行下采样。这里我们只显示第一个

特定阶段的 4 层。



（图2-5）Peak memory occupation in residual and plain

model. If the residual block maintains the size of feature

map, the peak value of extra memory occupied by feature

maps will be 2*×* as the input. The memory occupied by the

parameters is small compared to the features hence ignored.（爷尼玛翻译不过来了）

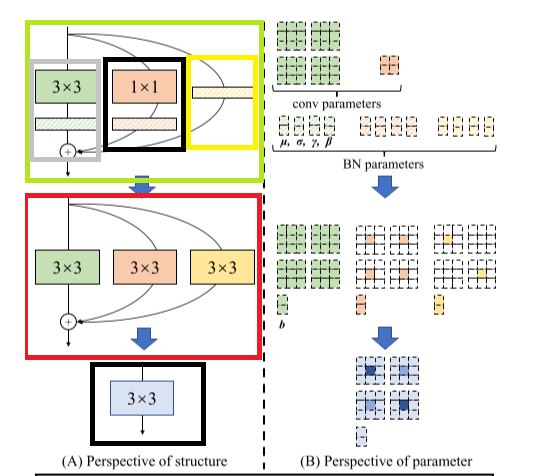
## 2.2.2.2 推理阶段Op融合细节详解

图2-6展现了模型推理阶段的重参数化过程，其实就是一个OP融合和OP替换的过程。图A从结构化的角度展示了整个重参数化流程， 图B从模型参数的角度展示了整个重参数化流程。整个重参数化步骤如下所示：

步骤1-首先通过式3将残差块中的卷积层和BN层进行融合。图2-6中的灰色框中执行Conv3x3和BN层的融合，图中的黑色矩形框中执行Conv1x1和BN层的融合，图中的黄色矩形框中执行Conv3x3(卷积核设置为全1)和BN层的融合。其中表示转换前的卷积层参数，表示BN层的均值，表示BN层的方差，和分别表示BN层的尺度因子和偏移因子，W’和b’分别表示融合之后的卷积的权重和偏置。

步骤2-将融合后的卷积层转换为Conv3\*3，即将具体不同卷积核的卷积均转换为具有3\*3大小的卷积核的卷积。由于整个残差块中可能包含Conv1\*1分支和Identity两种分支，如图中的黑框和黄框所示。对于Conv1\*1分支而言，整个转换过程就是利用3\*3的卷积核替换1\*1的卷积核，具体的细节如图中的紫框所示，即将1\*1卷积核中的数值移动到3\*3卷积核的中心点即可；对于Identity分支而言，该分支并没有改变输入的特征映射的数值，那么我们可以设置一个3\*3的卷积核，将所有的9个位置处的权重值都设置为1，那么它与输入的特征映射相乘之后，保持了原来的数值，具体的细节如图2-6中的褐色框所示。

步骤3-合并残差分支中的Conv3\*3。即将所有分支的权重W和偏置B叠加起来，从而获得一个融合之后的Conv3\*3网络层。



捕获

图2-6

## 2.2.2.3 RepVGG算法实现步骤

步骤1-获取并划分训练数据集，并对训练集执行数据增强操作；  
步骤2-搭建RepVGG训练网络，训练分类网络，直到网络收敛为止；  
步骤3-加载训练好的网络，对该网络执行重参数化操作;  
步骤4-加载重参数化后的模型，执行模型推理。

2.3损失函数

在[最优化](https://zh.wikipedia.org/wiki/%E6%9C%80%E4%BC%98%E5%8C%96" \o "最优化)，[统计学](https://zh.wikipedia.org/wiki/%E7%BB%9F%E8%AE%A1%E5%AD%A6" \o "统计学)，[决策论](https://zh.wikipedia.org/wiki/%E5%86%B3%E7%AD%96%E8%AE%BA" \o "决策论)，[机器学习](https://zh.wikipedia.org/wiki/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0" \o "机器学习)和[计算神经科学](https://zh.wikipedia.org/wiki/%E8%A8%88%E7%AE%97%E7%A5%9E%E7%B6%93%E7%A7%91%E5%AD%B8" \o "计算神经科学)的领域中，损失函数或成本函数是指一种将一个事件（在一个样本空间中的一个元素）映射到一个表达与其事件相关的经济成本或机会成本的[实数](https://zh.wikipedia.org/wiki/%E5%AE%9E%E6%95%B0" \o "实数)上的一种函数，借此直观表示的一些"成本"与事件的关联。在对系统模型训练中采用了交叉熵损失函数与负对数似然损失函数，下面将对这两种损失函数进行介绍。

2.3.1 交叉熵损失函数

交叉熵损失函数是多分类问题中很常用的损失函数，其表达式为：

其中：

M——类别的数量

——符号函数（0或1），如果样本 i的真实类别等于 c 取 1 ，否则取 0

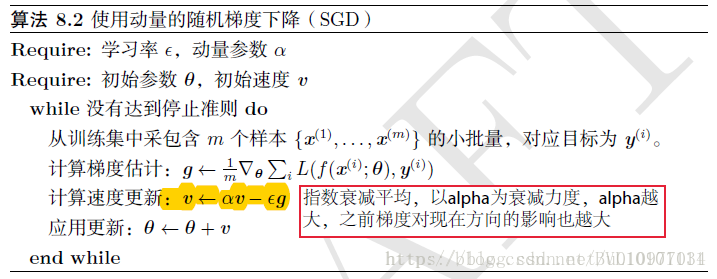
——观测样本 i属于类别 c的预测概率

2.3.2 负似然对数

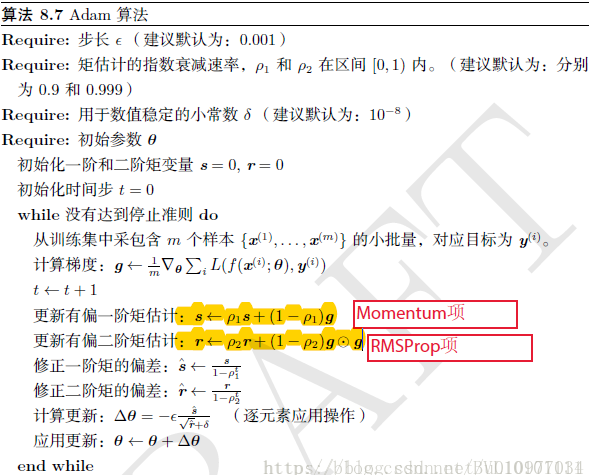
负对数似然是一种用于解决分类问题的  损失函数 ，它是似然函数得一种自然对数形式，可用于测量两种概率分布之间的相似性，其取负号是为了让最大似然值和最小损失相对应，是最大似然估计及相关领域的常见函数形式，其公式如下：

2.4优化器

2.4.1 Momentum优化器



2.4.1 Adam优化器



2.5本章小结

本章主要介绍了本系统开发过程中涉及到的理论基础和技术基础，包括系统采用的VGG16和RepVGG卷积神经网络和训练模型中采用的Adam，Momentum优化器，和交叉熵，负对数似然损失函数，以及用于搭建模型的paddlepaddle框架和用于部署线上api的AIStudio。

3 租赁业务风险控制安全中台需求分析

本章主要进行租赁业务风险控制安全的需求分析，从用户角度梳理基本需求，并按照功能模块对产品进行划分，针对每一个模块，通过用例图和用例描述的方式对该模块具体功能进行阐述。同时从产品使用和用户需求角度，对系统非功能性需求进行梳理。

3.1 需求分析综述

产品的诞生是为了解决用户需求，因此若想要设计出一款好的产品，需要了解用户是谁，用户有什么样的需求，才可以设计出正确、易用的产品。

租赁业务风险控制安全中台的目标是抵抗黑产攻击以及防止内部人员泄露敏感数据，系统使用者主要分为风控策略师、业务经理、安全部门负责人三类，其需求如下：

（1）风控策略师：需要系统提供规则配置功能。由于公司业务复杂、风险点多，且黑产的攻击手段变化快、无法预知，因此要求规则配置功能需方便修改、易复用、易迁移。并且系统需提供规则评价功能，以便于了解规则的准确性和有效性，对规则进行调整迭代。由于手动调整过于繁琐，需要系统提供简单的规则自动调整功能。

（2）业务经理：需要借助系统了解所在业务线的风险情况，要求风险情况展现较为直观，易于查看。除此之外，需要系统提供高危用户列表，以便于进行进一步的防控。最后，为了减少系统误判的可能性，系统内部需设立黑白名单机制。

（3）安全部门负责人：需要借助系统了解公司整体风险情况，且展现方式较为直观。需要了解高危用户列表，包括公司内部员工和外部用户。需要审查规则的合理性和有效性，当规则不再适用时督促风控策略师进行调整。

3.2 功能性需求分析

租赁业务风险控制安全中台是搭建在用户和公司各个子系统之间的的风险控制平台，以租赁业务和其余衍生业务为前提，接收用户操作，根据中台内部的规则体系对用户操作进行分析，向子系统提供防控建议。主要功能包括查看平台整体风险状况、检索用户在平台内具体操作、规则、规则组、决策树配置、监控规则执行情况、对规则进行评价、配置敏感数据告警规则等。

按照功能划分，将该系统分为四个主要功能，分别为敏感数据审计、风险情况分析、规则引擎构建、规则执行监控。下面将详述四个功能的具体功能性需求。

3.2.1 敏感数据审计功能

该功能主要实现对公司内部员工查看敏感数据的监控。该部分需与公司其余内部系统配合使用，按照数据分类分级原则，对公司内部系统所展示的所有数据进行分级，并将员工点击查看敏感数据的操作日志传入风险控制安全中台。风控策略师可在系统中针对员工角色配置一段时间内点击查看敏感次数上限阈值。系统使用定时任务，定期根据数据点击日志检查是否超过阈值，若超过则触发告警，向员工经理及安全部份发送告警信息。业务经理及安全部门负责人可在系统中查看具体告警内容及敏感数据点击日志。该功能用例图如图3-1所示。

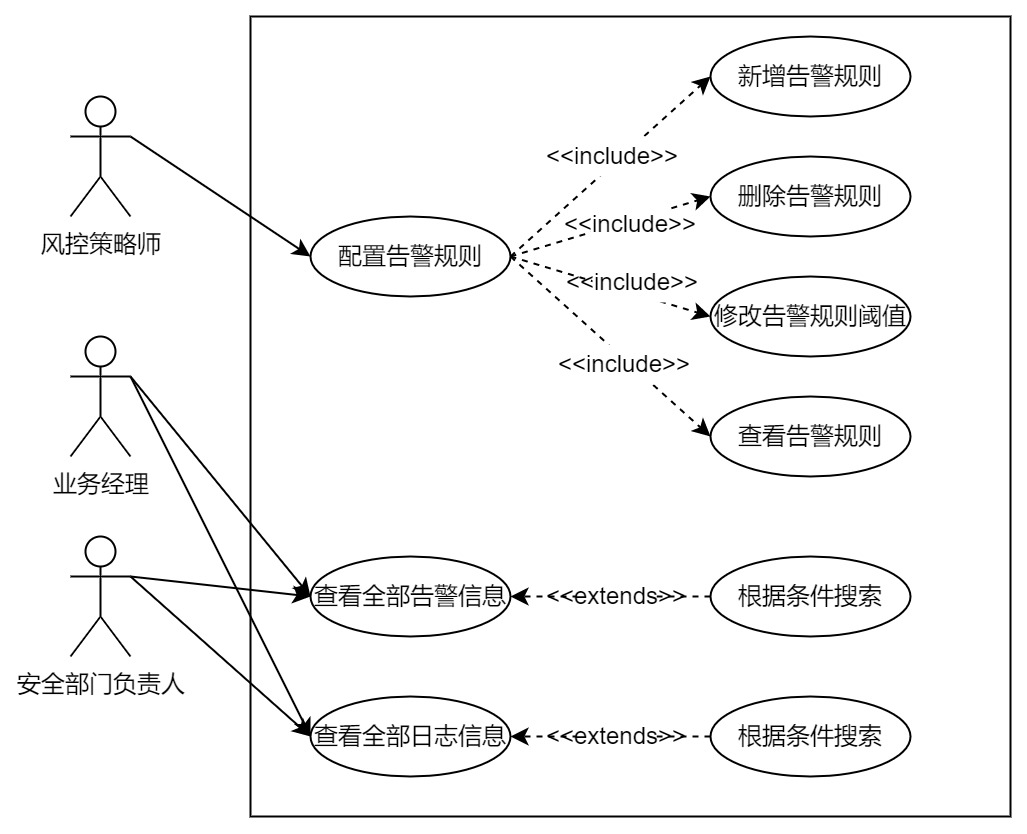


图3-1 敏感数据审计用例图

下面对于该部分中的三个主要用例配置告警规则、查看全部告警信息、查看全部日志信息进行详细描述。

表3-1 配置告警规则用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC1 | **用例名称** | 配置告警规则 |
| **活动者** | 风控策略师 | **优先级** | 高 |
| **描述** | 本用例描述风控策略师在配置告警规则时的一系列操作，包括对告警规则的增删改查 | | |
| **前置条件** | 风控策略师进入系统 | | |
| **基本流程** | 1. 风控策略师点击导航栏“告警-告警规则” 2. 界面上显示目前所有告警规则，包括职位名称、告警时间范围、一级告警阈值、二级告警阈值 | | |

续表3-1 配置告警规则用例说明

|  |  |
| --- | --- |
| **基本流程** | 1. 风控策略师点击下方“添加”按钮 2. 界面新增一行空规则，包含4个待输入选项：职位名称、时间范围、一级阈值、二级阈值 3. 风控策略师从职位名称下拉框中进行选择，并输入时间范围、一级和二级阈值 4. 风控策略师点击保存 5. 系统将该规则加入规则库中，提示保存成功 6. 风控策略师点击某一规则右侧“删除”按钮 7. 系统将该规则从规则库中删除，提示保存成功 8. 风控策略师修改任意规则时间范围、一级和二级阈值 9. 风控策略师点击保存按钮 10. 系统更新规则库，提示保存成功 |
| **扩展流程** | 7a. 如果风控策略师输入的时间范围不为正一位小数，或输入阈值不为正整数，或一级阈值小于二级阈值，提示输入错误  7b 如果风控策略师未点击“保存”按钮，直接点击其余位置，则系统不更新规则库，界面返回至2中界面 |
| **异常流程** | 无 |
| **后置条件** | 无 |
| **补充说明** | 无 |

表3-2 查看全部告警信息用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC2 | **用例名称** | 查看全部告警信息 |
| **活动者** | 业务经理、安全部门负责人 | **优先级** | 高 |
| **描述** | 本用例描述业务经理和安全部门负责人查看告警信息的一系列操作，包括根据条件进行筛选 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“告警-告警记录” 2. 界面上显示目前所有告警记录，包括时间、触发告警员工姓名、工号、所属部门、触发告警级别、规则时间内敏感数据点击次数 3. 用户在界面上方筛选框内，根据时间、员工姓名、工号、所属部门、触发告警级别进行筛选，点击“搜索” 4. 系统根据条件对所有告警记录进行过滤，界面显示所有满足条件的告警记录 5. 用户点击某一记录右侧“查看详情”按钮 6. 系统跳转至敏感数据点击日志查看界面，显示该员工全部点击日志 | | |

续表3-2 查看全部告警信息用例说明

|  |  |
| --- | --- |
| **扩展流程** | 4a 若没有满足筛选条件的告警记录，系统提示“无满足条件记录” |
| **异常流程** | 无 |
| **后置条件** | 无 |
| **补充说明** | 无 |

表3-3 查看全部日志信息用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC3 | **用例名称** | 查看全部日志信息 |
| **活动者** | 业务经理、安全部门负责人 | **优先级** | 中 |
| **描述** | 本用例描述业务经理和安全部门负责人查看日志信息的一系列操作，包括根据条件进行筛选 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“告警-日志详情” 2. 界面上显示目前所有日志信息，按照时间由近到远排序，显示内容包括时间、员工姓名、工号、所属部门、点击系统、字段名称、字段描述 3. 用户在界面上方筛选框内，根据时间、员工姓名、工号、所属部门、点击系统进行筛选，点击“搜索” 4. 系统根据条件对所有告警记录进行过滤，界面显示所有满足条件的告警记录 | | |
| **扩展流程** | 4a 若没有满足筛选条件的日志信息，系统提示“无满足条件记录” | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

3.2.2 风险情况分析功能

该功能主要实现对平台外部整体风险情况分析展示。在该功能中，实现对所有用户各业务线风险分数聚合及展示，以及根据用户风险分数评估平台风险程度。同时提供用户操作查看功能，可根据用户uid、手机号、设备号搜索该用户全部操作。提供黑白名单库，在该模块中可对黑白名单库进行手动维护。该模块用例图如图3-2所示。

下面对于该功能中的三个主要用例平台整体风险情况查看、用户具体操作查看、黑白名单库管理进行详细描述。

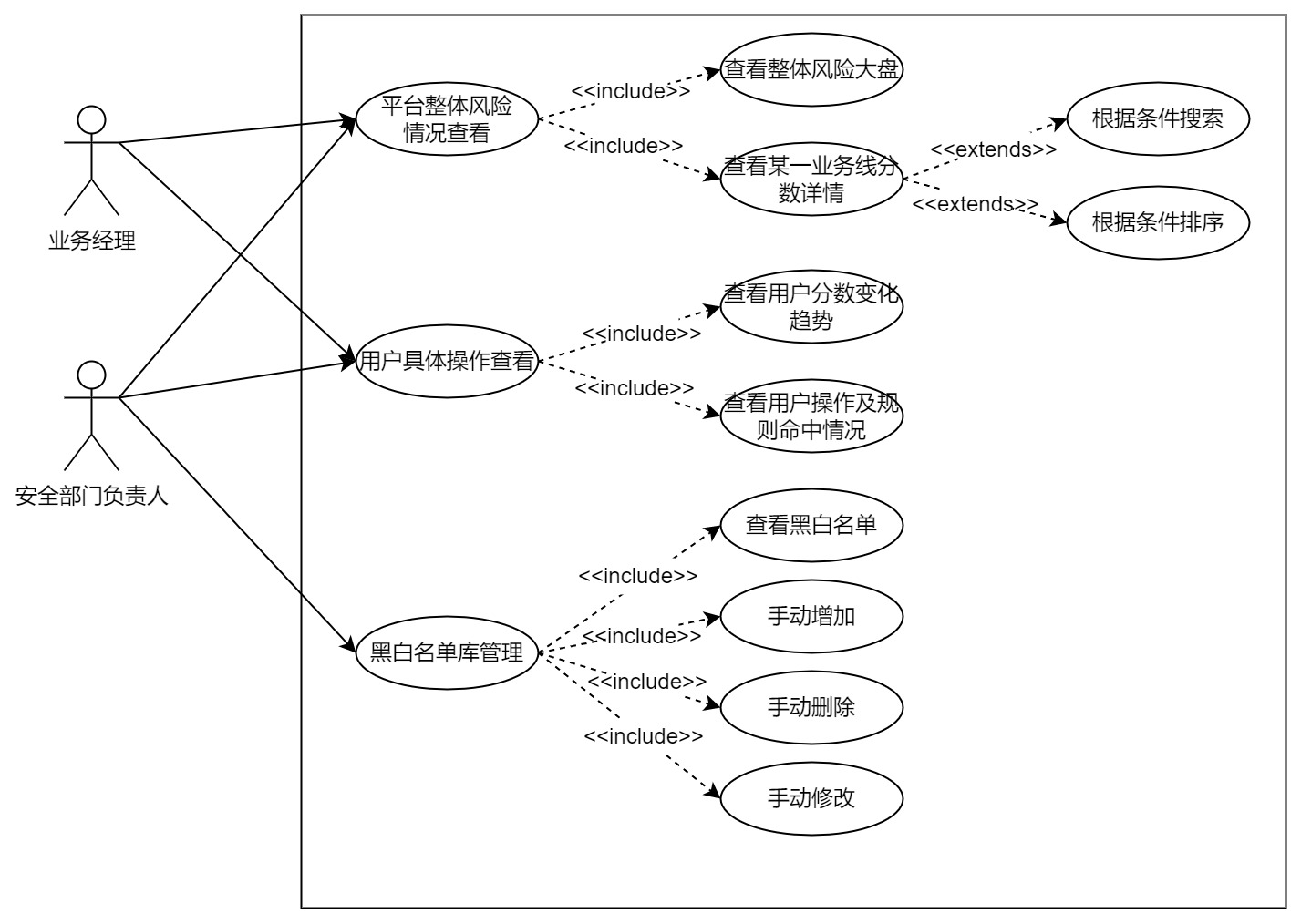


图3-2 风险情况分析模块用例图

表3-4 平台整体风险情况查看用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC4 | **用例名称** | 平台整体风险情况查看 |
| **活动者** | 业务经理、安全部门负责人 | **优先级** | 高 |
| **描述** | 本用例描述业务经理和安全部门负责人查看平台整体风险情况的具体操作，包括查看整体风险大盘和查看某一业务线风险详情 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“风险总览” 2. 界面上利用饼状图显示平台内用户风险总分分布情况，以及各业务线用户风险分分布情况 3. 用户将鼠标移动至饼状图某个区域上，显示该分数该区域内用户总人数及占总用户人数百分比 4. 用户点击平台总体风险饼状图 5. 界面跳转至平台总体风险详情界面，界面上方使用折线图显示各分数段用户人数，界面下方显示全部用户分数详情，包括用户id、最后一次登陆时间、风险总分、各业务线风险分、用户标签、用户风险级别 6. 用户按照风险总分、各业务线风险分、风险级别进行筛选排序 7. 界面按照筛选条件显示对应内容 8. 点击用户id字段 | | |

续表3-4 平台整体风险情况查看用例说明

|  |  |
| --- | --- |
| **基本流程** | 1. 界面跳转至用户操作详情界面 2. 用户返回风险总览界面，点击任意业务线风险饼状图，或整体风险展示饼状图 3. 界面跳转至该业务线风险详情界面，界面上方使用折线图显示各分数段用户人数，界面下方显示全部用户分数详情，包括用户id、最后一次登陆时间、风险总分、该业务线风险分、用户标签、用户风险级别 |
| **扩展流程** | 无 |
| **异常流程** | 无 |
| **后置条件** | 无 |
| **补充说明** | 用户各业务线风险分数为实时结果，用户总风险分数根据各业务线风险分数计算得出，该分数每1小时更新一次。 |

