七牛云 1024 创作节作品

FleetingFlow

不想起名队

2023 年 11 月 7 日

目录

1 引言 **2**

1.1 项目目的 2

1.2 项目分工 3

2 项目实现 **3**

2.0.1 前端技术 3

2.0.2 UI 设计 3

2.1 后端技术 3

2.2 人工智能技术 3

2.3 架构设计 5

2.4 技术选型 5

2.5 数据库设计 5

2.6 业务拆解 6

2.6.1 角色拆解 6

2.6.2 业务模块拆解 6

2.6.3 云存储和七牛视频相关产品 6

2.6.4 人工智能技术 6

2.7 项目代码介绍 12

2.7.1 业务流程介绍 13

2.7.2 前端架构 15

References **17**

1 引言

短视频内容在现代数字时代的崛起引发了广泛的兴趣和需求。用户越来越倾向于通过短视频形式来共享和获取信息、娱乐和沟通。为了满足这一不断增长的需求，我们团队在本次第二届七牛云 1024 创作节中，致力于开发一款创新的 Web 端短视频应用。

本项目主页界面展示于图1中。

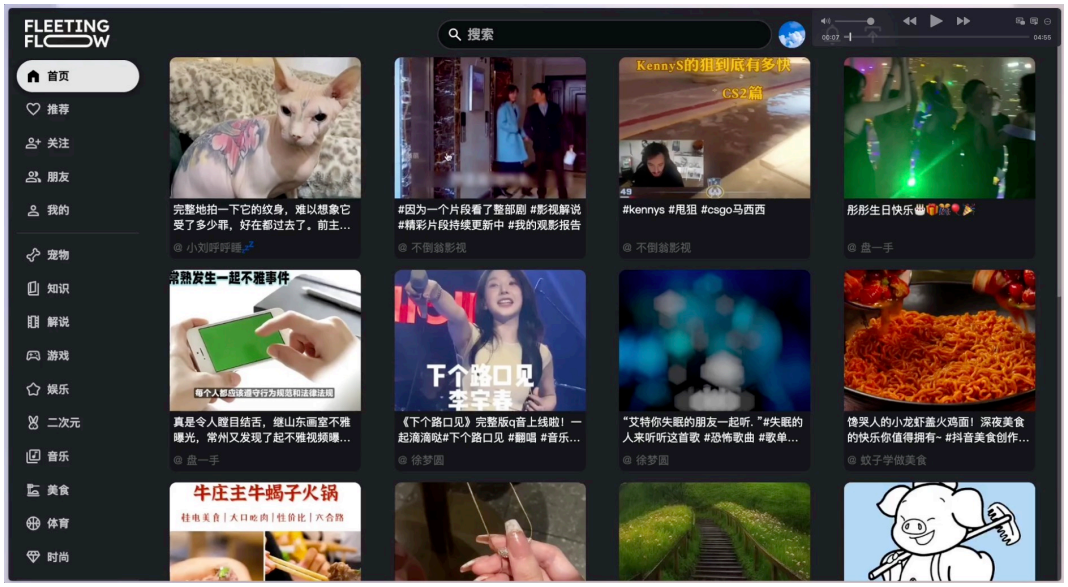

图 1: 本项目主页界面展示图

1.1 项目目的

我们的网页短视频应用旨在为用户提供一个无缝浏览短视频内容的平台。本报告将探讨应用的关键技术和功能，以确保用户在观看视频时获得良好的观看体验：

- 快速加载：利用七牛云存储的多样性视频功能，赋能视频快速转码，确保视频能够快速加载，减少缓冲时间。
- 响应式设计：确保应用在不同设备上都有良好的显示效果。
- 使用推荐算法：根据用户的观看历史和喜好，提供个性化的视频推荐。
- 用户友好的界面：清晰的界面设计和简单的导航，让用户轻松找到他们感兴趣的内容。

1.2 项目分工

团队成员	主要贡献
胡菁菁	队长，负责整个项目后端部分
肖骏锋	负责前端开发
王春实	负责机器学习模型、推荐系统和 Docker 部署

2 项目实现

2.0.1 前端技术

前端框架：使用 React 作为主要框架进行前端开发。React 的灵活性以及编程范式能够很好地辅助我们开发短视频应用，其强大的组件化以及虚拟 DOM、fiber 架构等特点能够实现高性能、高可维护性的 Web 应用。

