

Machine Learning System Design Interview

Ali Aminian

ALIA@ADOBE.COM

Adobe & ByteByeGo

Alex Xu

ALEX.XU@GMAIL.COM, SYSTEMDESIGNINSIDER@GMAIL.COM

ByteByeGo

Editor: A detailed contributor list can be found in the appendix of this paper.

Abstract

机器学习系统设计面试是所有技术面试问题中最难解决的。本书提供了可靠的策略和知识库，可用于解决广泛的 ML 系统设计问题。它提供了解决 ML 系统设计问题的分步框架。它包含许多现实世界的示例来说明系统方法，并提供您可以遵循的详细步骤。对于任何对 ML 系统设计感兴趣的人，无论是初学者还是经验丰富的工程师，本书都是必备资源。同时，如果您需要准备 ML 面试，这本书是专门为您编写的。你将获得：

1. 解决任何 ML 系统设计面试问题的 7 步框架。
2. 内部人士对面试官真正寻找什么以及为什么寻找的看法。
3. 10 个真实的 ML 系统设计面试问题，附有详细解决方案。
4. 211 张图表直观地解释了各种系统的工作原理。

Date: Oct 14, 2024

Copyright of Ali Aminian and Alex Xu

MLSD INTERVIEW

目录

1	Introduction and Overview 简介与概述	1
1.1	明确需求	2
1.2	将问题框定为机器学习任务	2
1.2.1	定义机器学习目标	3
1.2.2	指定系统的输入和输出	3
1.2.3	选择正确的机器学习类别	3
1.2.4	讨论要点	5
1.3	数据准备	5
1.3.1	数据工程	5
1.3.2	数据来源	6
1.3.3	数据存储	6
1.3.3.1	提取、转换和加载 (ETL)	6
1.3.3.2	数据类型	6
1.3.3.3	数值数据	8
1.3.3.4	类别数据	8
1.3.4	特征工程	8
1.3.5	特征工程操作	9
1.3.5.1	处理缺失值	9
1.3.5.2	特征缩放	9
1.3.5.3	离散化 (Bucketing 分箱)	10
1.3.5.4	编码类别特征	10
1.3.6	讨论要点	12
1.4	模型开发	12
1.4.1	模型选择	12
1.4.2	模型训练	13
1.4.2.1	构建数据集	14
1.4.2.2	选择损失函数	15
1.4.2.3	从头训练与微调	16
1.4.2.4	分布式训练	16
1.4.3	讨论要点	16
1.5	评估	17

1.5.1	离线评估	17
1.5.2	在线评估	17
1.5.3	讨论要点	18
1.6	部署与服务	18
1.6.1	云端部署与设备端部署	18
1.6.2	模型压缩	19
1.6.3	生产环境中的测试	19
1.6.4	影子部署	19
1.6.5	A/B 测试	20
1.6.6	预测 Pipeline	21
1.6.7	讨论要点	21
1.7	监控	22
1.7.1	系统在生产中为何会失败	22
1.7.2	需要监控什么	23
1.8	基础设施	23
1.9	总结	23
2	Visual Search System 视觉搜索系统	27
2.1	明确需求	27
2.2	将问题框定为机器学习任务	28
2.2.1	定义机器学习目标	28
2.2.2	指定系统的输入和输出	28
2.2.3	选择正确的机器学习类别	28
2.2.4	如何使用表示学习对图像进行排序？	29
2.3	数据准备	30
2.3.1	数据工程	30
2.3.2	特征工程	30
2.4	模型开发	31
2.4.1	模型选择	31
2.4.2	模型训练	31
2.4.3	构建数据集	32
2.4.4	选择损失函数	33
2.5	评估	35
2.5.1	离线指标	35
2.5.2	在线指标	39
2.6	服务	39
2.6.1	预测 Pipeline	40
2.6.1.1	嵌入生成服务	40
2.6.1.2	最近邻服务	40
2.6.1.3	重排序服务	41

2.6.2	索引 Pipeline	41
2.6.3	最近邻 (NN) 算法的性能	41
2.6.3.1	精确最近邻	41
2.6.3.2	近似最近邻 (ANN)	41
2.6.3.3	基于树的 ANN	42
2.6.3.4	基于局部敏感哈希 (Locality sensitive hashing, LSH)	42
2.6.3.5	我们应该使用哪种算法?	43
2.7	其他谈话要点	44
3	Google Street View Blurring System 街景模糊系统	47
3.1	明确需求	47
3.2	将问题框定为一个机器学习任务	48
3.2.1	定义机器学习目标	48
3.2.2	指定系统的输入和输出	48
3.2.3	选择正确的机器学习类别	48
3.2.4	双阶段网络 Two-stage networks	48
3.2.5	单阶段网络 One-stage networks	49
3.3	数据准备	49
3.3.1	数据工程	49
3.3.1.1	标注数据集	50
3.3.1.2	街景图像	50
3.3.2	特征工程	50
3.4	模型开发	52
3.4.1	模型选择	52
3.4.1.1	卷积层	52
3.4.1.2	区域建议网络 (RPN)	52
3.4.1.3	分类器	52
3.4.2	模型训练	52
3.5	评估	54
3.5.1	离线指标	55
3.5.2	在线指标	56
3.6	服务	57
3.6.1	重叠的边界框	57
3.6.2	ML 系统设计	57
3.7	其他讨论点	58
4	YouTube Video Search 视频搜索	62
4.1	Introduction	62
4.2	明确需求	62
4.3	将问题框定为 ML 任务	63
4.3.1	定义 ML 目标	63

4.3.2	指定系统的输入和输出	63
4.3.3	选择合适的 ML 分类	63
4.4	数据准备	65
4.4.1	数据工程	65
4.4.2	特征工程	66
4.5	模型开发	68
4.5.1	模型选择	68
4.5.2	统计方法	68
4.5.3	基于机器学习的方法	70
4.5.4	视频编码器	72
4.5.5	模型训练	73
4.6	评估	74
4.6.1	离线指标	74
4.6.2	在线指标	75
4.7	服务	75
4.7.1	预测流程	75
4.7.2	视频索引流程	77
4.7.3	文本索引流程	77
4.8	其他讨论要点	77
5	Harmful Content Detection 有害内容检测	81
5.1	Introduction	81
5.2	明确需求	81
5.3	将问题框架化为一个机器学习任务	82
5.3.1	定义机器学习目标	82
5.3.2	设定系统的输入和输出	82
5.3.2.1	后期融合	82
5.3.2.2	前期融合	83
5.3.2.3	我们应该使用哪种融合方法?	84
5.3.3	选择合适的机器学习类别	84
5.3.3.1	单一二分类器	85
5.3.3.2	针对每个有害类别的二分类器	86
5.3.3.3	多标签分类器	86
5.3.3.4	多任务分类器	87
5.4	数据准备	88
5.4.1	数据工程	88
5.4.1.1	用户	88
5.4.1.2	帖子	89
5.4.1.3	用户-帖子交互	90
5.4.2	特征工程	90

5.4.2.1	文本内容	90
5.4.2.2	图片或视频	91
5.4.2.3	用户对帖子的反应	91
5.4.2.4	作者特征	91
5.4.2.5	上下文信息	93
5.5	模型开发	93
5.5.1	模型选择	93
5.5.2	模型训练	93
5.6	评估	96
5.6.1	离线指标	96
5.6.2	在线指标	96
5.7	部署	97
5.7.1	有害内容检测服务	97
5.7.2	违规执行服务	97
5.7.3	降级服务	98
5.7.4	其他讨论点	98
6	Video Recommendation System 视频推荐系统	101
6.1	明确需求	101
6.2	将问题框定为机器学习任务	102
6.2.1	定义机器学习目标	102
6.2.2	指定系统的输入和输出	102
6.2.3	选择合适的机器学习类别	102
6.2.3.1	基于内容的过滤	103
6.2.3.2	协同过滤 (Collaborative Filtering, CF)	104
6.2.3.3	混合过滤 Hybrid filtering	105
6.2.3.4	我们应该选择哪种方法?	106
6.3	数据准备	106
6.3.1	数据工程	106
6.3.2	特征工程	107
6.4	模型开发	110
6.4.1	矩阵分解	110
6.4.1.1	反馈矩阵	110
6.4.1.2	矩阵分解训练	112
6.4.1.3	矩阵分解优化	113
6.4.1.4	矩阵分解推理	114
6.4.2	双塔神经网络	115
6.4.3	矩阵分解 vs. 双塔神经网络	117
6.5	评估	118
6.5.1	离线指标	118

6.5.2	在线指标	118
6.6	服务	119
6.6.1	候选生成	120
6.6.2	评分 Scoring	121
6.6.3	重排序 Re-ranking	122
6.7	视频推荐系统的挑战	122
6.7.1	服务速度	122
6.7.2	精确度 Precision	123
6.7.3	多样性 Diversity	123
6.7.4	冷启动问题	123
6.7.5	训练的可扩展性	123
6.8	其他谈话要点	123
7	Event Recommendation System 活动推荐系统	125
7.1	明确需求	125
7.2	将问题框架化为机器学习任务	126
7.2.1	定义机器学习目标	126
7.2.2	指定系统的输入和输出	126
7.2.3	选择合适的机器学习类别	126
7.2.3.1	点对点 (Pointwise) LTR	127
7.2.3.2	对对比 (Pairwise) LTR	127
7.2.3.3	列表对比 (Listwise) LTR	128
7.3	数据准备	128
7.3.1	数据工程	128
7.3.1.1	用户	129
7.3.1.2	活动	129
7.3.1.3	友谊关系	129
7.3.1.4	互动数据	129
7.3.2	特征工程	129
7.3.2.1	位置相关特征	130
7.3.2.2	时间相关特征	131
7.3.2.3	社交相关特征	133
7.3.2.4	用户相关特征	134
7.3.2.5	活动相关特征	134
7.4	模型开发	135
7.4.1	模型选择	135
7.4.1.1	逻辑回归 (LogReg)	136
7.4.1.2	决策树	137
7.4.1.3	梯度提升决策树 GBDT	139
7.4.1.4	神经网络 (NN)	140

7.4.2	我们应该选择哪个模型?	141
7.5	模型训练	142
7.5.1	构建数据集	142
7.5.2	选择损失函数	142
7.6	评估	142
7.6.1	离线指标	142
7.6.2	在线指标	143
7.7	服务	144
7.7.1	在线学习 pipeline	144
7.7.2	预测 pipeline	144
7.8	其他讨论点	144
8	Ad Click Prediction on Social Platforms 社交平台上的广告点击预测	149
8.1	Introduction	149
8.2	明确需求	149
8.3	将问题框定为机器学习任务	150
8.3.1	定义机器学习目标	150
8.3.2	规定系统的输入和输出	150
8.3.3	选择合适的机器学习类别	151
8.4	数据准备	151
8.4.1	数据工程	151
8.4.2	特征工程	152
8.4.3	广告特征	152
8.4.3.1	ID	152
8.4.3.2	图像/视频	152
8.4.3.3	广告类别和子类别	153
8.4.3.4	展示和点击次数	153
8.4.4	用户特征	153
8.5	模型开发	154
8.5.1	模型选择	154
8.5.1.1	逻辑回归 (LogReg)	155
8.5.1.2	特征交叉 + LogReg	155
8.5.1.3	梯度提升决策树 (GBDT)	156
8.5.1.4	GBDT + LogReg	157
8.5.1.5	神经网络 (NN)	158
8.5.1.6	深度与交叉网络 (DCN)	159
8.5.1.7	因子分解机 (FM)	160
8.5.1.8	深度因子分解机 (DeepFM)	161
8.5.2	模型训练	162
8.5.2.1	构建数据集	162

	8.5.2.2 选择损失函数	162
8.6	评估	163
	8.6.1 离线指标	163
	8.6.2 在线指标	163
8.7	服务	165
	8.7.1 数据准备 Pipeline	165
	8.7.2 持续学习 Pipeline	166
	8.7.3 预测 Pipeline	167
8.8	其他讨论点	167
9	Similar Listings on Vacation Rental Platforms 度假租赁平台上的相似房源	169
9.1	明确需求	169
9.2	将问题框架化为机器学习任务	170
	9.2.1 定义机器学习目标	170
	9.2.2 指定系统的输入和输出	170
	9.2.3 选择合适的机器学习类别	170
	9.2.4 基于会话的推荐系统	171
9.3	数据准备	172
	9.3.1 数据工程	172
	9.3.2 特征工程	172
9.4	模型开发	173
	9.4.1 模型选择	173
	9.4.2 模型训练	173
	9.4.2.1 构建数据集	174
	9.4.2.2 选择损失函数	174
9.5	评估	176
	9.5.1 离线指标	176
	9.5.2 在线指标	177
9.6	服务	178
	9.6.1 训练 Pipeline	178
	9.6.2 索引 Pipeline	178
	9.6.3 预测 Pipeline	179
9.7	其他讨论点	179
10	Personalized News Feed 个性化新闻推送	181
10.1	Introduction	181
10.2	明确需求	181
10.3	将问题框架化为机器学习任务	182
	10.3.1 定义机器学习目标	182
	10.3.2 指定系统的输入和输出	183
	10.3.3 选择合适的机器学习类别	183

10.4	数据准备	184
10.4.1	数据工程	184
10.4.1.1	用户	184
10.4.1.2	帖子	184
10.4.1.3	用户-帖子互动	185
10.4.1.4	朋友关系	185
10.4.2	特征工程	185
10.4.2.1	帖子特征	185
10.4.2.2	用户特征	187
10.4.2.3	用户-帖子历史互动	187
10.4.2.4	在帖子中被提及	188
10.4.2.5	用户-作者的关联性	188
10.5	模型开发	189
10.5.1	模型选择	189
10.5.2	模型训练	190
10.5.2.1	构建数据集	190
10.5.2.2	选择损失函数	192
10.6	评估	192
10.6.1	离线指标	192
10.6.2	在线指标	192
10.7	服务	193
10.8	其他讨论点	194
11	People You May Know 你可能认识的人	196
11.1	Introduction	196
11.2	明确需求	196
11.3	将问题框架化为机器学习任务	197
11.3.1	定义机器学习目标	197
11.3.2	指定系统的输入和输出	197
11.3.3	选择合适的机器学习类别	197
11.3.3.1	点对点 LTR	197
11.3.3.2	边预测	199
11.4	数据准备	199
11.4.1	数据工程	199
11.4.1.1	用户	200
11.4.1.2	连接	201
11.4.1.3	互动	201
11.4.2	特征工程	201
11.4.2.1	用户特征	201
11.4.2.2	用户-用户亲和度	201

11.5	模型开发	202
11.5.1	模型选择	202
11.5.2	GNNs	203
11.5.3	模型训练	204
11.5.3.1	构建数据集	204
11.5.3.2	选择损失函数	205
11.6	评估	205
11.6.1	离线指标	205
11.6.2	在线指标	206
11.7	服务	207
11.7.1	效率	207
11.7.1.1	利用朋友的朋友 (FoF)	207
11.7.1.2	预算算 PYMK	208
11.7.2	机器学习系统设计	208
11.8	其他讨论点	210

1 Introduction and Overview 简介与概述

我们编写这本书的目的是帮助机器学习（ML）工程师和数据科学家在机器学习系统设计面试中取得成功。本书也适用于那些希望了解机器学习在现实世界中的应用的读者。

许多工程师认为机器学习系统的全部内容就是像逻辑回归或神经网络这样的机器学习算法。然而，生产环境中的机器学习系统远不止模型开发。机器学习系统通常非常复杂，由多个组件组成，包括用于管理数据的数据栈、使系统可供数百万用户使用的服务基础设施、用于评估系统性能的评估 Pipeline，以及确保模型性能不会随着时间下降的监控系统。

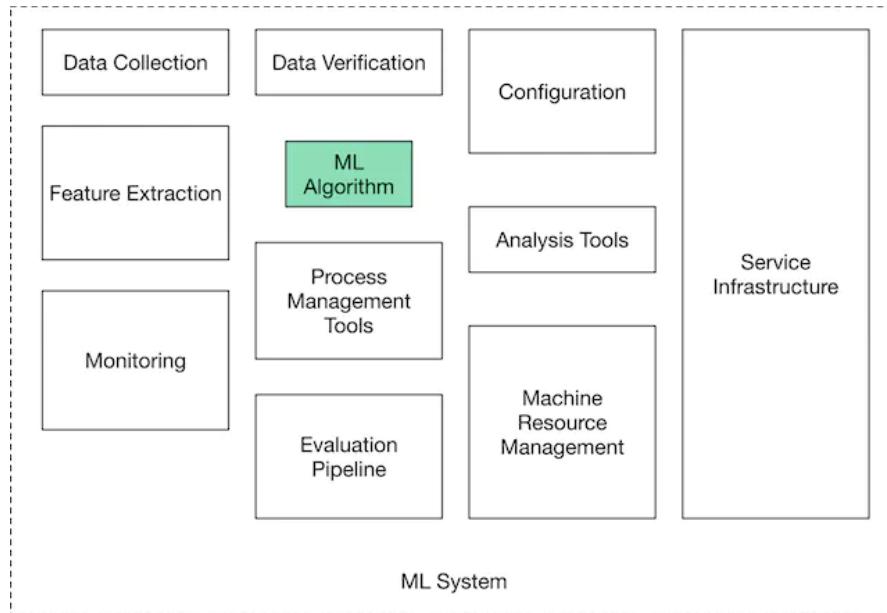


图 1.1: 生产就绪的机器学习系统的组件

在机器学习系统设计面试中，你需要回答开放性的问题。例如，你可能会被要求设计一个电影推荐系统或视频搜索引擎。没有唯一正确的答案。面试官希望评估你的思维过程、你对各种机器学习主题的深入理解、设计端到端系统的能力，以及基于各种选项权衡做出的设计选择。

要成功设计复杂的机器学习系统，遵循一个框架非常重要。结构不清晰的回答会使逻辑难以理解。在本简介中，我们提出了一个框架，该框架贯穿本书，以应对机器学习系统设计的问题。该框架包括以下关键步骤：

1. 明确需求
2. 将问题框定为机器学习任务
3. 数据准备
4. 模型开发
5. 评估
6. 部署与服务

7. 监控与基础设施



图 1.2: 机器学习系统设计步骤

每个机器学习系统设计面试都是不同的，因为问题是开放性的，并且没有一种放之四海而皆准的方法。该框架旨在帮助你组织思路，但不必严格遵循它。保持灵活性。如果面试官主要关注模型开发，你几乎总是应该遵循他们的关注点。

让我们通过框架中的每个步骤来进行讨论。

1.1 明确需求

机器学习系统设计问题通常故意含糊，信息最少。例如，一个面试问题可能是：“设计一个事件推荐系统”。第一步是提出明确的问题。那么该问什么问题呢？我们应该通过提问来理解确切的需求。以下是一些分类问题列表，帮助我们入门：

- **业务目标。**如果我们被要求创建一个系统来推荐度假租赁，两个可能的动机是增加预订数量和提高收入。
- **系统需要支持的功能。**系统预期支持哪些功能可能会影响我们的机器学习系统设计？例如，假设我们被要求设计一个视频推荐系统。我们可能想知道用户是否可以“点赞”或“不喜欢”推荐的视频，因为这些互动可以用于标记训练数据。
- **数据。**数据来源是什么？数据集有多大？数据是否有标签？
- **约束条件。**可用的计算能力有多少？是基于云的系统还是需要在设备上运行？模型是否需要随着时间自动改进？
- **系统规模。**我们有多少用户？处理多少个项目，比如视频？这些指标的增长率是多少？
- **性能。**预测速度有多快？是否需要实时解决方案？准确性更重要还是延迟更重要？

此列表并不详尽，但可以作为一个起点。记住，其他主题，如隐私和伦理，也可能很重要。在此步骤结束时，我们应该与面试官就系统的范围和需求达成一致。通常，将收集到的需求和约束列出是个好主意，这样可以确保大家达成共识。

1.2 将问题框定为机器学习任务

有效的问题框定对解决机器学习问题起着至关重要的作用。假设面试官要求你提高视频流平台的用户参与度。缺乏用户参与度确实是一个问题，但这不是一个机器学习任务。因此，我们需要将其框定为一个机器学习任务来解决。实际上，我们首先需要确定机器学习是否对解决问题有必要。在机器学习系统设计面试中，可以假设机器学习是有帮助的。因此，我们可以通过以下方式将问题框定为机器学习任务：

- 定义机器学习目标
- 指定系统的输入和输出
- 选择正确的机器学习类别

1.2.1 定义机器学习目标

业务目标可能是将销售额提高 20% 或改善用户留存率。但目标可能没有明确定义，我们不能仅仅通过告诉模型“提高 20% 的销售额”来训练它。为了让机器学习系统解决任务，我们需要将业务目标转化为明确的机器学习目标。一个好的机器学习目标是机器学习模型能够解决的目标。让我们来看一些示例，如表 1.1 所示。在后续章节中，我们将看到更多示例。

表 1.1：将业务目标转化为机器学习目标

应用	业务目标	机器学习目标
活动门票销售应用	增加票务销售	最大化活动注册量
视频流媒体应用	提高用户参与度	最大化用户观看视频的时间
广告点击预测系统	增加用户点击量	最大化点击率
社交媒体平台中的有害内容检测	提高平台的安全性	准确预测给定内容是否有害
朋友推荐系统	提高用户扩大社交网络的速度	最大化建立连接的数量

1.2.2 指定系统的输入和输出

一旦确定了机器学习目标，我们需要定义系统的输入和输出。例如，对于社交媒体平台的有害内容检测系统，输入是一个帖子，输出是该帖子是否被认为有害。

在某些情况下，系统可能由多个机器学习模型组成。如果是这样，我们需要指定每个机器学习模型的输入和输出。例如，对于有害内容检测，我们可能希望使用一个模型预测暴力内容，另一个模型预测裸露内容。系统依赖这两个模型来确定帖子是否有害。

另一个重要的考虑是，可能有多种方式来指定每个模型的输入输出。图 1.4 展示了一个示例。

1.2.3 选择正确的机器学习类别

将问题框定为机器学习任务有多种方式。大多数问题可以框定为图 1.5 所示的机器学习类别（叶节点）之一。由于大多数读者可能已经熟悉这些类别，我们这里只提供一个概述。

监督学习。监督学习模型通过使用带标签的数据集来学习任务。实际上，许多问题都属于这一类别，因为从带标签的数据中学习通常能得到更好的结果。

无监督学习。无监督学习模型通过处理不含正确答案的数据来进行预测。无监督学习模型的目标是从数据中发现潜在结构。

强化学习。在强化学习中，计算机智能体通过与环境的反复试错互动来学习执行任务。例如，可以训练机器人在房间中行走，也可以通过强化学习让 AlphaGo 这样的程序与对手在围棋中竞技。

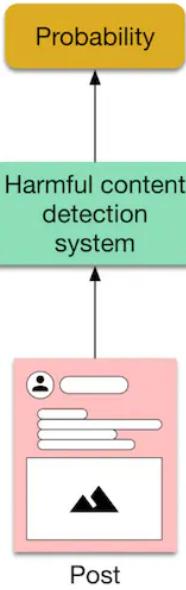


图 1.3: 有害内容检测系统输入输出

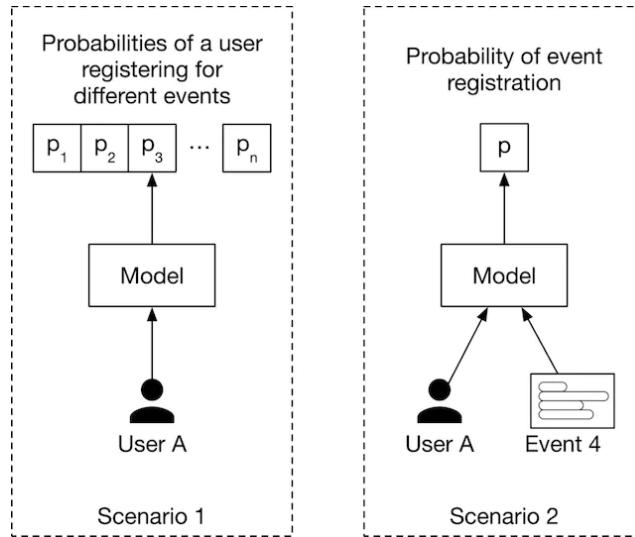


图 1.4: 指定模型输入输出的不同方式

与监督学习相比，无监督学习和强化学习在实际系统中的应用较少，因为在有训练数据时，机器学习模型通常能更好地学习特定任务。因此，本书中我们讨论的大多数问题都依赖于监督学习。让我们更详细地了解监督学习的不同类别。

分类模型。分类是预测离散类别标签的任务，例如预测输入图像是否应该被分类为“狗”、“猫”或“兔子”。分类模型可以分为两组：

- **二分类模型。**预测一个二元的结果。例如，模型预测图像中是否包含狗。
- **多分类模型。**将输入分类为多个类别。例如，我们可以将图像分类为狗、猫或兔子。

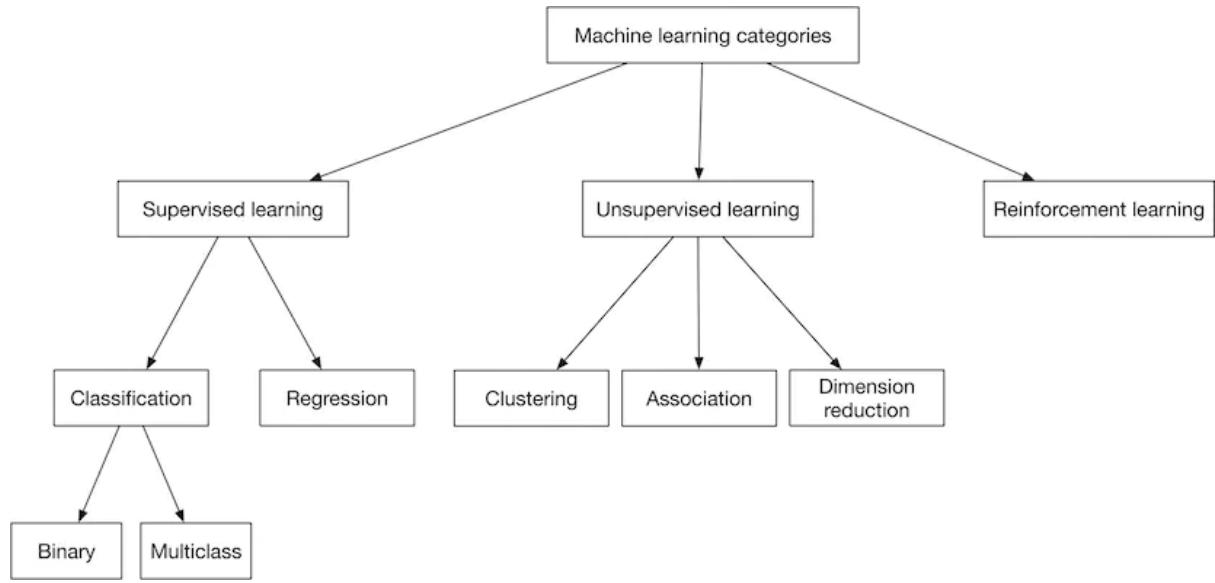


图 1.5: 常见的机器学习类别

在这一步，你需要选择正确的机器学习类别。后续章节将提供在面试中如何选择正确类别的示例。

1.2.4 讨论要点

以下是我们面试中可能要讨论的一些话题：

- 什么是好的机器学习目标？不同的机器学习目标如何比较？各有什么优缺点？
- 给定机器学习目标，系统的输入和输出是什么？
- 如果机器学习系统涉及多个模型，每个模型的输入和输出是什么？
- 任务是需要以监督学习的方式还是无监督学习的方式来完成？
- 是用回归模型还是分类模型更好？如果是分类，是二分类还是多分类？如果是回归，输出范围是什么？

1.3 数据准备

机器学习模型直接从数据中学习，这意味着具有预测能力的数据对于训练机器学习模型至关重要。本节旨在通过数据工程和特征工程这两个关键过程，为机器学习模型准备高质量的输入。我们将涵盖每个过程中重要的方面。

1.3.1 数据工程

数据工程是设计和构建用于收集、存储、检索和处理数据的 Pipeline 的实践。让我们简要回顾数据工程的基础知识，以了解我们可能需要的核心组件。

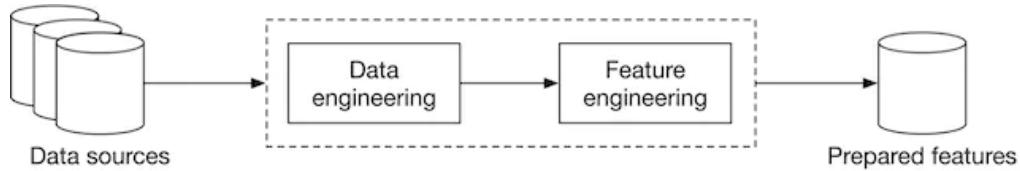


图 1.6: 数据准备过程

1.3.2 数据来源

机器学习系统可以处理来自许多不同来源的数据。了解数据来源是回答许多背景问题的好方法，这些问题可能包括：谁收集的数据？数据有多干净？数据来源是否可信？数据是用户生成的还是系统生成的？

1.3.3 数据存储

数据存储，也称为数据库，是一个用于持久存储和管理数据集合的存储库。不同的数据库是为满足不同的使用场景而构建的，因此了解不同类型的工作原理是很重要的。在机器学习系统设计面试中，通常不需要深入了解数据库的内部结构。

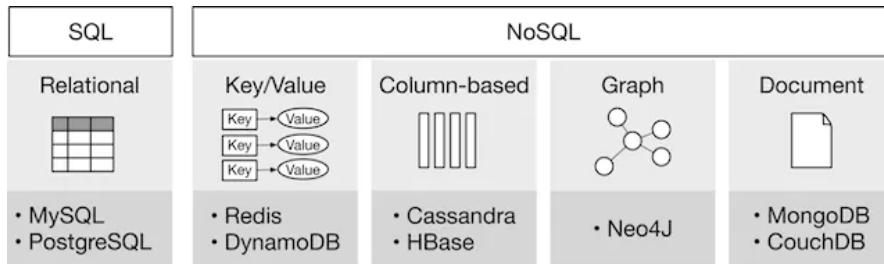


图 1.7: 不同类型的数据库

1.3.3.1 提取、转换和加载 (ETL)

ETL 包含三个阶段：

- **提取 (Extract)** 这个过程从不同的数据源中提取数据。
- **转换 (Transform)** 在此阶段，数据通常会被清洗、映射并转换为特定格式以满足操作需求。
- **加载 (Load)** 转换后的数据被加载到目标目的地，可以是文件、数据库或数据仓库¹。

1.3.3.2 数据类型

在机器学习中，数据类型与编程语言中的数据类型（如 `int`、`float`、`string` 等）有所不同。从高层次来看，数据类型可以分为两类：结构化数据和非结构化数据，如图1.9所示。

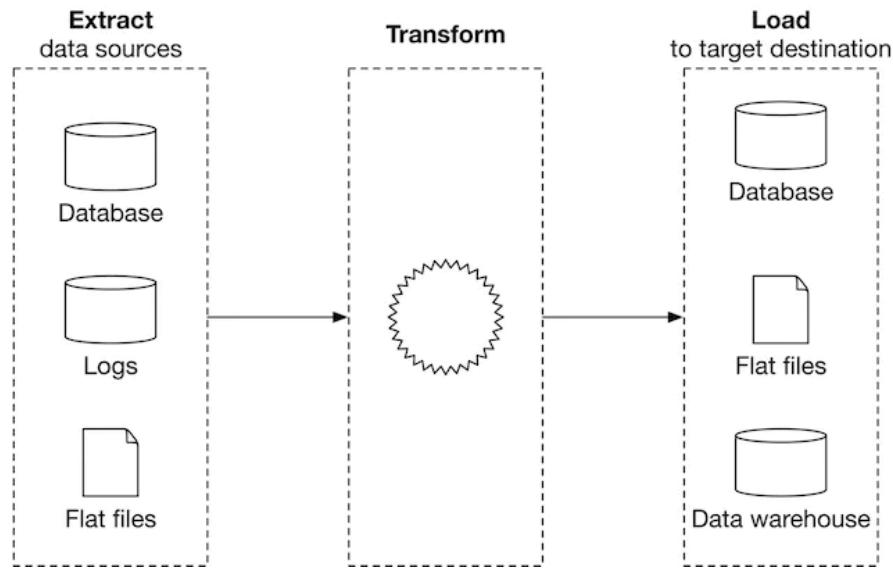


图 1.8: ETL 过程概览

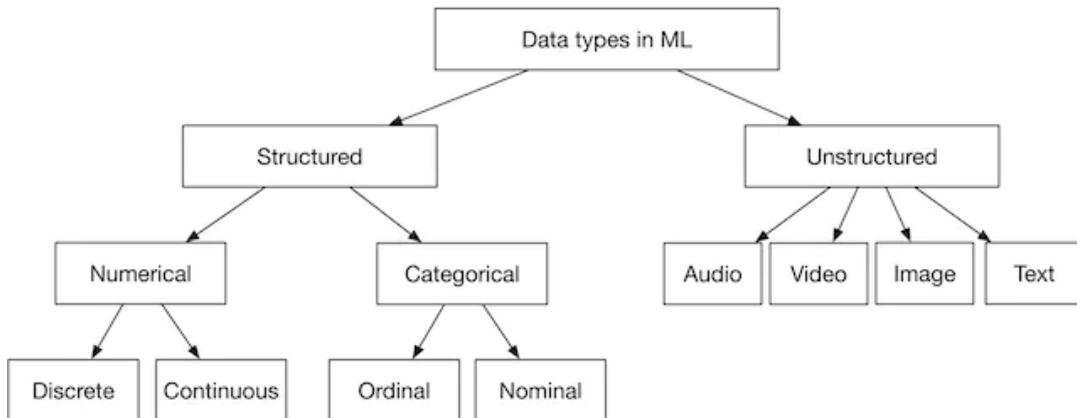


图 1.9: 机器学习中的数据类型

结构化数据遵循预定义的数据模式，而非结构化数据则不遵循。例如，日期、姓名、地址、信用卡号，以及任何可以用行和列表示的内容，都可以被视为结构化数据。非结构化数据指的是没有底层数据模式的数据，如图像、音频文件、视频和文本。表1.2总结了结构化和非结构化数据的关键差异。

表 1.2: 结构化数据与非结构化数据的对比总结

	结构化数据	非结构化数据
特点	<ul style="list-style-type: none"> • 预定义模式 • 易于搜索 	<ul style="list-style-type: none"> • 无模式 • 难以搜索
存储位置	<ul style="list-style-type: none"> • 关系型数据库 • 数据仓库 	<ul style="list-style-type: none"> • NoSQL 数据库 • 数据湖
示例	<ul style="list-style-type: none"> • 日期、电话号码 • 信用卡号、地址 	<ul style="list-style-type: none"> • 文本文件、音频文件 • 图像、视频

如图1.10所示，机器学习模型根据数据类型的不同而表现不同。了解并澄清数据是结构化还是非结构化，有助于我们在模型开发阶段选择合适的机器学习模型。

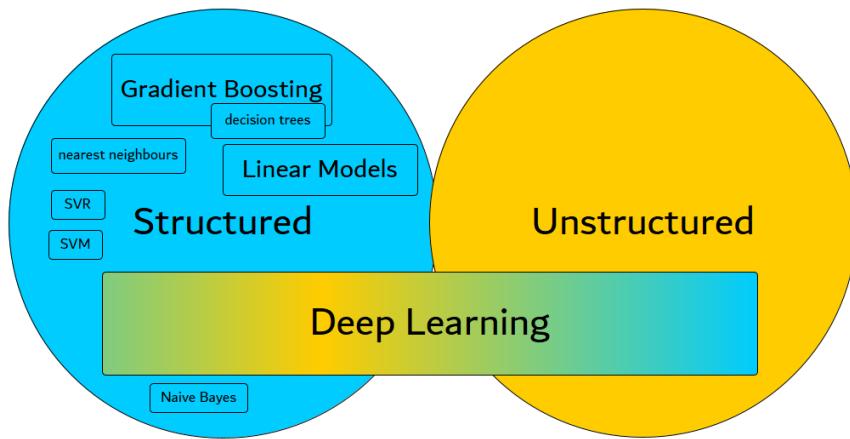


图 1.10: 适用于结构化和非结构化数据的模型

1.3.3.3 数值数据

数值数据是由数字表示的任何数据点。如图1.9所示，数值数据分为连续数值数据和离散数值数据。例如，房价可以被视为连续数值数据，因为房价可以在一定范围内取任何值。相反，过去一年中售出的房屋数量可以被视为离散数值数据，因为它只取有限的离散值。

1.3.3.4 类别数据

类别数据指的是可以基于分配的名称或标签进行存储和识别的数据。例如，性别是类别数据，因为它的值来自有限的范围。类别数据可以分为两组：名义数据和顺序数据。

名义数据是指其类别之间没有数值关系的数据。例如，性别是名义数据，因为“男性”和“女性”之间没有数值关系。顺序数据是指具有预定或顺序关系的数据。例如，评分数据，它可以从“不满意”、“中性”和“满意”这三个唯一值中取值，是顺序数据的一个示例。

1.3.4 特征工程

特征工程包括两个过程：

- 使用领域知识从原始数据中选择和提取具有预测能力的特征
- 将预测特征转换为模型可用的格式

选择合适的特征是开发和训练机器学习模型时最重要的决策之一。选择最有价值的特征至关重要，而这一特征工程过程需要领域专家的知识，同时也高度依赖于具体任务。为了帮助你掌握这个过程，我们在整本书中提供了许多示例。

一旦选择了预测特征，就需要通过特征工程操作将它们转换为合适的格式，下面我们将详细探讨这些操作。

1.3.5 特征工程操作

通常，一些选定的特征并不是模型可以直接使用的格式。特征工程操作可以将这些特征转换为模型可使用的格式。技术包括处理缺失值、缩放值有偏分布的数据，以及编码类别特征。以下列表并不全面，但包含了一些针对结构化数据的常见操作。

1.3.5.1 处理缺失值

生产环境中的数据通常会有缺失值，通常可以通过两种方式来处理：删除或填补。

删除。这种方法会删除任何包含缺失值的记录。删除可以分为行删除和列删除。在列删除中，如果某个特征的缺失值太多，我们会移除整个表示该特征的列。在行删除中，如果某个数据点有许多缺失值，我们会移除这一行。

Column-1	Column-2
A	10
B	19
A	NaN
A	12

Column-1
A
B
A
A

图 1.11: 列删除

删除的缺点是，它减少了模型可以用于训练的数据量。这并不是理想的，因为机器学习模型通常在接触更多数据时表现更好。

填补。或者，我们可以通过填入某些值来补充缺失值。一些常见的做法包括：

- 用默认值填补缺失值
- 用均值、中位数或众数（最常见的值）填补缺失值

填补的缺点是它可能会给数据引入噪声。需要注意的是，没有一种处理缺失值的方法是完美的，每种方法都有其权衡。

1.3.5.2 特征缩放

特征缩放是指将特征缩放到标准范围和分布的过程。让我们先来看一下为什么可能需要特征缩放。

当数据集中的特征处于不同的范围时，许多机器学习模型在学习任务时会遇到困难。例如，年龄和收入这样的特征可能有不同的取值范围。此外，当一个特征具有偏态分布时，一些模型在学习任务时也可能遇到困难。那么，有哪些特征缩放技术呢？让我们来看看。

归一化（最小-最大缩放 (min-max scaling)）。在这种方法中，特征被缩放到所有值都在 [0, 1] 的范围内，使用以下公式：

$$z = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

需要注意的是，归一化不会改变特征的分布。如果要将特征的分布更改为标准分布，则使用标准化。

标准化 Standardization (即 Z-score normalization)。标准化是将特征的分布更改为均值为 0, 标准差为 1 的过程。使用以下公式对特征进行标准化:

$$z = \frac{x - \mu}{\sigma}$$

其中, μ 是特征的均值, σ 是标准差。

对数缩放。为了减小特征的偏度, 可以使用一种称为对数缩放的常用技术, 其公式为:

$$z = \log(x)$$

对数变换可以使数据分布更不偏态, 并加快优化算法的收敛速度。

1.3.5.3 离散化 (BUCKETING 分箱)

离散化是将连续特征转换为类别特征的过程。例如, 我们可以将身高表示为一个连续特征, 也可以将身高划分为离散的区间, 并用这些区间表示每个身高。这使得模型可以专注于学习少量的类别, 而不是试图学习无限多的可能性。

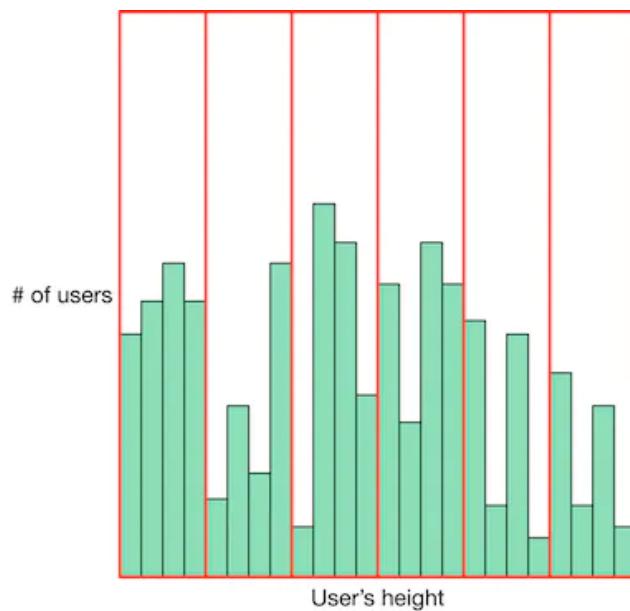


图 1.12: 将用户的身高离散化为 6 个区间

离散化也可以应用于离散特征。例如, 用户的年龄是一个离散特征, 但离散化可以减少类别数量, 如表1.3所示。

1.3.5.4 编码类别特征

在大多数机器学习模型中, 所有的输入和输出都必须是数值型的。这意味着如果某个特征是类别型的, 我们应该在将其传递给模型之前将其编码为数值。将类别特征转换为数值表示的三种常见方法是整数 (Integer) 编码、独热 (One-hot) 编码和嵌入学习 (Embedding learning)。

表 1.3: 离散化数值型年龄属性

区间	年龄范围
1	0-9
2	10-19
3	20-39
4	40-59
5	60+

整数编码 (Integer Encoding)。为每个唯一的类别值分配一个整数值。例如，“优秀”是 1，“良好”是 2，“差”是 3。这种方法在整数值之间具有自然关系时非常有用。

The diagram illustrates the process of Integer Encoding. On the left, a table titled "Feature (Height)" lists five categories: Short, Medium, Short, Tall, and Medium. An arrow labeled "Integer Encoding" points to the right, where another table titled "Integer Encoded Feature" shows the corresponding integer values: 1, 2, 1, 3, and 1 respectively.

Feature (Height)	Integer Encoded Feature
Short	1
Medium	2
Short	1
Tall	3
Medium	1

图 1.13: 整数编码

然而，当类别特征之间没有顺序关系时，整数编码并不是一个好的选择。接下来我们将探讨的独热编码就解决了这个问题。

独热编码 (One-hot Encoding)。使用这种技术，会为每个唯一值创建一个新的二进制特征。如图1.14所示，我们用三个新的二进制特征（红色、绿色和蓝色）替换了原始特征（颜色）。例如，如果一个数据点的颜色是“红色”，我们就用“1, 0, 0”替换它。

The diagram illustrates the process of One-Hot Encoding. On the left, a table titled "Feature (Color)" lists five categories: Red, Green, Yellow, Green, and Red. An arrow labeled "One Hot Encoding" points to the right, where two tables are shown side-by-side. The first table, titled "One Hot Encoded Vector", shows the binary representations: [1,0,0], [0,1,0], [0,0,1], [0,1,0], and [1,0,0] respectively. The second table, titled "Red", "Green", and "Yellow", shows the binary values for each color component across the five categories.

Feature (Color)	One Hot Encoded Vector	Red	Green	Yellow
Red	[1,0,0]	1	0	0
Green	[0,1,0]	0	1	0
Yellow	[0,0,1]	0	0	1
Green	[0,1,0]	0	1	0
Red	[1,0,0]	1	0	0

图 1.14: 独热编码

嵌入学习 (Embedding learning)。另一种编码类别特征的方法是使用嵌入学习。嵌入是一种将类别特征映射到 N 维向量的方法。嵌入学习是为类别特征可能取的每一个唯一值学习一个 N 维向量的过程。当特征可能取的唯一值数量非常大时，这种方法很有用。在这种情况下，独热编码不是一个好的选择，因为它会导致非常大的向量大小。在后续章节中我们将看到更多的例子。

1.3.6 讨论要点

以下是在面试中可能想要讨论的一些话题：

- **数据可用性和数据收集：**数据来源是什么？我们可以获取哪些数据，如何收集这些数据？数据的大小有多大？新数据的输入频率是多少？
- **数据存储：**数据目前存储在哪里？是在云端还是在用户设备上？哪种数据格式适合存储这些数据？我们如何存储多模态数据，例如同时包含图像和文本的数据点？
- **特征工程：**我们如何将原始数据处理成对模型有用的形式？我们该如何处理缺失数据？这个任务是否需要特征工程？我们使用哪些操作将原始数据转换为机器学习模型可用的格式？我们是否需要对特征进行归一化？我们应该从原始数据中构建哪些特征？我们计划如何结合不同类型的数据，如文本、数字和图像？
- **隐私：**可用数据的敏感性如何？用户是否关心他们数据的隐私？是否需要对用户数据进行匿名化？是否可以将用户的数据存储在我们的服务器上，还是只能在用户设备上访问他们的数据？
- **偏差：**数据中是否存在偏差？如果有，存在哪些类型的偏差，我们如何进行修正？

1.4 模型开发

模型开发是指选择合适的机器学习模型并对其进行训练，以解决当前任务的过程。

1.4.1 模型选择

模型选择是为预测建模问题选择最佳机器学习算法和架构的过程。实际上，选择模型的典型过程包括：

- **建立一个简单的基线。**例如，在视频推荐系统中，可以通过推荐最受欢迎的视频来获得基线。
- **尝试简单的模型。**在有了基线之后，探索一些训练速度较快的机器学习算法是一个好的做法，例如逻辑回归。
- **切换到更复杂的模型。**如果简单模型无法提供令人满意的结果，那么我们可以考虑更复杂的模型，如深度神经网络。
- **如果需要更精确的预测，可以使用一个模型的集成（ensemble）。**使用多个模型的集成而不是单一模型可能会提高预测质量。创建模型集成有三种方式：bagging [3](#)、boosting [4](#) 和 stacking [5](#)，这些将在后续章节中讨论。

在面试中，探索各种模型选项并讨论它们的优缺点很重要。一些常见的模型选项包括：

- 逻辑回归
- 线性回归

- 决策树
- 梯度提升决策树和随机森林
- 支持向量机
- 朴素贝叶斯
- 因子分解机 (FM)
- 神经网络

在分析不同的选项时，最好简要解释算法并讨论其权衡。例如，逻辑回归可能是学习线性任务的好选择，但如果任务比较复杂，我们可能需要选择不同的模型。在选择机器学习算法时，重要的是考虑模型的不同方面。例如：

- 模型需要用于训练的数据量
- 训练速度
- 超参数选择和超参数调优技术
- 持续学习的可能性
- 计算需求。一个更复杂的模型可能提供更高的准确性，但可能需要更多的计算能力，如 GPU 而非 CPU
- 模型的可解释性 [6](#)。一个更复杂的模型可能提供更好的性能，但其结果可能更难以解释

没有一种能够解决所有问题的最优算法。面试官希望看到你对不同机器学习算法的深入理解、它们的优缺点，以及你根据需求和约束选择模型的能力。为了帮助你改进模型选择，本书提供了各种模型选择的示例。本书假设你已经熟悉常见的机器学习算法。如果需要回顾，可以阅读 [7](#)。

1.4.2 模型训练

一旦完成了模型选择，就可以开始训练模型了。在这个步骤中，有一些话题可以在面试中讨论，比如：

- 构建数据集
- 选择损失函数
- 从头训练与微调
- 分布式训练

让我们逐一进行探讨。

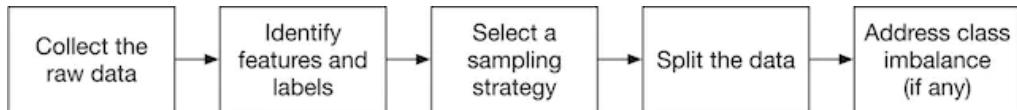


图 1.15: 数据集构建步骤

1.4.2.1 构建数据集

在面试中，讨论模型训练和评估的数据集构建通常是一个好主意。如图 1.15 所示，构建数据集有五个步骤。

除“识别特征和标签”外的所有步骤都是通用操作，可以应用于任何机器学习系统设计任务。在本章中，我们将仔细研究每个步骤，但在后续章节中，我们主要关注与任务相关的“识别特征和标签”。

收集原始数据 这一点在数据准备步骤中已经详细讨论，因此在此不再赘述。

识别特征和标签 在特征工程步骤中，我们已经讨论了应该使用哪些特征。因此，让我们专注于为数据创建标签。获取标签有两种常见方式：人工标注和自然标注。

人工标注。这意味着个别标注者手动标注数据。例如，个别标注者标注一个帖子是否包含虚假信息。由于有人工参与，人工标注通常能得到准确的标签。然而，人工标注也有很多缺点；它成本高且速度慢，会引入偏差，需要领域知识，并且对数据隐私构成威胁。

自然标注。在自然标注中，真实标签是自动推断的，无需人工标注。让我们通过一个例子更好地理解自然标签。

假设我们想设计一个基于相关性对新闻推送进行排序的机器学习系统。解决这一任务的一个可能方法是训练一个模型，该模型以用户和帖子为输入，并输出用户在看到该帖子后按下“点赞”按钮的概率。在这种情况下，训练数据是 用户, 帖子 对及其对应的标签，如果用户点赞该帖子则标签为 1，否则为 0。通过这种方式，我们能够自然地对训练数据进行标注，而无需依赖人工标注。

在这一步中，清晰地说明我们如何获取训练标签以及训练数据的样子非常重要。

选择采样策略 通常，收集所有数据是不现实的，因此采样是减少系统数据量的一种有效方式。常见的采样策略包括便利采样、滚雪球采样、分层采样、水库采样和重要性采样。想要了解更多关于采样方法的信息，可以参考⁸。

划分数据 数据划分是指将数据集分为训练集、评估（验证）集和测试集的过程。要了解更多数据划分技术，可以参考⁹。

解决类别不平衡问题 具有偏斜类别标签的数据集被称为不平衡数据集。数据集中占较大比例的类别称为多数类，而占较小比例的类别称为少数类。

不平衡数据集是模型训练中的一个严重问题，因为这意味着模型可能没有足够的数据来学习少数类。有多种技术可以缓解这个问题。让我们来看看两种常用的方法：重新采样训练数据和更改损失函数。

重新采样训练数据 重新采样是指调整不同类别之间的比例，使数据更加平衡。例如，我们可以对少数类进行过采样（图 1.17）或对多数类进行欠采样（图 1.18）。

更改损失函数 此技术通过更改损失函数，使其对类别不平衡更具鲁棒性。高层次的思路是对来自少数类的数据点赋予更高的权重。在损失函数中赋予更高的权重，可以在模型对少数类预测错误时对其进行

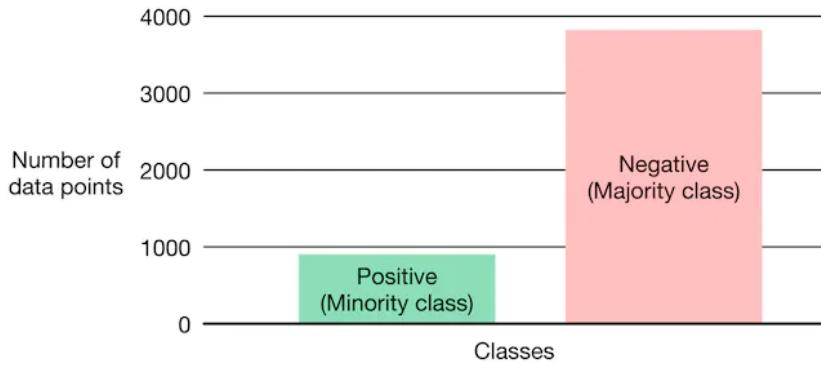


图 1.16: 具有两类的不平衡数据集

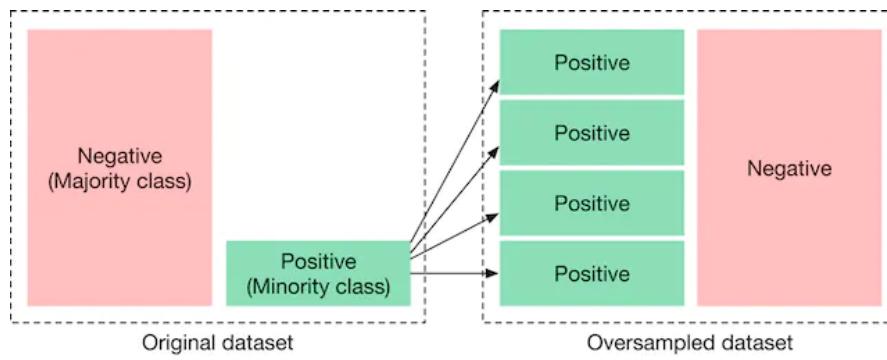


图 1.17: 对少数类进行过采样

更大的惩罚，从而迫使模型更有效地学习少数类。两种常用的缓解类别不平衡的损失函数是类别平衡损失¹⁰和焦点损失^{11 12}。

1.4.2.2 选择损失函数

一旦构建了数据集，我们需要选择一个合适的损失函数来训练模型。损失函数是衡量模型在预测预期结果时的准确性。损失函数使优化算法能够在训练过程中更新模型参数，以最小化损失。

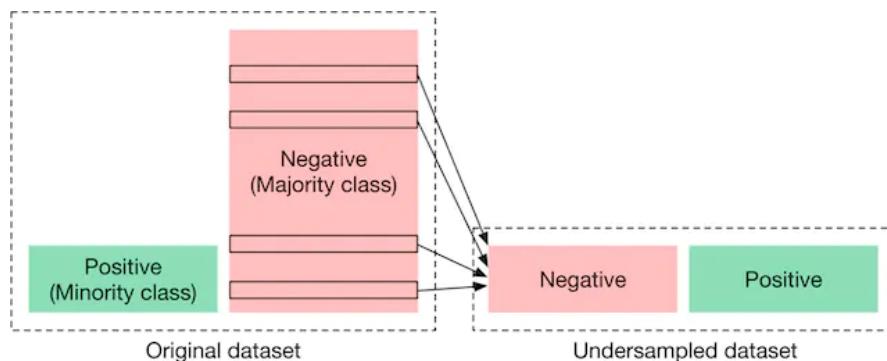


图 1.18: 对多数类进行欠采样

设计一个新颖的损失函数不容易。在机器学习面试中，通常期望你从现有损失函数列表中选择一个合适的损失函数，并根据问题的具体需求做出选择。有时，你可能需要对损失函数进行一些小改动，使其更适合特定问题。在后续章节中，我们会提供更多示例。

1.4.2.3 从头训练与微调

一个值得简要讨论的话题是从头训练与微调。微调是指通过对模型的已学参数进行微小调整，继续在新数据上训练模型。这是一个你可能需要与面试官讨论的设计决策。

1.4.2.4 分布式训练

大规模训练变得越来越重要，因为模型规模不断增大，数据集的大小也显著增加。分布式训练通常用于训练模型，通过在多个工作节点之间分配任务。这样，工作节点可以并行操作，从而加快模型训练速度。分布式训练有两种主要类型：数据并行[13](#)和模型并行[14](#)。

根据我们要解决的任务，使用分布式训练可能是必要的。在这种情况下，与面试官讨论这个话题很重要。请注意，分布式训练是一个可以不受具体任务限制进行讨论的通用话题。

1.4.3 讨论要点

以下是一些讨论要点：

- **模型选择：**哪些机器学习模型适合该任务，它们的优缺点是什么。在模型选择过程中可以考虑以下话题：
 - 训练所需时间
 - 模型期望的训练数据量
 - 模型可能需要的计算资源
 - 模型在推理时的延迟
 - 模型是否可以部署在用户设备上？
 - 模型的可解释性。使模型更复杂可能提高其性能，但结果可能更难解释
 - 我们能否利用持续训练，还是需要从头开始训练？
 - 模型有多少参数？需要多少内存？
 - 对于神经网络，你可能需要讨论典型架构/模块，如 ResNet 或基于 Transformer 的架构。你还可以讨论超参数选择，如隐藏层数、神经元数量、激活函数等。
- **数据集标签：**我们应该如何获取标签？数据是否经过标注，如果是，标注的质量如何？如果有自然标签，我们如何获取它们？我们如何接收用户对系统的反馈？获取自然标签需要多长时间？
- **模型训练。**
 - 我们应该选择什么损失函数？（例如，交叉熵 [15](#)，MSE [16](#)，MAE [17](#)，Huber 损失[18](#) 等）

- 我们应该使用什么正则化方法? (例如, L1 [19](#), L2 [19](#), 熵正则化 [20](#), K 折交叉验证 [21](#), 或 dropout [22](#))
 - 什么是反向传播?
 - 你可能需要描述一些常见的优化方法 [23](#), 如 SGD [24](#), AdaGrad [25](#), Momentum [26](#) 和 RMSProp [27](#)。
 - 我们想使用哪些激活函数以及原因是什么? (例如, ELU [28](#), ReLU [29](#), Tanh [30](#), Sigmoid [31](#)).
 - 如何处理不平衡数据集?
 - 什么是偏差/方差权衡?
 - 过拟合和欠拟合的可能原因是什么? 如何解决它们?
- **持续学习:** 我们是否希望在每个新数据点上在线训练模型? 我们是否需要将模型个性化到每个用户? 我们多长时间重新训练一次模型? 一些模型需要每天或每周重新训练, 另一些则需要每月或每年。

1.5 评估

模型开发完成后的下一步是评估, 这是使用不同的指标来理解机器学习模型性能的过程。在本节中, 我们将讨论两种评估方法: 离线评估和在线评估。

1.5.1 离线评估

离线评估是指在模型开发阶段评估机器学习模型的性能。为了评估模型, 我们通常首先使用评估数据集进行预测。接下来, 我们使用各种离线指标来衡量预测值与真实值的接近程度。表1.4展示了常用于不同任务的指标。

表 1.4: 离线评估中的常用指标

Task	Offline metrics
Classification	Precision, recall, F1 score, accuracy, ROC-AUC, PR-AUC, confusion matrix
Regression	MSE, MAE, RMSE
Ranking	Precision@k, recall@k, MRR, mAP, nDCG
Image generation	FID 32 , Inception score 33
Natural language processing	BLEU 34 , METEOR 35 , ROUGE 36 , CIDEr 37 , SPICE 38

在面试中, 识别适合的离线评估指标很重要, 这取决于当前任务及我们对任务的定义方式。例如, 如果我们试图解决一个排序问题, 我们可能需要讨论排序指标及其权衡。

1.5.2 在线评估

在线评估是指在模型部署后, 评估其在生产环境中的表现。为了衡量模型的影响, 我们需要定义不同的指标。在线指标是指在在线评估中使用的指标, 通常与业务目标相关。表1.5展示了针对不同问题的各种指标。

表 1.5: 在线评估中的可能指标

Problem	Online metrics
Ad click prediction	Click-through rate, revenue lift, etc.
Harmful content detection	Prevalence, valid appeals, etc.
Video recommendation	Click-through rate, total watch time, number of completed videos, etc.
Friend recommendation	Number of requests sent per day, number of requests accepted per day, etc.