表3-5 用户具体操作查看用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC5 | **用例名称** | 用户具体操作查看 |
| **活动者** | 业务经理、安全部门负责人 | **优先级** | 高 |
| **描述** | 本用例描述业务经理和安全部门负责人查看某一用户具体操作的一系列操作，包括查看用户分数变化趋势和查看用户具体操作及规则命中情况 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“用户操作详情” 2. 跳转至操作详情查看界面，顶端显示用户id和手机号输入框以及搜索按钮 3. 用户输入至少一个搜索条件，点击搜索 4. 界面上显示满足条件的用户详情，上方显示近一周用户各项风险分数变化情况，下方显示用户所有操作，包括时间、触发事件名称、操作所属业务线、操作触发规则名称、操作前后分数变化详情 | | |
| **扩展流程** | 3a 若用户没有输入任何条件，直接点击搜索，系统提示请输入搜索条件  3b 若经查询，系统中没有同时满足输入用户id和手机号条件的用户，系统提示“无该用户” | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

表3-6 黑白名单库管理用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC6 | **用例名称** | 黑白名单库管理 |
| **活动者** | 安全部门负责人 | **优先级** | 高 |
| **描述** | 本用例描述安全部门负责人管理黑白名单一系列操作，包括根据条件进行筛选 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“黑白名单库-黑名单” 2. 界面上显示黑名单库中四张表的全部内容，四张表分别为uid表、IP表、手机号表、设备号表，内容包括账号、加入时间、加入原因、最后修改人、修改时间 3. 用户点击“添加”按钮，在弹窗中选择要添加的表名，填写具体内容和添加理由，点击“确定” 4. 系统更新黑名单库，并刷新界面 5. 用户点击某一黑名单右侧“修改”按钮，可对加入原因进行修改 6. 系统更新黑名单库，同时修改最后修改人、修改时间，并刷新界面 7. 用户点击某一黑名单右侧“删除”按钮 8. 系统更新黑名单库，并刷新界面   用户点击导航栏“黑白名单库-白名单”，重复流程2-8 | | |
| **扩展流程** | 4a 若用户添加黑名单中的uid表，但输入的uid不存在于用户库中，系统提示该用户不存在 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

3.2.3 规则引擎构建功能

该功能主要实现规则引擎构建及调整。风控策略师建立特征库，对不同特征接收参数进行解析，定义特征计算规则；建立防控措施库，定义不同措施处置规则；构建由事件、业务线、规则、规则组组成的决策流，并实现规则、规则组执行逻辑、参数配置调整功能。

该模块用例图如图3-3所示，包括两个主要用例特征库建立，决策树建立，两个主要用例中又包含5个小用例。

下面对于该功能中的两个主要用例进行详细描述。

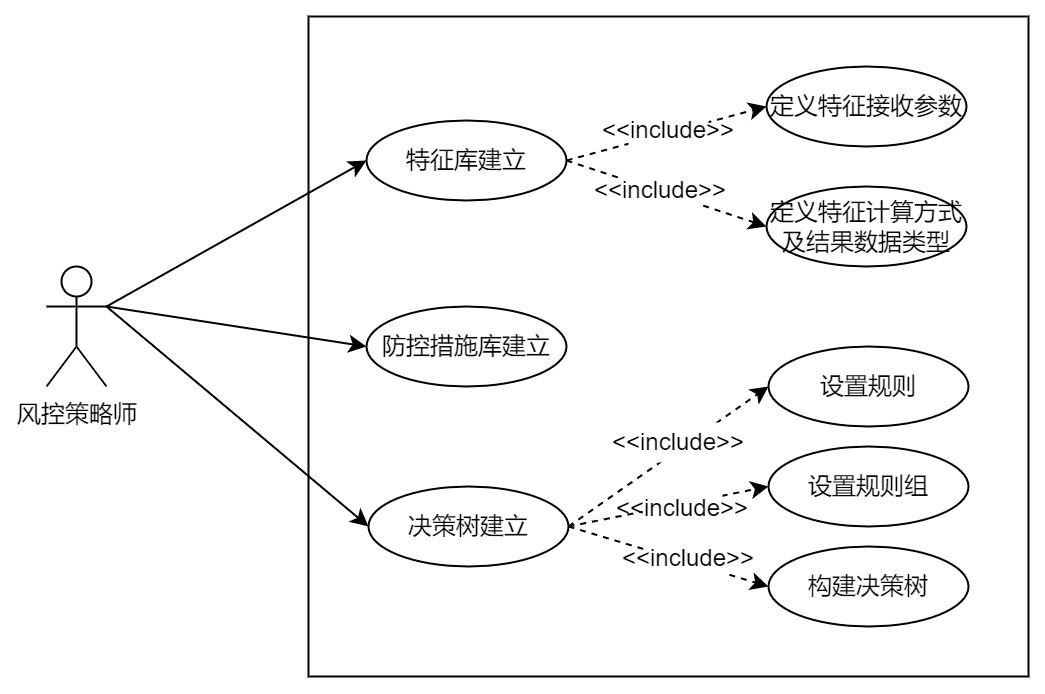


图3-3 风险情况分析模块用例图

表3-7 特征库建立用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC7 | **用例名称** | 特征库建立 |
| **活动者** | 风控策略师 | **优先级** | 高 |
| **描述** | 本用例描述风控策略师建立特征库的具体操作，包括定义特征接收参数、计算方式及结果数据类型 | | |
| **前置条件** | 无 | | |
| **基本流程** | 1. 用户在系统目录下，建立java文件featureLibrary，作为特征定义文件，后续若增加特征，直接在该文件中添加即可。 2. 定义特征函数，设定接收参数 3. 撰写特征计算方法，其中需要数据清洗的操作可使用Flink定义job完成，将结果存放至redis中。 4. 定义特征计算结果数据类型 5. 将特征名称、特征函数、结果数据类型存入数据库中，待建立规则时使用。 | | |
| **扩展流程** | 5a 更新数据库时，检查特征名称是否重复，若重复需修改featureLibrary中的特征名称。 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

表3-8 决策树建立用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC8 | **用例名称** | 决策树建立 |
| **活动者** | 风控策略师 | **优先级** | 高 |
| **描述** | 本用例描述风控策略师建立决策树的一系列操作，包括规则、规则组设置及决策流构建 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“规则引擎-规则建立” 2. 跳转至规则建立界面，显示所有现有规则，包括规则名称、特征、操作符、阈值、惩罚措施 3. 点击“新增规则”，在弹窗中设置规则名称、特征、操作符、阈值、惩罚措施，点击“确认” 4. 系统更新规则库，并刷新前端界面 5. 点击对应规则后的“编辑”按钮，可编辑规则名称、特征、操作符、阈值、惩罚措施，点击“保存” 6. 系统更新规则库，并刷新前端界面 7. 用户点击导航栏“规则引擎-规则组建立” 8. 跳转至规则组建立界面，显示所有现有规则组，包括规则组名称、包含规则列表、规则组命中逻辑、惩罚措施 9. 点击“新增规则组”，在弹窗中设置规则组名称、选择组内规则，填写规则组命中逻辑及惩罚措施，点击“确认” 10. 系统更新规则库，并刷新前端界面 11. 用户点击导航栏“规则引擎-决策树建立” 12. 跳转至决策树界面，根据不同事件显示决策树，包括规则组名称、规则组组合关系 13. 点击“新增规则组”，在弹窗中设置决策树对应事件，定义规则组排列方式，点击“确认” 14. 系统更新规则库，并刷新前端界面 | | |
| **扩展流程** | 4a 新增或修改规则时，系统根据操作符和特征计算结果数据类型检查阈值设置是否合法，若非法则提示阈值非法  5a 若修改规则后未点击保存，直接点击页面其余地方，则不更新规则库，视为无改动 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

3.2.4 规则执行监控功能

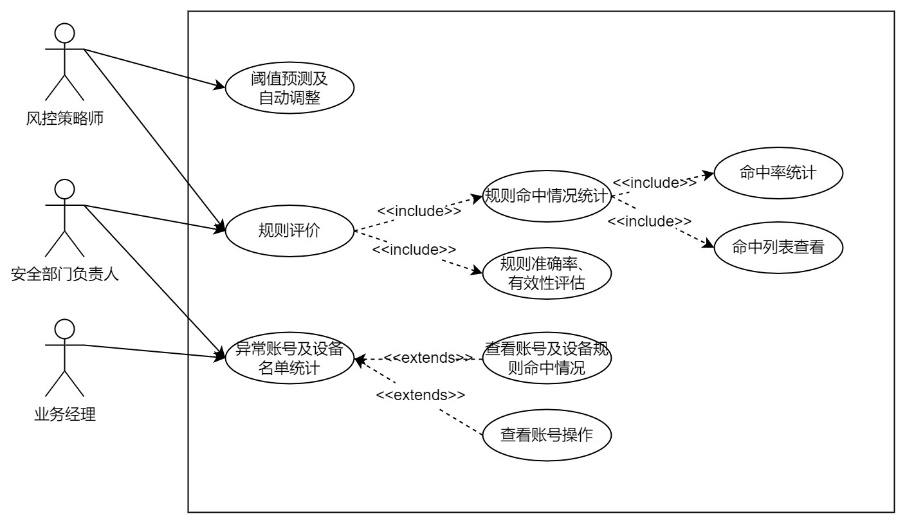
该功能主要实现规则执行监控，为安全部门负责人及风控策略师提供规则执行结果统计、评价规则准确性及合理性、规则阈值预测及自动调整功能，以及为安全部门负责人和业务经理提供异常账号及设备名单，以便进行后续操作。该功能用例图如图3-4所示。

图3-4 规则执行监控模块用例图

下面对于该功能中的三个主要用例特征库建立、决策树建立进行详细描述。

表3-9 规则评价用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC9 | **用例名称** | 规则评价 |
| **活动者** | 安全部门负责人、风控策略师 | **优先级** | 高 |
| **描述** | 本用例描述风控策略师和安全部门负责人对规则进行评价的具体操作，包括查看规则命中情况，以及对规则准确率和有效性进行评估。 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“规则评价-规则命中情况统计” 2. 系统跳转至规则命中情况统计界面，系统左侧菜单显示全部事件 3. 用户从菜单中选择事件 4. 系统界面上利用饼状图显示该事件中所有规则组的命中情况，每个饼状图中显示该规则组内所有规则的命中次数和命中率，以及规则组整体命中率 5. 用户点击界面上的任一规则组或任意规则 6. 系统界面上显示该规则组或该规则的全部命中列表 7. 用户点击导航栏“规则评价-有效性评估” 8. 系统跳转至规则有效性评估界面，系统左侧菜单显示全部事件 9. 用户从菜单中选择事件 | | |

续表3-9 规则评价用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC9 | **用例名称** | 规则评价 |
| **活动者** | 安全部门负责人、风控策略师 | **优先级** | 高 |
| **描述** | 本用例描述风控策略师和安全部门负责人对规则进行评价的具体操作，包括查看规则命中情况，以及对规则准确率和有效性进行评估。 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“规则评价-规则命中情况统计” 2. 系统跳转至规则命中情况统计界面，系统左侧菜单显示全部事件 3. 用户从菜单中选择事件 4. 界面上利用饼状图显示该事件中所有规则组的命中情况，每个饼状图中显示该规则组内所有规则的命中次数和命中率，以及规则组整体命中率 5. 用户点击界面上的任一规则组或任意规则 6. 系统界面上显示该规则组或该规则的全部命中列表 7. 用户点击导航栏“规则评价-有效性评估” 8. 系统跳转至规则有效性评估界面，系统左侧菜单显示全部事件 9. 用户从菜单中选择事件 10. 界面上利用饼状图显示该事件中所有规则及规则组的准确率和有效性 | | |
| **扩展流程** | 无 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 规则及规则组的准确率和有效性每5分钟计算一次 | | |

表3-10 阈值预测用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC10 | **用例名称** | 阈值预测 |
| **活动者** | 风控策略师 | **优先级** | 中 |
| **描述** | 本用例描述系统根据风控策略师前几次设定的阈值，对未来一段时间内阈值自动调整的一系列操作 | | |
| **前置条件** | 风控策略师为规则设置初始阈值，并且规则部署上线超过3天 | | |
| **基本流程** | 1. 系统根据所有历史阈值，对未来24小时内阈值进行预测 2. 系统将规则阈值自动调整为预测结果 3. 系统每5分钟计算所有规则及规则组的准确率和有效性 | | |
| **扩展流程** | 3a 若在新阈值使用过程，任意规则及规则组准确率或有效性低于80%，系统向风控策略师发送告警，提示修改阈值 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

表3-11 异常用户统计用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | UC11 | **用例名称** | 异常用户统计 |
| **活动者** | 业务经理、安全部门负责人 | **优先级** | 中 |
| **描述** | 本用例描述业务经理和安全部门负责人查看异常用户的一系列操作，包括查看账号及设备规则命中情况和查看账号操作 | | |
| **前置条件** | 用户进入系统 | | |
| **基本流程** | 1. 用户点击导航栏“规则评价-异常用户统计” 2. 系统跳转至异常用户统计界面，显示所有异常设备及异常账号列表，包括设备/账号名称、命中规则次数 3. 用户点击账号名称，跳转至用户操作统计界面 4. 用户点击规则命中次数，跳转至规则命中列表界面 | | |
| **扩展流程** | 无 | | |
| **异常流程** | 无 | | |
| **后置条件** | 无 | | |
| **补充说明** | 无 | | |

3.3 非功能性需求分析

租赁业务风险控制安全中台作为企业级风控系统，除上述功能性需求外，还需满足以下非功能性需求。

（1）易用性。系统界面需对用户友好，界面简洁，布局合理，交互操作符合用户习惯，交互体验感好。

（2）灵活性。风控系统涉及业务及事件多种多样，后续可能还会有新的业务线或者事件接入，因此在系统构建过程中需要做到高内聚、低耦合，使系统易扩展。

（3）正确性。由于风控决策会对用户造成或多或少的影响，有时甚至出现阻断性防控措施，因此需保证风控策略的正确性，不可对业务造成负面影响。

（4）高性能。安全中台是介于用户和业务系统中间的一环，若处理时间过长，会影响用户操作体验。因此需将数据处理时延降低至用户可接受范围内。

3.4 本章小结

本章首先对系统用户及用户根本需求进行分析，之后对系统各功能进行了详细的需求分析，包括敏感数据审计、风险情况分析、规则引擎构建、规则执行监控四大功能，最后对系统非功能性需求进行阐述。

4 租赁业务风险控制安全中台概要设计

本章以系统整体需求为基础，对系统的概要设计进行阐述。首先给出系统整体体系架构，然后按照功能模块，依次对每个模块的设计进行相关介绍，包括敏感数据审计、风险情况分析、规则引擎构建、规则执行监控，最后对系统的数据库设计进行阐述。

4.1 系统体系结构

租赁业务风险控制安全中台的总体架构图如图4-1所示。系统前端搭建使用了Vue框架，搭配了IView组件库，包含众多UI组件，达到界面美观且减少前端开发工作量的目的。使用Echarts组件库完成前端图表绘制，不仅可定制化实现各类可视化图表，且大大简化工作量。

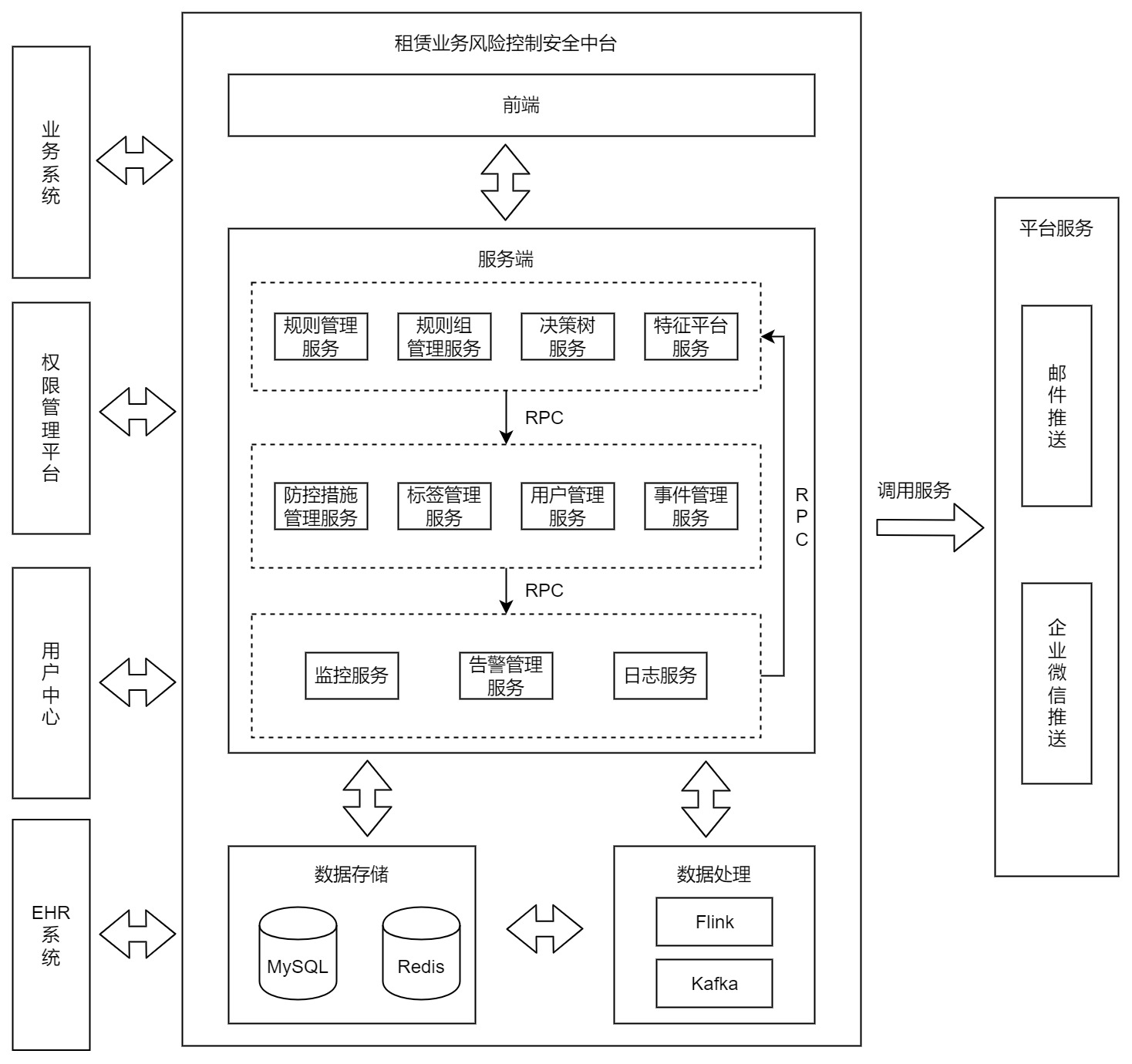


图4-1 系统架构图

系统后端采用java语言，使用SpringBoot框架简化开发工作量。系统采用微服务架构，将功能拆分为多个微服务，减少系统后端各模块之间的耦合度，易于修改、扩展和后期维护。各服务独立运行，使得系统部署更加便捷。不同服务之间的通信采用RPC。

系统存储的数据来源于三类：第一类为公司内部各系统敏感数据点击的埋点日志，通过Kafka发送至系统，经过数据清洗后存入MySQL数据库，等待敏感数据审计规则匹配；第二类为用户中心传来的用户登陆及注册数据，第三类为各业务系统传入的用户其余操作数据，这两类数据均通过Kafka发送至系统，持久化存储到MySQL数据库中，同时通过特征平台过滤，传入Flink进行数据清洗及特征聚合，将处理好的特征结果存入redis供规则计算使用。权限管理平台和EHR系统传来的数据接收即使用，不在系统内进行存储。

邮件通知和企业微信通知功能由公司基础服务平台提供，系统直接调用服务，不再进行实现。

4.2 系统功能结构

在第三章需求分析的过程中，本系统有4大模块11个用例。4大模块分别是敏感数据审计、风险情况分析、规则引擎构建、规则执行监控。系统总体功能模块结构图如图4-2所示。

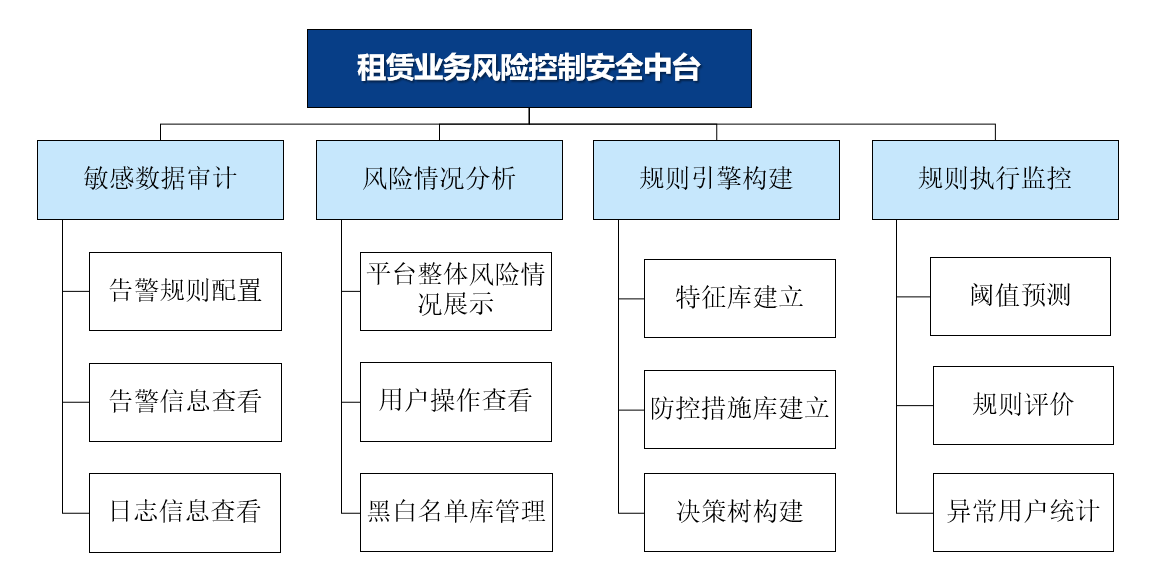


图4-2 租赁业务风险控制安全中台总体结构图

敏感数据审计模块主要是对员工点击内部系统上的敏感数据次数进行监控，每5分钟从MySQL数据库中按照规则限定的时间完成一次数据聚合，并与规则阈值进行比较。超出阈值调用HER系统数据，向对应管理层发送告警信息。