TypeScript 与 Sass：使用 TypeScript 对项目进行严格的类型规范，并提供强大的类型推导功能，提升了开发的体验与降低了维护的难度。使用 Sass 对样式进行编辑，能够更加灵活、高复用地实现样式和布局，并且项目采用模块化的 Scss，进一步强化了样式的管理。

数据管理：使用 zustand 作为项目的状态管理工具，具有轻量、灵活的特点，能够将数据从组件中解耦并提供 hook 或直接读取状态的方法，保证了前端状态的一致性。

视频解析：项目使用七牛云的 Hls 对视频进行转码，因此前端引入 Hls.js 库以辅助视频解码，实现动态清晰度转换等功能。

2.0.2 UI 设计

2.1 后端技术

2.2 人工智能技术

服务名称	英文名	端口号	服务类型
视频推荐服务	recommender-system	7670	业务服务
标题分类服务	tags-recommender	7671	业务服务
Tags 推荐服务	video-search-service	7672	业务服务

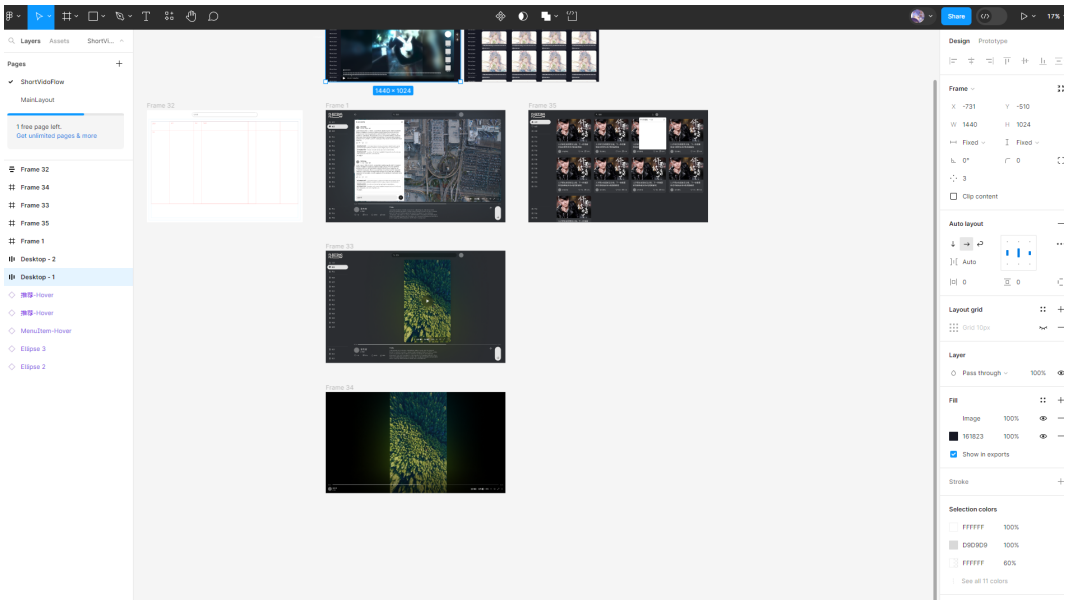

图 2: Figma 用于 UI 设计

服务名称	英文名	端口号	服务类型
数据库	MySQL	3306	环境依赖
缓存	Redis	6379	环境依赖
搜索引擎	ElasticSearch	9200	环境依赖
可视化平台	Kibana	5601	环境依赖
消息队列	RabbitMQ	5672, 15672	环境依赖
注册中心	Nacos	8848	环境依赖
网关服务	gateway-service	4000	业务服务
用户服务	user-service	8000	业务服务
视频服务	video-service	8001	业务服务
评论服务	comment-service	8002	业务服务
消息服务	notification-service	8003	业务服务
搜索服务	video-search-service	8004	业务服务

