**7OxCloud**

**—少年先疯队**

### 项目分工

|  |  |
| --- | --- |
| 团队成员 | 主要贡献 |
| 王鑫耀（队长） | 后端代码编写，与推荐系统交互 |
| 涂国彬 | 前端代码编写，demo视频录制 |
| 郑嘉睦 | 推荐系统实现及相关模型训练 |

### 架构设计

### 

本项目后端采用分布式微服务架构，用Gin框架实现API网关，完成HTTP请求的转发，在请求到达网关前，统一由Nginx进行反向代理，网关集群分布在8080，8081等端口，分别赋予对应的权重，从而用Nginx实现了负载均衡。到达网关后，使用JWT进行鉴权，拦截非法的请求。用GRPC框架实现微服务集群，采用Consul进行服务的注册，来自网关的路由请求到达后，会通过Consul进行服务发现，由于相同的微服务可能会开启多个，在进行RPC远程调用前又会进行一次负载均衡（轮询），进而远程调用相应的微服务，相应的微服务进行具体的业务逻辑处理，然后返回响应给网关，最后到达前端web界面。共有四个微服务，也可以算五个。前四个分别为用户微服务、视频微服务、互动微服务以及社交微服务。可参照接口文档：

<https://apifox.com/apidoc/shared-ddccc651-caf3-4d78-b296-eef873d9a6cd>

用户微服务负责用户的登录注册、个人信息修改等等。

视频微服务负责视频流相关接口，比如首页视频流，分类视频流、热点视频流，推荐视频流、话题视频流、搜索视频流、历史视频流以及视频发布及删除等等。

互动微服务负责点赞、收藏、评论等互动操作。

社交微服务负责关注、粉丝等相关逻辑。

第五个微服务也可以算作“调度微服务”，使用Cron来开启定时任务，例如每隔一小时会将用户访问的日志（包含浏览、点赞、收藏视频）通过RPC远程调用传给推荐系统，供其分析用户的行为特征及偏好。同时该微服务会定期清理缓存，由于用户观看视频有历史记录，并存在了Redis中，为了防止其占用大量内存，所以新建定时任务定期清理。

项目使用了MySQL和Redis两种数据库，MySQL数据库中共有十张表，每张表的设计均符合相关范式的要求，另外Redis使用了两个数据库，0号数据库用来缓存用户短信的验证码，1号数据库用来缓存用户的浏览记录，设置为一个月的过期时间（也就是定时任务开始生效），使用Gorm框架进行与数据库的交互，该框架自带防止SQL注入的功能，在数据库的查询时有做多项优化，例如使用关联查询以及子查询来减少查询次数，同时使用事务来进行对MySQL数据库的操作，提高了查询速率，也保证了数据的一致性。在对Redis的操作中使用了管道，这也减少了与Redis的连接次数，提高了性能。

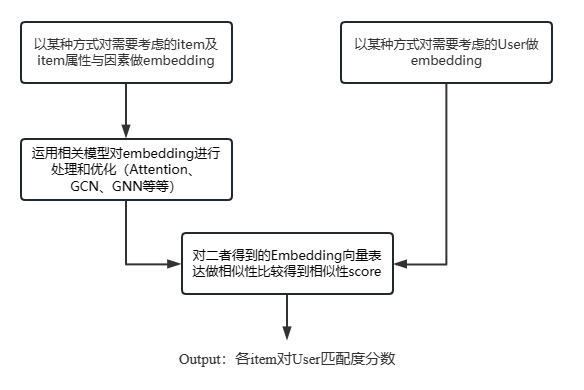
7OxCloud不乏优秀的第三方开源库。比如使用了Zap进行日志的管理，极大的方便了开发人员，同时使用Viper和Nacos来对分布式配置进行管理，通过Viper读取Nacos配置，通过Nacos的UI界面来管理项目各模块的配置，方便高效。本项目也使用了RabbitMQ消息队列来进行异步处理。由于需要记录相关的日志，并将其转发给推荐系统，如果不引入消息队列，用户的体验将欠佳。用户在执行浏览，点赞，收藏视频等通常为高并发的操作时，加入了消息队列，异步记录到CSV表，然后同步到数据库中，实现了流量削峰的效果。

用户上传的视频均放到了七牛云上，视频加载体验良好。在用户注册或者登录，忘记密码时使用阿里云的短信服务来保证整个过程的安全性，对于用户的密码，后端使用了SHA512加密算法，同时引入随机盐值，极大的保证了用户数据的安全性。

本项目使用了Docker安装相应的第三方依赖，比如MySQL、Redis、Nacos、Consul、RabbitMQ等，提高了开发与部署的效率。

7OxCloud的一大亮点是使用了推荐系统来为用户实时推荐感兴趣的视频。后端通过GRPC与推荐系统进行交互，后端部分使用Go编写客户端，推荐系统部分使用Python来编写服务端，调用端口为8976，它们之间主要有两种交互，一是通过定时任务将用户的日志以csv的形式记录再传给推荐系统，推荐系统将会进行模型的训练并迭代更新。二是用户在访问推荐视频流相关接口时，视频微服务会充当客户端将用户的id远程传给推荐系统，推荐系统会根据用户的id和已经实时训练好的模型，返回相应的视频id列表，由于后端在生成视频id的时候使用了雪花算法，来满足高并发场景下大量id的需要，推荐系统会用索引进行相关转换。如果当前应用平台的短视频较少，推荐系统可能无法完全发挥其作用，有时候会返回空列表给后端，后端加以甄别后，如果是空列表，则会从Redis中读取用户的浏览记录，最终从数据库返回用户并未观看的视频，实现了去重处理。