风险情况分析模块主要对平台外部用户风险进行分析展示，利用Flink框架处理用户各业务线风险分数，计算业务线加权参数得到用户在平台整体风险情况并进行展示，并提供黑白名单库管理即用户操作查看功能。

规则引擎构建模块主要实现风控规则构建及运行。风控策略师通过脚本定义特征，特征计算过程中利用Flink框架完成特征聚合，并将特征结果存入Redis。风控策略师利用定义好的特征构建规则、规则组及决策树。当有用户数据流入，该模块根据事件匹配决策树，并根据规则从Redis中取出对应特征进行计算，最后根据规则计算结果从防控措施库中匹配对应防控措施，完成对用户请求的风险判定和处理。

规则执行监控模块分为阈值预测、规则评价和异常用户统计。阈值预测中使用三次指数平滑法，利用以往规则阈值对未来24小时阈值进行预测，并自动调整。规则评价中，每5分钟根据规则命中情况，利用公式计算查准率和召回率，若低于80%发送告警。异常用户统计中，记录所有用户uid和设备号规则命中情况，并对异常用户进行统计。

4.3 系统数据库设计

本节主要介绍租赁业务风险控制安全中台的数据库设计，先给出系统的ER图，共包括16个实体和3个实体间关系，并针对每一张表给出详细设计，包括字段、类型、属性及字段描述。系统ER图如图4-3所示。

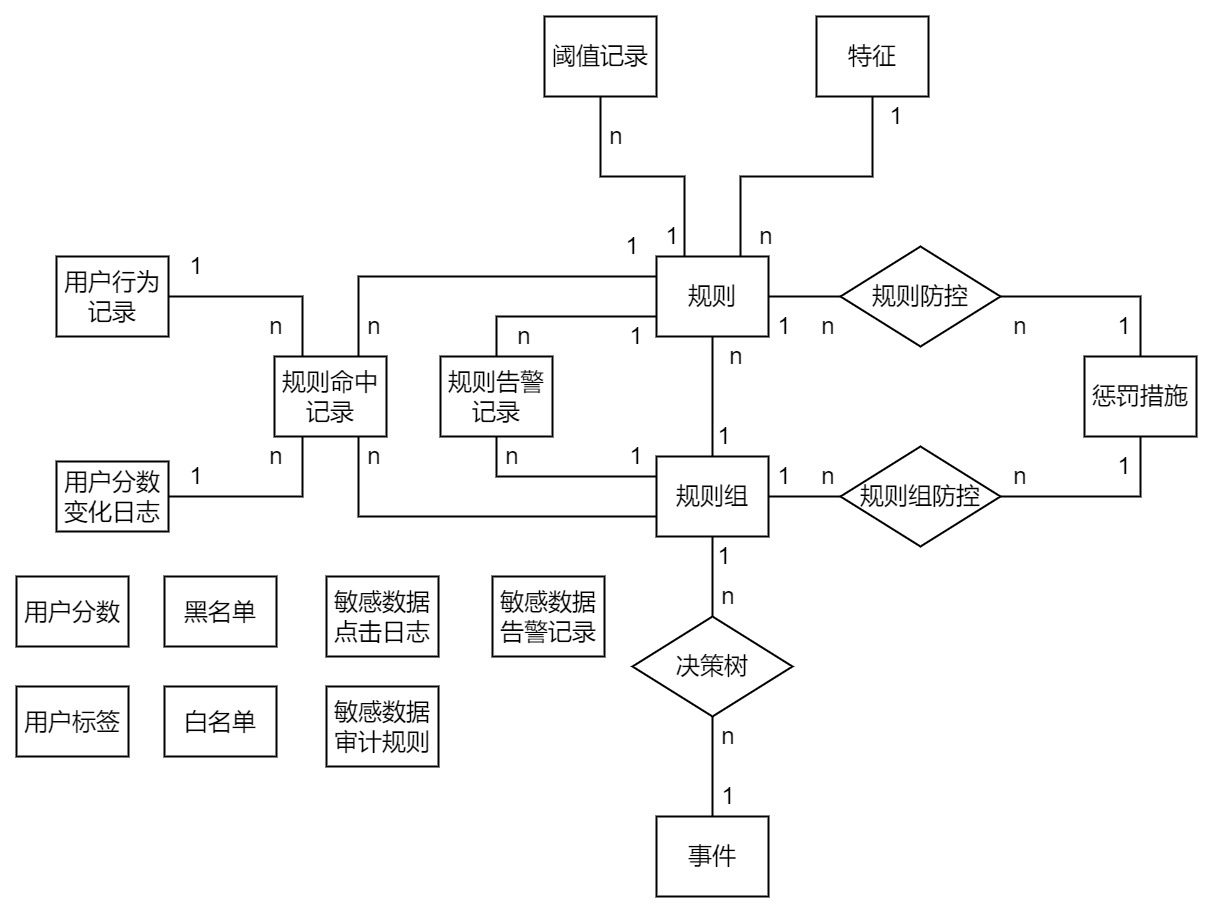


图4-3 系统ER图

表详细设计如表4-1至4-19所示。

表4-1 敏感数据点击日志表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：data\_click\_log** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | operation\_time | datetime | - | 点击时间 |
| 3 | operation\_username | varchar(50) | - | 操作人邮箱前缀 |
| 4 | operation\_uid | varchar(10) | - | 操作人工号 |
| 5 | operation\_name | varchar(10) | - | 操作人姓名 |
| 6 | operation\_field | varchar(255) | - | 操作字段名称 |
| 7 | system | varchar(50) | - | 操作系统名称 |
| 8 | operation\_status | varchar(10) | - | 操作人状态（在职/离职） |
| 9 | data\_origin | varchar(255) | - | 操作数据来源 |

表4-2 敏感数据审计规则表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：data\_click\_rules** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | role\_code | varchar(10) | - | 职位编码 |
| 3 | role\_name | varchar(50) | - | 职位名称 |
| 4 | role\_city | varchar(10) | - | 职位城市编码 |
| 5 | level1\_threshold | int | - | 一级阈值 |
| 6 | level2\_threshold | int | - | 二级阈值 |
| 7 | time\_window | int | - | 时间窗口 |
| 8 | create\_time | datetime | - | 创建时间 |
| 9 | create\_user | varchar(255) | - | 创建人 |

表4-3 敏感数据告警记录表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：data\_click\_alarm\_record** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | alarm\_level | varchar(10) | - | 告警级别 |
| 3 | operation\_username | varchar(50) | - | 操作人邮箱前缀 |
| 4 | operation\_uid | varchar(10) | - | 操作人工号 |
| 5 | operation\_name | varchar(10) | - | 操作人姓名 |
| 6 | operation\_time | int | - | 操作人查看次数 |
| 7 | leader\_uid | varchar(10) | - | 上级领导员工编号 |
| 8 | leader\_name | varchar(10) | - | 上级领导姓名 |
| 9 | alarm\_time | datetime | - | 告警时间 |

表4-4 特征表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：feture\_info** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | 特征id |
| 2 | feature\_name | varchar(50) | - | 特征名称 |
| 3 | feature\_format | varchar(10) | - | 特征计算结果格式 |
| 4 | feature\_field | varchar(255) | - | 参与特征计算的参数 |
| 5 | time\_windows | int(10) | - | 时间窗口（单位：秒） |
| 6 | description | varchar(255) | - | 特征描述 |

表4-5 规则表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rules** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | 规则id |
| 2 | rule\_name | varchar(50) | - | 规则名称 |
| 3 | group\_id | int | FOREIGN KEY | 所属规则组id |
| 4 | feature\_id | varchar(255) | FOREIGN KEY | 参与特征计算的参数 |
| 5 | operator | varchar(4) | - | 操作符 |
| 6 | threshold | varchar(50) | - | 阈值 |
| 7 | description | varchar(255) | - | 规则描述 |

表4-6 规则阈值记录表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rules\_threshold\_log** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | rule\_id | int | FOREIGN KEY | 规则id |
| 3 | threshold | varchar(50) | - | 阈值 |
| 4 | time | datetime | - | 时间 |
| 5 | isValid | bool | - | 该阈值是否有效 |

表4-7 规则组表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rule\_group** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | 规则组id |
| 2 | group\_name | varchar(50) | - | 规则组名称 |
| 3 | business\_code | varchar(50) | - | 所属业务线 |
| 4 | hit\_logic | varchar(255) | - | 命中逻辑 |
| 5 | description | varchar(255) | - | 规则组描述 |

表4-8 惩罚措施表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：protection** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | 惩罚措施id |
| 2 | protection\_name | varchar(50) | - | 惩罚措施名称 |
| 3 | protection\_type | varchar(50) | - | 惩罚措施类型 |

表4-9 规则防控表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rule\_protection** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | rule\_id | int | FOREIGN KEY | 规则id |
| 3 | protection\_id | varchar(50) | FOREIGN KEY | 惩罚措施id |

表4-10 规则组防控表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rule\_group\_protection** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | rule\_group\_id | int | FOREIGN KEY | 规则组id |
| 3 | protection\_id | varchar(50) | FOREIGN KEY | 惩罚措施id |

表4-11 事件表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：events** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | 事件id |
| 2 | event\_name | varchar(100) | - | 事件名称 |
| 3 | enable\_status | tinyint(4) | - | 是否启用 |
| 4 | description | varchar(255) | - | 事件描述 |

表4-12 决策树表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：decision\_tree** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | group\_id | int | FOREIGN KEY | 规则组id |
| 3 | event\_id | int | FOREIGN KEY | 所属事件id |
| 4 | is\_first\_group | bool | - | 是否为该决策树第一个规则组 |
| 5 | hit\_decision | int | FOREIGN KEY | 命中待处理规则组id |
| 6 | unhit\_decision | int | FOREIGN KEY | 未命中待处理规则组id |

表4-13 用户行为记录表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：user\_behavior\_record** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | bigint | PRIMARY KEY | id |
| 2 | uid | varchar(50) | - | 用户uid |
| 3 | time | datetime | - | 用户操作时间 |
| 4 | phone | varchar(30) | - | 用户手机号 |
| 5 | imei | varchar(30) | - | 设备号 |
| 6 | ip | varchar(30) | - | 操作IP |
| 6 | event\_id | int | FOREIGN KEY | 事件id |
| 7 | business\_code | varchar(50) | - | 所属业务线 |
| 8 | sys | varchar(20) | - | 操作来源系统 |

表4-14 规则命中记录表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：rule\_hit\_record** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | uid | varchar(50) | - | 用户uid |
| 3 | time | datetime | - | 规则命中时间 |
| 4 | phone | varchar(30) | - | 用户手机号 |
| 5 | imei | varchar(30) | - | 设备号 |
| 6 | event\_id | int | FOREIGN KEY | 事件id |
| 7 | rule\_id | int | FOREIGN KEY | 命中规则id |
| 8 | rule\_group\_id | int | FOREIGN KEY | 命中规则组id |
| 9 | is\_blacklist | bool | - | 该请求是否来自黑名单 |
| 10 | is\_valid | bool | - | 是否验证通过 |
| 11 | user\_behavior\_record\_id | bigint | FOREIGN KEY | 对应用户行为id |

表4-15 用户分数变化日志表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：user\_score\_log** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | bigint | PRIMARY KEY | id |
| 2 | uid | varchar(50) | - | 用户uid |
| 3 | business\_code | varchar(50) | - | 业务线 |
| 4 | change\_score | int | - | 本次修改的分数 |
| 5 | before\_score | int | - | 修改前分数 |
| 6 | after\_score | int | - | 修改后分数 |
| 7 | change\_reason | varchar(50) | - | 修改原因 |
| 8 | business\_code | varchar(50) | - | 所属业务线 |
| 9 | rule\_hit\_record\_id | int | FOREIGN KEY | 对应规则命中记录id |

表4-16 用户分数表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：user\_score** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | bigint | PRIMARY KEY | id |
| 2 | uid | varchar(50) | - | 用户uid |
| 3 | business\_code | varchar(50) | - | 业务线 |
| 4 | current\_score | int | - | 当前分数 |

表4-17 用户标签表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：user\_tag** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | tag\_name | varchar(50) | - | 标签名称 |
| 3 | tag\_score | int | - | 标签权重 |

表4-18 黑名单表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：blacklist** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | black\_key | varchar(50) | - | 黑名单key值，可为手机号、设备号等 |
| 3 | type | varchar(20) | - | 黑名单类型（IP/phone/uid…） |
| 4 | business\_code | varchar(50) | - | 业务线 |
| 5 | add\_reason | varchar(255) | - | 加入原因 |
| 6 | add\_user | varchar(50) | - | 添加人邮箱前缀 |
| 7 | add\_time | datetime | - | 添加时间 |
| 8 | is\_del | bool | - | 是否删除 |

表4-19 白名单表设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表名：blacklist** | | | | |
| **序号** | **字段名** | **类型** | **属性** | **描述** |
| 1 | id | int | PRIMARY KEY | id |
| 2 | white\_key | varchar(50) | - | 白名单key值，可为手机号、设备号等 |
| 3 | type | varchar(20) | - | 白名单类型（IP/phone/uid…） |
| 4 | business\_code | varchar(50) | - | 业务线 |
| 5 | add\_reason | varchar(255) | - | 加入原因 |
| 6 | add\_user | varchar(50) | - | 添加人邮箱前缀 |
| 7 | add\_time | datetime | - | 添加时间 |
| 8 | is\_del | bool | - | 是否删除 |

4.4 本章小结

本章在第三章需求分析的基础上，介绍了系统的整体架构设计，并对系统功能模块进行划分，简要介绍了各个功能模块的实现方法。最后介绍系统的数据库设计，提供了系统整体ER图以及各个表的设计细节。

5 租赁业务风险控制安全中台详细设计与实现

本章将介绍租赁业务风险控制安全中台对的详细设计与实现，以第四章中概要设计为基础，分别针对敏感数据审计模块、风险情况分析模块、规则引擎构建模块和规则执行监控模块介绍实现的类图与时序图，并对实现效果进行展示。

5.1 敏感数据审计模块

该模块主要实现对公司内部员工查看敏感数据的监控功能。包括敏感数据审计规则管理、敏感数据审计规则执行、敏感数据审计结果展示三部分。业务流程如图5-1所示。

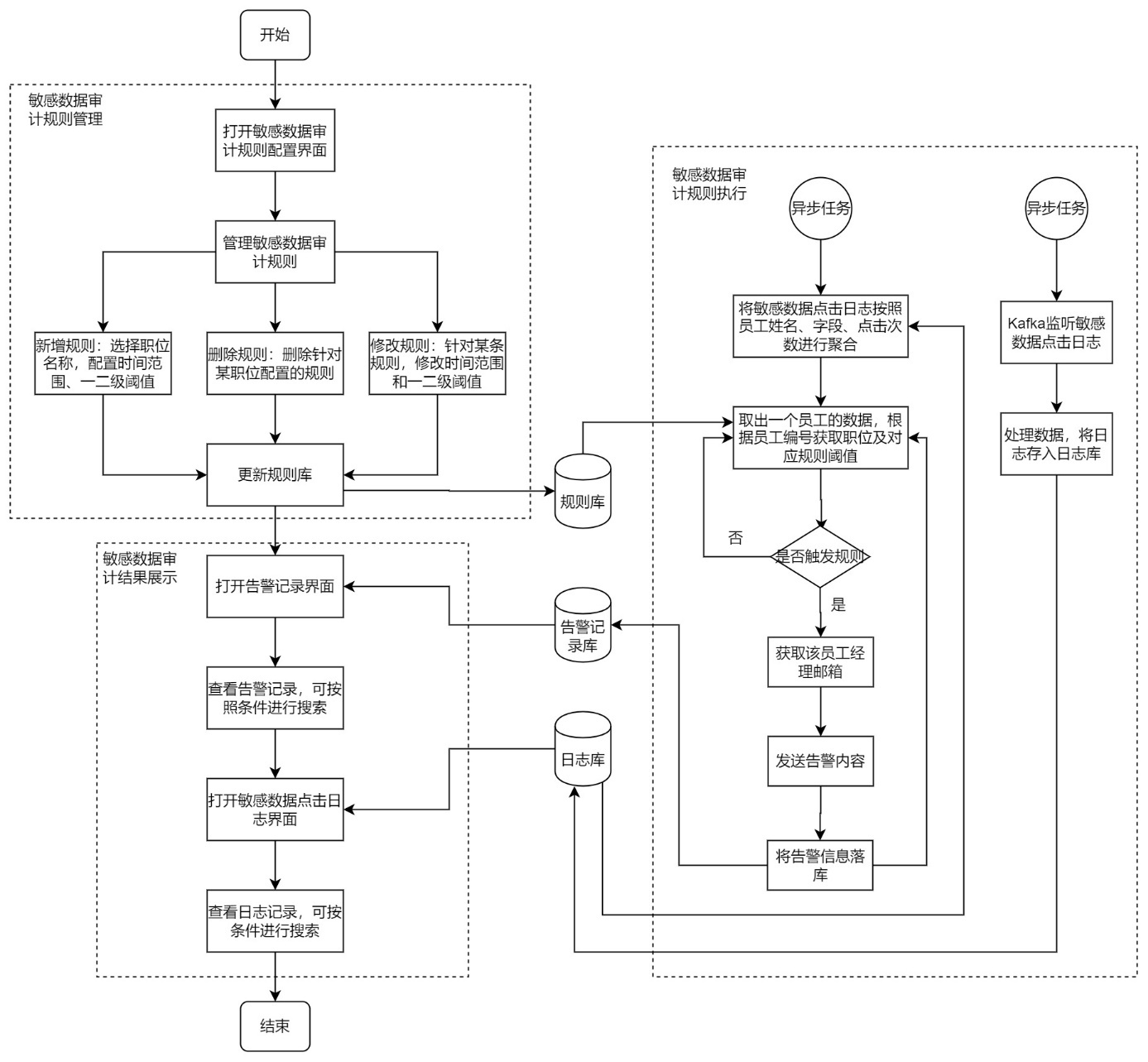


图5-1 敏感数据审计模块业务流程图

下面将分别介绍各部分的详细设计方案。

5.1.1 敏感数据审计规则管理

该部分主要实现敏感数据审计规则的添加、修改、删除、查看功能，类图如图5-2所示。该部分主要涉及6个类。dataClickRulesController类主要负责接收前端请求，dataClickRules类为敏感数据审计规则的实体类，dataClickRulesService接口接收dataClickRules实例，响应前端请求，并交由dataClickRulesImpl类进行实现，其中主要包括三个方法，完成对规则的修改和查询。dataClickRulesMapper类配合dataClickRulesMapper.xml文件，完成对数据库的操作。CityEnum类中保存规则管理过程中使用到的常量，包括城市和职位信息等。

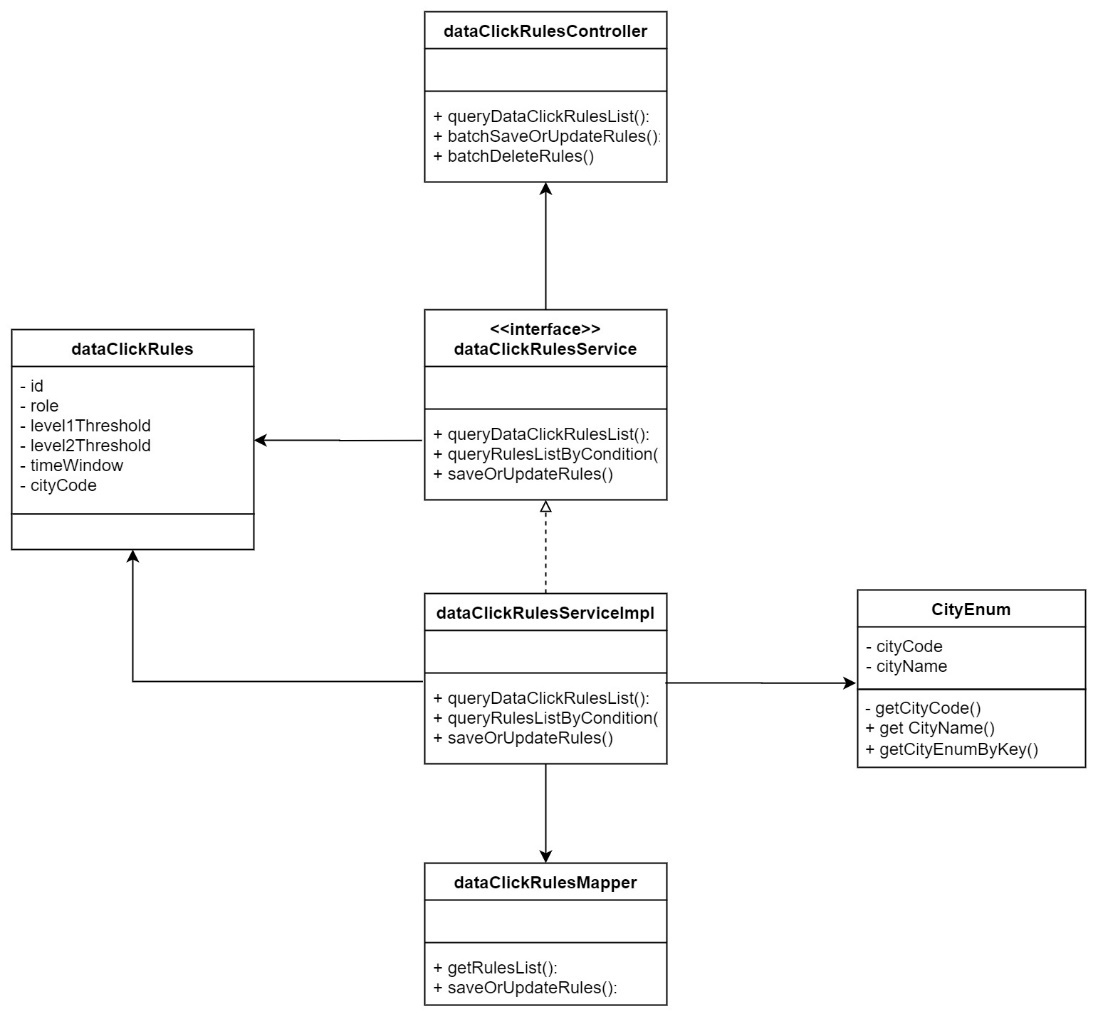


图5-2 敏感数据审计规则管理类图

下面以添加规则为例简单介绍规则管理流程，时序图如图5-3所示。dataClickRulesController类接收前端请求，当前端调用该类的queryDataClickRulesList方法发出查看规则列表请求时，controller调用dataClickRulesService的同名方法，并由dataClickRulesImpl类实现。Impl类调用dataClickRulesMapper类中的getRulesList方法，由mapper操作数据库，并将规则列表返回，由service将结果返回至controller，由controller发送至前端，在界面上进行展示。此时风控策略师通过前端界面添加规则，前端调用controller类中的batchSaveOrUpdateRules方法将请求发送至后端，controller调用service的saveOrUpdateRules方法进行处理。service先从cityEnum类中获取规则相应参数，包括城市、职位等，再调用mapper类操作数据库完成规则添加。