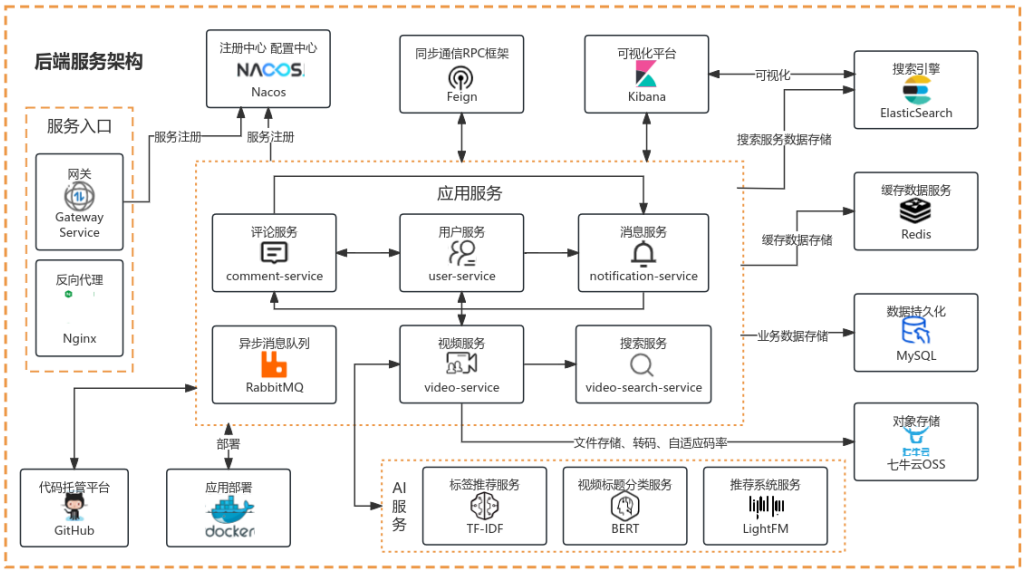

图 3: 系统架构图

2.3 架构设计

2.4 技术选型

2.5 数据库设计

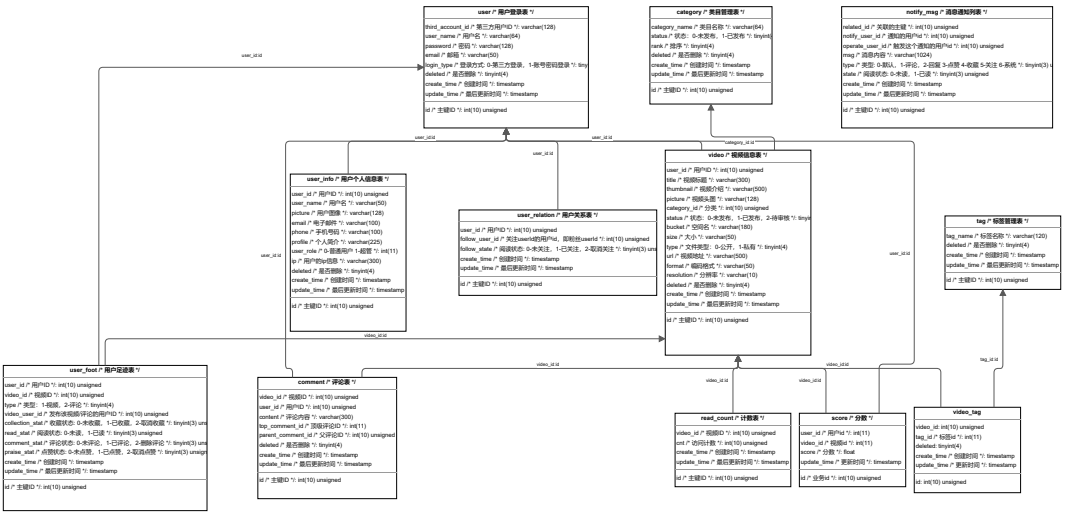

图 4: 数据库设计结构图（请放大查看）

2.6 业务拆解

2.6.1 角色拆解

2.6.2 业务模块拆解

对于整个系统设计，我们按照系统业务边界大致将系统分为以下五个模块，然后再针对以上五个模块进行简单的细化拆分：

图 5: 业务拆解图

2.6.3 云存储和七牛视频相关产品

七牛云存储：我们将使用七牛云存储来存储用户上传的视频内容、图片以及其他媒体文件。七牛的高可用性和 CDN 加速将确保视频内容能够快速加载，提供卓越的观看体验。

七牛视频相关产品：我们将集成七牛的视频相关产品，如视频截帧、视频转码等，以确保视频内容的兼容性和性能最佳化。

2.6.4 人工智能技术

人工智能（AI）无疑是当前科技领域中最火热、最引人注目的领域之一。它的热度在过去几年里不断增长。

AI 技术带来的优势是显而易见的，具体可以概述为下面几个方面：广泛的应用领域、人机交互、数据驱动决策和自动化效率提升。同样，AI 技术对短视频应用的内容生成、推荐、分类和审核等方面产生了深远的影响。它提