在实际操作中，公司通常会跟踪许多在线指标。在面试中，我们需要选择一些最重要的指标来衡量系统的影响。与离线指标不同，选择在线指标是主观的，并且依赖于产品负责人和业务利益相关者的意见。

在这个步骤中，面试官会评估你的业务敏感度。因此，清晰地表达你的思考过程以及选择某些指标的原因会很有帮助。

1.5.3 讨论要点

以下是一些关于评估步骤的讨论要点：

- **在线指标：**哪些指标对衡量机器学习系统在线效果很重要？这些指标与业务目标有何关系？
- **离线指标：**哪些离线指标在开发阶段评估模型的预测效果较好？
- **公平性和偏差：**模型是否可能在不同属性（如年龄、性别、种族等）上存在偏差？你会如何解决这个问题？如果有人恶意获取你的系统会发生什么？

1.6 部署与服务

为在线和离线评估选择合适的指标后的自然下一步是将模型部署到生产环境中，为数百万用户提供服务。需要涵盖的一些重要主题包括：

- 云端部署与设备端部署
- 模型压缩
- 生产环境中的测试
- 预测 Pipeline

让我们一一探讨。

1.6.1 云端部署与设备端部署

将模型部署到云端与部署到移动设备上有所不同。表1.6总结了设备端部署与云端部署之间的主要区别。

表 1.6: 云端与设备端部署的权衡

	Cloud	On-device
Simplicity	✓ Simple to deploy and manage using cloud-based services	✗ Deploying models on a device is not straightforward
Cost	✗ Cloud costs might be high	✓ No cloud cost when computations are performed on-device
Network latency	✗ Network latency is present	✓ No network latency
Inference latency	✓ Usually faster inference due to more powerful machines	✗ ML models run slower
Hardware constraints	✓ Fewer constraints	✗ More constraints, such as limited memory, battery consumption, etc.
Privacy	✗ Less privacy as user data is transferred to the cloud	✓ More privacy since data never leaves the device
Dependency on internet connection	✗ Internet connection needed to send and receive data to the cloud	✓ No internet connection needed

1.6.2 模型压缩

模型压缩是指使模型更小的过程。这对于减少推理延迟和模型大小是必要的。压缩模型的三种常用技术是：

- **知识蒸馏**: 知识蒸馏的目标是训练一个小模型（学生模型）来模仿一个大模型（教师模型）。
- **剪枝**: 剪枝是指找到最不重要的参数并将它们置零。这样可以得到更稀疏的模型，使存储更加高效。
- **量化**: 模型参数通常用 32 位浮点数表示。在量化过程中，我们使用更少的位数来表示参数，从而减小模型的大小。量化可以在训练过程中或训练后进行³⁹。

如果你想了解更多关于模型压缩的内容，可以参考⁴⁰。

1.6.3 生产环境中的测试

确保模型在生产环境中表现良好的唯一方法是用真实流量进行测试。常用的模型测试技术包括影子部署⁴¹、A/B 测试⁴²、金丝雀发布⁴³、交叉实验⁴⁴、多臂赌博机⁴⁵等。

为了展示我们对如何在生产中进行测试的理解，提到至少一种这些方法是个好主意。让我们简要回顾影子部署和 A/B 测试。

1.6.4 影子部署

在这种方法中，我们与现有模型并行部署新模型。每个传入请求都会同时路由到两个模型，但只有现有模型的预测会提供给用户。

通过影子部署模型，我们可以将不可靠预测的风险降到最低，直到新开发的模型经过彻底测试。然而，这种方法代价高昂，因为它使预测数量翻倍。

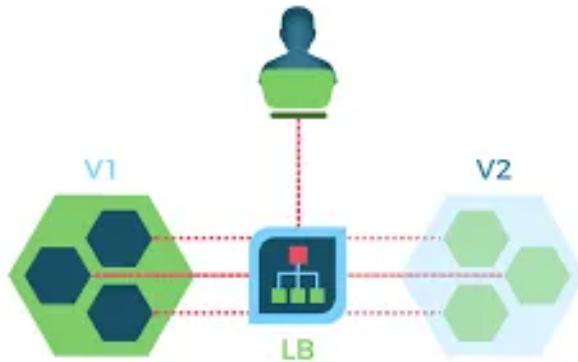


图 1.19: 影子部署

1.6.5 A/B 测试

使用这种方法，我们与现有模型并行部署新模型。部分流量会被路由到新开发的模型，而剩余请求则路由到现有模型。

为了正确执行 A/B 测试，有两个重要因素需要考虑。首先，路由到每个模型的流量必须是随机的。其次，A/B 测试应该在足够多的数据点上进行，以确保结果具有合法性。

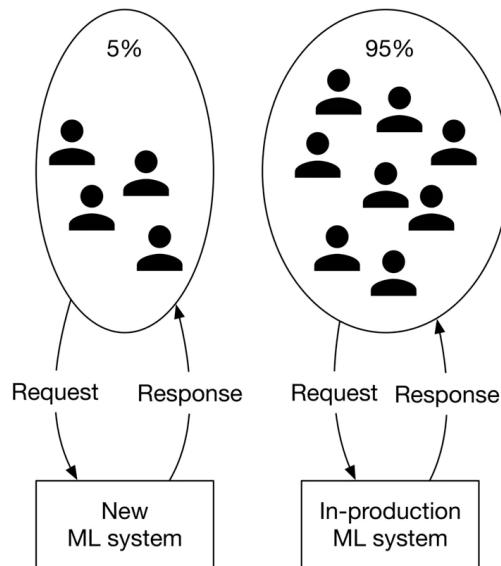


图 1.20: A/B 测试

1.6.6 预测 Pipeline

要在生产中服务请求，我们需要一个预测 Pipeline。一个重要的设计决策是在线预测与批量预测之间的选择。

批量预测。在批量预测中，模型会定期进行预测。由于预测是预先计算的，我们不需要担心模型生成预测时的时间问题。

然而，批量预测有两个主要缺点。首先，模型对用户偏好的变化反应不够灵敏。其次，只有当我们提前知道需要预计算的内容时，批量预测才可行。例如，在语言翻译系统中，我们无法提前进行翻译，因为这完全取决于用户的输入。

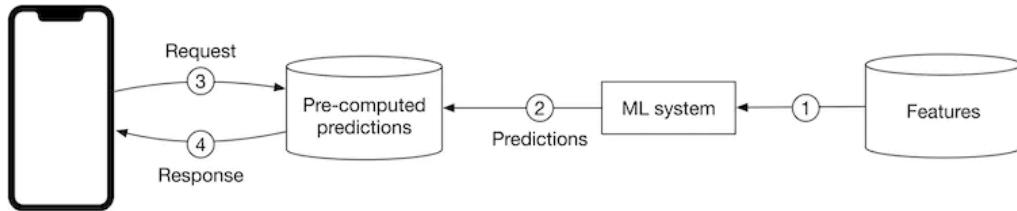


图 1.21: 批量预测工作流

在线预测。在线预测中，预测会在请求到达时立即生成并返回。在线预测的主要问题是模型可能需要较长时间来生成预测。

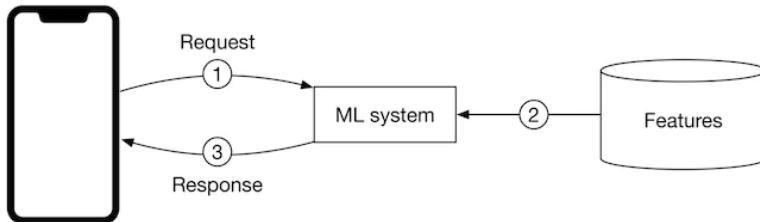


图 1.22: 在线预测工作流

批量预测与在线预测的选择主要取决于产品需求。当我们无法提前知道需要计算的内容时，通常更倾向于使用在线预测。批量预测则更适合于处理大量数据且不需要实时结果的系统。

如前所述，机器学习系统的开发不仅仅是机器学习建模。在面试中提出一个整体的机器学习系统设计，能够展示你对不同组件如何协同工作的深入理解。面试官往往将其视为一个重要的信号。

图1.23展示了个性化新闻推送系统的机器学习系统设计示例。我们将在第十章中对其进行更深入的探讨。

1.6.7 讨论要点

- 是否需要进行模型压缩？有哪些常用的压缩技术？
- 在线预测还是批量预测更合适？有哪些权衡？
- 实时访问特征是否可行？有哪些挑战？
- 我们应该如何测试已部署的模型？

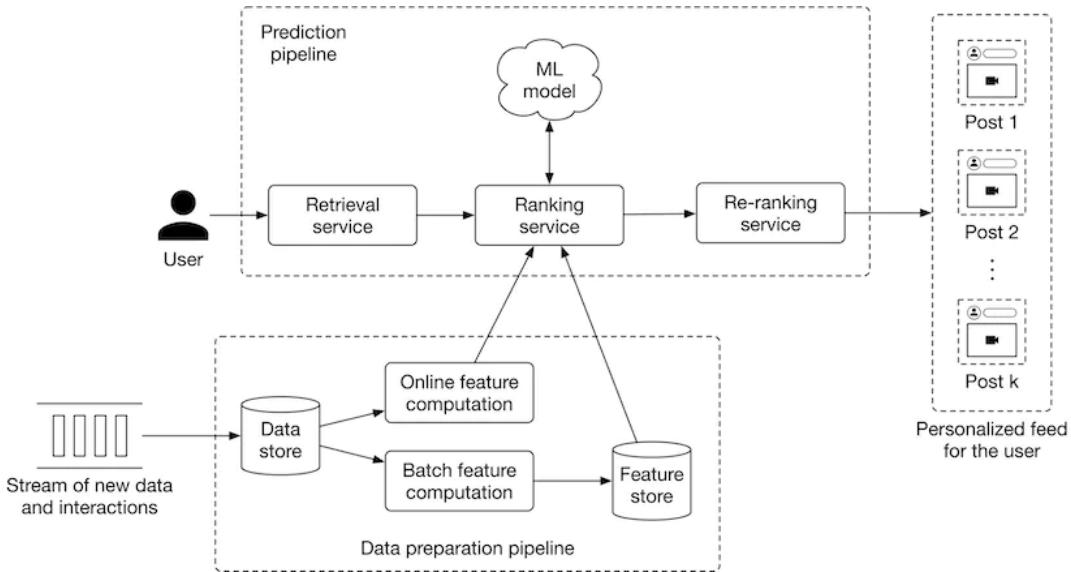


图 1.23: 个性化新闻推送系统的机器学习系统设计

- 一个机器学习系统由各种组件协同工作来处理请求。在建议的设计中，每个组件的职责是什么？
- 我们应该使用哪些技术来确保服务的速度和可扩展性？

1.7 监控

监控是指跟踪、测量和记录各种指标的任务。生产环境中的机器学习系统可能由于多种原因而失败。监控有助于在系统出现故障时及时检测，以便能够尽快修复。

在机器学习系统设计面试中，监控是一个重要的讨论话题。你可能需要讨论的两个主要领域是：

- 系统在生产中为何会失败
- 需要监控什么

1.7.1 系统在生产中为何会失败

机器学习系统在部署到生产环境后可能失败的原因有很多。最常见的原因之一是数据分布变化。

数据分布变化是指模型在生产环境中遇到的数据与训练时遇到的数据不同的情况。图1.24展示了一个例子，其中训练数据包括从正面看到的杯子图像，但在服务时，一个从不同角度看到的杯子图像被传递给机器学习模型。

现实世界中的数据分布不断变化。换句话说，随着时间的推移，训练时使用的数据可能会变得不再相关。这会导致模型陈旧，性能随时间下降。因此，我们应该持续监控系统以检测这一问题。应对数据分布变化的两种常见方法是：

- 使用大型数据集进行训练。足够大的训练数据集可以使模型学习到全面的分布，因此生产环境中遇到的任何数据点很可能来自这个学习到的分布。



图 1.24: 数据分布变化

- 定期使用新分布中的有标签数据重新训练模型。

1.7.2 需要监控什么

这是一个广泛的话题，这里我们关注生产后的监控技术。我们的目标是检测故障并识别机器学习系统中的变化。总体而言，我们可以将机器学习系统中的监控技术分为两类：与操作相关的指标和机器学习特定的指标。

与操作相关的指标：这些指标确保系统正常运行。它们包括平均服务时间、吞吐量、预测请求数量、CPU/GPU 利用率等。

机器学习特定的指标：

- 监控输入/输出。模型的表现取决于它所接收的数据，因此监控模型的输入和输出至关重要。
- 漂移。监控系统的输入和模型的输出，以检测它们底层分布的变化。
- 模型准确性。例如，我们期望准确性在特定范围内。
- 模型版本。监控部署的是哪个版本的模型。

1.8 基础设施

基础设施是训练、部署和维护机器学习系统的基础。

在许多机器学习面试中，你不会被问到与基础设施相关的问题。然而，一些机器学习职位，如 DevOps 和 MLOps，可能需要基础设施知识。因此，明确面试官对此主题的期望很重要。

基础设施是一个非常广泛的主题，不可能在几行之内总结。如果你有兴趣了解更多关于机器学习基础设施的内容，可以参考[46](#) [47](#) [48](#)。

1.9 总结

在本章中，我们提出了一个机器学习系统设计面试的框架。虽然本章讨论的许多话题是与特定任务相关的，但有些是通用的，适用于广泛的任务。为了避免重复，在整本书中，我们只关注与当前问题相

关的独特讨论点。例如，与部署、监控和基础设施相关的主题通常在不同任务中相似。因此，我们不会在后续章节中重复通用主题，但在面试中通常期望你会讨论这些内容。

最后，没有哪个工程师能在机器学习生命周期的各个方面都成为专家。有些工程师专注于部署和生产，而另一些则专注于模型开发。一些公司可能对基础设施不太关注，而另一些公司则非常重视监控和基础设施。数据科学职位通常需要更多的数据工程，而应用机器学习职位更关注模型开发和生产化。根据职位和面试官的偏好，有些步骤可能会被详细讨论，而另一些则可能会被简略讨论甚至跳过。一般来说，候选人应该尝试引导对话，同时在面试官提出问题时，能够跟随他们的节奏。

现在你已经了解了这些基础知识，我们准备开始解决一些最常见的机器学习系统设计面试问题。

References

1. Data warehouse. <https://cloud.google.com/learn/what-is-a-data-warehouse>.
2. Structured vs. unstructured data. <https://signal.onepointltd.com/post/102gjab/machine-learning-libra>
3. Bagging technique in ensemble learning. https://en.wikipedia.org/wiki/Bootstrap_aggregating.
4. Boosting technique in ensemble learning. <https://aws.amazon.com/what-is/boosting/>.
5. Stacking technique in ensemble learning. <https://machinelearningmastery.com/stacking-ensemble-machine-learning/>
6. Interpretability in Machine Learning. <https://blog.ml.cmu.edu/2020/08/31/6-interpretability/>.
7. Traditional machine learning algorithms. <https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/>
8. Sampling strategies. <https://www.scribbr.com/methodology/sampling-methods/>.
9. Data splitting techniques. <https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-models/>
10. Class-balanced loss. <https://arxiv.org/pdf/1901.05555.pdf>.
11. Focal loss paper. <https://arxiv.org/pdf/1708.02002.pdf>.
12. Focal loss. <https://medium.com/swlh/focal-loss-an-efficient-way-of-handling-class-imbalance-4855>
13. Data parallelism. <https://www.telesens.co/2017/12/25/understanding-data-parallelism-in-machine-learning/>
14. Model parallelism. <https://docs.aws.amazon.com/sagemaker/latest/dg/model-parallel-intro.html>.
15. Cross entropy loss. https://en.wikipedia.org/wiki/Cross_entropy.
16. Mean squared error loss. https://en.wikipedia.org/wiki/Mean_squared_error.
17. Mean absolute error loss. https://en.wikipedia.org/wiki/Mean_absolute_error.
18. Huber loss. https://en.wikipedia.org/wiki/Huber_loss.
19. L1 and L2 regularization. <https://www.analyticssteps.com/blogs/l2-and-l1-regularization-machine-learning>
20. Entropy regularization. <https://paperswithcode.com/method/entropy-regularization>.
21. K-fold cross validation. [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).
22. Dropout paper. <https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf>.
23. Overview of optimization algorithm. <https://ruder.io/optimizing-gradient-descent/>.
24. Stochastic gradient descent. https://en.wikipedia.org/wiki/Stochastic_gradient_descent.
25. AdaGrad optimization algorithm. <https://optimization.cbe.cornell.edu/index.php?title=AdaGrad>.

26. Momentum optimization algorithm. <https://optimization.cbe.cornell.edu/index.php?title=Momentum>.
27. RMSProp optimization algorithm. <https://optimization.cbe.cornell.edu/index.php?title=RMSProp>.
28. ELU activation function. https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#elu.
29. ReLU activation function. https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#relu.
30. Tanh activation function. https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#tanh.
31. Sigmoid activation function. https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#softmax.
32. FID score. https://en.wikipedia.org/wiki/Fr%C3%A9chet_inception_distance.
33. Inception score. https://en.wikipedia.org/wiki/Inception_score.
34. BLEU metrics. <https://en.wikipedia.org/wiki/BLEU>.
35. METEOR metrics. <https://en.wikipedia.org/wiki/METEOR>.
36. ROUGE score. [https://en.wikipedia.org/wiki/ROUGE_\(metric\)](https://en.wikipedia.org/wiki/ROUGE_(metric)).
37. CIDEr score. <https://arxiv.org/pdf/1411.5726.pdf>.
38. SPICE score. <https://arxiv.org/pdf/1607.08822.pdf>.
39. Quantization-aware training. <https://pytorch.org/docs/stable/quantization.html>.
40. Model compression survey. <https://arxiv.org/pdf/1710.09282.pdf>.
41. Shadow deployment. <https://christophergs.com/machine%20learning/2019/03/30/deploying-machine-learning-models-with-a-shadow-deployment/>.
42. A/B testing. https://en.wikipedia.org/wiki/A/B_testing.
43. Canary release. <https://blog.getambassador.io/cloud-native-patterns-canary-release-1cb8f82d371a>.
44. Interleaving experiment. <https://netflixtechblog.com/interleaving-in-online-experiments-at-netflix>.
45. Multi-armed bandit. <https://vwo.com/blog/multi-armed-bandit-algorithm/>.
46. ML infrastructure. <https://www.run.ai/guides/machine-learning-engineering/machine-learning-infrastructure>.
47. Interpretability in ML. <https://fullstackdeeplearning.com/spring2021/lecture-6/>.
48. Chip Huyen. *Designing Machine Learning Systems: An Iterative Process for Production-Ready Application*. O’ Reilly Media, Inc., 2022.

2 Visual Search System 视觉搜索系统

该系统帮助用户发现与所选图像在视觉上相似的图像。在本章中，我们设计了一个类似于 Pinterest 的视觉搜索系统 1 2。

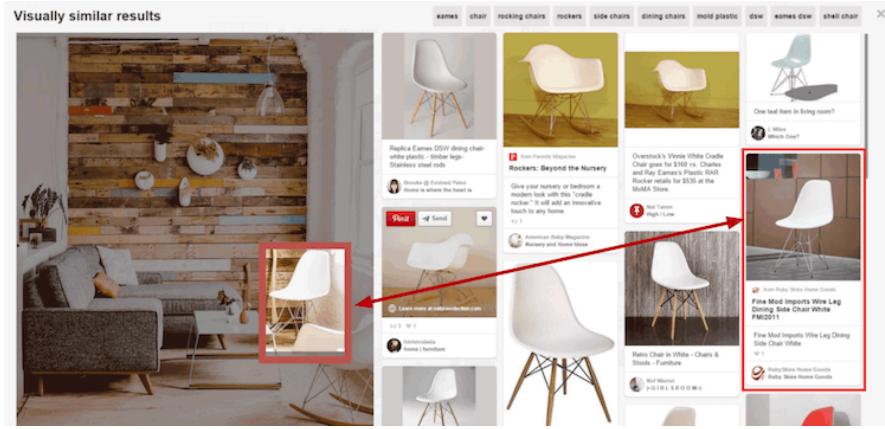


图 2.1: 检索到的与所选裁剪图像视觉上相似的图像

2.1 明确需求

以下是候选人与面试官之间的典型对话。

候选人: 我们是否需要将结果从最相似到最不相似排序?

面试官: 列表中最先出现的图像应该与查询图像更相似。

候选人: 系统是否也需要支持视频?

面试官: 我们只关注图像。

候选人: 像 Pinterest 这样的平台允许用户选择图像的一个部分并检索相似的图像。我们是否需要支持这种功能?

面试官: 是的。

候选人: 显示的图像是针对用户个性化的吗?

面试官: 为了简单起见，我们不关注个性化。无论是谁进行搜索，查询图像都会得到相同的结果。

候选人: 模型可以使用查询图像的元数据，例如图像标签吗?

面试官: 实际上，模型会使用图像的元数据。但为了简单起见，我们假设不依赖元数据，仅依赖图像像素。

候选人: 用户可以执行其他操作吗，例如保存、分享或点赞？这些操作可以帮助标记训练数据。

面试官: 这个观点很好。为了简单起见，我们假设唯一支持的操作是图像点击。

候选人: 我们可以在线构建训练数据，并根据用户交互进行标注。这是预期的构建训练数据的方式吗?

面试官: 是的，这听起来很合理。

候选人: 搜索速度需要多快？假设平台上 100-2000 亿张图片，系统应该能够快速检索到相似的图像。这个假设合理吗?

面试官: 是的，这是一个合理的假设。

让我们总结一下问题描述。我们需要设计一个视觉搜索系统。该系统根据用户提供的查询图像检索相似的图像，并根据它们与查询图像的相似度进行排序，然后显示给用户。平台仅支持图像，不允许视频或文本查询。为简化起见，不需要个性化。

2.2 将问题框定为机器学习任务

在本节中，我们选择一个明确的机器学习目标，并将视觉搜索问题框定为一个机器学习任务。

2.2.1 定义机器学习目标

为了使用机器学习模型解决这个问题，我们需要创建一个明确的机器学习目标。一个潜在的机器学习目标是准确地检索与用户搜索的图像在视觉上相似的图像。

2.2.2 指定系统的输入和输出

视觉搜索系统的输入是用户提供的查询图像。系统的输出是与查询图像在视觉上相似的图像，并按相似度对输出图像进行排序。图 2.2 展示了视觉搜索系统的输入和输出。

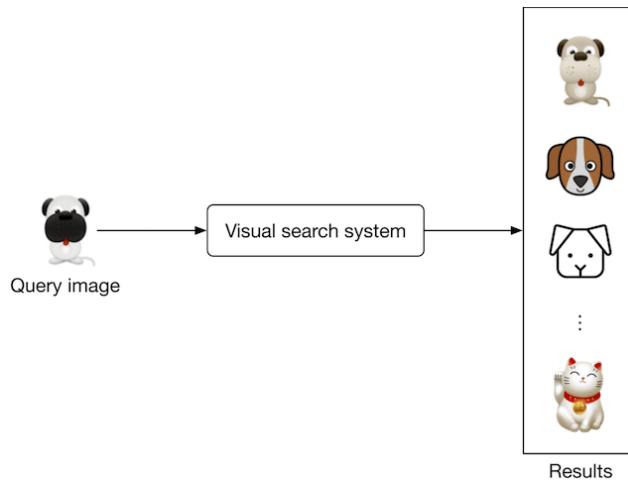


图 2.2: 视觉搜索系统的输入输出

2.2.3 选择正确的机器学习类别

模型的输出是一组与查询图像相似的排序图像。因此，视觉搜索系统可以被框定为一个排序问题。通常，排序问题的目标是根据与查询的相关性对一组项目（如图像、网站、产品等）进行排序，使得更相关的项目在搜索结果中排名更高。许多机器学习应用，如推荐系统、搜索引擎、文档检索和在线广告，都可以被框定为排序问题。在本章中，我们将使用一种广泛使用的方法，称为表示学习。让我们更详细地探讨这一点。

表示学习。在表示学习 3 中，模型被训练将输入数据（如图像）转换为称为嵌入的表示。另一种描述方式是，模型将输入图像映射到称为嵌入空间的 N 维空间中的点。学习这些嵌入，使得相似的图像在空间中具有彼此接近的嵌入。图 2.3 展示了两个相似的图像如何被映射到嵌入空间中的两个接近的点。为

演示起见，我们在一个二维空间中可视化图像嵌入（以'xxx' 表示）。实际上，这个空间是 N 维的，其中 N 是嵌入向量的大小。

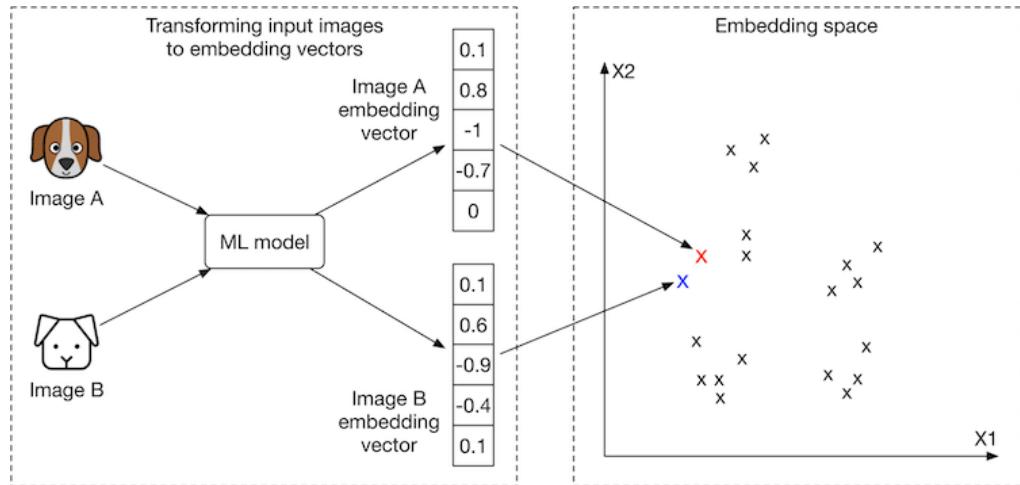


图 2.3: 嵌入空间中的相似图像

2.2.4 如何使用表示学习对图像进行排序？

首先，输入图像被转换为嵌入向量。接下来，通过测量查询图像与平台上其他图像在嵌入空间中的距离来计算它们的相似度分数。图像根据相似度分数进行排序，如图 2.4 所示。

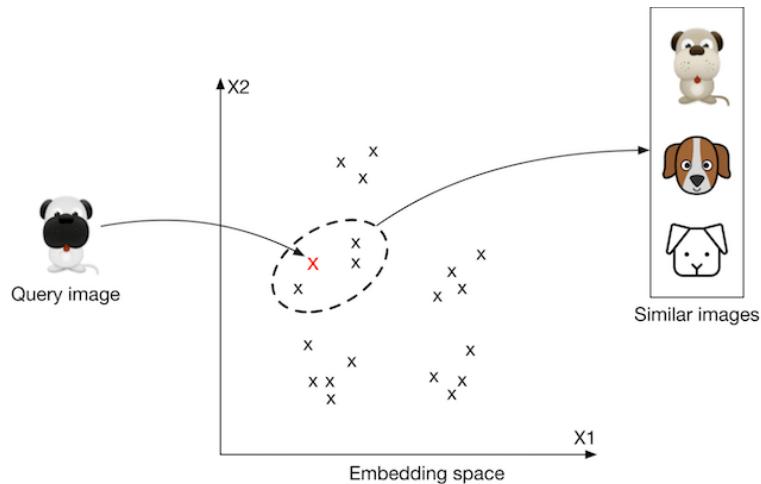


图 2.4: 与查询图像最相似的前三张图像

到目前为止，你可能会有很多问题，包括如何确保相似图像在嵌入空间中彼此靠近，如何定义相似度，以及如何训练这样的模型。我们将在模型开发部分进一步讨论这些问题。

2.3 数据准备

2.3.1 数据工程

除了通用的数据工程基础知识外，了解可用的数据也很重要。由于视觉搜索系统主要关注用户和图像，我们有以下可用数据：

图像、用户、用户-图像交互

图像 创作者上传图像，系统存储这些图像及其元数据，如所有者 ID、上下文信息（如上传时间）、标签等。表 2.1 展示了图像元数据的简化示例。

ID	Owner ID	Upload time	Manual tags
1	8	1658451341	Zebra
2	5	1658451841	Pasta, Food, Kitchen
3	19	1658821820	Children, Family, Party

表 2.1: 图像元数据

用户 用户数据包含与用户相关的人口统计属性，如年龄、性别等。表 2.2 展示了用户数据的示例。

ID	Username	Age	Gender	City	Country	Email
1	johnduo	26	M	San Jose	USA	john@gmail.com
2	hs2008	49	M	Paris	France	hsieh@gmail.com
3	alexish	16	F	Rio	Brazil	alexh@yahoo.com

表 2.2: 用户数据

用户-图像交互 交互数据包含不同类型的用户交互。根据收集到的需求，主要的交互类型是展示和点击。

表 2.3 展示了交互数据的概述。

User ID	Query image ID	Displayed image ID	Position in the displayed list	Interaction type	Location (lat, long)	Timestamp
8	2	6	1	Click	38.8951 -77.0364	1658450539
6	3	9	2	Click	38.8951 -77.0364	1658451341
91	5	1	2	Impression	41.9241 -89.0389	1658451365

表 2.3: 用户-图像交互数据

2.3.2 特征工程

在这一部分，你需要讨论如何设计良好的特征，并将其准备为模型输入。这通常取决于我们如何框定任务以及模型的输入是什么。在前面的“将问题框定为机器学习任务”部分中，我们将视觉搜索系统框定为一个排序问题，并使用表示学习来解决它。特别是，我们使用了一个期望图像作为输入的模型。图像在传递给模型之前需要进行预处理。让我们来看一些常见的图像预处理操作：

- **调整大小:** 模型通常需要固定的图像尺寸（例如， 224×224 ）
- **缩放:** 将图像的像素值缩放到 0 和 1 的范围内
- **Z-score 归一化:** 将像素值缩放为均值为 0，方差为 1
- **颜色一致性:** 确保图像具有一致的颜色模式（例如，RGB 或 CMYK）

2.4 模型开发

2.4.1 模型选择

我们选择神经网络的原因是：

- 神经网络擅长处理非结构化数据，例如图像和文本。
- 与许多传统机器学习模型不同，神经网络能够生成我们在表示学习中所需的嵌入。

我们应该使用哪种神经网络架构？架构必须适用于图像。基于 CNN 的架构，如 ResNet 4，或者更新的基于 Transformer 的架构 5，如 ViT 6，在处理图像输入时表现良好。图 2.5 展示了一个将输入图像转换为嵌入向量的简化模型架构。卷积层的数量、全连接层中的神经元数量以及嵌入向量的大小是通常通过实验选择的超参数。

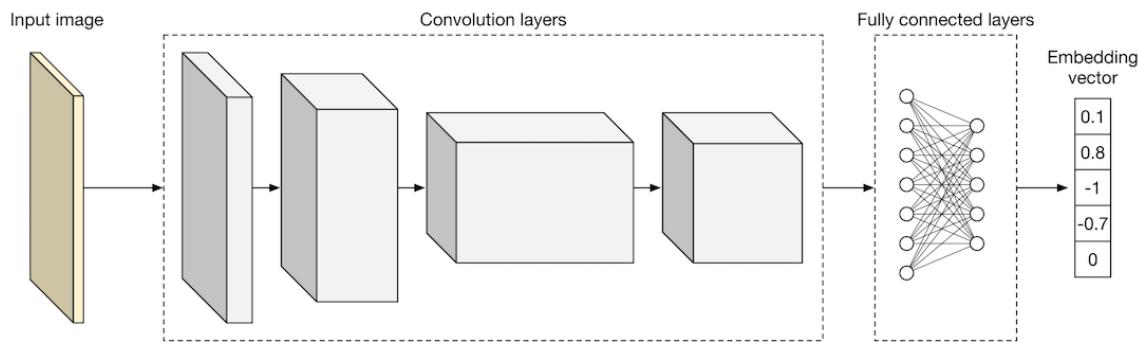


图 2.5: 简化的模型架构

2.4.2 模型训练

为了检索视觉上相似的图像，模型必须在训练过程中学习表示（嵌入）。在本节中，我们讨论如何训练模型以学习图像表示。

一种常见的学习图像表示的技术是对比训练 7。使用这种技术，我们训练模型以区分相似和不相似的图像。如图 2.6 所示，我们为模型提供一个查询图像（左），一个与查询图像相似的图像（右侧高亮的狗图像），以及一些不相似的图像（同样在右侧）。在训练过程中，模型学习生成的表示使得相似图像比图 2.6 右侧的其他图像更接近查询图像。

要使用对比训练技术训练模型，我们首先需要构建训练数据。

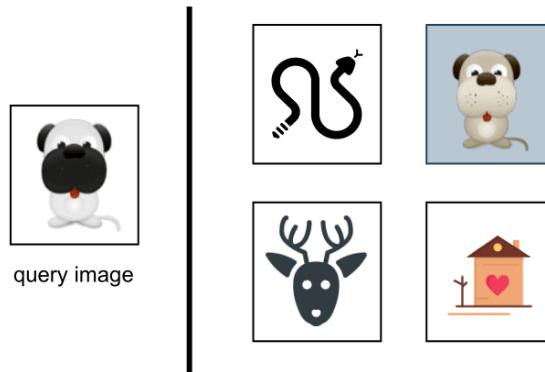


图 2.6: 对比训练

2.4.3 构建数据集

如前所述，每个用于训练的数据点包含一个查询图像、一个与查询图像相似的正样本图像，以及 $n - 1$ 个与查询图像不相似的负样本图像。数据点的真实标签是正样本图像的索引。如图 2.7 所示，除了查询图像 (q) 之外，我们还有 n 个其他图像，其中一个与 q 相似（狗的图像），其他 $n - 1$ 个图像是不相似的。该数据点的真实标签是正样本图像的索引，即 2（图 2.7 中 n 个图像中的第二张图像）。

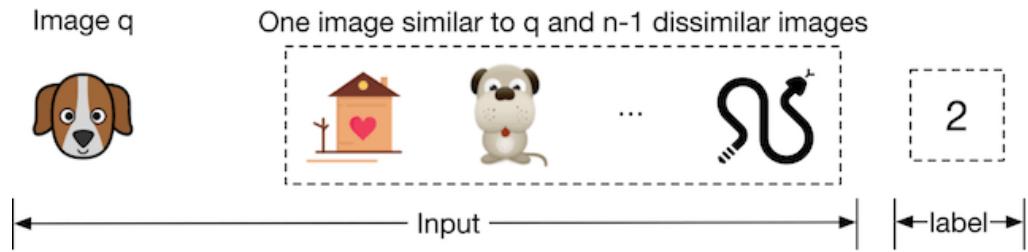


图 2.7: 训练数据点

为了构建一个训练数据点，我们随机选择一个查询图像和 $n - 1$ 个负样本图像。选择正样本图像时，我们有以下三种选择：

- 使用人工判断
- 使用用户点击等交互数据作为相似性的代理
- 人工生成与查询图像相似的图像，即自监督

让我们评估每种选择。

使用人工判断 这种方法依赖人工标注者手动查找相似图像。人工参与能够产生准确的训练数据，但使用人工标注成本高且耗时。

使用用户点击等交互数据作为相似性的代理 在这种方法中，我们基于交互数据来衡量相似性。例如，当用户点击某个图像时，点击的图像被视为与查询图像 q 相似。

这种方法不需要人工操作，可以自动生成训练数据。然而，点击信号通常噪声很大。用户有时即使在图像与查询图像不相似时也会点击。此外，这种数据非常稀疏，对于很多图像我们可能没有点击数据。使用噪声大且稀疏的训练数据会导致较差的性能。

人工生成与查询图像相似的图像 在这种方法中，我们人工生成一个与查询图像相似的图像。例如，我们可以通过旋转查询图像进行增强，并将新生成的图像作为相似图像。最近开发的框架，如 SimCLR 7 和 MoCo 8，采用了相同的方法。

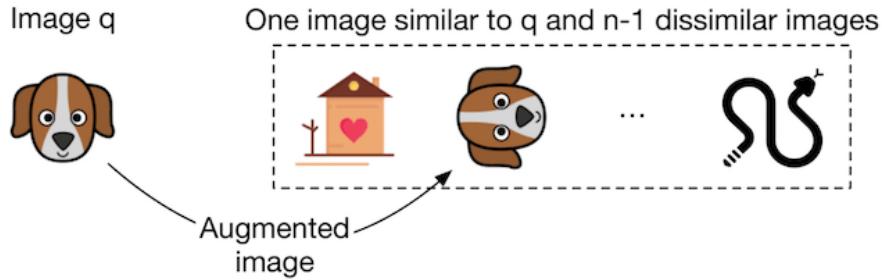


图 2.8: 使用数据增强创建相似图像

这种方法的优势在于不需要人工操作。我们可以通过简单的数据增强逻辑来创建相似图像。此外，构建的训练数据没有噪声，因为增强图像总是与原始图像相似。这种方法的主要缺点是构建的训练数据与真实数据不同。在实际操作中，相似图像并不是查询图像的增强版本；它们在视觉和语义上相似，但却是不同的。

哪种方法在我们的情况下效果最好？ 在面试中，提出各种选择并讨论它们的权衡是至关重要的。通常情况下，没有一种方法总是最佳的。在这里，我们选择自监督选项有两个原因。首先，它不需要前期成本，因为该过程可以自动化。其次，像 SimCLR 7 这样的框架在大型数据集上训练时表现出了良好的结果。由于我们可以访问平台上的数十亿张图片，这种方法可能是一个不错的选择。

如果实验结果不理想，我们可以随时切换到其他标注方法。例如，我们可以从自监督选项开始，之后使用点击数据进行标注。我们也可以结合不同的选项。例如，我们可以使用点击数据构建初始训练数据，并依赖人工标注者识别和删除噪声数据点。与面试官讨论不同的选择和权衡对做出良好的设计决策至关重要。

一旦构建了数据集，就可以使用合适的损失函数训练模型。

2.4.4 选择损失函数

如图 2.9 所示，模型将图像作为输入，并为每个输入图像生成一个嵌入。 E_x 表示图像 x 的嵌入。

训练的目标是优化模型参数，使相似图像在嵌入空间中彼此靠近。如图 2.10 所示，正样本图像和查询图像在训练过程中应变得更加接近。

为了实现这一目标，我们需要使用损失函数来衡量生成的嵌入的质量。不同的损失函数是为对比训练设计的，面试官通常不会期望你深入讨论。然而，了解对比损失函数的工作原理是很重要的。

我们将简要讨论简化的对比损失的工作方式。如果你对对比损失有更多兴趣，可以参考 9。

如图 2.11 所示，我们在三个步骤中计算对比损失。

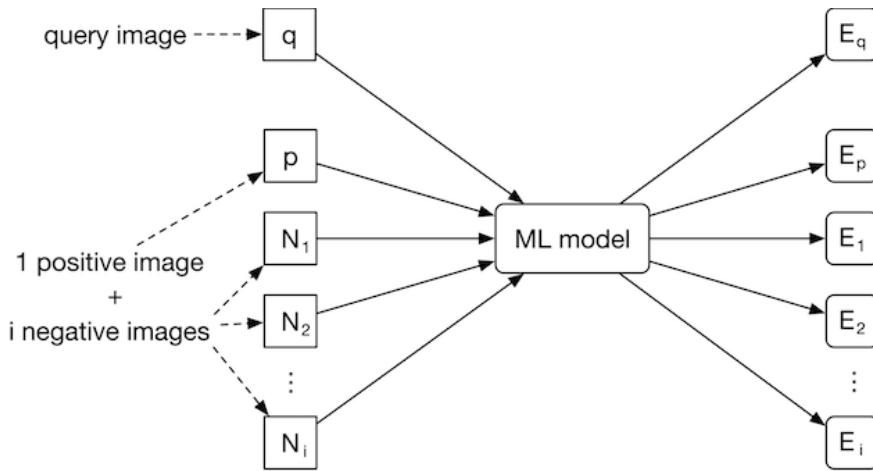


图 2.9: 模型输入输出

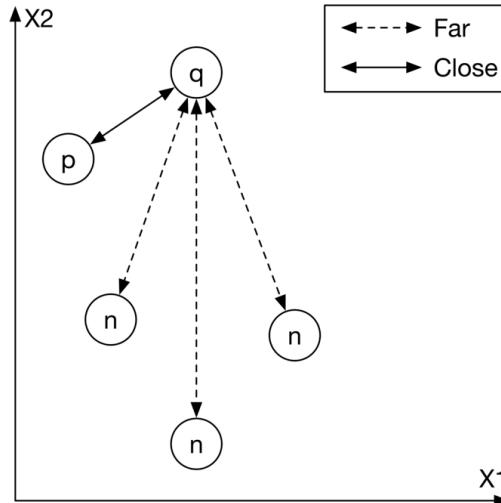


图 2.10: 输入图像映射到嵌入空间中

计算相似度。首先，我们计算查询图像与其他图像嵌入之间的相似度。点积 10 和余弦相似度 11 广泛用于测量嵌入空间中点之间的相似度。欧几里得距离 12 也可以用来测量相似度。然而，欧几里得距离在高维空间中通常表现较差，这是由于维度灾难 13 的影响。想要了解更多关于维度灾难的问题，可以参考 14。

Softmax。在计算的距离上应用 Softmax 函数。这确保了值的总和为 1，从而可以将这些值解释为概率。

交叉熵。交叉熵 15 衡量预测概率与真实标签的接近程度。当预测概率接近真实标签时，说明嵌入足以区分正样本图像和负样本图像。

在面试中，你也可以讨论使用预训练模型的可能性。例如，我们可以利用一个预训练的对比模型，并使用训练数据对其进行微调。这些预训练模型已经在大型数据集上进行了训练，因此它们学到了良好的图像表示。与从头开始训练模型相比，这可以显著减少训练时间。

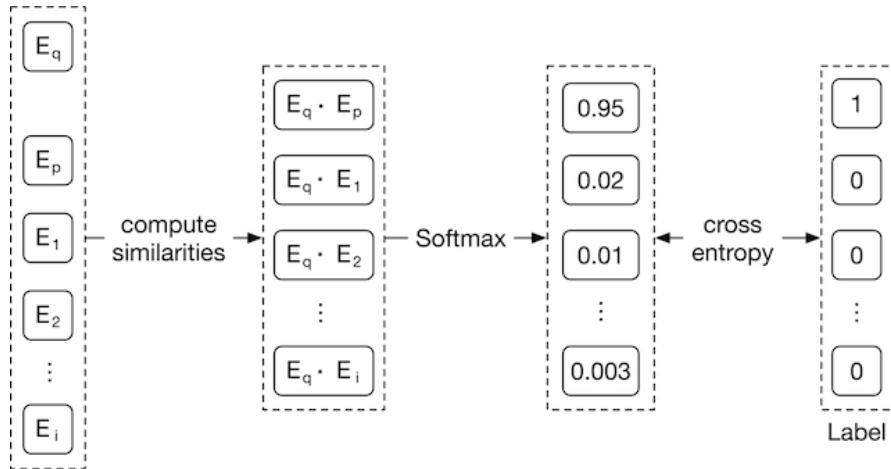


图 2.11: 简化的对比损失

2.5 评估

在我们开发完模型后，可以讨论评估。在本节中，我们将介绍离线和在线评估的重要指标。

2.5.1 离线指标

根据给定的需求，评估数据集可用于离线评估。假设每个数据点都有一个查询图像、一些候选图像，以及每个候选图像和查询图像对的相似度得分。相似度得分是一个介于 0 到 5 之间的整数，其中 0 表示没有相似性，5 表示两个图像在视觉上和语义上非常相似。对于评估数据集中的每个数据点，我们将模型生成的排序与基于真实得分的理想排序进行比较。

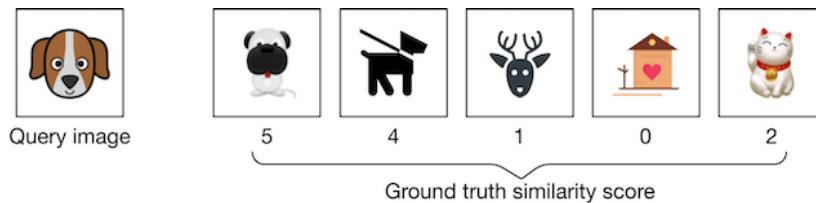


图 2.12: 评估数据集中的一个数据点

现在，让我们来看看搜索系统中常用的离线指标。注意，搜索、信息检索和推荐系统通常共享相同的离线指标。

- 平均倒数排名 (MRR)
- 召回率 @k
- 准确率 @k
- 平均准确率 (mAP)
- 归一化折扣累积增益 (nDCG)

MRR. 该指标通过考虑模型生成的每个输出列表中第一个相关项的排名，并对它们进行平均来衡量模型的质量。公式为：

$$\text{MRR} = \frac{1}{m} \sum_{i=1}^m \frac{1}{\text{rank}_i}$$

其中， m 表示输出列表的总数， rank_i 表示第 i 个输出列表中第一个相关项的排名。

图 2.13 展示了该指标的工作原理。对于每个排序列表，我们计算其倒数排名 (RR)，然后计算 RRs 的平均值以得到 MRR。

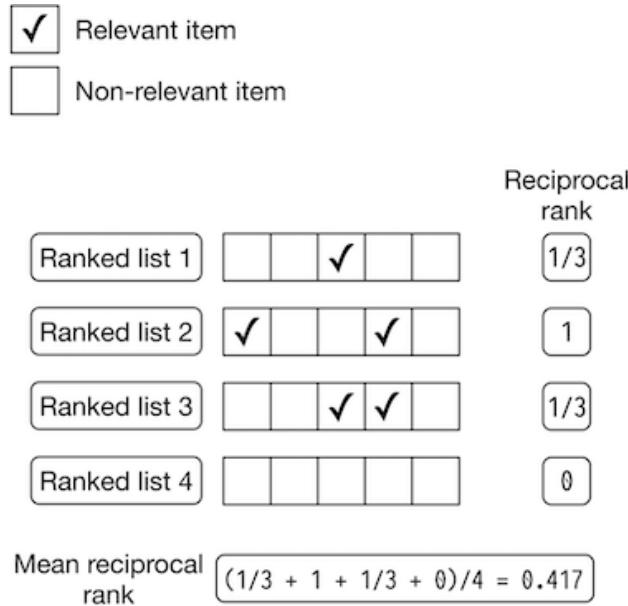


图 2.13: MRR 计算示例

让我们看看这个指标的缺点。由于 MRR 只考虑第一个相关项，忽略了列表中的其他相关项，因此它无法衡量排序列表的精确度和排名质量。例如，图 2.14 展示了两个不同模型的输出。模型 1 的输出有 3 个相关项，而模型 2 的输出只有 1 个相关项。然而，这两个模型的倒数排名都是 0.5。鉴于这一缺点，我们不会使用这个指标。

	MRR
Model 1 output	$1/2$
Model 2 output	$1/2$

图 2.14: 两个不同模型的 MRR

召回率 @k. 该指标衡量输出列表中相关项的数量与整个数据集中可用的相关项总数之间的比率。公式为：

$$\text{recall}@k = \frac{\text{number of relevant items among the top } k \text{ items in the output list}}{\text{total relevant items}}$$

尽管召回率 @ k 衡量了模型未包含在输出列表中的相关项数量，但这并不总是一个好的指标。让我们来理解为什么。在某些系统（如搜索引擎）中，相关项的总数可能非常高。这会对召回率产生负面影响，因为分母非常大。例如，当查询图像是一张狗的图片时，数据库中可能包含数百万张狗的图片。我们的目标不是返回每一张狗的图片，而是检索几张最相似的狗图片。

鉴于召回率 @ k 不能衡量模型的排名质量，我们不会使用它。

准确率 @ k . 该指标衡量输出列表中前 k 项中相关项的比例。公式为：

$$\text{precision}@k = \frac{\text{number of relevant items among the top } k \text{ items in the output list}}{k}$$

这个指标衡量输出列表的精确度，但它不考虑排名质量。例如，在图 2.15 中，如果我们在列表中将更多的相关项排在前面，精确度不会发生变化。这个指标并不适合我们的使用场景，因为我们需要同时衡量结果的精确度和排名质量。

					Precision@5
Model 1 output	✓	✓			2/5
Model 2 output		✓		✓	2/5

图 2.15: 两个不同模型的精确率 @5

mAP. 该指标首先计算每个输出列表的平均精确度 (AP)，然后对 AP 值进行平均。

让我们首先理解 AP 是什么。它接受一个包含 k 项（如图像）的列表，并在不同的 k 值下对精确度 @ k 进行平均。如果更多的相关项位于列表的顶部，AP 就会更高。对于大小为 k 的列表，AP 的公式为：

$$\text{AP} = \frac{\sum_{i=1}^k \text{Precision}@i \text{ if } i\text{'th item is relevant to the user}}{\text{total relevant items}}$$

让我们通过一个示例更好地理解这个指标。图 2.16 显示了模型生成的每个输出列表的 AP 计算。

由于我们对精确度进行了平均，因此考虑了列表的整体排名质量。然而，mAP 设计用于二元相关性；换句话说，当每个项要么相关要么不相关时，它表现良好。对于连续相关性得分，nDCG 是更好的选择。

nDCG. 该指标衡量输出列表的排名质量，并显示与理想排名相比，排名的好坏。首先，让我们解释 DCG，然后讨论 nDCG。

什么是 DCG？ DCG 通过对列表中每个项的相关性得分进行累加来计算列表的累积增益。然后，得分从输出列表的顶部开始累加，并对较低排名的结果进行折扣。公式为：

$$\text{DCG}_p = \sum_{i=1}^p \frac{\text{rel}_i}{\log_2(i+1)}$$

其中， rel_i 是排名在位置 i 处的图像的真实相关性得分。

什么是 nDCG？ 由于 DCG 对项的相关性得分进行累加并对其位置进行折扣，DCG 的结果可能是任意值。为了获得更有意义的得分，我们需要对 DCG 进行归一化。为此，nDCG 将 DCG 除以理想排名的 DCG。公式为：

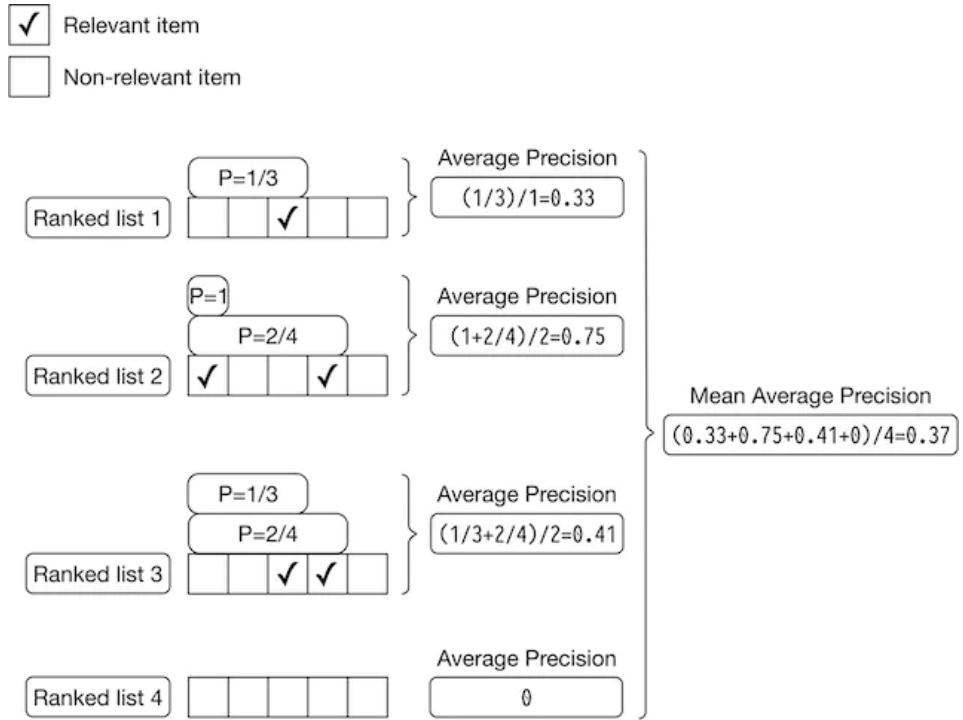


图 2.16: mAP 计算

$$\text{nDCG}_p = \frac{\text{DCG}_p}{\text{IDCG}_p}$$

其中, IDCG_p 是理想排名的 DCG。请注意, 在一个完美的排序系统中, DCG 等于 IDCG。

让我们通过一个示例更好地理解 nDCG。在图 2.17 中, 我们可以看到一个搜索系统生成的输出图像列表及其关联的真实相关性得分。

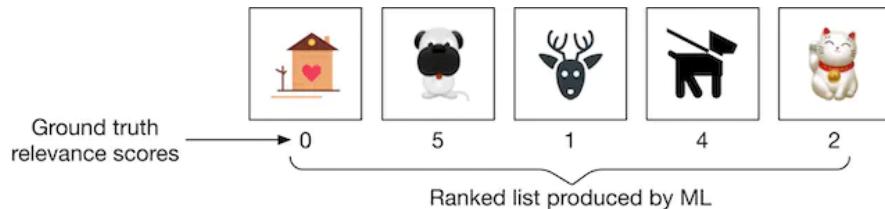


图 2.17: 搜索系统生成的排序列表

我们可以通过 3 个步骤来计算 nDCG:

1. 计算 DCG
2. 计算 IDCG
3. 将 DCG 除以 IDCG

计算 DCG: 模型生成的当前排序的 DCG 为:

$$\text{DCG}_p = \sum_{i=1}^p \frac{\text{rel}_i}{\log_2(i+1)} = \frac{0}{\log_2(2)} + \frac{5}{\log_2(3)} + \frac{1}{\log_2(4)} + \frac{4}{\log_2(5)} + \frac{2}{\log_2(6)} = 6.151$$

计算 IDCG: 理想排名的计算与 DCG 计算相同，只不过它首先推荐最相关的项（图 2.18）。

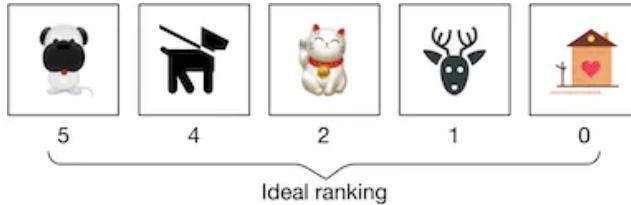


图 2.18: 理想排名列表

理想排名的 IDCG 为：

$$\text{IDCG}_p = \sum_{i=1}^p \frac{\text{rel}_i}{\log_2(i+1)} = \frac{5}{\log_2(2)} + \frac{4}{\log_2(3)} + \frac{2}{\log_2(4)} + \frac{1}{\log_2(5)} + \frac{0}{\log_2(6)} = 8.9543$$

将 DCG 除以 IDCG:

$$\text{nDCG}_p = \frac{\text{DCG}_p}{\text{IDCG}_p} = \frac{6.151}{8.9543} = 0.6869$$

nDCG 在大多数情况下表现良好。其主要缺点是，获取真实相关性得分并不总是可能的。在我们的案例中，由于评估数据集中包含相似性得分，我们可以使用 nDCG 来衡量模型在离线评估中的表现。

2.5.2 在线指标

在本节中，我们探讨了一些用于衡量用户多快能发现他们喜欢的图像的常用在线指标。

点击率 (CTR)。 该指标显示用户点击显示项的频率。CTR 可以使用以下公式计算：

$$\text{CTR} = \frac{\text{Number of clicked images}}{\text{Total number of suggested images}}$$

较高的 CTR 表示用户经常点击显示的项。CTR 通常在搜索和推荐系统中用作在线指标，后续章节将对此进行更详细的讨论。

建议图像的日均、周均和月均花费时间。 该指标显示用户对建议图像的参与度。当搜索系统准确时，我们期望此指标增加。

2.6 服务

在服务时，系统根据查询图像返回一个排序的相似图像列表。图 2.19 显示了预测 Pipeline 和索引 Pipeline。让我们更仔细地看一下每个 Pipeline。

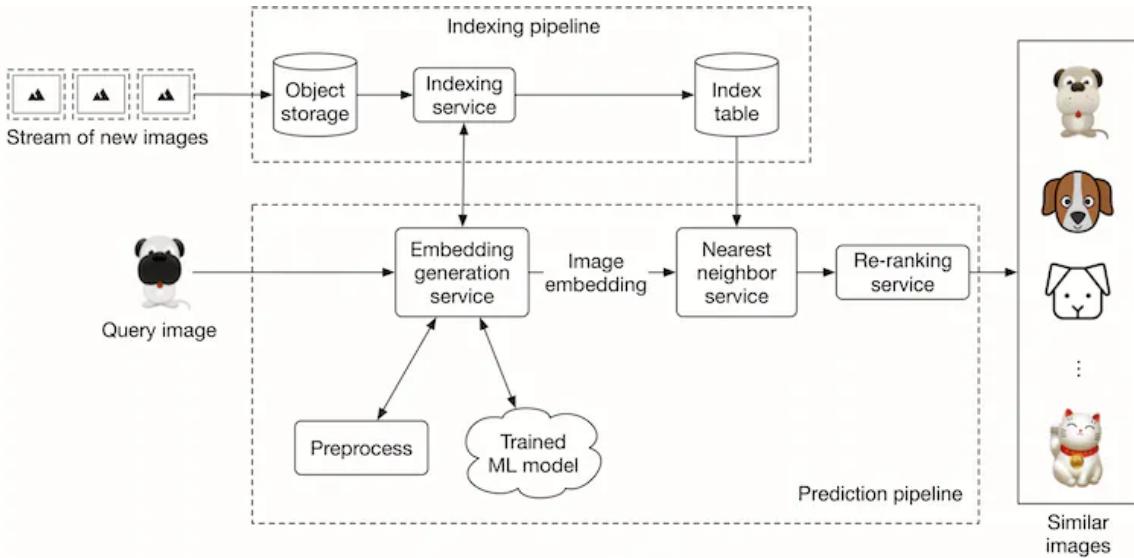


图 2.19: 预测和索引 Pipeline

2.6.1 预测 Pipeline

2.6.1.1 嵌入生成服务

该服务计算输入查询图像的嵌入。如图 2.20 所示，它对图像进行预处理并使用训练好的模型来确定嵌入。

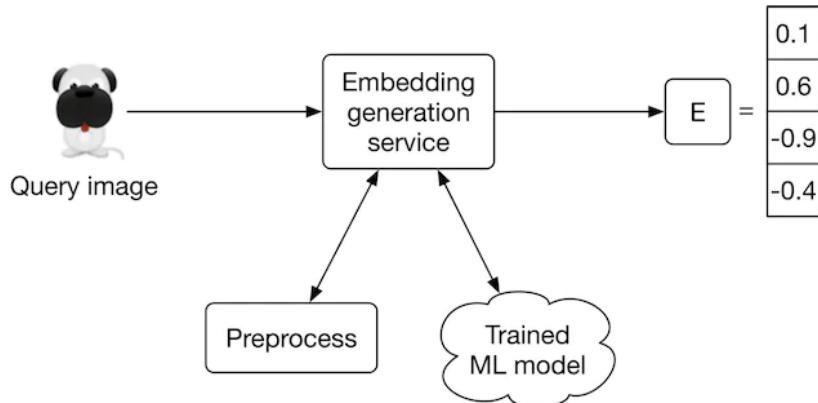


图 2.20: 嵌入生成服务

2.6.1.2 最近邻服务

一旦我们得到了查询图像的嵌入，就需要从嵌入空间中检索相似的图像。最近邻服务执行此操作。

让我们更正式地定义最近邻搜索。给定一个查询点 “ q ” 和一组其他点 S ，它在集合 S 中找到最接近 “ q ” 的点。请注意，图像嵌入是 N 维空间中的一个点，其中 N 是嵌入向量的大小。图 2.21 显示了图像 q 的前 3 个最近邻。我们将查询图像表示为 q ，其他图像表示为 xxx 。

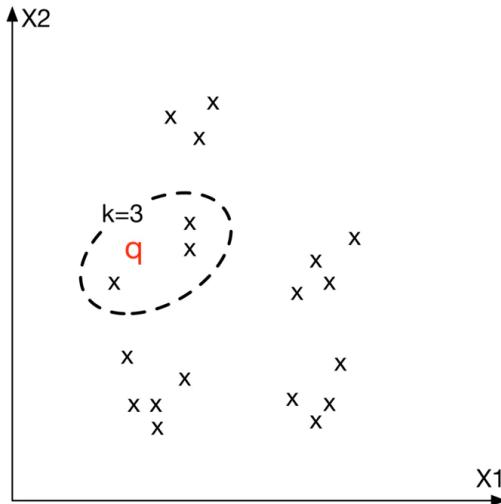


图 2.21: 嵌入空间中与图像 q 最接近的前 3 个邻居

2.6.1.3 重排序服务

该服务结合了业务层面的逻辑和策略。例如，它过滤不适当的结果，确保不包含私密图像，去除重复或近似重复的结果，并在向用户显示最终结果之前执行其他类似逻辑。

2.6.2 索引 Pipeline

索引服务 平台上的所有图像都由该服务进行索引，以提高搜索性能。

索引服务的另一个职责是保持索引表的更新。例如，当创作者向平台添加新图像时，该服务会对新图像的嵌入进行索引，以便最近邻搜索能够找到它。

索引会增加内存使用，因为我们在索引表中存储了所有图像的嵌入。有多种优化方法可用于减少内存使用，例如向量量化 [16](#) 和产品量化 [17](#)。

2.6.3 最近邻 (NN) 算法的性能

最近邻搜索是信息检索、搜索和推荐系统的核心组件。其效率的微小改进会带来显著的整体性能提升。鉴于这一组件的重要性，面试官可能会希望你深入探讨这个主题。

NN 算法可以分为两类：精确和近似。让我们更详细地探讨每种情况。

2.6.3.1 精确最近邻

精确最近邻，也称为线性搜索，是 NN 的最简单形式。它通过搜索整个索引表，计算每个点与查询点 q 的距离，并检索最近的 k 个点。时间复杂度为 $O(N \times D)$ ，其中 N 是点的总数， D 是点的维度。

在 N 可能轻松达到数十亿的大规模系统中，线性时间复杂度太慢。

2.6.3.2 近似最近邻 (ANN)

在许多应用中，展示足够相似的项给用户就足够了，没有必要执行精确的最近邻搜索。

在 ANN 算法中，使用特定的数据结构来将 NN 搜索的时间复杂度降低到次线性（例如， $O(D \times \log N)$ ）。它们通常需要前期的预处理或额外的空间。

ANN 算法可以分为以下三类：

- 基于树的 ANN
- 基于局部敏感哈希（LSH）的 ANN
- 基于聚类的 ANN

每个类别中都有各种算法，面试官通常不会期望你了解每一个细节。具备它们的高层次理解就足够了。因此，让我们简要介绍每个类别。

2.6.3.3 基于树的 ANN

基于树的算法通过将空间划分为多个部分来形成树。然后，它们利用树的特性进行更快的搜索。

我们通过逐步向每个节点添加新标准来形成树。例如，根节点的一个标准可以是：gender = male。这意味着具有女性属性的点属于左子树。

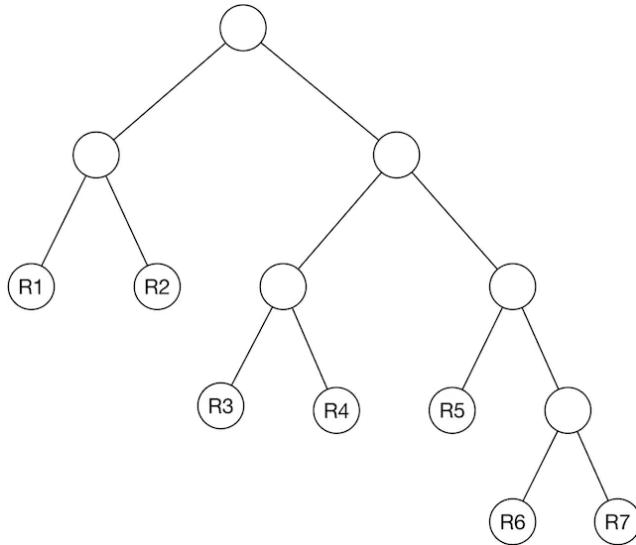


图 2.22: 从点形成的树

在树中，非叶节点根据标准将空间划分为两个部分。叶节点表示空间中的特定区域。图 2.23 显示了一个分为 7 个区域的空间示例。算法只搜索查询点所属的分区。

典型的基于树的方法包括 R-树 [18](#)、Kd-树 [19](#) 和 Annoy (Approximate Nearest Neighbor Oh Yeah) [20](#)。

2.6.3.4 基于局部敏感哈希 (LOCALITY SENSITIVE HASHING, LSH)

LSH 使用特定的哈希函数来减少点的维度，并将它们分组到桶 (bucket) 中。这些哈希函数将彼此靠近的点映射到同一个桶中。LSH 仅搜索属于与查询点 q 同一个桶中的点。你可以通过阅读 [21](#) 了解更多关于 LSH 的信息。

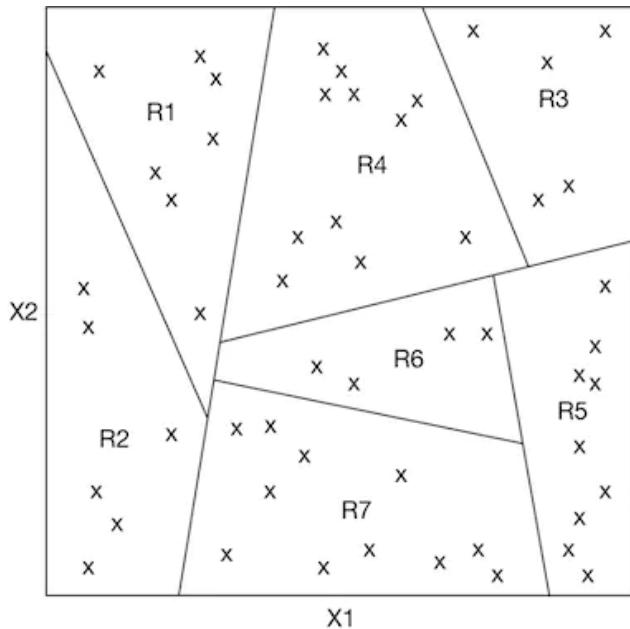


图 2.23: 树划分的空间

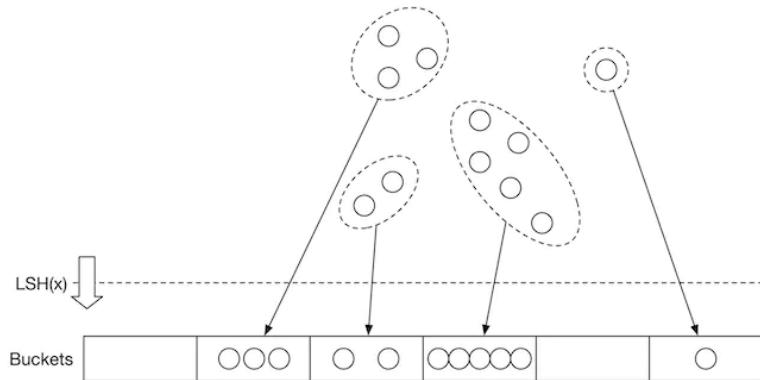


图 2.24: 使用 LSH 将数据点分组到 buckets 中

基于聚类的 ANN 这些算法通过根据相似性将点进行分组来形成聚类。一旦形成了聚类，算法只搜索查询点所属的聚类中的点子集。

2.6.3.5 我们应该使用哪种算法？

精确最近邻方法的结果保证是准确的。这使它成为当数据点有限或需要精确最近邻时的良好选择。然而，当点的数量很多时，运行该算法效率较低。在这种情况下，通常使用 ANN 方法。尽管它们可能不会返回精确的点，但它们在找到最近的点时更高效。

鉴于当今系统中可用的数据量，ANN 方法是更务实的解决方案。在我们的视觉搜索系统中，我们使用 ANN 来查找相似的图像嵌入。

对于应用型机器学习角色，面试官可能会要求你实现 ANN。两个广泛使用的库是 Faiss [22](#)（由 Meta 开发）和 ScaNN [23](#)（由 Google 开发）。每个库都支持本章中描述的大多数方法。建议你熟悉至少其中一个库，以便更好地理解这些概念，并在机器学习编程面试中有信心实现最近邻搜索。

2.7 其他谈话要点

如果面试结束时还有额外时间，您可能会被问到后续问题或被要求讨论高级话题，这取决于各种因素，例如面试官的偏好、候选人的专业知识、角色要求等。下面列出了一些需要准备的主题，尤其是对于高级职位。

- 通过识别和阻止不适当的图像来调节系统中的内容。[24](#)
- 系统中存在不同的偏差，例如位置偏差。[25](#) [26](#)
- 如何使用标签等图像元数据来改进搜索结果。第 3 章 Google 街景模糊系统介绍了这一点。
- 使用对象检测进行智能裁剪。[27](#)
- 如何使用图形神经网络学习更好的表示。[28](#)
- 支持通过文本查询搜索图像的能力。我们将在第 4 章中对此进行研究。
- 如何使用主动学习。[29](#) 或人机交互 [30](#) ML 更有效地注释数据。

References

23. ScaNN library. <https://github.com/google-research/google-research/tree/master/scann>.
24. Content moderation with ML. <https://appen.com/blog/content-moderation/>.
25. Bias in AI and recommendation systems. <https://www.searchenginejournal.com/biases-search-recommender-systems/339319/close>.
26. Positional bias. <https://eugeneyan.com/writing/position-bias/>.
27. Smart crop. https://blog.twitter.com/engineering/en_us/topics/infrastructure/2018/Smart-Auto-Cropping-of-Images.
28. Better search with GNNs. <https://arxiv.org/pdf/2010.01666.pdf>.
29. Active learning. [https://en.wikipedia.org/wiki/Active_learning_\(machine_learning\)](https://en.wikipedia.org/wiki/Active_learning_(machine_learning)).
30. Human-in-the-loop ML. <https://arxiv.org/pdf/2108.00941.pdf>.