图5-3 敏感数据点击规则添加时序图

敏感数据审计规则管理实现效果如图5-4所示，其中部分涉密数据已遮挡。

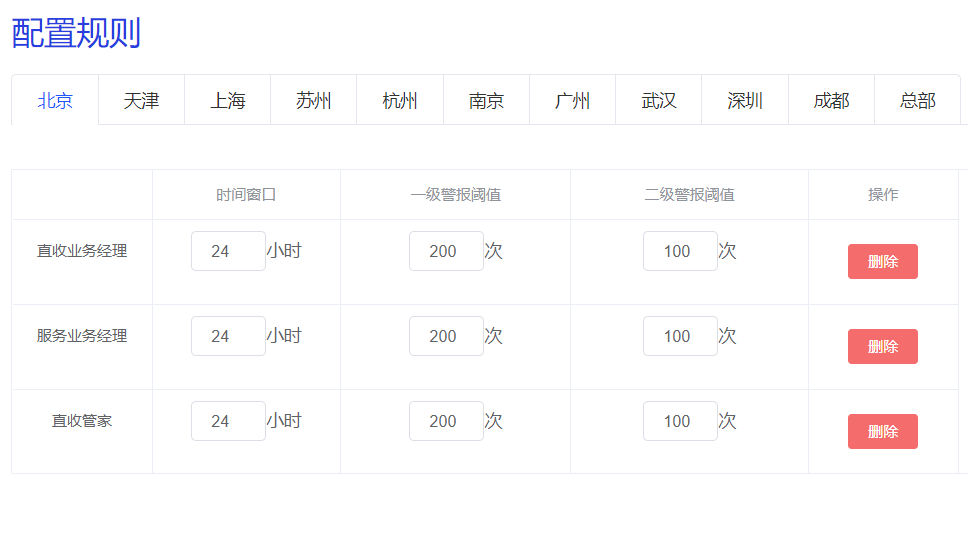


图5-4 敏感数据规则管理实现效果

5.1.2 敏感数据规则执行及结果展示

该部分主要实现执行已经配置好的敏感数据审计规则及发送告警功能，主要包括14个类，类图如图5-5所示。其中Service类均为接口，负责实现主要功能，Impl类为对应Service的实现类。Mapper类负责和数据库交互。

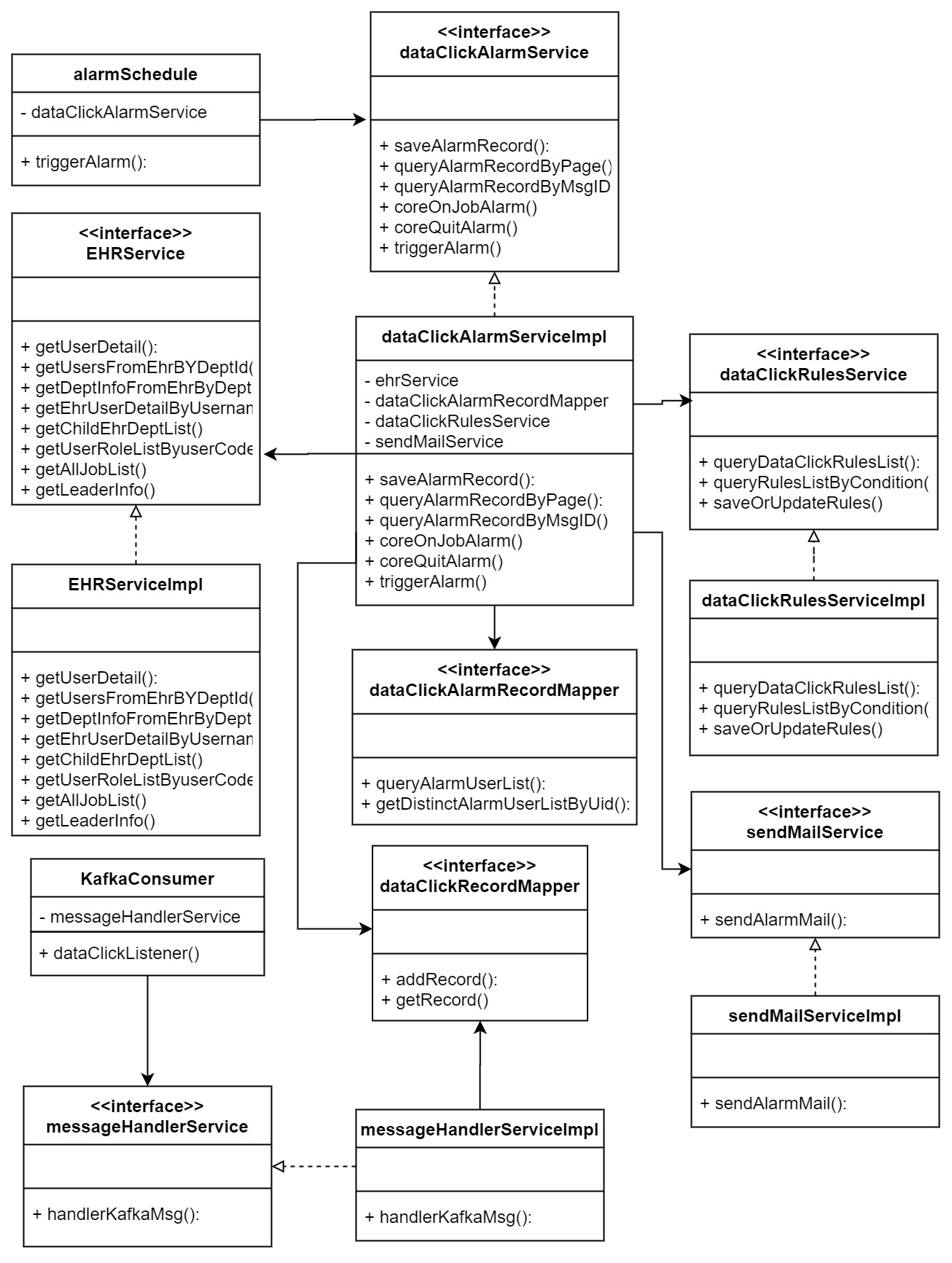


图5-5 敏感数据审计规则执行类图

其中messageHandlerService、messageHandlerServiceImpl、KafkaConsumer、dataClickRecordMapper这四个类主要完成敏感数据点击日志的接收与保存，KafkaConsumer负责监听Kafka中的日志消息，并交由messageHandlerService进行数据格式转化与保存，dataClickRecordMapper负责操作数据库，messageHandlerServiceImpl为messageHandlerService的实现类。

其余10个类负责规则执行与告警信息发布。由alarmSchedule类开启定时任务，调用triggerAlarm方法执行规则，dataClickAlarmService为规则执行核心类，负责聚合用户点击数据，并根据用户状态（在职/离职）和用户城市与职位编码与规则进行匹配。其中，用户点击数据的聚合由dataClickRecordMapper类完成，用户状态与职位可借助EHRService类调用EHR系统（公司内部人力资源系统）接口进行查询，规则数据可通过调用dataClickRulesService类中的query方法获得。若触发规则，则通过EHRService类获得员工领导及安全部门负责人信息，并通过sendMailService类完成邮件和企业微信发送，同时使用dataClickAlarmRecordMapper类更新告警记录数据库。四个Impl类为对应接口的具体实现类。

敏感数据点击规则运行及告警时序图如图5-6所示。

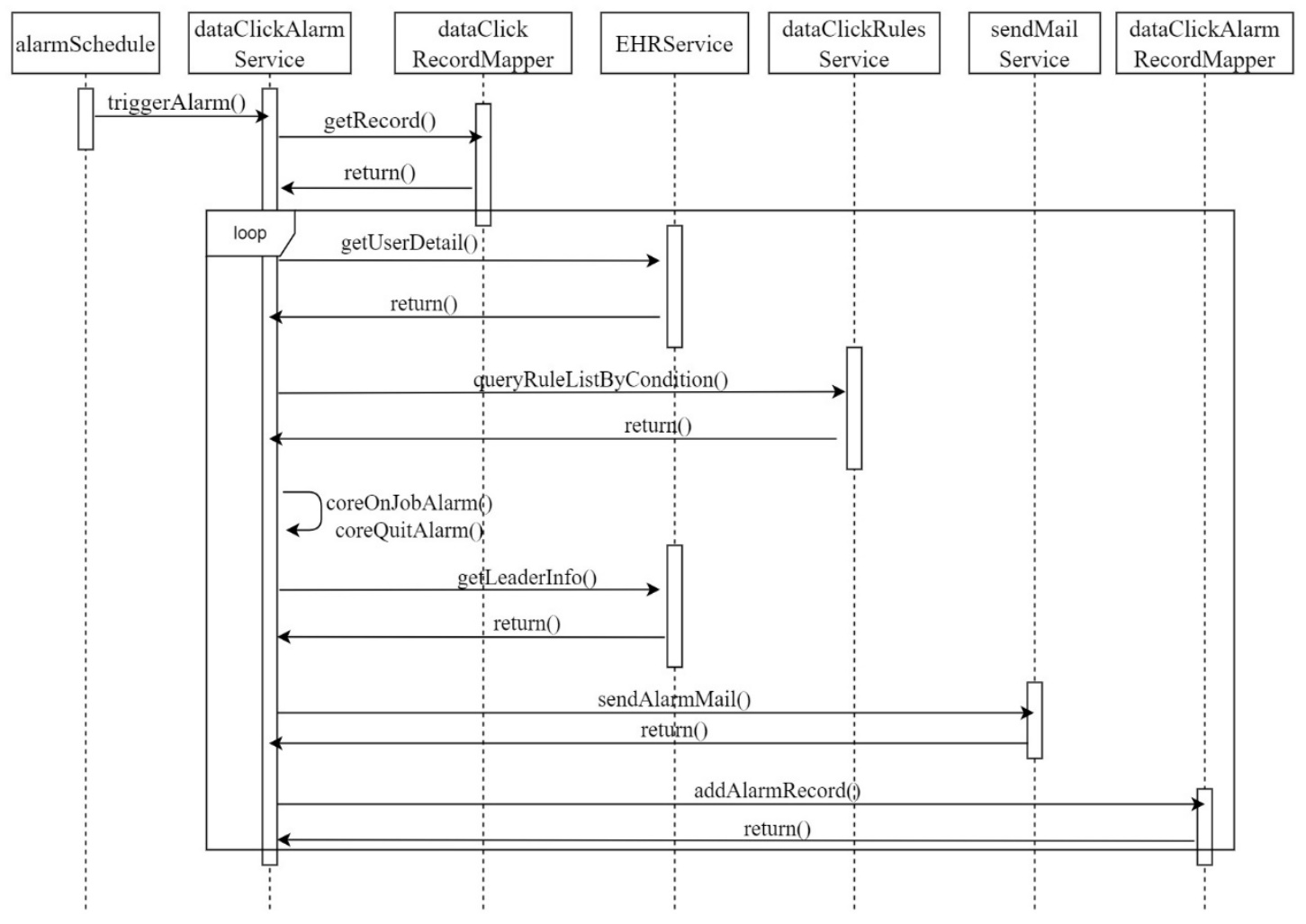


图5-6 敏感数据点击规则运行及告警时序图

其中，调用dataClickRecordMapper类中的getRecord方法返回的是一个list，里面包含每一个员工的工号以及点击次数，因此在dataClickAlarmService的triggerAlarm方法中，使用循环遍历list中的所有员工，依次判断是否触发规则。

敏感数据规则结果展示实现效果如图5-7和图5-8所示，其中部分涉密数据已遮挡。



图5-7 敏感数据点击日志查看及搜索实现效果

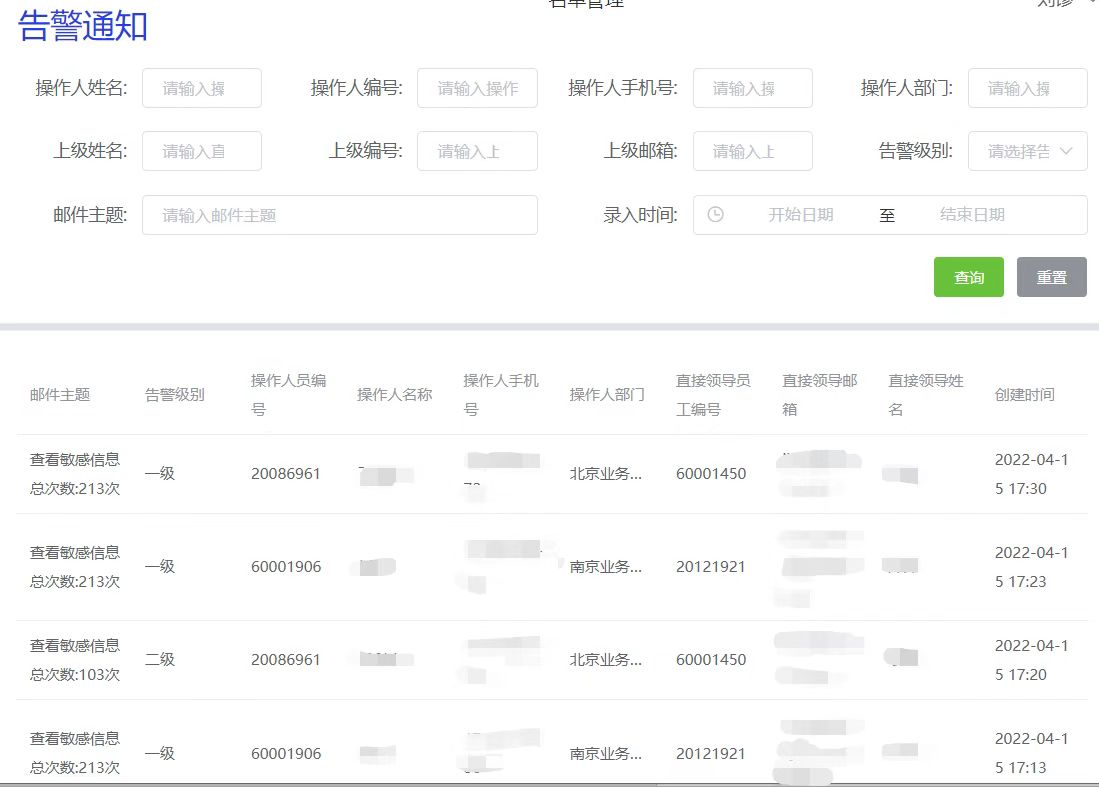


图5-8 敏感数据告警查看及搜索实现效果

5.2 风险情况分析模块

该模块主要实现平台整体风险分析及黑白名单库管理的功能，其中平台整体风险情况分析包括各业务线风险分数聚合、风险大盘展示、用户操作查看。业务流程图如图5-6所示。

平台整体风险情况分析类图如图5-9所示。共涉及11个类，其中riskScoreController类和userOperationController类接收前端请求，当前端需要获取用户分数及用户操作时，Controller类调用riskScoreService类中的方法获取数据，按照前端要求进行相应处理，如按照条件筛选、聚合等，并将结果返回至Controller类。其中，涉及到操作数据库的部分由userScoreMapper和userScoreLogMapper两个类进行处理，涉及到的数据传输借助UserScoreDto和userScoreLogDto完成。

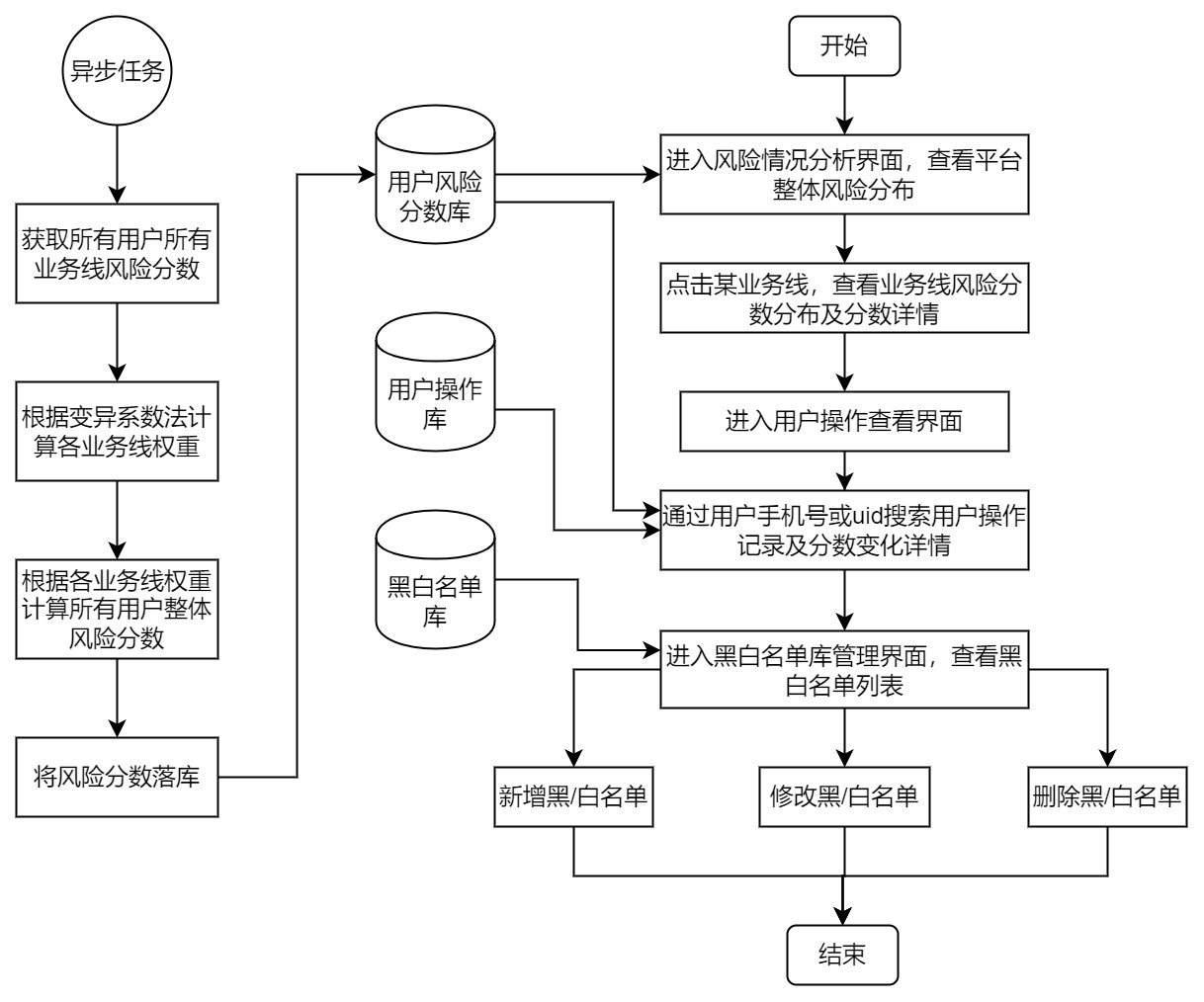


图5-9 平台整体风险情况分析业务流程图

下面以“查看某一业务线所有用户风险分数详情”这一场景简单介绍该模块的处理流程。时序图如5-10所示。riskScoreController类收到前端请求，调用自身getRiskDetailByBusiness函数，该函数内调用riskScoreService类中的同名函数，利用userScoreMapper访问数据库，通过getUserScoreByBusiness函数，从数据库中查找该业务线所有用户风险分数详情。riskScoreService类将结果表示为UserScoreDto组成的list返回给controller类，由controller类返回给前端。

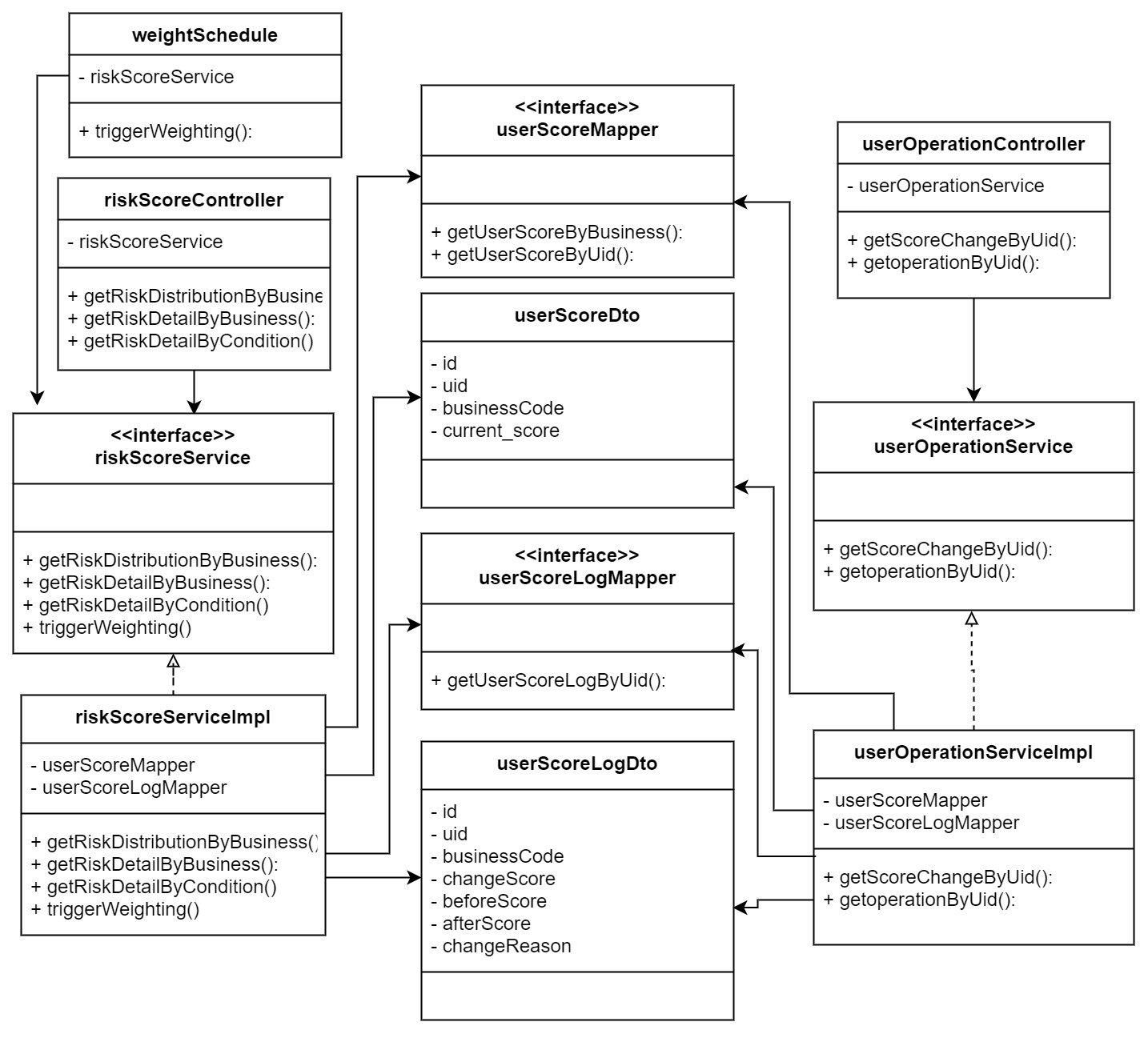


图5-10 平台整体风险情况分析类图

用户各业务线风险分数是通过规则执行情况自动计算实时更新，但用户在平台整体风险分数是利用各业务线分数，通过加权平均计算求得。且各业务线加权系数是利用变异系数法，通过全部用户全部业务线分数情况计算得到，该值并非固定值，每1小时更新一次。使用WeightSchedule类开启定时任务，每1小时调用一次triggerWeighting函数，利用riskScoreService类中的同名函数进行计算，并使用userScoreMapper类更新数据库。