高了用户体验，丰富了内容创作工具，也提供了更多机会来创造有趣、引人入胜的短视频。

综上，我们将较为前沿的 AI 模型应用于本次挑战赛项目，如 Bidirectional Encoder Representations from Transformers[1]（BERT）和基于机器学习技术的推荐系统 LightFM[2]，具体介绍如下。

BERT 文本分类模型 本节代码位于

https://github.com/qing-wq/FleetingFlow/tree/master/machine-learning/title-classification

BERT 简介

BERT 是由 Google 开发的自然语言处理模型，可学习文本的双向表示，显著提升在情境中理解许多不同任务中的无标记文本的能力 [3]。

双向模型在自然语言处理 (NLP) 领域早已有应用。这些模型涉及从左到右以及从右到左两种文本查看顺序。BERT 的创新之处在于借助 Transformer[4] 学习双向表示，Transformer 是一种深度学习组件，不同于递归神经网络 (RNN) 对顺序的依赖性，它能够并行处理整个序列。因此可以分析规模更大的数据集，并加快模型训练速度。Transformer 能够使用注意力机制收集词语相关情境的信息，并以表示该情境的丰富向量进行编码，从而同时处理（而非单独处理）与句中所有其他词语相关的词语。该模型能够学习如何从句段中的每个其他词语衍生出给定词语的含义。

由于绝大多数 BERT 参数专门用于创建高质量情境化词嵌入，因此该框架非常适用于迁移学习。通过使用语言建模等自我监督任务（不需要人工标注的任务）训练 BERT，可以利用 WikiText 和 BookCorpus 等大型无标记数据集，这些数据集包含超过 33 亿个词语。要学习其他任务（如问答），可以使用适合相应任务的内容替换并微调最后一层。

图6中的箭头表示三个不同 NLP 模型中从一层到下一层的信息流。

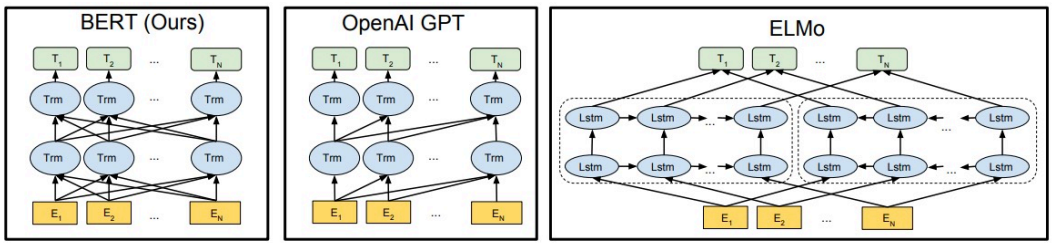

图 6: 预训练模型架构的差异。

BERT 可以针对许多 NLP 任务进行微调。它是翻译、问答、情感分析和句子分类等语言理解任务的理想之选。

短视频分类数据集获取

为了获取训练 BERT 的文本数据，我们使用爬虫技术，针对抖音短视频平台进行搜索，采集了宠物、知识、解说、游戏、娱乐、二次元、音乐、美食、体育、时尚这 10 大数据种类的 808 条数据。