3 Google Street View Blurring System 街景模糊系统

Google 街景¹是Google地图中的一项技术，它提供了全球许多公共道路网络的街道级互动全景图像。2008年，Google创建了一个自动模糊人脸和车牌的系统，以保护用户隐私。本章中，我们设计了一个类似于Google街景的模糊系统。

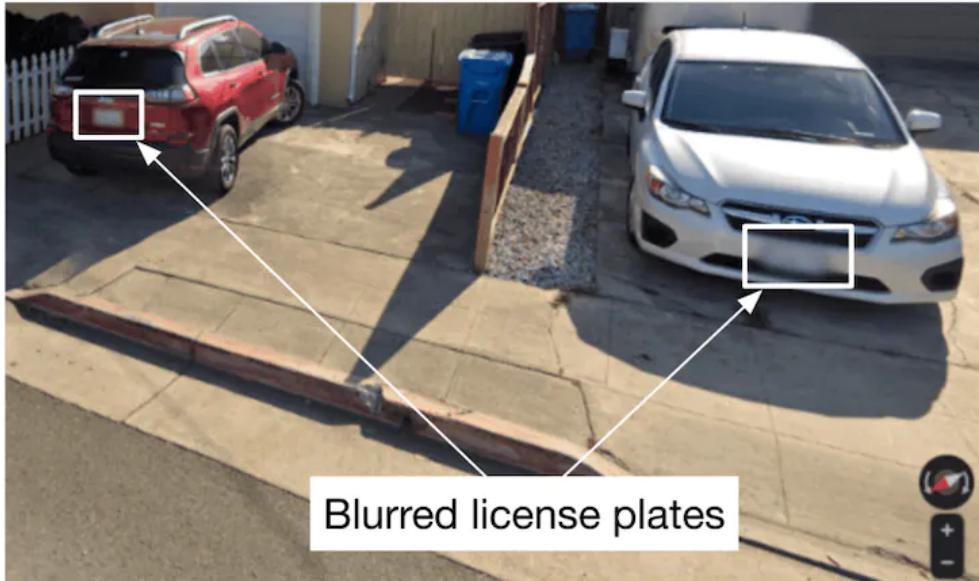


图 3.1: 带有模糊车牌的街景图像

3.1 明确需求

以下是候选人与面试官之间的典型对话。

候选人: 可以说系统的业务目标是保护用户隐私吗？

面试官: 是的。

候选人: 我们希望设计一个检测街景图像中所有人脸和车牌并在向用户展示之前对它们进行模糊处理的系统。这样理解对吗？我可以假设用户可以举报未正确模糊的图像吗？

面试官: 是的，这些假设是合理的。

候选人: 我们是否有该任务的标注数据集？

面试官: 假设我们抽取了100万张图像。人脸和车牌在这些图像中是手动标注的。

候选人: 数据集中可能不包含某些种族特征的人脸，这可能导致对某些人类属性（如种族、年龄、性别等）的偏见。这个假设合理吗？

面试官: 说得很好。为简化讨论，今天我们不关注公平性和偏见问题。

候选人: 我的理解是延迟不是一个大问题，因为系统可以离线检测对象并进行模糊处理。对吗？

面试官: 是的。我们可以在处理新图像时向用户展示现有图像。

让我们总结一下问题陈述。我们希望设计一个街景模糊系统，可以自动模糊车牌和人脸。我们得到了一个包含100万张带有人脸和车牌标注的图像的训练数据集。系统的业务目标是保护用户隐私。

3.2 将问题框定为一个机器学习任务

在本节中，我们将问题框定为一个机器学习任务。

3.2.1 定义机器学习目标

该系统的业务目标是通过模糊街景图像中可见的车牌和人脸来保护用户隐私。但保护用户隐私并不是一个机器学习目标，因此我们需要将其转化为机器学习系统可以解决的目标。一个可能的机器学习目标是准确地检测图像中的感兴趣对象。如果机器学习系统能够准确地检测到这些对象，那么我们就可以在向用户展示图像之前对这些对象进行模糊处理。

在本章中，为了简洁起见，我们使用“对象”代替“人脸和车牌”。

3.2.2 指定系统的输入和输出

对象检测模型的输入是一张包含零个或多个位于不同位置的对象的图像。模型检测这些对象并输出它们的位置。图 3.2 显示了一个对象检测系统及其输入和输出。

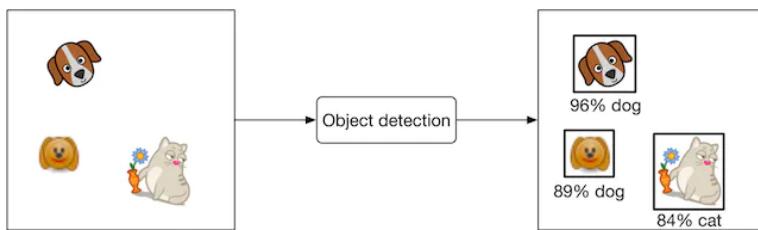


图 3.2: 对象检测系统的输入输出

3.2.3 选择正确的机器学习类别

一般来说，对象检测系统有两个职责：

- 预测图像中每个对象的位置
- 预测每个边界框的类别（例如，狗、猫等）

第一个任务是一个回归问题，因为位置可以用 (x, y) 坐标表示，这些是数值值。第二个任务可以框定为一个多分类问题。

传统上，对象检测架构分为单阶段网络和双阶段网络。最近，基于 Transformer 的架构如 DETR² 显示出了良好的效果，但在本章中，我们主要探讨双阶段和单阶段架构。

3.2.4 双阶段网络 Two-stage networks

顾名思义，双阶段网络中使用了两个独立的模型：

1. **区域建议网络 (Region proposal network, RPN):** 扫描图像并提出可能是对象的候选区域。
2. **分类器:** 处理每个建议区域并将其分类为一个对象类别。

图 3.3 显示了这两个阶段。

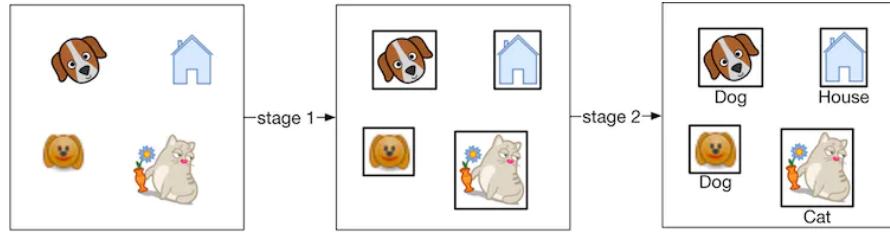


图 3.3: 双阶段网络

常用的双阶段网络包括: R-CNN 3、Fast R-CNN 4 和 Faster R-CNN 5。

3.2.5 单阶段网络 One-stage networks

在这些网络中，两个阶段被合并。使用一个单一网络，同时生成边界框和对象类别，而不需要显式的区域建议检测。图 3.4 显示了一个单阶段网络。

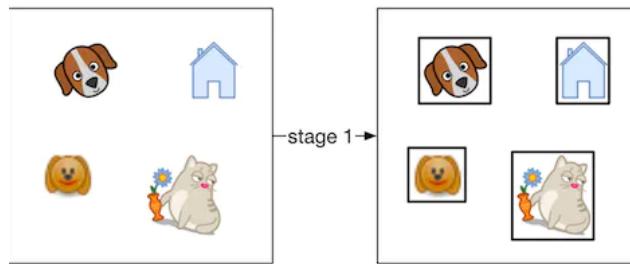


图 3.4: 单阶段网络

常用的单阶段网络包括: YOLO 6 和 SSD 7 架构。

单阶段 vs. 双阶段 双阶段网络包含两个顺序运行的组件，因此通常速度较慢，但更准确。

在我们的案例中，数据集包含 100 万张图像，这在现代标准下并不算大。这表明使用双阶段网络不会显著增加训练成本。因此，在本次练习中，我们从双阶段网络开始。当训练数据增加或需要更快地进行预测时，我们可以切换到单阶段网络。

3.3 数据准备

3.3.1 数据工程

在介绍章节中，我们讨论了数据工程的基础知识。此外，讨论特定任务可用的数据通常是个好主意。对于这个问题，我们有以下数据可用：

- 标注数据集
- 街景图像

让我们更详细地讨论每一项。

3.3.1.1 标注数据集

根据需求，我们有 100 万张标注图像。每张图像都有一个边界框列表和相关的对象类别。表 3.1 显示了数据集中的一些数据点：

Image path	Objects	Bounding boxes
dataset/image1.jpg	human face	[10, 10, 25, 50]
	human face	[120, 180, 40, 70]
	license plate	[80, 95, 35, 10]
dataset/image2.jpg	human face	[170, 190, 30, 80]
	license plate	[25, 30, 210, 220]
dataset/image3.jpg	human face	[30, 40, 30, 60]

表 3.1: 标注数据集中的一些数据点

每个边界框是由 4 个数字组成的列表：左上角 X 和 Y 坐标，以及对象的宽度和高度。

3.3.1.2 街景图像

这些是数据采集团队收集的街景图像。ML 系统处理这些图像以检测人脸和车牌。表 3.2 显示了图像的元数据。

Image path	Location (lat, lng)	Pitch, Yaw, Roll	Timestamp
tmp/image1.jpg	(37.432567, -122.143993)	(0,10,20)	1646276421
tmp/image2.jpg	(37.387843, -122.091086)	(0,10,-10)	1646276539
tmp/image3.jpg	(37.542081, -121.997640)	(10,-20,45)	1646276752

表 3.2: 街景图像的元数据

3.3.2 特征工程

在特征工程中，我们首先应用标准的预处理操作，如调整大小和归一化。之后，我们通过数据增强技术增加数据集的大小。让我们更详细地了解这一点。

数据增强 数据增强是一种通过对原始数据进行轻微修改来添加副本，或从原始数据中人工创建新数据的技术。随着数据集大小的增加，模型能够学习到更复杂的模式。当数据集不平衡时，该技术特别有用，因为它可以增加少数类的数据点数量。

一种特殊的数据增强类型是图像增强。常用的增强技术包括：

- 随机裁剪
- 随机饱和度
- 垂直或水平翻转
- 旋转和/或平移

- 仿射变换
- 改变亮度、饱和度或对比度

图 3.5 展示了应用了各种数据增强技术的图像。

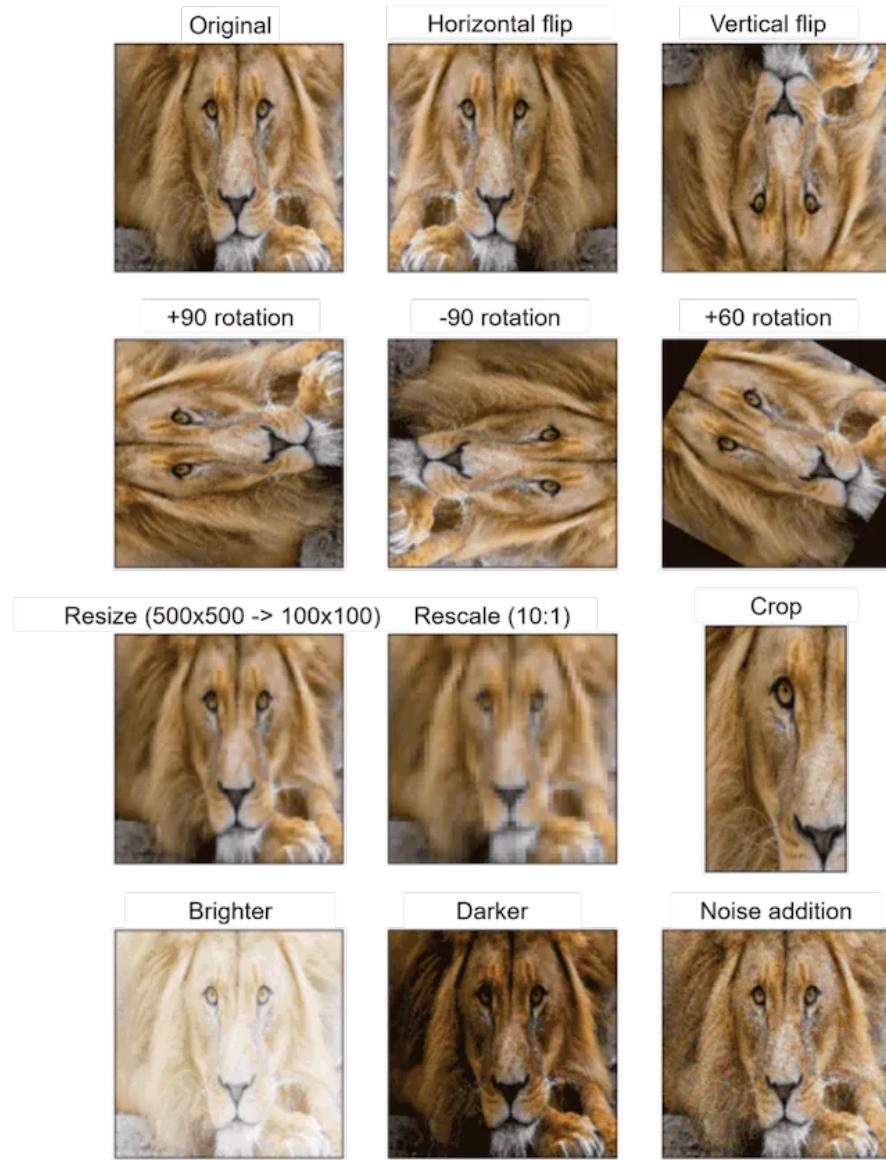


图 3.5: 增强图像 (来源 [8](#))

需要注意的是，对于某些类型的增强，真实边界框也需要进行相应的变换。例如，当旋转或翻转原始图像时，真实边界框也必须进行变换。

数据增强可以以离线或在线形式使用。

- **离线:** 在训练之前增强图像
- **在线:** 在训练过程中实时增强图像

在线 vs. 离线：在离线数据增强中，训练速度较快，因为不需要额外的增强操作。然而，它需要额外的存储空间来存储所有增强后的图像。虽然在线数据增强会减慢训练速度，但不会消耗额外的存储空间。

在在线和离线数据增强之间的选择取决于存储和计算能力的限制。更重要的是，在面试中你需要谈论不同的选择并讨论权衡。在我们的案例中，我们进行离线数据增强。

图 3.6 显示了数据集准备流程。通过预处理，图像被调整大小、缩放和归一化。通过图像增强，图像数量会增加。假设数量从 100 万增加到 1000 万。

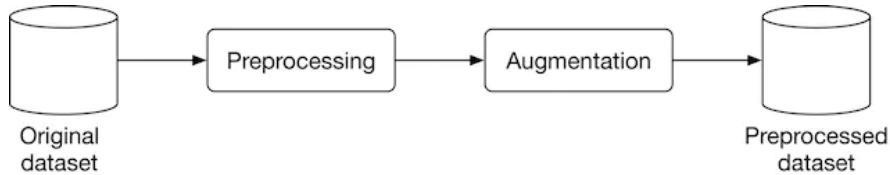


图 3.6: 数据集准备工作流

3.4 模型开发

3.4.1 模型选择

如“将问题框定为一个 ML 任务”部分所述，我们选择了双阶段网络。图 3.7 显示了一个典型的双阶段架构。

让我们检查每个组件。

3.4.1.1 卷积层

卷积层 9 处理输入图像并输出特征图。

3.4.1.2 区域建议网络 (RPN)

RPN 提出可能包含对象的候选区域。它使用神经网络作为其架构，并将卷积层生成的特征图作为输入，输出图像中的候选区域。

3.4.1.3 分类器

分类器确定每个候选区域的对象类别。它将特征图和建议的候选区域作为输入，并为每个区域分配一个对象类别。该分类器通常基于神经网络。

在 ML 系统设计面试中，通常不需要详细讨论这些神经网络的架构。更多信息请参见 10。

3.4.2 模型训练

训练神经网络的过程通常包括三个步骤：前向传播、损失计算和反向传播。读者应该对这些步骤有所了解，但更多信息请参见 11。在本节中，我们讨论了常用于检测对象的损失函数。

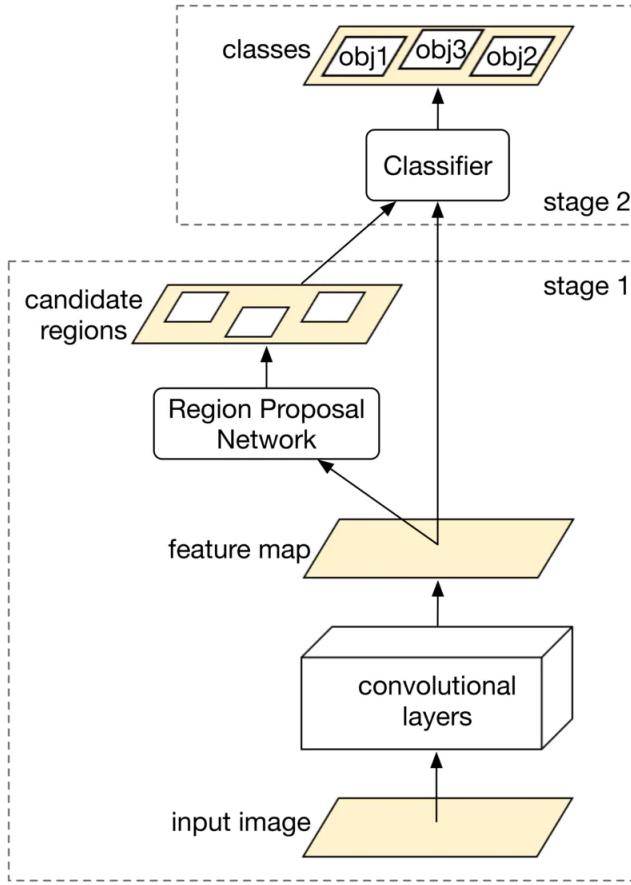


图 3.7: 双阶段对象检测网络

对象检测模型需要很好地执行两项任务。首先，预测的对象边界框应该与真实边界框高度重合。这是一个回归任务。其次，每个对象类别的预测概率应该准确。这是一个分类任务。让我们为每个任务定义一个损失函数。

回归损失：该损失度量预测的边界框与真实边界框的对齐程度。我们使用标准回归损失函数，如均方误差（MSE）[12](#)，并将其表示为 L_{reg} ：

$$L_{reg} = \frac{1}{M} \sum_{i=1}^M \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2 \right]$$

分类损失：这衡量每个检测到的对象的预测概率的准确性。这里，我们使用标准分类损失，如对数损失（交叉熵）[13](#)，并将其表示为 L_{cls} ：

$$L_{cls} = -\frac{1}{M} \sum_{i=1}^M \sum_{c=1}^C y_c \log \hat{y}_c$$

为了定义一个衡量模型整体性能的最终损失，我们将分类损失和回归损失按一个平衡参数 λ 组合：

$$L = L_{cls} + \lambda L_{reg}$$

3.5 评估

在面试过程中，讨论如何评估机器学习系统是至关重要的。面试官通常想知道你会选择哪些指标以及原因。本节描述了对象检测系统通常如何进行评估，并为离线和在线评估选择了重要的指标。

一个对象检测模型通常需要检测图像中的 N 个不同的对象。为了衡量模型的整体性能，我们分别评估每个对象的检测结果，然后对结果进行平均。

图3.8显示了一个对象检测模型的输出。图中展示了检测到的边界框和实际的边界框。模型检测到了 6 个边界框，而我们实际上只有两个目标实例。

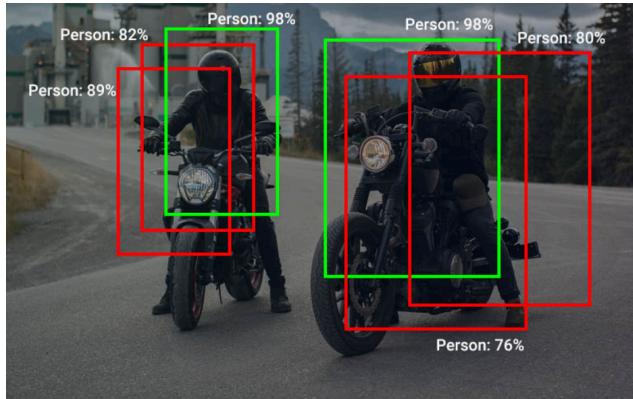


图 3.8: 实际的边界框和检测到的边界框

什么时候预测的边界框被认为是正确的？要回答这个问题，我们需要了解“交并比”（Intersection Over Union, IOU）的定义。

交并比 (Intersection Over Union, IOU): IOU 衡量两个边界框的重叠程度。图 3.9 显示了 IOU 的可视化表示。

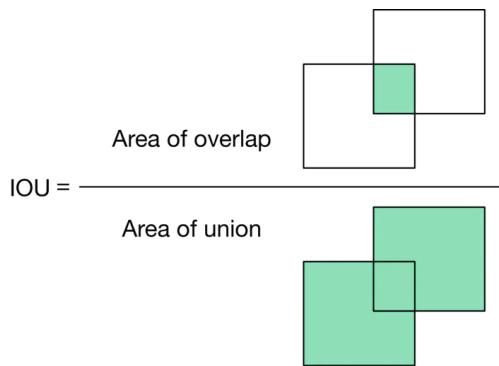


图 3.9: IOU 公式

IOU 用来确定检测到的边界框是否正确。IOU 为 1 时是理想的，表示检测到的边界框和实际边界框完全重合。在实践中，IOU 为 1 的情况很少见。IOU 越高，表示预测的边界框越准确。通常会使用 IOU 阈值来决定检测到的边界框是正确（真阳性）还是错误（假阳性）。例如，IOU 阈值为 0.7 时，任何与实际边界框有 0.7 或更高重叠度的检测结果都被认为是正确的检测。

现在我们了解了 IOU 以及如何判断边界框预测的正确性，接下来讨论离线评估的指标。

3.5.1 离线指标

模型开发是一个迭代过程。我们使用离线指标来快速评估新开发的模型性能。以下是一些可能对对象检测系统有用的指标：

- 精确率
- 平均精度 (AP)
- 平均平均精度 (mAP)

精确率 这是所有图像中所有检测中正确检测的比例。较高的精确率表明系统的检测结果更加可靠。

$$\text{Precision} = \frac{\text{Correct detections}}{\text{Total detections}}$$

为了计算精确率，我们需要选择一个 IOU 阈值。让我们通过一个例子来更好地理解这一点。图 3.10 显示了一组实际的边界框和检测到的边界框以及它们各自的 IOU 值。

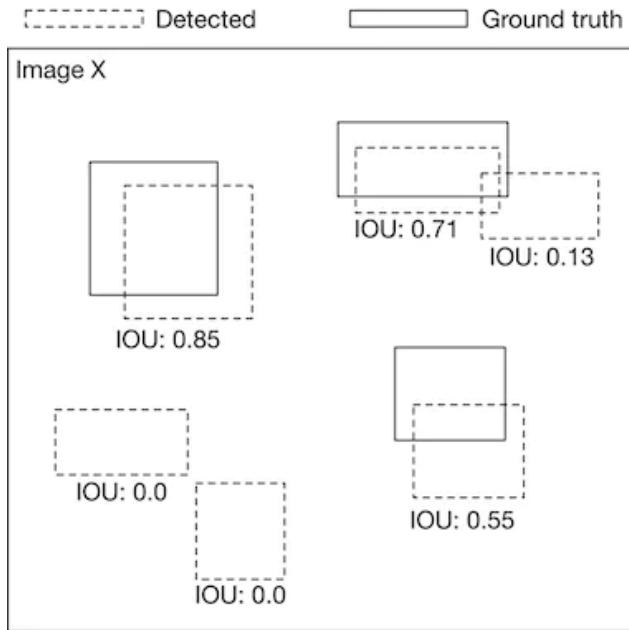


图 3.10: 实际的边界框和检测到的边界框

让我们计算三个不同 IOU 阈值下的精确率：0.7、0.5 和 0.1。

- **IOU 阈值 = 0.7:** 在 6 个检测结果中，有 2 个的 IOU 高于 0.7。因此，我们在此阈值下有 2 个正确预测。

$$\text{Precision}_{0.7} = \frac{\text{Correct detections}}{\text{Total detections}} = \frac{2}{6} = 0.33$$

- **IOU 阈值 = 0.5:** 在此阈值下，有 3 个检测结果的 IOU 高于 0.5：

$$\text{Precision}_{0.5} = \frac{\text{Correct detections}}{\text{Total detections}} = \frac{3}{6} = 0.5$$

- **IOU 阈值 = 0.1:** 这次我们有 4 个正确的检测：

$$\text{Precision}_{0.1} = \frac{\text{Correct detections}}{\text{Total detections}} = \frac{4}{6} = 0.67$$

你可能已经注意到，精确率的主要缺点是它随着不同的 IOU 阈值而变化。因此，仅通过查看特定 IOU 阈值下的精确率很难了解模型的整体性能。平均精度（Average Precision, AP）解决了这一限制。

平均精度 (Average Precision, AP): 该指标在各种 IOU 阈值下计算精确率并取其平均值。AP 的公式为：

$$AP = \int_0^1 P(r) dr$$

其中， $P(r)$ 是 IOU 阈值为 r 时的精确率。

上面的公式可以通过对预定义的阈值列表进行离散求和来近似。例如，在 Pascal VOC2008 基准中，AP 是在 11 个均匀分布的阈值上计算的。

$$AP = \frac{1}{11} \sum_{n=0}^{n=10} P(n)$$

AP 汇总了模型在特定对象类别（例如人脸）上的整体精确率。为了衡量模型在所有对象类别（例如人脸和车牌）上的整体精确率，我们需要使用平均平均精度（Mean average precision, mAP）。

平均平均精度 (Mean average precision, mAP): 这是 AP 在所有对象类别上的平均值。该指标总结了模型的整体性能。公式如下：

$$mAP = \frac{1}{C} \sum_{c=1}^C AP_c$$

其中， C 是模型检测的总对象类别数。

mAP 指标常用于评估对象检测系统。要了解标准基准中使用的阈值，请参考相关文献。

3.5.2 在线指标

根据需求，系统需要保护个人隐私。评估这一点的一个方法是统计用户举报和投诉的数量。我们还可以依赖人工标注员来抽查错误模糊的图像百分比。其他衡量偏差和公平性的指标也很重要。例如，我们希望在人脸模糊中对不同种族和年龄组同样准确地进行处理。但如需求中所述，衡量偏差超出了本文的讨论范围。

总结评估部分，我们使用 mAP 和 AP 作为离线指标。mAP 衡量模型的整体精确率，而 AP 可以帮助我们了解模型在特定类别中的精确率。在线评估的主要指标是“用户举报”数量。

3.6 服务

本节首先讨论对象检测系统中可能出现的一个常见问题：重叠的边界框。然后，我们提出一个完整的 ML 系统设计。

3.6.1 重叠的边界框

在图像上运行对象检测算法时，边界框重叠是非常常见的情况。这是因为 RPN 网络会在每个对象周围提出多个高度重叠的边界框。在推理过程中，将这些边界框精简为每个对象的单个边界框非常重要。

一种广泛使用的解决方案是称为“非极大值抑制 (Non-maximum suppression, NMS)” 的算法 17。让我们来看看它是如何工作的。

NMS NMS 是一种后处理算法，旨在选择最合适的选择器。它保留置信度较高的边界框，并移除重叠的边界框。图 3.11 展示了一个示例。

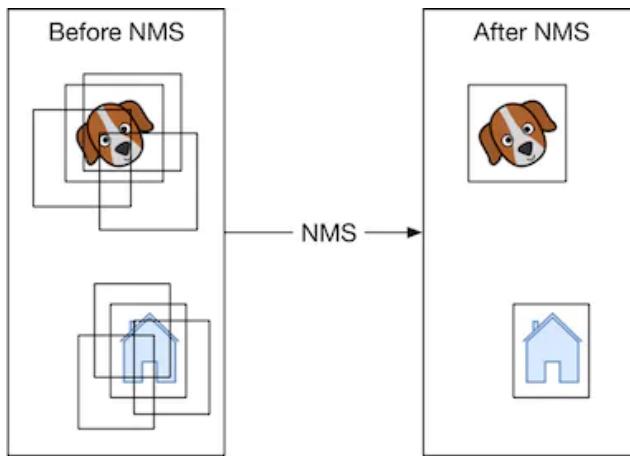


图 3.11: 应用 NMS 前后

NMS 是 ML 系统设计面试中常被问到的算法，因此建议深入理解它 18。

3.6.2 ML 系统设计

如图 3.12 所示，我们为模糊化系统提出了一个 ML 系统设计。

让我们更详细地检查每个 Pipeline。

批量预测 Pipeline 根据收集的需求，延迟不是一个大问题，因为我们在处理新图像时向用户展示现有的图像。由于不需要即时结果，我们可以利用批量预测预先计算对象检测结果。

预处理。 原始图像通过该组件进行预处理。本节不讨论预处理操作，因为我们已经在特征工程部分讨论过。

模糊化服务。 该服务对 Street View 图像执行以下操作：

1. 提供图像中检测到的对象列表。
2. 使用 NMS 组件优化检测到的对象列表。

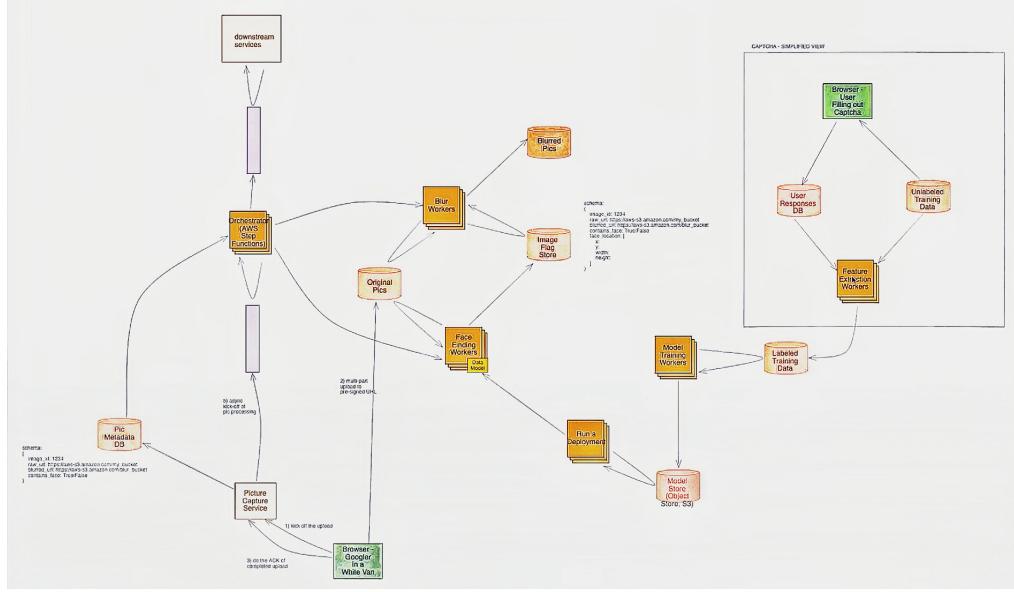


图 3.12: ML 系统设计

3. 对检测到的对象进行模糊处理。
4. 将模糊处理后的图像存储在对象存储中（模糊处理的 Street View 图像）。

请注意，预处理和模糊化服务在设计中是分开的。原因是预处理图像往往是一个 CPU 密集型过程，而模糊化服务则依赖于 GPU。将这些服务分开有两个好处：

- 根据每个服务接收到的工作负载独立扩展服务。
- 更好地利用 CPU 和 GPU 资源。

数据 Pipeline 该 Pipeline 负责处理用户报告，生成新的训练数据，并准备供模型使用的训练数据。数据 Pipeline 的组件大多是自解释的。困难负样本挖掘是唯一需要更多解释的组件。

困难负样本挖掘。 困难负样本是从错误预测的示例中专门创建的负样本，并添加到训练数据集中。当我们在更新的训练数据集上重新训练模型时，模型应该表现得更好。

3.7 其他讨论点

如果时间允许，这里有一些其他讨论点：

- Transformer-based 对象检测架构与单阶段或双阶段模型有何不同，以及它们的优缺点 19。
- 提高在大规模数据集上对对象检测的分布式训练技术 20 21。
- 欧洲的《通用数据保护条例 (GDPR)》可能如何影响我们的系统 22。
- 评估面部检测系统中的偏差 23 24。

- 如何持续微调模型 [25](#)。
- 如何使用主动学习 [26](#) 或人机协作学习 [27](#) 来选择训练的数据点。

References

1. Google Street View. <https://www.google.com/streetview>.
2. DETR. <https://github.com/facebookresearch/detr>.
3. RCNN family. <https://lilianweng.github.io/posts/2017-12-31-object-recognition-part-3>.
4. Fast R-CNN paper. <https://arxiv.org/pdf/1504.08083.pdf>.
5. Faster R-CNN paper. <https://arxiv.org/pdf/1506.01497.pdf>.
6. YOLO family. <https://pyimagesearch.com/2022/04/04/introduction-to-the-yolo-family>.
7. SSD. <https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox>.
8. Data augmentation techniques. <https://www.kaggle.com/getting-started/190280>.
9. CNN. https://en.wikipedia.org/wiki/Convolutional_neural_network.
10. Object detection details. <https://dudeperf3ct.github.io/object/detection/2019/01/07/Mystery-of-Object-Detection>.
11. Forward pass and backward pass. <https://www.youtube.com/watch?v=qzPQ8cEsVK8>.
12. MSE. https://en.wikipedia.org/wiki/Mean_squared_error.
13. Log loss. https://en.wikipedia.org/wiki/Cross_entropy.
14. Pascal VOC. <http://host.robots.ox.ac.uk/pascal/VOC/voc2008/index.html>.
15. COCO dataset evaluation. <https://cocodataset.org/#detection-eval>.
16. Object detection evaluation. <https://github.com/rafaelpadilla/Object-Detection-Metrics>.
17. NMS. <https://en.wikipedia.org/wiki/NMS>.
18. Pytorch implementation of NMS. <https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/>.
19. Recent object detection models. <https://viso.ai/deep-learning/object-detection/>.
20. Distributed training in Tensorflow. https://www.tensorflow.org/guide/distributed_training.
21. Distributed training in Pytorch. https://pytorch.org/tutorials/beginner/dist_overview.html.
22. GDPR and ML. <https://www.oreilly.com/radar/how-will-the-gdpr-impact-machine-learning>.
23. Bias and fairness in face detection. <http://sibgrapi.sid.inpe.br/col/sid.inpe.br/sibgrapi/2021/09.04.19.00/doc/103.pdf>.

24. AI fairness. <https://www.kaggle.com/code/alexisbcook/ai-fairness>.
25. Continual learning. <https://towardsdatascience.com/how-to-apply-continual-learning-to-your-machine-learning-model-98a2a2a2a2a2>.
26. Active learning. [https://en.wikipedia.org/wiki/Active_learning_\(machine_learning\)](https://en.wikipedia.org/wiki/Active_learning_(machine_learning)).
27. Human-in-the-loop ML. <https://arxiv.org/pdf/2108.00941.pdf>.

4 YouTube Video Search 视频搜索

4.1 Introduction

在 YouTube 等视频共享平台上，视频数量可以迅速增长到数十亿级别。在本章中，我们设计一个视频搜索系统，能够高效地处理如此大量的内容。如图 4.1 所示，用户在搜索框中输入文本，系统展示与该文本最相关的视频。

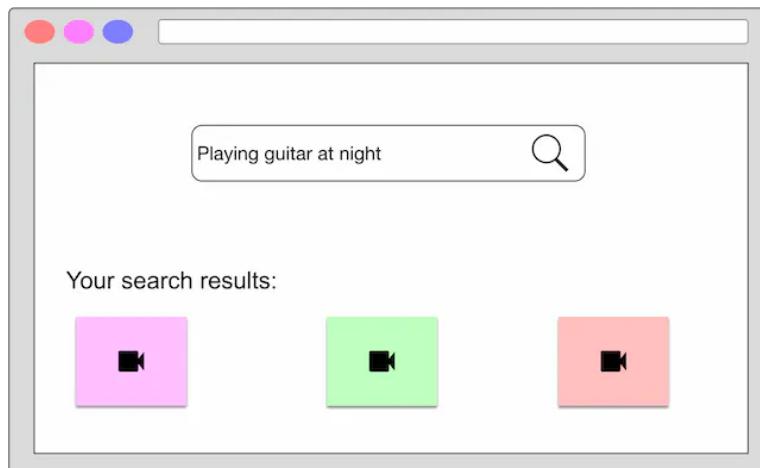


图 4.1: 使用文本查询搜索视频

4.2 明确需求

以下是候选人与面试官之间的典型对话。

候选人: 输入查询是仅限文本，还是用户可以用图像或视频进行搜索？

面试官: 仅限文本查询。

候选人: 平台上的内容是否仅限于视频形式？图片或音频文件呢？

面试官: 平台仅提供视频。

候选人: YouTube 的搜索系统非常复杂。可以假设视频的相关性仅由其视觉内容和与视频相关的文本数据（如标题和描述）来决定吗？

面试官: 是的，这个假设合理。

候选人: 是否有可用的训练数据？

面试官: 是的，假设我们有一千万对（视频，文本查询）。

候选人: 搜索系统需要支持其他语言吗？

面试官: 为简化问题，假设仅支持英语。

候选人: 平台上有多少视频？

面试官：十亿个视频。

候选人：我们需要个性化结果吗？是否需要根据用户的过往互动来为不同用户排名结果？

面试官：与个性化在推荐系统中至关重要不同，在搜索系统中并不一定需要个性化。为简化问题，假设不需要个性化。

让我们总结一下问题陈述。我们需要设计一个视频搜索系统。输入是文本查询，输出是与文本查询相关的视频列表。为了搜索相关视频，我们利用视频的视觉内容和文本数据。我们拥有一个包含一千万对〈视频，文本查询〉的数据集用于模型训练。

4.3 将问题框定为 ML 任务

4.3.1 定义 ML 目标

用户期望搜索系统提供相关且有用的结果。将其转化为 ML 目标的一种方法是根据视频与文本查询的相关性对视频进行排名。

4.3.2 指定系统的输入和输出

如图 4.2 所示，搜索系统以文本查询为输入，并输出按与文本查询相关性排序的视频列表。

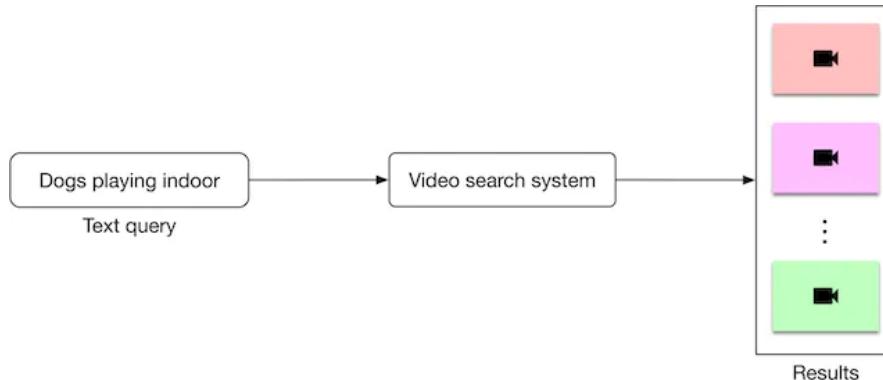


图 4.2: 视频搜索系统的输入-输出

4.3.3 选择合适的 ML 分类

为了确定视频与文本查询之间的相关性，我们利用视频的视觉内容和文本数据。如图 4.3 所示，设计的概述展示了这一点。

让我们简要讨论每个组件。

视觉搜索 该组件接受文本查询作为输入，并输出视频列表。这些视频根据文本查询与其视觉内容之间的相似性进行排名。

表征学习是一种常用于通过处理视频视觉内容进行搜索的方法。在该方法中，文本查询和视频分别使用两个编码器进行编码。如图 4.4 所示，ML 模型包含一个视频编码器，用于从视频生成嵌入向量，以及一个文本编码器，用于从文本生成嵌入向量。通过计算视频与文本表示的点积来计算相似度得分。

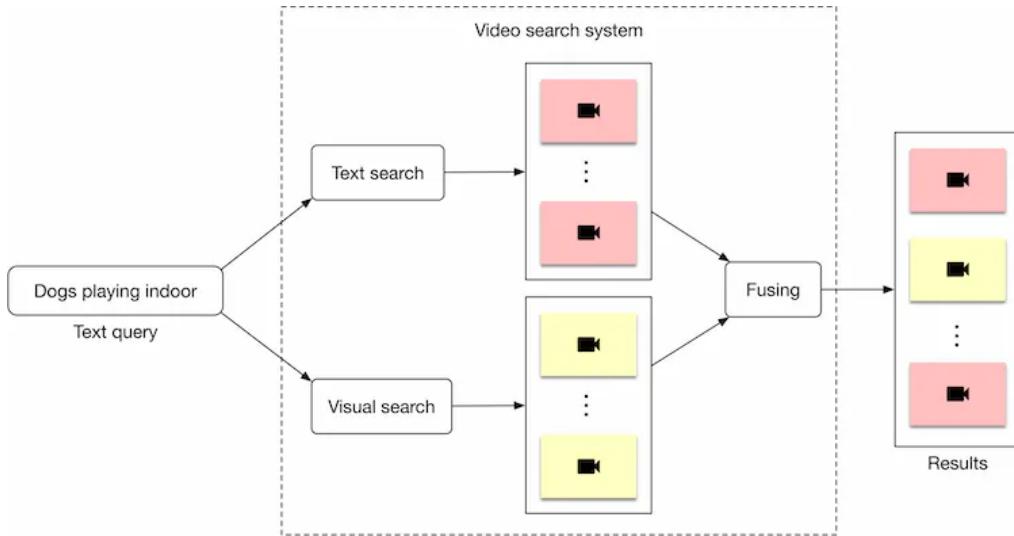


图 4.3: 搜索系统的高层次概述

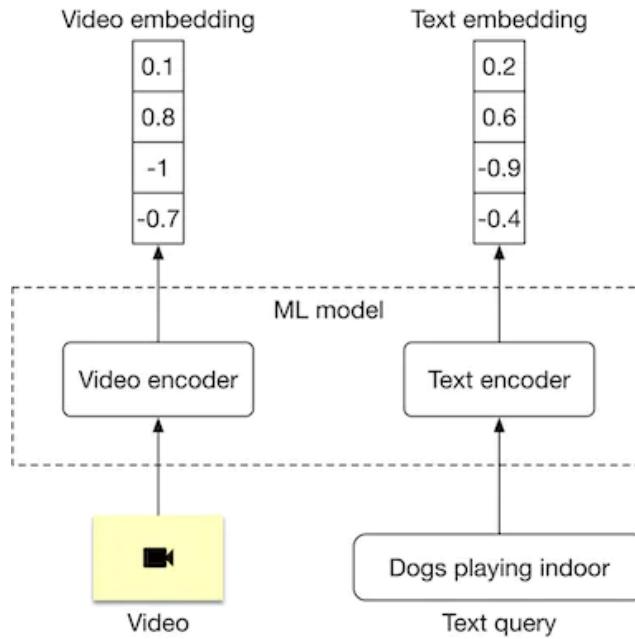


图 4.4: ML 模型的输入-输出

为了对与文本查询在视觉和语义上相似的视频进行排序，我们在嵌入空间中计算文本与每个视频之间的点积，然后根据相似度得分对视频进行排名。

文本搜索 图 4.5 展示了当用户输入文本查询 “dogs playing indoor” 时，文本搜索的工作原理。标题、描述或标签与文本查询最相似的视频会显示为输出。

倒排索引是一种用于创建基于文本的搜索组件的常用技术，允许在数据库中进行高效的全文搜索。由于倒排索引不是基于机器学习的，因此没有训练成本。许多搜索引擎公司经常使用的一个流行搜索引擎



图 4.5: 文本搜索

是 Elasticsearch，这是一种可扩展的搜索引擎和文档存储库。有关 Elasticsearch 的更多详细信息和深入理解，请参阅 [1](#)。

4.4 数据准备

4.4.1 数据工程

由于我们已经获得了一个带注释的数据集来训练和评估模型，因此不需要进行数据工程处理。表 4.1 显示了带注释的数据集的示例。

视频名称	查询	数据集划分类型
76134.mp4	孩子们在游泳池游泳！	训练集
92167.mp4	庆祝毕业	训练集
2867.mp4	一群青少年在踢足球	验证集
28543.mp4	Tensorboard 是如何工作的	验证集
70310.mp4	冬季公路旅行	测试集

表 4.1: 带注释的数据集

4.4.2 特征工程

几乎所有的机器学习算法只接受数值输入。因此，诸如文本和视频等非结构化数据需要在此步骤中转换为数值表示。让我们来看看如何为模型准备文本和视频数据。

文本数据准备 如图 4.6 所示，文本通常通过三个步骤转换为数值向量：文本归一化、标记化和将标记转换为 ID 2。

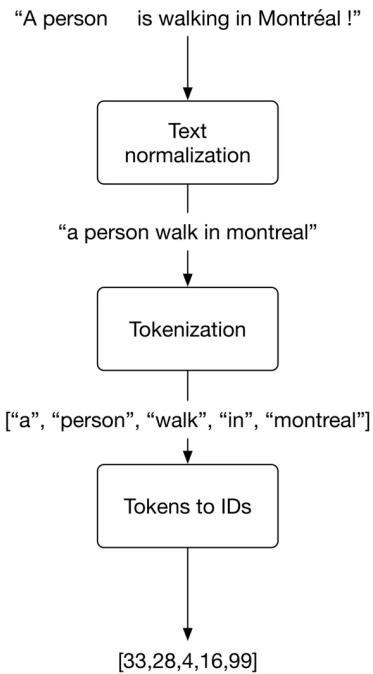


图 4.6: 用数值向量表示文本

让我们更详细地看一下每个步骤。

文本归一化 文本归一化——也称为文本清理——确保单词和句子的一致性。例如，同一个单词可能有略微不同的拼写形式，如”dog”、“dogs”和”DOG!”都指的是同一事物，但拼写不同。句子也是如此。以下是两个示例句子：

- “A person walking with his dog in Montréal !”
- “a person walks with his dog, in Montreal.”

这两句话的含义相同，但标点和动词形式不同。以下是一些常见的文本归一化方法：

- 转换为小写：将所有字母转换为小写，因为这不会改变单词或句子的含义
- 移除标点符号：移除文本中的标点符号。常见的标点符号包括句号、逗号、问号、感叹号等
- 去除空格：去除前后多余空格
- 正规化形式 KD (NFKD) 3：将组合字符分解为更简单的字符组合

- 去除重音符号：从单词中移除重音符号。例如：Màlaga → Malaga, Noël → Noel
- 词形还原与词干提取：识别一组相关词形的标准代表。例如：walking, walks, walked → walk

标记化 标记化是将文本拆分为更小的单元（称为标记）的过程。通常有三种类型的标记化：

- 单词标记化：根据特定的分隔符将文本拆分为单个单词。例如，短语”I have an interview tomorrow”会被拆分为

”I”, ”have”, ”an”, ”interview”, ”tomorrow”
- 子词标记化：将文本拆分为子词（或 n-gram 字符）
- 字符标记化：将文本拆分为字符集不同的标记化算法通常不是机器学习系统设计面试的重点。如果您有兴趣了解更多，请参阅 [4](#)。

将标记转换为 ID 一旦我们获得了标记，就需要将它们转换为数值（ID）。可以通过两种方式表示标记的数值：

查找表。在这种方法中，每个唯一的标记都映射到一个 ID。接着，创建一个查找表来存储这些 1:1 映射。图 4.7 显示了映射表的示例。

Word	ID
⋮	
animals	18
⋮	
art	35
⋮	
car	128
⋮	
insurance	426
⋮	
travel	1239
⋮	

图 4.7: 查找表

哈希。也称为“特征哈希”或“哈希技巧”，是一种节省内存的方法，它使用哈希函数获取 ID，而无需保留查找表。图 4.8 显示了如何使用哈希函数将单词转换为 ID。

让我们比较一下查找表和哈希方法。

视频数据准备 图 4.9 展示了处理原始视频的典型工作流程。

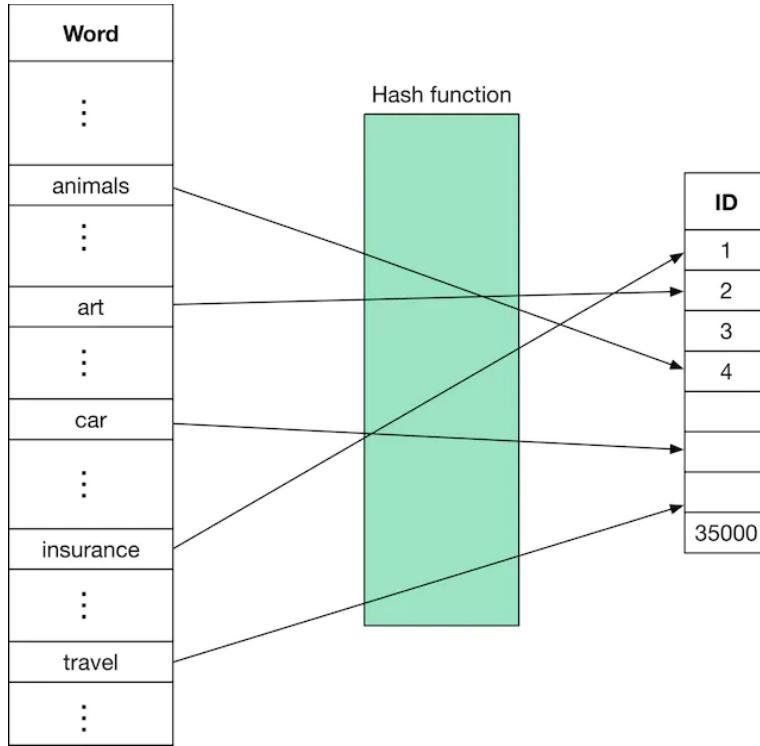


图 4.8: 使用哈希获取单词 ID

	查找表	哈希
速度	✓ 将标记转换为 ID 的速度快	✗ 需要计算哈希函数以将标记转换为 ID
ID 转标记	✓ 通过反向索引表轻松将 ID 转换为标记	✗ 无法将 ID 转换回标记
内存	✗ 表存储在内存中，大量标记会导致内存需求增加	✓ 哈希函数足以将任意标记转换为其 ID
未见标记	✗ 无法很好地处理新或未见过的单词	✓ 通过对任何单词应用哈希函数，轻松处理新或未见过的单词
碰撞 5	✓ 无碰撞问题	✗ 碰撞可能是一个潜在的问题

表 4.2: 查找表与特征哈希对比

4.5 模型开发

4.5.1 模型选择

正如在“将问题框定为机器学习任务”部分中讨论的那样，文本查询通过文本编码器转换为嵌入，视频则通过视频编码器转换为嵌入。在本节中，我们将研究每个编码器可能的模型架构。

图 4.10 显示了一个典型的文本编码器的输入和输出。

文本编码器将文本转换为向量表示 6。例如，如果两个句子的含义相似，它们的嵌入也会更相似。要构建文本编码器，有两大类方法可供选择：统计方法和基于机器学习的方法。让我们分别讨论。

4.5.2 统计方法

这些方法依赖统计数据将句子转换为特征向量。两种流行的统计方法是：

- 词袋模型（Bag of Words, BoW）

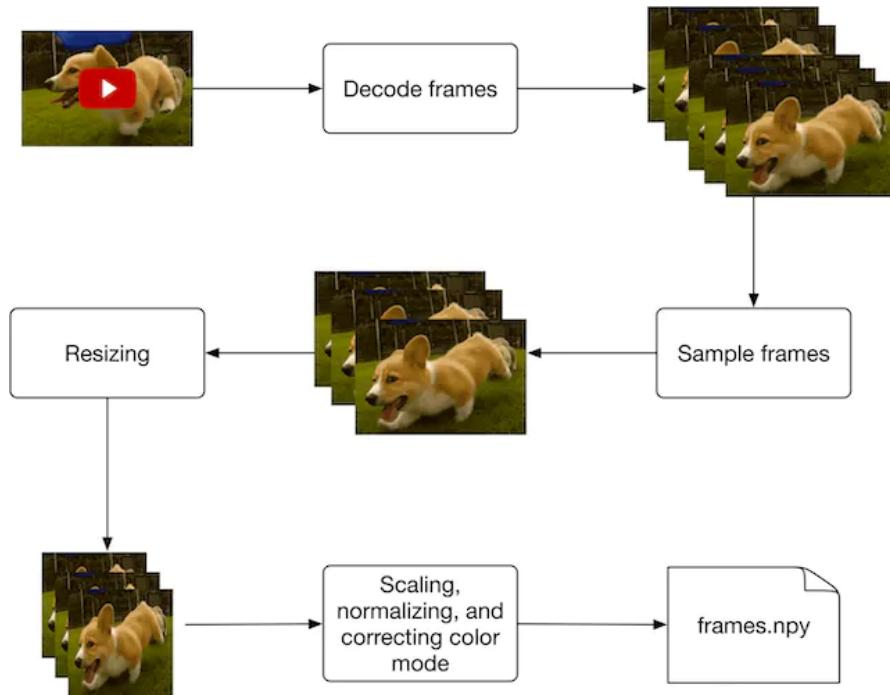


图 4.9: 视频预处理工作流程

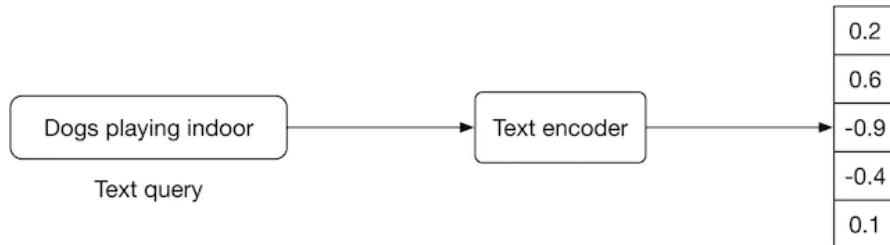


图 4.10: 文本编码器的输入-输出

- 词频-逆文档频率 (Term Frequency-Inverse Document Frequency, TF-IDF)

词袋模型 (Bag of Words, BoW)。该方法将一个句子转换为固定长度的向量。它通过创建一个矩阵来表示句子中的单词出现情况，其中行表示句子，列表示单词索引。表 4.3 显示了 BoW 的示例。

	best	holiday	is	nice	person	this	today	trip	very	with
this person is nice very nice	0	0	1	2	1	1	0	0	1	0
today is holiday	0	1	1	0	0	0	1	0	0	0
this trip with best person is best	2	0	1	0	1	1	0	1	0	1

表 4.3: 不同句子的 BoW 表示

BoW 是一种计算句子表示速度较快的方法，但它有以下局限性：

- 它不考虑句子中单词的顺序。例如，“let's watch TV after work” 和 “let's work after watch TV” 会有相同的 BoW 表示。

- 获得的表示无法捕捉句子的语义和上下文含义。例如，具有相同含义但使用不同单词的两个句子，其表示完全不同。
- 表示向量是稀疏的。表示向量的大小等于我们拥有的唯一标记总数。这个数字通常非常大，因此每个句子表示主要由零填充。

词频-逆文档频率 (Term Frequency-Inverse Document Frequency, TF-IDF)。这是一个反映单词在文档集合或语料库中的重要性的数值统计。TF-IDF 创建了与 BoW 相同的句子-单词矩阵，但它根据单词的频率对矩阵进行归一化。要了解有关其背后数学的更多信息，请参阅 [7](#)。

由于 TF-IDF 对频繁出现的单词赋予较低权重，因此它的表示通常比 BoW 更好。然而，它也有以下局限性：

- 当添加新句子时，需要重新计算词频进行归一化。
- 它不考虑句子中单词的顺序。
- 获得的表示无法捕捉句子的语义含义。
- 表示是稀疏的。

总的来说，统计方法通常较快。然而，它们无法捕捉句子的上下文含义，并且表示是稀疏的。基于机器学习的方法解决了这些问题。

4.5.3 基于机器学习的方法

在这些方法中，机器学习模型将句子转换为有意义的单词嵌入，以使两个嵌入之间的距离反映相应单词的语义相似性。例如，如果两个单词（如“rich”和“wealth”）在语义上相似，它们的嵌入在嵌入空间中就会很接近。图 4.12 显示了 2D 嵌入空间中单词嵌入的简单可视化。正如您所看到的，相似的单词被分组在一起。

有三种常见的基于机器学习的方法可以将文本转换为嵌入：

- 嵌入（查找）层
- Word2vec
- 基于 Transformer 的架构

嵌入（查找）层。在这种方法中，使用嵌入层将每个 ID 映射到一个嵌入向量。图 4.13 显示了一个示例。

使用嵌入层是将稀疏特征（如 ID）转换为固定大小嵌入的一种简单有效的解决方案。在后续章节中，我们将看到更多它的使用示例。

Word2vec。 Word2vec [8](#) 是一系列用于生成单词嵌入的相关模型。这些模型使用浅层神经网络架构，并利用单词在局部上下文中的共现来学习单词嵌入。特别是，该模型在训练阶段学习从上下文词中预测中心词。在训练阶段之后，该模型能够将单词转换为有意义的嵌入。

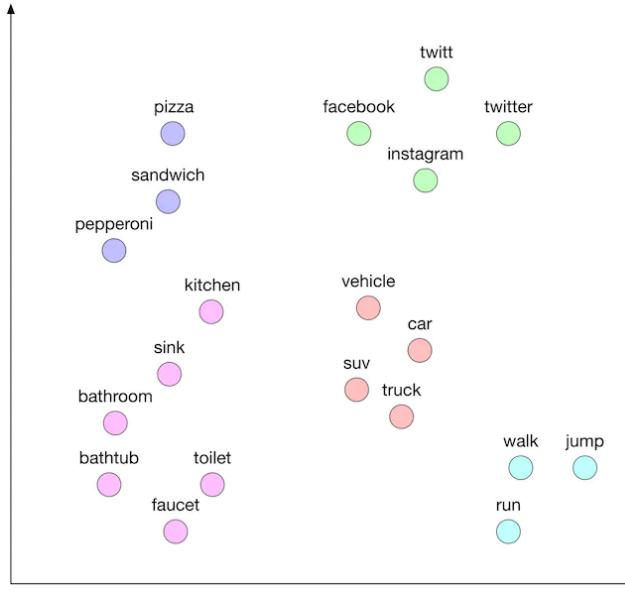


图 4.11: 2D 嵌入空间中的单词

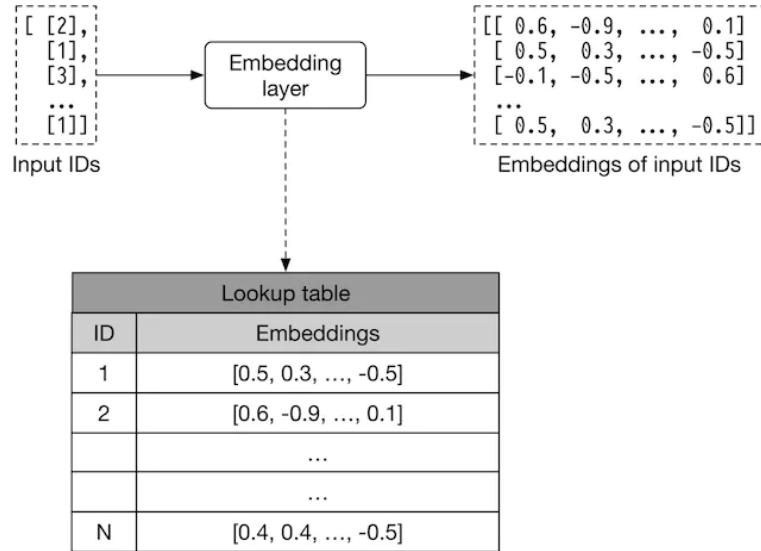


图 4.12: 嵌入查找方法

基于 word2vec 的主要模型有两个：连续词袋模型（CBOW）[9](#) 和 Skip-gram [10](#)。图 4.14 显示了 CBOW 的高级工作原理。如果您有兴趣了解这些模型，请参阅 [8](#)。

尽管 word2vec 和嵌入层方法简单有效，但最近基于 Transformer 的架构表现出了很好的效果。

基于 Transformer 的模型。这些模型在将单词转换为嵌入时，会考虑句子中的单词上下文。与 word2vec 模型不同，它们根据上下文为相同的单词生成不同的嵌入。

Transformers 在理解上下文和生成有意义的嵌入方面非常强大。BERT [11](#)、GPT3 [12](#) 和 BLOOM [13](#) 等模型展示了 Transformers 在执行各种自然语言处理任务中的潜力。在我们的案例中，我们选择 BERT 作为文本编码器。

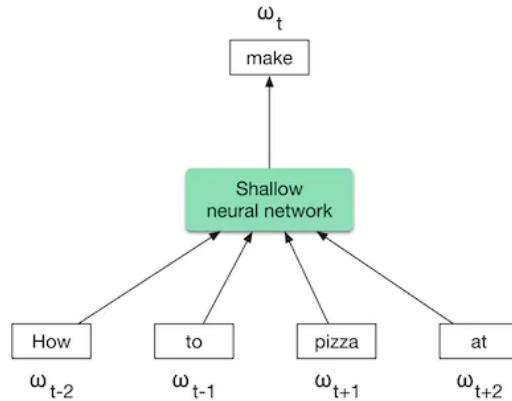


图 4.13: CBOW 方法

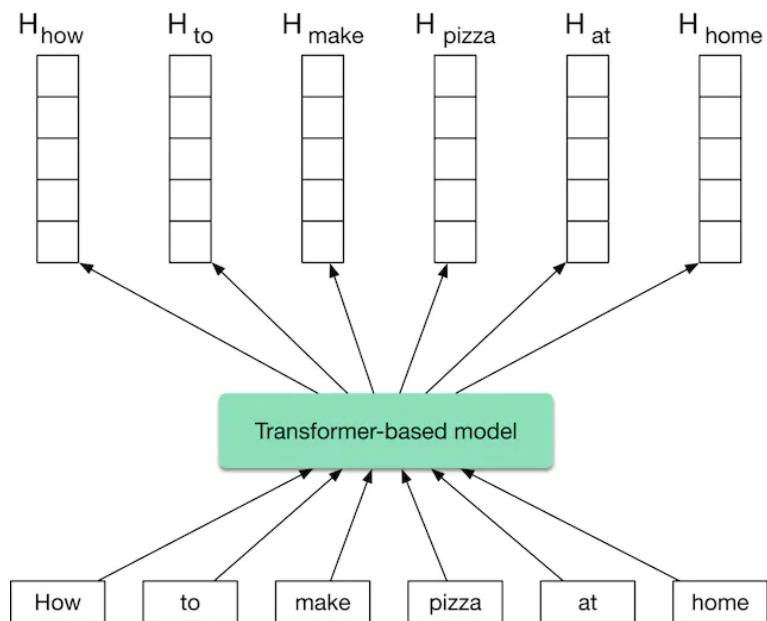


图 4.14: 基于 Transformer 的模型的输入-输出

在一些面试中，面试官可能希望您深入探讨基于 Transformer 模型的细节。如需了解更多，请参阅 [14](#)。

4.5.4 视频编码器

我们有两种编码视频的架构选择：

- 视频级模型
- 帧级模型

视频级模型处理整个视频以创建嵌入，如图 4.16 所示。该模型架构通常基于 3D 卷积 [15](#) 或 Transformers。由于模型处理的是整个视频，因此计算量大。

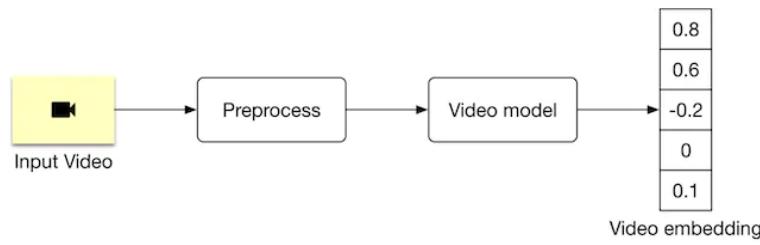


图 4.15: 视频级模型

帧级模型的工作方式不同。可以通过三步将视频分解为帧级模型并提取嵌入：

- 对视频进行预处理并对其进行帧采样。
- 使用模型对采样帧运行，以生成帧嵌入。
- 聚合（例如，求平均值）帧嵌入以生成视频嵌入。

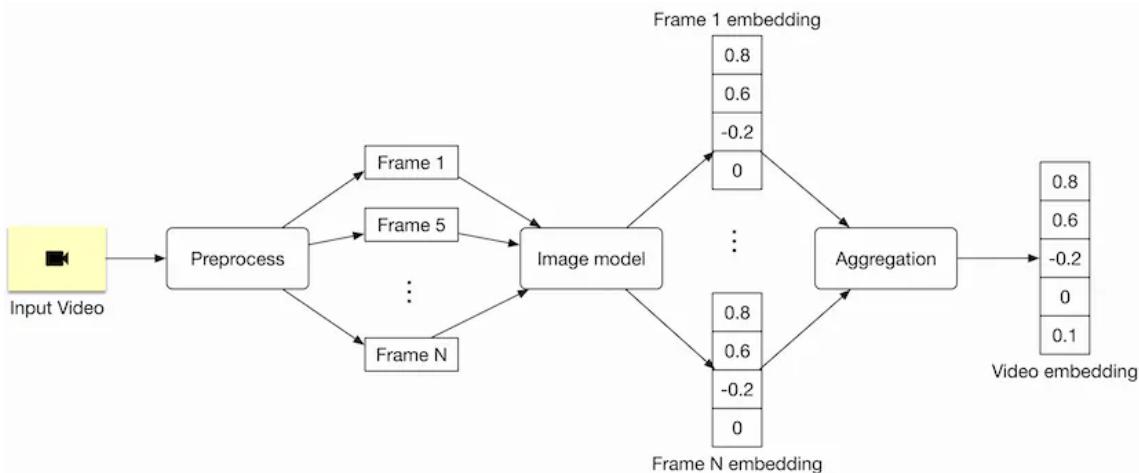


图 4.16: 帧级模型

由于此模型在帧级工作，因此通常更快且计算成本较低。然而，帧级模型通常无法理解视频的时间方面，如动作和运动。在实践中，帧级模型在视频的时间理解不太重要的情况下更受欢迎。在这里，我们采用帧级模型（如 ViT 16），原因如下：