黑白名单库管理部分类图如图5-11所示。

该部分共涉及8个类，其中BlackListController和WhiteListController类接收前端请求，交由BlackListService和WhiteListService处理，涉及数据库操作部分由blackListMapper类完成。

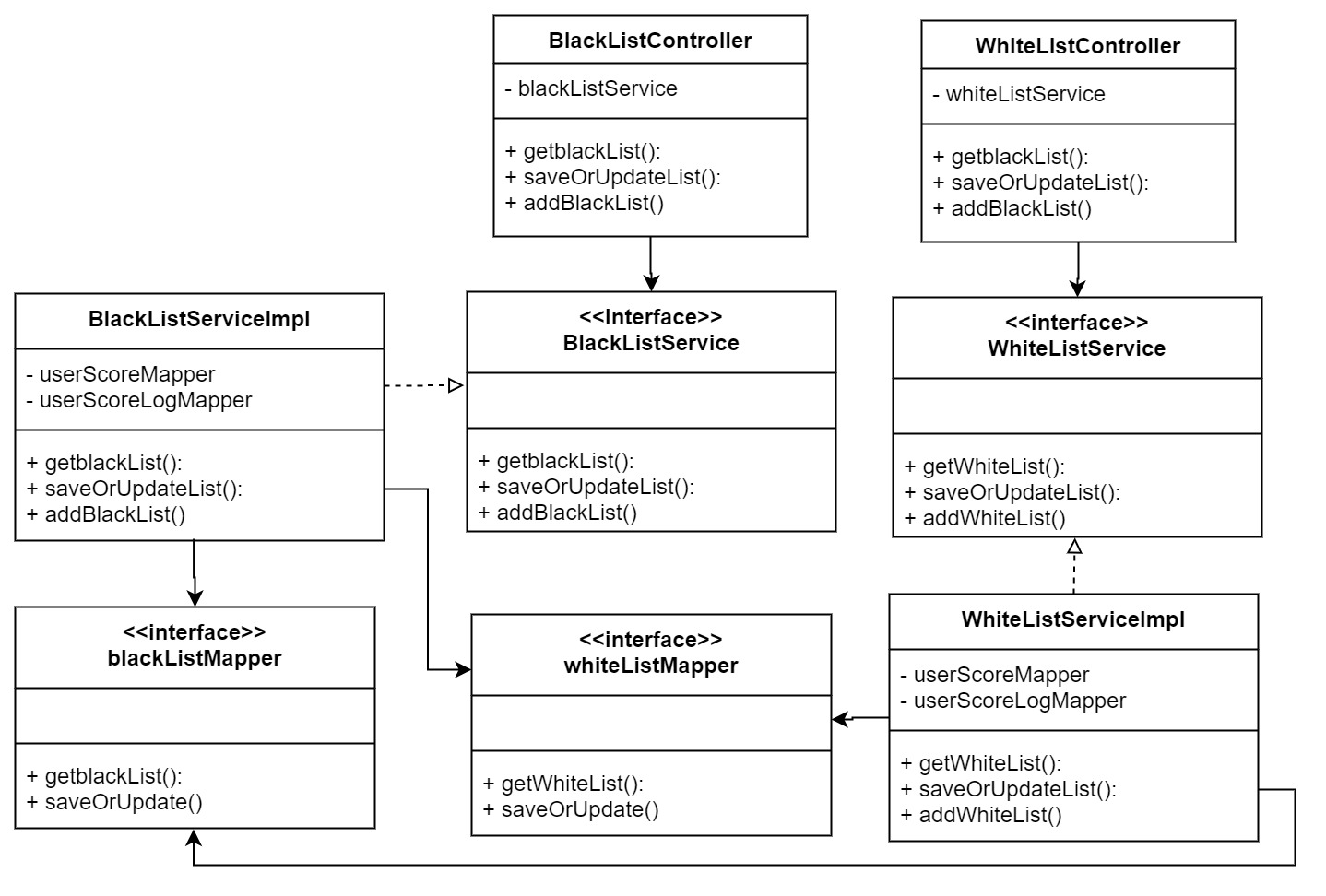


图5-11 黑白名单库管理类图

风险情况分析模块实现效果如图5-12—5-15所示，敏感数据已用测试数据代替或使用掩码遮挡。

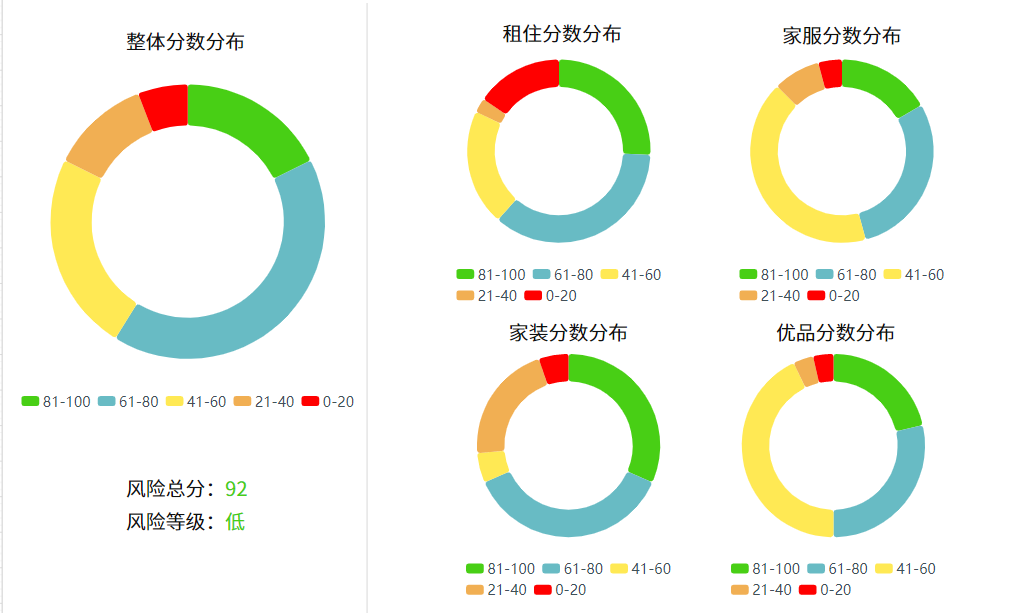


图5-12 平台整体风险情况分析实现效果



图5-13 业务线风险情况分析实现效果

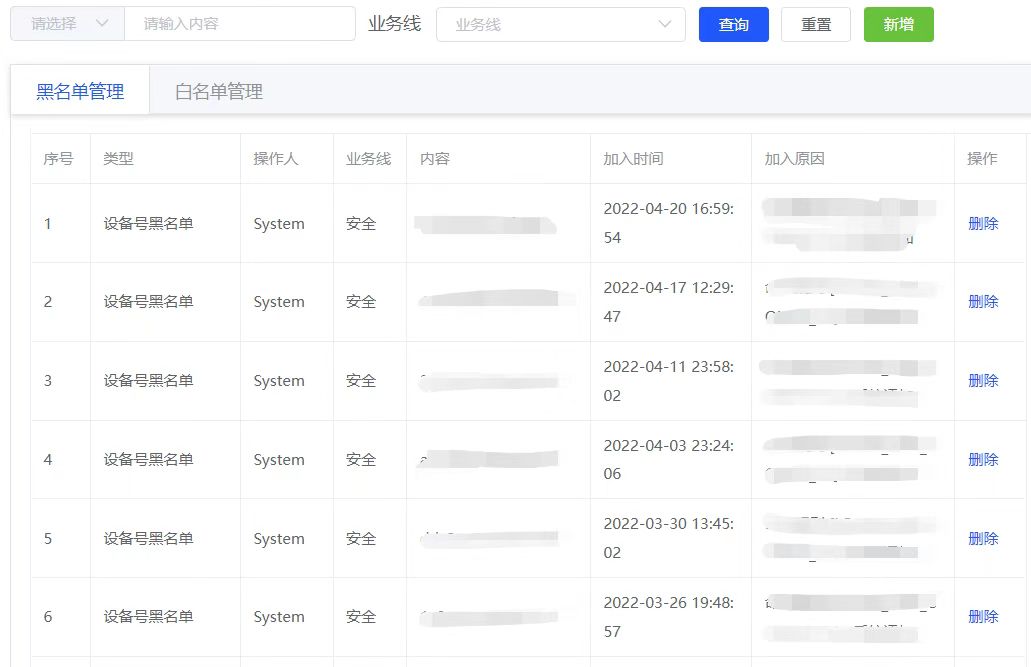


图5-14 黑白名单库管理实现效果



图5-15 用户操作查看实现效果

5.3 规则引擎构建模块

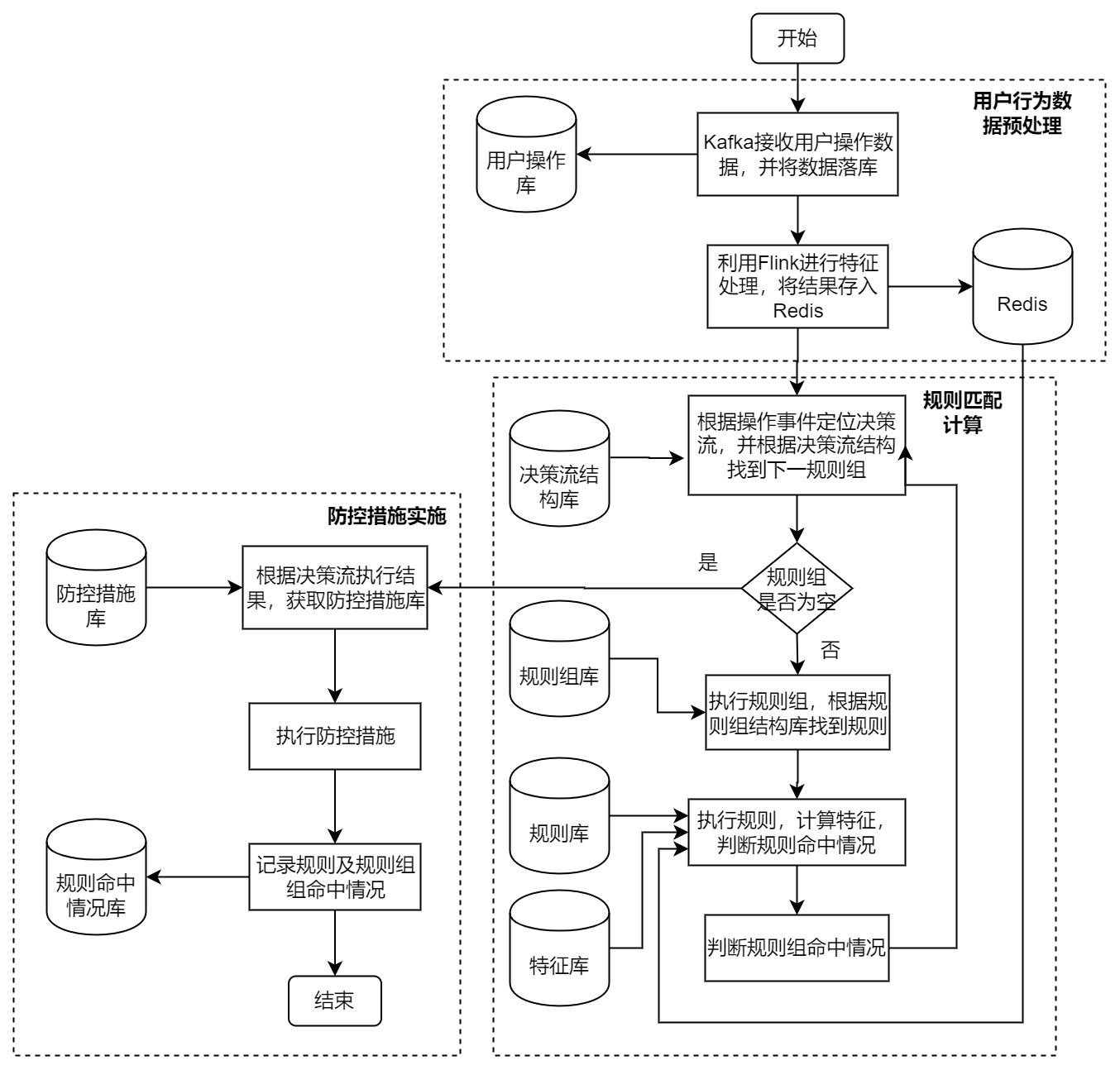
该模块业务流程图如图5-16所示。该模块主要实现规则、规则组、决策树三层规则引擎构建以及规则引擎运行功能，其中又分为3个部分：用户行为数据预处理、规则匹配计算、防控措施实施。

图5-16 规则引擎构建模块业务流程图

用户行为数据预处理部分负责接收用户行为数据，完成数据落库，并利用Flink对部分特征进行计算。规则匹配计算部分负责根据规则引擎构建结果，匹配规则进行计算，并返回结果。防控措施实施部分负责根据规则执行结果实施对应防控措施，并更新规则执行结果库。下面依次介绍这3部分的详细设计。

5.3.1 用户行为数据预处理

该部分类图如图5-17所示。该模块共涉及6个类，模块入口为KafkaConsumer类，负责监听Kafka中发送的事件，在该类中为每个事件设立一个Listener函数。此处为方便描述，仅在图中展示登陆、注册两个事件作为示例。EventHandlerService类负责对事件进行处理，实现用户操作落库，利用Flink完成对部分用户操作的处理，如计数等，并将处理结果存入Redis中。对Redis的操作由RedisUtil工具类完成，UserBehaviorRecordMapper和UserScoreInfoMapper类负责操作数据库user\_behavior\_record和user\_score表，记录用户操作数据。

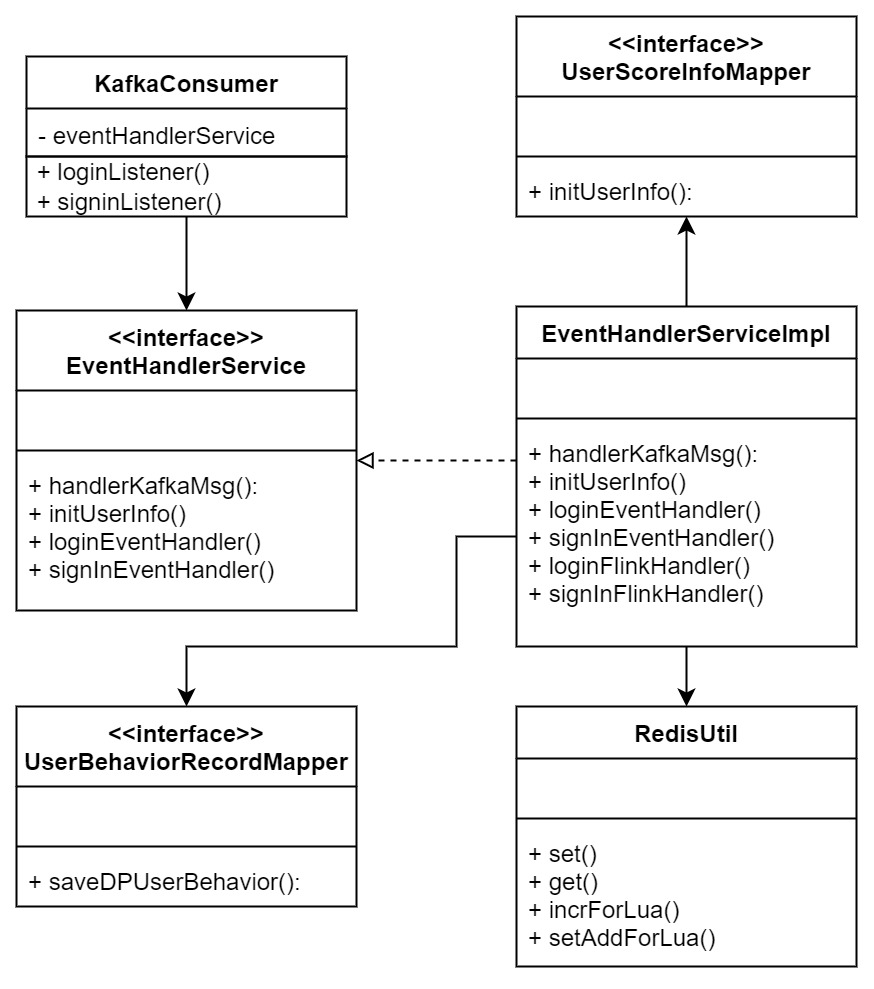


图5-17 用户行为数据预处理类图

用户行为数据预处理时序图如图5-18所示，以注册事件为例。

KafkaConsumer类通过调用自身signInListener方法，从Kafka中监听用户注册事件行为数据，当监听到事件发生后，调用EventHandlerService类中handlerKafaMsg方法，完成用户初始化和操作数据落库。EventHandlerService类调用自身initUserInfo函数，借助UserScoreInfoMapper类中的同名方法，完成对用户进行初始化操作，待Mapper类返回结果后，调用UserBehaviorRecordMapper类中的saveDPUserBehavior方法，将用户操作保存至Mysql数据库中。EventHandlerService向KafkaConsumer返回用户初始化及操作落库结果。之后，KafkaConsumer调用EventHandlerService中signInEventHandler方法，完成对用户注册这一事件中部分特征的计算。EventHandlerService调用自身signInFlinkHandler方法，利用Flink完成特征计算并将结果存储至Redis。在注册事件中，特征计算包括将该时间段内该IP、imei、phone的注册次数+1，并完成定义时间内注册次数的聚合。对Redis的操作借助RedisUtil类中set()、incrForLua()等方法实现。

其余事件实现时序图与注册事件类似，每个事件定义了不同的FlinkHandler函数，根据参数不同，对特征有不同的操作。

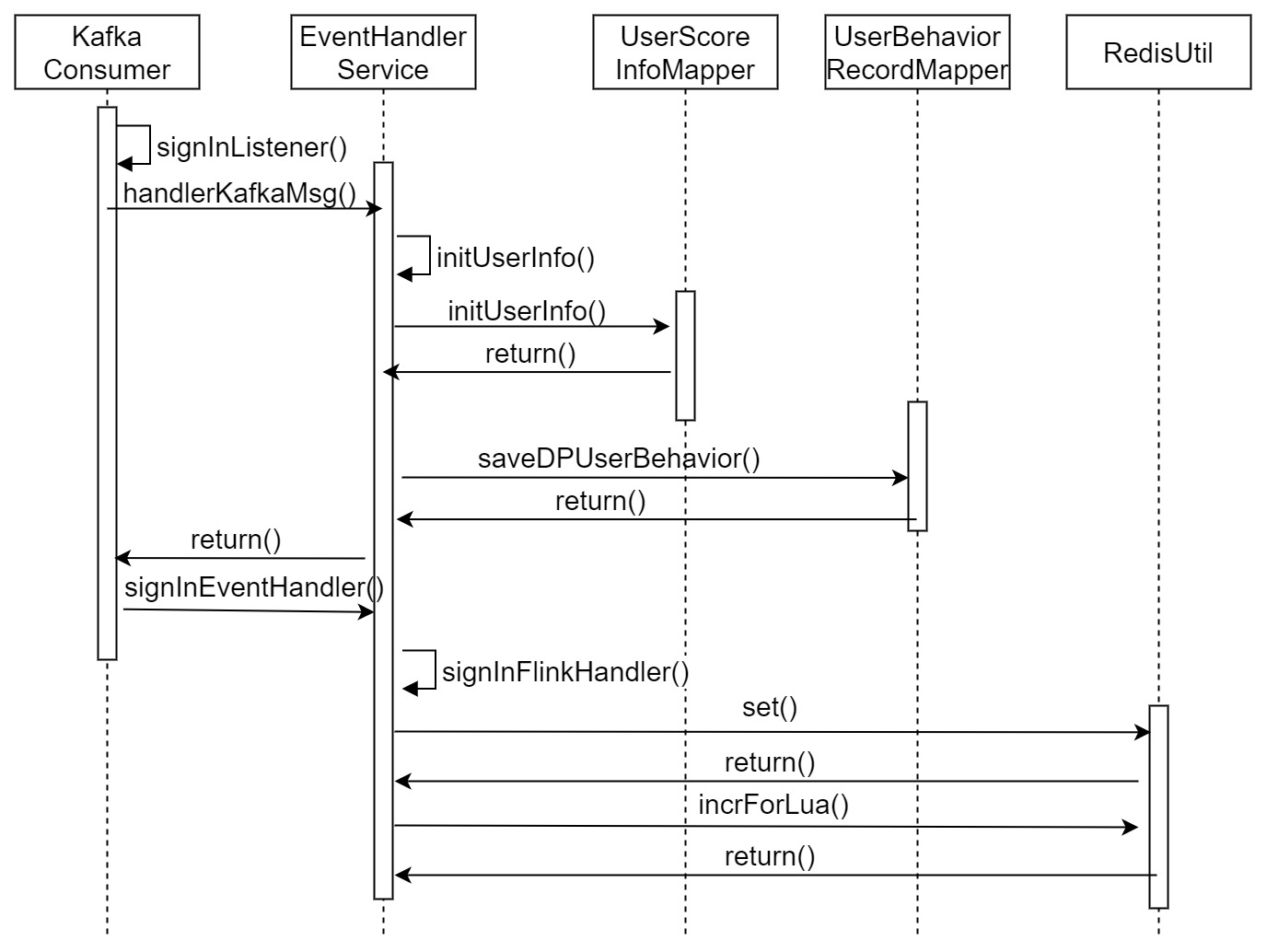


图5-18 注册事件用户行为数据预处理时序图

5.3.2 规则匹配计算

规则匹配计算类图如图5-19所示。该部分共涉及18个类，其中EventHandlerService为规则计算起始类，EngineService、RuleGroupService、RuleService、FeatureService四个类分别负责完成决策树、规则组、规则及特征的匹配与计算，FeatureMap中存储了特征名称及特征实体间的对应关系，Mapper类负责操作数据库。