其中数据的分布情况如图7所示

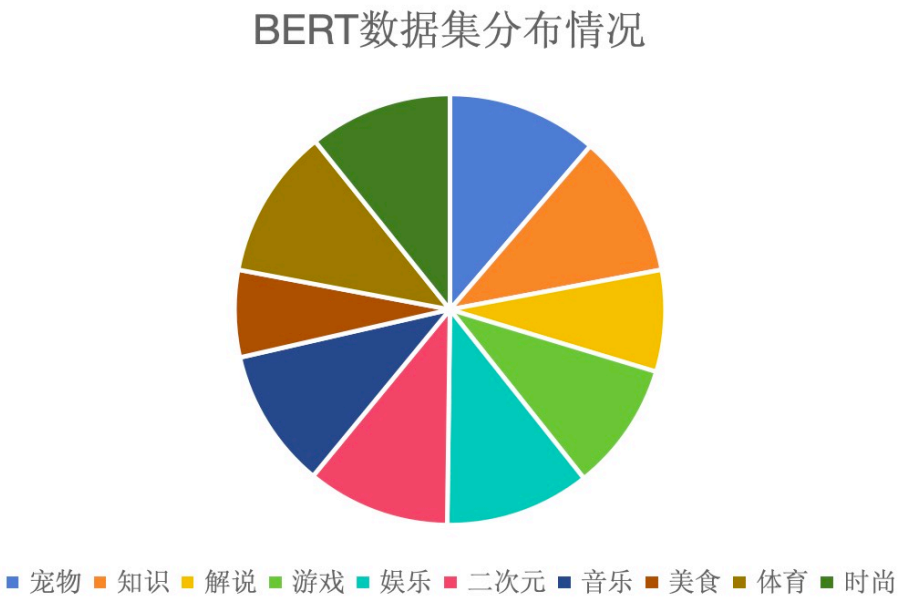

图 7: 采集的数据集中数据分布情况

数据集以 CSV 的格式进行存储，表2展示了部分数据集的情况。

BERT 模型微调

使用以下步骤对数据集进行预处理：

- 分词：将文本分割成标记，通常是单词或子词。分词有助于将文本转化为模型可接受的输入。
- 标记化：将标记映射到 BERT 的词汇表中，以获得每个标记的唯一 ID。这创建了输入特征，通常是 Tokenized 文本的 ID 序列。
- 添加特殊标记：每个样本的开头和结尾通常会添加特殊标记，如”[CLS]”和”[SEP]”。”[CLS]” 用于表示分类任务的开始，”[SEP]” 用于标识句子的结束。

之后，我们构建 BERT 模型进行微调，我们使用预训练的 BERT 模型作为基本模型。可以选择不同大小的 BERT 模型，具体取决于任务和计算资源，因为我们的任务是文本分类，任务难度较低且短视频应用用户量大，对模型的速度有较高的要求，所以我们采用了较小的 BERT 模型进行微调。模型预训练权重来自 HuggingFace：https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-chinese.tar.gz

在模型微调阶段，使用数据加载器将小批次的数据输入到 BERT 模型中。然后计算损失，并通过反向传播和优化算法来更新模型的权重。通常，需要多个训练周期来训练模型，同时要监控损失的下降。我们使用 RTX3090 GPU 进行微调，训练 12 个 Epochs，BatchSize 大小为 24，PadSize 为 32，学习率为 5e-5，HiddenSize 为 768。

最终，模型微调后在 Test 集上的准确率为 88.89%，量化结果展示在表1中。

表 1: BERT 模型测试集量化结果

分类	Precision↑	Recall↑	F1-Score↑	Support
宠物	1.0000	1.0000	1.0000	2
知识	0.8333	1.0000	0.9091	5
解说	0.3333	0.5000	0.4000	2
游戏	1.0000	0.9231	0.9600	13
娱乐	0.8333	0.7692	0.8000	13
二次元	0.9231	0.9231	0.9231	13
音乐	0.8750	0.8750	0.8750	8
美食	1.0000	1.0000	1.0000	2
体育	0.8750	0.8750	0.8750	8
时尚	1.0000	1.0000	1.0000	6

模型经过微调，可以将其用于测试数据或实际应用中，以进行文本分类等任务的预测。模型将输出每个类别的概率分布，可以根据概率来进行分类。

之后我们使用 GradIO 进行模型推理 API 的搭建，效果如图8所示。

推荐系统 ：

LightFM 简介

LightFM 是许多流行的隐式和显式反馈推荐算法的 Python 实现。

它还使得将项目和用户元数据合并到传统的矩阵分解算法中成为可能。它将每个用户和项目表示为其特征的潜在表示的总和，从而允许推荐泛化到新项目（通过项目特征）和新用户（通过用户特征）[2]。