- 提高训练和推理速度
- 减少计算量

4.5.5 模型训练

为了训练文本编码器和视频编码器，我们使用对比学习方法。如果您有兴趣了解更多，请参阅第 2 章视觉搜索系统中的“模型训练”部分。

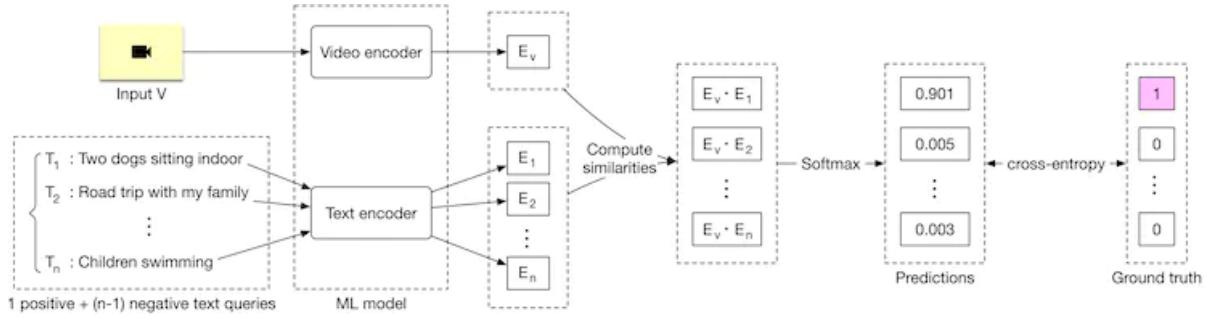


图 4.17: 损失计算

4.6 评估

4.6.1 离线指标

以下是搜索系统中常用的一些离线指标。让我们看看哪些是最相关的。

Precision@k 和 mAP

$$\text{precision}@k = \frac{\text{number of relevant items among the top } k \text{ items in the ranked list}}{k}$$

在评估数据集中，给定的文本查询只关联一个视频。这意味着 precision@k 公式的分子最多为 1。这导致 precision@k 的值较低。例如，对于给定的文本查询，即使我们将其关联的视频排名在列表顶部，precision@10 也只有 0.1。由于这一局限性，precision 类指标（如 precision@k 和 mAP）并不十分有用。

Recall@k. 这衡量搜索结果中相关视频数量与总相关视频数量之间的比率。

$$\text{recall}@k = \frac{\text{Number of relevant videos among the top } k \text{ videos}}{\text{Total number of relevant videos}}$$

如前所述，“总相关视频数”始终为 1。因此，我们可以将 recall@k 公式转换为以下内容：

$$\text{recall}@k = \begin{cases} 1 & \text{如果相关视频在前 } k \text{ 个视频中} \\ 0 & \text{否则} \end{cases}$$

这个指标有哪些优缺点？

优点

- 它有效地衡量了模型查找给定文本查询相关视频的能力。

缺点

- 它依赖于 k 的选择。选择合适的 k 可能具有挑战性。
- 当相关视频不在输出列表的前 k 个视频中时，recall@k 始终为 0。例如，假设模型 A 将相关视频排在第 15 位，模型 B 将相同视频排在第 50 位。如果我们使用 recall@10 来衡量这两个模型的质量，那么两者的 recall@10 都为 0，即使模型 A 比模型 B 更好。

平均倒数排名 (MRR)。该指标通过计算每个搜索结果中第一个相关项的排名并求平均来衡量模型的质量。公式为：

$$\text{MRR} = \frac{1}{m} \sum_{i=1}^m \frac{1}{\text{rank}_i}$$

该指标解决了 recall@k 的不足之处，可以作为我们的离线指标。

4.6.2 在线指标

作为在线评估的一部分，公司会跟踪多种指标。以下是一些最重要的指标：

- 点击率 (CTR)
- 视频完成率
- 搜索结果的总观看时间

点击率 (CTR)。该指标显示用户点击检索到的视频的频率。CTR 的主要问题是它不能跟踪点击的视频是否对用户有用。尽管存在这个问题，CTR 仍然是一个很好的指标，因为它显示了有多少人点击了搜索结果。

视频完成率。该指标衡量有多少视频出现在搜索结果中，并被用户观看到结束。该指标的问题在于，用户可能只观看了视频的一部分，但仍然觉得它相关。视频完成率本身无法反映搜索结果的相关性。

搜索结果的总观看时间。该指标跟踪用户观看搜索结果返回的视频的总时间。如果搜索结果相关，用户往往会花更多时间观看。该指标是衡量搜索结果相关性的良好指标。

4.7 服务

在服务阶段，系统根据给定的文本查询显示一份排序的视频列表。图 4.19 显示了一个简化的 ML 系统设计。

让我们更详细地讨论每个流程。

4.7.1 预测流程

该流程包括：

- 视觉搜索
- 文本搜索
- 融合层
- 重新排序服务

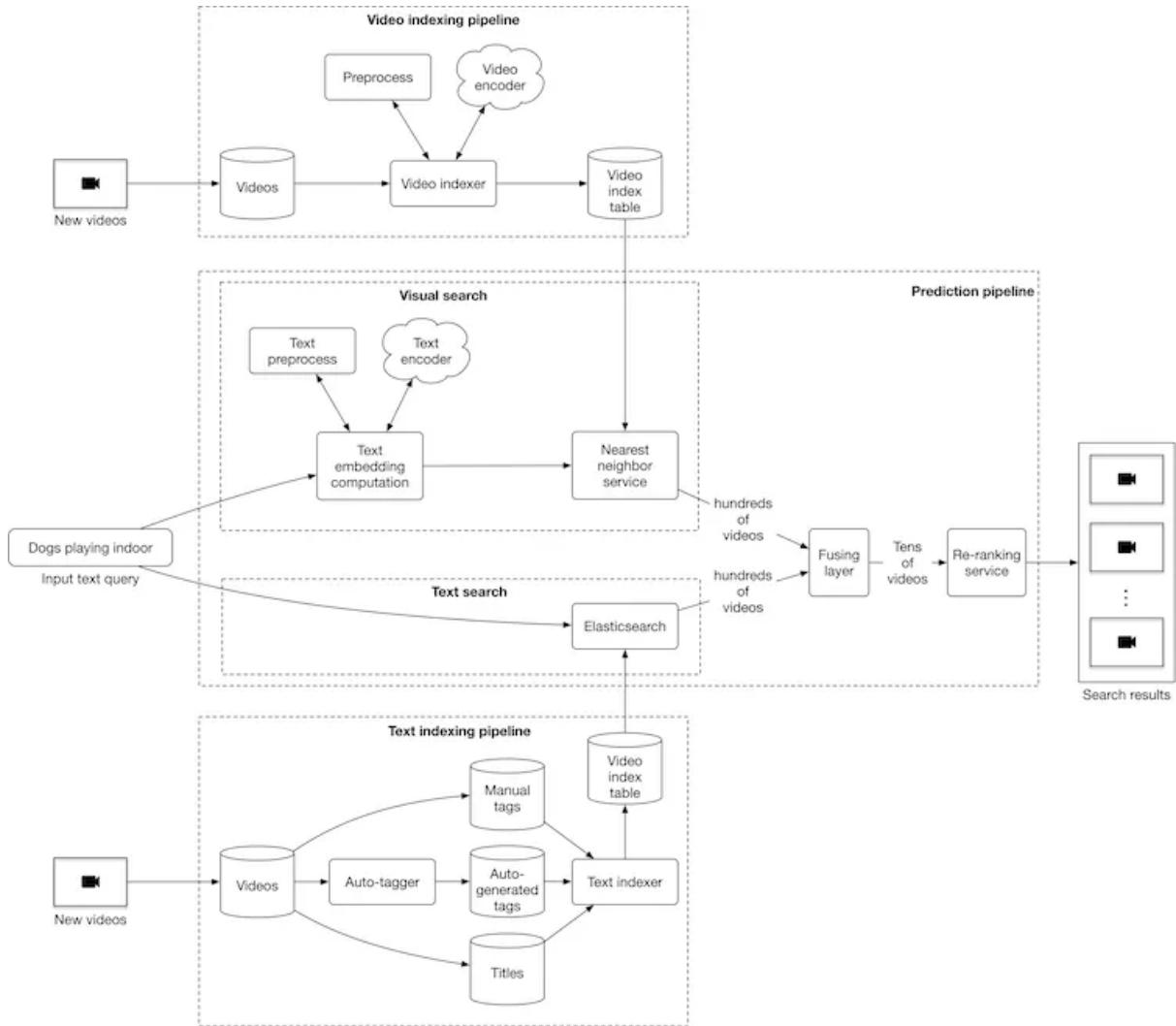


图 4.18: ML 系统设计

视觉搜索。 该组件对文本查询进行编码，并使用最近邻服务查找与文本嵌入最相似的视频嵌入。为了加速最近邻搜索，我们使用近似最近邻（ANN）算法，如第 2 章视觉搜索系统中所述。

文本搜索。 使用 Elasticsearch，该组件查找与文本查询重叠的标题和标签的视频。

融合层。 该组件从前一步中获取两个不同的相关视频列表，并将它们合并为一个新的视频列表。

融合层可以通过两种方式实现，其中最简单的一种是根据预测相关性分数的加权和重新排序视频。更复杂的方法是采用额外的模型来重新排序视频，这种方法更昂贵，因为它需要模型训练。此外，它在服务时更慢。因此，我们使用前一种方法。

重新排序服务。 该服务通过结合业务层逻辑和策略来修改视频的排序列表。

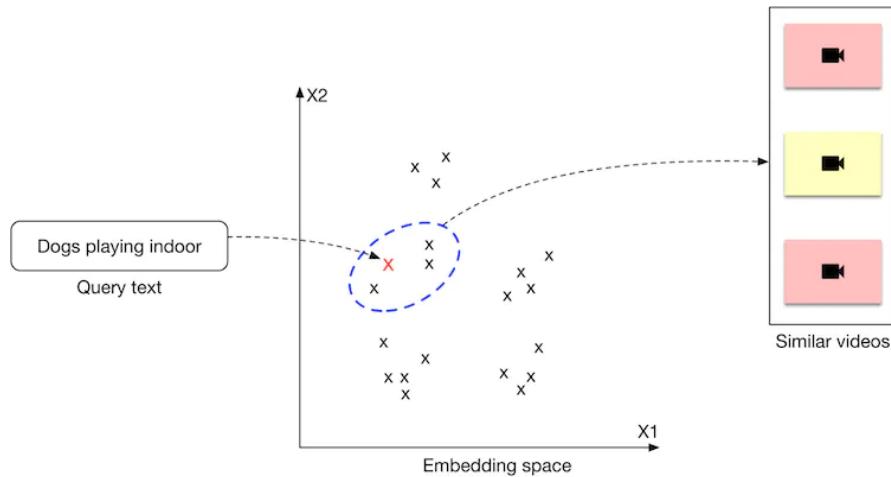


图 4.19: 检索给定文本查询的前 3 个结果

4.7.2 视频索引流程

使用训练好的视频编码器计算视频嵌入，然后对其进行索引。这些索引的视频嵌入由最近邻服务使用。

4.7.3 文本索引流程

该流程使用 Elasticsearch 对标题、手动标签和自动生成的标签进行索引。

通常，当用户上传视频时，他们会提供标签以帮助更好地识别视频。但是，如果他们没有手动输入标签怎么办？一个选择是使用独立模型来生成标签。我们将这个组件称为自动标签器，它在视频没有手动标签的情况下特别有价值。虽然这些标签可能比手动标签更嘈杂，但它们仍然有价值。

4.8 其他讨论要点

在结束本章之前，需要注意的是，我们简化了视频搜索系统的系统设计。实际上，它要复杂得多。一些改进可能包括：

- 使用多阶段设计（候选生成 + 排名）。
- 使用更多视频特征，如视频长度、视频受欢迎程度等。
- 不要依赖注释数据，而是使用交互（例如点击、喜欢等）来构建和标记数据。这使我们能够不断训练模型。
- 使用 ML 模型查找与文本查询语义相似的标题和标签。此模型可以与 Elasticsearch 结合使用以提高搜索质量。

如果访谈结束时还有时间，以下是一些额外的谈话要点：

- 搜索系统中的一个重要主题是查询理解，例如拼写纠正、查询类别识别和实体识别。如何构建查询理解组件？[17](#)。

- 如何构建处理语音和音频以改进搜索结果的多模式系统 [18](#)。
- 如何扩展这项工作以支持其他语言 [19](#)。
- 最终输出中的近似重复视频可能会对用户体验产生负面影响。如何检测近似重复的视频，以便在显示结果之前将其删除 [20](#)？
- 文本查询可分为头部、躯干和尾部查询。每种情况下常用的不同方法是什么 [21](#)？
- 生成输出列表时如何考虑流行度和新鲜度 [22](#)？
- 现实世界的搜索系统如何工作 [23](#) [24](#) [25](#)。

References

1. Elasticsearch. https://www.tutorialspoint.com/elasticsearch/elasticsearch_query_dsl.htm.
2. Preprocessing text data. <https://huggingface.co/docs/transformers/preprocessing>.
3. NFKD normalization. <https://unicode.org/reports/tr15/>.
4. What is Tokenization summary. https://huggingface.co/docs/transformers/tokenizer_summary.
5. Hash collision. https://en.wikipedia.org/wiki/Hash_collision.
6. Deep learning for NLP. http://cs224d.stanford.edu/lecture_notes/notes1.pdf.
7. TF-IDF. <https://en.wikipedia.org/wiki/Tf%E2%80%93idf>.
8. Word2Vec models. <https://www.tensorflow.org/tutorials/text/word2vec>.
9. Continuous bag of words. <https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-fea.html>.
10. Skip-gram model. <http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/>.
11. BERT model. <https://arxiv.org/pdf/1810.04805.pdf>.
12. GPT3 model. <https://arxiv.org/pdf/2005.14165.pdf>.
13. BLOOM model. <https://bigscience.huggingface.co/blog/bloom>.
14. Transformer implementation from scratch. <https://peterbloem.nl/blog/transformers>.
15. 3D convolutions. <https://www.kaggle.com/code/shivamb/3d-convolutions-understanding-use-case/notebook>.
16. Vision Transformer. <https://arxiv.org/pdf/2010.11929.pdf>.
17. Query understanding for search engines. <https://www.linkedin.com/pulse/ai-query-understanding-daniel-tunkelang/>.
18. Multimodal video representation learning. <https://arxiv.org/pdf/2012.04124.pdf>.
19. Multilingual language models. <https://arxiv.org/pdf/2107.00676.pdf>.
20. Near-duplicate video detection. <https://arxiv.org/pdf/2005.07356.pdf>.
21. Generalizable search relevance. <https://livebook.manning.com/book/ai-powered-search/chapter-10/v-10/20>.
22. Freshness in search and recommendation systems. <https://developers.google.com/machine-learning/recognition/dnn/re-ranking>.

23. Semantic product search by Amazon. <https://arxiv.org/pdf/1907.00937.pdf>.
24. Ranking relevance in Yahoo search. <https://www.kdd.org/kdd2016/papers/files/adf0361-yinA.pdf>.
25. Semantic product search in E-Commerce. <https://arxiv.org/pdf/2008.08180.pdf>.

5 Harmful Content Detection 有害内容检测

5.1 Introduction

许多社交媒体平台，如 Facebook 1、LinkedIn 2 和 Twitter 3，都有标准指南来维护平台的完整性并确保用户的安全。这些指南禁止某些对社区有害的用户行为、活动和内容。因此，拥有识别有害内容和不良行为者的技术和资源至关重要。我们可以将完整性维护的重点分为两个类别：

- **有害内容：**包含暴力、裸露、自残、仇恨言论等的帖子。
- **不良行为/不良行为者：**假账户、垃圾信息、网络钓鱼、组织的不道德活动以及其他不安全行为。

本章我们重点讨论检测可能包含有害内容的帖子。具体来说，我们设计一个系统，主动监控新发布的帖子，检测有害内容，并在内容违反平台指南时删除或降级这些内容。要了解公司如何在实践中构建有害内容检测系统，请参考 4 5 6。

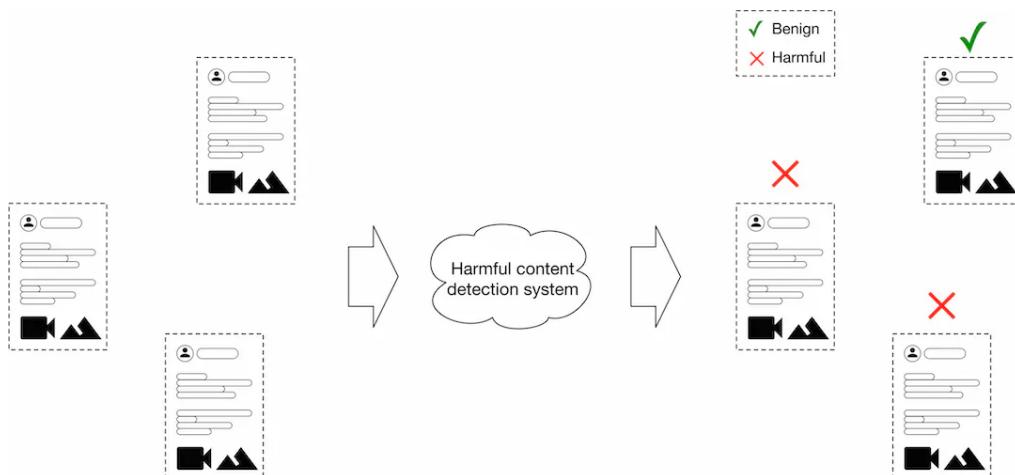


图 5.1: 有害内容检测系统

5.2 明确需求

以下是候选人与面试官之间的典型对话。

候选人：系统是否检测有害内容和不良行为者？

面试官：两者同样重要。为了简单起见，我们仅关注检测有害内容。

候选人：帖子只能包含文本，还是可以包含图像和视频？

面试官：帖子的内容可以是文本、图像、视频或这些内容的任意组合。

候选人：系统支持哪些语言？仅限于英语吗？

面试官：系统应能检测多种语言的有害内容。为了简单起见，假设我们可以使用预训练的多语言模型来嵌入文本内容。

候选人：我们希望识别哪些特定类别的有害内容？我能想到暴力、裸露、仇恨言论、虚假信息等。还有其他需要考虑的有害类别吗？

面试官：你提到了主要的类别。虚假信息更为复杂且有争议。为了简单起见，我们不关注虚假信息。

候选人：是否有人工标注人员可以手动标记帖子？

面试官：平台每天接收超过 5 亿条帖子。让人类标注所有这些帖子将非常昂贵且耗时。然而，可以假设每天有 10,000 条帖子可以被人工标注。

候选人：允许用户举报有害内容有助于了解系统的不足之处。我可以假设系统具备此功能吗？

面试官：很好的观点。是的，用户可以举报有害帖子。

候选人：我们是否需要解释为什么某个帖子被认为是有害的并因此被删除？

面试官：是的。向用户解释为什么我们删除帖子很重要。这有助于用户确保他们的未来帖子符合指南。

候选人：系统的延迟要求是什么？我们是否需要实时预测，即系统立即检测到有害内容并屏蔽它，还是可以依赖批量预测，即每小时或每天离线检测有害内容？

面试官：这是一个非常重要的问题。你怎么看？

候选人：我认为对于不同类型的有害内容，要求可能有所不同。例如，暴力内容可能需要实时解决方案，而对于其他类型的内容，延迟检测可能也可以接受。

面试官：这些假设是合理的。

那么，让我们总结一下问题陈述。我们将设计一个有害内容检测系统，该系统可以识别有害帖子，然后删除或降级它们，并告知用户为什么该帖子被识别为有害。帖子的内容可以是文本、图像、视频或这些内容的任意组合，并且可以使用不同的语言。用户可以举报有害帖子。

5.3 将问题框架化为一个机器学习任务

5.3.1 定义机器学习目标

我们的机器学习目标是准确地预测有害帖子。因为如果我们能够准确地检测到有害帖子，就可以将其删除或降级，从而创建一个更安全的平台。

5.3.2 设定系统的输入和输出

系统接收一个帖子作为输入，并输出该帖子是有害的概率。

让我们更深入地探讨输入的帖子。如图 5.3 所示，一个帖子可以是异构的，并且可能是多模态的。

为了做出准确的预测，系统应考虑所有模态。让我们讨论两种常用的融合方法来结合异构数据：后期融合和前期融合。

5.3.2.1 后期融合

在后期融合中，机器学习模型独立处理不同的模态，然后结合它们的预测来得出最终的预测。下图展示了后期融合的工作原理。

后期融合的优点是我们可以独立地训练、评估和改进每个模型。

然而，后期融合有两个主要缺点。首先，要训练这些单独的模型，我们需要为每个模态准备单独的训练数据，这可能耗时且成本高昂。

其次，即使每个模态单独来看都是良性的，它们的组合也可能是有害的。尤其是在结合图像和文本的表情包（memes）中，这种情况很常见。在这些情况下，后期融合无法正确预测内容是否有害。这是因

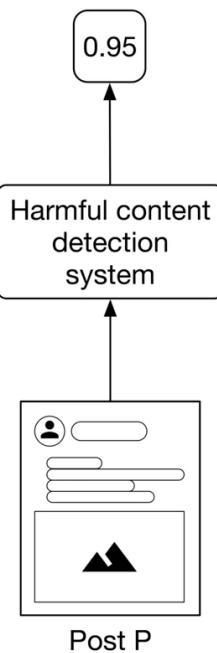


图 5.2: 有害内容检测系统的输入输出

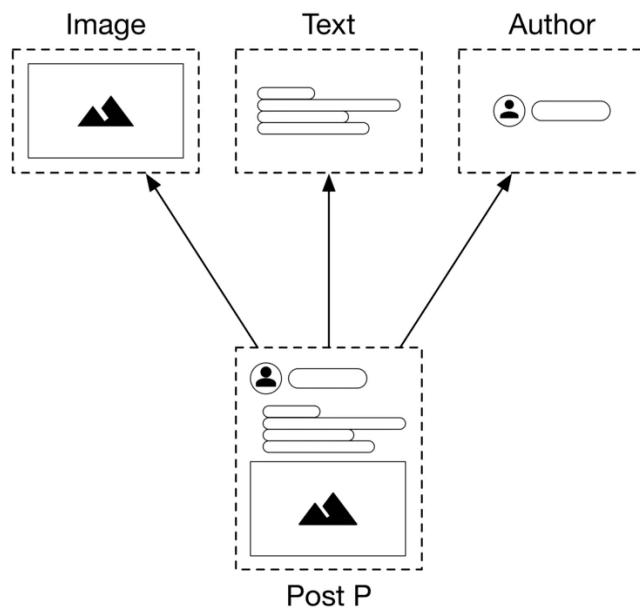


图 5.3: 异构的帖子数据

为每个模态单独预测时都是良性的，所以融合层的输出也是良性的。但这其实是错误的，因为模态的组合可能是有害的。

5.3.2.2 前期融合

在前期融合中，首先将各个模态结合，然后模型进行预测。图 5.5 展示了前期融合的工作原理。

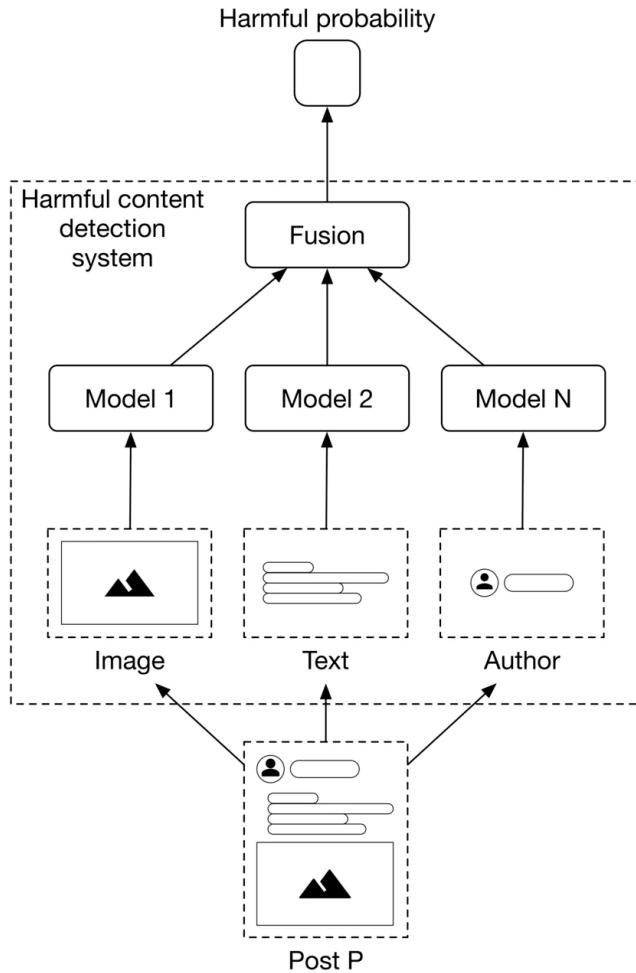


图 5.4: 后期融合

前期融合有两个主要优点。首先，不需要为每个模态单独收集训练数据。由于只需训练一个模型，我们只需为该模型收集训练数据。其次，模型会考虑所有模态，如果每个模态单独来看都是良性的，但它们的组合是有害的，那么模型可以在统一的特征向量中捕捉到这一点。

然而，由于模态之间关系复杂，模型学习这个任务更加困难。在缺乏足够训练数据的情况下，模型很难学习复杂的关系并做出良好的预测。

5.3.2.3 我们应该使用哪种融合方法？

我们使用前期融合方法，因为它使我们能够捕捉那些整体上可能有害的帖子，即使每个模态单独来看是良性的。此外，每天发布的帖子数量约为 5 亿，模型有足够的数据来学习这一任务。

5.3.3 选择合适的机器学习类别

在本节中，我们考察以下机器学习类别的选项：

- 单一二分类器

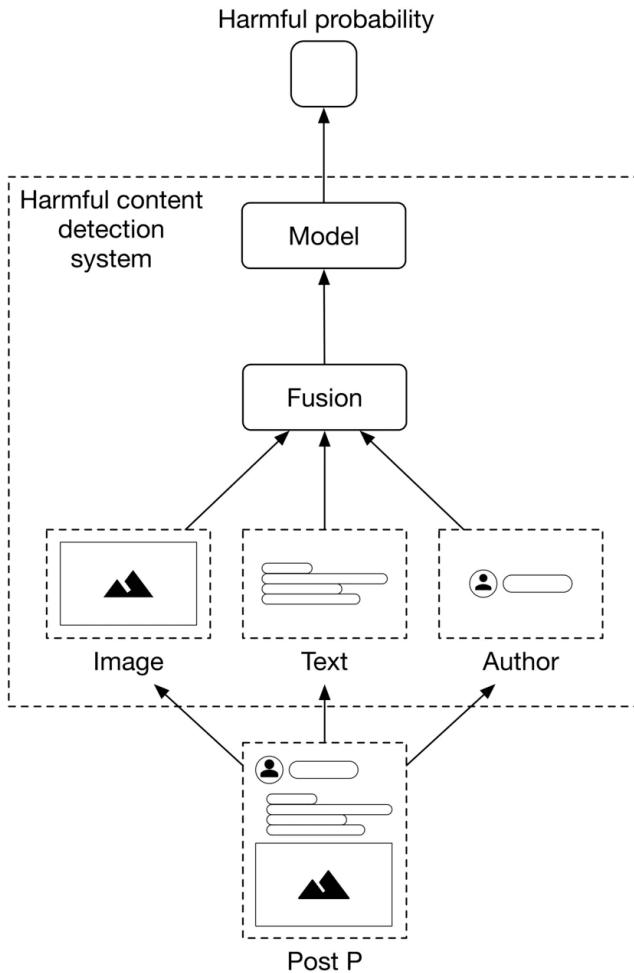


图 5.5: 前期融合

- 针对每个有害类别的二分类器
- 多标签分类器
- 多任务分类器

5.3.3.1 单一二分类器

在这种情况下，模型以融合的特征为输入，并预测该帖子为有害的概率（图 5.6）。由于输出是二元结果，该模型为二分类器。

这种选择的缺点是，很难确定帖子属于哪种有害类别，例如暴力。这一限制会导致两个主要问题：

- 由于系统仅输出表示帖子整体是否有害的二元值，我们很难向用户解释为什么要删除某个帖子。我们无法知道该帖子属于哪个特定的有害类别。
- 难以识别系统在表现不佳的有害类别，从而无法改进在这些类别上的检测能力。

由于解释删除帖子原因至关重要，单一二分类器不是一个好的选择。

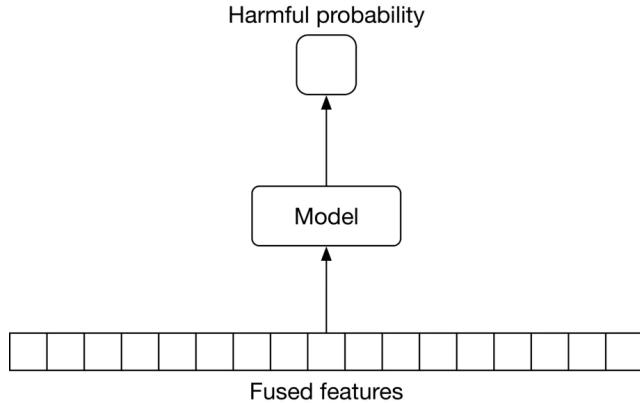


图 5.6: 单一二分类器

5.3.3.2 针对每个有害类别的二分类器

在这种情况下，我们为每个有害类别采用一个二分类器。如图 5.7 所示，每个模型确定一个帖子是否属于某个特定的有害类别。每个模型以融合的特征为输入，并预测该帖子属于某个有害类别的概率。

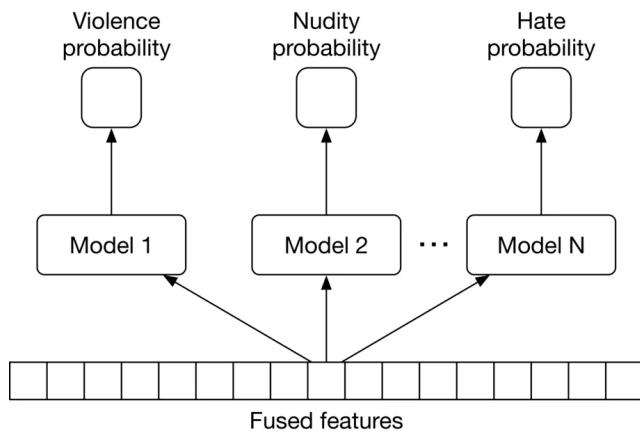


图 5.7: 针对每个有害类别的二分类器

这种选择的优点是我们可以向用户解释为什么某个帖子被删除。此外，我们可以独立地监控和改进不同的模型。

然而，这种选择有一个主要缺点。由于有多个模型，它们必须分别进行训练和维护。分别训练这些模型耗时且成本高。

5.3.3.3 多标签分类器

在多标签分类中，我们希望分类的数据点可能属于任意数量的类别。在这种情况下，使用一个单一模型作为多标签分类器。如图 5.8 所示，模型的输入是融合的特征，模型预测每个有害类别的概率。

通过使用一个共享模型处理所有有害类别，训练和维护的成本较低。如果你想了解更多关于此方法的信息，可以参考一种称为 WPIE 的方法 [7](#)。

然而，使用共享模型来预测每个有害类别的概率并不理想，因为输入特征可能需要不同的转换方式。

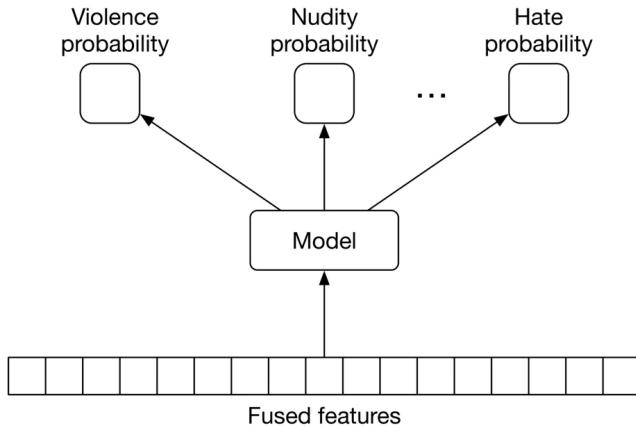


图 5.8: 多标签分类器

5.3.3.4 多任务分类器

多任务学习是指模型同时学习多个任务的过程。这使模型能够学习任务之间的相似性。通过这种方式，当某个输入转换对多个任务有用时，可以避免不必要的计算。

在我们的案例中，我们将不同的有害类别（如暴力和裸露）视为不同的任务，并使用多任务分类模型来学习每个任务。如图 5.9 所示，多任务分类有两个阶段：共享层和任务特定层。

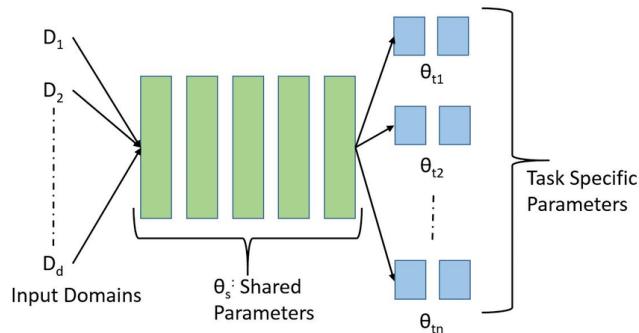


图 5.9: 多任务分类概述

共享层 如图 5.10 所示，共享层是一组隐藏层，将输入特征转换为新的特征。这些新转换的特征用于对每个有害类别进行预测。

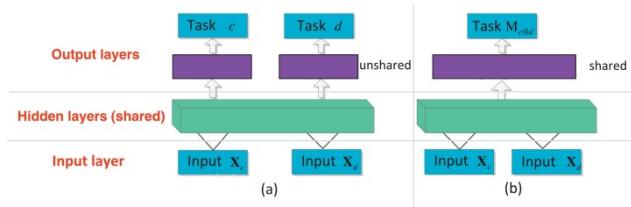


图 5.10: 共享层

任务特定层 任务特定层是一组独立的机器学习层（也称为分类头）。每个分类头以最适合预测某个特定有害类别概率的方式转换特征。

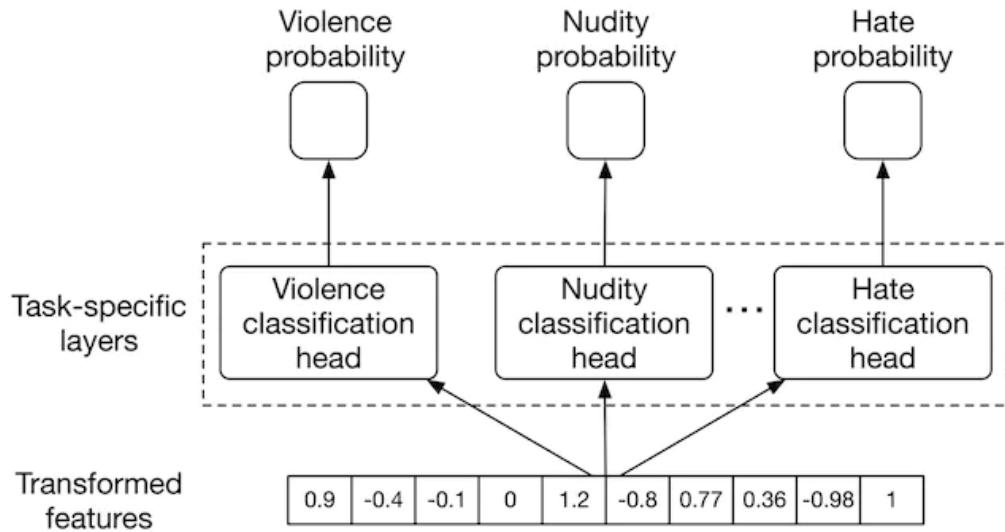


图 5.11: 任务特定层

多任务分类有三个优点。首先，由于使用的是单一模型，训练和维护的成本较低。其次，共享层以有利于每个任务的方式转换特征。这避免了冗余计算，使多任务分类更高效。最后，每个任务的训练数据都可以促进其他任务的学习。当某个任务的数据有限时，这尤其有帮助。

由于这些优势，我们采用多任务分类方法。图 5.12 展示了我们如何将问题框架化。

5.4 数据准备

5.4.1 数据工程

我们有以下可用数据：

- 用户
- 帖子
- 用户-帖子交互

5.4.1.1 用户

以下是用户数据的模式。

ID	Username	Age	Gender	City	Country	Email
----	----------	-----	--------	------	---------	-------

表 5.1: 用户数据模式

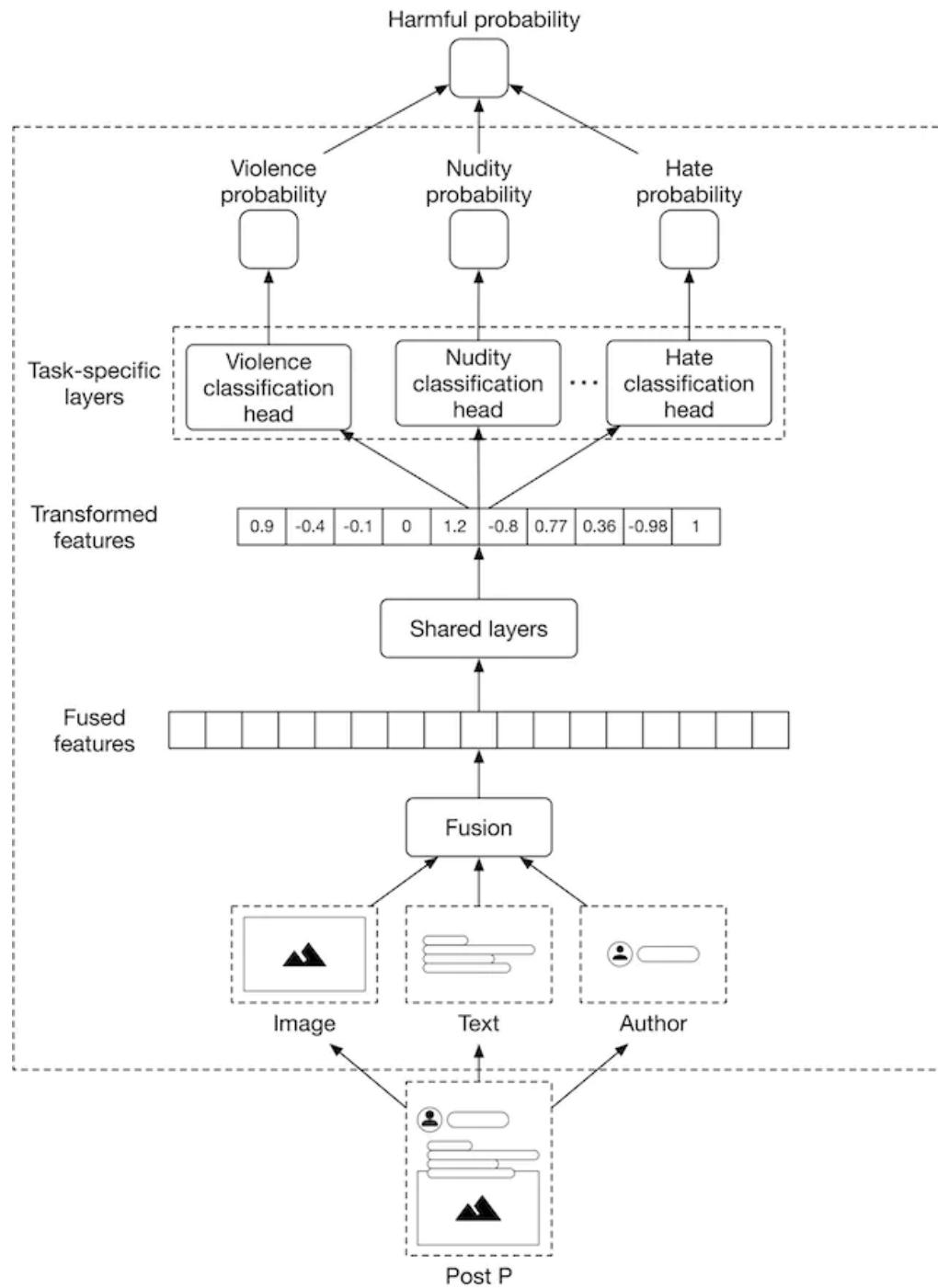


图 5.12: 将问题框架化为一个机器学习任务

5.4.1.2 帖子

帖子数据包含诸如作者、上传时间等字段。表 5.2 展示了一些最重要的属性。实际上，每个帖子通常有数百个相关属性。

Post ID	Author ID	On-device	Timestamp	Textual content	Images or videos	Links
1	1	73.93.220.240	1658469431	Today, I am starting my diet.	http://cdn.mysite.com/u1.jpg	-
2	11	89.42.110.250	1658471428	The video amazed me! Please donate	http://cdn.mysite.com/t3.mp4	gofundme.com/f/3u1njd32
3	4	39.55.180.020	1658489233	What is a good restaurant in the Bay area?	http://cdn.mysite.com/t5.jpg	-

表 5.2: 帖子数据

5.4.1.3 用户-帖子交互

用户-帖子交互数据主要包括用户对帖子的反应，如点赞、评论、收藏、分享等。用户还可以举报帖子为有害或请求申诉。表 5.3 展示了这些数据的样例。

User ID	Post ID	Interaction type	Interaction value	Timestamp
11	6	Impression	-	1658450539
4	20	Like	-	1658451341
11	7	Comment	This is disgusting	1658451365
4	20	Share	-	1658435948
11	7	Report	violence	1658451849

表 5.3: 用户-帖子交互数据

5.4.2 特征工程

在“将问题框架化为机器学习任务”一节中，我们将问题框架化为一个多任务分类任务，其中输入是帖子。在本节中，我们探讨可以从帖子中提取的预测特征。一个帖子可能包括以下元素：

- 文本内容
- 图片或视频
- 用户对帖子的反应
- 作者
- 上下文信息

让我们看看每个元素。

5.4.2.1 文本内容

帖子的文本内容可以用来判断帖子是否有害。如第 4 章 YouTube 视频搜索中所述，文本数据通常分两步准备：

- 文本预处理（如归一化、分词）
- 向量化：将预处理后的文本转换为有意义的特征向量

让我们专注于向量化，因为这是本章的独特内容。为了向量化文本并提取特征向量，可以使用统计方法或基于机器学习的方法。统计方法如 BoW 或 TF-IDF 易于实现且计算速度快。然而，它们无法编码文本的语义。对于我们的系统，理解文本内容的语义对判断是否有害很重要，因此我们采用基于机器学习的方法。为了将文本转换为特征向量，我们使用预训练的基于 Transformer 的语言模型，如 BERT⁸。然而，原始的 BERT 存在两个问题：

- 由于模型规模较大，生成文本嵌入需要很长时间。由于这个过程较慢，将其用于在线预测并不理想。
- BERT 是在仅包含英语的数据上训练的。因此，对于其他语言的文本，它无法生成有意义的嵌入。

DistilBERT⁹ 是 BERT 的更高效变体，解决了这两个问题。如果两句话的意思相同，但用不同语言表达，它们的嵌入会非常相似。如果你对多语言语言模型感兴趣，可以参考¹⁰。

5.4.2.2 图片或视频

通常可以通过查看帖子中的图片或视频来了解其内容。以下两个步骤常用于准备图像或视频等非结构化数据：

- **预处理**：解码、调整大小和归一化数据。
- **特征提取**：预处理后，我们使用预训练模型将非结构化数据转换为特征向量。这使我们能够用特征向量表示图像或视频。对于图像，预训练的图像模型如 CLIP 的视觉编码器¹¹ 或 SimCLR¹² 是可行的选择。对于视频，像 VideoMoCo¹³ 这样的预训练模型可能表现良好。

5.4.2.3 用户对帖子的反应

根据用户的反应，也可以判断帖子是否有害，尤其是在内容模棱两可的情况下。如图 5.13 所示，随着评论数量的增加，越来越明显地表明该帖子包含与自残相关的内容。

由于用户反应在判断有害内容时至关重要，让我们探讨一些可以基于用户反应构建的特征。

点赞数、分享数、评论数和举报数：我们通常会对这些数值进行缩放，以加快模型训练时的收敛速度。

评论：如图 5.13 所示，评论可以帮助我们识别有害内容。为了准备特征，我们通过以下方式将评论转换为数值表示：

- 使用之前使用的预训练模型获取每条评论的嵌入。
- 聚合（如平均）嵌入以获得最终的嵌入。

图 5.14 总结了我们迄今为止描述的特征。

5.4.2.4 作者特征

作者的过去互动记录可以用来判断帖子是否有害。让我们为帖子作者构建相关特征。

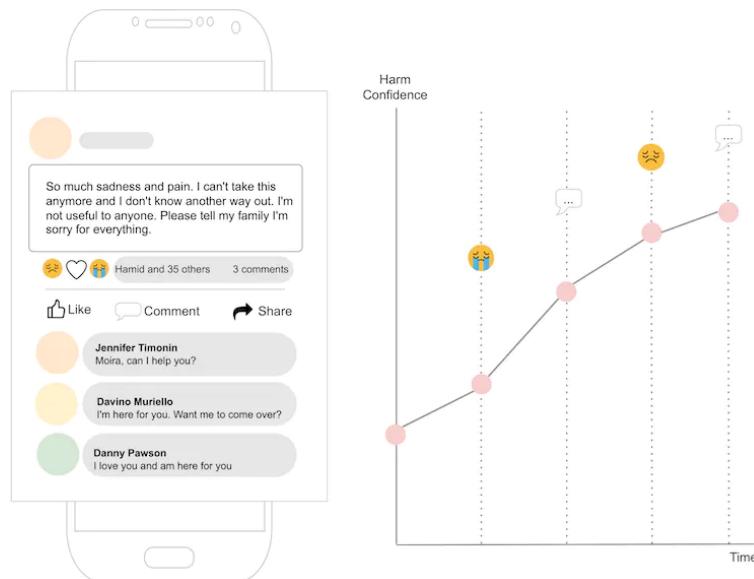


图 5.13: 带有自残内容的帖子

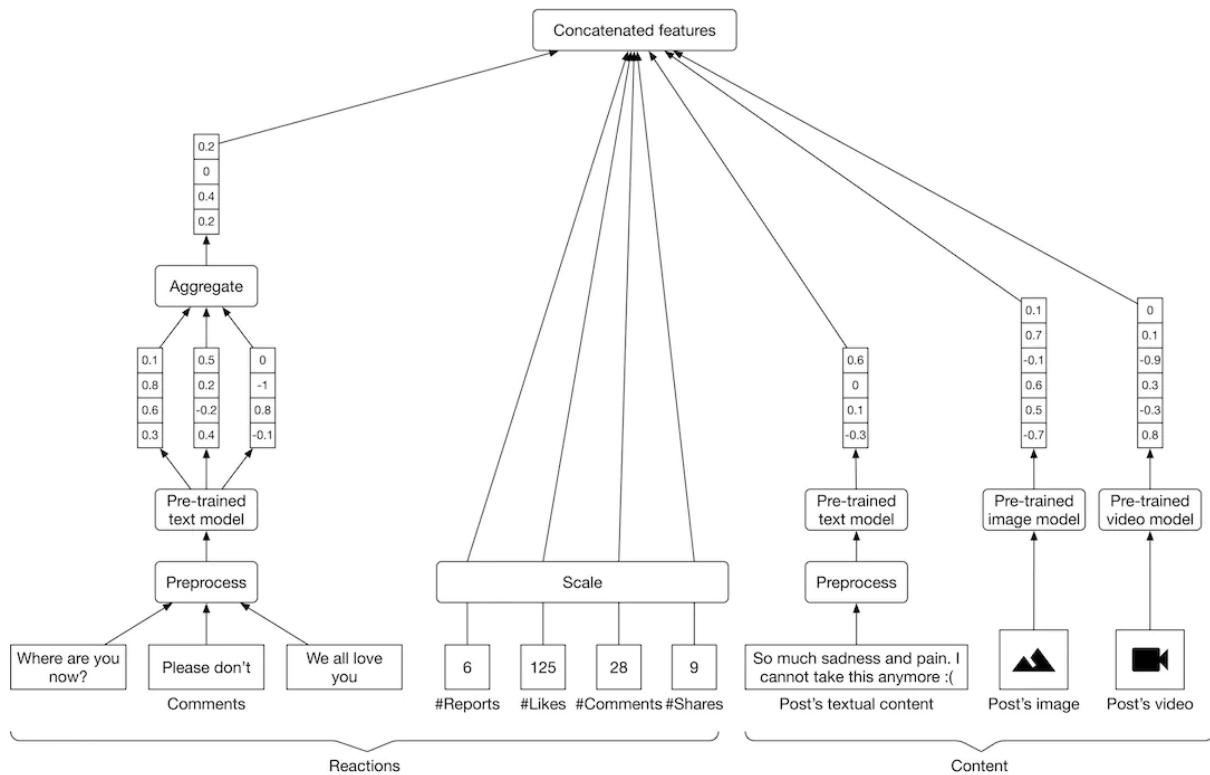


图 5.14: 针对反应和内容的特征工程

作者的违规历史

- **违规次数:** 这是一个数值，表示作者过去违反平台规定的次数。
- **总用户举报数:** 一个数值，表示用户举报该作者帖子次数的总和。

- **脏话率**: 这是一个数值，表示作者之前的帖子和评论中脏话的比例。我们使用预定义的脏话列表来判断一个词是否是脏话。

作者的用户信息

- **年龄**: 用户年龄是最重要的预测特征之一。
- **性别**: 这是表示用户性别的分类特征。我们使用独热编码来表示性别。
- **城市和国家**: 城市和国家有很多不同的取值。我们使用嵌入层将城市和国家转换为特征向量。请注意，使用独热编码表示城市和国家并不是一种高效的方法，因为它们的表示会很长且稀疏。

账户信息

- **关注者和关注数量**
- **账户年龄**: 这是一个数值，表示作者账户的年龄。这是一个预测特征，因为账户年龄较低的更有可能是垃圾账户或违反平台规定的账户。

5.4.2.5 上下文信息

- **一天中的时间**: 这是作者发帖时的一天中的时间。我们将其分类为多个类别，如早晨、中午、下午、傍晚或晚上，并使用独热编码来表示该特征。
- **设备**: 作者使用的设备，如智能手机或桌面电脑。我们使用独热编码表示该特征。

图 5.15 总结了有害内容检测系统的一些最重要特征。

5.5 模型开发

5.5.1 模型选择

神经网络是多任务学习中最常用的模型。在开发我们的模型时，我们采用神经网络。

选择神经网络时应考虑哪些因素？有必要确定神经网络的架构设计和最佳超参数选择，如其隐藏层、激活函数、学习率等。最佳超参数的选择通常通过超参数调优来确定。让我们简要介绍一下。

超参数调优是为模型找到最佳超参数值以实现最佳性能的过程。网格搜索通常用于调优超参数。其过程包括为每种超参数值组合训练一个新模型，评估每个模型，然后选择表现最好的超参数组合。如果你想了解更多关于超参数调优的内容，请参考 [14](#)。

5.5.2 模型训练

构建数据集 要训练多任务分类模型，我们首先需要构建数据集。该数据集包含模型的输入（特征）和模型预期预测的输出（标签）。为构建输入，我们将帖子按批次离线处理，并计算前述的融合特征。这些特征可以存储在特征库中以供未来训练使用。为了为每个输入创建标签，我们有两种选择：

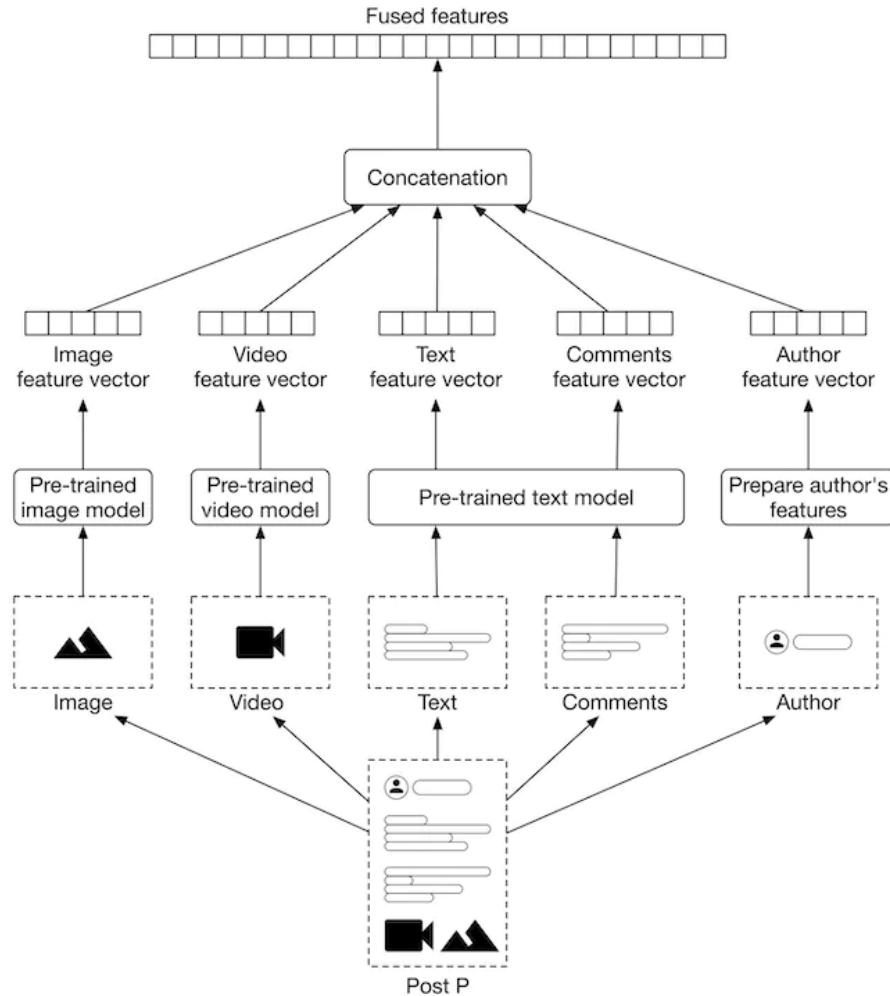


图 5.15: 特征工程总结

- 手动标注
- 自然标注

在手动标注中，人类标注人员手动标记帖子。这种方法能生成准确的标签，但成本高且耗时长。在自然标注中，我们依赖用户举报自动标记帖子。虽然这种方法产生的标签噪声较大，但标签生成速度更快。在评估数据集中，我们使用手动标注以优先保证标签的准确性；而在训练数据集中，我们使用自然标注以优先保证标记速度。图 5.16 显示了构建的数据集中的一个数据点。

选择损失函数 训练多任务神经网络与我们通常训练神经网络模型的方式非常相似。前向传播进行计算以进行预测，损失函数测量预测的准确性，反向传播优化模型的参数以在下一次迭代中减少损失。让我们看看损失函数。在多任务训练中，每个任务根据其机器学习类别分配一个损失函数。在我们的例子中，每个任务都被框定为二元分类，所以我们为每个任务采用标准的二元分类损失（如交叉熵）。总体损失是通过结合特定任务的损失来计算的，如图 5.17 所示。

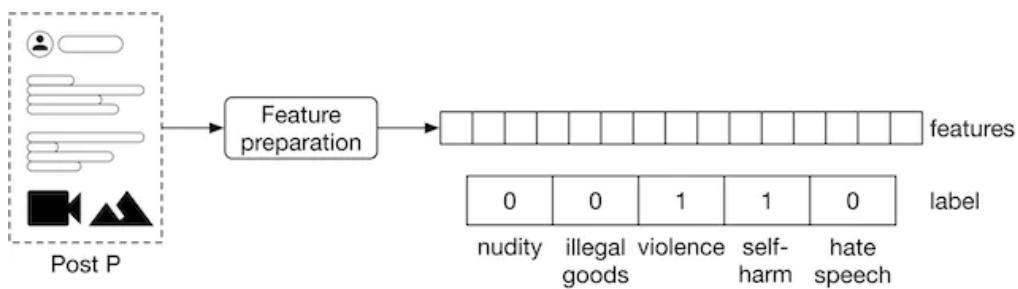


图 5.16: 构建的数据点

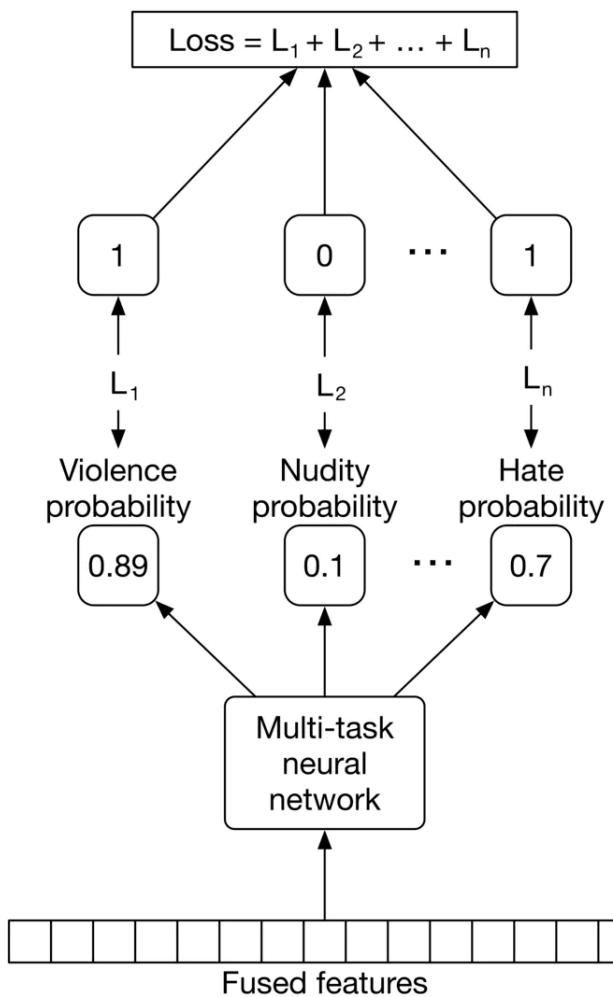


图 5.17: 模型训练

训练多模态系统的一个常见挑战是过拟合¹⁵。例如，当不同模态的学习速度不同时，某一模态（如图像）可能会主导学习过程。解决这个问题的两种技术是梯度混合和聚焦损失。如果你想了解更多关于这些技术的内容，请参考^{16 17}。

5.6 评估

5.6.1 离线指标

为了评估二元分类模型的性能，通常使用离线指标，如精确率、召回率和 f1 分数。然而，单独的精确率或召回率不足以理解整体性能。例如，精确率高的模型可能会有很低的召回率。精确率-召回（PR）曲线和受试者工作特征（ROC）曲线可以解决这些限制。让我们探讨一下每个指标。

PR 曲线。 PR 曲线显示了模型精确率和召回率之间的权衡。如图 5.18 所示，我们通过绘制模型在不同概率阈值下的精确率获得 PR 曲线，阈值范围从 0 到 1。为了总结精确率和召回率之间的权衡，PR-AUC（精确率-召回曲线下的面积）计算了 PR 曲线下的面积。一般来说，PR-AUC 越高，模型越准确。

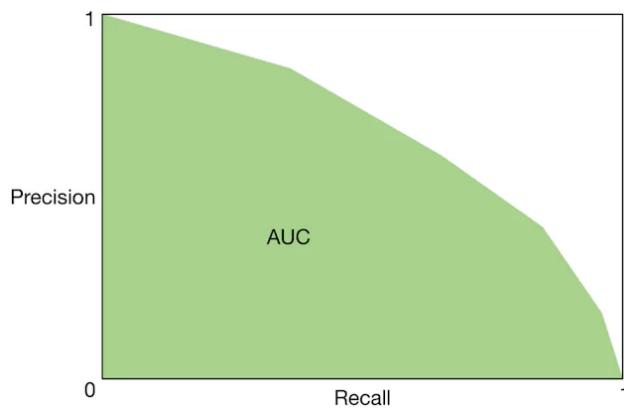


图 5.18: PR 曲线

ROC 曲线。 ROC 曲线显示了真正率（召回率）和假正率之间的权衡。与 PR 曲线类似，ROC-AUC 通过计算 ROC 曲线下的面积来总结模型的性能。

ROC 和 PR 曲线是总结分类模型性能的两种不同方法。要了解 PR 曲线和 ROC 曲线之间的区别，请阅读 [18](#)。

在我们的案例中，我们使用 ROC-AUC 和 PR-AUC 作为我们的离线指标。

5.6.2 在线指标

让我们探讨一些重要指标，以了解平台的安全性。

流行率。 该指标测量我们未能防止的有害帖子与平台上所有帖子之比。

$$\text{Prevalence} = \frac{\text{Number of harmful posts we didn't prevent}}{\text{Total number of posts on the platform}}$$

该指标的缺点是它将有害帖子一视同仁。例如，一个拥有 100K 次观看或展示的有害帖子比两个每个只有 10 次观看的帖子更有害。

有害展示次数。 相比流行率，我们更倾向于使用该指标。其原因在于平台上的有害帖子数量并未显示有多少人受到这些帖子的影响，而有害展示次数则捕捉到了这一信息。

有效申诉。 被判定为有害但通过申诉恢复的帖子百分比。

$$\text{Appeals} = \frac{\text{Number of reversed appeals}}{\text{Number of harmful posts detected by the system}}$$

主动率。系统在用户举报前发现并删除的有害帖子百分比。

$$\text{Proactive rate} = \frac{\text{Number of harmful posts detected by the system}}{\text{Number of harmful posts detected by the system} + \text{reported by users}}$$

每个有害类别的用户举报数。该指标通过查看用户对每个有害类别的举报情况来评估系统性能。

5.7 部署

图 5.19 展示了高层次的机器学习系统设计。让我们更详细地看看每个组件。

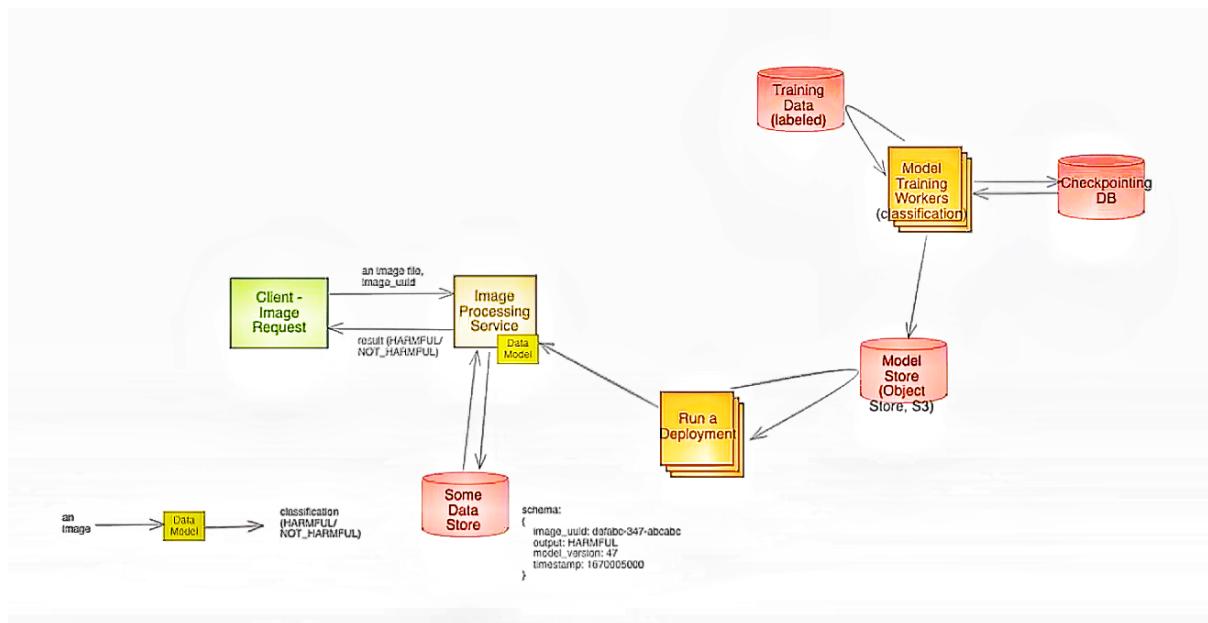


图 5.19: 机器学习系统设计

5.7.1 有害内容检测服务

给定一个新帖子，该服务预测其有害的概率。根据要求，由于某些有害类型的敏感性，需要立即处理。当发生这种情况时，违规执行服务会立即下架该帖子。

5.7.2 违规执行服务

当有害内容检测服务高置信度地预测到有害内容时，违规执行服务会立即下架该帖子。它还会通知用户为什么该帖子被删除。

5.7.3 降级服务

如果有害内容检测服务对有害的预测置信度较低，降级服务会暂时降低帖子的排名，以减少其在用户中传播的可能性。

然后，该帖子会存储在存储中，以供人工审核。审核团队手动审核该帖子，并从预定义的有害类别中分配标签。我们将在未来的训练迭代中使用这些标记的帖子来改进模型。

5.7.4 其他讨论点

- 处理人工标注引入的偏见 [19](#)。
- 使系统适应检测新的有害类别（如 Covid-19，选举）[20](#)。
- 如何构建一个利用用户行为序列等时间信息的有害内容检测系统 [21](#) [22](#)。
- 如何有效地选择帖子样本进行人工审核 [23](#)。
- 如何检测真实和虚假账户 [24](#)。
- 如何处理边界内容 [25](#)，即那些不被指南禁止但接近指南界限的内容。
- 如何使有害内容检测系统更高效，从而能在设备上部署 [26](#)。
- 如何用线性 transformers 代替基于 transformers 的架构，以创建更高效的系统 [27](#) [28](#)。

References

1. Facebook's inauthentic behavior. <https://transparency.fb.com/policies/community-standards/inauthentic-behavior/>.
2. LinkedIn's professional community policies. <https://www.linkedin.com/legal/professional-community-policies/>
3. Twitter's civic integrity policy. <https://help.twitter.com/en/rules-and-policies/election-integrity-policy>
4. Facebook's integrity survey. <https://arxiv.org/pdf/2009.10311.pdf>.
5. Pinterest's violation detection system. <https://medium.com/pinterest-engineering/how-pinterest-fights-spam-with-machine-learning-103a2a2a2a2a>
6. Abusive detection at LinkedIn. <https://engineering.linkedin.com/blog/2019/isolation-forest>.
7. WPIE method. <https://ai.facebook.com/blog/community-standards-report/>.
8. BERT paper. <https://arxiv.org/pdf/1810.04805.pdf>.
9. Multilingual DistilBERT. <https://huggingface.co/distilbert-base-multilingual-cased>.
10. Multilingual language models. <https://arxiv.org/pdf/2107.00676.pdf>.
11. CLIP model. <https://openai.com/blog/clip/>.
12. SimCLR paper. <https://arxiv.org/pdf/2002.05709.pdf>.
13. VideoMoCo paper. <https://arxiv.org/pdf/2103.05905.pdf>.
14. Hyperparameter tuning. <https://cloud.google.com/ai-platform/training/docs/hyperparameter-tuning>
15. Overfitting. <https://en.wikipedia.org/wiki/Overfitting>.
16. Focal loss. <https://amaarora.github.io/2020/06/29/FocalLoss.html>.
17. Gradient blending in multimodal systems. <https://arxiv.org/pdf/1905.12681.pdf>.
18. ROC curve vs precision-recall curve. <https://machinelearningmastery.com/roc-curves-and-precision-recall/>
19. Introduced bias by human labeling. <https://labelyourdata.com/articles/bias-in-machine-learning>.
20. Facebook's approach to quickly tackling trending harmful content. <https://ai.facebook.com/blog/harmful-content-can-evolve-quickly-our-new-ai-system-adapts-to-tackle-it/>.
21. Facebook's TIES approach. <https://arxiv.org/pdf/2002.07917.pdf>.
22. Temporal interaction embedding. <https://www.facebook.com/atscaleevents/videos/730968530723238/>.
23. Building and scaling human review system. <https://www.facebook.com/atscaleevents/videos/1201751883328695/>.
24. Abusive account detection framework. <https://www.youtube.com/watch?v=YeX4MdU0JNk>.