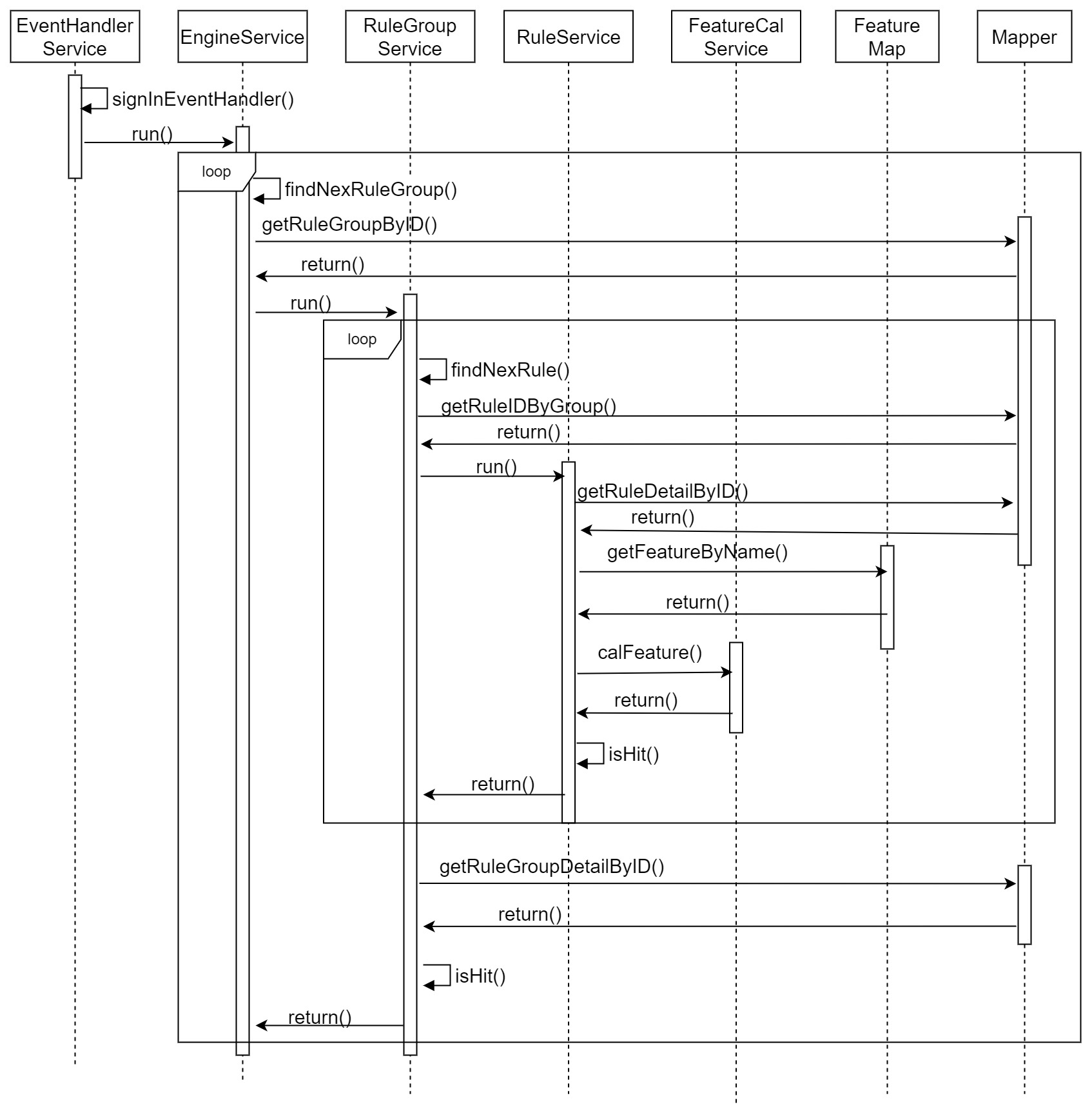


图5-19 规则匹配计算时序图

四个Service类通过run方法逐层调用计算，并逐层将结果返回，及FeatureService将特征计算结果返回给RuleService，RuleService将规则计算结果返回给RuleGroupService，RuleGroupService将规则组计算结果返回给EngineService。RuleGroup、Rule、Feature分别为规则组、规则和特征的实体类，不同特征通过继承Feature类，重写calFeature方法实现，该模块中涉及的5个Mapper类负责操作数据库中对应的表，Impl类为对应Service类接口的具体实现类。规则匹配计算时序图如图5-20所示。

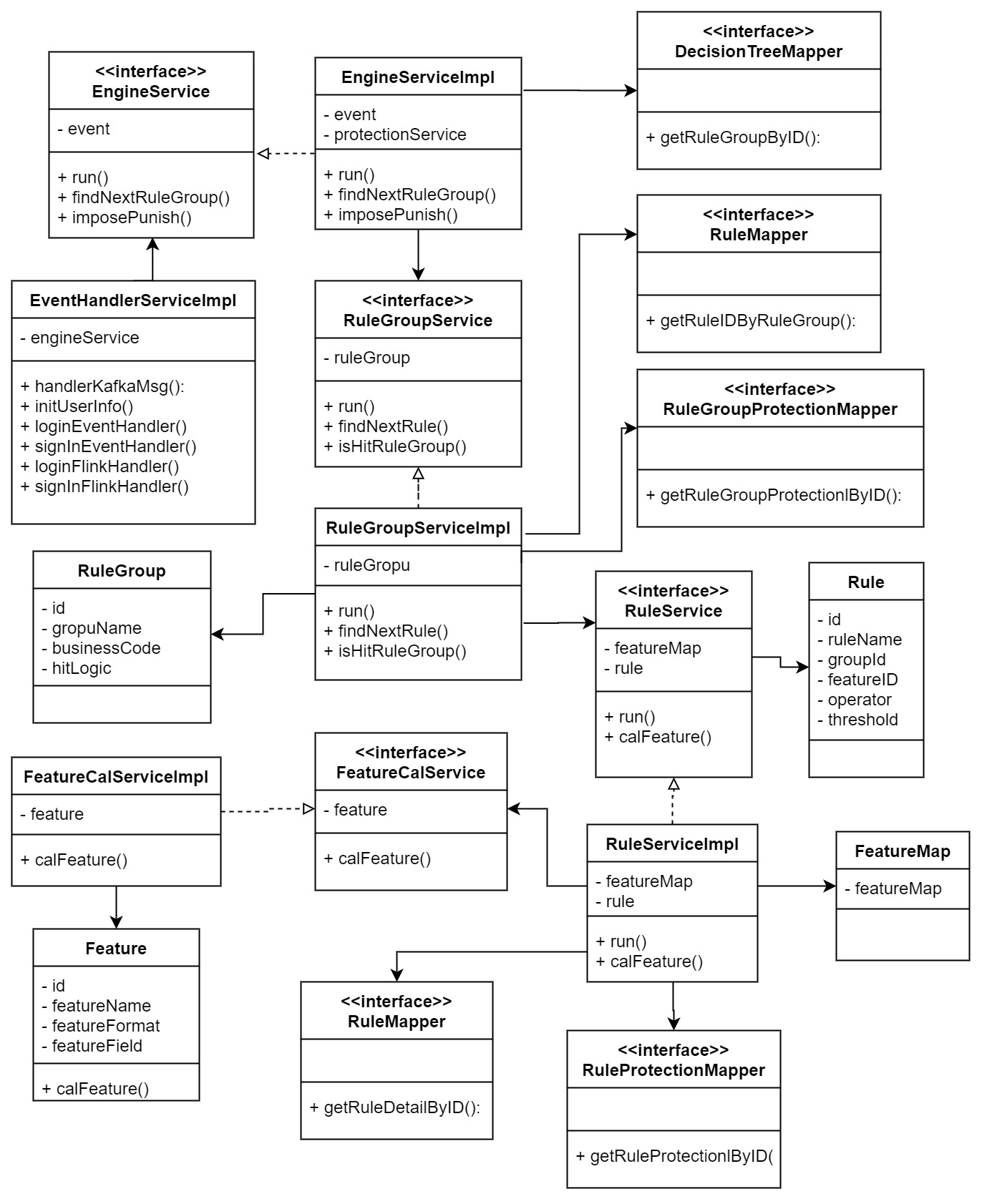


图5-20 规则匹配计算类图

5.3.3 防控措施实施

防控措施实施部分类图如图5-21所示，共涉及7个类。其中，EngineService为防控措施实施逻辑控制类，通过RuleGroupProtectionMapper类访问数据库获取防控措施详情，并使用imposePunish方法调用ProtectionService类中的exeProtection方法实施防控措施。其中Protection类为防控措施实体。最后，通过RuleHitRecordMapper类修改数据库中rule\_hit\_mapper表，将规则命中情况落库。

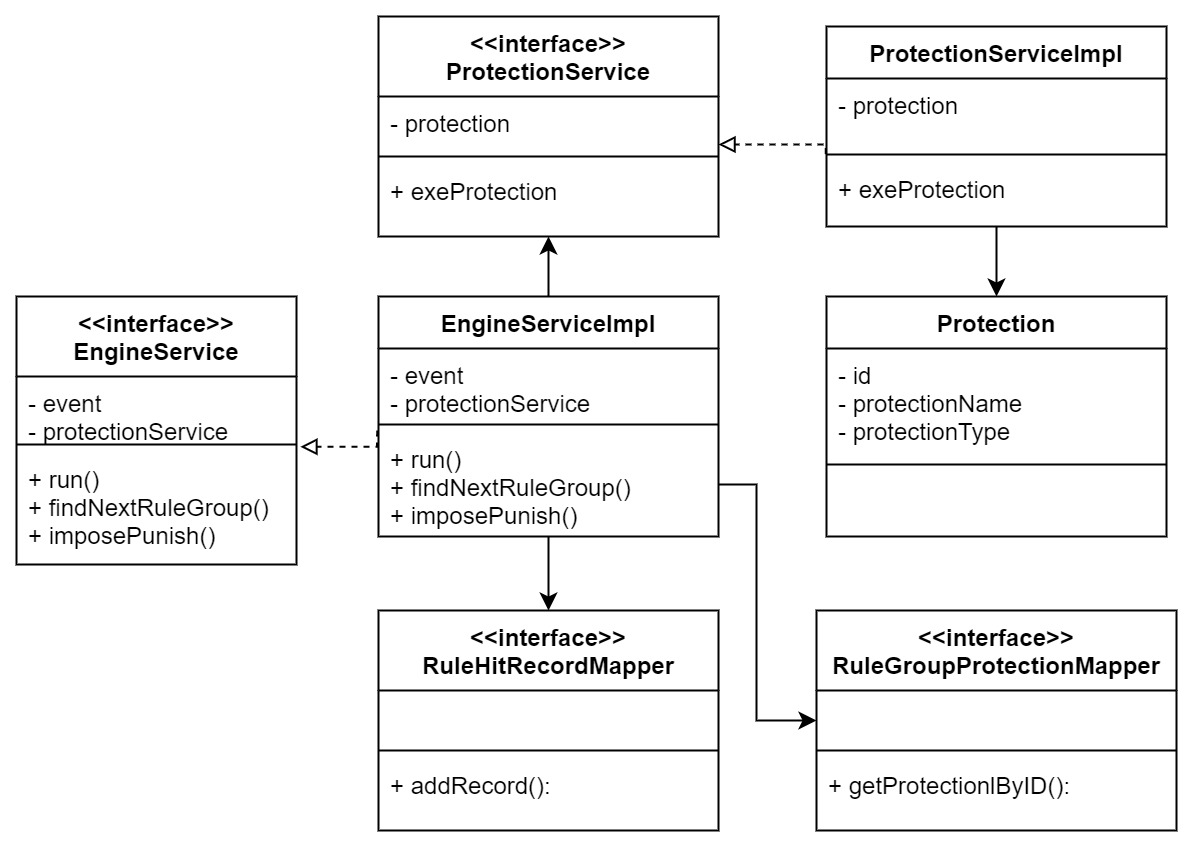


图5-21 防控措施实施类图

5.4 规则执行监控模块

该模块主要实现规则阈值预测及自动调整和规则评价，其中规则评价部分包括规则有效性分析、规则命中情况统计、异常用户名单分析功能。下面分别介绍规则阈值预测和规则评价两部分的详细设计。

5.4.1 规则阈值预测及自动调整

规则阈值预测及自动调整业务逻辑如图5-22所示。每日零点执行一次该任务，从规则历史阈值库中获取所有规则历史数据，循环遍历所有规则，将该规则历史数据传入三次指数平滑法计算函数中，得到当日阈值的预测值，并修改规则库中该规则阈值。

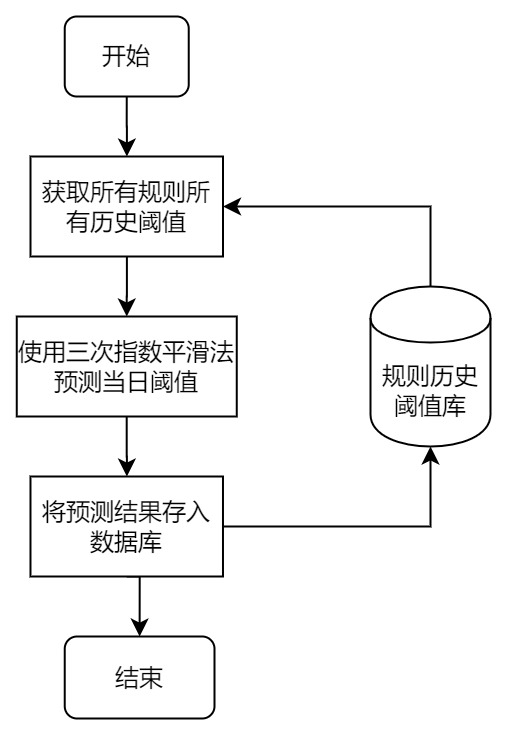


图5-22 规则阈值预测及自动调整业务流程图

类图如图5-23所示，共涉及5个类。ForecastSchedule类通过triggerForecast方法启动定时任务，每天零点进行阈值预测与调整。ThresholdForecastService类为阈值预测核心类，通过coreForecast方法，利用三次指数平滑法，以同一规则历史阈值为基础进行预测，并通过setThreshold方法将预测结果配置进数据库中。ThresholdForecastServiceImpl类为Service接口的具体实现类。涉及对数据库的操作，通过RuleThresholdLogMapper和RulesMapper两个类完成。

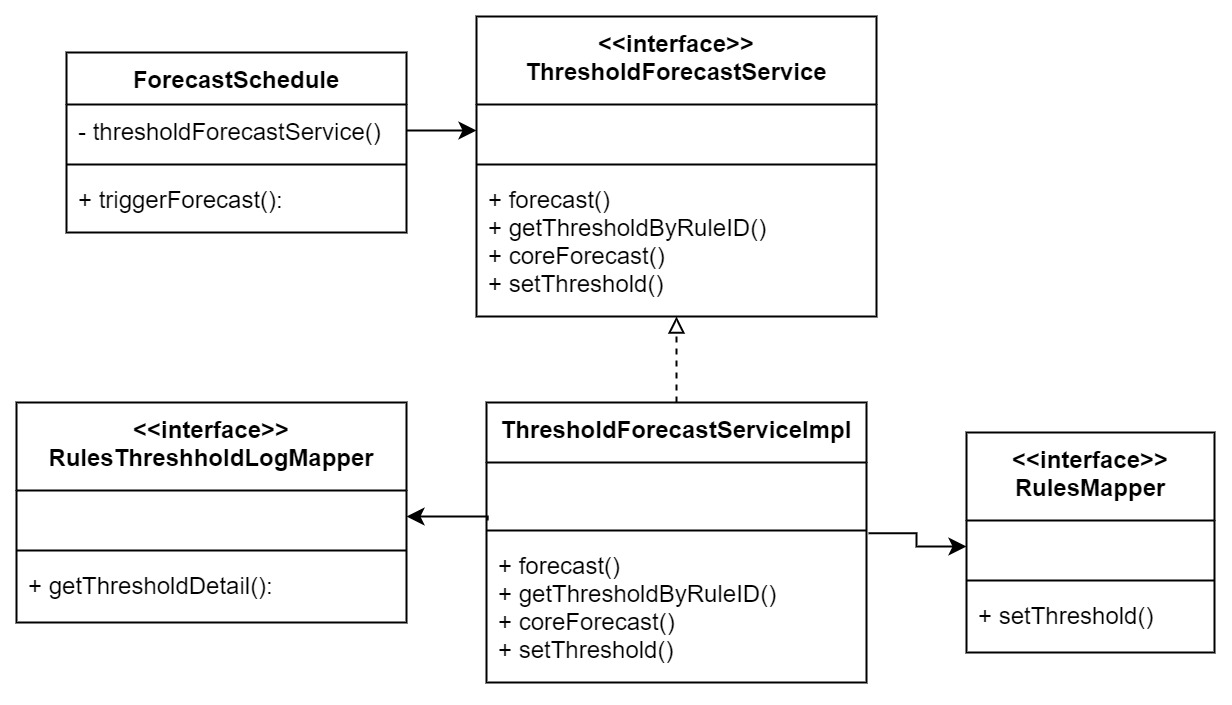


图5-23 规则阈值预测类图

5.4.2 规则评价

规则评价业务流程如图5-24所示。主任务为规则及规则组、准确性、有效性评估查看，及异常用户列表查看，包括异常账号异常设备两种异常用户。规则评价中还包含异步任务，每5分钟执行一次，从规则库中获取所有规则列表，并根据规则命中情况计算规则的准确率和有效性，若低于80%触发告警，向风控策略师发送告警信息提示修改规则。

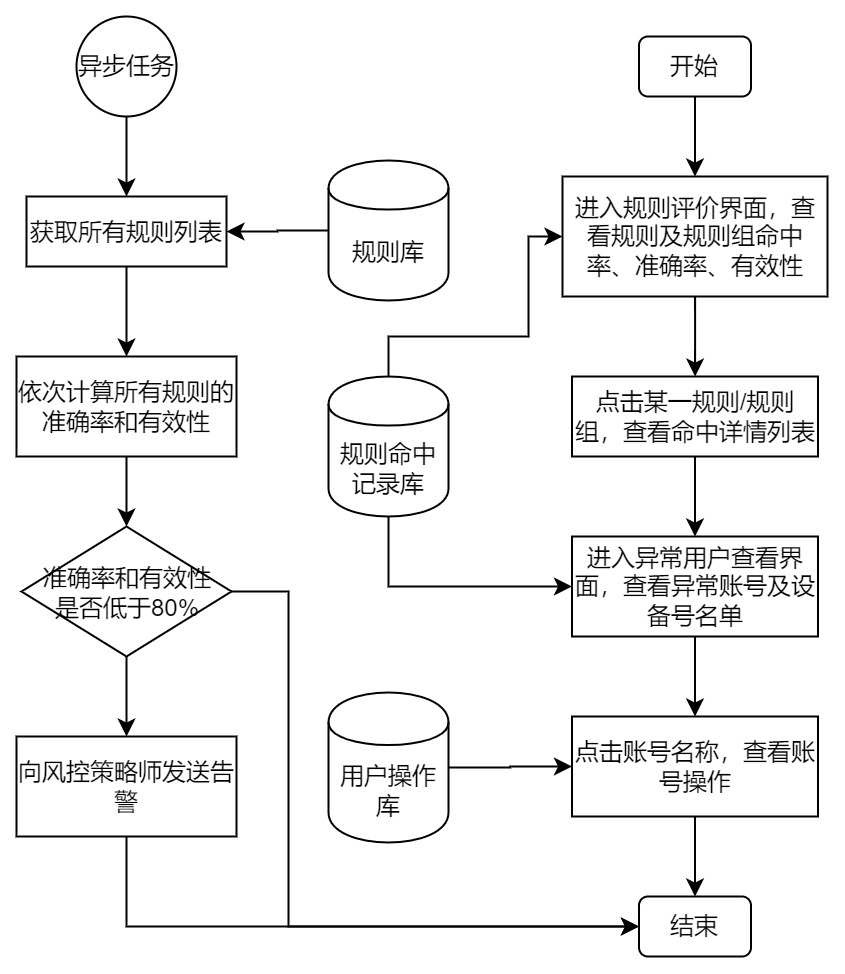


图5-24 规则评价业务流程图

规则评价部分类图如图5-25所示，共涉及10个类。RuleHitController和RuleHitService两个类负责实现规则命中率查看、具体命中列表查看以及异常用户分析展示功能。Controller类负责接收前端请求，并调用Service中的方法进行实现，RuleHitServiceImpl是Service的具体实现类。ForecastSchedule和RuleEvaluationService两个类负责规则有效性和准确性评估，由ForecastSchedule负责调度，每5分钟执行一次定时任务，由RuleEvaluationService类完成准确率和有效性的计算，若某一规则准确率和有效性低于80%，则通过sendMailService类发送告警。其中，涉及对数据库的操作由RuleHitRecordMapper完成，数据传输借助RuleHitRecordDto数据传输对象完成。