图 8: BERT 文本分类模型的演示效果

表 2: 随机选出的 20 个采集到的数据集样例

标题	类别序号
“别灰心普普通通的你也值得被万般宠溺” # 戴上耳机 # 音乐合集 # 一起听歌 # 谁 # 这还拿不下你	6
宝这个 99% 的人都踩过的坑，这份避坑指南你需要了解一下！ # 涨知识 # 贴牌商标 # 科普 # 冷知识	1
# 门将摧毁每一个想要进球的英雄梦想这就是我的足球 #GK# 体育生 # 两米 # 女足	8
# 射击游戏	3
超动感重低音，# 戴上耳机极致享受。# 车载音乐 #dj# 嗨曲	6
她真的好“可爱”，我都想去表白了。 大型纪录片《女生干扰起跑》传奇，持续为您播出 # 运动员 # 显眼包	8
半碗剩饭留了十年，该要精神损失费的应该是李金铭吧... # 李金铭 # 明星到我家 # 综艺	4
处处皆学问 # 穿草鞋的知识搬运工	1
辛芷蕾真的好有综艺效果！# 辛芷蕾睡衣穿反了 # 辛芷蕾	4
娱乐大爆料	4
# 二次元 # 动漫 # 漫画	5
如果这个起跑让我打分，不好意思我只能给到 8.8 分.# 体育生 # 起跑加速	8
秋冬穿搭公式学起来!! # 图文伙伴计划 # 让秋冬穿搭充满秀场感 # 秋冬穿搭	9
简单零失败的葱爆煎鸡蛋，太香太好吃了 # 家常菜 # 下饭菜 # 跟着抖音学做菜	7
抢跑田联定义，开炮 0.1 秒这期间算抢跑，我认为这是反应灵敏加上有意听枪训练，所以起跑比较快，各位的看法如何？ # 体育生 # 体考 # 大学生 # 高中生	8
太清醒了，所以只接受看得见的喜欢和明确的爱 # 用情 # 我用情付诸流水 # 翻唱 # 用情付诸流水爱比不爱可悲 # 我用情付诸流水爱比不爱可悲	6
太恐怖了，这个身体冷知识一定要牢记啊！# 人体 # 冷知识 # 科普 # 车祸	1
男朋友的歌单太有感觉了	6
终于找到满意的裤子了!! 这款裤子上身真的好显瘦呀！休闲时尚又减龄！不挑身材不挑年龄！梨形、微胖、小个子身材都能轻松驾驭！关键是非常百搭！遮胯遮肉显腿直！质量厚实又舒服！# 谁穿谁好看 # 秋冬新款 # 一眼就爱上的神裤 # 不挑年龄不挑身材 # 百搭裤子	9
禁止废话：听歌也能提高智商？# 冷知识 # 万万没想到 # 有趣的知识又增长了 # 奇妙知识在抖音	1

在 MovieLens 上评估显式反馈推荐的性能，在 CrossValidated（StackExchange 网站之一）上评估隐式反馈推荐的性能。在这两种情况下，都会测试

图 9: LightFM 库图标

热启动和冷启动场景。热启动是通过以每个项目和每个用户仍然在训练交互数据中表示的方式进行交互来测试的。通过保留某些项目的所有交互来测试冷启动。模型准确性的测量方法是考虑测试交互集中的每个用户，考虑将每个项目标记为已交互或未交互的二元分类任务，然后测量相关 ROC 曲线的曲线下面积。平均值是测试集中的所有用户。

LightFM 在**冷启动和热启动场景中都表现良好**,所以本项目采用 LightFM 作为推荐系统，实现对用户进行个性化推荐。

标签推荐算法 ：

为了做到能够在用户上传视频时获得**更加顺畅的体验**，我们针对视频标题构建了一个标签推荐算法，能够在用户输入标题后给出数据库中最为相近的标签，以供用户选择。

其中，本算法涉及以下技术：

- 分词技术：是指将一段文本切割成一系列有意义的词汇单位的过程。本项目采用了 jieba 库，它是一款开源的中文分词工具，可以对中文文本进行精确、快速的分词。
- 文本向量化：是指将文本转换为机器学习算法可以处理的数据结构的过程。本项目采用的是 gensim 库，它可以生成文档-词项矩阵，将每篇文档转化为一个向量表示。
- TF-IDF 算法：是指一种用于信息检索和文本挖掘的统计方法，用于评估一篇文档在语料库中的重要程度。本项目中采用了 gensim 库中的 Tfidf-

Model 类，实现了基于词频和逆文档频率的 TF-IDF 特征表示。

- 余弦相似度：是指一种计算两个非零向量之间夹角余弦值的方法，可以衡量两个向量之间的相似度。本项目中使用了 sklearn 库中的余弦相似度函数，计算测试数据和训练数据之间的余弦距离。

具体而言，TF-IDF 由以下定义给出：

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

$$\mathrm{idf_i} = \log \frac{|D|}{\left|\{j : t_i \in d_j\}\right|} \tag{2}$$

其中 $|D|$ 表示语料库中的文件总数；$\left|\{j : t_i \in d_j\}\right|$ 定义为包含词语 t_i 的文件数目。那么有：

$$\mathrm{tfidf}_{i,j} = \mathrm{tf}_{i,j} \times \mathrm{idf}_i \tag{3}$$

某一特定文件内的高词语频率，以及该词语在整个文件集合中的低文件频率，可以产生出高权重的 TF-IDF。因此，TF-IDF 倾向于过滤掉常见的词语，保留重要的词语。故我们采用此算法作为我们的标签推荐算法的核心，其速度和推荐效果都能达到较好的水平。