25. Borderline contents. <https://transparency.fb.com/features/approach-to-ranking/content-distribution-content-borderline-to-the-community-standards/>.
26. Efficient harmful content detection. <https://about.fb.com/news/2021/12/metas-new-ai-system-tackles-harmful-content/>.
27. Linear Transformer paper. <https://arxiv.org/pdf/2006.04768.pdf>.
28. Efficient AI models to detect hate speech. <https://ai.facebook.com/blog/how-facebook-uses-super-efficient-ai-models-to-detect-hate-speech/>.

6 Video Recommendation System 视频推荐系统

推荐系统在视频和音乐流媒体服务中起着关键作用。例如，YouTube 向用户推荐他们可能喜欢的视频，Netflix 向用户推荐他们可能喜欢观看的电影，Spotify 向用户推荐音乐。

在本章中，我们设计一个类似于 YouTube 的视频推荐系统¹。该系统基于用户的个人资料、先前的互动等内容在用户的主页上推荐视频。

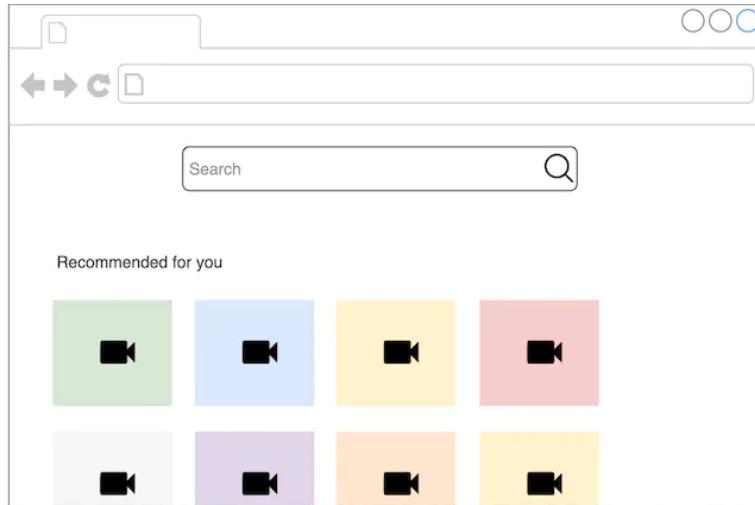


图 6.1: 主页视频推荐

推荐系统的设计通常非常复杂，开发一个高效且可扩展的系统需要投入大量的工程工作量。不过别担心，没有人期望你在 45 分钟的面试中构建出完美的系统。面试官主要希望观察你的思维过程、沟通技巧、设计机器学习系统的能力，以及讨论权衡取舍的能力。

6.1 明确需求

以下是候选人和面试官之间的典型互动。

候选人：我可以假设构建视频推荐系统的业务目标是提高用户参与度吗？

面试官：没错。

候选人：系统是推荐用户正在观看的视频的类似视频吗？还是在用户加载主页时向他们显示个性化的视频列表？

面试官：这是一个主页视频推荐系统，它在用户加载主页时向用户推荐个性化视频。

候选人：既然 YouTube 是一个全球服务，我可以假设用户遍布全球，视频有不同的语言吗？

面试官：这是一个合理的假设。

候选人：我可以假设我们可以根据用户与视频内容的互动来构建数据集吗？

面试官：是的，这样听起来不错。

候选人：用户是否可以通过创建播放列表将视频分组？播放列表在学习阶段对机器学习模型来说是很有用的信息。

面试官：为了简单起见，我们假设不存在播放列表功能。

候选人：平台上有多少个视频？

面试官：我们大约有 100 亿个视频。

候选人：系统应该多快向用户推荐视频？我可以假设推荐时间不超过 200 毫秒吗？

面试官：听起来很合理。

让我们总结一下问题陈述。我们被要求设计一个主页视频推荐系统。业务目标是提高用户参与度。每当用户加载主页时，系统推荐最具吸引力的视频。用户遍布全球，视频可能有不同的语言。平台上大约有 100 亿个视频，并且推荐应该快速提供。

6.2 将问题框定为机器学习任务

6.2.1 定义机器学习目标

系统的业务目标是提高用户参与度。有几种方法可以将业务目标转化为明确的机器学习目标。我们将检查其中一些并讨论它们的权衡取舍。

最大化用户点击数。视频推荐系统可以设计为最大化用户点击数。然而，该目标有一个主要缺点。模型可能会推荐所谓的“标题党”视频，即标题和缩略图看起来很吸引人，但视频内容可能很无聊、不相关甚至具有误导性。标题党视频会随着时间的推移降低用户的满意度和参与度。

最大化完成视频的数量。系统还可以推荐用户可能会完整观看的视频。该目标的一个主要问题是，模型可能会推荐用户更容易看完的短视频。

最大化观看总时长。该目标会产生用户花更多时间观看的推荐视频。

最大化相关视频的数量。该目标会产生与用户相关的视频推荐。工程师或产品经理可以根据一些规则定义相关性。这些规则可以基于隐式和显式用户反馈。例如，一个定义可以规定，如果用户明确按下“喜欢”按钮或观看了至少一半的视频，则该视频是相关的。一旦我们定义了相关性，就可以构建数据集并训练模型以预测用户与视频之间的相关性分数。

在这个系统中，我们选择最终的目标作为机器学习目标，因为我们可以更好地控制使用哪些信号。此外，它没有前述其他选项的缺点。

6.2.2 指定系统的输入和输出

如图 6.2 所示，视频推荐系统以用户为输入，并输出按相关性分数排序的视频列表。

6.2.3 选择合适的机器学习类别

在本节中，我们将研究三种常见的个性化推荐系统类型。

- 基于内容的过滤
- 协同过滤
- 混合过滤

让我们更详细地研究每种类型。

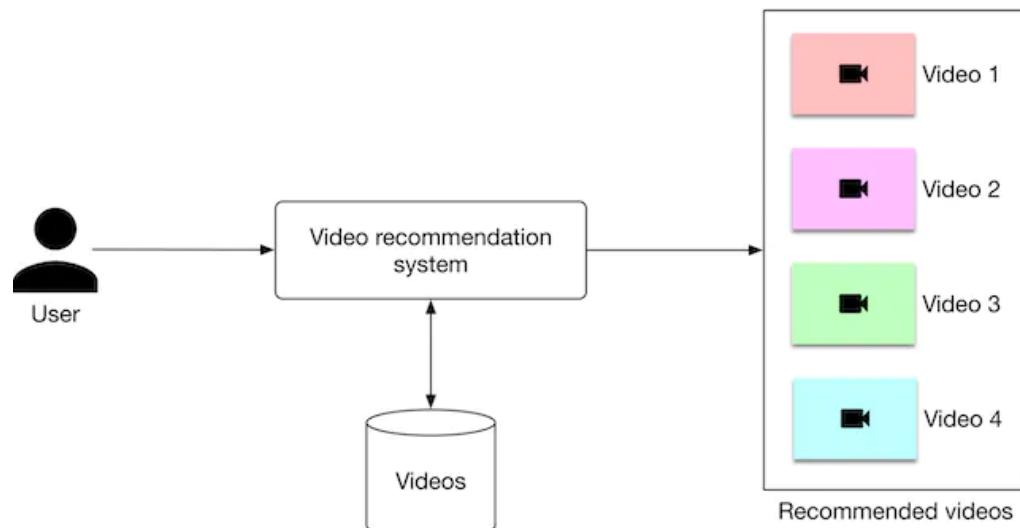


图 6.2: 视频推荐系统的输入-输出

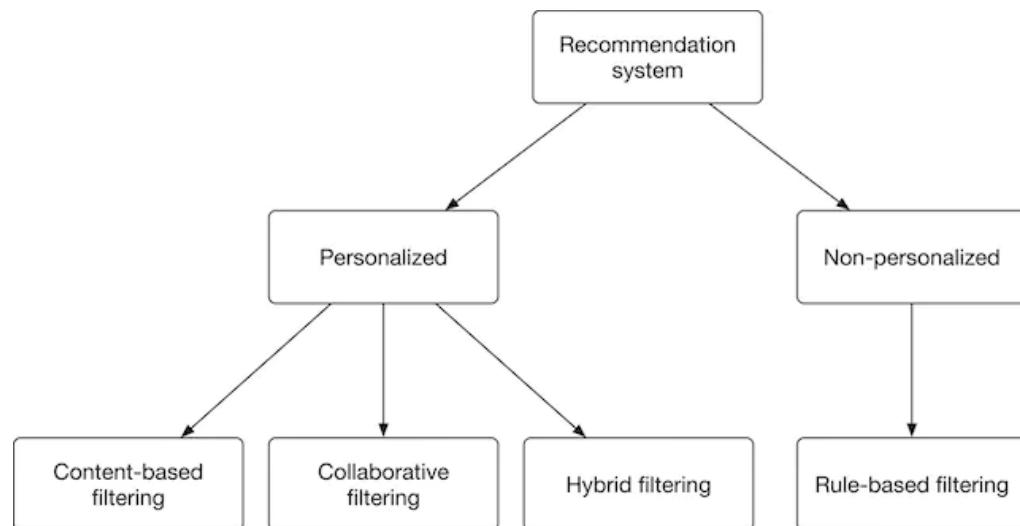


图 6.3: 常见的推荐系统类型

6.2.3.1 基于内容的过滤

该技术使用视频特征来推荐新视频——这些新视频相似于用户觉得相关的视频。例如，如果用户过去与许多滑雪视频有互动，那么此方法将推荐更多滑雪视频。图 6.4 显示了一个例子。

以下是图表的解释。

1. 用户 A 过去与视频 X 和 Y 有互动。
2. 视频 Z 与视频 X 和视频 Y 相似。
3. 系统向用户 A 推荐视频 Z。

基于内容的过滤有其优缺点。

优点：

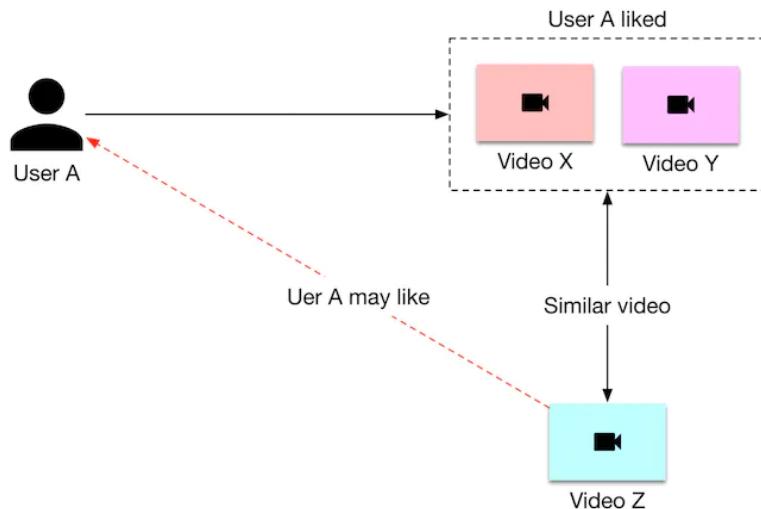


图 6.4: 基于内容的过滤

- 能够推荐新视频。使用此方法，我们无需等待用户的交互数据即可为新视频构建视频档案。视频档案完全依赖于其特征。
- 能够捕捉用户的独特兴趣。这是因为我们根据用户过去的互动推荐视频。

缺点:

- 难以发现用户的新兴趣。
- 该方法需要领域知识。我们通常需要手动设计视频特征。

6.2.3.2 协同过滤 (COLLABORATIVE FILTERING, CF)

协同过滤使用用户-用户相似性（基于用户的 CF）或视频-视频相似性（基于物品的 CF）来推荐新视频。CF 的直观想法是相似的用户对相似的视频感兴趣。你可以在图 6.5 中看到一个基于用户的 CF 示例。

让我们解释一下图表。目标是向用户 A 推荐一个新视频。

1. 根据用户 A 之前的互动找到一个与之相似的用户，比如用户 B。
2. 找到用户 B 观看过但用户 A 尚未观看的视频，比如视频 Z。
3. 向用户 A 推荐视频 Z。

基于内容的过滤和协同过滤的一个主要区别在于，协同过滤不使用视频特征，而是完全依赖于用户的历史互动来进行推荐。让我们看看 CF 的优缺点。

优点:

- 不需要领域知识。CF 不依赖于视频特征，这意味着无需领域知识来从视频中提取特征。

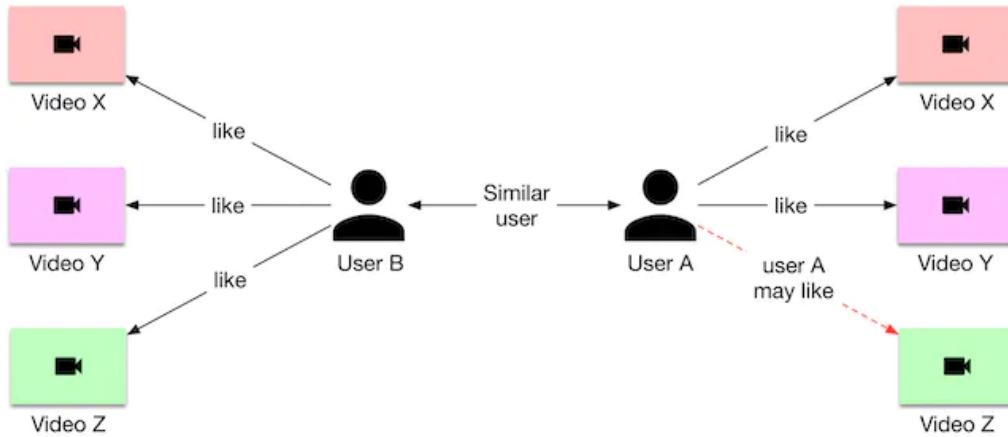


图 6.5: 基于用户的协同过滤

- 容易发现用户的新兴趣领域。系统可以推荐其他相似用户过去与之互动的新主题视频。
- 高效。基于 CF 的模型通常比基于内容的过滤更快且计算量更少，因为它们不依赖于视频特征。

缺点：

- 冷启动问题。这指的是在新视频或用户的数据有限时，系统无法进行准确推荐的情况。由于新用户或视频缺乏历史互动数据，CF 存在冷启动问题。这种互动的缺乏使 CF 无法找到相似的用户或视频。我们将在部署部分稍后讨论系统如何处理冷启动问题。
- 无法处理小众兴趣。CF 难以处理具有特殊或小众兴趣的用户。CF 依赖于相似用户进行推荐，而找到具有小众兴趣的相似用户可能会很困难。

	基于内容的过滤	协同过滤
能处理新视频	✓	✗
发现新兴趣领域	✗	✓
不需要领域知识	✗	✓
效率	✗	✓

表 6.1: 基于内容的过滤与 CF 的比较

表 6.1 显示了两种过滤类型的比较。正如你所看到的，这两种方法是互补的。

6.2.3.3 混合过滤 HYBRID FILTERING

混合过滤同时使用 CF 和基于内容的过滤。正如图 6.6 所示，混合过滤将基于 CF 的推荐器和基于内容的推荐器按顺序或并行组合。在实践中，公司通常使用顺序混合过滤 2。

这种方法会带来更好的推荐，因为它使用了两种数据源：用户的历史互动和视频特征。视频特征使系统能够基于用户过去互动的视频推荐相关视频，而基于 CF 的过滤帮助用户发现新兴趣领域。

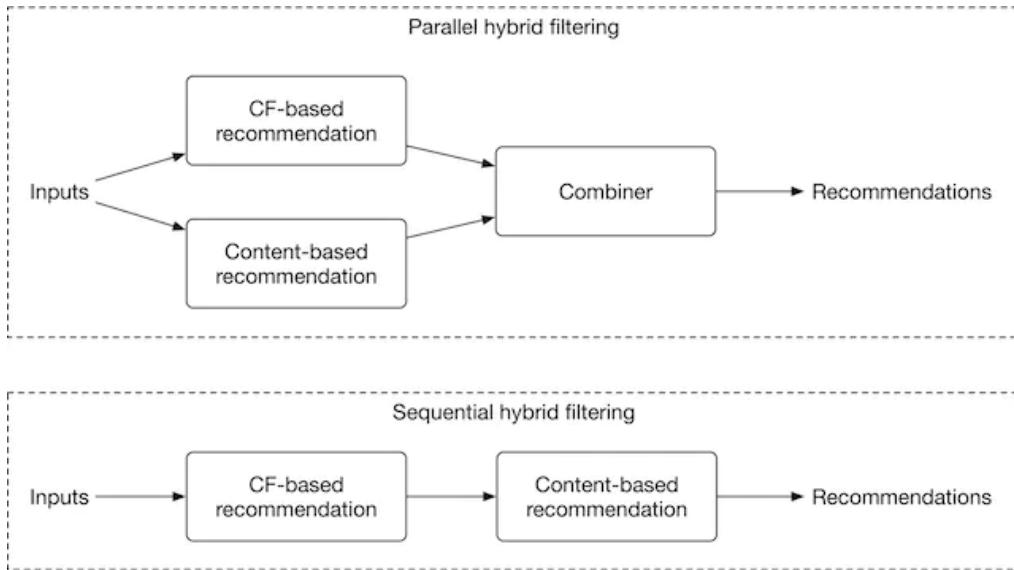


图 6.6: 混合过滤方法

6.2.3.4 我们应该选择哪种方法?

许多公司使用混合过滤来提供更好的推荐。例如，Google 发布的一篇论文²描述了 YouTube 如何使用 CF 模型作为第一阶段（候选生成器），然后使用基于内容的模型作为第二阶段来推荐视频。由于混合过滤的优势，我们选择此选项。

6.3 数据准备

6.3.1 数据工程

我们拥有以下可用数据：

- 视频
- 用户
- 用户-视频互动

视频 视频数据包含原始视频文件及其相关的元数据，例如视频 ID、视频长度、视频标题等。这些属性有些是由视频上传者显式提供的，有些则可以由系统隐式确定，例如视频长度。

Video ID	Length	Manual tags	Manual title	Likes	Views	Language
1	28	Dog, Family	Our lovely dog playing!	138	5300	English
2	300	Car, Oil	How to change your car oil?	5	250	Spanish
3	3600	Ouli, Vlog	Ooneymoon to Bali	2200	255K	Arabic

表 6.2: 视频元数据

ID	Username	Age	Gender	City	Country	Language	Time zone
----	----------	-----	--------	------	---------	----------	-----------

表 6.3: 用户数据模式

用户 以下简单的模式表示用户数据。

用户-视频互动 用户-视频互动数据主要包括用户与视频的各种互动，包括点赞、点击、展示和过去的搜索。互动记录会包括其他上下文信息，如位置和时间戳。下表展示了用户-视频互动的存储方式。

User ID	Video ID	Interaction type	Interaction value	Location (lat, long)	Timestamp
4	18	Like	—	38.8951, -77.0364	1658451361
2	18	Impression	8 seconds	38.8951, -77.0364	1658451841
2	6	Watch	46 minutes	41.9241, -89.0389	1658822820
6	9	Click	—	22.7531, 47.9642	1658832118
9	—	Search	Basics of clustering	22.7531, 47.9642	1659259402
8	6	Comment	Amazing video. Thanks	37.5189, 122.6405	1659244197

表 6.4: 用户-视频互动数据

6.3.2 特征工程

机器学习系统需要预测与用户相关的视频。让我们来设计特征，以帮助系统做出有依据的预测。

视频特征 一些重要的视频特征包括：

- 视频 ID
- 时长
- 语言
- 标题和标签

视频 ID ID 是分类数据。为了用数值向量表示它们，我们使用一个嵌入层，并在模型训练期间学习嵌入层。

时长 时长定义了视频从开始到结束大约持续多长时间。此信息很重要，因为有些用户可能更喜欢短视频，而其他用户可能更喜欢长视频。

语言 视频使用的语言是一个重要的特征。这是因为用户通常更喜欢特定的语言。由于语言是分类变量，并且具有一组有限的离散值，我们使用嵌入层来表示它。

标题和标签 标题和标签用于描述视频。它们可以由上传者手动提供，也可以由独立的机器学习模型隐式预测。视频的标题和标签是有价值的预测因子。例如，一个标题为“如何制作披萨”的视频表明该视频与披萨和烹饪相关。

如何准备它？对于标签，我们使用一个轻量级的预训练模型，例如 CBOW 3，将它们映射为特征向量。

对于标题，我们使用上下文感知的词嵌入模型（如预训练的 BERT 4）将其映射为特征向量。

图 6.7 显示了视频特征准备的概述。

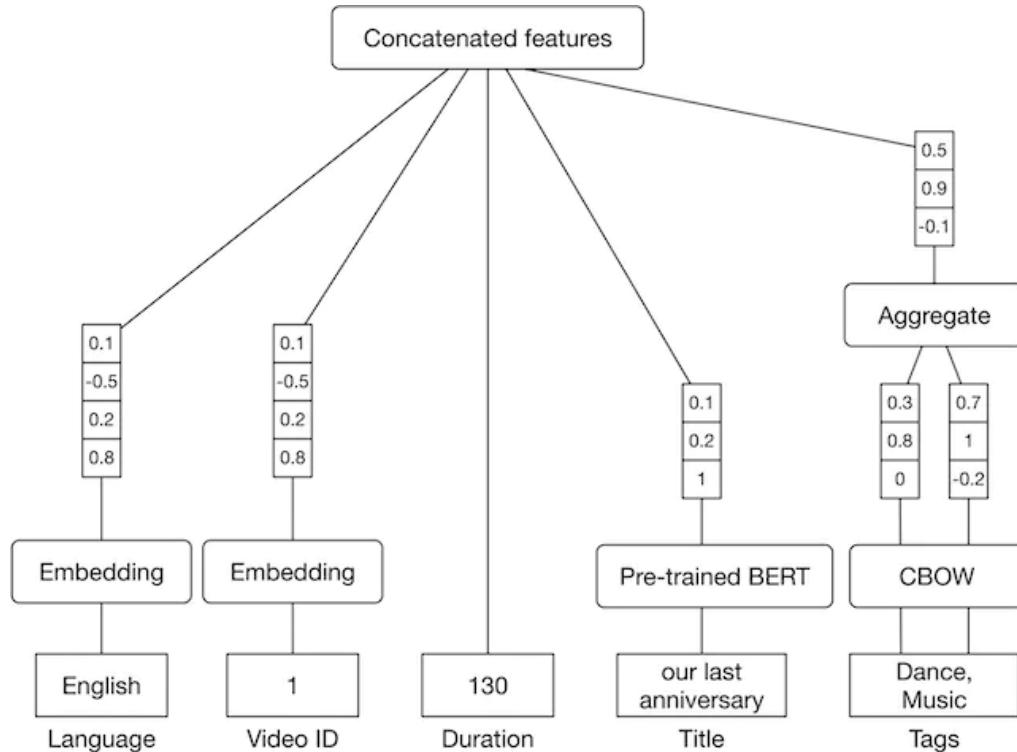


图 6.7: 视频特征准备

用户特征 我们将用户特征分为以下几类：

- 用户人口统计特征
- 上下文信息
- 用户历史互动

用户人口统计特征 图 6.8 显示了用户人口统计特征的概述。

上下文信息 以下是一些重要的上下文信息特征：

- **一天中的时间**。用户可能在一天中的不同时间观看不同的视频。例如，一名软件工程师可能在晚上观看更多的教育视频。
- **设备**。在移动设备上，用户可能更喜欢较短的视频。
- **一周中的某天**。根据一周中的不同天数，用户对视频的偏好可能会有所不同。

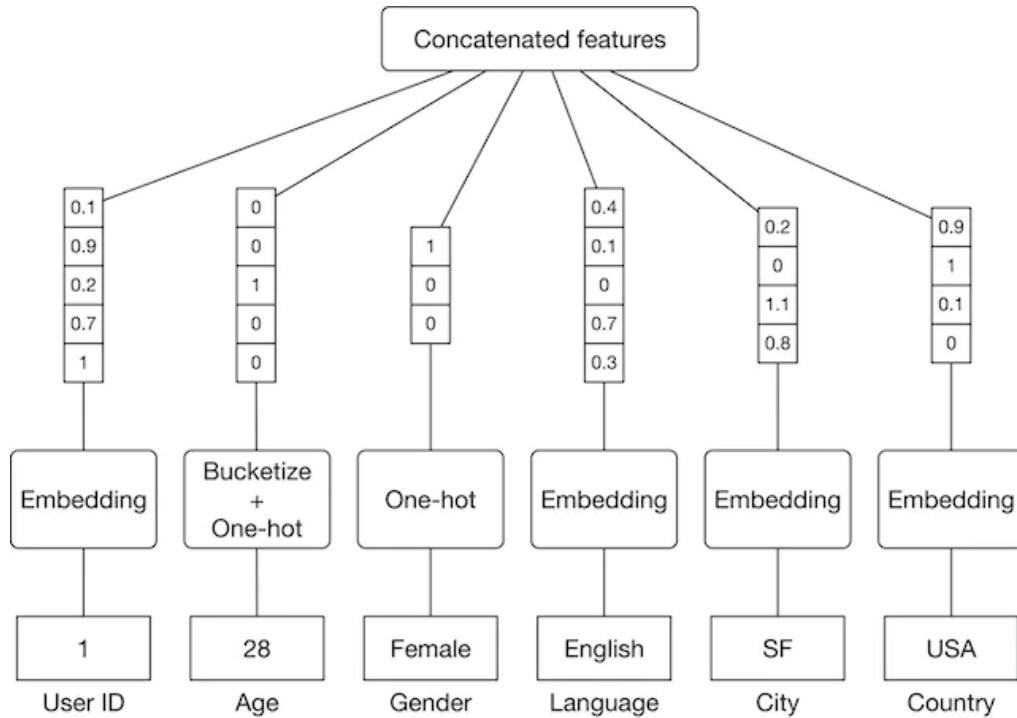


图 6.8: 基于用户人口统计的特征

用户历史互动 用户历史互动在理解用户兴趣方面起着重要作用。以下是一些与历史互动相关的特征：

- 搜索历史
- 点赞的视频
- 已观看的视频和展示

1. 搜索历史

为什么它很重要？ 先前的搜索表明用户过去在寻找什么，而过去的行为往往是未来行为的一个指标。

如何准备它？ 使用预训练的词嵌入模型（如 BERT）将每个搜索查询映射为嵌入向量。请注意，用户的搜索历史是一个可变大小的文本查询列表。为了创建一个固定大小的特征向量来总结所有搜索查询，我们对查询嵌入进行平均。

2. 点赞的视频

为什么它很重要？ 用户之前点赞的视频有助于确定他们感兴趣的内容类型。

如何准备它？ 视频 ID 使用嵌入层映射为嵌入向量。与搜索历史类似，我们对点赞的嵌入进行平均，得到一个固定大小的点赞视频向量。

3. 已观看的视频和展示

“已观看的视频”和“展示”的特征工程过程与我们对点赞视频所做的非常相似。所以我们不会重复它。

图 6.10 总结了与用户-视频互动相关的特征。

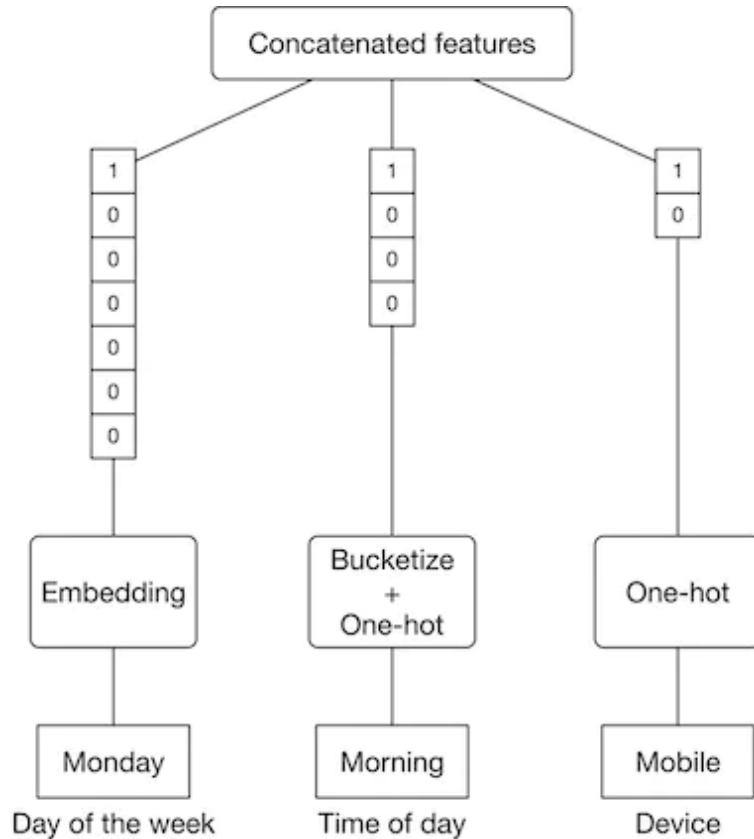


图 6.9: 与上下文信息相关的特征

6.4 模型开发

在本节中，我们将研究两种通常用于基于协同过滤（CF）或基于内容推荐器的嵌入模型：

- 矩阵分解
- 双塔神经网络

6.4.1 矩阵分解

为了理解矩阵分解模型，了解什么是反馈矩阵很重要。

6.4.1.1 反馈矩阵

反馈矩阵也称为效用矩阵，是表示用户对视频的意见的矩阵。图 6.11 显示了一个二值用户-视频反馈矩阵，其中每一行代表一个用户，每一列代表一个视频。矩阵中的条目将用户的 opinion 指定为 1，表示“已观察”或“正面”。

我们如何确定用户是否认为推荐的视频相关？我们有三种选择：

- 显式反馈
- 隐式反馈

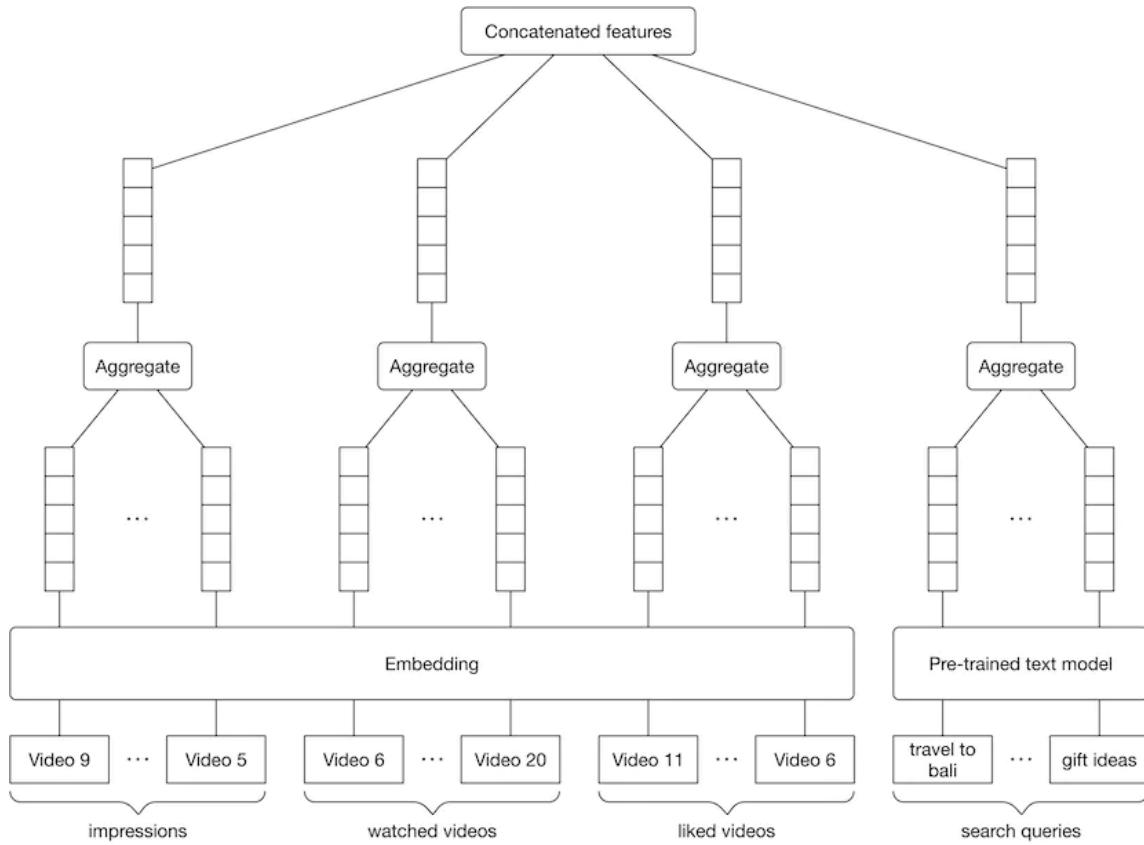


图 6.10: 与用户-视频互动相关的特征

	Video 1	Video 2	Video 3	Video 4	Video 5
User 1	1	1			
User 2			1	1	
User 3		1			1

图 6.11: 用户-视频反馈矩阵

- 显式和隐式反馈的组合

显式反馈。反馈矩阵是基于明确表明用户对视频看法的互动构建的，例如点赞和分享。显式反馈能够准确反映用户的意愿，因为用户明确表达了他们对视频的兴趣。然而，这种选择有一个主要缺点：矩阵稀疏，因为只有少数用户提供显式反馈。稀疏性使得机器学习模型难以训练。

隐式反馈。该选项使用隐式表明用户对视频看法的互动，例如“点击”或“观看时间”。使用隐式反馈，可以获得更多的数据点，从而在训练后获得更好的模型。其主要缺点是它不能直接反映用户的意愿，可能会有噪声。

显式和隐式反馈的组合。此选项使用启发式方法结合显式和隐式反馈。

构建反馈矩阵的最佳选择是什么？由于模型需要学习反馈矩阵的值，因此构建与我们之前选择的机器学习目标对齐的矩阵很重要。在我们的案例中，机器学习目标是最大化相关性，其中相关性定义为显式和隐式反馈的组合。因此，结合显式和隐式反馈的最终选项是最佳选择。

矩阵分解模型 矩阵分解是一种简单的嵌入模型。该算法将用户-视频反馈矩阵分解为两个低维矩阵的乘积。一个低维矩阵代表用户嵌入，另一个代表视频嵌入。换句话说，模型学习将每个用户映射为一个嵌入向量，将每个视频映射为一个嵌入向量，使得它们之间的距离表示它们的相关性。图 6.12 显示了反馈矩阵如何分解为用户和视频嵌入。

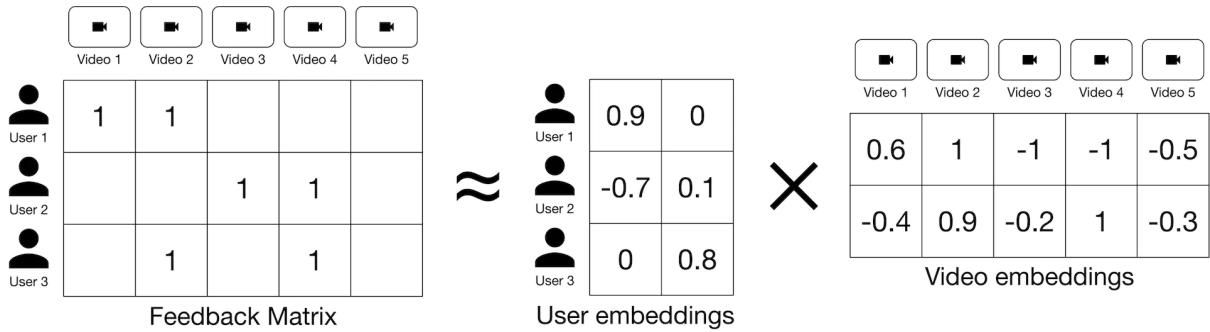


图 6.12: 将反馈矩阵分解为两个矩阵

6.4.1.2 矩阵分解训练

在训练过程中，我们的目标是生成用户和视频嵌入矩阵，使它们的乘积能够很好地近似反馈矩阵（图 6.13）。

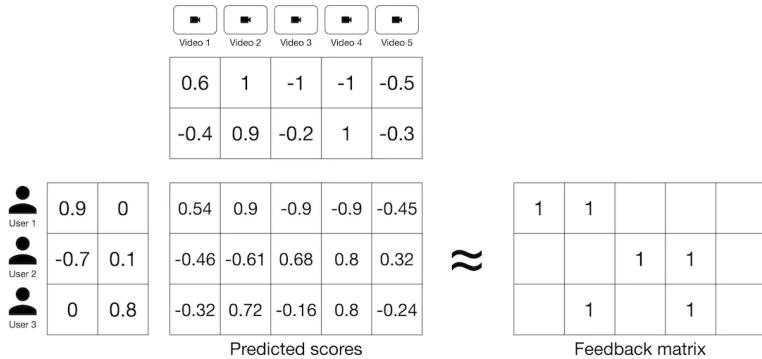


图 6.13: 嵌入的乘积应近似反馈矩阵

为了学习这些嵌入，矩阵分解首先随机初始化两个嵌入矩阵，然后逐步优化这些嵌入，以减少“预测分数矩阵”和“反馈矩阵”之间的损失。损失函数的选择是一个重要的考虑因素。让我们探索一些选项：

- 在观测到的〈用户，视频〉对上的平方距离
- 观测到的对和未观测到的对上的平方距离的加权组合

在观测到的〈用户，视频〉对上的平方距离。该损失函数测量反馈矩阵中所有观测对（非零值）条目上的平方距离之和，如图 6.14 所示。

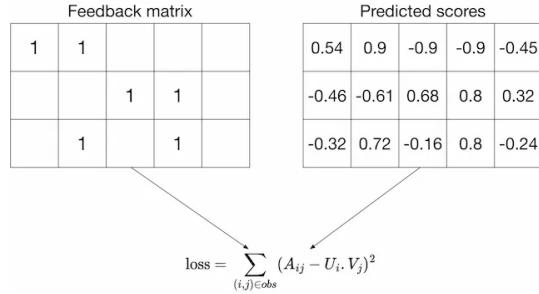


图 6.14: 在观测到的〈用户，视频〉对上的平方距离

仅在观测对上求和会导致较差的嵌入，因为损失函数不会对未观测对的错误预测进行惩罚。例如，所有值为 1 的嵌入矩阵在训练数据上可能有零损失，但这些嵌入可能对未见的〈用户，视频〉对表现不佳。该损失函数将未观测对视为负数据点，并在反馈矩阵中对其进行分配零。如图 6.15 所示，损失计算反馈矩阵中所有条目的平方距离之和。该损失函数通过对未观测条目的错误预测进行惩罚解决了之前的问题。然

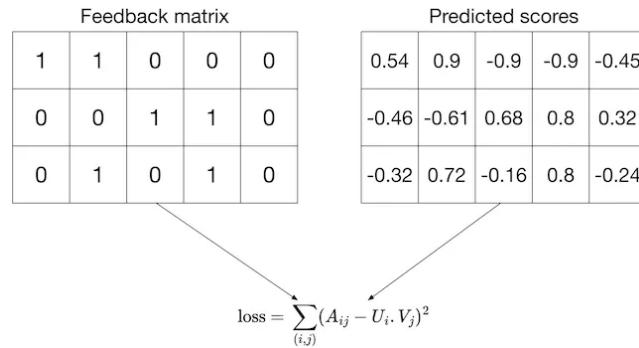


图 6.15: 在所有〈用户，视频〉对上的平方距离

而，该损失有一个主要缺点。反馈矩阵通常是稀疏的（许多未观测对），因此在训练过程中，未观测对占据了主导地位。这导致预测结果大多接近于零。这是不理想的，并导致在未见的〈用户，视频〉对上的泛化性能较差。

观测对和未观测对上平方距离的加权组合。为了克服前述损失函数的缺点，我们选择了两者的加权组合。损失公式中的第一个求和计算观测对的损失，第二个求和计算未观测对的损失。 W 是一个超参数，用于在训练阶段对两个求和项进行权重调整。这个具有适当调整的 W 的损失函数在实践中表现良好 5。我们为系统选择了这个损失函数。

6.4.1.3 矩阵分解优化

为了训练机器学习模型，需要一个优化算法。矩阵分解中常用的两个优化算法是：

- 随机梯度下降 (SGD)：这是一个用于最小化损失的优化算法 6。
- 加权交替最小二乘 (WALS)：这是一个专门用于矩阵分解的优化算法。WALS 的过程如下：

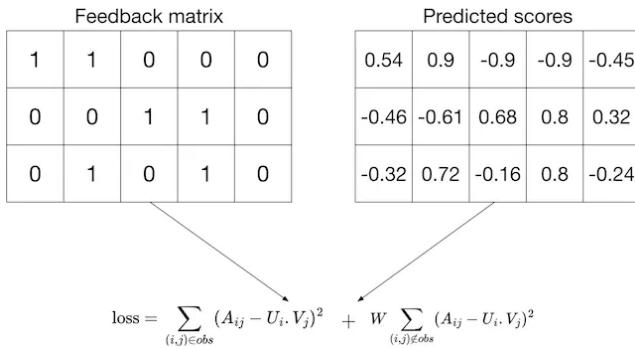


图 6.16: 组合损失

1. 固定一个嵌入矩阵 (U)，并优化另一个嵌入 (V)
2. 固定另一个嵌入矩阵 (V)，并优化嵌入矩阵 (U)
3. 重复。WALS 通常收敛得更快并且可并行化。要了解更多关于 WALS 的信息，请阅读 7。在这里，我们使用 WALS，因为它收敛得更快。

6.4.1.4 矩阵分解推理

为了预测任意用户与候选视频之间的相关性，我们使用相似性度量（如点积）来计算它们嵌入之间的相似性。例如，如图 6.17 所示，用户 2 和视频 5 之间的相关性得分为 0.32。

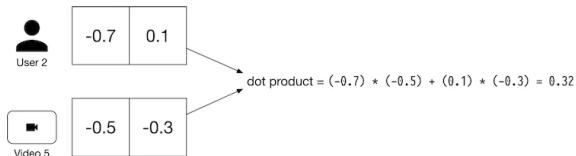


图 6.17: 用户 2，视频 5 对的相关性得分

图 6.18 显示了所有〈用户，视频〉对的预测得分。系统根据相关性得分返回推荐视频。

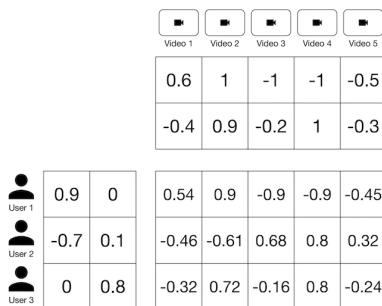


图 6.18: 预测的两两相关性得分

提示：由于矩阵分解仅使用用户-视频互动数据，它通常用于协同过滤。

在总结矩阵分解之前，让我们讨论一下这种模型的优缺点。

优点:

- **训练速度:** 矩阵分解在训练阶段非常高效。这是因为只需要学习两个嵌入矩阵。
- **推理速度:** 矩阵分解在服务时非常快。学习到的嵌入是静态的，这意味着一旦我们学会了它们，我们就可以在查询时重复使用它们，而无需转换输入。

缺点:

- **依赖用户-视频互动数据:** 矩阵分解仅依赖于用户-视频互动数据。它不使用其他特征，例如用户的年龄或语言。这限制了模型的预测能力，因为诸如语言等特征对于提高推荐质量很有帮助。
- **处理新用户很困难:** 对于新用户，模型缺乏足够的互动来生成有意义的嵌入。因此，矩阵分解无法通过计算嵌入之间的点积来确定视频是否与用户相关。

让我们看看双塔神经网络如何解决矩阵分解的缺点。

6.4.2 双塔神经网络

双塔神经网络由两个编码器塔组成：用户塔和视频塔。用户编码器将用户特征作为输入并将其映射到嵌入向量（用户嵌入）。视频编码器将视频特征作为输入并将其映射为嵌入向量（视频嵌入）。它们在共享嵌入空间中的距离表示它们的相关性。

图 6.19 显示了双塔架构。与矩阵分解相比，双塔架构足够灵活，可以结合各种特征，以更好地捕捉用户的特定兴趣。

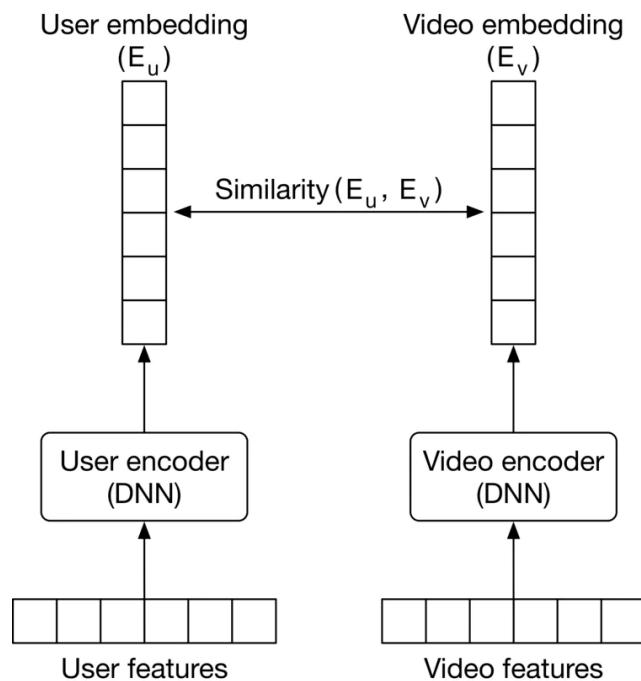


图 6.19: 双塔神经网络

构建数据集 我们通过从不同的〈用户，视频〉对中提取特征并根据用户的反馈对它们进行正负标签构建数据集。例如，如果用户明确喜欢视频或至少观看了一半，我们将对标记为“正面”。

为了构建负面数据点，我们可以选择随机视频（这些视频与用户不相关），或者选择那些用户通过点击“不喜欢”按钮明确表示不喜欢的视频。图 6.20 显示了构建的数据点示例。

#	User-related features	Video-related features	Label
1	0 0 1 0.7 -0.6 0 0	0 1 0 0.9 0.9 1	1 (positive)
2	0 1 1 0.2 0.1 1 0	0 1 0 -0.1 0.3 1	0 (negative)

图 6.20: 构建的两个数据点

请注意，用户通常只会认为少数视频是相关的。在构建训练数据时，这导致数据集不平衡，即负样本对远多于正样本对。在不平衡的数据集上训练模型是有问题的。我们可以使用第 1 章引言和概述中描述的技术来解决数据不平衡问题。

选择损失函数 由于双塔神经网络被训练用于预测二值标签，该问题可以归类为分类任务。我们使用典型的分类损失函数，例如交叉熵，以在训练期间优化编码器。这一过程如图 6.21 所示。

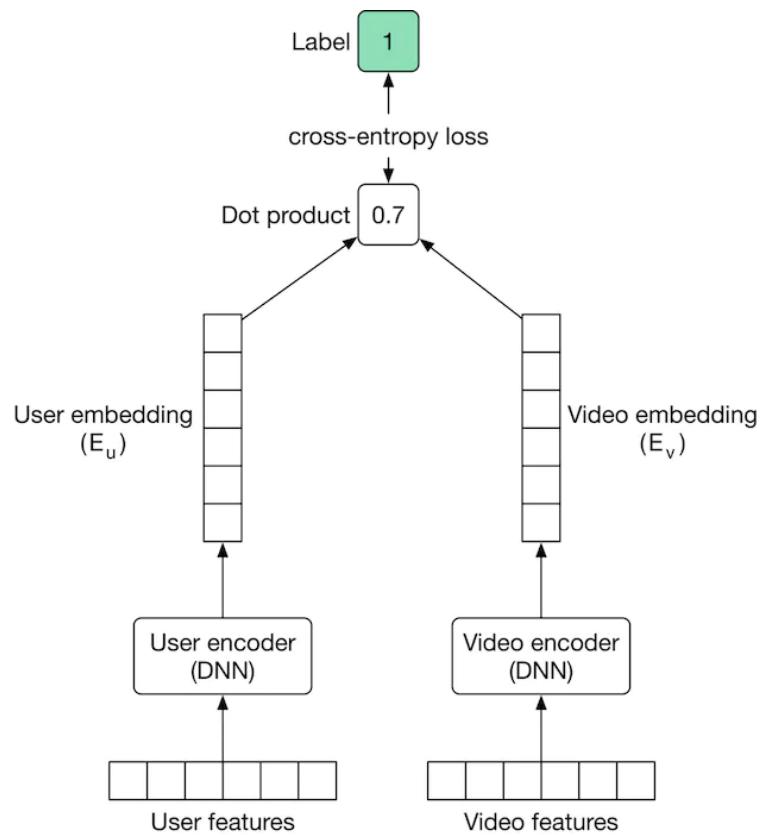


图 6.21: 双塔神经网络训练工作流程

双塔神经网络推理 在推理时，系统使用嵌入来为给定用户找到最相关的视频。这是一个经典的“最近邻”问题。我们使用近似最近邻方法来有效地找到最相似的视频嵌入。

双塔神经网络用于基于内容的过滤和协同过滤。当双塔架构用于协同过滤时，如图 6.22 所示，视频编码器仅仅是一个将视频 ID 转换为嵌入向量的嵌入层。这样，模型不依赖于其他视频特征。

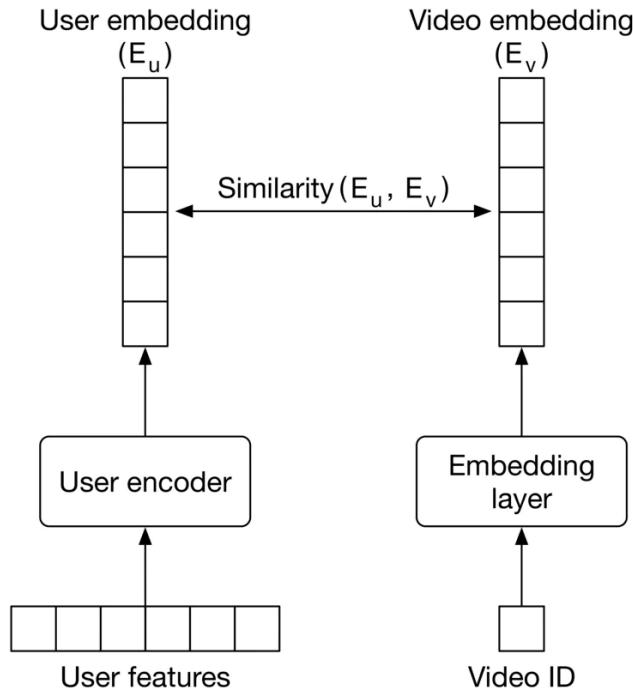


图 6.22: 用于协同过滤的双塔神经网络

让我们看看双塔神经网络模型的优缺点。

优点:

- **利用用户特征。**该模型接受用户特征（如年龄和性别）作为输入。这些预测性特征有助于模型做出更好的推荐。
- **处理新用户。**该模型依赖于用户特征（如年龄、性别等），因此可以轻松处理新用户。

缺点:

- **服务速度较慢。**模型需要在查询时计算用户嵌入。这使得模型在服务请求时速度较慢。此外，如果我们将该模型用于基于内容的过滤，模型需要将视频特征转换为视频嵌入，这会增加推理时间。
- **训练成本更高。**双塔神经网络比矩阵分解具有更多的学习参数。因此，训练更加计算密集。

6.4.3 矩阵分解 vs. 双塔神经网络

表 6.5 总结了矩阵分解与双塔神经网络架构之间的差异。

	矩阵分解	双塔神经网络
训练成本	✓ 更高效	✗ 成本更高
推理速度	✓ 更快，因为嵌入是静态的且可预算	✗ 查询时需要转换用户特征
冷启动问题	✗ 难以处理新用户	✓ 依赖于用户特征，因此能处理新用户
推荐质量	✗ 模型不使用用户/视频特征，效果有限	✓ 使用更多特征，推荐质量更高

表 6.5: 矩阵分解 vs. 双塔神经网络

6.5 评估

系统的性能可以通过离线和在线指标来评估。

6.5.1 离线指标

我们评估推荐系统中常用的以下离线指标。

Precision@k 该指标衡量推荐视频前 k 个视频中相关视频的比例。可以使用多个 k 值（例如 1, 5, 10）。

mAP 该指标衡量推荐视频的排序质量。它非常适合，因为在我们的系统中相关性得分是二进制的。

多样性 该指标衡量推荐视频之间的相似度有多低。该指标非常重要，因为用户对多样化的视频更感兴趣。为了衡量多样性，我们计算列表中视频之间的平均成对相似度（例如，余弦相似度或点积）。较低的平均成对相似度得分表示列表更加多样化。

请注意，仅使用多样性作为质量度量可能会导致误导性的解释。例如，如果推荐的视频很丰富但与用户无关，用户可能不会觉得推荐很有用。因此，我们应结合其他离线指标一起使用多样性，以确保既有相关性又有多样性。

6.5.2 在线指标

在实践中，企业在进行在线评估时会跟踪许多指标。让我们来看看其中一些最重要的指标：

- 点击率 (CTR)
- 完整观看的视频数量
- 总观看时间
- 用户显式反馈

CTR 点击视频的数量与推荐视频总数的比率。公式为：

$$\text{CTR} = \frac{\text{number of clicked videos}}{\text{total number of recommended videos}}$$

CTR 是衡量用户参与度的一个有用指标，但其缺点是无法捕捉或衡量点击诱导视频 (clickbait)。

完整观看的视频数量。用户观看到结尾的视频总数。通过跟踪这一指标，我们可以了解系统推荐的视频有多频繁地被完整观看。

总观看时间。 用户观看推荐视频所花费的总时间。当推荐内容吸引用户时，他们会花更多的时间观看视频。

用户显式反馈。 用户明确表示喜欢或不喜欢的视频总数。该指标准确反映用户对推荐视频的意见。

6.6 服务

在服务阶段，系统通过从数十亿个视频中缩小选择范围，为给定用户推荐最相关的视频。在本节中，我们将提出一个高效且准确的预测 Pipeline，以应对服务请求。

考虑到我们有数十亿个可用视频，如果选择一个需要大量特征作为输入的复杂模型，则服务速度会变慢。另一方面，如果选择轻量级模型，推荐质量可能不高。那么，该怎么办？一个自然的决定是使用多个模型的多阶段设计。例如，在两阶段设计中，轻量级模型在第一阶段快速缩小视频范围，称为候选生成。第二阶段使用一个更复杂的模型精确评分和排序视频，称为评分。图 6.23 显示了候选生成和评分如何协同工作以生成相关视频。

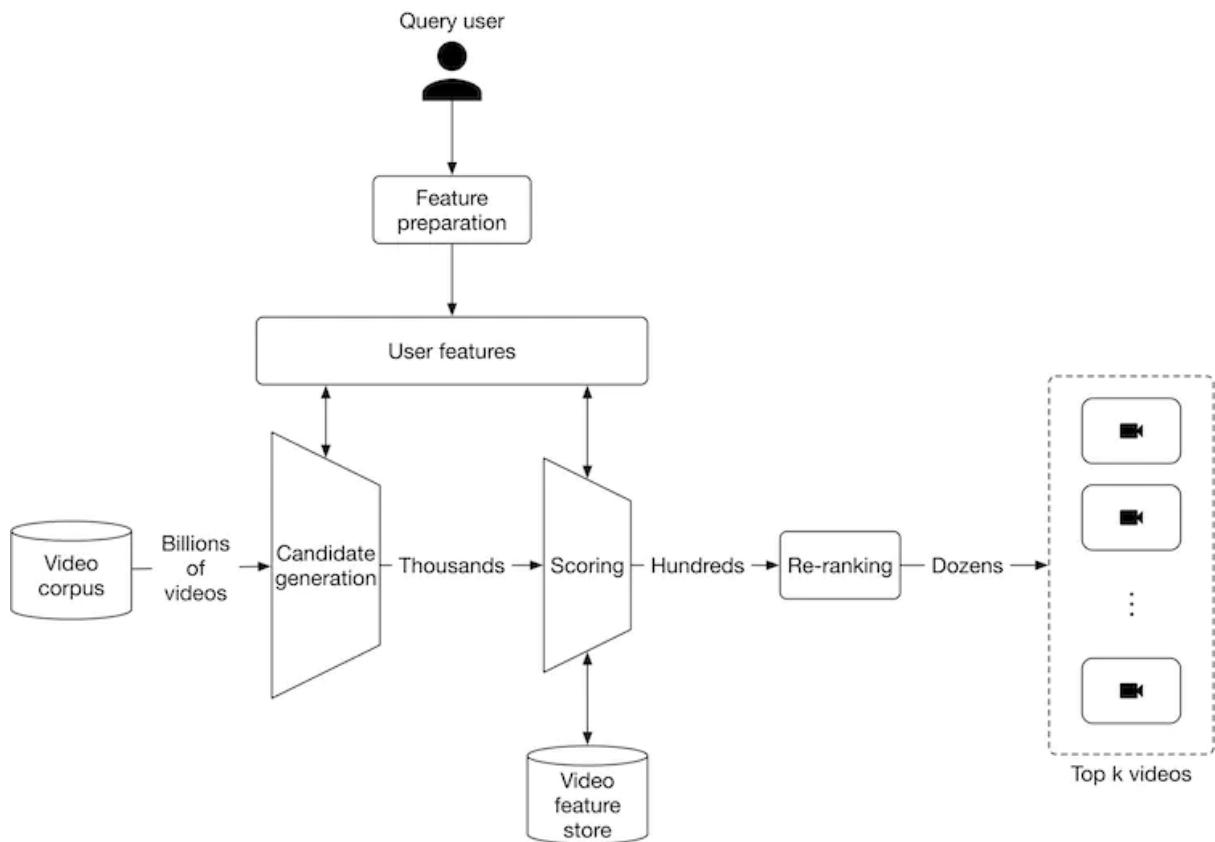


图 6.23: 预测 Pipeline

让我们更详细地了解预测 Pipeline 的各个组件。

- 候选生成
- 评分
- 重排序

6.6.1 候选生成

候选生成的目标是将视频从数十亿缩小到数千个。在这个阶段，我们优先考虑效率而非准确性，因此不必担心误报。

为了保持候选生成的快速性，我们选择不依赖视频特征的模型。此外，该模型应能够处理新用户。双塔神经网络非常适合这一阶段。

图 6.24 显示了候选生成工作流。候选生成从用户编码器中获取用户的嵌入。一旦计算完成，它从近似最近邻服务中检索出最相似的视频。这些视频根据嵌入空间中的相似度进行排序，并作为输出返回。

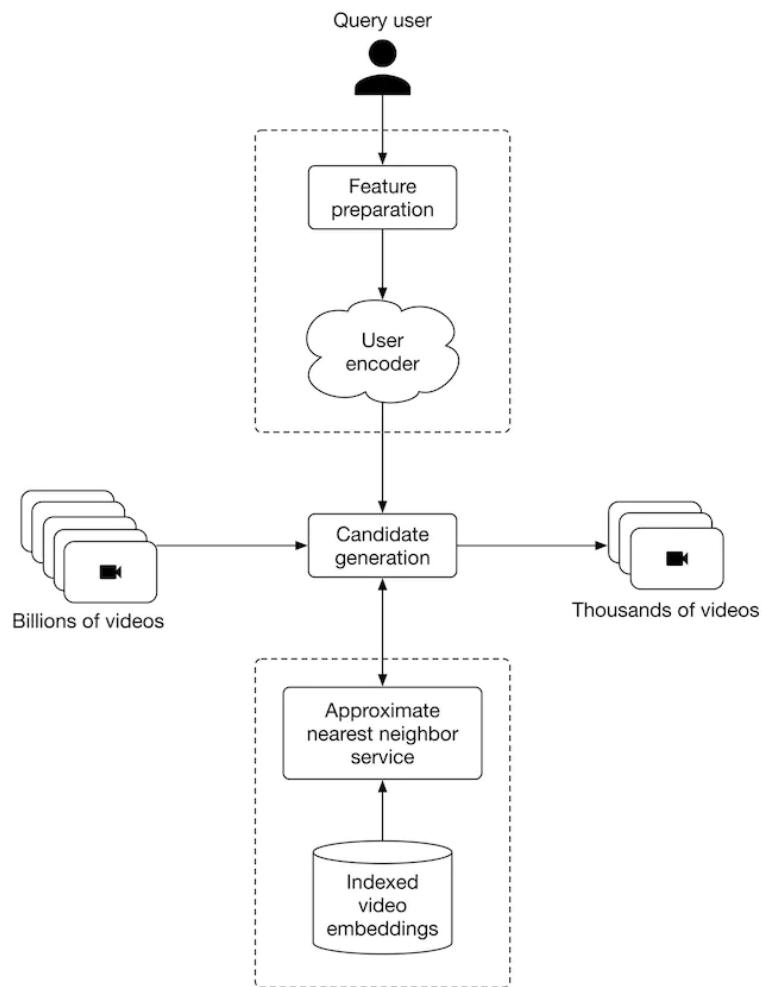
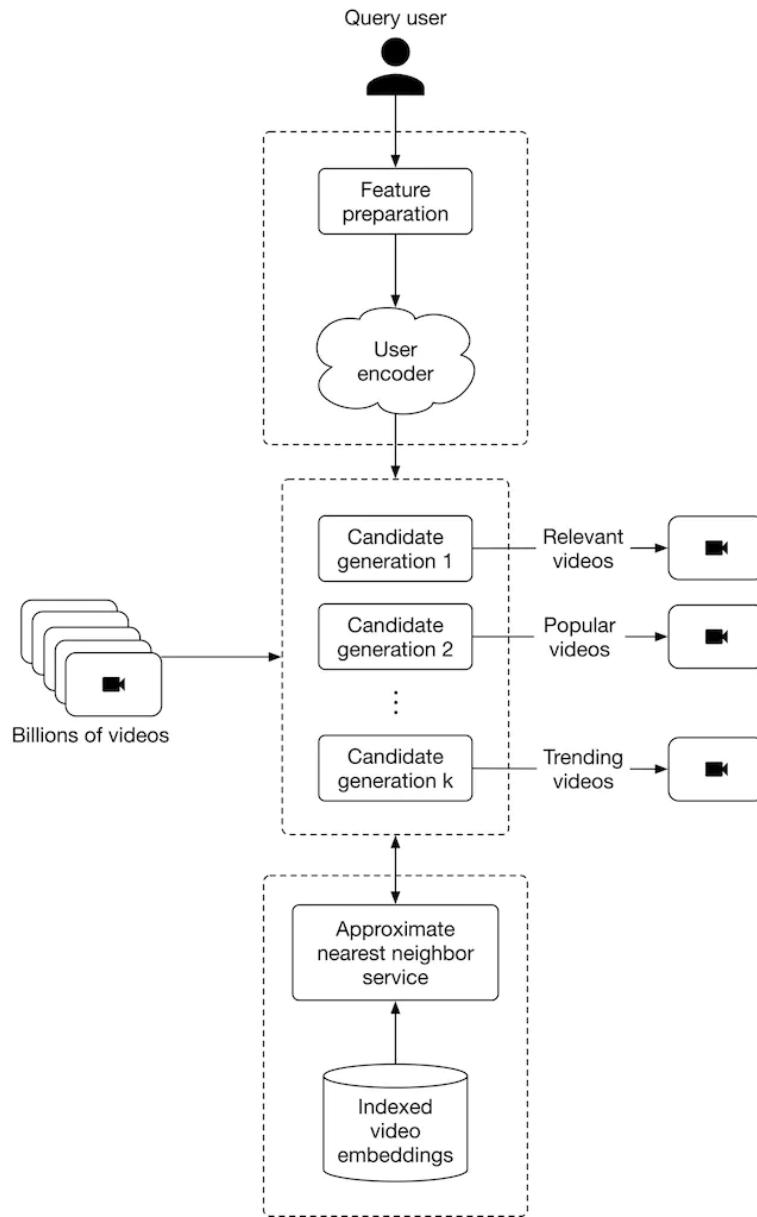


图 6.24: 候选生成工作流

在实际中，企业可能会选择使用多个候选生成，因为这可以提高推荐的性能。让我们看看为什么。

用户可能出于多种原因对视频感兴趣。例如，用户可能会选择观看热门、流行或与他们所在位置相关的视频。为了将这些视频包含在推荐中，通常会使用多个候选生成，如图 6.25 所示。

一旦我们将潜在的视频数量从数十亿缩小到数千个，我们可以使用评分组件在显示之前对这些视频进行排序。

图 6.25: 使用 k 个候选生成来多样化推荐视频

6.6.2 评分 Scoring

评分 Scoring (亦为排序) 将用户和候选视频作为输入，对每个视频进行评分，并输出一个排序后的视频列表。

在这一阶段，我们优先考虑准确性而非效率。为此，我们选择基于内容的过滤方法，并选择一个依赖视频特征的模型。双塔神经网络是这一阶段的常用选择。由于在评分阶段只有少量视频需要排序，我们可以采用具有更多参数的复杂模型。图 6.26 显示了评分组件的概述。

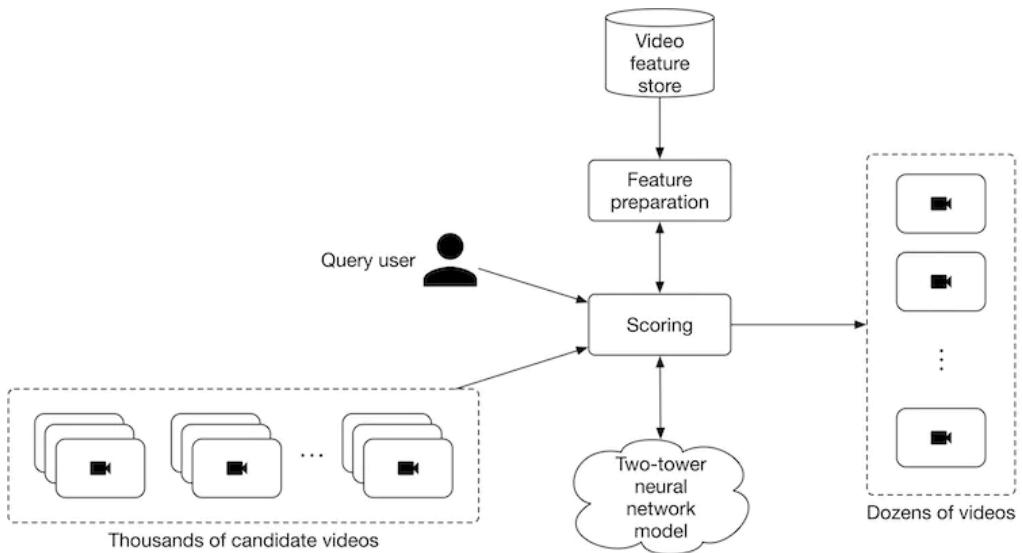


图 6.26: 评分组件概述

6.6.3 重排序 Re-ranking

该组件通过添加附加标准或约束对视频重新排序。例如，我们可以使用独立的机器学习模型来确定视频是否是点击诱导视频。以下是在构建重排序组件时需要考虑的一些重要事项：

- 区域受限的视频
- 视频的新鲜度
- 传播错误信息的视频
- 重复或近似重复的视频
- 公平性和偏见

6.7 视频推荐系统的挑战

在结束本章之前，让我们看看我们的设计如何应对视频推荐系统中的典型挑战。

6.7.1 服务速度

快速推荐视频至关重要。然而，由于系统中有数十亿个视频，因此高效且准确地推荐它们具有挑战性。为了解决这个问题，我们采用了两阶段设计。

具体而言，我们在第一阶段使用轻量级模型快速将候选视频从数十亿缩小到数千个。YouTube 采用了类似的方法²，Instagram 也采用了多阶段设计⁸。

6.7.2 精确度 Precision

为了确保精确度，我们使用评分组件，该组件使用依赖于更多特征（包括视频特征）的强大模型对视频进行排序。使用更强大的模型不会影响服务速度，因为在候选生成阶段之后只选择了少量视频。

6.7.3 多样性 Diversity

大多数用户更喜欢在推荐中看到多样化的视频选择。为了确保我们的系统生成多样化的视频集，我们采用了多个候选生成，如候选生成部分所述。

6.7.4 冷启动问题

我们的系统如何解决冷启动问题？

对于新用户：当新用户开始使用我们的平台时，我们没有任何关于他们的互动数据。在这种情况下，系统会基于诸如年龄、性别、语言、位置等特征使用双塔神经网络进行预测。即使对于新用户，推荐的视频在一定程度上也是个性化的。随着用户与更多视频的互动，我们可以基于新的互动做出更好的预测。

对于新视频：当系统中添加一个新视频时，视频的元数据和内容是可用的，但没有互动数据。处理这一问题的一种方法是使用启发式方法。我们可以将视频展示给随机用户并收集互动数据。一旦我们收集到足够的互动，就可以使用新的互动数据微调双塔神经网络。

6.7.5 训练的可扩展性

在成本效益上训练大规模数据集上的模型具有挑战性。在推荐系统中，不断添加新的互动数据，模型需要快速适应以提供准确的推荐。为了快速适应新数据，模型应该能够进行微调。

在我们的案例中，模型基于神经网络，并设计为易于微调。

6.8 其他谈话要点

如果访谈结束时还有时间，以下是一些额外的谈话要点：

- 推荐系统中的探索-利用权衡 [9](#)。
- 推荐系统中可能存在不同类型的偏见 [10](#)。
- 构建推荐系统时与道德相关的重要考虑因素 [11](#)。
- 在推荐系统中考虑季节性的影响——用户在不同季节的行为变化 [12](#)。
- 针对多个目标而不是单个目标优化系统 [13](#)。
- 如何从不喜欢等负面反馈中获益 [14](#)。
- 利用用户搜索历史或观看历史中的视频序列 [2](#)。

References

1. YouTube recommendation system. <https://blog.youtube/inside-youtube/on-youtubes-recommendation-systems>.
2. DNN for YouTube recommendation. <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45530.pdf>.
3. CBOW paper. <https://arxiv.org/pdf/1301.3781.pdf>.
4. BERT paper. <https://arxiv.org/pdf/1810.04805.pdf>.
5. Matrix factorization. <https://developers.google.com/machine-learning/recommendation/collaborative/matrix>.
6. Stochastic gradient descent. https://en.wikipedia.org/wiki/Stochastic_gradient_descent.
7. WALS optimization. <https://fairyonice.github.io/Learn-about-collaborative-filtering-and-weighted-least-squares.html>.
8. Instagram multi-stage recommendation system. <https://ai.facebook.com/blog/powerd-by-ai-instantgrams>.
9. Exploration and exploitation trade-offs. https://en.wikipedia.org/wiki/Multi-armed_bandit.
10. Bias in AI and recommendation systems. <https://www.searchenginejournal.com/biases-search-recommendations-339319/#close>.
11. Ethical concerns in recommendation systems. <https://link.springer.com/article/10.1007/s00146-020-00950-y>.
12. Seasonality in recommendation systems. <https://www.computer.org/csdl/proceedings-article/big-data/2019/09005954/1hJsfT0qL6>.
13. A multitask ranking system. <https://daiwk.github.io/assets/youtube-multitask.pdf>.
14. Benefit from a negative feedback. <https://arxiv.org/abs/1607.04228?context=cs>.