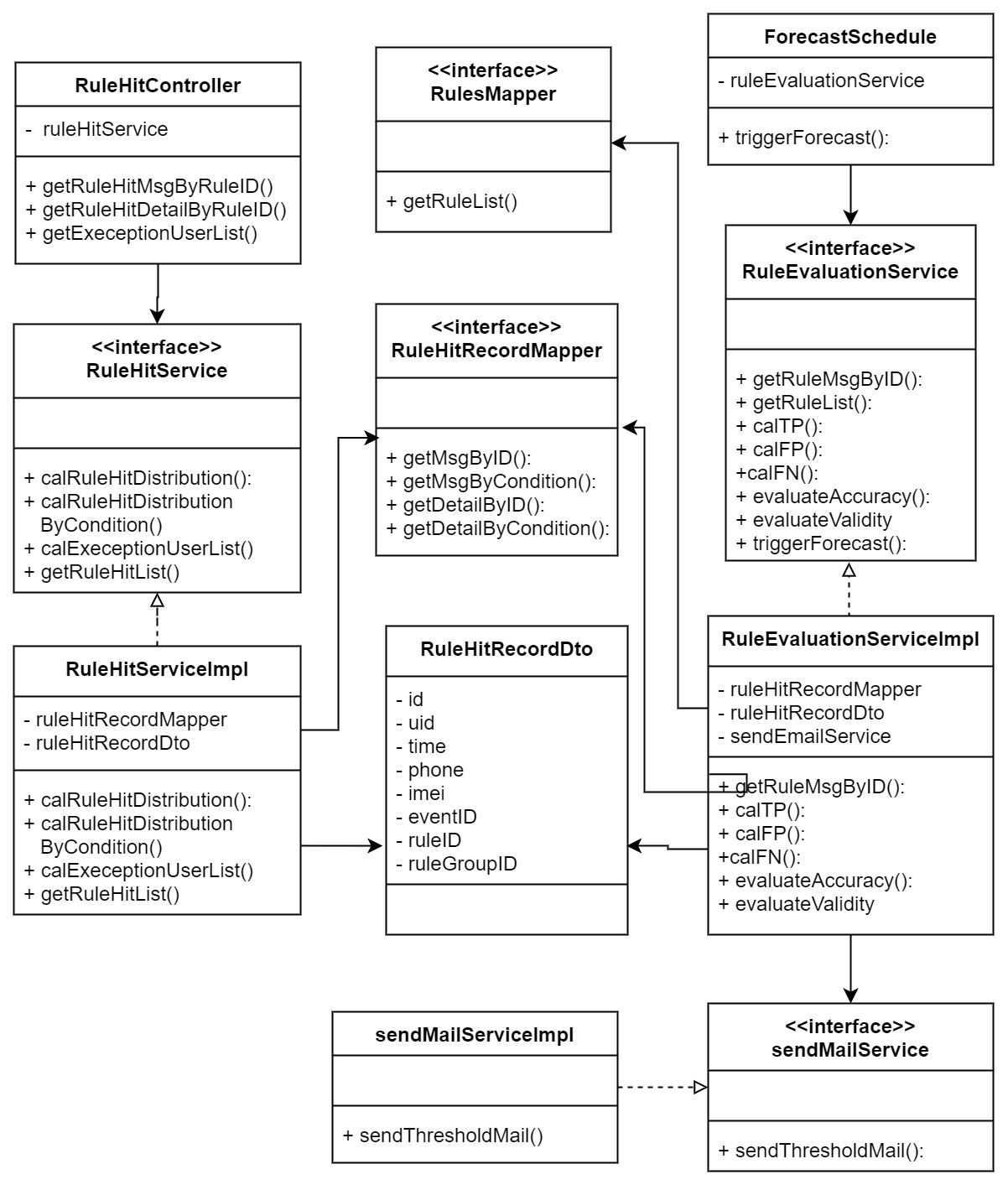


图5-25 规则评价类图

规则准确性和有效性评估部分时序图如图5-25所示。ForecastSchedule类调用RuleEvaluationService类的triggerForecast方法开启定时任务，RuleEvaluationService类调用自身getRuleList方法获得规则列表，其中访问数据库的操作，通过调用RuleMapper类中的getRuleList方法实现，循环计算规则列表中所有规则的有效性和准确性，通过RuleHitRecordMapper类访问数据库可以获得规则命中记录。若准确率和有效性低于80%，利用SendMailService类的sendThresholdMail方法发送告警邮件，其中风控策略师的邮箱通过参数传入。

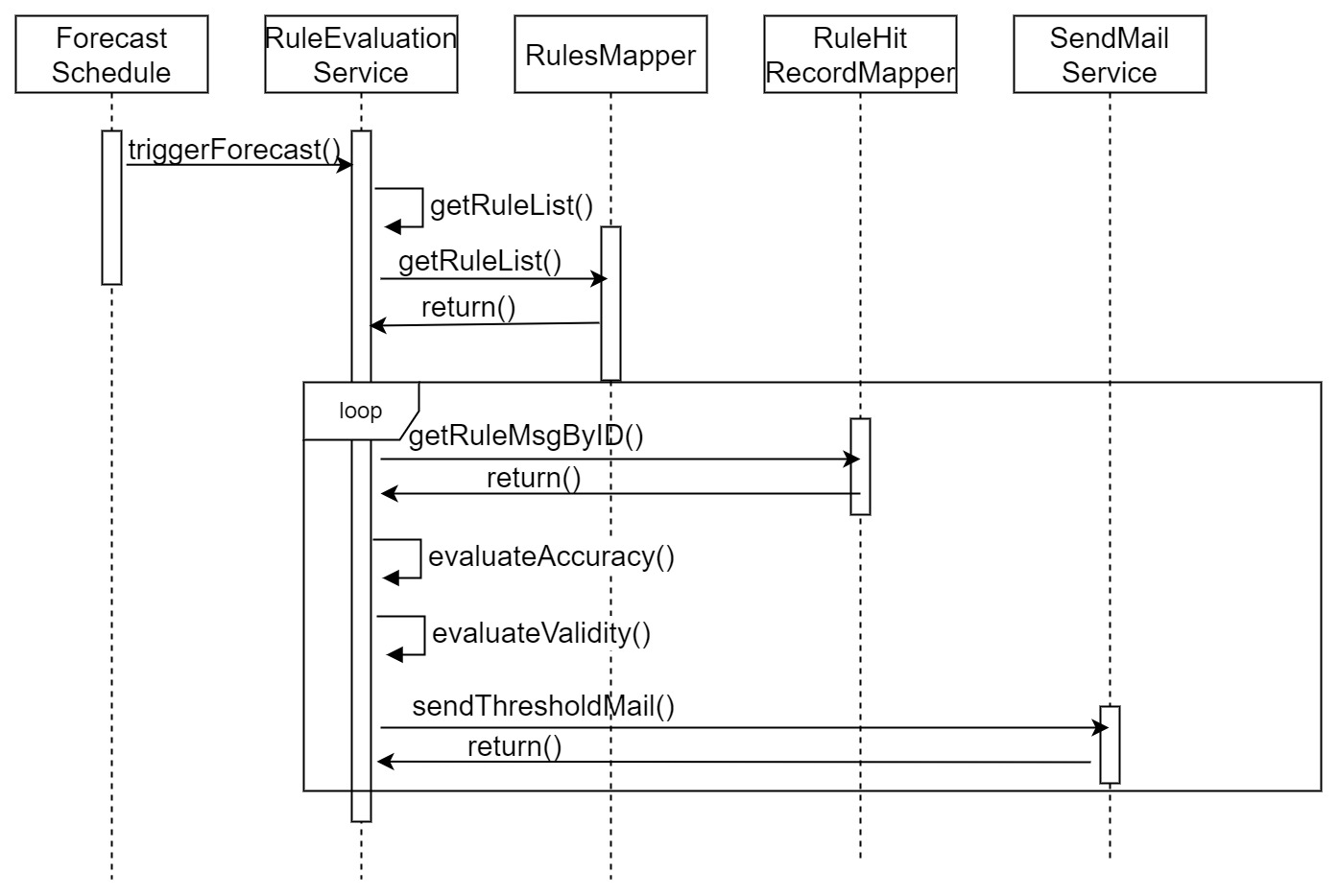


图5-26 规则准确性和有效性评估时序图

规则评价实现效果如图5-27和5-28所示。其中敏感信息已采用测试数据或使用掩码进行遮挡。图5-27展示了登陆事件两个规则组中各个规则命中情况的饼状图，以及规则组的准确率和有效性，图表采用Echart组件绘制，图5-28展示了规则命中具体列表。

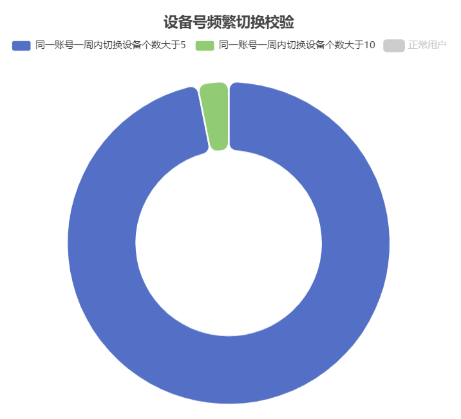
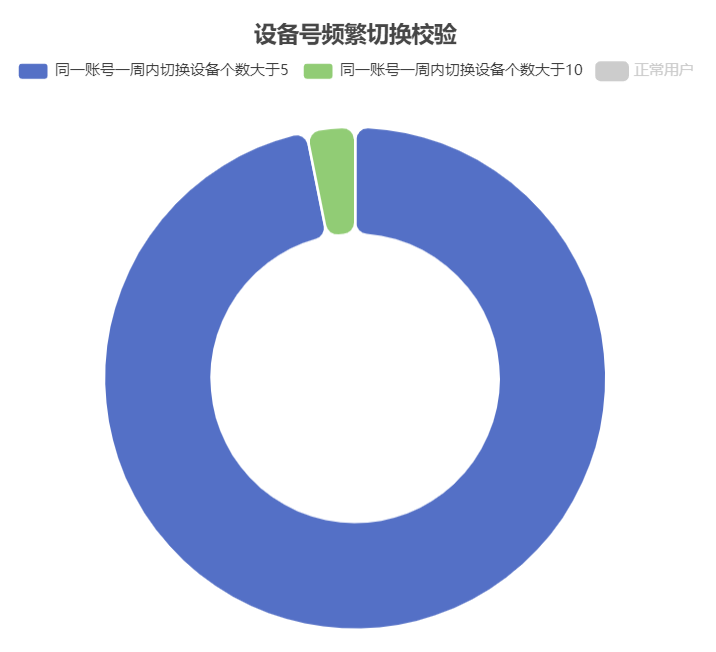
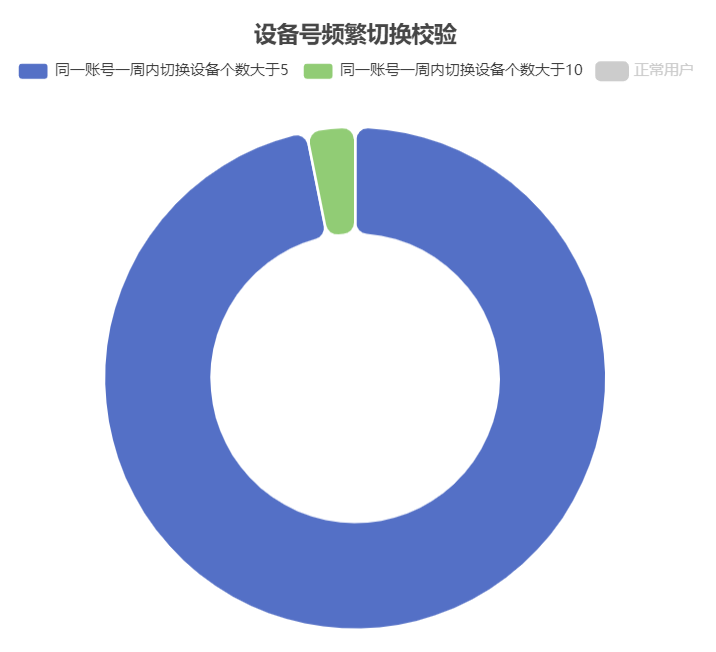
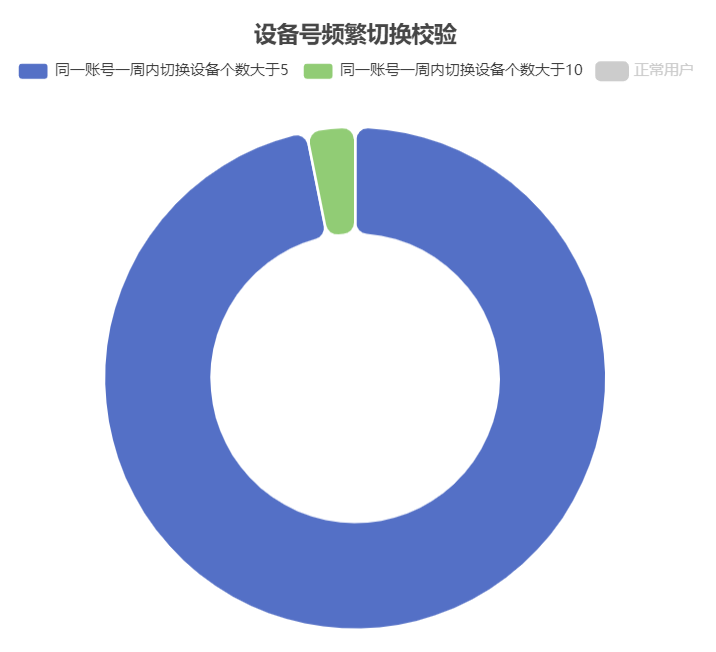
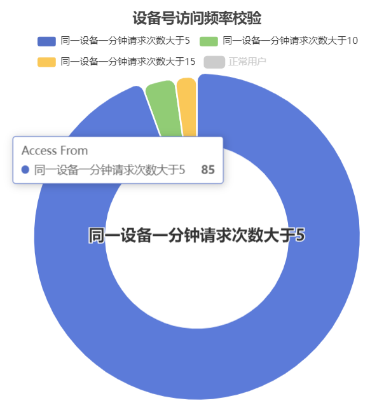
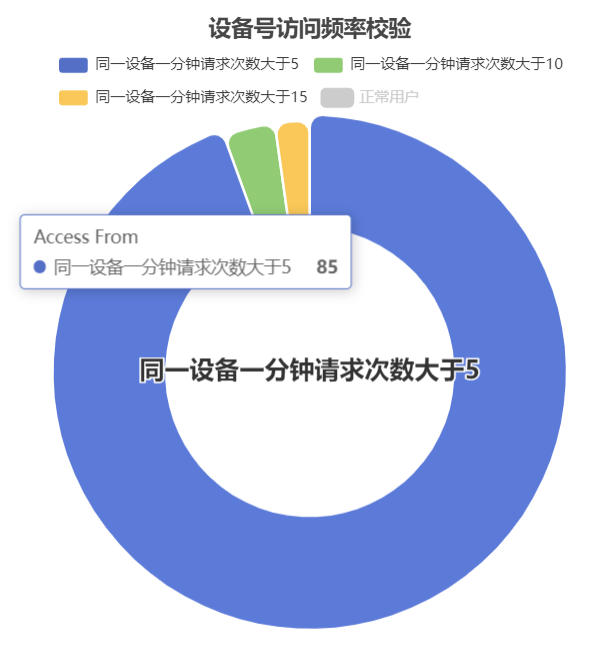
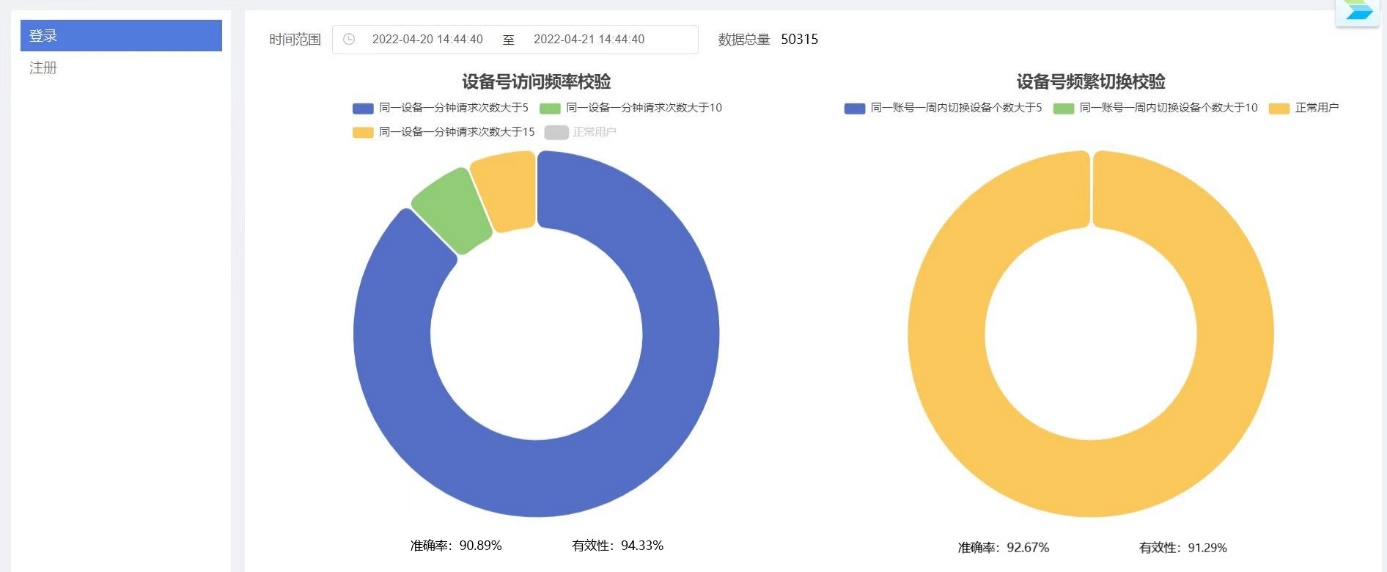
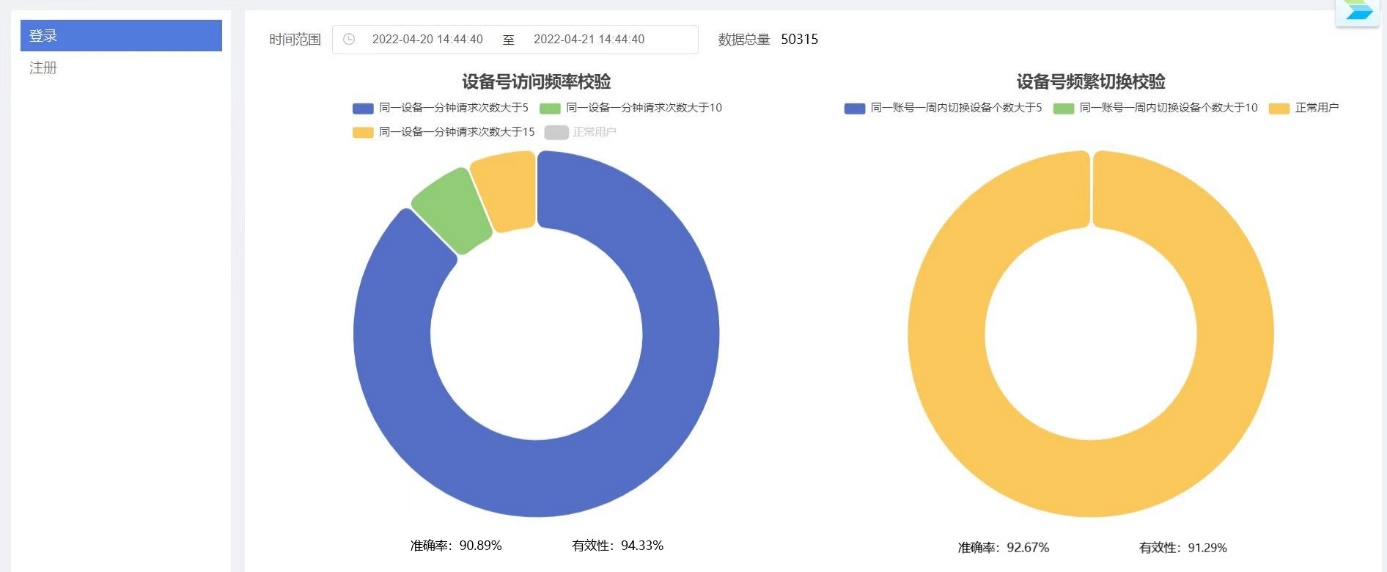
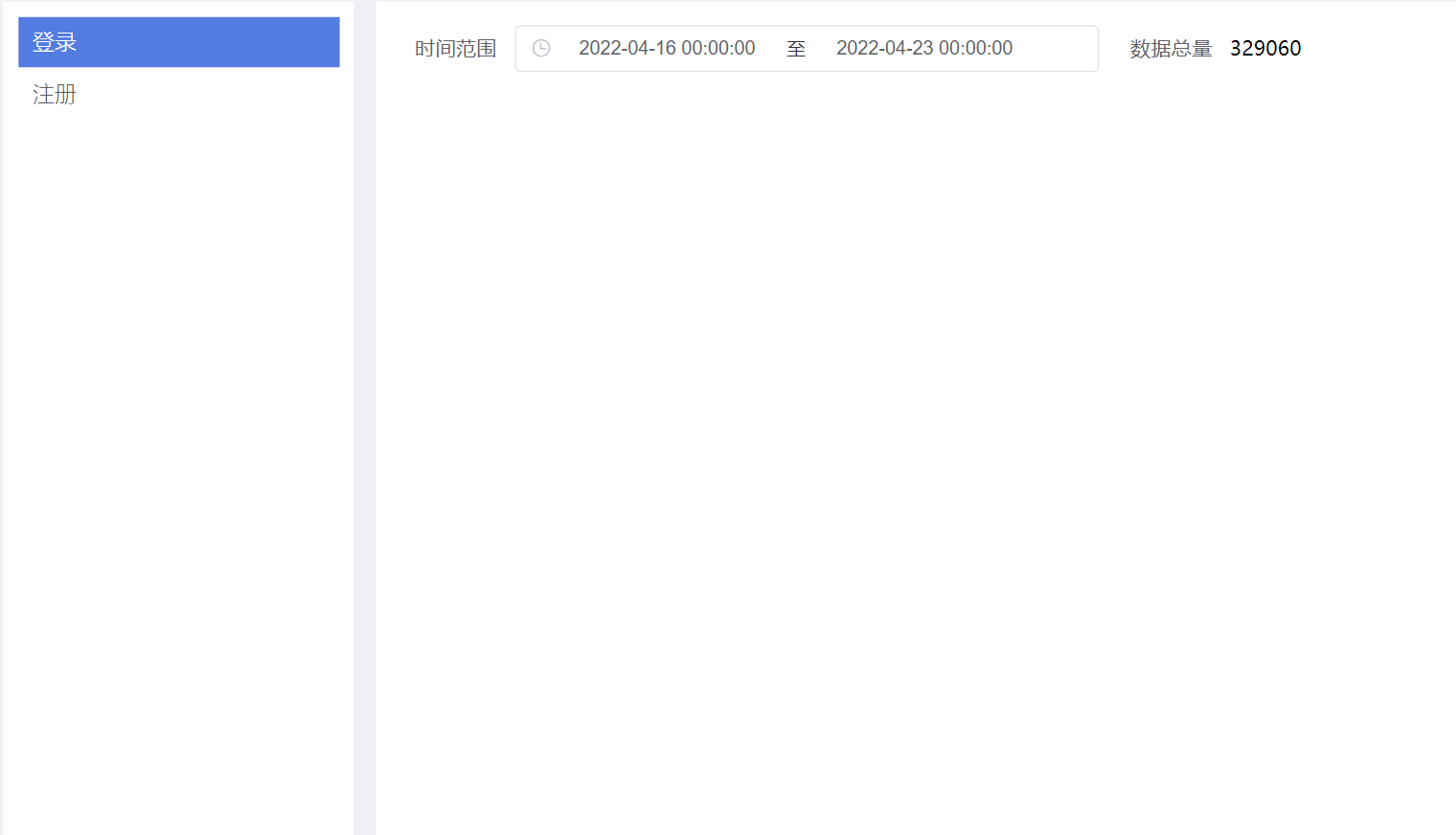