Tags 推荐效果如图10所示。

图 10: Tags 推荐系统效果图

2.7 项目代码介绍

本次项目的后端服务相关目录如下：

```
backend                          # 后端服务（Java语言）
 ├── comment-service             # 评论服务
 │   ├── comment-client
 │   ├── comment-server
 │   └── target
 ├── common                      #通用模块
 │   ├── common-auth
 │   ├── common-cache
 │   ├── common-core
 │   └── target
 ├── document
 ├── gateway-service             # 网关服务
 │   ├── src
 │   └── target
```

```
├── notification-service          # 消息服务
│   ├── notification-client
│   └── notification-server
└── user-service                  # 用户服务
  ├── user-client
  └── user-server
└── video-service                 # 视频服务
    ├── video-client
    ├── video-search-server
    └── video-server
machine-learning                  # 人工智能相关服务（Python语言）
  ├── recommender-system          # 推荐系统相关代码
  └── title-classification        # BERT文本分类相关代码
    ├── bert_pretrain             # BERT预训练模型
    ├── models                    # 模型定义
    ├── pytorch_pretrained        # 定义PyTorch相关函数
    └── THUCNews                  # 训练数据集
```

2.7.1 业务流程介绍

视频上传接口 ：

对于视频上传，我们使用七牛云 SDK 对视频进行分段并发上传，从而大幅提高文件上传速度。视频上传之后，在预设转码的前提下将视频转化为多码率格式，并封装成 m3u8 以实现自适应分辨率。

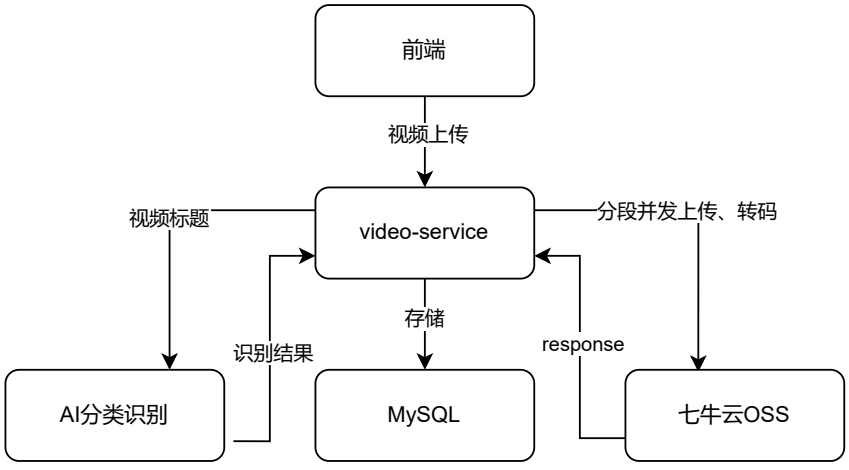

图 11: 视频上传流程图

视频埋点 ：

用户在刷视频的同时将会触发用户埋点，后端接受到数据处理并存储到 MySQL 服务器中，视频推荐算法将自动扫描数据库中信息的记录，并实时监听信息以计算视频权重，并在视频推荐时发挥作用。