7 Event Recommendation System 活动推荐系统

在本章中，我们设计了一个类似于 Eventbrite 的活动推荐系统。Eventbrite 是一个流行的活动管理和票务市场，允许用户创建、浏览和注册活动。推荐系统个性化用户体验，显示与用户相关的活动。

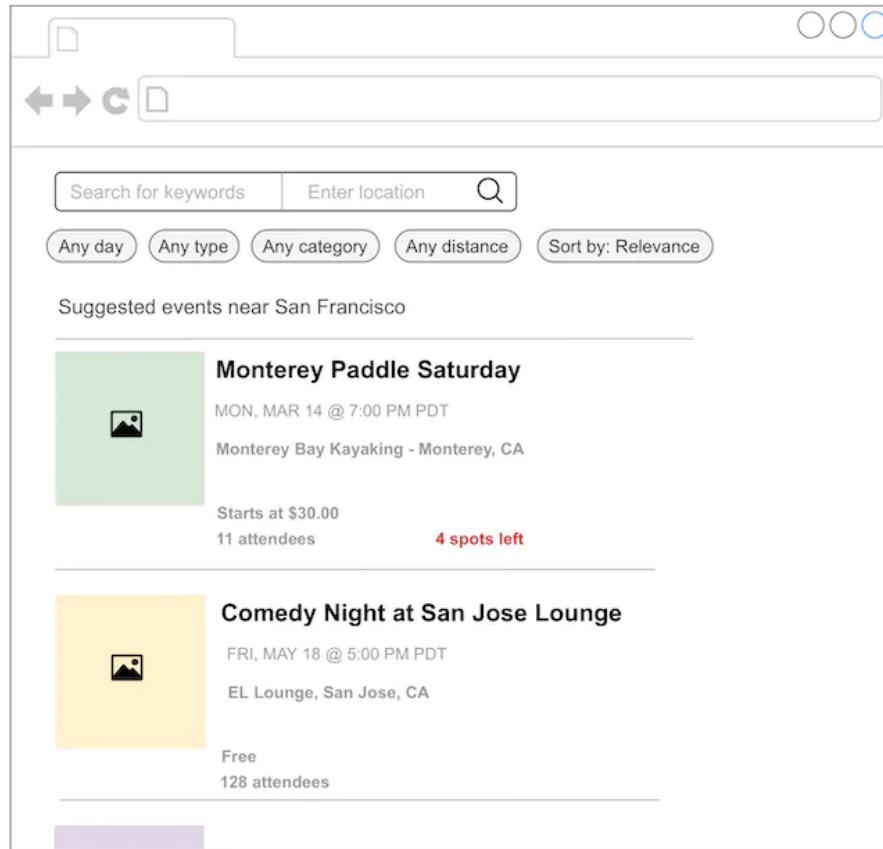


图 7.1: 推荐的活动

7.1 明确需求

以下是候选人与面试官之间的典型对话。

候选人: 商业目标是什么？我可以假设主要的商业目标是增加门票销售吗？

面试官: 是的，这个假设不错。

候选人: 除了参加活动，用户可以在平台上预订酒店或餐馆吗？

面试官: 为了简化问题，我们假设平台仅支持活动。

候选人: 活动被认为是一种短暂的一次性事件，它只发生一次，然后过期。这个假设正确吗？

面试官: 这是个很好的观察。

候选人: 有哪些活动属性？我可以假设我们可以访问活动的文本描述、价格范围、位置、日期和时间等信息吗？

面试官: 当然，这些假设是合理的。

候选人: 我们有标注的数据吗？

面试官：我们没有手动标注的数据集。你可以使用活动和用户互动数据来构建训练数据集。

候选人：我们能否访问用户的当前位置？

面试官：是的。由于这个问题的重点是基于位置的推荐系统，假设用户同意分享他们的位置数据。

候选人：用户可以在平台上互相成为朋友吗？好友信息对构建个性化的活动推荐系统很有价值。

面试官：这是个好问题。是的，假设用户可以在我们的平台上建立友谊关系。友谊关系是双向的，这意味着如果 A 是 B 的朋友，那么 B 也是 A 的朋友。

候选人：用户可以邀请他人参加活动吗？

面试官：是的。

候选人：用户可以 RSVP（回复参加活动）吗？

面试官：为了简化问题，假设活动只有注册选项。

候选人：活动是免费的还是收费的？

面试官：我们需要支持两者。

候选人：平台上有多少用户和活动？

面试官：我们每个月大约有 100 万个活动。

候选人：每天有多少活跃用户访问网站/应用程序？

面试官：假设我们每天有 100 万唯一用户。

候选人：由于我们正在构建基于位置的活动推荐系统，高效地计算两个位置之间的距离和旅行时间非常重要。我们可以假设可以使用外部 API（如 Google Maps API 或其他地图服务）来获取这些数据吗？

面试官：很好的问题。假设我们可以使用第三方服务来获取位置数据。

总结问题陈述。我们被要求设计一个活动推荐系统，向用户显示个性化的活动列表。当活动结束时，用户不能再注册该活动。除了注册活动外，用户还可以邀请他人参加活动并建立友谊关系。训练数据应该从用户交互中在线构建。该系统的主要目标是增加总票销售量。

7.2 将问题框架化为机器学习任务

7.2.1 定义机器学习目标

基于需求，业务目标是增加门票销售量。将其转化为明确的机器学习目标的一种方法是最大化活动注册数量。

7.2.2 指定系统的输入和输出

系统的输入是一个用户，输出是按与用户相关性排序的前 k 个活动。

7.2.3 选择合适的机器学习类别

有多种方法可以解决推荐问题：

- 简单规则，例如推荐热门活动
- 基于嵌入的模型，这些模型依赖于基于内容的或协同过滤的方式
- 将其重新表述为排序问题

基于规则的方法是形成基线的良好起点。然而，基于机器学习的方法通常会带来更好的结果。在本章中，我们将任务重新表述为排序问题，并使用学习排序 (Learning to Rank, LTR) 来解决它。

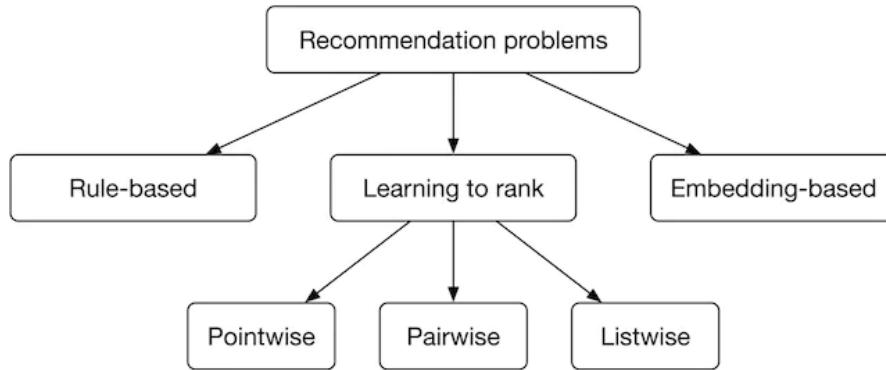


图 7.2: 解决推荐问题的不同方法

学习排序 (LTR) 是一种应用监督学习解决排序问题的算法技术。排序问题可以形式化定义为：“在给定一个查询和一个项目列表的情况下，如何按项目对查询的相关性从高到低进行最佳排序？”通常有三种 LTR 方法：点对点、对对比和列表对比。让我们简单地了解每种方法。请注意，这些方法的详细解释超出了本书的范围。如果你有兴趣了解更多 LTR 的知识，请参考 [1](#)。

7.2.3.1 点对点 (POINTWISE) LTR

在这种方法中，我们遍历每个项目，并使用分类或回归方法预测查询与该项目之间的相关性。请注意，一个项目的得分是独立预测的，不受其他项目的影响。

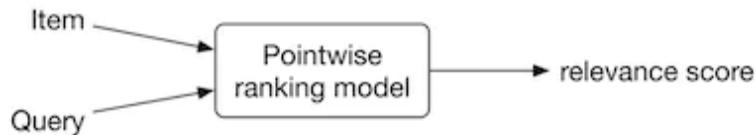


图 7.3: 点对点排序模型

最终的排序是通过对预测的相关性分数进行排序来实现的。

7.2.3.2 对对比 (PAIRWISE) LTR

在这种方法中，模型取两个项目，并预测哪个项目对查询更相关。

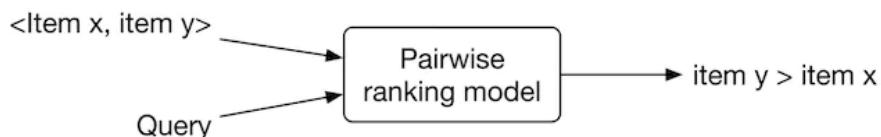


图 7.4: 对对比排序模型

一些最流行的对对比 LTR 算法包括 RankNet [2](#)、LambdaRank [3](#) 和 LambdaMART [4](#)。

7.2.3.3 列表对比 (LISTWISE) LTR

列表对比方法在给定查询的情况下，预测整个项目列表的最佳排序。

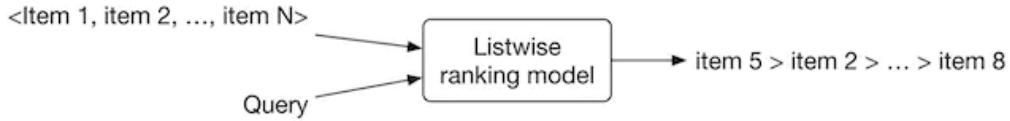


图 7.5: 列表对比排序模型

一些流行的列表对比 LTR 算法包括 SoftRank 5、ListNet 6 和 AdaRank 7。

一般来说，对对比和列表对比方法能够产生更精确的结果，但它们更难实现和训练。为了简化问题，我们在这里使用点对点方法。具体而言，我们采用一个二元分类模型，该模型每次接受一个活动并预测用户会注册该活动的概率。此方法如图 7.6 所示。

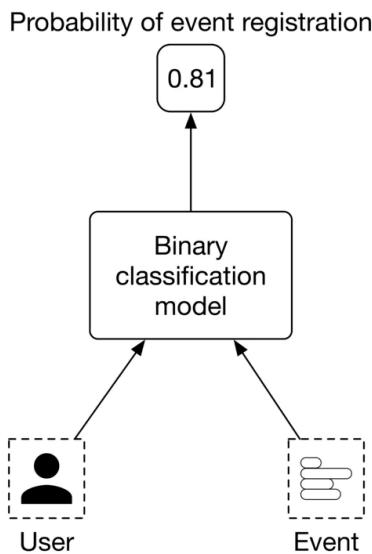


图 7.6: 二元分类模型

7.3 数据准备

7.3.1 数据工程

为了构造出良好的特征，我们首先需要了解系统中可用的原始数据。由于活动管理平台主要围绕用户和活动，因此我们假设以下数据是可用的：

- 用户
- 活动
- 友谊关系
- 互动数据

7.3.1.1 用户

用户数据架构如下所示。

ID	Username	Age	Gender	City	Country	Language	Time zone
----	----------	-----	--------	------	---------	----------	-----------

表 7.1: 用户数据架构

7.3.1.2 活动

表 7.2 显示了活动数据的可能格式。

ID	Host User ID	Category /Subcategory	Description	Price	Location	Date/Time
1	5	Music Concert	Dua Lipa Tour in Miami	200-900	American Airlines Arena Miami, FL	09/18/2022 19:00-24:00
2	11	Sports Basketball	Golden State Warriors vs. Milwaukee Bucks	140-2500	Chase Center SF, CA	09/22/2022 17:00-19:00
3	7	Art Theater	The Comedy and Magic of Robert Hall	Free	San Jose Improv San Jose, CA	09/06/2022 18:00-19:30

表 7.2: 活动数据

7.3.1.3 友谊关系

在表 7.3 中，每一行代表两个用户之间建立的友谊以及建立友谊的时间戳。

User ID 1	User ID 2	Timestamp when friendship was formed
28	3	1658451341
7	39	1659281720
11	25	1659312942

表 7.3: 友谊关系数据

7.3.1.4 互动数据

表 7.4 存储了用户与活动的互动数据，如活动注册、邀请和曝光。在实际操作中，我们可能会将互动数据存储在不同的数据库中，但为了简单起见，我们将它们包含在一个表中。

7.3.2 特征工程

基于活动的推荐比传统推荐更具挑战性。一个活动本质上不同于电影或书籍，因为在活动结束后不再有消费行为。活动通常是短暂的，这意味着从活动创建到结束的时间很短。因此，针对给定活动的历史互动数据较少。因此，基于活动的推荐本质上是冷启动的，并且面临持续的新项目问题。

User ID	Event ID	Interaction type	Interaction value	Location (lat, long)	Timestamp
4	18	Impression	-	38.8951,-77.0364	1658450539
4	18	Register	Confirmation number	38.8951,-77.0364	1658451341
4	18	Invite	User 9	41.9241,-89.0389	1658451365

表 7.4: 互动数据

为了克服这些问题，我们在特征工程方面投入更多的精力，以创建尽可能多的有意义的特征。由于篇幅限制，我们只讨论一些最重要的特征。在实际操作中，预测性特征的数量可能会更高。在本节中，我们创建了与以下类别相关的特征：

- 位置相关特征
- 时间相关特征
- 社交相关特征
- 用户相关特征
- 活动相关特征

7.3.2.1 位置相关特征

活动地点的可达性如何？ 活动地点的可达性是一个重要因素。例如，如果一个活动位于远离公共交通的山上，那么通勤可能会让用户不愿意参加。我们创建以下特征来捕捉可达性：

- **步行分数：** 步行分数是介于 0 到 100 之间的一个数字，用于衡量某个地址的可步行性，基于与附近设施的距离。它是通过分析诸如到设施的距离、行人友好度、人口密度等因素计算得出的。我们假设步行分数可以从外部数据源（如 Google 地图、Open Street Map 等）中获得。表 7.5 显示了步行分数的分类。

Category	Walk score	Description
1	90-100	No car needed
2	70-89	Very walkable
3	50-69	Somewhat walkable
4	25-49	Car-dependent
5	0-24	Requires a car

表 7.5: 步行分数分类

- **步行分数相似性：** 活动步行分数与用户之前注册的活动的平均步行分数之间的差异。
- **公交分数、公交分数相似性、骑行分数、骑行分数相似性。**

活动是否与用户位于同一国家和城市？ 对于用户来说，活动是否在他们所在的国家和城市是一个非常重要的决定因素。可以创建以下两个特征：

- 如果用户的国家与活动所在国家相同，则该特征为 1，否则为 0。
- 如果用户的国家与活动所在城市相同，则该特征为 1，否则为 0。

用户是否对距离感到舒适？ 有些用户可能更喜欢非常接近他们位置的活动，而其他人则更喜欢远一些的活动。我们使用以下特征来捕捉这一点：

- 用户位置与活动位置之间的距离。这个值可以从外部 API 获得，并划分为几个类别。例如：
 - 0: 小于 1 英里
 - 1: 1-5 英里
 - 2: 5-20 英里
 - 3: 20-50 英里
 - 4: 50-100 英里
 - 5: +100 英里
- **距离相似性：**到活动的距离与用户之前注册活动的平均距离之间的差异（在实际操作中，可以使用中位数或百分位范围）。

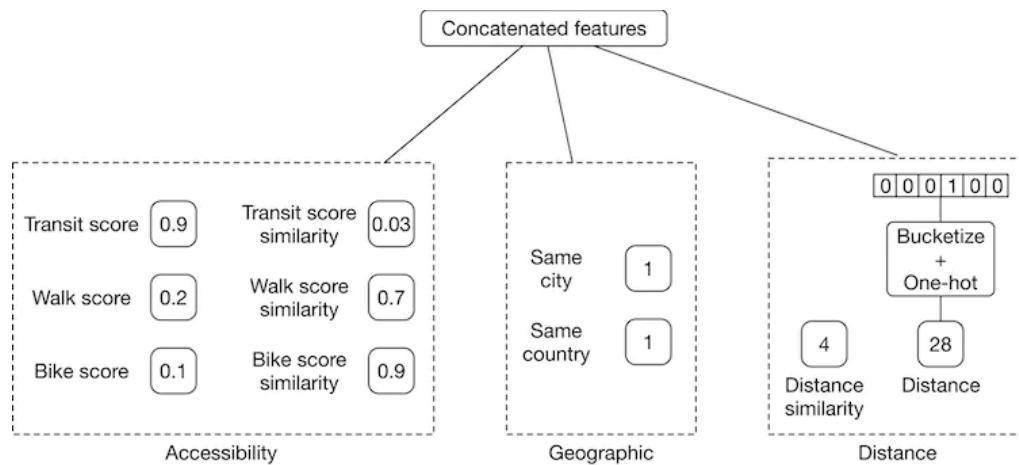


图 7.7: 位置相关特征

7.3.2.2 时间相关特征

距离活动开始还有多少时间合适？ 有些用户可能会提前几天规划活动，而其他人则不会。我们创建以下特征来捕捉这一点：

- **距离活动开始的剩余时间。**这个特征可以划分为不同的类别，并使用独热编码。例如：

- 0: 距离活动开始不到 1 小时
- 1: 1-2 小时
- 2: 2-4 小时
- 3: 4-6 小时
- 4: 6-12 小时
- 5: 12-24 小时
- 6: 1-3 天
- 7: 3-7 天
- 8: +7 天

- **剩余时间相似性:** 活动的“剩余时间”与用户之前注册活动的平均“剩余时间”之间的差异。
- **从用户位置到活动位置的预估旅行时间。**这个值将从外部服务中获得，并划分为类别。
- **预估旅行时间相似性:** 活动的预估旅行时间与用户之前注册活动的平均预估旅行时间之间的差异。

活动日期和时间是否方便用户？有些用户可能更喜欢在周末参加活动，而其他人则喜欢工作日。一些用户喜欢早晨的活动，而其他用户则更喜欢晚间活动。为了捕捉用户对一周中不同天的历史偏好，我们创建了一个用户档案。这个用户档案是一个大小为 7 的向量，每个值表示用户在特定日期参加活动的次数。通过将这些值除以总的参加活动次数，我们可以得到用户在一周中每一天参加活动的历史比率。图 7.8 显示了用户以前参加的活动在每一天的分布情况。可以看到，这个用户从未在周一或周三参加过活动，因此为这个用户显示在周三举行的活动可能不是一个好的推荐。可以用类似的方法创建每小时的用户档案。同样，我们添加了日期和时间的相似性。

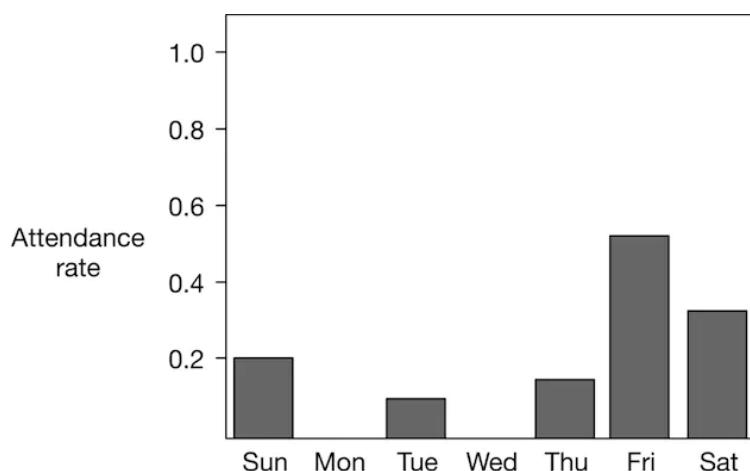


图 7.8: 用户之前参加活动的每周分布

图 7.9 显示了时间相关特征的概述。

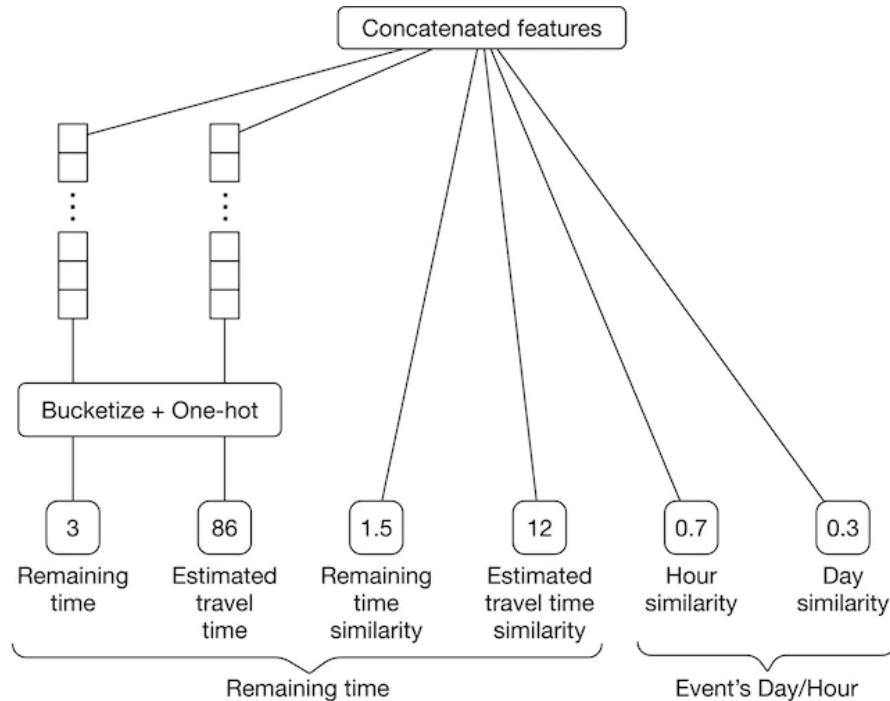


图 7.9: 时间相关特征概述

7.3.2.3 社交相关特征

有多少人参加了这个活动？ 一般来说，如果有很多其他参与者，用户更有可能注册参加活动。我们提取以下特征来捕捉这一点：

- 注册参加这个活动的用户数量
- 注册用户总数与曝光次数的比例
- **注册用户相似性**：待评估活动的注册用户数量与用户之前注册活动的差异

与朋友出席相关的特征。 如果用户的朋友参加了某个活动，那么他们更有可能注册参加。我们可以使用以下特征：

- 用户的朋友中有多少人注册了这个活动
- 注册朋友数量与朋友总数的比例
- **注册朋友相似性**：待评估活动的注册朋友数量与用户之前注册活动的差异

用户是否被其他人邀请参加此活动？ 用户更有可能参加那些他们被邀请参加的活动。一些可能有帮助的特征是：

- 邀请用户参加活动的朋友数量
- 邀请用户参加活动的其他用户数量

活动的主持人是用户的朋友吗？ 用户倾向于参加他们朋友创建的活动。我们创建一个二元特征来反映这一点：如果活动的主持人是用户的朋友，该值为 1，否则为 0。

用户以前参加过多少次此主持人创建的活动？ 有些用户对特定主持人的活动感兴趣。

7.3.2.4 用户相关特征

年龄和性别。 一些活动面向特定的年龄段和性别群体。例如，“Women in Tech” 和 “30 岁如何取得人生成功” 是可能针对某些特定人群的活动。我们创建两个特征来捕捉这一点：

- 用户的性别，使用独热编码
- 用户的年龄，划分为多个类别并使用独热编码

7.3.2.5 活动相关特征

活动的价格： 活动的价格可能会影响用户是否注册参加。可以使用的一些特征包括：

- 活动的价格，划分为几个类别。例如：
 - 0: 免费
 - 1: \$1-\$99
 - 2: \$100-\$499
 - 3: \$500-\$1,999
 - 4: +\$2,000
- **价格相似性：** 待评估活动的价格与用户之前注册活动的平均价格之间的差异。

这个活动的描述与之前注册的描述有多相似？ 这反映了用户的兴趣，基于他们之前注册的活动。例如，如果在之前活动描述中反复出现“音乐会”一词，这可能表明用户对音乐会类活动感兴趣。为了捕捉这一点，我们创建一个特征，用于表示活动描述与用户之前注册活动描述之间的相似性。为了计算相似性，将描述转换为数值向量（使用 TF-IDF），并使用余弦距离计算相似性。

注意，由于描述是由主持人手动提供的，这个特征可能会带有噪声。我们可以通过在有和没有这个特征的情况下训练模型来实验，以衡量其重要性。

图 7.10 显示了用户特征、活动特征和社交相关特征的概述。

上述列出的特征并不详尽。在实际操作中，还可以创建很多其他预测性特征。例如，与主持人相关的特征，如主持人的受欢迎程度、用户的搜索历史、活动的类别、自动生成的活动标签等。在面试中，不必严格按照本节内容。你可以以此为起点，然后讨论对面试官更相关的主题。以下是一些你可能想要详细讨论的潜在话题：

- **批处理特征与流式特征：** 批处理（静态）特征是指变化不太频繁的特征，如年龄、性别和活动描述。可以通过批处理定期计算这些特征，并将其存储在特征存储库中。相比之下，流式（也称为动态）特征变化较快。例如，注册某个活动的用户数量和活动的剩余时间，都是动态特征。面试官可能希望你深入讨论这个话题以及机器学习中的批处理与在线处理。如果你有兴趣了解更多，请参考 8。

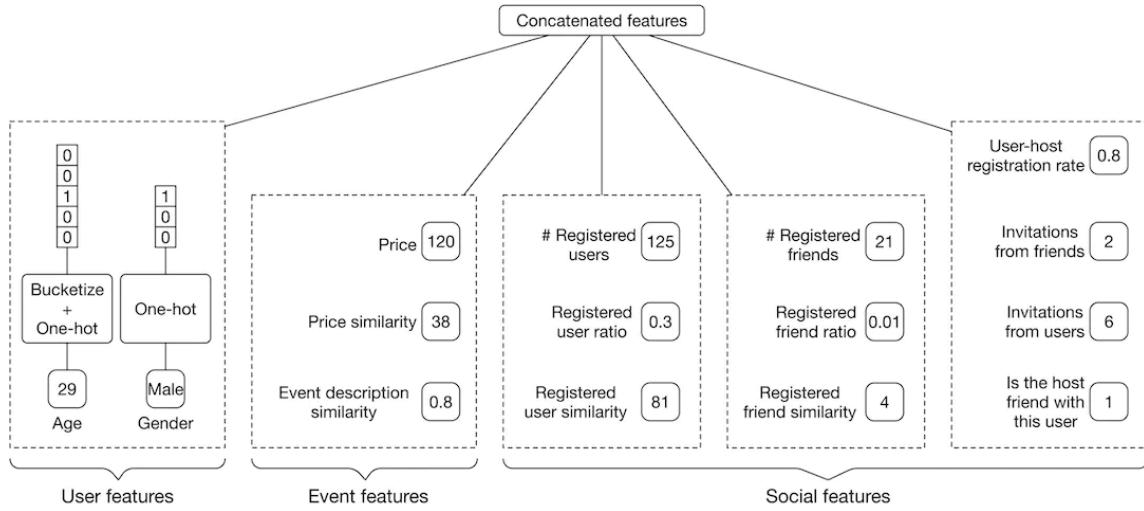


图 7.10: 用户、活动和社交特征

- **特征计算效率。** 实时计算特征并不高效。你可能想讨论这个问题以及可能的解决方法。例如，我们可以不将用户当前位置与活动位置之间的距离计算为特征，而是将这两个位置作为两个单独的特征传递给模型，并依靠模型根据两个位置隐式计算有用的信息。要了解更多关于如何为机器学习模型准备位置数据的信息，请参考 9。
- **使用衰减因子。** 用于依赖于用户最近 X 次互动的特征。衰减因子为用户最近的互动/行为赋予更多的权重。
- **使用嵌入学习。** 将每个活动和用户转换为嵌入向量。这些嵌入向量用于作为活动和用户的特征。
- **从用户属性创建特征可能会引入偏差。** 例如，依赖年龄或性别来决定求职者是否适合某份工作，可能会导致歧视。由于我们从用户属性中创建特征，因此需要意识到潜在的偏差问题。

7.4 模型开发

7.4.1 模型选择

可以使用多种机器学习方法来解决二元分类问题。让我们来看一下以下几种方法：

- 逻辑回归
- 决策树
- 梯度提升决策树 (Gradient-boosted decision tree, GBDT)
- 神经网络

7.4.1.1 逻辑回归 (LOGREG)

LogReg 通过使用一个或多个特征的线性组合来建模二元结果的概率。关于 LogReg 的详细信息，请参考 [10](#)。

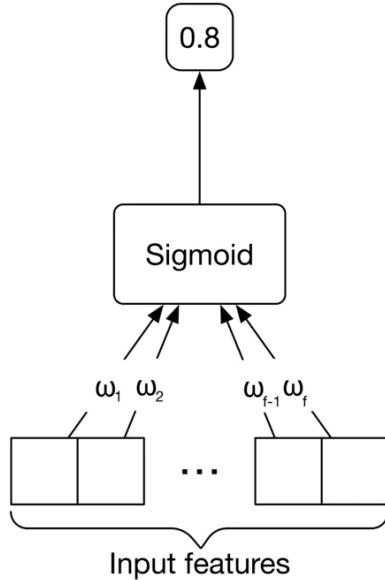


图 7.11: 逻辑回归

让我们看看 LogReg 的优缺点。

优点:

- **推理速度快。** 计算输入特征的加权组合速度很快。
- **训练效率高。** 由于结构简单，易于实现、解释，并且训练速度快。
- 在数据线性可分的情况下表现良好（图 7.12）。
- **可解释且易于理解。** 每个特征分配的权重指示了不同特征的重要性，这使我们能够了解做出决策的原因。

缺点:

- **无法解决非线性问题**，因为它使用输入特征的线性组合。
- **多重共线性**出现在两个或多个特征高度相关时。LogReg 的已知局限之一是，当输入特征中存在多重共线性时，它无法很好地学习任务。

在我们的系统中，输入特征的数量可能非常多。通常，这些特征与目标变量（二元结果）之间具有复杂的非线性关系。对于 LogReg 来说，这种复杂性可能很难学习。

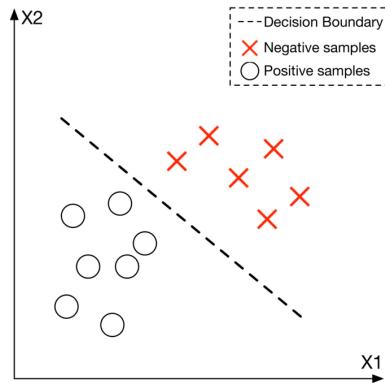


图 7.12: 逻辑回归的决策边界与线性可分数据

7.4.1.2 决策树

决策树是另一类学习方法，使用树状模型的决策及其可能的后果进行预测。图 7.13 显示了一个简单的决策树，其特征为年龄和性别，并展示了对应的决策边界。决策树中的每个叶节点表示一个二元结果，其中“+”表示给定输入被分类为正，“-”表示为负。有关决策树的详细信息，请参考 11。

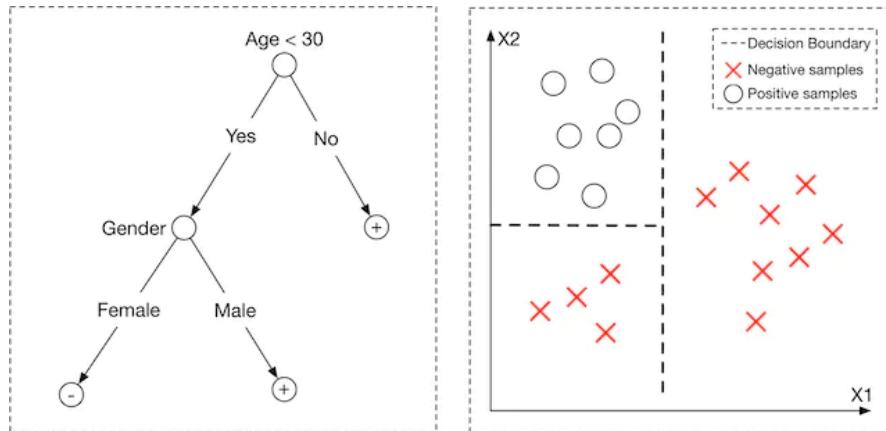


图 7.13: 决策树（左）及其学习到的决策边界（右）

优点：

- **训练速度快：** 决策树的训练速度很快。
- **推理速度快：** 决策树在推理时能快速做出预测。
- **几乎不需要数据准备：** 决策树模型不需要对数据进行归一化或缩放，因为算法不依赖于输入特征的分布。
- **可解释且易于理解。** 可视化树结构可以很好地展示做出决策的原因和重要的决策因素。

缺点：

- **非最优的决策边界：** 决策树模型会在特征空间中生成与坐标轴平行的决策边界（图 7.13）。对于某些数据分布而言，这可能不是找到决策边界的最佳方式。
- **过拟合：** 决策树对数据的微小变化非常敏感。输入数据的细微变化可能导致推理时的不同结果。同样，训练数据的微小变化也可能导致完全不同的树结构。这是一个主要问题，使得预测不太可靠。

在实际操作中，很少使用简单的决策树。原因是它们对输入数据的变化过于敏感。为减少决策树的敏感性，常用两种技术：

- 自举聚合（Bagging）
- 提升（Boosting）

这两种技术在科技行业中广泛使用。了解它们的工作原理至关重要。让我们深入了解一下。

自举聚合（Bagging） Bagging 是一种集成学习方法，它在多个训练数据子集上并行训练一组机器学习模型。在 Bagging 中，结合这些已训练模型的预测结果，以生成最终预测。这显著减少了模型对数据变化的敏感性（方差）。

Bagging 的一个例子是常用的“随机森林”模型 12。随机森林在训练过程中并行构建多个决策树，以减少模型的敏感性。为了进行预测，每棵决策树独立预测给定输入的输出类别（正或负），然后使用投票机制将这些预测结合起来生成最终预测。图 7.14 显示了一个有三棵决策树的随机森林。

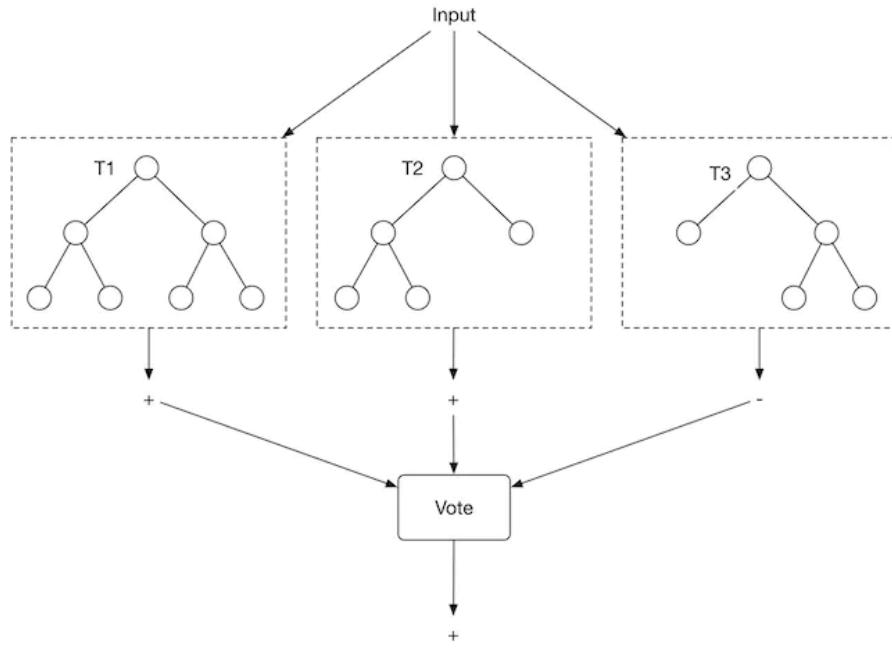


图 7.14: 随机森林

Bagging 技术有以下优点：

- 减少过拟合的影响（高方差）。
- 不显著增加训练时间，因为决策树可以并行训练。

- 在推理时不会增加太多延迟，因为决策树可以并行处理输入。

尽管有这些优点，Bagging 在模型面临欠拟合（高偏差）时并不有效。为了克服 Bagging 的缺点，我们来讨论另一种称为 Boosting 的技术。

提升（Boosting） 在机器学习中，Boosting 是指通过顺序训练多个弱分类器来减少预测误差的过程。“弱分类器”是指比随机猜测稍好的简单分类器。在 Boosting 中，多个弱分类器被组合成一个强学习模型。图 7.15 显示了一个 Boosting 的例子。

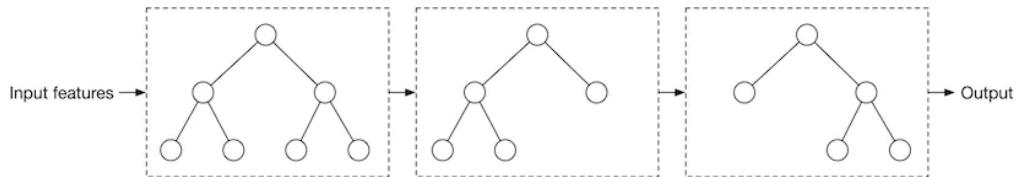


图 7.15: Boosting 示例

优点：

- **Boosting 减少了偏差和方差。** 将弱分类器结合在一起生成对数据变化不敏感的强模型。要了解更多关于偏差/方差权衡的信息，请参考 13。

缺点：

- **训练和推理较慢。** 由于分类器是基于前一个分类器的错误进行训练的，因此它们是顺序工作的。由于 Boosting 的顺序性质，这增加了推理时间。

在实际操作中，Boosting 通常比 Bagging 更受欢迎，因为 Bagging 在偏差问题上无效，而 Boosting 能减轻偏差和方差的影响。

典型的基于 Boosting 的决策树包括 Adaboost 14、XGBoost 15 和 Gradient boost 16。它们通常用于训练分类模型。

7.4.1.3 梯度提升决策树 GBDT

梯度提升决策树 (Gradient-boosted decision tree, GBDT) 是一种常用的基于树的模型，利用 GradientBoost 改进决策树。GBDT 的一些变体，如 XGBoost 15，在各种机器学习竞赛中表现出色 17。如果你有兴趣了解更多关于 GBDT 的信息，请参考 18 19。

以下是 GBDT 模型的优缺点。

优点：

- **数据准备简单：**与决策树类似，它不需要数据准备。
- **降低方差：** GBDT 使用提升技术来降低方差。
- **减少偏差：** GBDT 通过利用多个弱分类器，迭代改进之前分类器的误分类数据点，来减少预测误差。
- **适用于结构化数据。**

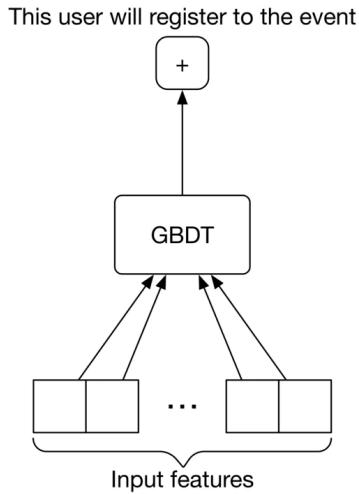


图 7.16: 具有二元输出的 GBDT 模型

缺点:

- **有很多超参数需要调整**, 如迭代次数、树深度、正则化参数等。
- GBDT 不适用于处理图像、视频、音频等非结构化数据。
- **不适合从流式数据中进行持续学习**。

在我们的案例中, 由于创建的特征是结构化数据, GBDT 或其变体之一 (如 XGBoost) 是一个值得尝试的好选择。

GBDT 的一个主要缺点是它不适合持续学习。在事件推荐系统中, 新数据会持续流入系统, 例如最近的用户互动、注册、新事件, 甚至是新用户。此外, 用户的兴趣和偏好可能会随着时间变化。一个好的事件推荐系统必须能够持续适应新数据。如果没有持续学习的可能性, 那么定期从头重新训练 GBDT 的成本非常高。接下来, 我们将探索能够克服这一限制的神经网络。

7.4.1.4 神经网络 (NN)

在事件推荐系统中, 我们有许多特征, 可能与结果之间没有线性相关性。学习这些复杂关系是困难的。此外, 适应新数据的持续学习也是必要的。

神经网络在解决这些挑战方面表现出色。它们能够学习具有非线性决策边界的复杂任务。此外, 神经网络模型可以很容易地对新数据进行微调, 使它们非常适合持续学习。如果你不熟悉神经网络的详细信息, 建议阅读 [20](#)。

让我们看看它的优缺点。

优点:

- **持续学习**: 神经网络被设计为可以从数据中学习并不断改进。
- 适用于非结构化数据, 例如文本、图像、视频或音频。

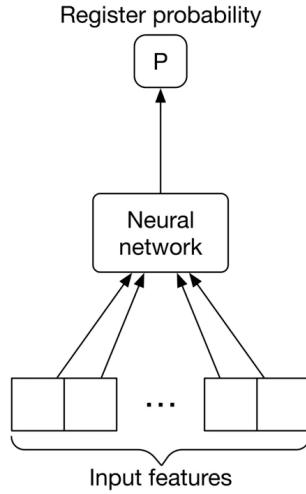


图 7.17: 神经网络的输入输出

- **表现力:** 神经网络具有很强的表达能力, 因为它们有大量的学习参数。它们可以学习非常复杂的任务和非线性决策边界。

缺点:

- **训练计算量大。**
- **输入数据的质量强烈影响结果:** 神经网络对输入数据敏感。例如, 如果输入特征的范围差异很大, 模型在训练阶段可能会收敛得很慢。神经网络的一个重要步骤是数据准备, 例如归一化、对数缩放、单热编码等。
- **需要大量的训练数据来训练神经网络。**
- **黑箱性质:** 神经网络不可解释, 这意味着很难理解每个特征对结果的影响, 因为输入特征通过多个非线性变换层。

7.4.2 我们应该选择哪个模型?

选择合适的模型是具有挑战性的。我们通常需要尝试不同的模型, 以确定哪个效果最好。可以根据各种因素选择合适的模型:

- 任务的复杂性
- 数据分布和数据类型
- 产品要求或约束条件, 例如训练成本、速度、模型大小等。

在这个问题中, GBDT 和神经网络都是值得实验的候选模型。我们从 GBDT 变体 XGBoost 开始, 因为它实现和训练速度都很快。结果可以作为初始基准。

一旦我们有了基准, 就可以探索使用神经网络构建更好的模型。神经网络在以下方面有望表现出色:

- 我们的系统中有大量的训练数据。用户通过注册活动、邀请朋友、发布新活动等不断与系统交互。考虑到用户数量，这会为训练提供大量数据。
- 数据可能不是线性可分的，而神经网络可以学习非线性数据。

在设计神经网络架构时，需要考虑多个超参数，包括隐藏层的数量、每层的神经元数量、激活函数等。这些可以通过超参数调优技术来确定。神经网络的架构细节通常不是机器学习系统设计面试的重点，因为没有系统的方法来选择正确的架构。

7.5 模型训练

7.5.1 构建数据集

构建训练和评估数据集是开发模型的重要步骤。举个例子，让我们看看如何计算特征及其标签。

为了构建一个数据点，我们从交互数据中提取一个〈用户，活动〉对，并从该对中计算输入特征。如果用户已注册该活动，则将数据点标记为 1，否则标记为 0。

#	Extracted ⟨user, event⟩ features							Label
1		1	0	1	1	0	1	
2		0	0	0	1	1	0	

图 7.18: 构建的数据集

构建数据集后，我们可能会遇到一个问题，即类别不平衡。原因是用户可能在注册活动之前会浏览几十或几百个活动。因此，负样本〈用户，活动〉对的数量远高于正样本。我们可以使用以下技术之一来解决类别不平衡问题：

- 使用焦点损失或类别平衡损失训练分类器
- 欠采样多数类别

7.5.2 选择损失函数

由于该模型是二元分类模型，我们使用典型的分类损失函数，例如二元交叉熵 (Binary Cross-Entropy)，来优化神经网络模型。

7.6 评估

7.6.1 离线指标

为了评估排序系统，我们考虑以下选项。

Recall@k 或 Precision@k。这些指标不太适合，因为它们不考虑输出的排序质量。

MRR, nDCG, 或 mAP。这三种指标通常用于衡量排序质量。但哪一个是最佳选择呢？

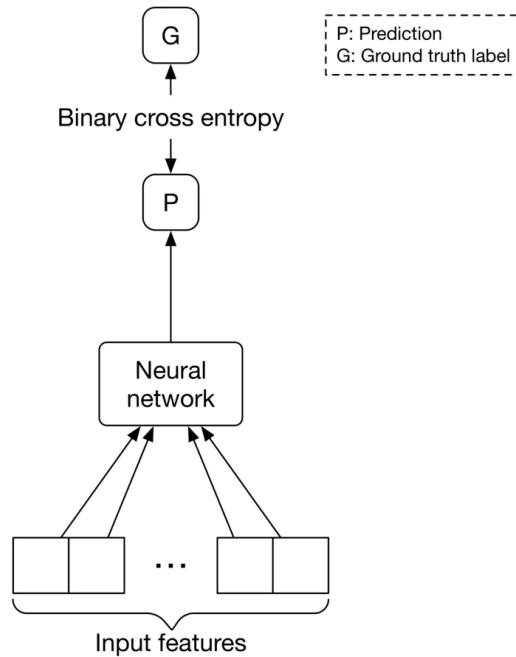


图 7.19: 预测值和标签之间的损失

MRR 关注列表中第一个相关项的排名位置，适用于只期望检索到一个相关项的系统。然而，在事件推荐系统中，多个推荐的事件可能对用户相关。MRR 不太适用。

nDCG 适用于用户和项目之间的相关性分数是非二元的情况。相比之下，mAP 仅适用于相关性分数为二元的情况。由于事件要么相关（用户已注册），要么不相关（用户看到事件但未注册），因此 mAP 更适合。

7.6.2 在线指标

在我们的案例中，业务目标是通过增加售票量来提高收入。为了衡量系统对收入的影响，让我们探讨以下指标：

- 点击率 (CTR)
- 转化率
- 收藏率
- 收入提升

CTR。该比率显示用户在看到推荐事件后点击事件的频率。

$$\text{CTR} = \frac{\text{total number of clicked events}}{\text{total number of impressions}}$$

较高的 CTR 表明我们的系统在推荐用户点击的事件方面表现良好。更多的点击通常意味着更多的事件注册。

然而，仅依赖 CTR 作为在线指标可能不够充分。一些事件可能具有吸引点击的标题（clickbait）。理想情况下，我们希望衡量推荐的事件对用户的相关性。这个指标被称为转化率，我们现在将讨论它。

转化率。该比率显示用户在看到推荐事件后实际注册的频率。公式如下：

$$\text{Conversion rate} = \frac{\text{total number of event registrations}}{\text{total number of impressions}}$$

较高的转化率表明用户更频繁地注册推荐的事件。例如，转化率为 0.3 表示用户平均在每 10 个推荐事件中注册 3 个。

收藏率。该比率显示用户收藏推荐事件的频率。基于平台允许用户保存或收藏事件的假设。

收入提升。这是由于事件推荐而导致的收入增加。

7.7 服务

在本节中，我们提出一个可以用于处理请求的机器学习系统设计。如图 7.20 所示，该设计有两个主要的 pipeline：

- 在线学习 pipeline
- 预测 pipeline

7.7.1 在线学习 pipeline

如前所述，事件推荐本质上是冷启动的，并且面临不断的新项问题。因此，模型必须不断进行微调以适应新数据。这个 pipeline 负责通过合并新数据不断训练新模型、评估训练好的模型并进行部署。

7.7.2 预测 pipeline

预测 pipeline 负责预测给定用户的前 k 个最相关的事件。让我们讨论预测 pipeline 的一些最重要的组件。

事件过滤 事件过滤组件将查询用户作为输入，并将事件从 100 万缩小到一小部分事件。这基于简单的规则，如事件位置或其他类型的用户过滤条件。例如，如果用户添加了“仅限音乐会”的过滤器，该组件可以快速将列表缩小到候选事件的子集。由于这些类型的过滤器在事件推荐系统中很常见，它们可以显著减少搜索空间，从可能的数百万个事件减少到数百个候选事件。

排序服务 此服务将用户和事件过滤组件生成的候选事件作为输入，为每个（用户，事件）对计算特征，根据模型预测的概率对事件进行排序，并输出用户最相关的前 k 个事件的排序列表。

排序服务与负责计算模型所需特征的特征计算组件进行交互。静态特征从特征存储中获取，而动态特征则是从原始数据中实时计算的。

7.8 其他讨论点

如果在面试结束时还有时间，可以讨论以下附加点：

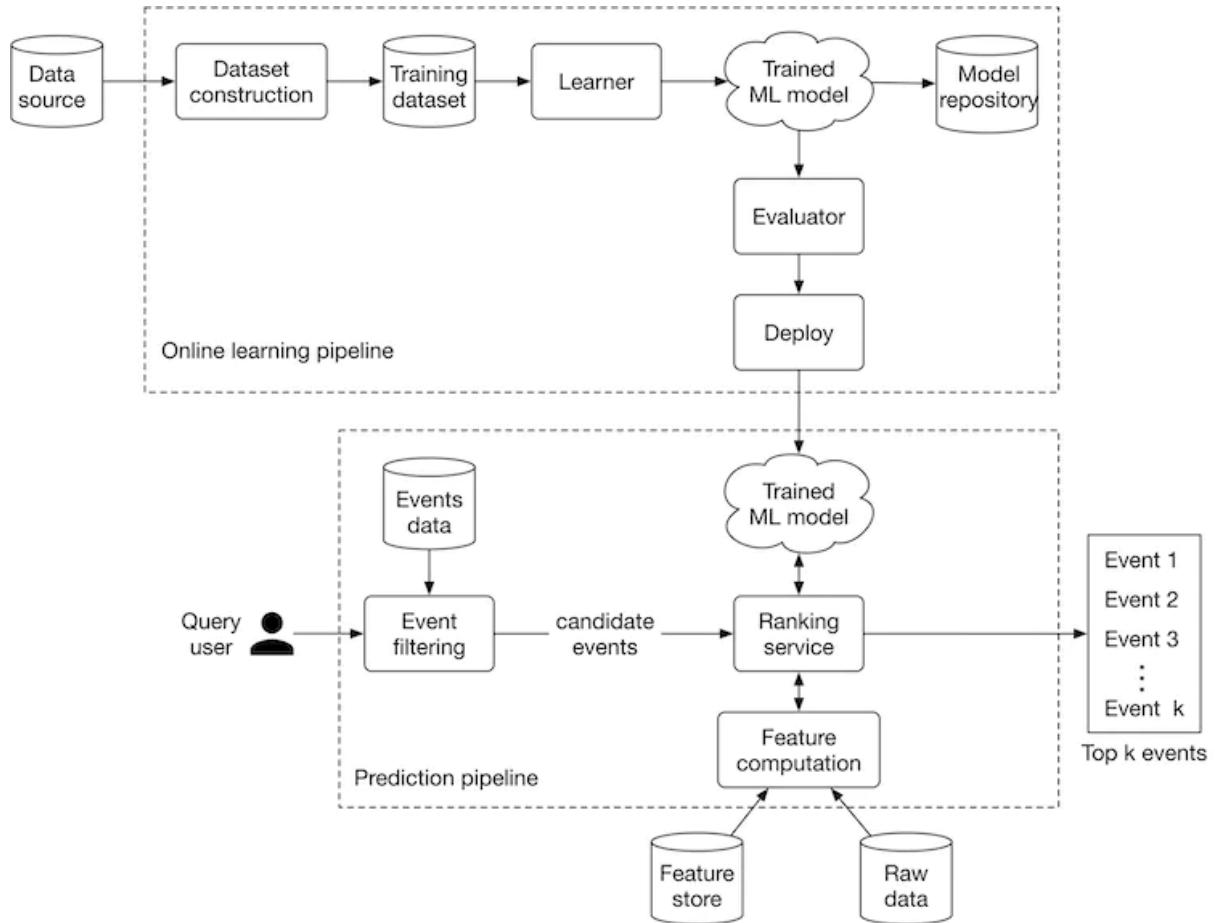


图 7.20: 机器学习系统设计

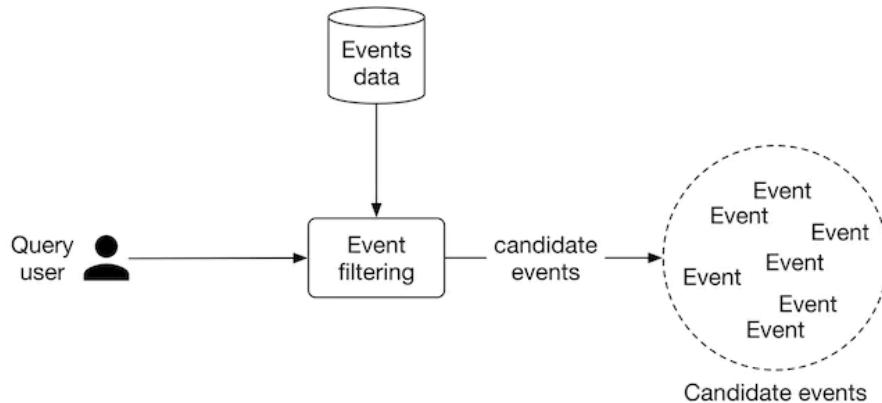


图 7.21: 事件过滤的输入输出

- 我们可能会在系统中观察到的不同类型的偏差。[21](#)
- 如何利用特征交叉来提高模型的表现力。[22](#)
- 有些用户喜欢看到多样化的事件列表。如何确保推荐的事件既多样化又新鲜?[23](#)

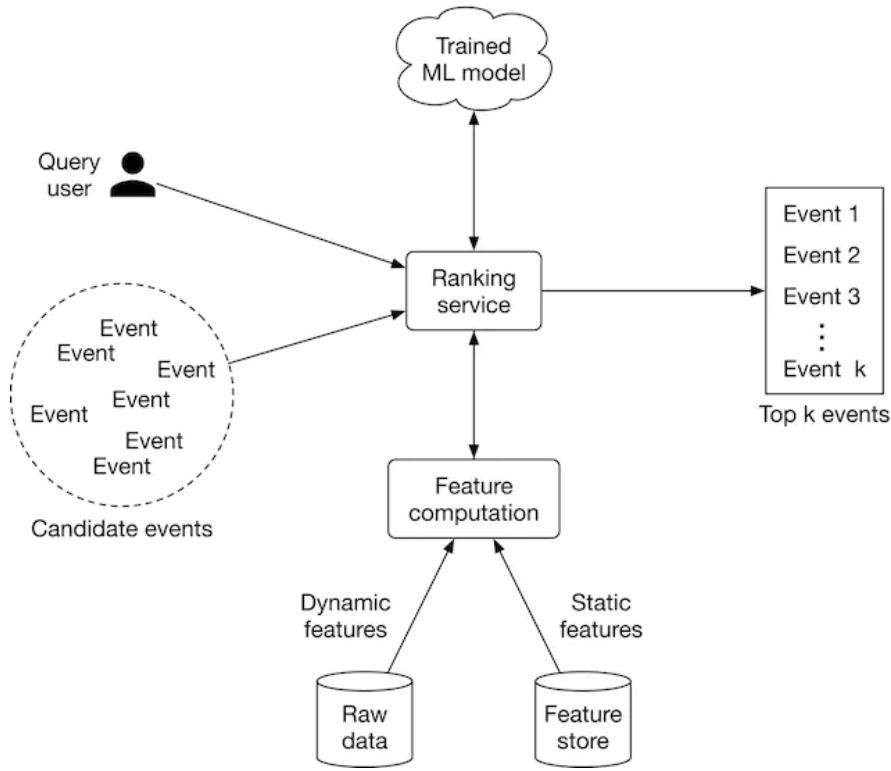


图 7.22: 排序服务工作流程

- 我们使用用户属性来训练模型，同时依赖用户的实时位置。有哪些与隐私和安全相关的额外考虑因素？[24](#)
- 事件管理平台通常是双边市场，活动主办方是供应方，用户满足需求方。如何确保系统不会只针对一方进行优化？此外，如何确保平台对不同主办方公平？要了解更多关于双边市场中的独特挑战，请参考[25](#)。
- 构建数据集时如何避免数据泄漏。[26](#)
- 如何确定更新模型的合适频率。[27](#)

References

1. Learning to rank methods. <https://livebook.manning.com/book/practical-recommender-systems/chapter-13/53>.
 2. RankNet paper. https://icml.cc/2015/wp-content/uploads/2015/06/icml_ranking.pdf.
 3. LambdaRank paper. <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/lambdarank.pdf>.
 4. LambdaMART paper. <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf>.
 5. SoftRank paper. <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/SoftRankWsdm08Submitted.pdf>.
 6. ListNet paper. <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2007-40.pdf>.
 7. AdaRank paper. <https://dl.acm.org/doi/10.1145/1277741.1277809>.
 8. Batch processing vs stream processing. <https://www.confluent.io/learn/batch-vs-real-time-data-processing/#:~:text=Batch%20processing%20is%20when%20the,data%20flows%20through%20a%20system>.
 9. Leveraging location data in ML systems. <https://towardsdatascience.comleveraging-geolocation-data-for-machine-learning#:~:text=Location%20data%20is%20an%20important,base%20on%20your%20customer%20data>.
 10. Logistic regression. <https://www.youtube.com/watch?v=yIYKR4sgzI8>.
 11. Decision tree. <https://careerfoundry.com/en/blog/data-analytics/what-is-a-decision-tree/>.
 12. Random forests. https://en.wikipedia.org/wiki/Random_forest.
 13. Bias/variance trade-off. <http://www.cs.cornell.edu/courses/cs578/2005fa/CS578.bagging.boosting.lecture.pdf>.
 14. AdaBoost. <https://en.wikipedia.org/wiki/AdaBoost>.
 15. XGBoost. <https://xgboost.readthedocs.io/en/stable/>.
 16. Gradient boosting. <https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithms/>.
 17. XGBoost in Kaggle competitions. <https://www.kaggle.com/getting-started/145362>.
 18. GBDT. <https://blog.paperspace.com/gradient-boosting-for-classification/>.
 19. An introduction to GBDT. <https://www.machinelearningplus.com/machine-learning/an-introduction-to-gradient-boosting/>.
 20. Introduction to neural networks. <https://www.youtube.com/watch?v=0twSSFZN9Mc>.
 21. Bias issues and solutions in recommendation systems. <https://www.youtube.com/watch?v=pPq9iyGIZZ8>.