图5-27 规则命中率查看实现效果



图5-28 规则命中详情查看实现效果

5.5 本章小结

本章对系统敏感数据审计、风险情况分析、规则引擎构建和规则执行监控四大模块的具体设计与实现方法进行了阐述，先使用流程图描述了模块的具体业务逻辑，之后使用类图展示了各个类之间的关系，并使用时序图展示了功能实现的具体流程。最后展示了系统最终实现的界面效果。

6 系统测试

本章节中，以第三章需求分析为基础，对租赁业务风险控制安全中台进行功能性和非功能性。功能性测试主要针对11个用例进行，非功能性测试主要针对系统的正确性和响应时间进行测试。

6.1 测试目标及准备

本系统的测试主要分为两方面，功能测试和非功能测试。功能测试主要验证系统是否满足提出的需求，所有交互是否按照预期结果进行响应，以及响应内容是否正确。非功能测试针对系统的响应时间开展。

测试环境所需的硬件与软件版本信息如表6-1所示。

表6-1 系统测试环境

|  |  |
| --- | --- |
| 服务 | 测试环境 |
| 硬件详情 | CPU：i7 8550U，内存：8G |
| 数据库服务 | MySQL：5.7，Redis：5.0.6 |
| 软件详情 | Flink：1.9.2，Kafka：2.4.0 |
| 操作系统 | Windows 10 |

6.2 功能测试

功能测试按照第三章中用例描述划分的功能模块，使用黑盒测试的方法，按照用例描述中的使用流程进行测试，通过比较预期结果与实际结果，判断测试是否通过。

表6-2 配置告警规则测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC1 | **用例名称** | 配置告警规则 |
| **测试日期** | 2022-3-10 | **优先级** | 高 |
| **测试目的** | 是否可以查看、配置、修改告警规则 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 点击导航栏“告警-告警规则” 2. 点击下方“添加”按钮 3. 从职位名称下拉框中进行选择，并输入时间范围、一级和二级阈值，点击保存。其中，输入的时间范围不为正一位小数，一级阈值小于二级阈值 4. 修改规则，输入正确的时间和阈值 5. 点击某一规则右侧“删除”按钮 | | |

续表6-2 配置告警规则测试用例

|  |  |
| --- | --- |
| **测试流程** | 1. 修改任意规则时间范围、一级和二级阈值，点击保存按钮 |
| **预期结果** | 1. 界面上显示目前所有告警规则，包括职位名称、告警时间范围、一级告警阈值、二级告警阈值 2. 界面新增一行空规则，包含4个待输入选项：职位名称、时间范围、一级阈值、二级阈值 3. 系统提示输入错误 4. 系统提示保存成功，查看数据库规则已新增 5. 系统提示保存成功，查看数据库规则已删除 6. 系统提示保存成功，查看数据库规则已更新 |
| **实际结果** | 测试结果符合预期，测试用例通过 |

表6-3 查看全部告警信息测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC2 | **用例名称** | 查看全部告警信息 |
| **测试日期** | 2022-3-10 | **优先级** | 高 |
| **测试目的** | 违背规则时是否触发告警，并且告警内容是否正确 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 为职务编码为101071的员工配置告警规则，时间范围为1小时，一级阈值10，二级阈值5 2. 使用该员工账号登陆某系统，点击某敏感字段5次，记录点击时间、点击字段 3. 进入系统，点击“告警记录” 4. 点击记录右侧“查看详情”按钮 5. 使用该员工账号登陆某系统，再点击某敏感字段5次，记录点击时间、点击字段 | | |
| **预期结果** | 1. 系统显示保存成功 2. 5分钟后，触发二级告警 3. 界面上显示所有告警记录，告警记录具体内容包括时间、触发告警员工姓名、工号、所属部门、触发告警级别、规则时间内敏感数据点击次数显示正确 4. 系统跳转至敏感数据点击日志查看界面，显示该员工全部敏感数据点击日志 5. 5分钟后，触发一级告警 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-4 查看全部日志测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC3 | **用例名称** | 查看全部日志 |
| **测试日期** | 2022-3-10 | **优先级** | 高 |
| **测试目的** | 点击敏感数据是否发送日志，且日志是否正确 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 使用职务编码为101071的员工账号登陆某系统，点击某敏感字段 2. 进入系统，点击“告警日志” 3. 在界面上方筛选框内，根据工号进行筛选，点击“搜索” | | |
| **预期结果** | 1. 界面上显示全部日志信息，按照时间由近到远排序，显示内容包括时间、员工姓名、工号、所属部门、点击系统、字段名称、字段描述 2. 界面显示该员工的点击日志，内容正确 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-5 平台整体风险情况查看测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC4 | **用例名称** | 平台整体风险情况查看 |
| **测试日期** | 2022-3-30 | **优先级** | 高 |
| **测试目的** | 平台整体风险情况展示是否正确 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 用户点击导航栏“风险总览” 2. 用户将鼠标移动至饼状图某个区域上 3. 用户点击平台总体风险饼状图 4. 用户按照风险总分、各业务线风险分、风险级别进行筛选排序 5. 点击用户id字段 6. 用户返回风险总览界面，点击任意业务线风险饼状图 | | |
| **预期结果** | 1. 界面上利用饼状图显示平台内用户风险总分分布情况，以及各业务线用户风险分分布情况 2. 显示该分数该区域内用户总人数及占总用户人数百分比 3. 界面跳转至平台总体风险详情界面，界面上方使用折线图显示各分数段用户人数，界面下方显示全部用户分数详情，包括用户id、最后一次登陆时间、风险总分、各业务线风险分、用户标签、用户风险级别 4. 界面按照筛选条件显示对应内容 5. 界面跳转至用户操作详情界面 6. 界面跳转至该业务线风险详情界面，界面上方使用折线图显示各分数段用户人数，界面下方显示全部用户分数详情，包括用户id、最后一次登陆时间、风险总分、该业务线风险分、用户标签、用户风险级别 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-6 用户具体操作查看测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC5 | **用例名称** | 用户具体操作查看 |
| **测试日期** | 2022-3-30 | **优先级** | 高 |
| **测试目的** | 用户操作是否展示正确 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 使用平台某账号进行一系列操作 2. 点击导航栏“用户操作详情” 3. 输入账号的手机号，点击搜索 | | |
| **预期结果** | 1. 跳转至操作详情查看界面，顶端显示用户id和手机号输入框 2. 界面上显示该用户详情，上方显示近一周用户各项风险分数变化情况，下方显示用户所有操作，包括时间、触发事件名称、操作所属业务线、操作触发规则名称、操作前后分数变化详情 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-7 黑白名单库管理测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC6 | **用例名称** | 黑白名单库管理 |
| **测试日期** | 2022-3-30 | **优先级** | 高 |
| **测试目的** | 黑白名单库管理部分功能是否正确 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 点击导航栏“黑白名单库-黑名单” 2. 用户点击“添加”按钮，在弹窗中选择要添加的表名，填写具体内容和添加理由，点击“确定” 3. 用户点击某一黑名单右侧“修改”按钮，对加入原因进行修改 4. 用户点击某一黑名单右侧“删除”按钮 | | |
| **预期结果** | 1. 界面上显示黑名单库中四张表的全部内容，四张表分别为uid表、IP表、手机号表、设备号表，内容包括账号、加入时间、加入原因、最后修改人、修改时间 2. 界面上显示新添加黑名单，检查数据库已更新，检查添加时间、添加人是否正确。 3. 界面上更新该黑名单原因、修改时间，检查数据库已更新，且最后修改人、最后修改时间均正确， 4. 界面上删除该黑名单，检查数据库已更新 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-8 特征库建立测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC7 | **用例名称** | 特征库建立 |
| **测试日期** | 2022-2-23 | **优先级** | 高 |
| **测试目的** | 检测根据现有机制，是否可以添加特征 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 新建TestFeature类，继承Feature类 2. 设定接收参数、操作符、阈值，重载calFeature方法定义计算方式 3. 将特征名称、特征函数、结果数据类型存入数据库中 | | |
| **预期结果** | 1. 检查数据库，已新增特征 2. 新建规则，在选择特征时可使用该特征 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-9 决策树运行测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC8 | **用例名称** | 决策树运行 |
| **测试日期** | 2022-2-23 | **优先级** | 高 |
| **测试目的** | 检测是否可以配置规则，以及规则是否生效 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 用户点击导航栏“规则引擎-规则建立” 2. 点击“新增规则”，在弹窗中设置规则参数，点击“确认” 3. 用户点击导航栏“规则引擎-规则组建立” 4. 点击“新增规则组”，在弹窗中设置规则组名称、选择组内规则，填写规则组命中逻辑及惩罚措施，点击“确认” 5. 用户点击导航栏“规则引擎-决策树建立” 6. 点击“新增决策树”，在弹窗中设置决策树，点击“确认” 7. 使用某用户账号，做出违反规则操作 | | |
| **预期结果** | 1. 跳转至规则建立界面，显示所有现有规则，包括规则名称、特征、操作符、阈值、惩罚措施 2. 系统更新规则库，并刷新前端界面 3. 跳转至规则组建立界面，显示所有现有规则组 4. 系统更新规则组库，并刷新前端界面 5. 跳转至决策树界面，根据不同事件显示决策树 6. 系统更新规则库，并刷新前端界面 7. 系统执行规则，在规则命中表中增加用户本次操作，并实施配置的防控措施。 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-10 规则评价用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC9 | **用例名称** | 规则评价 |
| **测试日期** | 2022-3-25 | **优先级** | 高 |
| **测试目的** | 检测规则评价机制是否生效，是否可以查看规则命中情况，以及对规则准确率和有效性进行评估。 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 点击导航栏“规则评价-规则命中情况统计” 2. 从菜单中选择事件 3. 点击界面上的任一规则组或任意规则 4. 点击导航栏“规则评价-有效性评估” 5. 从菜单中选择事件 | | |
| **预期结果** | 1. 系统跳转至规则命中情况统计界面，系统左侧菜单显示全部事件 2. 系统界面上利用饼状图显示该事件中所有规则组的命中情况，每个饼状图中显示该规则组内所有规则的命中次数和命中率，以及规则组整体命中率 3. 系统界面上显示该规则组或该规则的全部命中列表 4. 系统跳转至规则有效性评估界面，系统左侧菜单显示全部事件 5. 系统界面上利用饼状图显示该事件中所有规则及规则组的准确率和有效性 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-11 阈值预测用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC10 | **用例名称** | 阈值预测 |
| **测试日期** | 2022-3-25 | **优先级** | 高 |
| **测试目的** | 检测规则阈值是否可以进行自动调整。 | | |
| **测试准备** | 为规则设置初始阈值 | | |
| **测试流程** | 1. 连续3天根据情况对阈值做出调整 2. 第四天凌晨检查阈值是否自动进行调整 3. 第四天检查阈值有效性和合理性是否低于80% 4. 检查风控策略师是否收到邮件 | | |
| **预期结果** | 1. 第四天0：00系统调用算法，对阈值进行预测，检查数据库中阈值已经自动修改为预测结果，并且规则运行时按照新阈值进行判定。 2. 若有效性和合理性低于80%，向风控策略师发送告警 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

表6-12 异常用户统计用例说明

|  |  |  |  |
| --- | --- | --- | --- |
| **用例编号** | TC11 | **用例名称** | 异常用户统计 |
| **测试日期** | 2022-3-25 | **优先级** | 中 |
| **测试目的** | 检测系统是否可以展示异常用户名单，并可查看账号及设备规则命中情况 | | |
| **测试准备** | 用户进入系统 | | |
| **测试流程** | 1. 点击导航栏“规则评价-异常用户统计” 2. 点击账号名称 3. 点击规则命中次数 | | |
| **预期结果** | 1. 系统跳转至异常用户统计界面，显示所有异常设备及异常账号列表，包括设备/账号名称、命中规则次数 2. 系统跳转至用户操作统计界面 3. 系统跳转至规则命中列表界面 | | |
| **实际结果** | 测试结果符合预期，测试用例通过 | | |

6.3 非功能测试

非功能测试的目的是保证系统在满足功能性需求基础上，在易用性、正确性及性能方面也满足非功能性需求。由于系统易用性、灵活性不方便进行评估，正确性保障不是本人重点工作内容，因此本次非功能测试主要以性能测试为主，以响应时间为衡量标准来判断是否满足需求。

本次非功能测试主要分为两部分，第一部分是站在本系统使用者角度，对系统的部分数据展示功能进行点击，测试系统响应时间，具体测试结果如下：

表6-13 系统部分数据展示功能响应时间测试结果

|  |  |  |
| --- | --- | --- |
| **界面** | **操作** | **响应时间** |
| 平台整体风险分数查看界面 | 点击某一业务线，展示业务线分数详情 | 84ms |
| 某一业务线分数详情界面 | 设置筛选条件，进行搜索 | 76ms |
| 用户具体操作查看界面 | 输入用户uid进行搜索 | 65ms |
| 黑白名单库管理 | 查看黑名单 | 61ms |
| 黑白名单库管理 | 添加黑名单 | 63ms |
| 规则命中情况统计界面 | 查看某一规则组命中率 | 91ms |
| 规则命中情况统计界面 | 查看某一规则命中列表 | 82ms |
| 异常用户统计界面 | 查看异常账号及设备名单 | 79ms |
| 异常用户统计界面 | 查看某一账号操作列表 | 66ms |
| 敏感数据审计告警查看界面 | 查看敏感数据告警记录 | 71ms |
| 敏感数据审计日志查看界面 | 查看敏感数据点击日志 | 69ms |

第二部分是站在平台用户角度，测试风控系统的加入对于平台用户操作的处理时延影响情况。本次测试主要关注从系统接收用户操作，到所有防控措施实施完成的时间。具体测试结果如下：

表6-14 系统部分数据展示功能响应时间测试结果

|  |  |  |  |
| --- | --- | --- | --- |
| **事件** | **用户** | **是否触发规则** | **响应时间** |
| 注册 | 用户1 | 否 | 118ms |
| 注册 | 用户1 | 是 | 130ms |
| 注册 | 用户2 | 是 | 127ms |
| 注册 | 用户2 | 是 | 128ms |
| 登陆 | 用户1 | 否 | 102ms |
| 登陆 | 用户1 | 否 | 114ms |
| 登陆 | 用户1 | 是 | 167ms |
| 登陆 | 用户2 | 否 | 108ms |
| 登陆 | 用户2 | 是 | 124ms |
| 登陆 | 用户2 | 是 | 129ms |

测试结果表明，系统响应时间符合要求，非功能测试通过。

6.4 本章小结

本章主要对系统进行了功能测试和非功能测试，首先介绍了测试目标以及测试使用的硬件、软件版本信息，之后对于第三章中需求分析获得的11个用例依次进行黑盒测试，最后从系统使用者角度及平台用户角度，对系统性能进行了测试。功能测试和非功能测试均全部通过。

7 总结与展望

本章主要对项目的完整开发周期、研究意义及论文整体工作进行总结，同时结合企业风控实际应用，分析目前系统设计及实现过程中存在的不足，提出改进方案，并对系统未来发展方向进行展望。

7.1 全文总结

首先，本文介绍了风控系统产生的背景，并以租赁业务为例，分析了公司业务特点及易被黑产攻击的风险点所在，以及公司现有风控系统存在的问题，以此本系统的意义，以公司业务为基础，通过建立公司范围的风险控制体系，设定一系列风险控制策略和规则，在黑灰产攻击平台时迅速识别，进行拦截或者反攻击，从而保障平台安全。

接下来，从用户角度出发，分析该系统的使用者及其原始需求，将用户需求进行提炼整合，将系统按照业务逻辑拆分为4个模块：敏感数据审计、风险情况分析、规则引擎构建、规则执行监控，并针对四个模块进行了需求细化与用例描述。同时对系统非功能性需求进行了分析。

然后，学习了系统搭建过程中涉及到的技术与理论，包括SpringBoot框架、Redis缓存机制、Kafka、Flink流式计算引擎、三次指数平滑法、变异系数法以及规则引擎相关理论，并对系统进行架构设计，按照微服务架构对系统服务进行拆分。同时完成了数据库的设计，梳理各个实体之间的关系，并设计表结构。

在概要设计基础上，对4大模块分别进行详细设计，按照SpringBoot框架和RPC通信方式，将类分为实体类、前端控制器类（Controller）、数据服务层及实现类（Service和ServiceImpl）、数据库操作类（Mapper）、数据传输对象类（Dto）、工具类（Util）、常量接口类（Constant）及相关配置信息（Config），用类图说明类与类之间的关系，并用时序图阐述类之间的交互机制。

最后，根据详细设计对系统进行编码实现，并以需求分析为依据，对系统进行功能测试和非功能测试。

该系统的实现，不仅是帮助企业构建了专属的、可定制化的业务风控系统，更是保障了平台内上千万用户的信息与财产安全。

7.2 系统展望

目前该系统已经完成一期内容，交由公司上线使用。但是在系统的设计与实现过程中，依然存在可以改进的地方，在二期中可以对系统加以完善。

首先，在规则引擎构建方面，目前仅仅是基于配置化规则进行风险防控，虽然可以借助算法对规则阈值进行调整，但在系统应用过程中，依旧需要耗费较大人力成本用于规则调整。在后续发展中，可以加入模型、评分卡等机制，或通过机器学习的相关方法丰富规则引擎的防控机制，提高防控效果，且减少人力成本。

其次，是在规则执行方面。目前的规则状态仅包括生效和未生效两种，缺少灰度机制，即在规则生效之前，应存在试运行阶段，使用部分真实用户数据对规则有效性和准确性进行评估，评估通过后再正式生效。灰度机制可减少规则正式执行过程中造成的错杀或漏杀情况。

最后，风控系统永远不是一成不变的。随着业务的发展以及黑灰产攻击手段的更改，风控场景、规则、防控措施都应随之做出调整，以适应新的应用场景。风控之路任重而道远，但随着技术发展，风控系统也将日益完善，最终在企业和用户与不法分子之间构建起严密的安全防线。

参考文献

[1] [Convolutional Neural Networks (LeNet) - DeepLearning 0.1 documentation](http://deeplearning.net/tutorial/lenet.html). DeepLearning 0.1. LISA Lab. [31 August 2013].

[2] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. [Imagenet classification with deep convolutional neural networks](https://web.archive.org/web/20150216025624/http://papers.nips.cc/paper/4824-imagenet). Advances in Neural Information Processing Systems. 2012, 1: 1097–1105 [2016-11-20].

# [3]RepVGG: Making VGG-style ConvNets Great Again

# [4] Very Deep Convolutional Networks for Large-Scale Image Recognition

[5] [Benchmark analysis of representative deep neural network architectures](https://ieeexplore.ieee.org/abstract/document/8506339/)

[S Bianco](https://scholar.google.com/citations?user=P08LSD0AAAAJ&hl=zh-CN&oi=sra), [R Cadene](https://scholar.google.com/citations?user=U05sSWsAAAAJ&hl=zh-CN&oi=sra), [L Celona](https://scholar.google.com/citations?user=F9vDCKAAAAAJ&hl=zh-CN&oi=sra), [P Napoletano](https://scholar.google.com/citations?user=-1Va_uwAAAAJ&hl=zh-CN&oi=sra) - IEEE access, 2018 - ieeexplore.ieee.org

[6][Regression Loss Functions All Machine Learners Should Know](https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0). Heartbeat. 2018-06-05 [2018-10-23].

致 谢

在本篇论文完成的同时，我要感谢我的导师王方石老师，在毕业论文的全过程中对我悉心指导，从选题到开题、中期检查，再到最后的论文撰写都提出了很多建设性的指导意见。通过老师的指导，我不仅在技术方面有所提高，更学会怎样的表达方式更能准确的向读者传达我的想法，在此向导师表示衷心的感谢。

其次我要感谢我的父母一直以来对我的照顾、包容、理解、支持，因为有他们的存在，我才可以无后顾之忧地做自己想做的事，将最真实的自己展现给这个世界。

我还要感谢四年来对我提供帮助的老师们，他们教会我知识，教会我表达，教会我为人处世。

我要感谢我实习单位的同事们，感谢他们包容第一次正式迈向社会的我的无知，包容我犯下的小错误，包容我因为冬奥和疫情导致的长时间远程办公，并且无私的向我提供帮助。因为他们的存在，让我的实习期充实、快乐、有所收获。

我要感谢我的朋友们，感谢你们长久以来对我的信任、理解、陪伴，我会珍惜和你们每个人的情谊，愿友谊长存。

最后我要感谢我自己，感谢我4年来的努力，感谢我低谷时的坚持，感谢我得意时的冷静。

本科毕业，是一段旅程的结束，也是下一段旅程的开始。我会带着所有人的爱与鼓励，奔赴下一场山海。长路漫漫，愿你我在未来的某一天，顶峰相见。