BERT 标题分类接口 ：

如图12所示。

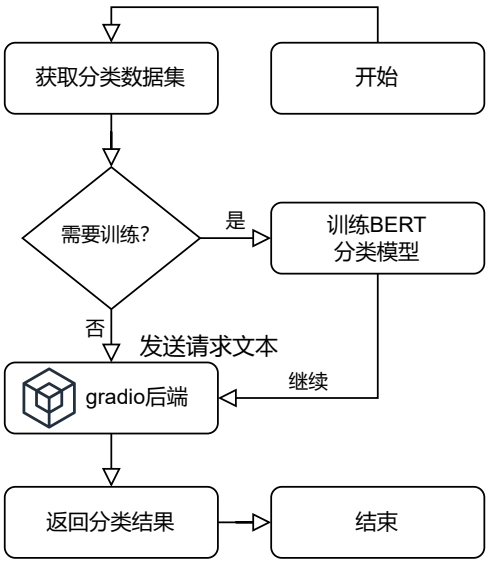

图 12: BERT 文本分类接口流程图

- 使用 `gradio` 作为后端，提供服务。
- BERT 模型仅需要一次训练。
- CPU 上即可运行，推理速度快。

推荐系统接口 ：

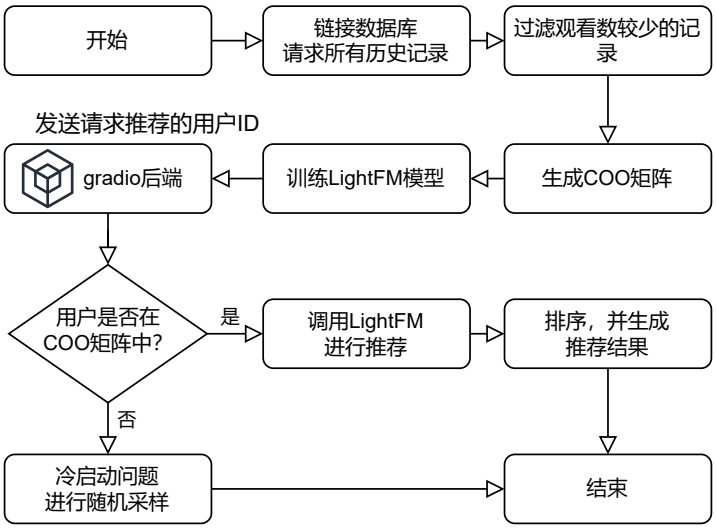

图 13: 推荐系统接口流程图

Tags 推荐接口 ：

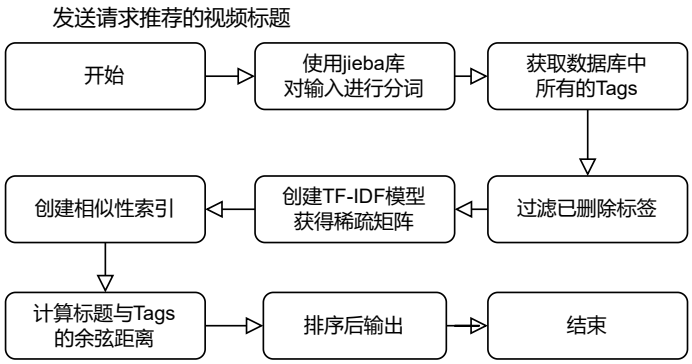

图 14: Tags 推荐接口流程图

- 使用纯机器学习算法，预测速度快。
- 无需训练即可达到较好的推荐效果。

2.7.2 前端架构

路由结构 采用配置式路由，易于管理，同时在不同页面的嵌套中将公用组件提升，能够提高页面的流畅性。

数据流 前端需要从 Api 获取数据并在处理后展示到页面上，其中最主要的就是获取视频流的过程。

由于获取的数据接口大多数是需要带页数的分页形式，也有一部分是推荐类的，只需要直接请求即可，但是这些接口获取的数据都是类列表格式的，因此存在第一个可以复用的地方——数据列表。在经过对多种数据类型的抽象后，建立了一个名为 'useLoadPerPage.ts' 的分页加载接口，仅需提供获取数据的方法，即可自动进行分页请求，并返回维护好的数据列表及更新方法，使得分页数据能够像其他接口一样重复调用即可。

第二个复用的地方是在视频展示的平铺视图与流式视图之间，两个视图可以是递进的关系，因此需要对两个组件的数据流进行复用。在平铺视图，可以对数据进行渲染，而在流式视图中，视频是以包装后的 Video 组件的形式存在的，对浏览器资源的占用较高，因此不能全部一起渲染，需要维护一个虚拟列表。此时就创建了第二个 hook——'useDataFlowShow.ts'，用于将能持续更新的视频流转换为最大长度为 3 的虚拟列表，并提供相应的翻页方法，在滚动到列表边界时能够自动加载下一页数据。

前端主要的数据经过两个 hook 的处理后能构造出一个性能优秀、架构清晰、易于拓展的数据流。

视频列表 视频采用 Hls 协议进行加载，并用原生的 video 进行渲染，相对来说性能不错。但是受制于视频解析本身的性能占用，因此在视频列表中使用虚拟列表，一次最多渲染 3 个视频，大幅提升性能。

参考文献

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] M. Kula, "Metadata embeddings for user and item cold-start recommendations," *arXiv preprint arXiv:1507.08439*, 2015.

[3] "什么是 bert? | 数据科学 | nvidia 术语表." https://www.nvidia.cn/glossary/data-science/bert/. Accessed: 2023-11-05.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.