22. Feature crossing to encode non-linearity. <https://developers.google.com/machine-learning/crash-course/feature-crosses/encoding-nonlinearity>.
23. Freshness and diversity in recommendation systems. <https://developers.google.com/machine-learning/recommendation/dnn/re-ranking>.
24. Privacy and security in ML. <https://www.microsoft.com/en-us/research/blog/privacy-preserving-machine-learning/>.
25. Two-sides marketplace unique challenges. <https://www.uber.com/blog/uber-eats-recommending-marketplace-challenges/>.
26. Data leakage. <https://machinelearningmastery.com/data-leakage-machine-learning/>.
27. Online training frequency. <https://huyenchip.com/2022/01/02/real-time-machine-learning-challenges.html#towards-continual-learning>.

8 Ad Click Prediction on Social Platforms 社交平台上的广告点击预测

8.1 Introduction

在线广告允许广告商在平台上投标并展示他们的广告，以获得可衡量的响应，如展示、点击和转化。向用户展示相关广告是许多在线平台（如 Google、Facebook 和 Instagram）的基本功能。

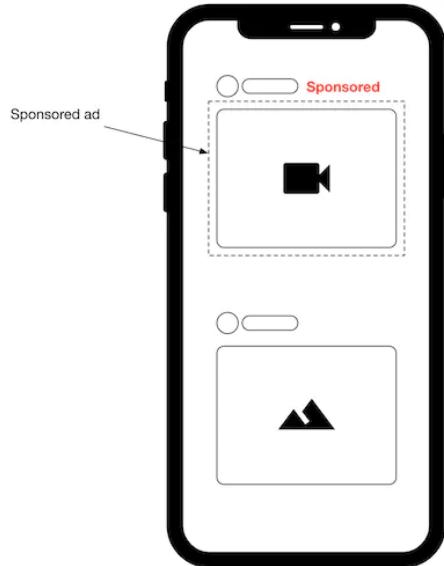


图 8.1: 用户时间线上的赞助广告

在本章中，我们设计一个类似于流行社交媒体平台使用的广告点击预测系统。

8.2 明确需求

以下是候选人与面试官之间的典型对话。

候选人：我可以假设建立广告预测系统的业务目标是最大化收入吗？

面试官：是的，没错。

候选人：广告有不同的类型，如视频广告和图片广告。此外，广告可以以不同的尺寸和格式展示，比如用户的时间线、弹出广告等。为了简化起见，我可以假设广告只展示在用户的时间线上，并且每次点击产生相同的收入吗？

面试官：可以。

候选人：系统可以多次向同一用户展示相同的广告吗？

面试官：是的，我们可以多次展示一个广告。有时，一个广告在多次展示后才会被点击。实际上，公司通常有一个“疲劳期”，即如果用户多次忽略某个广告，系统在 X 天内不会再次向用户展示该广告。为了简化，我们假设没有疲劳期。

候选人：我们是否支持“隐藏此广告”功能？“屏蔽此广告商”呢？这些负面反馈有助于检测不相关的广告。

面试官：好问题。假设用户可以隐藏他们不喜欢的广告。“屏蔽此广告商”是一个有趣的功能，但我们现在不需要支持它。

候选人：是否可以假设训练数据集应基于用户和广告数据构建，标签应基于用户与广告的互动？

面试官：可以。

候选人：我们可以通过用户点击生成正样本，但如何生成负样本？是否可以假设任何未点击的展示都是负样本？如果用户快速滚动，没花时间看广告呢？如果我们将一次展示算作负样本，但最终用户点击了该广告呢？

面试官：这些问题很好。你有什么想法？

候选人：如果广告在用户屏幕上可见一段时间但未被点击，我们可以将其视为负样本。另一种方法是假设展示为负样本，直到出现点击。此外，我们可以依靠“隐藏此广告”等负面反馈来标记负样本。

面试官：很有道理！在实践中，我们可能会使用其他复杂的技术来标记负样本。对于这次面试，让我们按你的建议进行。

候选人：在广告点击预测系统中，模型从新互动中不断学习是非常关键的。假设持续学习是必要的，这样合理吗？

面试官：很好的观点。实验表明，更新模型时的5分钟延迟就可能损害性能¹。

让我们总结一下问题描述。我们需要设计一个广告点击预测系统。系统的业务目标是最大化收入。广告只展示在用户的时间线上，每次点击产生相同的收入。模型需要持续学习新互动数据。我们从用户和广告数据中构建数据集，并根据互动对其进行标注。在本章中，我们不会讨论广告技术(AdTech)的具体话题，因为它们与机器学习面试无关。想了解更多关于广告技术的信息，请参考²。

8.3 将问题框定为机器学习任务

8.3.1 定义机器学习目标

广告点击预测系统的目标是通过向用户展示他们更可能点击的广告来增加收入。这可以转换为以下机器学习目标：预测广告是否会被点击。这是因为通过正确预测点击概率，系统可以向用户展示相关广告，从而提高收入。

8.3.2 规定系统的输入和输出

广告点击预测系统以用户为输入，并基于点击概率输出排序的广告列表。

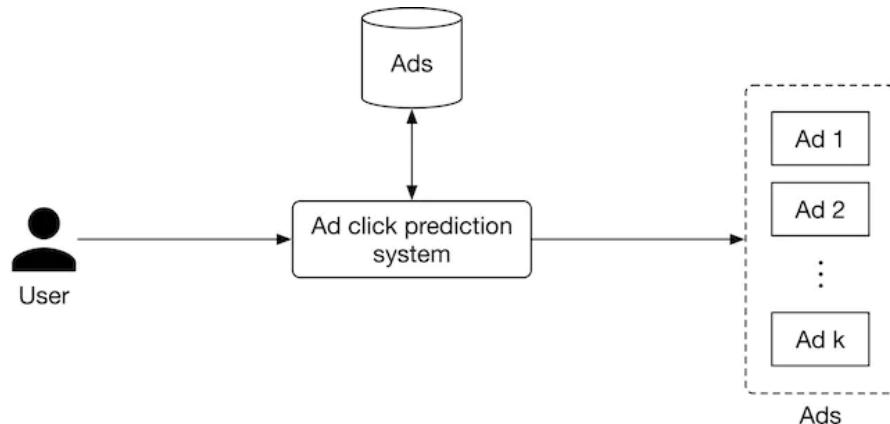


图 8.2: 广告点击预测系统的输入和输出

8.3.3 选择合适的机器学习类别

图 8.2 展示了如何将广告预测框定为一个排序问题。如第 7 章《事件推荐系统》中所述，点对点学习排序 (LTR) 是解决排序问题的一个很好起点。点对点 LTR 采用一个二分类模型，以〈用户，广告〉对作为输入，预测用户是否会点击广告。图 8.3 显示了模型的输入和输出。

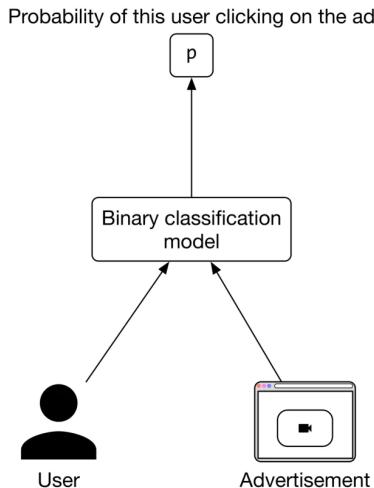


图 8.3: 二分类模型的输入输出

8.4 数据准备

8.4.1 数据工程

以下是该系统中的一些原始数据：

- 广告
- 用户
- 用户-广告互动

广告 广告数据如表 8.1 所示。实际上，我们可能会为每个广告关联数百个属性。为了简化，我们仅列出重要的属性。

Ad ID	Advertiser ID	Ad group ID	Campaign ID	Category	Subcategory	Images or Videos
1	1	4	7	travel	hotel	http://cdn.mysite.com/u1.jpg
2	7	2	9	insurance	car	http://cdn.mysite.com/t3.mp4
3	9	6	28	travel	airline	http://cdn.mysite.com/t5.jpg

表 8.1: 广告数据

ID	Username	Age	Gender	City	Country	Language	Time zone
----	----------	-----	--------	------	---------	----------	-----------

表 8.2: 用户数据模式

用户 用户数据的模式如下所示。

用户-广告互动 该表存储了用户与广告的互动数据，如展示、点击和转化。

User ID	Ad ID	Interaction type	Dwell time	Location (lat, long)	Timestamp
11	6	Impression	5sec	38.8951 -77.0364	165845053
11	7	Impression	0.4 sec	41.9241 -89.0389	1658451365
4	20	Click		22.7531 47.9642	1658435948
	6	Conversion		22.7531 47.9642	

表 8.3: 用户-广告互动数据

8.4.2 特征工程

我们在本节中的目标是构建有助于预测用户点击的特征。

8.4.3 广告特征

广告特征包括以下内容：

- ID
- 图像/视频
- 类别和子类别
- 展示和点击次数

让我们详细研究每个特征。

8.4.3.1 ID

包括广告商 ID、活动 ID、广告组 ID、广告 ID 等。

为什么重要？这些 ID 表示广告商、活动、广告组和广告本身。它们作为预测特征，用于捕捉不同广告商、活动、广告组和广告的独特特征。

如何准备？嵌入层将稀疏特征（如 ID）转换为密集特征向量。每种 ID 类型都有其自己的嵌入层。

8.4.3.2 图像/视频

为什么重要？帖子中的视频或图像是另一个有助于预测广告内容的信号。例如，飞机的图像可能表明广告与旅行有关。

如何准备？首先对图像或视频进行预处理。之后，我们使用预训练模型（如 SimCLR 3）将非结构化数据转换为特征向量。

8.4.3.3 广告类别和子类别

由广告商提供的广告类别和子类别。例如，这里列出了一些可定位的广泛类型的类别：艺术与娱乐、汽车与车辆、美容与健身等。

为什么重要？它有助于模型理解广告所属的类别。

如何准备？这些类别和子类别是由广告商根据预定义的列表手动提供的。想了解更多文本数据的准备方法，请参考第4章《YouTube视频搜索》。

8.4.3.4 展示和点击次数

- 广告的总展示/点击次数
- 广告商提供的广告的总展示/点击次数
- 活动的总展示次数

为什么重要？这些数字表明其他用户对该广告的反应。例如，用户更有可能点击点击率（CTR）较高的广告。

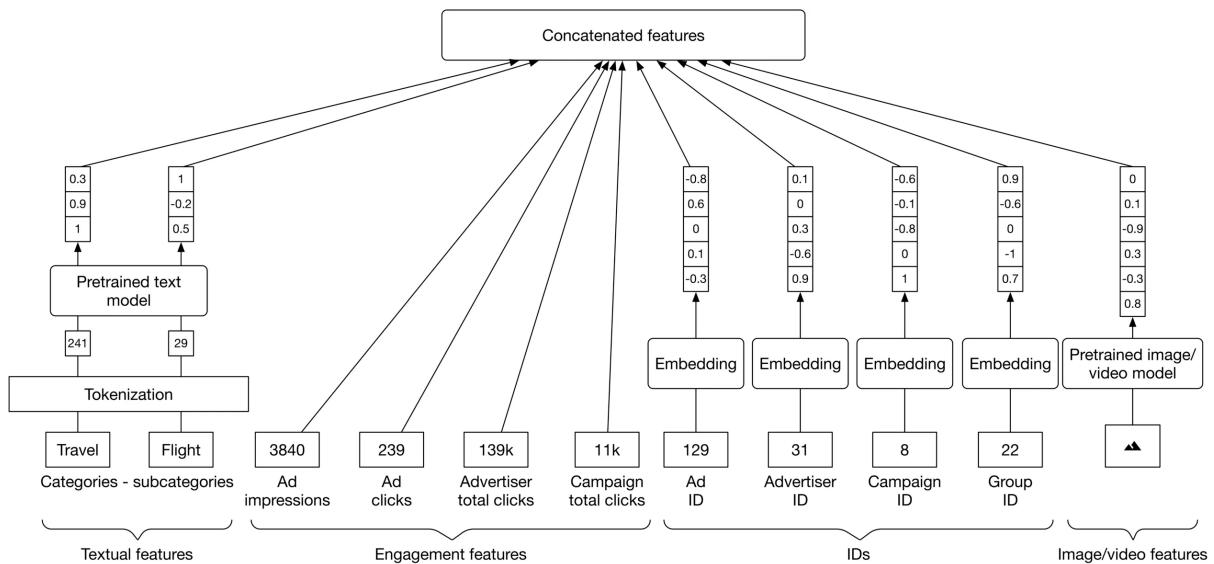


图 8.4: 广告相关特征准备概述

8.4.4 用户特征

与前几章类似，我们选择以下特征：

- 人口统计：年龄、性别、城市、国家等
- 上下文信息：设备、一天中的时间等
- 互动相关特征：点击的广告、用户的历史参与统计数据等

让我们更详细地看看与互动相关的特征。

点击的广告 用户之前点击过的广告。

为什么重要？之前的点击表明了用户的兴趣。例如，当用户点击了大量与保险相关的广告时，这表明他们可能会再次点击类似的广告。

如何准备？方法与“广告特征”中的描述相同。

用户的历史参与统计 这是用户的历史参与数据，如总广告展示次数和广告点击率。

为什么重要？个人的历史参与是未来参与的良好预测指标。通常，如果用户过去经常点击广告，他们将更有可能在未来点击广告。

如何准备？参与统计作为数值表示。为了准备这些数据，我们将其值缩放到类似的范围。

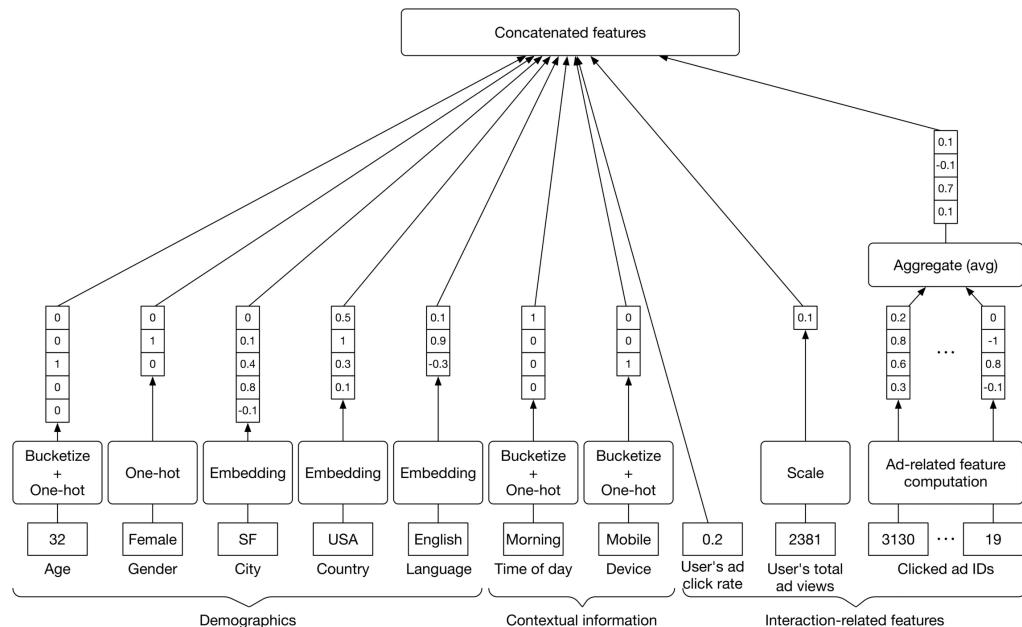


图 8.5: 用户元数据和互动的特征准备

在结束数据准备部分之前，让我们来看看广告点击预测系统中常见的挑战。在大多数情况下，这些系统需要处理大量高基数 (cardinality) 的类别特征。例如，“广告类别”取值来自可能的所有类别的庞大列表。类似地，“广告商 ID”和“用户 ID”可能有数百万个唯一值，具体取决于平台上有多少活跃的用户或广告商。由于通常存在巨大的特征空间，因此常见的情况是成千上万个特征大多是零。在模型选择部分中，我们将介绍克服这些独特挑战的技术。

8.5 模型开发

8.5.1 模型选择

如“将问题框架化为机器学习任务”部分所述，我们选择使用二元分类模型来解决排名问题。二元分类可以通过多种方式建模。以下是广告点击预测系统中常见的选择：

- 逻辑回归
- 特征交叉 + 逻辑回归

- 梯度提升决策树 (GBDT)
- 梯度提升决策树 + 逻辑回归
- 神经网络
- 深度与交叉网络 (DCN)
- 因子分解机 (FM)
- 深度因子分解机 (DeepFM)

8.5.1.1 逻辑回归 (LOGREG)

LogReg 使用一个或多个特征的线性组合来建模二元结果的概率。LogReg 训练速度快，易于实现。然而，基于 LogReg 的广告点击预测系统存在以下缺点：

- **无法解决非线性问题。** LogReg 使用输入特征的线性组合来解决任务，这导致了线性决策边界。在广告点击预测系统中，数据通常不是线性可分的，因此 LogReg 的表现可能较差。
- **无法捕捉特征交互。** LogReg 无法捕捉特征之间的交互。在广告预测系统中，特征之间的各种交互非常常见。当特征之间存在交互时，输出概率不能简单地表示为特征效果之和，因为一个特征的效果取决于另一个特征的值。

鉴于这两个缺点，LogReg 并不是广告预测系统的最佳选择。然而，由于它易于实现且训练速度快，许多公司会使用它来创建基线模型。

8.5.1.2 特征交叉 + LOGREG

为了更好地捕捉特征交互，我们使用一种称为特征交叉的技术。

什么是特征交叉？

特征交叉是一种用于机器学习的技术，用于从现有特征中创建新特征。它通过取现有特征的乘积、和或其他组合来生成新特征。通过这种方式，可以捕捉原始特征之间的非线性交互，从而提高机器学习模型的性能。例如，“年轻且喜欢篮球”或“美国和足球”之类的交互可能会积极影响模型预测点击概率的能力。

如何创建特征交叉？

在特征交叉中，我们根据先验知识手动将新特征添加到现有特征中。如图 8.6 所示，交叉两个特征（例如“国家”和“语言”）会为现有特征空间添加六个新特征。要了解更多关于交叉的内容，请参考 4。

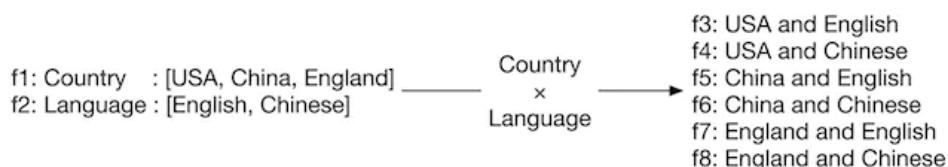


图 8.6: 交叉两个特征：国家和语言

如何使用特征交叉 + LogReg ?

如图 8.7 所示，特征交叉 + LogReg 的工作流程如下：

1. 对原始特征集进行特征交叉以提取新特征（交叉特征）。
2. 使用原始特征和交叉特征作为 LogReg 模型的输入。

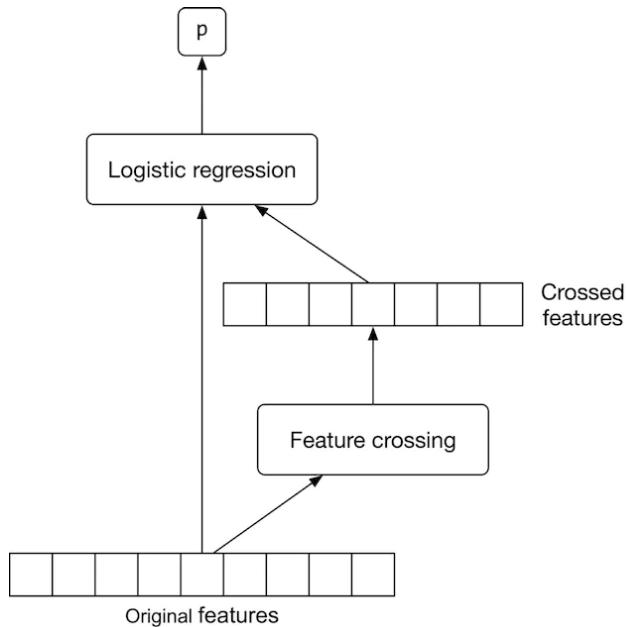


图 8.7: 对原始特征进行特征交叉

该方法允许模型捕捉某些二阶特征交互。然而，它有三个缺点：

- **手动过程：**需要人为选择要交叉的特征，这既耗时又昂贵。
- **需要领域知识：**特征交叉需要领域专业知识。要确定哪些特征交互是模型的预测信号，我们需要提前了解问题和特征空间。
- **无法捕捉复杂交互：**交叉特征可能不足以捕捉来自数千个稀疏特征的所有复杂交互。
- **稀疏性：**原始特征可能是稀疏的。随着特征交叉的引入，交叉特征的 cardinality 可能变得更大，从而导致更大的稀疏性。

鉴于这些缺点，该方法并不是广告预测系统的理想解决方案。

8.5.1.3 梯度提升决策树 (GBDT)

我们在第 7 章《活动推荐系统》中讨论了 GBDT。这里，我们将仅探讨 GBDT 在广告点击预测系统中的优缺点。

优点

- GBDT 具有可解释性且易于理解。

缺点

- 不适合持续学习。在广告点击预测系统中，我们不断收集新的数据，如用户、广告和互动数据。要持续地对新数据进行模型训练，通常有两种选择：1) 从头训练，或 2) 在新数据上微调模型。GBDT 并非为新数据微调设计的。因此，我们通常需要从头训练模型，这在大规模情况下效率不高。
- 无法训练嵌入层。在广告预测系统中，通常有许多稀疏的类别特征，而嵌入层是表示这些特征的有效方式。然而，GBDT 无法利用嵌入层的优势。

8.5.1.4 GBDT + LOGREG

该方法包含两个步骤：

1. 训练 GBDT 模型以学习任务。
2. 不直接使用训练好的模型进行预测，而是用其选择和提取新的预测特征。新生成的特征和原始特征一起用作 LogReg 模型的输入，以进行点击预测。

使用 GBDT 进行特征选择。特征选择旨在减少输入特征的数量，仅保留那些最有用和信息量大的特征。通过决策树，我们可以根据特征的重要性选择一个特征子集。要更好地了解如何使用决策树进行特征生成，请参考 5。

使用 GBDT 进行特征提取。特征提取的目的是通过从现有特征中创建新特征来减少特征数量。新提取的特征预计具有更好的预测能力。图 8.8 解释了如何使用 GBDT 提取特征。

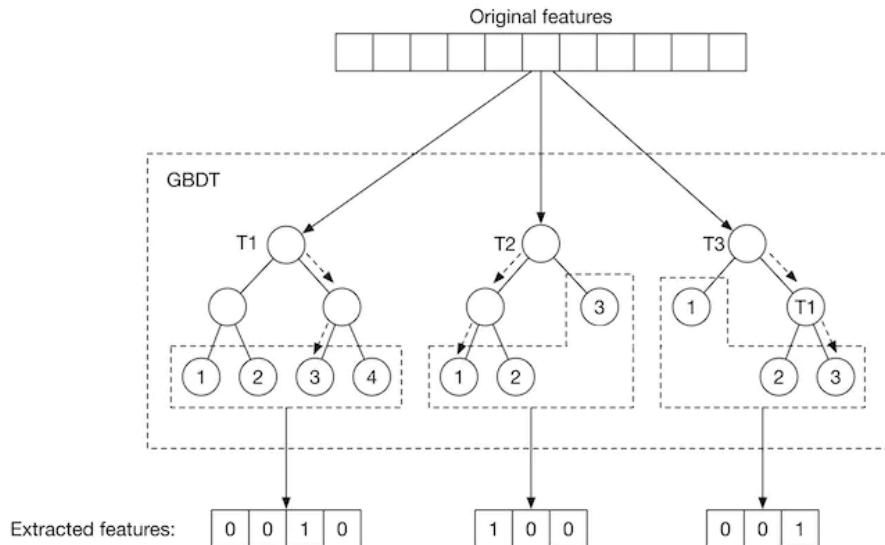


图 8.8: 使用 GBDT 进行特征提取

GBDT + LogReg 的概述如图 8.9 所示。

让我们探讨这种方法的优缺点。

优点：

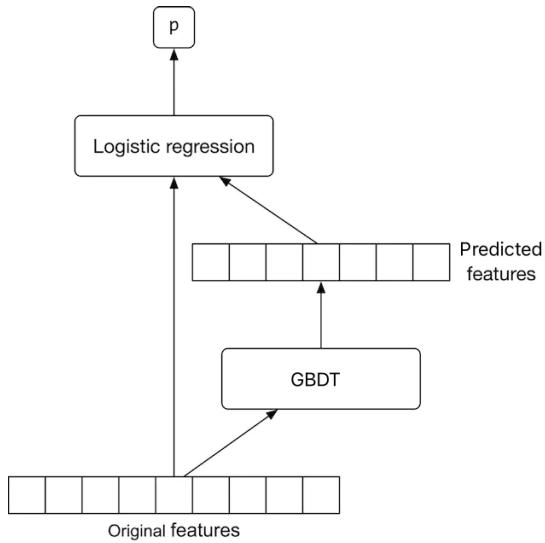


图 8.9: GBDT + LogReg 概述

- 与现有特征相比，GBDT 生成的新特征更具预测性，使得 LogReg 模型更容易学习任务。

缺点：

- 无法捕捉复杂的交互。**与 LogReg 类似，该方法无法学习成对的特征交互。
- 持续学习速度慢。**在新数据上微调 GBDT 模型需要时间，这减慢了整体持续学习的速度。

8.5.1.5 神经网络 (NN)

NN 是构建广告点击预测系统的另一种候选方法。使用 NN 来预测点击概率时，我们有两种架构选项：

- 单塔架构 NN
- 双塔架构

单塔架构 NN： 使用原始特征作为输入，神经网络输出点击概率（见图 8.10）。

双塔架构： 在这个选项中，我们使用两个编码器：用户编码器和广告编码器。广告和用户嵌入之间的相似度用于确定相关性，即点击概率。图 8.11 展示了此架构的概览。

尽管神经网络有很多好处，但它们可能并不是广告点击预测系统的最佳选择，因为：

- 稀疏性：**由于特征空间通常非常大且稀疏，大多数特征填充的都是零。神经网络可能无法有效地学习任务，因为它没有足够的数据点可用。
- 难以捕捉所有特征之间的交互：**由于特征数量巨大，很难捕捉到所有成对的特征交互。

鉴于这些限制，我们不会使用神经网络。

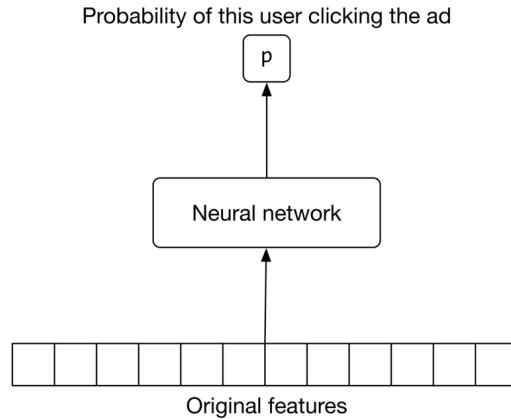


图 8.10: NN 架构

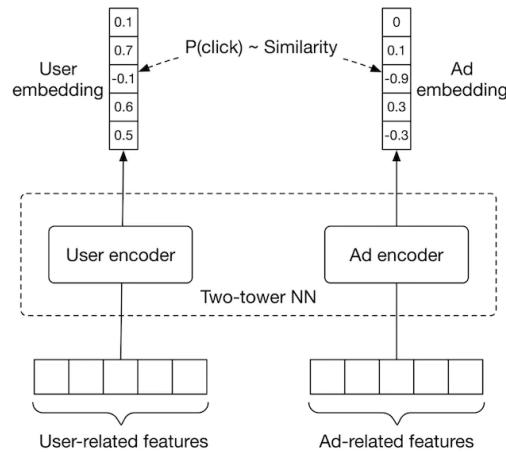


图 8.11: 基于嵌入的神经网络架构

8.5.1.6 深度与交叉网络 (DCN)

在 2017 年，Google 提出了一个名为 DCN 的架构，用来自动找到特征交互。这解决了手动特征交互方法的挑战。此方法使用以下两个并行网络：

- **深度网络：** 使用深度神经网络 (DNN) 架构学习复杂且具有泛化性的特征。
- **交叉网络：** 自动捕捉特征交互并学习良好的特征交叉。

深度网络和交叉网络的输出将被连接起来进行最终的预测。

DCN 架构有两种类型：堆叠和并行。图 8.12 展示了并行 DCN 的架构。如果你想了解更多关于堆叠架构的内容，可以参考相关资料。在机器学习系统设计面试中通常不需要详细了解 DCN，如果你有兴趣，可以深入学习。

DCN 架构比神经网络更有效，因为它隐式地学习特征交叉。然而，交叉网络只建模了某些特定的特征交互，这可能会对模型的性能产生负面影响。

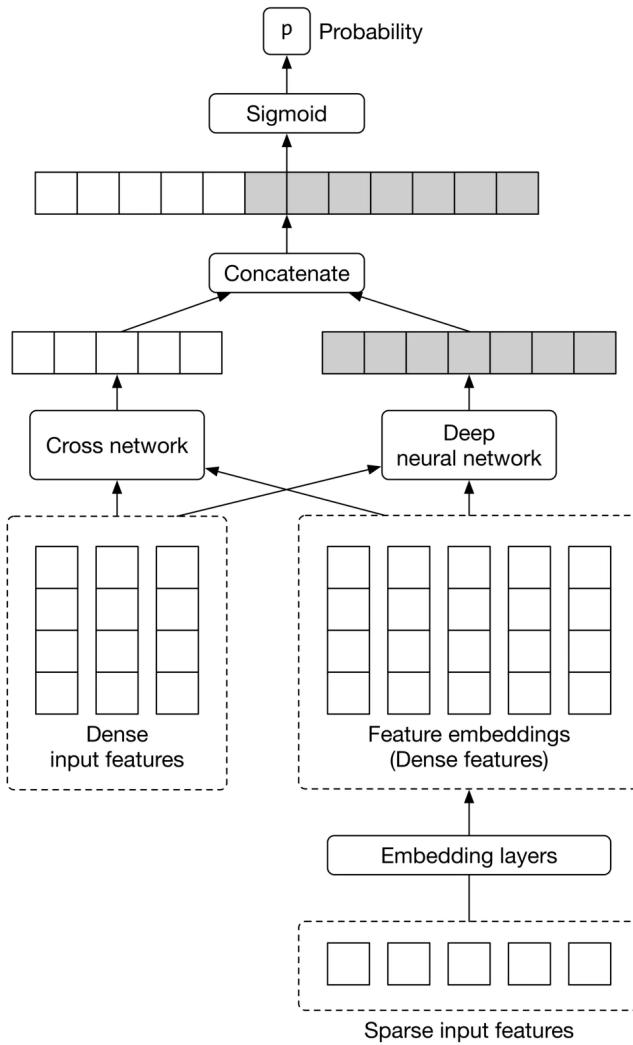


图 8.12: DCN 架构

8.5.1.7 因子分解机 (FM)

FM 是一种基于嵌入的模型，它通过自动建模所有成对特征交互来改进逻辑回归。在广告点击预测系统中，FM 被广泛使用，因为它可以有效地建模特征之间的复杂交互。

让我们了解一下 FM 的工作原理。FM 通过学习每个特征的嵌入向量来自动建模所有成对的特征交互。两个特征之间的交互由它们嵌入的点积决定。其公式如下：

$$\hat{y}(x) = w_0 + \sum_i w_i x_i + \sum_i \sum_j \langle v_i, v_j \rangle x_i x_j$$

其中， x_i 表示第 i 个特征， w_i 是学习到的权重， v_i 表示第 i 个特征的嵌入。 $\langle v_i, v_j \rangle$ 表示两个嵌入之间的点积。

这个公式看起来复杂，但实际上很容易理解。前两个项计算特征的线性组合，类似于逻辑回归的工作方式。第三项则建模成对的特征交互。图 8.13 展示了 FM 的高级概览。

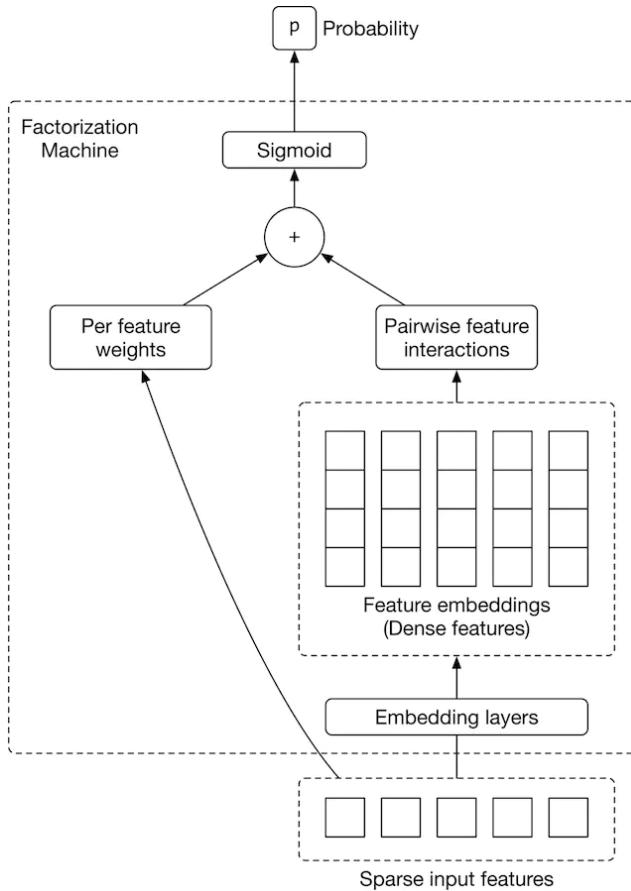


图 8.13: 因子分解机架构

FM 及其变体（如 FFM）可以有效地捕捉特征之间的成对交互。然而，FM 无法像神经网络那样学习复杂的高阶特征交互。在下一种方法中，我们结合 FM 和 DNN 来克服这个问题。

8.5.1.8 深度因子分解机 (DeepFM)

DeepFM 是一种结合了神经网络和因子分解机优点的机器学习模型。DNN 网络捕捉复杂的高阶特征，而 FM 捕捉低阶的成对特征交互。图 8.14 展示了 DeepFM 的高级架构。如果你有兴趣了解更多关于 DeepFM 的内容，可以参考相关资料。

一种潜在的改进方法是将 GBDT 与 DeepFM 结合。GBDT 将原始特征转换为更具预测性的特征，而 DeepFM 在这些新特征上进行操作。这种方法在各种广告预测系统比赛中屡获佳绩。然而，添加 GBDT 会对训练和推理速度产生负面影响，并且减慢了持续学习的过程。

在实际操作中，我们通常通过实验来选择正确的模型。在我们的案例中，我们首先使用简单的逻辑回归模型创建基线模型。接下来，我们尝试使用 DCN 和 DeepFM 进行实验，因为它们在技术行业中被广泛使用。

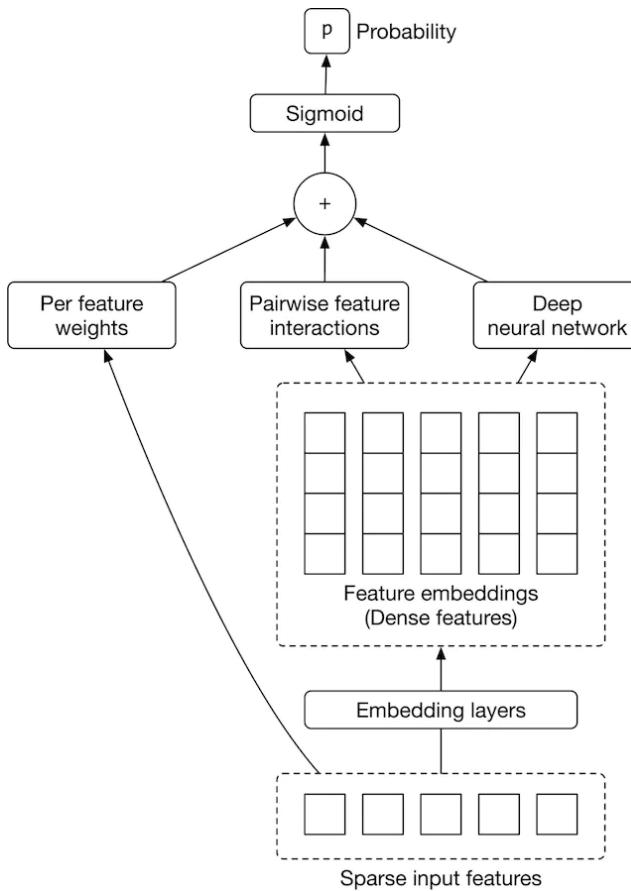


图 8.14: DeepFM 概览

8.5.2 模型训练

8.5.2.1 构建数据集

对于每个广告展示，我们构建一个新的数据点。输入特征从用户和广告中计算得出。根据以下策略为数据点分配标签：

- **正标签:** 如果用户在广告展示后 t 秒内点击广告，我们将该数据点标记为“正”标签。需要注意的是， t 是一个可以通过实验调整的超参数。
- **负标签:** 如果用户在广告展示后 t 秒内未点击广告，我们将该数据点标记为“负”标签。

实际上，公司通常使用更复杂的方法来确定标记负标签的最佳策略。想了解更多内容，请参考相关资料。

为了使模型适应新数据，它必须不断地进行训练。因此，应通过新的交互持续生成新的训练数据点。我们将在“服务”部分进一步讨论持续学习。

8.5.2.2 选择损失函数

由于我们正在训练一个二元分类模型，因此我们选择交叉熵作为分类损失函数。

#	User and interaction features	Ad features	Label
1	1 0 1 0.8 0.1 1 0	0 1 1 0.4 0.9 0	Positive
2	1 1 0 -0.6 0.9 1 1	1 1 0 0.2 0.7 1	Negative

图 8.15: 构建的数据集

8.6 评估

8.6.1 离线指标

评估广告点击预测系统通常使用以下两种指标：

- 交叉熵 (Cross-Entropy, CE)
- 归一化交叉熵 (Normalized Cross-Entropy, NCE)

交叉熵 (CE) 该指标衡量模型预测概率与真实标签的接近程度。如果系统理想地对负类预测为 0, 对正类预测为 1, 则 CE 为 0。CE 越低, 预测的准确性越高。公式为:

$$H(p, q) = - \sum_{c=1}^C p_c \log q_c$$

其中, p 是真实值, q 是预测概率, C 是类别总数。

对于二元分类, CE 公式可以重写为:

$$H(p, q) = - \sum_i p_i \log q_i = - \sum_i (y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i))$$

其中, y_i 是第 i 个数据点的真实标签, \hat{y}_i 是第 i 个数据点的预测概率。

让我们看一个具体的例子, 如图 8.16 所示。

注意, 除了用 CE 作为评估指标外, 它也通常用作分类任务中模型训练的标准损失函数。

归一化交叉熵 (NCE) NCE 是模型的 CE 与背景点击率 (训练数据中的平均点击率) 对应的 CE 之间的比值。换句话说, NCE 将模型与始终预测背景点击率的简单基线进行比较。较低的 NCE 表示模型优于简单基线。 $NCE \geq 1$ 表示模型的表现不比简单基线好。

$$\text{Normalized cross entropy} = \frac{\text{CE(ML model)}}{\text{CE(Simple baseline)}}$$

让我们通过一个具体的例子来更好地理解 NCE 的计算。如图 8.17 所示, 一个简单的基线模型始终预测 0.6 (训练数据中的点击率)。在这种情况下, NCE 值为 0.324 (小于 1), 表明模型 A 优于简单基线。

8.6.2 在线指标

让我们探讨一些在线评估中使用的指标。

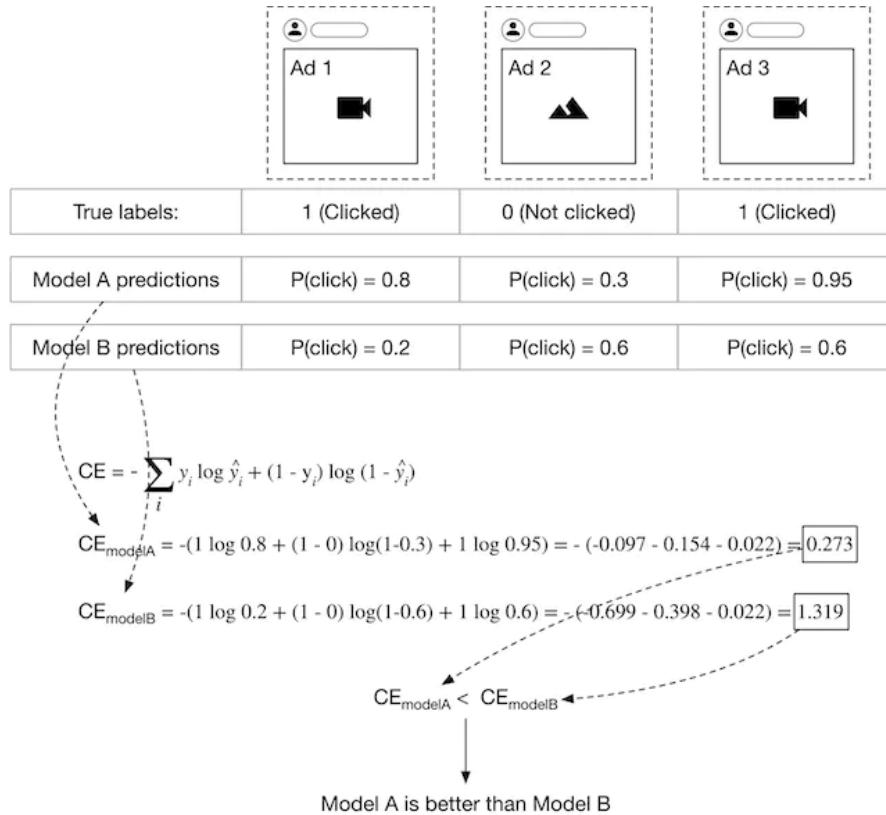


图 8.16: 两个机器学习模型的 CE 比较

- 点击率 (CTR)
- 转化率
- 收入增长
- 隐藏率

CTR. 该指标衡量点击广告的数量与显示广告的总数量之间的比率。

$$\text{CTR} = \frac{\text{Number of clicked ads}}{\text{Number of shown ads}}$$

CTR 是广告点击预测系统的一个很好的在线指标，因为最大化用户对广告的点击直接关系到收入的增加。

转化率。 该指标衡量转化次数与显示广告的总次数之间的比率。

$$\text{Conversion rate} = \frac{\text{Number of conversions}}{\text{Number of impressions}}$$

该指标很重要，因为它显示了广告主实际受益于系统的次数。如果广告未能带来转化，广告主最终会失去兴趣并停止在广告上投入。

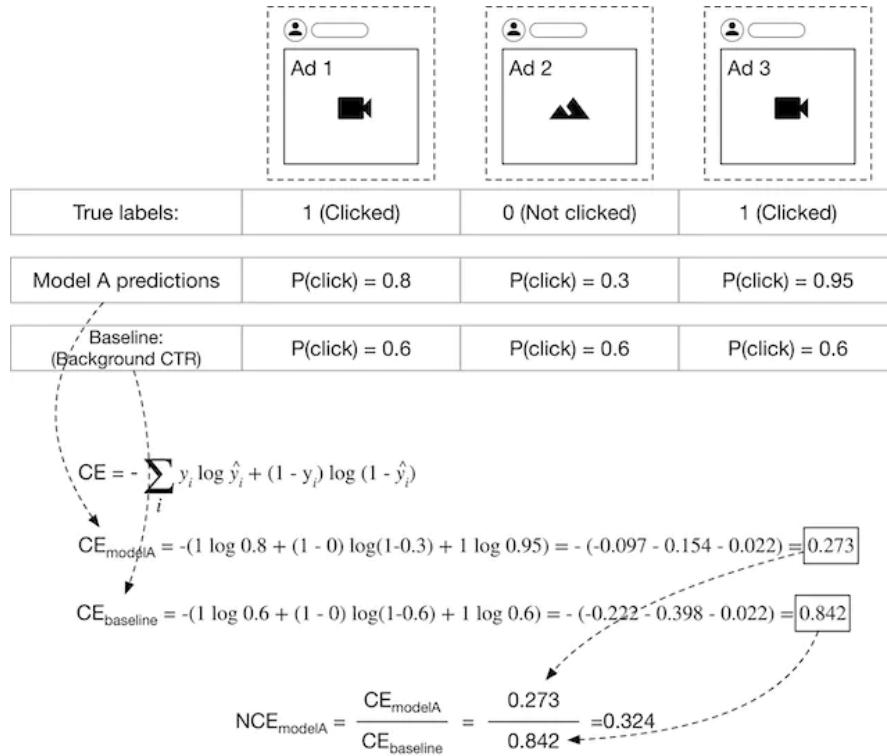


图 8.17: 模型 A 的 NCE 计算

收入增长。 该指标衡量一段时间内收入的增长百分比。

隐藏率。 该指标衡量用户隐藏广告的次数与显示广告的总次数之间的比率。

$$\text{Hide rate} = \frac{\text{Number of ads hidden by users}}{\text{Number of shown ads}}$$

该指标有助于了解系统向用户展示了多少不相关的广告，即所谓的假阳性。

8.7 服务

在服务阶段，系统负责输出按点击概率排序的广告列表。建议的机器学习系统设计如图 8.18 所示。让我们来详细分析以下几个 Pipeline：

- 数据准备 Pipeline
- 持续学习 Pipeline
- 预测 Pipeline

8.7.1 数据准备 Pipeline

数据准备 Pipeline 执行以下两个任务：

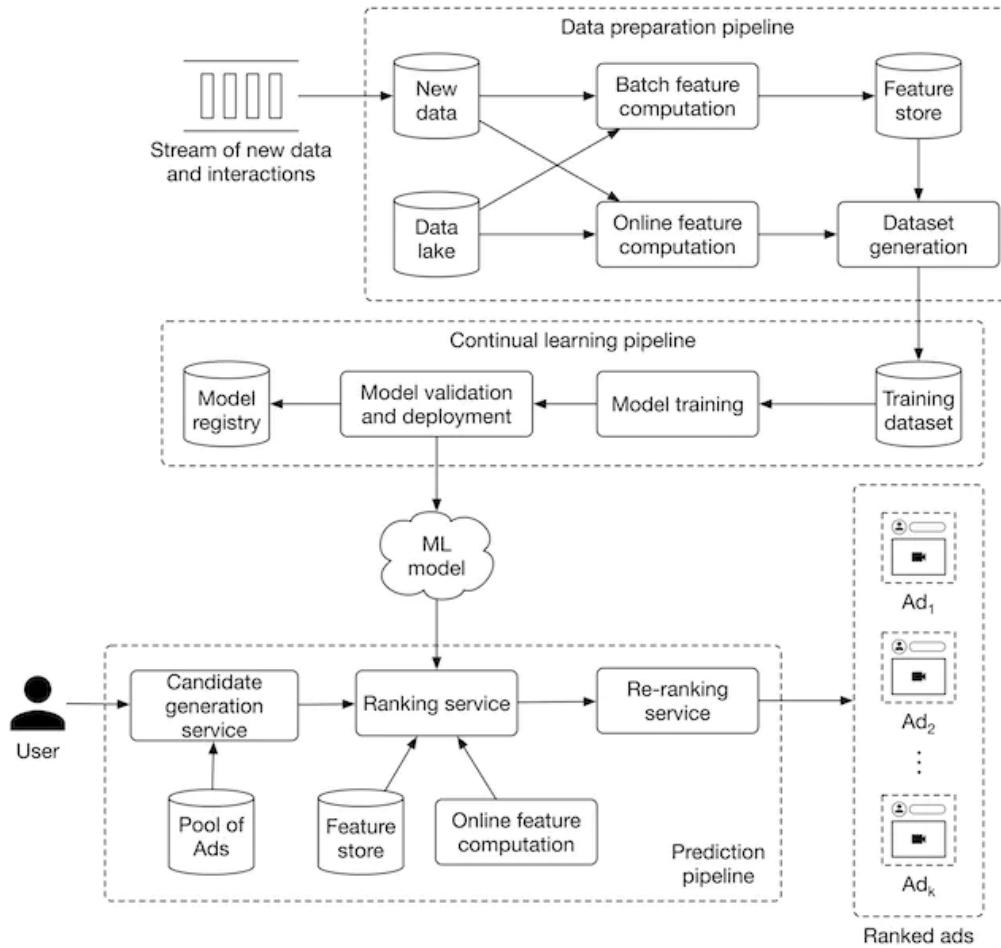


图 8.18: 机器学习系统设计

1. 计算在线和批量特征
2. 从新广告和互动中持续生成训练数据

为了计算特征，可以使用以下两种方法：批量特征计算和在线特征计算。让我们来看看它们之间的区别。

批量特征计算 我们选择的一些特征是静态的，这意味着它们变化很少。例如，广告的图片和类别就是静态特征。此组件定期（例如，每几天或几周）通过批处理作业计算静态特征，然后将特征存储在特征存储中。这提高了服务过程中的系统性能，因为特征已经预先计算好。

在线特征计算 一些特征是动态的，因为它们变化频繁。例如，广告展示次数和点击次数就是动态特征的例子。这些特征需要在查询时计算，此组件用于计算动态特征。

8.7.2 持续学习 Pipeline

根据要求，持续学习模型至关重要。此 Pipeline 负责在新训练数据上微调模型，评估新模型，并在模型改进指标时部署模型。它确保预测 Pipeline 始终使用适应最新数据的模型。

8.7.3 预测 Pipeline

预测 Pipeline 以查询用户为输入，输出按点击概率排序的广告列表。由于模型依赖的一些特征是动态的，我们不能使用批量预测。相反，系统在请求到达时使用在线预测进行服务。

正如我们在前几章中看到的，预测 Pipeline 中使用了两阶段架构。首先，我们使用候选生成服务来高效地将可用的广告池缩小到一小部分广告。在这种情况下，我们使用广告商通常提供的广告投放标准，如目标年龄、性别和国家。

接下来，我们使用排名模型从候选生成服务中获取候选广告，根据点击概率对它们进行排序，并输出排名靠前的广告。此组件与相同的特征存储和在线特征计算组件交互。获取静态和动态特征后，排名服务使用模型为每个候选广告获取预测的点击概率。这些概率用于对广告进行排序，并输出点击概率最高的广告。

最后，再排名服务通过结合额外的逻辑和启发式方法来调整广告列表。例如，我们可以通过移除列表中非常相似的广告来增加广告的多样性。

8.8 其他讨论点

如果在面试结束时还有时间，可以与面试官讨论以下一些潜在的讨论点：

- 在排名和推荐系统中，避免数据泄露非常重要 [12](#) [13](#)。
- 在广告点击预测系统中，模型需要进行校准。讨论模型校准以及校准模型的技术 [14](#)。
- 因子分解机（FM）的一个常见变体是字段感知因子分解机（FFM）。讨论 FFM 以及它与 FM 的区别 [15](#)。
- 深度因子分解机（DeepFM）的一个常见变体是 XDeepFM。讨论 XDeepFM 以及它与 DeepFM 的区别 [10](#)。
- 我们已经描述了为什么持续学习对于广告点击预测系统是必要的。然而，在新数据上进行持续学习可能导致灾难性遗忘。讨论什么是灾难性遗忘以及常见的解决方案 [16](#)。

References

1. Addressing delayed feedback. <https://arxiv.org/pdf/1907.06558.pdf>.
2. AdTech basics. <https://advertising.amazon.com/library/guides/what-is-adtech>.
3. SimCLR paper. <https://arxiv.org/pdf/2002.05709.pdf>.
4. Feature crossing. <https://developers.google.com/machine-learning/crash-course/feature-crosses/video-lecture>.
5. Feature extraction with GBDT. <https://towardsdatascience.com/feature-generation-with-gradient-boosting-trees-101>.
6. DCN paper. <https://arxiv.org/pdf/1708.05123.pdf>.
7. DCN V2 paper. <https://arxiv.org/pdf/2008.13535.pdf>.
8. Microsoft's deep crossing network paper. <https://www.kdd.org/kdd2016/papers/files/adf0975-shanA.pdf>.
9. Factorization Machines. <https://www.jefkine.com/recsys/2017/03/27/factorization-machines/>.
10. Deep Factorization Machines. https://d2l.ai/chapter_recommender-systems/deepfm.html.
11. Kaggle's winning solution in ad click prediction. <https://www.youtube.com/watch?v=4Go5crRVyuU>.
12. Data leakage in ML systems. <https://machinelearningmastery.com/data-leakage-machine-learning/>.
13. Time-based dataset splitting. https://www.linkedin.com/pulse/time-based-splitting-determining-training-test-set-when-training-machine-learning-models-1400000000000000000?trk=public_profile_article_view.
14. Model calibration. <https://machinelearningmastery.com/calibrated-classification-model-in-scikit-learn/>.
15. Field-aware Factorization Machines. <https://www.csie.ntu.edu.tw/~cjlin/papers ffm.pdf>.
16. Catastrophic forgetting problem in continual learning. <https://www.cs.uic.edu/~liub/lifelong-learning/continual-learning.pdf>.

9 Similar Listings on Vacation Rental Platforms 度假租赁平台上的相似房源

为用户推荐与其当前查看的内容相似的项目，是一种关键技术，使人们能够在大型平台上发现可能相关的内容。例如，Airbnb 推荐类似的住宿房源，亚马逊推荐类似的商品，Expedia 向用户推荐类似的体验。

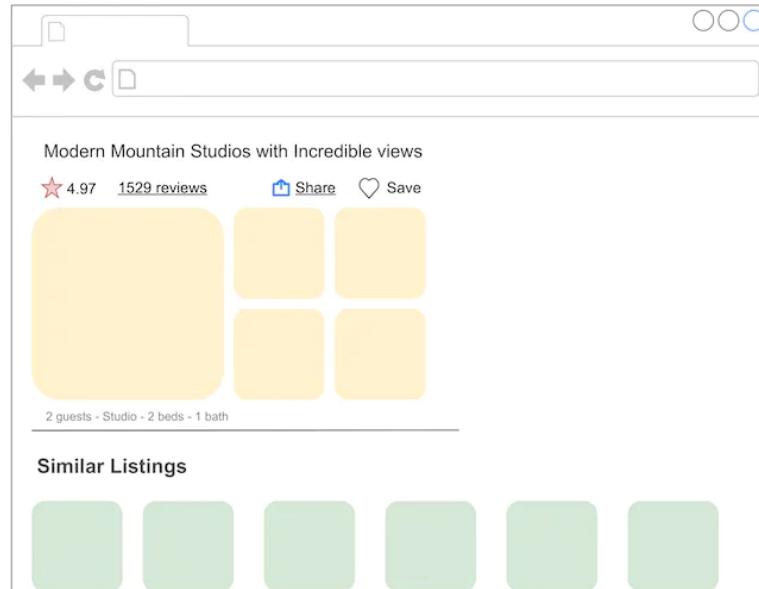


图 9.1: 推荐的相似房源

在本章中，我们设计一个“相似房源”功能，类似于 Airbnb 和 Vrbo 等度假租赁网站使用的功能。当用户点击某个特定房源时，会向他们推荐一组相似的房源。

9.1 明确需求

以下是候选人与面试官之间的典型对话。

候选人：我可以假设业务目标是增加预订数量吗？

面试官：是的。

候选人：“相似性”的定义是什么？推荐的房源是否需要与用户当前查看的房源相似？

面试官：是的，没错。当两个房源位于同一街区、城市、价格区间等时，它们被定义为相似。

候选人：推荐的房源是否需要针对用户进行个性化？

面试官：我们希望该功能能同时适用于已登录用户和匿名用户。实际上，我们会对这两类用户进行不同的处理，并对已登录用户进行个性化推荐。然而，为了简化，我们假设对已登录用户和匿名用户的处理是相同的。

候选人：平台上有多少个房源？

面试官：500 万个房源。

候选人：我们如何构建训练数据集？

面试官：这是个好问题。在这次面试中，我们假设只使用用户与房源的互动数据。模型不会利用用户的属性（例如年龄或位置）或房源的属性（如价格和位置）。

候选人：新发布的房源需要多长时间才能出现在相似房源的结果中？

面试官：假设新房源在发布后一天内可以作为推荐内容出现。在此期间，系统会收集新房源的互动数据。

让我们总结一下问题陈述。我们需要为度假租赁平台设计一个“相似房源”功能。输入是用户当前查看的特定房源，输出是用户可能接下来点击的相似房源的排序列表。对于匿名用户和已登录用户，推荐的房源应相同。平台上约有 500 万个房源，新房源在发布一天后可以出现在推荐中。系统的业务目标是增加预订数量。

9.2 将问题框架化为机器学习任务

9.2.1 定义机器学习目标

用户点击的房源序列通常具有相似的特征，例如位于同一城市或具有相似的价格区间。我们基于这一观察，定义机器学习目标为：在用户当前查看某房源的情况下，准确预测用户接下来会点击哪个房源。

9.2.2 指定系统的输入和输出

如图 9.2 所示，“相似房源”系统以用户当前查看的房源作为输入，然后输出一个房源的排序列表，该列表按用户点击的概率排序。

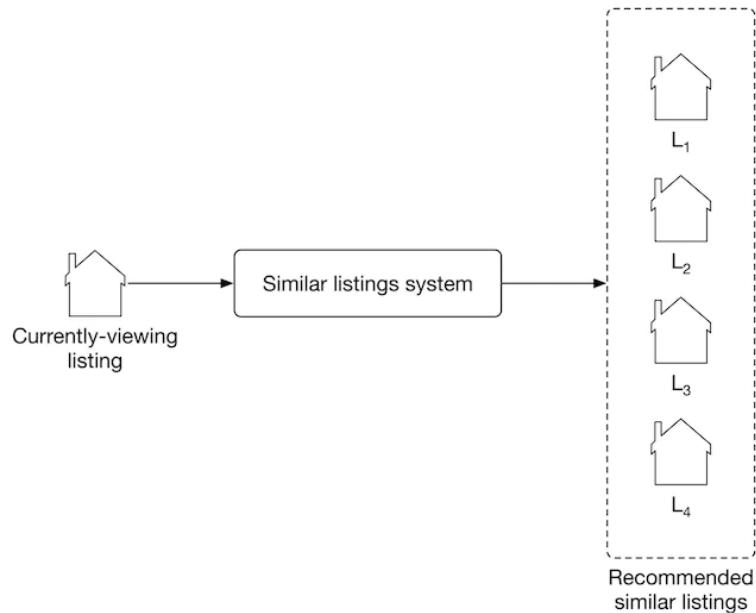


图 9.2: 相似房源系统的输入输出

9.2.3 选择合适的机器学习类别

大多数推荐系统依赖于用户的历史互动，以了解他们的长期兴趣。然而，这种推荐系统可能不适合解决相似房源问题。在我们的情况下，最近查看的房源比很久之前查看的房源更具信息价值。在这种情况下，通常使用基于会话的推荐系统。

像 Airbnb 这样的许多电子商务和旅行预订平台更依赖于短期兴趣进行推荐。在高质量推荐更依赖于最近互动而不是长期兴趣的系统中，基于会话的推荐通常是传统推荐系统的替代方案。基于会话的推荐系统根据用户的当前浏览会话进行推荐。现在，让我们更详细地了解基于会话的推荐系统。

9.2.4 基于会话的推荐系统

基于会话的推荐系统旨在根据用户浏览的一系列最近项目来预测下一个项目。在该系统中，用户的兴趣是上下文相关的，并且变化迅速。一个好的推荐很大程度上取决于用户最近的互动，而不是他们的泛兴趣。



图 9.3: 一个产品浏览会话

基于会话的推荐系统与传统推荐系统如何比较？ 在传统的推荐系统中，用户的兴趣是上下文无关的，不会频繁变化。而在基于会话的推荐中，用户的兴趣是动态的，并且变化迅速。传统推荐系统的目标是了解用户的泛兴趣。相比之下，基于会话的推荐系统旨在基于用户最近的浏览历史了解他们的短期兴趣。

一种广泛使用的构建基于会话的推荐系统的技术是使用用户浏览历史中项目的共现来学习项目的嵌入。例如，Instagram 学习账户嵌入来支持其“探索”功能 1，Airbnb 学习房源嵌入来支持其相似房源功能 2，word2vec 3 使用类似的方法来学习有意义的词嵌入。

在本章中，我们将“相似房源”问题框架化为一个基于会话的推荐任务。我们通过训练一个模型来构建系统，该模型将每个房源映射到一个嵌入向量中，以便如果两个房源在用户浏览历史中经常共现，它们的嵌入向量在嵌入空间中彼此接近。

为了推荐相似的房源，我们在嵌入空间中搜索与当前查看的房源最接近的房源。让我们来看一个例子。在图 9.4 中，每个房源都被映射到一个 2D 空间。为了向 L_t 推荐相似房源，我们选择嵌入最接近的前 3 个房源。

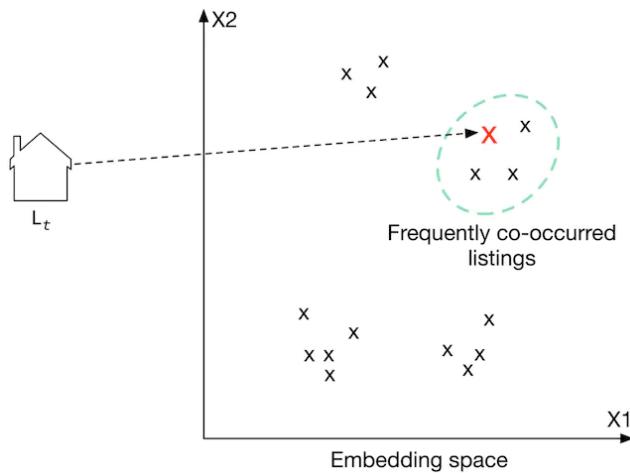


图 9.4: 嵌入空间中的相似房源

9.3 数据准备

9.3.1 数据工程

以下是可用的数据：

- 用户
- 房源
- 用户-房源交互

用户 下面展示了简化的用户数据模式。

ID	Username	Age	Gender	City	Country	Language	Time zone
1	Alice	29	Female	New York	USA	English	EST
2	Bob	35	Male	San Francisco	USA	English	PST

表 9.1: 用户数据模式

房源 房源数据包含与每个房源相关的属性，如价格、床位数、房主 ID 等。表 9.2 展示了房源数据的一个简单示例。

ID	Host ID	Price	Sq ft	Rate	Type	City	Beds	Max guests
1	135	135	1060	4.97	Entire place	NYC	3	4
2	81	80	830	4.6	Private room	SF	1	2
3	64	65	2540	5.0	Shared room	Boston	4	6

表 9.2: 房源数据

用户-房源交互 表 9.3 存储了用户与房源的交互信息，如展示次数、点击和预订。

ID	User ID	Listing ID	Position of the listing in the displayed list	Interaction type	Source	Timestamp
2	18	26	2	Click	Search feature	1655121925
3	5	18	5	Book	Similar listing feature	1655135257

表 9.3: 用户-房源交互数据

9.3.2 特征工程

如“将问题框架化为机器学习任务”部分所述，模型在训练过程中只利用用户的浏览历史。其他信息（如房源价格、用户年龄等）没有被使用。

在本章中，浏览历史被称为“搜索会话”。一个搜索会话是一个包含点击的房源 ID 序列，并最终以预订房源结束的序列，中间没有中断。图 9.5 展示了一个搜索会话的示例，其中用户的会话从点击 L_1 开始，最终预订了 L_{20} 。