## 推荐系统部分

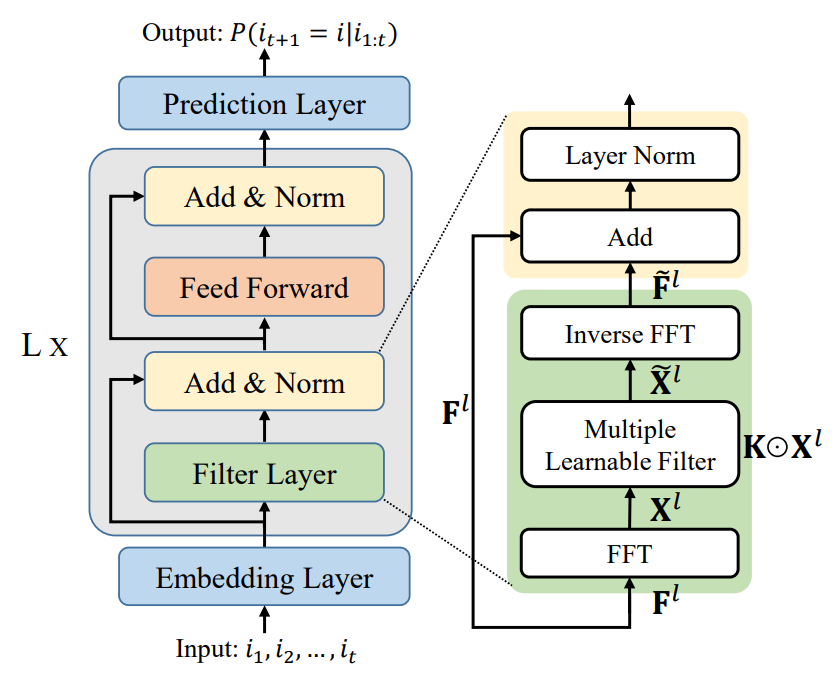


推荐系统进行信息处理的逻辑大致可以抽象为上述步骤，首先定义User与item分别为用户以及要推荐的商品，首先模型要对二者进行嵌入，得到二者在计算机中的唯一表示——Embedding表示，然后将item本身的嵌入以及需要考虑的各种item因素的嵌入比如点击时间、大致的类别等等进行concatenate连接起来，然后运用一些通用的或经过特别设计的模型结构来处理和调整embedding向量，这一步类似于信息聚合和信息过滤，将需要考虑的因素聚合成一个固定长度的embedding向量，同时也需要将其中可能会影响判断的noise给过滤掉。最后大多数推荐系统会通过一个相似性比较的方式来判断一个User和item的“匹配程度”，具体操作就是二者的embedding向量进行一个内积运算，最后会得到一个标量，这

个标量我们把它称为“相似度分数”，然后会选取top-k个分数比较高的item来推荐给User。

总结下来有如下几个特点：

* 对需要匹配的A、B两个对象进行Embedding嵌入
* 对其中有额外因素要考虑的对象的Embedding做一些深度学习上的处理以实现信息聚合和信息过滤。
* 对二者进行相似性比较得到相似性分数



本项目基于2022 WWW顶级学术会议的Filter-enhanced MLP is All You Need for Sequential Recommendation的相关思路，该文章创造性的将信号滤波领域表现卓越的傅里叶变换引入了推荐系统领域，用于对推荐系统模型的embedding表达进行频率域的滤波，深入考察嵌入的隐式表达。我们采用了他们使用基于傅里叶变换进行滤波的模型架构FMLP，将傅里叶变换融合进信息处理中，经过了傅里叶变换（FFT）之后使用一个可学习的W参数矩阵和FFT后的嵌入表达进行element-wise multiplication（元素级乘法）来进行相关元素的滤波，经理论证明该W参数矩阵可以拟合任何目的滤波器。然后经过逆傅里叶变换之后，通过一些Dropout和layerNorm层来缓解梯度消失和训练不稳定的问题。最后会输出Prediction layer会将输出的用户的embedding表达以及我们的重要性标准的embedding嵌入进行相似性计算以得出每个用户的“相似性得分”即重要性得分，最后分数排名靠前的用户我们将认定其为传播重要节点。

离散傅里叶变换（DFT）在数字信号处理领域是非常重要内容，在本项目中我们只需要考虑一维DFT。

给定一个数字序列，其中，一维DFT将序列转换到频域的公式为：

(2-1)

其中，i是虚数单位，对于每个k，DFT创造了一个新的表达作为所有原始输入token和所谓的“扭曲因子”的乘积的积。这样，表示序列在频率时的光谱。注意到DFT是一个一对一的转换，给定DFT，我们可以通过逆DFT（IDFT）来恢复到原始序列：

(2-2)

Filter-enhanced MLP is All You Need for Sequential Recommendation这篇文章系统证明了一个可学习的滤波器矩阵W等价于循环卷积，在FMLP的filter layer中，输入信息即用户的Embedding表达首先通过FFT转换为频域表示，然后进一步和可学习滤波器W进行元素级点乘，这里可学习滤波器W可以视为一组可学习的频域滤波器的乘积，其中d表示模型的hidden-dimension的大小，根据卷积定理，在频域上的乘法等价于在时域上的循环卷积。它是两个具有相同周期的周期函数之间的周期卷积的一种特殊情况。

在此定义为模型的输入矩阵即各用户第L层的Embedding嵌入表达，为一维傅里叶变换（FFT），为一维逆傅里叶变换，对于来自嵌入表达矩阵的第t维特征以及一个滤波器，他们的循环卷积被定义为，循环卷积计算公式为：

(2-3)

其中，mod表示整数模运算，n为序列长度。考虑到序列的DFT结果，我们有如下推导：

(2-4)

这里推导的最右边，事实上就是频域里的第k个数，就是第t个滤波器的第k个数，总的来说，到这里我们就证明了：

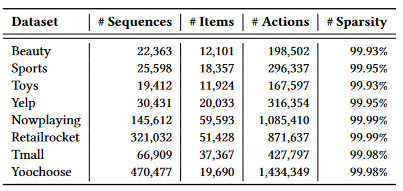
(2-5)

其中 “∗”代表循环卷积，“⊙”代表element-wise multiplication。

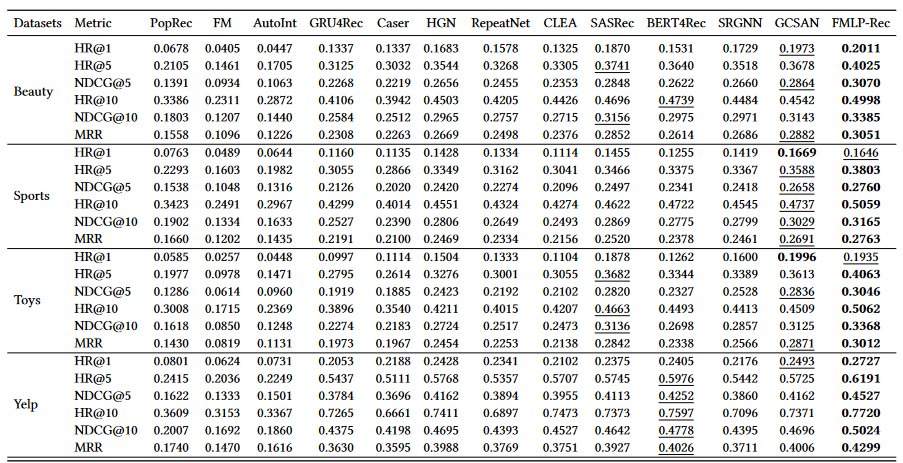
相似度计算一般主要采用余弦相似度：

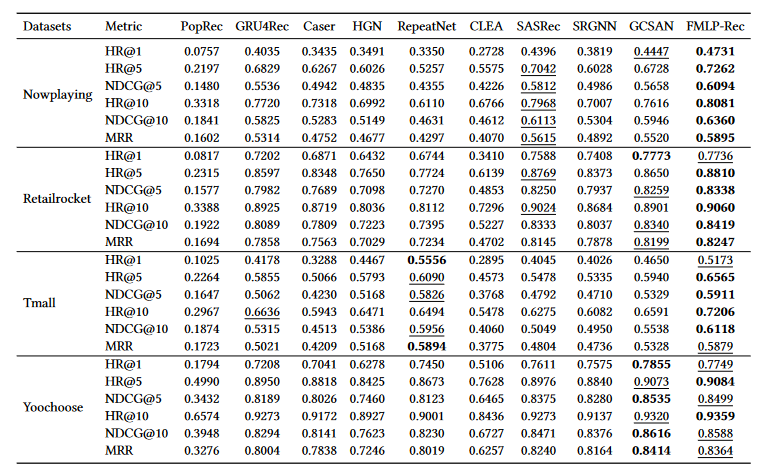
(2-6)

余弦相似度用来衡量用户向量i和j之间的向量夹角大小,夹角越小,证明余弦相似度越大,说明两个用户越相似,若把用户向量换成物品向量,则同理可得物品向量之间的相似度。然后一般在实际项目中，只用i和j两个向量的点乘作为余弦相似度。



上述展现了部分推荐系统领域公认的具有挑战性的数据集，根据论文效果展示以及本地复现，我们得到了以下结果：





可以看出本模型无论在序列推荐数据集还是会话推荐数据集上都具备非常大的性能优势并成为SOTA模型。同时本模型基于MLP架构，模型非常轻量，反向训练更新及前向传播计算的速度都非常快，符合短视频平台及时更新、及时计算的特点。