在特征工程步骤中，我们从交互数据中提取搜索会话。表 9.4 显示了搜索会话的一个简单示例。

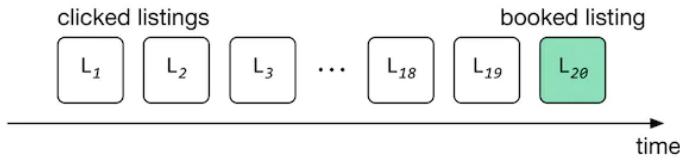


图 9.5: 一个搜索会话

Session ID	Clicked listing IDs	Eventually booked listing ID
1	1,5,4,9	26
2	6,8,9,21,6,13,6	5
3	5,9	11

表 9.4: 搜索会话数据

9.4 模型开发

9.4.1 模型选择

神经网络是学习嵌入的标准方法。选择一个好的神经网络架构取决于多个因素，比如任务的复杂性、训练数据量等。选择神经网络架构的超参数（如神经元数量、层数、激活函数等）的一种常见方法是进行实验，并选择表现最好的架构。在我们的案例中，我们选择使用浅层神经网络架构来学习房源嵌入。

9.4.2 模型训练

如图 9.6 所示，对于给定的输入房源，模型的任务是预测该输入房源上下文中的房源。

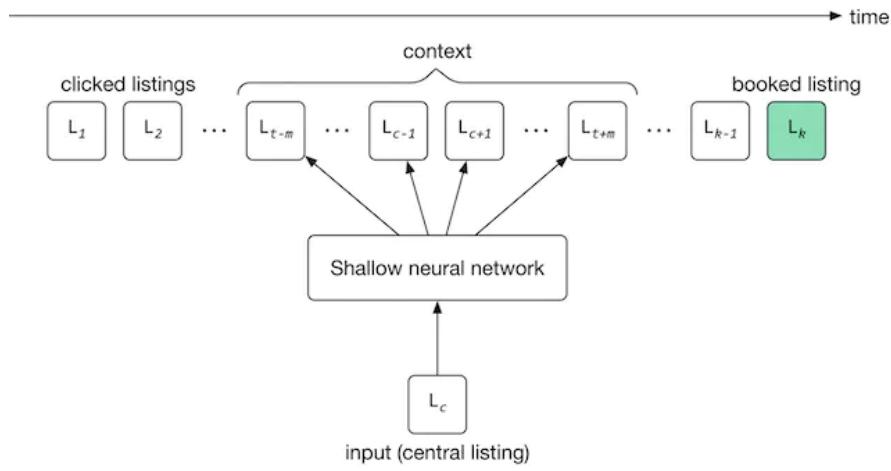


图 9.6: 预测相邻房源

训练过程从将房源嵌入初始化为随机向量开始。通过使用滑动窗口方法遍历搜索会话，逐渐学习这些嵌入。当窗口滑动时，窗口中间房源的嵌入会更新为与窗口内其他房源的嵌入相似，并与窗口外的房源嵌入不相似。然后，模型使用这些嵌入来预测给定房源的上下文。

为了适应新房源，我们每天对新构建的训练数据进行训练。

9.4.2.1 构建数据集

构建数据集有多种方法。在我们的案例中，我们选择一种称为“负采样 (negative sampling)”的技术⁴，常用于学习嵌入。

为了构建训练数据，我们从搜索会话中创建正样本对和负样本对。正样本对是预期具有相似嵌入的房源，而负样本对是预期具有不相似嵌入的房源。

更具体地说，对于每个会话，我们使用滑动窗口方法遍历房源。当窗口滑动时，我们使用窗口中间的房源及其上下文房源创建正样本对。我们使用窗口中间的房源和随机采样的房源形成负样本对。正样本对的真实标签为 1，负样本对的标签为 0。

图 9.7 展示了通过滑动窗口生成正样本对和负样本对的过程。

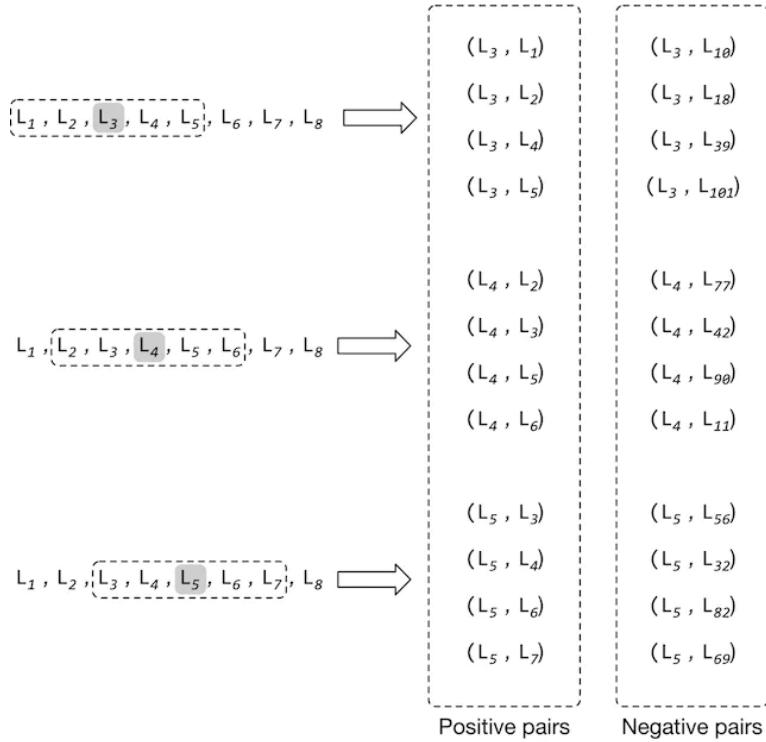


图 9.7: 构建的正样本对和负样本对

9.4.2.2 选择损失函数

损失函数用于衡量预测概率与真实标签之间的匹配度。如果两个房源构成正样本对，则它们的嵌入应接近；如果两个房源构成负样本对，则它们的嵌入应远离。更正式地，计算损失的步骤如下：

1. 计算两个嵌入之间的距离（例如点积）。
2. 使用 Sigmoid 函数将计算出的距离转换为 0 到 1 之间的概率值。
3. 使用交叉熵作为标准分类损失，测量预测概率与真实标签之间的损失。

图 9.8 展示了损失计算步骤。

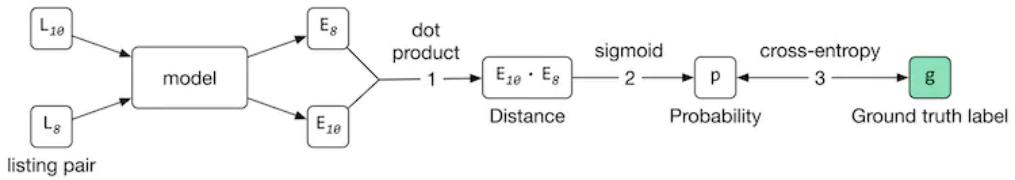


图 9.8: 损失计算步骤

损失函数可以通过以下公式表示：

$$\text{loss} = \sum_{(c,p) \in D_p} \log \frac{1}{1 + e^{-E_p \cdot E_c}} + \sum_{(c,n) \in D_n} \log \frac{1}{1 + e^{E_n \cdot E_c}}$$

其中：

- c 是一个中心房源， p 是一个正样本房源（在上下文中与 c 共现），而 n 是一个负样本房源（没有与 c 共现）。
- E_c 代表中心房源 c 的嵌入向量。
- E_n 代表负样本房源 n 的嵌入向量。
- E_p 代表正样本房源 p 的嵌入向量。
- D_p 是正样本对集合 $\langle c, p \rangle$ ，表示（中心房源，上下文房源）元组，其向量被推向彼此。
- D_n 是负样本对集合 $\langle c, n \rangle$ ，表示（中心房源，随机房源）元组，其向量被推远。

第一个求和计算正样本对的损失，第二个求和计算负样本对的损失。

我们可以改进损失函数以学习更好的嵌入吗？

前面描述的损失函数是一个很好的起点，但它有两个不足之处。首先，在训练过程中，中心房源的嵌入被推向其上下文中的嵌入，但没有被推向最终预订的房源的嵌入。这会导致模型生成的嵌入在预测相邻点击房源时效果很好，但在预测最终预订的房源时效果不佳。这对于帮助用户发现能促成预订的房源并不是最优的。

其次，前面生成的负样本对主要由不同地区的房源组成，因为它们是随机采样的。然而，用户通常只在某个特定区域内进行搜索，例如旧金山。这可能导致模型生成的嵌入在同一区域的房源中效果不佳，即那些虽然在上下文中没有共现，但来自同一区域的房源。

将最终预订的房源作为全局上下文 为了学习更好地预测最终预订房源的嵌入，我们在训练阶段将最终预订的房源视为全局上下文。当窗口滑动时，一些房源会进入或离开上下文集合，而最终预订的房源始终留在全局上下文中，并用于更新中心房源的向量。

为了在训练期间将最终预订的房源作为全局上下文，我们将〈中心房源，最终预订的房源〉对添加到我们的训练数据中，并将其标记为正样本。这促使模型在训练过程中将最终预订房源的嵌入推向会话中的每个点击房源的嵌入，如图 9.9 所示。

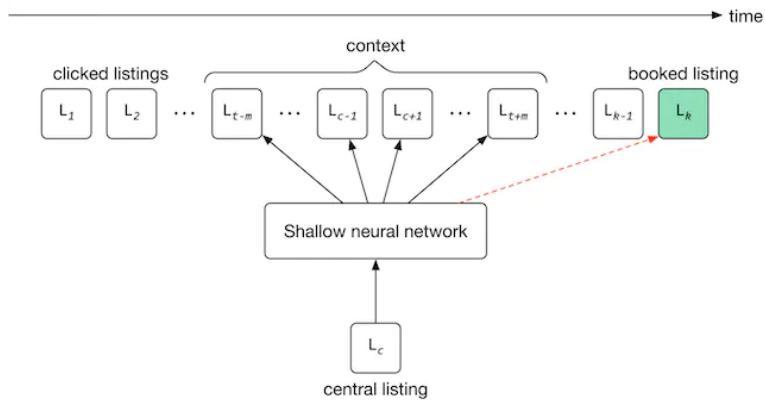


图 9.9: 将最终预订的房源添加到正样本对中

将同一区域的负样本对添加到训练数据中 当窗口滑动时，我们选择一个来自与中心房源相同社区的房源，但它不在中心房源的上下文中。我们将该对标记为负样本，并将其添加到我们的训练数据中。让我们来看一下考虑新添加的训练数据的更新损失函数。

$$\text{loss} = \sum_{(c,p) \in D_p} \log \frac{1}{1 + e^{-E_c \cdot E_p}} + \sum_{(c,n) \in D_n} \log \frac{1}{1 + e^{E_c \cdot E_n}} + \\ \sum_{(c,b) \in D_{\text{booked}}} \log \frac{1}{1 + e^{-E_c \cdot E_b}} + \sum_{(c,n) \in D_{\text{hard}}} \log \frac{1}{1 + e^{E_c \cdot E_n}}$$

其中：

- E_b 代表最终预订房源 b 的嵌入向量。
- D_{booked} 是 $\langle c, b \rangle$ 对，表示（中心房源，预订房源）元组，其向量被推向彼此。
- D_{hard} 是难负样本对 $\langle c, n \rangle$ ，表示（中心房源，同区域负样本房源）元组，其向量被推远。

前两个求和之前已经解释过。第三个求和计算包含全局上下文的新增正样本对的损失。它有助于模型将中心房源的嵌入推向最终预订房源的嵌入。

第四个求和计算来自同一区域的新负样本对的损失。它要求模型将这些嵌入彼此推远。

9.5 评估

9.5.1 离线指标

在模型开发阶段，我们使用离线指标来衡量模型输出的质量，并将新开发的模型与旧模型进行比较。评估学习到的嵌入的一种方法是测试它们在基于用户的最新点击预测最终预订房源时的效果。让我们创建一个名为“最终预订房源的平均排名”的指标，并详细讨论这一点。

最终预订房源的平均排名。 我们通过一个例子来理解这个指标。图 9.10 显示了一个用户的搜索会话。可以看到，搜索会话总共包括七个房源。第一个房源是用户首先查看的 (L_0)。接下来的五个是用户按顺序点击的房源。最后一个 (L_6) 是用户最终预订的房源。

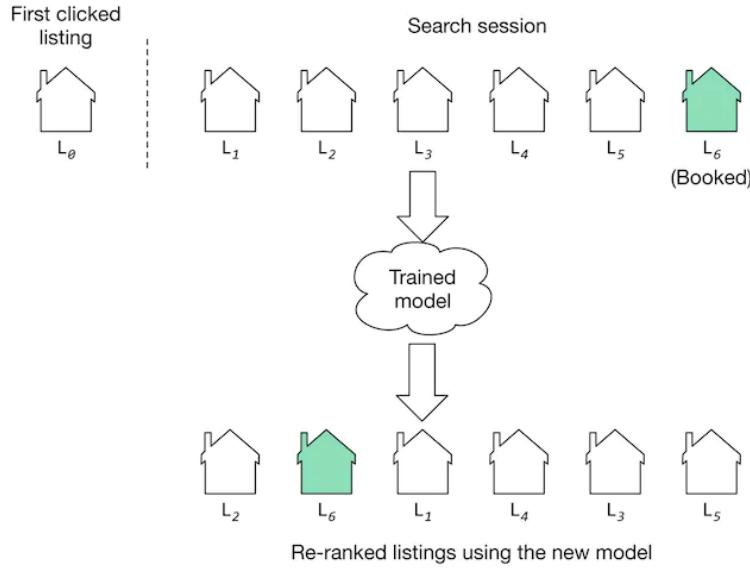


图 9.10: 由模型重新排名的会话

我们使用模型来计算第一个点击的房源与嵌入空间中其他房源之间的相似度。相似度计算完成后，对房源进行排名。最终预订房源的位置表明我们使用新模型时能将其推荐到多高的位置。如图 9.10 所示，新模型（第二行）能够将最终预订房源 (L6) 排在第二位。

如果模型能够高排名地推荐最终预订房源，则表明学习到的嵌入可以更早地将最终预订房源推荐到用户面前。我们通过对验证数据集中所有会话的最终预订房源排名进行平均，来计算此指标的值。

9.5.2 在线指标

根据需求，业务目标是增加预订数量。以下是一些在线指标选项：

- 点击率 (Click-through rate, CTR)
- 会话预订率 Session book rate

点击率。这是一个比例，用于显示看到推荐房源的人有多大概率最终点击它们。

$$\text{CTR} = \frac{\text{点击的房源数量}}{\text{推荐的房源数量}}$$

该指标用于衡量用户参与度。例如，当用户更频繁地点击房源时，被点击的房源有更高的可能性成为预订。但由于 CTR 并未衡量平台上实际完成的预订数量，我们使用“会话预订率”指标来补充 CTR。

会话预订率 (Session book rate)。这是一个比例，用于显示有多少搜索会话最终转化为预订。

$$\text{会话预订率} = \frac{\text{转化为预订的会话数量}}{\text{总会话数量}}$$

此指标与我们的业务目标——增加预订数量——直接相关。“会话预订率”越高，平台产生的收入就越多。

9.6 服务

在服务阶段，系统向用户推荐与其当前查看的房源相似的房源。图 9.11 显示了 ML 系统设计的概览。

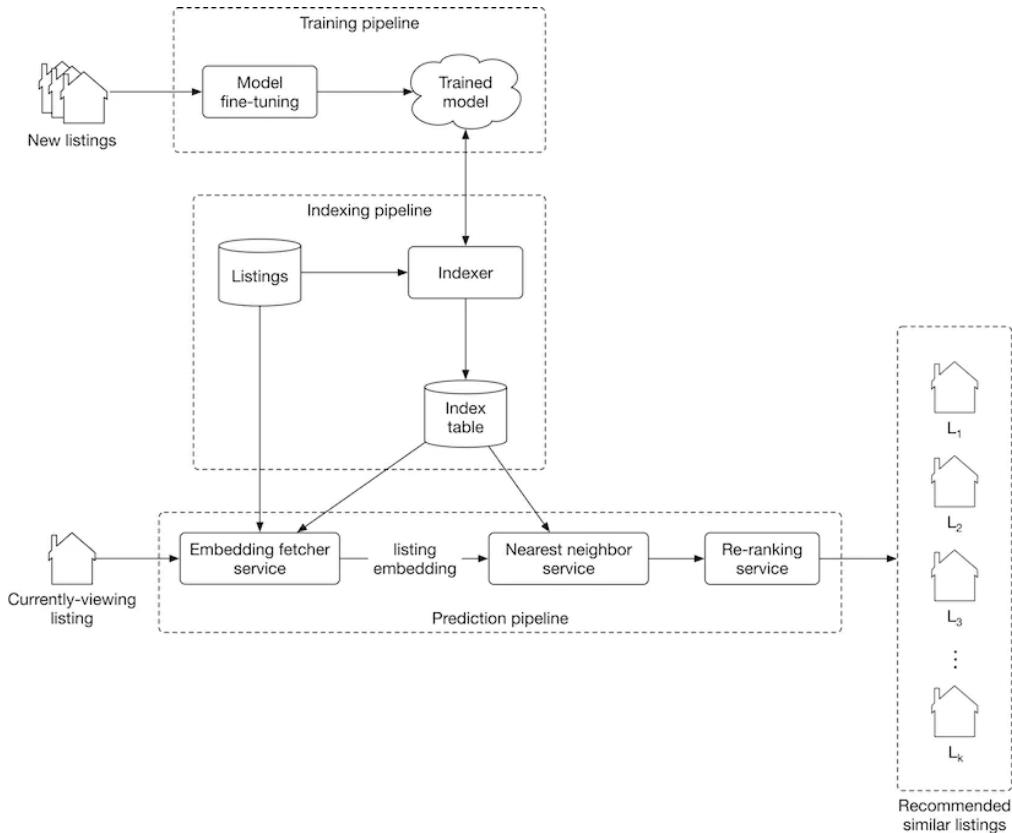


图 9.11: ML 系统设计

让我们详细分析主要组件。

9.6.1 训练 Pipeline

训练 Pipeline 使用新房源和用户-房源交互数据对模型进行微调。这确保了模型始终适应新的交互和房源。

9.6.2 索引 Pipeline

使用训练好的模型，可以预先计算平台上所有房源的嵌入并存储在索引表中。这显著加快了预测 Pipeline 的速度。

索引 Pipeline 负责创建和维护索引表。例如，当一个新房源的嵌入可用时，Pipeline 会将其嵌入添加到索引表中。此外，当新训练的模型可用时，Pipeline 会使用新模型重新计算所有嵌入，并更新索引表。

9.6.3 预测 Pipeline

预测 Pipeline 推荐与用户当前查看的房源相似的房源。预测 Pipeline，如图 9.11 所示，包括：

- 嵌入获取服务
- 最近邻服务
- 重排名服务

让我们检查每个组件。

嵌入获取服务 该服务将用户当前查看的房源作为输入，并根据该房源在训练期间是否被模型见过来采取不同的操作。

输入房源在训练期间被模型见过

如果房源在训练期间被见过，它的嵌入向量已经被学习，并且可以在索引表中找到。在这种情况下，嵌入获取服务直接从索引表中获取房源嵌入。

输入房源在训练期间未被模型见过

如果输入房源是新的，模型在训练期间未见过。这是个问题，因为如果我们没有给定房源的嵌入，就无法找到相似的房源。

为了解决这个问题，嵌入获取服务使用启发式方法处理新房源。例如，当房源是新的时，我们可以使用地理上临近的房源的嵌入。当为新房源收集到足够的交互数据时，训练 Pipeline 通过微调模型来学习其嵌入。

最近邻服务 为了推荐相似房源，我们需要计算当前查看的房源嵌入与平台上其他房源嵌入之间的相似度。这就是最近邻服务的作用。该服务计算这些相似度，并输出嵌入空间中最近邻的房源。

记住，我们的平台上有 500 万个房源。计算如此多房源的相似度需要时间，可能会减慢服务速度。因此，我们使用近似最近邻方法来加速搜索。

重排名服务 该服务通过应用用户过滤条件和某些约束来调整推荐的房源列表。例如，如果房源的价格高于用户设置的价格过滤条件，这一层会将其移除。此外，来自当前查看房源所在城市以外的房源在展示给用户之前也会被移除。

9.7 其他讨论点

如果在面试结束时还有时间，可以讨论以下一些额外话题：

- 什么是位置偏差，如何解决它 5。
- 会话式推荐方法与随机游走有何不同 6，以及如何使用带重启的随机游走 (RWR) 来推荐相似房源 7。
- 如何通过考虑用户的长期兴趣（会话内个性化）来个性化会话式推荐系统的结果 2。
- 考虑到季节性对度假租赁的影响很大，如何将季节性因素纳入相似房源系统中 8。

References

1. Instagram's Explore recommender system. <https://ai.facebook.com/blog/powerd-by-ai-instagrams-explore-recommender-system/>
2. Listing embeddings in search ranking. <https://medium.com/airbnb-engineering/listing-embeddings-for-search-ranking-103a2a2a3a1>
3. Word2vec. <https://en.wikipedia.org/wiki/Word2vec>.
4. Negative sampling technique. <https://www.baeldung.com/cs/nlp-word2vec-negative-sampling>.
5. Positional bias. <https://eugeneyan.com/writing/position-bias/>.
6. Random walk. https://en.wikipedia.org/wiki/Random_walk.
7. Random walk with restarts. https://www.youtube.com/watch?v=HbzQzUaJ_9I.
8. Seasonality in recommendation systems. <https://www.computer.org/csdl/proceedings/article/big-data/2019/09005954/1hJsfgT0qL6>.

10 Personalized News Feed 个性化新闻推送

10.1 Introduction

新闻推送是社交网络平台的一项功能，通过在用户的时间线上显示朋友的最新动态来保持用户的参与度。大多数社交网络，如 Facebook 1、Twitter 2 和 LinkedIn 3，都通过个性化新闻推送来维持用户的参与度。

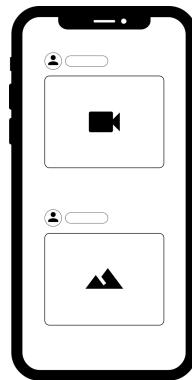


图 10.1: 用户时间线上的个性化推送

在本章中，我们需要设计一个个性化新闻推送系统。

10.2 明确需求

以下是候选人与面试官之间的典型对话。

候选人：我可以假设个性化新闻推送的目的是为了让用户保持对平台的参与度吗？

面试官：是的，我们在帖子之间展示广告，更多的参与度意味着更多的收入。

候选人：当用户刷新时间线时，我们会向用户显示包含新动态的帖子。我可以假设这些动态包括未读的帖子和未读评论的帖子吗？

面试官：这是一个合理的假设。

候选人：一个帖子可以包含文本内容、图片、视频或任何组合吗？

面试官：它可以是任何组合。

候选人：为了让用户保持参与度，系统应该将最具吸引力的内容放在时间线的顶部，因为人们更有可能与前几条帖子互动。这听起来对吗？

面试官：是的，正确。

候选人：我们是否优化某种特定类型的互动？我认为互动有不同类型，比如点击、点赞和分享。

面试官：这个问题很好。不同的互动在我们的平台上具有不同的价值。例如，点赞比单纯点击更有价值。理想情况下，我们的系统在对帖子进行排序时应考虑主要的互动类型。接下来，我会让你来定义“互动”并选择你的模型应该优化的内容。

候选人：平台上有哪些主要的互动类型？我假设用户可以点击、点赞、分享、评论、隐藏、屏蔽其他用户以及发送好友请求。还有其他互动类型需要考虑吗？

面试官：你提到的都是主要的互动类型。我们就关注这些吧。

候选人：系统需要多快运行？

面试官：我们期望系统在用户刷新时间线或打开应用程序后，快速展示排序后的帖子。如果时间过长，用户会感到无聊并离开。假设系统需要在 200 毫秒内展示排序后的帖子。

候选人：我们有多少日活跃用户？我们每天预计有多少次时间线更新？

面试官：我们总共有大约 30 亿用户，其中约 20 亿是日活跃用户，每天查看他们的推送两次。

让我们总结一下问题陈述。我们需要设计一个个性化新闻推送系统。系统获取未读帖子或包含未读评论的帖子，并根据它们对用户的吸引力进行排序。这个过程应在 200 毫秒内完成。系统的目标是提高用户参与度。

10.3 将问题框架化为机器学习任务

10.3.1 定义机器学习目标

让我们研究以下三种可能的机器学习目标：

- 最大化特定隐式互动的数量，例如停留时间或用户点击
- 最大化特定显式互动的数量，例如点赞或分享
- 基于隐式和显式互动的加权得分最大化

让我们更详细地讨论每个选项。

选项 1：最大化特定隐式互动的数量，例如停留时间或用户点击。在此选项中，我们选择隐式信号作为用户参与度的代理。例如，我们优化机器学习系统以最大化用户点击量。

优点是我们拥有比显式互动更多的隐式互动数据。更多的训练数据通常会带来更准确的模型。

缺点是隐式互动不一定总能反映用户对帖子的真实看法。例如，用户可能点击了一篇帖子，但觉得它不值得阅读。

选项 2：最大化特定显式互动的数量，例如点赞、分享和隐藏。

在此选项中，我们选择显式互动作为用户对帖子的意见的代理。

该方法的优点在于显式信号通常比隐式信号更有分量。例如，用户点赞一篇帖子比他们只是点击更能传达强烈的参与信号。

主要缺点是很少有用户会通过显式互动表达他们的看法。例如，用户可能觉得一篇帖子很有吸引力，但不对其做出反应。在这种情况下，模型很难在训练数据有限的情况下做出准确的预测。

选项 3：基于隐式和显式互动的加权得分最大化。

在此选项中，我们同时使用隐式和显式互动来确定用户对帖子的参与度。特别地，我们根据每种互动对我们的价值为其分配一个权重。然后我们优化机器学习系统以最大化互动的加权得分。

表 10.1 显示了不同互动与权重之间的映射。正如你所看到的，按下“点赞”按钮的权重比点击更高，而分享比点赞更有价值。此外，负面互动（如隐藏和屏蔽）具有负权重。请注意，这些权重可以根据业务需求进行选择。

选择哪个选项？

我们选择最终的混合选项，因为它允许我们为不同的互动分配不同的权重。这很重要，因为我们可以根据业务的重点来优化系统。

Reaction	Click	Like	Comment	Share	Friendship request	Hide	Block
Weight	1	5	10	20	30	-20	-50

表 10.1: 不同互动的权重

10.3.2 指定系统的输入和输出

如图 10.2 所示，个性化新闻推送系统以用户作为输入，输出一个未读帖子或包含未读评论的帖子列表，并按参与度得分进行排序。

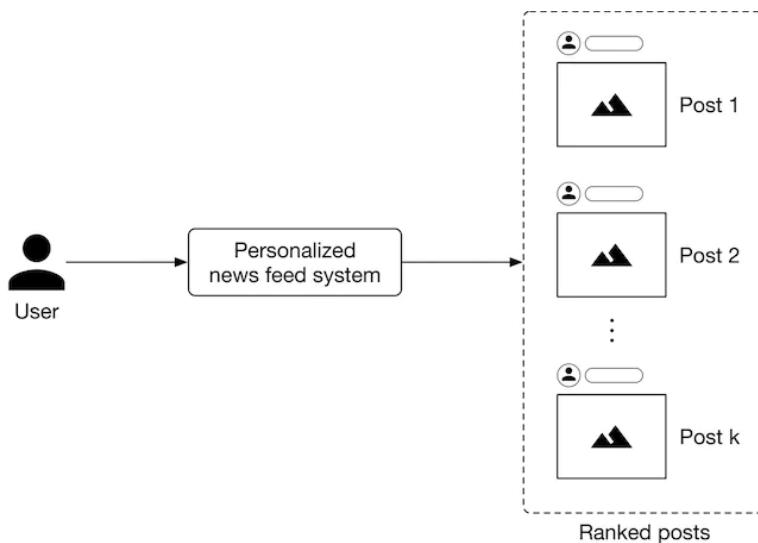


图 10.2: 个性化新闻推送系统的输入输出

10.3.3 选择合适的机器学习类别

个性化新闻推送系统根据帖子对用户的吸引力生成排序的帖子列表。点对点 LTR (Pointwise Learning to Rank) 7.2.3.1 是一种简单但有效的方法，通过基于参与度得分对帖子进行排序来个性化新闻推送。为了了解如何计算用户与帖子之间的参与度得分，让我们来看一个具体的例子。

如图 10.2 所示，我们使用多个二元分类器来预测 $\langle \text{user}, \text{post} \rangle$ 对的各种隐式和显式互动的概率。

Reaction	Click	Like	Comment	Share	Friendship request	Hide	Block
Predicted probability	23%	48%	12%	4%	0.1%	0.005%	0.0003%
Value	1	5	10	20	30	-20	-50

表 10.2: Reaction probabilities and values

一旦预测出这些概率，我们就计算参与度得分。表 10.3 显示了如何计算参与度得分的示例。

Reaction	Click	Like	Comment	Share	Friendship request	Hide	Block
Predicted probability	23%	48%	12%	4%	0.1%	0.005%	0.0003%
Value	1	5	10	20	30	-20	-50
Score	0.23	2.4	1.2	0.8	0.03	-0.001	-0.00015
Engagement score = 4.65885							

表 10.3: 计算参与度得分

10.4 数据准备

10.4.1 数据工程

在设计预测特征之前，了解系统中可用的原始数据通常是有帮助的。在这里，我们假设系统中提供以下几类原始数据：

- 用户
- 帖子
- 用户-帖子互动
- 朋友关系

10.4.1.1 用户

用户数据的模式如下所示。

ID	Username	Age	Gender	City	Country	Language	Time zone
----	----------	-----	--------	------	---------	----------	-----------

表 10.4: 用户数据模式

10.4.1.2 帖子

Author ID	Textual Content	Hashtags	Mentions	Images or videos	Timestamp
5	Today at our fav place with my best friend	life_is_good, happy	hs2008		1658450539
1	It was the best trip we ever had	Travel, Maldives	Alexish, shan.tony	htcdn.mysite.com/maldives.jpg	1658451341
29	Today I had a bad experience I would like to tell you about. I went...				1658451365

表 10.5: 帖子数据

User ID	Post ID	Interaction type	Interaction value	Location (lat, long)	Timestamp
4	18	Like		38.8951 -77.0364	1658450539
4	18	Share	User 9	41.9241 -89.0389	1658451365
9	18	Comment	You look amazing	22.7531 47.9642	1658435948
9	18	Block		22.7531 47.9642	1658451849
6	9	Impression		37.5189 122.6405	1658821820

表 10.6: 用户-帖子互动数据

10.4.1.3 用户-帖子互动

10.4.1.4 朋友关系

朋友关系表存储用户之间的连接数据。我们假设用户可以指定他们的亲密朋友和家人。表 10.7 显示了朋友关系数据的示例。

User ID 1	User ID 2	Time when friendship was formed	Close friend	Family member
28	3	1558451341	True	False
7	39	1559281720	False	True
11	25	1559312942	False	False

表 10.7: 朋友关系数据

10.4.2 特征工程

在本节中，我们将预测特征工程化并为模型准备这些特征。我们特别从以下几类中生成特征：

- 帖子特征
- 用户特征
- 用户与作者的关联性

10.4.2.1 帖子特征

在实践中，每个帖子都有很多属性。我们无法涵盖所有内容，因此只讨论最重要的特征。

- 文本内容
- 图片或视频
- 互动
- 话题标签
- 帖子的时效性

文本内容 是什么？这是帖子的文本内容，即主体部分。

为什么重要？ 文本内容有助于确定帖子的主题。

如何准备？ 我们对文本内容进行预处理，并使用预训练语言模型将文本转换为数值向量。由于文本内容通常是句子形式而不是单个单词，因此我们使用上下文感知的语言模型，如 BERT 4。

图片或视频 是什么？帖子中可能包含图片或视频。

为什么重要？ 我们可以从图片中提取重要的信号。例如，枪支的图片可能表明帖子不适合儿童观看。

如何准备？ 首先，对图片或视频进行预处理。接下来，使用预训练模型将非结构化的图片/视频数据转换为嵌入向量。例如，我们可以使用 ResNet 5 或最近推出的 CLIP 模型 6 作为预训练模型。

互动 是什么？这指的是帖子的点赞数、分享数、回复数等。

为什么重要？ 点赞、分享、隐藏等数量表明用户对帖子的参与度。用户更可能与数千个点赞的帖子互动，而不是与只有十个点赞的帖子互动。

如何准备？ 这些值以数值表示。我们将这些数值进行缩放，以使其在相似范围内。

话题标签 **为什么重要？** 用户使用话题标签将内容聚合到某个主题下。这些话题标签表示帖子所涉及的主题。例如，带有“#women_in_tech”话题标签的帖子表示内容与科技和女性相关，因此模型可能决定为对科技感兴趣的人更高地排序。

如何准备？ 文本预处理的详细步骤已在第 4 章《YouTube 视频搜索》中解释，这里我们只关注为话题标签准备的独特步骤。

- **分词：**话题标签如“lifeisgood”或“programmer_lifestyle”包含多个单词。我们使用如 Viterbi 7 的算法对话题标签进行分词。例如，“lifeisgood”会变成三个单词：“life”“is”“good”。
- **标记到 ID：**话题标签在社交媒体平台上发展迅速，并随着趋势的变化而变化。特征哈希技术是一个不错的选择，因为它能够为未见过的话题标签分配索引。
- **向量化：**我们使用简单的文本表示方法，如 TF-IDF 8 或 word2vec 9，而不是基于 Transformer 的模型来向量化话题标签。让我们来看看原因。Transformer 模型在数据的上下文至关重要时很有用。在话题标签的情况下，每个话题标签通常是一个单词或短语，通常不需要上下文来理解其含义。因此，首选更快且更轻量的文本表示方法。

帖子的时效性 是什么？此特征显示了作者发布内容后经过的时间。

为什么重要？ 用户往往更倾向于与较新的内容互动。

如何准备？ 我们将帖子的时效性分为几个类别，并使用 one-hot 编码进行表示。例如，我们使用以下分类：

- 0: 少于 1 小时
- 1: 1 到 5 小时之间
- 2: 5 到 24 小时之间
- 3: 1 到 7 天之间

- 4: 7 到 30 天之间
- 5: 超过一个月

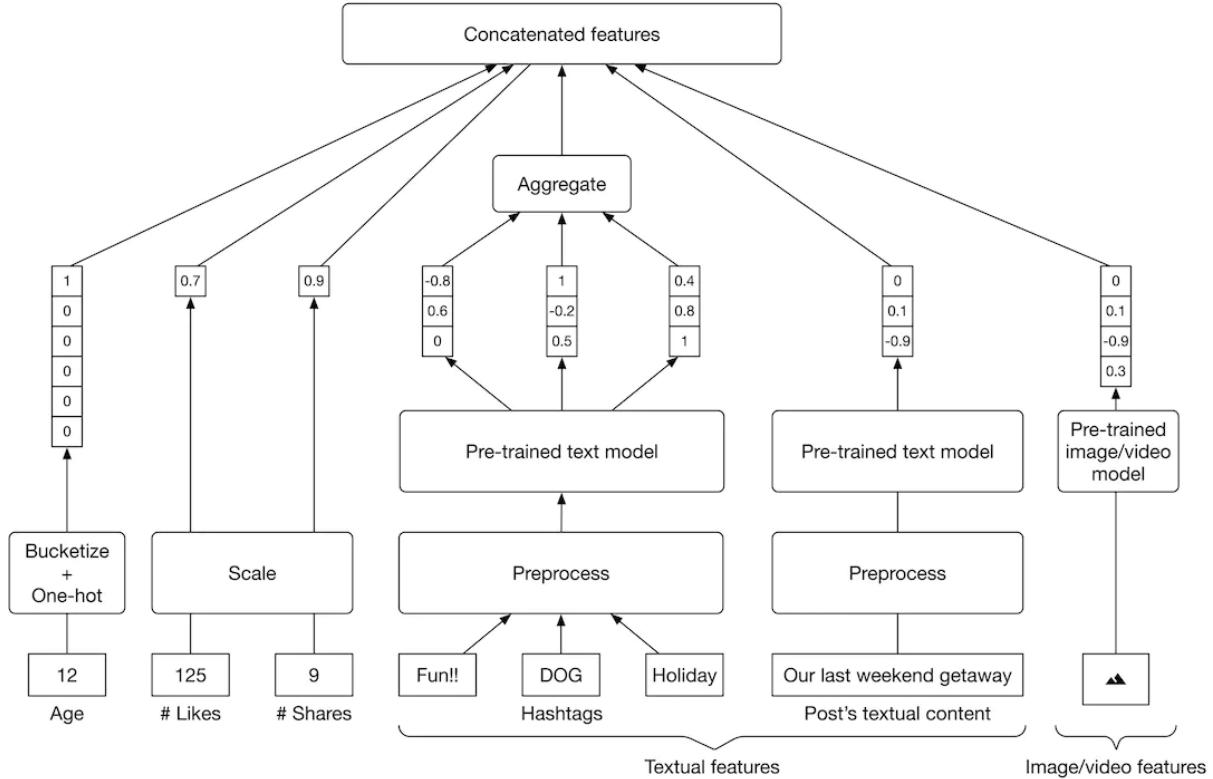


图 10.3: 与帖子相关的特征准备

10.4.2.2 用户特征

一些最重要的用户相关特征是：

- 人口统计信息：年龄、性别、国家等
- 上下文信息：设备、一天中的时间等
- 用户-帖子历史互动
- 在帖子中被提及

由于我们在前几章中已经讨论了用户的人口统计信息和上下文信息，这里我们只分析剩下的两个特征。

10.4.2.3 用户-帖子历史互动

用户点赞的所有帖子都由一个帖子 ID 列表表示。同样的逻辑适用于分享和评论。

为什么重要？ 用户之前的互动通常有助于预测他们未来的互动。

如何准备？ 从用户互动过的每个帖子中提取特征。

10.4.2.4 在帖子中被提及

是什么？这意味着用户是否在帖子中被提及。

为什么重要？用户通常会更关注提到他们的帖子。

如何准备？该特征由一个二元值表示。如果用户在帖子中被提及，则该特征为 1，否则为 0。

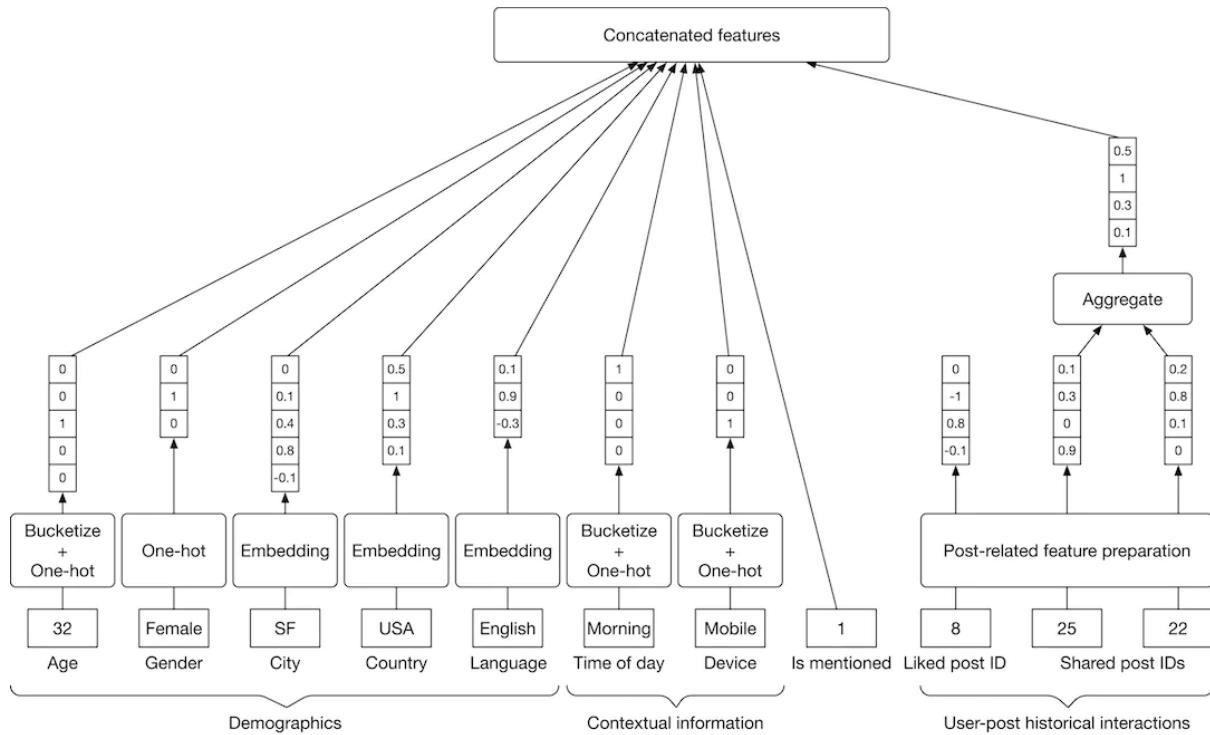


图 10.4: 用户相关数据的特征准备

10.4.2.5 用户-作者的关联性

根据研究，关联性特征，如用户与作者之间的连接，是预测用户在 Facebook 上参与度的最重要因素之一 [10](#)。让我们为捕捉用户-作者关联性生成一些特征。

点赞/点击/评论/分享率 这是用户对某个作者的帖子做出反应的比例。例如，点赞率为 0.95 表示用户对该作者帖子 95% 的时间都会点赞。

朋友关系时长 用户与作者在平台上成为朋友的天数。该特征可以从朋友关系数据中获得。

为什么重要？用户往往更愿意与他们的朋友互动。

亲密朋友和家人 一个二元值，表示用户与作者是否将对方包含在其亲密朋友和家人列表中。

为什么重要？用户对亲密朋友和家人的帖子更加关注。

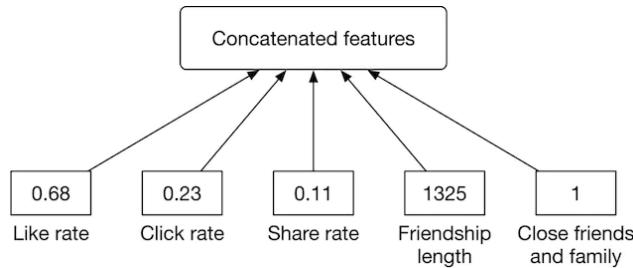


图 10.5: 用户-作者关联性特征

10.5 模型开发

10.5.1 模型选择

我们选择神经网络的原因如下：

- 神经网络在处理文本和图像等非结构化数据时效果良好。
- 神经网络允许我们使用嵌入层来表示类别特征。
- 使用神经网络架构时，我们可以对特征工程过程中使用的预训练模型进行微调。这在其他模型中是不可能的。

在训练神经网络之前，我们需要选择其架构。有两种神经网络架构可供选择：

- N 个独立的 DNN
- 多任务 DNN

让我们来探讨每个选项。

选项 1：N 个独立的 DNN

在此选项中，我们使用 N 个独立的深度神经网络（DNN），每个反应类型一个。这在图 10.6 中有所展示。

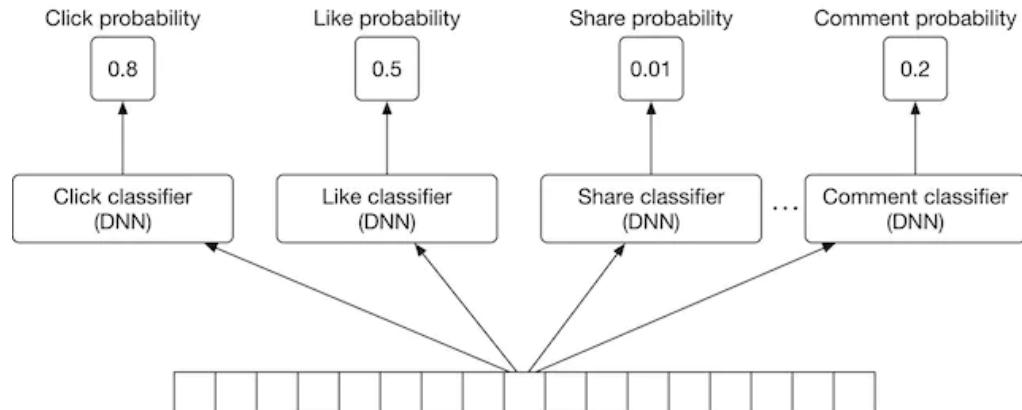


图 10.6: 使用独立的 DNN

此选项有两个缺点：

- **训练代价高。**训练多个独立的 DNN 计算密集且耗时。
- **对于不常见的反应，可能没有足够的训练数据。**这意味着我们的系统无法准确预测不常见的反应的概率。

选项 2：多任务 DNN

为了解决这些问题，我们使用多任务学习方法（图 10.7）。

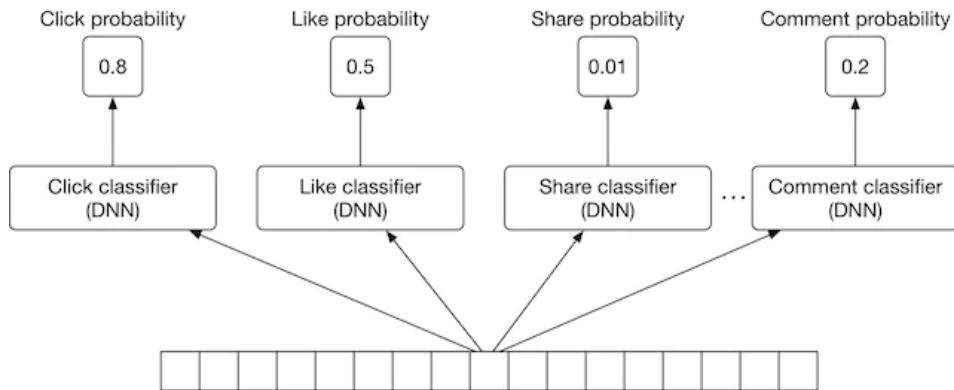


图 10.7: 多任务 DNN

我们在第 5 章《有害内容检测》中解释了多任务学习，这里只做简要讨论。总结来说，多任务学习是指同时学习多个任务的过程。这样可以让模型学习任务之间的相似性，避免不必要的计算。对于多任务神经网络模型，选择合适的架构至关重要。架构的选择和相关超参数通常通过实验确定。这意味着在不同的架构上进行模型训练和评估，选择表现最好的那个。

针对被动用户改进 DNN 架构 到目前为止，我们使用 DNN 来预测分享、点赞、点击和评论等反应。然而，许多用户是被动使用平台的，这意味着他们很少与时间线上内容互动。对于这些用户，当前的 DNN 模型将对所有反应预测非常低的概率，因为他们很少对帖子做出反应。因此，我们需要改变 DNN 架构以考虑到被动用户。

为此，我们在任务列表中添加了两个隐性反应：

- 停留时间：用户在某个帖子上停留的时间。
- 跳过：如果用户在某个帖子上停留的时间少于 t 秒（例如 0.5 秒），则可以认为用户跳过了该帖子。

图 10.8 显示了包含额外任务的多任务 DNN 模型。

10.5.2 模型训练

10.5.2.1 构建数据集

在此步骤中，我们从原始数据中构建数据集。由于 DNN 模型需要学习多个任务，我们为每个任务（如点击、点赞等）创建正负数据点。

我们将使用“点赞”反应类型作为示例，解释如何创建正/负数据点。每当用户点赞一个帖子时，我们会在数据集中添加一个数据点，计算〈用户，帖子〉特征，然后标记为正样本。

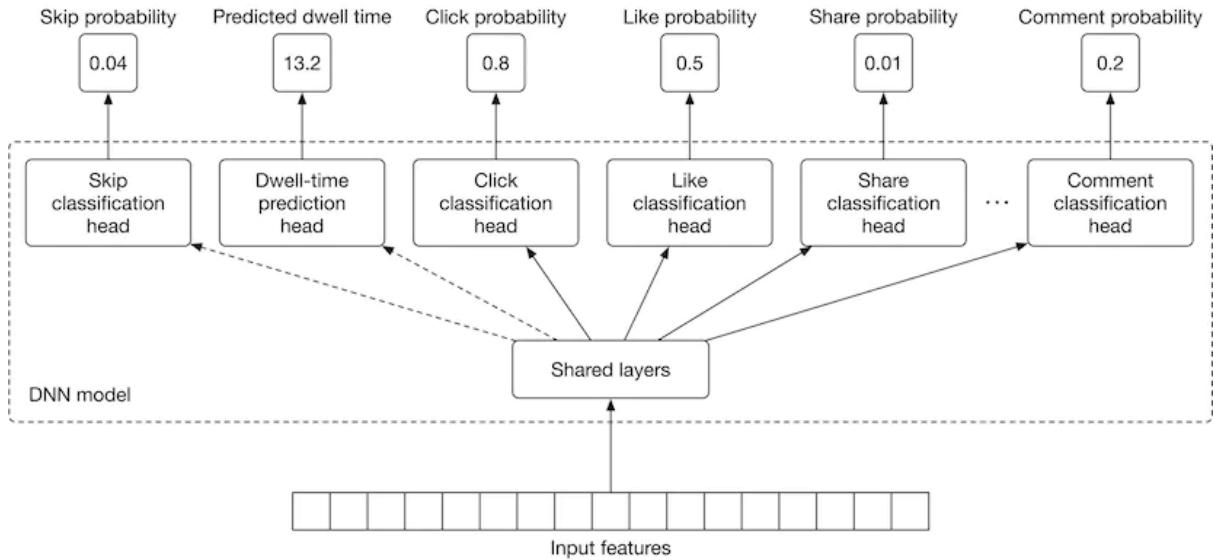


图 10.8: 包含两个新任务的多任务 DNN 模型

要创建负样本，我们选择未导致“点赞”反应的展示次数。请注意，负样本的数量通常远高于正样本。为了避免数据集不平衡，我们创建与正样本数量相等的负样本。图 10.9 显示了“点赞”反应的正负数据点。

#	User features	Post features	Affinity features	Label
1	1 0 1 0.8 0.1 1	0 1 1 0.4 0	0.9 0.6 0.3 8 0	Positive
2	0 0 0 0.4 0.9 0	1 1 0 0.3 1	1 0.9 0.8 120 1	Positive
3	1 1 0 0.1 0.5 0	0 1 0 0.9 1	0.1 0 0 2 0	Negative

图 10.9: 点赞分类任务的训练数据

相同的过程可以用来为其他反应创建正负标签。然而，由于停留时间是一个回归任务，我们需要以不同的方式构建它。如图 10.10 所示，真实标签是展示的停留时间。

#	User features	Post features	Affinity features	Dwell-time
1	0 0 0 0.1 0.9 1	1 1 0 0.1 1	0.6 0.6 0.3 0.2 5 0	8.1
2	1 1 1 0.9 0.1 0	1 1 0 0.8 0	0.1 0.9 0.3 0.1 3 1	11.5

图 10.10: 停留时间任务的训练数据

10.5.2.2 选择损失函数

多任务模型旨在同时学习多个任务。这意味着我们需要分别计算每个任务的损失，然后将它们组合以得到整体损失。通常，我们根据任务的机器学习类别为每个任务定义一个损失函数。在我们的情况下，我们对每个二元分类任务使用二元交叉熵损失，对于回归任务（停留时间预测）使用回归损失，如 MAE 11、MSE 12 或 Huber 损失 13。整体损失通过组合特定任务的损失来计算，如图 10.11 所示。

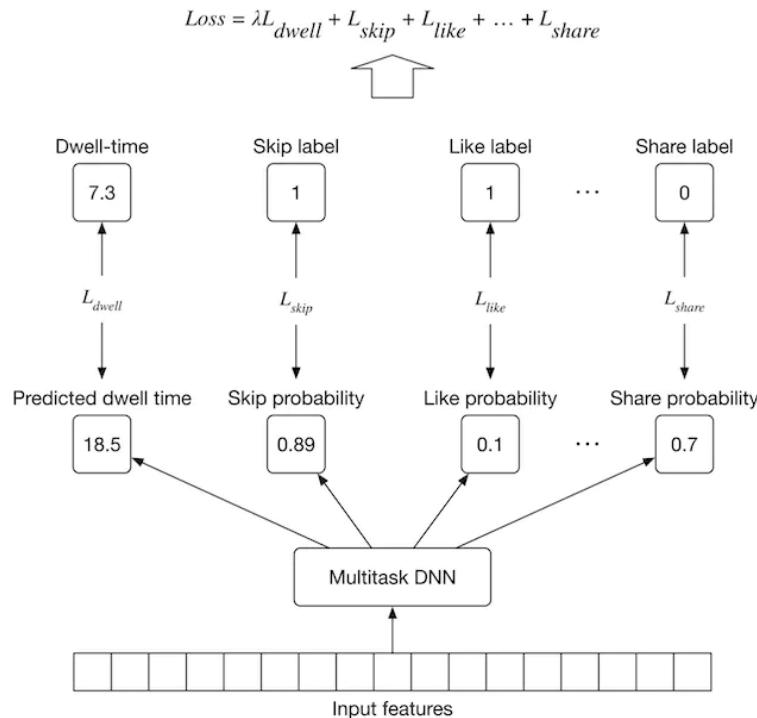


图 10.11: 训练流程

10.6 评估

10.6.1 离线指标

在离线评估中，我们衡量模型在预测不同反应类型上的表现。为了评估单一类型反应的表现，可以使用二元分类指标，如精度和召回率。然而，仅靠这些指标可能不足以全面理解二元分类模型的表现。因此，我们使用 ROC 曲线来理解真正例率与假正例率之间的权衡。此外，我们计算 ROC 曲线下的面积 (ROC-AUC)，以数值形式总结二元分类的表现。

10.6.2 在线指标

我们使用以下指标从不同角度衡量用户参与度：

- 点击率 (CTR)
- 反应率

- 总停留时间
- 用户调查中的满意度

点击率 (CTR) 指的是点击次数与展示次数的比率。

$$\text{CTR} = \frac{\text{点击的帖子数量}}{\text{展示次数}}$$

高 CTR 并不总是意味着更高的用户参与度。例如，用户可能点击一个低价值的标题党帖子，并迅速意识到它不值得阅读。尽管有这个限制，它仍然是一个重要的跟踪指标。

反应率 是一组反映用户反应的指标。例如，点赞率衡量的是用户点赞的帖子数量与展示在用户动态中的帖子总数之间的比率。

$$\text{点赞率} = \frac{\text{点赞的帖子数量}}{\text{展示次数}}$$

类似地，我们跟踪其他反应，如“分享率”、“评论率”、“隐藏率”、“屏蔽率”和“跳过率”。这些比 CTR 更强的信号，因为用户明确表达了他们的偏好。

到目前为止，我们讨论的指标基于用户的反应。但对于被动用户呢？这些用户往往对大多数帖子没有反应。为了捕捉个性化新闻推荐系统对被动用户的效果，我们增加了以下两个指标。

总停留时间 指的是用户在一段固定时间内（如一周）在动态页面上的总停留时间。这个指标衡量了被动和活跃用户的整体参与度。

用户调查中的满意度 另一种衡量个性化新闻推荐系统有效性的方法是明确询问用户对动态的看法，或他们对帖子的兴趣程度。由于我们寻求的是明确的反馈，这是一种准确衡量系统有效性的方法。

10.7 服务

在服务时，系统通过输出一个排序的帖子列表来响应请求。图 10.12 展示了个性化新闻推荐系统的架构图。该系统包含以下 Pipeline：

- 数据准备 Pipeline
- 预测 Pipeline

我们不会详细讨论数据准备 Pipeline，因为它与第 8 章《社交平台中的广告点击预测》中描述的非常相似。让我们来探讨预测 Pipeline。

预测 Pipeline

预测 Pipeline 由以下组件组成：检索服务、排序服务和重新排序服务。

检索服务。该组件检索用户尚未看到的帖子，或者用户尚未看到评论的帖子。要了解有关高效获取未见帖子的信息，请阅读 14。

排序服务。该组件通过为每个帖子分配参与度分数来对检索到的帖子进行排序。

重新排序服务。该服务通过加入额外的逻辑和用户过滤器来修改帖子列表。例如，如果用户明确表示对某个主题（如足球）感兴趣，该服务会为相关帖子分配更高的排名。

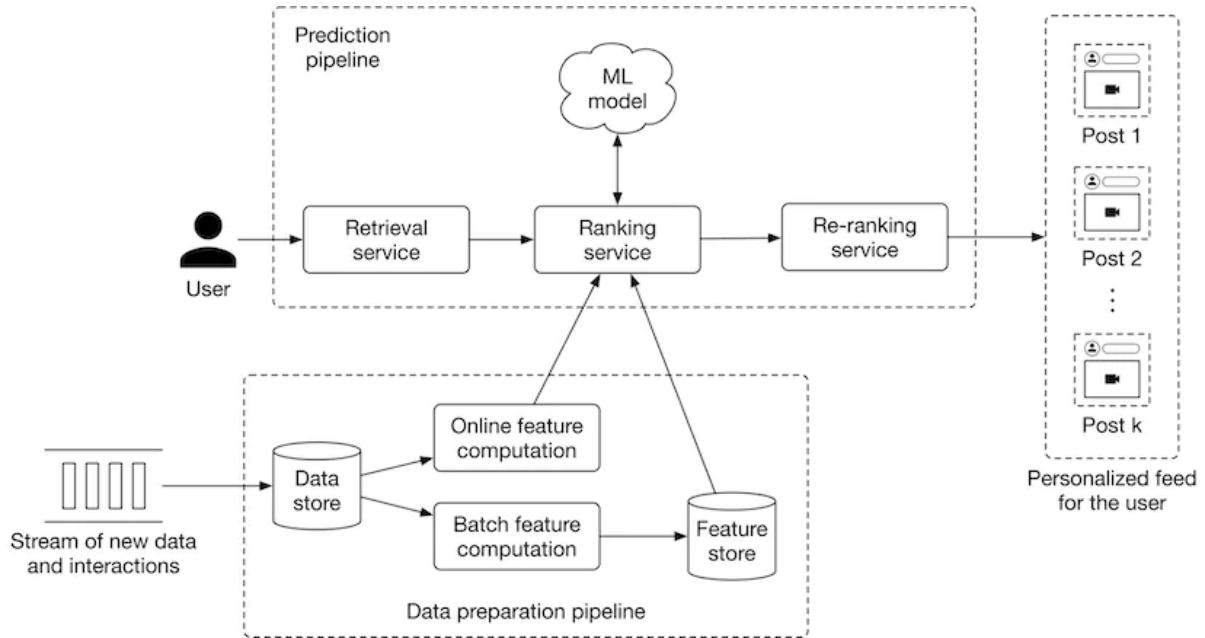


图 10.12: 个性化新闻推荐系统的机器学习系统设计

10.8 其他讨论点

如果面试最后还有时间，这里有一些额外的讨论点：

- 如何处理正在走红的帖子 [15](#)。
- 如何为新用户个性化动态 [16](#)。
- 如何减轻系统中的位置偏差 [17](#)。
- 如何确定合适的重新训练频率 [18](#)。

References

1. News Feed ranking in Facebook. <https://engineering.fb.com/2021/01/26/ml-applications/news-feed-ranking/>.
2. Twitter's news feed system. https://blog.twitter.com/engineering/en_us/topics/insights/2017/using-deep-learning-at-scale-in-twitters-timelines.
3. LinkedIn's News Feed system. <https://engineering.linkedin.com/blog/2020/understanding-feed-dwell-time-and-recommendations>.
4. BERT paper. <https://arxiv.org/pdf/1810.04805.pdf>.
5. ResNet model. <https://arxiv.org/pdf/1512.03385.pdf>.
6. CLIP model. <https://openai.com/blog/clip/>.
7. Viterbi algorithm. https://en.wikipedia.org/wiki/Viterbi_algorithm.
8. TF-IDF. <https://en.wikipedia.org/wiki/Tf%E2%80%93idf>.
9. Word2vec. <https://en.wikipedia.org/wiki/Word2vec>.
10. Serving a billion personalized news feed. <https://www.youtube.com/watch?v=Xpx5RYNTQvg>.
11. Mean absolute error loss. https://en.wikipedia.org/wiki/Mean_absolute_error.
12. Mean squared error loss. https://en.wikipedia.org/wiki/Mean_squared_error.
13. Huber loss. https://en.wikipedia.org/wiki/Huber_loss.
14. A news feed system design. <https://liuzhenglaichn.gitbook.io/system-design/news-feed/design-a-news-feed-system>.
15. Predict viral tweets. <https://towardsdatascience.com/using-data-science-to-predict-viral-tweets-61a2a2a2a2a2>.
16. Cold start problem in recommendation systems. [https://en.wikipedia.org/wiki/Cold_start_\(recommender_systems\)](https://en.wikipedia.org/wiki/Cold_start_(recommender_systems)).
17. Positional bias. <https://eugeneyan.com/writing/position-bias/>.
18. Determine retraining frequency. <https://huyenchip.com/2022/01/02/real-time-machine-learning-challenges.html#towards-continual-learning>.

11 People You May Know 你可能认识的人

11.1 Introduction

“你可能认识的人”(PYMK)是一份基于共同点(例如共同的朋友、学校或工作地点)推荐你可能想要联系的用户的列表。许多社交网络,如Facebook、LinkedIn和Twitter,都利用机器学习来驱动PYMK功能。

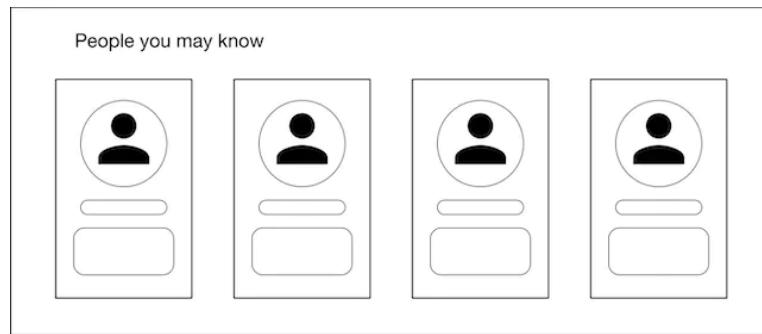


图 11.1: PYMK 功能

在本章中,我们将设计一个类似于 LinkedIn 的 PYMK 功能。该系统以用户为输入,并推荐一组潜在的联系对象。

11.2 明确需求

以下是候选人与面试官之间的典型对话。

候选人: 我能否假设构建 PYMK 功能的目的是帮助用户发现潜在的联系人并扩大他们的网络?

面试官: 是的,这是一个很好的假设。

候选人: 为了推荐潜在的联系人,必须考虑很多因素,例如位置、教育背景、工作经历、现有联系、过去的活动等。我是否应该专注于最重要的因素,例如教育背景、工作经历和用户的社交背景?

面试官: 听起来不错。

候选人: 在 LinkedIn 上,只有当双方都是对方的好友时,才能算作朋友关系。这对吗?

面试官: 是的,朋友关系是对称的。当有人向另一位用户发送联系请求时,接受者需要接受请求,才能建立联系。

候选人: 平台上总共有多少用户?其中有多少是日活跃用户?

面试官: 我们有近 10 亿用户,其中有 3 亿日活跃用户。

候选人: 平均每个用户有多少个联系人?

面试官: 1,000 个联系人。

候选人: 大多数用户的社交图谱变化不大,也就是说,他们的联系人不会在短时间内发生显著变化。我在设计 PYMK 时可以做出这个假设吗?

面试官: 这是一个很好的观点。是的,这是一个合理的假设。

让我们总结一下问题陈述。我们需要设计一个类似于 LinkedIn 的 PYMK 系统。该系统以用户为输入，并推荐一个排序的潜在联系人列表。构建该系统的目的是使用户能够更轻松地发现新联系人并扩大他们的网络。平台上总共有 10 亿用户，平均每个用户有 1,000 个联系人。

11.3 将问题框架化为机器学习任务

11.3.1 定义机器学习目标

在 PYMK 系统中，一个常见的机器学习目标是最大化用户之间建立联系的数量。这有助于用户快速扩展他们的网络。

11.3.2 指定系统的输入和输出

PYMK 系统的输入是一个用户，输出是按与用户相关性排序的联系列表。如图 11.2 所示。

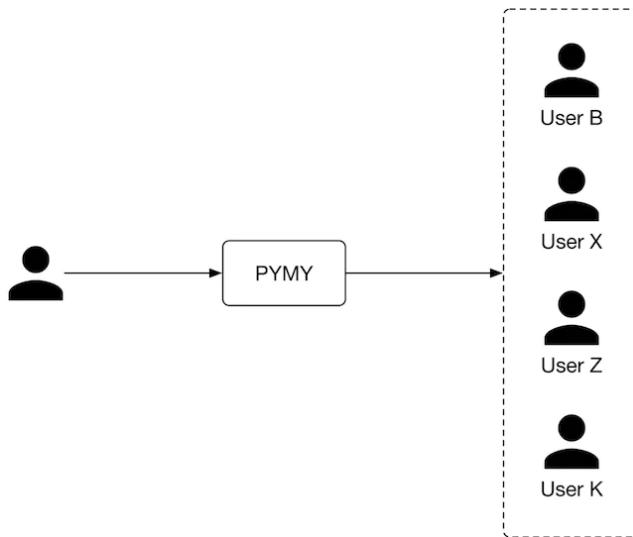


图 11.2: PYMK 系统的输入输出

11.3.3 选择合适的机器学习类别

让我们来看看构建 PYMK 的两种常见方法：点对点学习排序（LTR）和边预测。

11.3.3.1 点对点 LTR

在这种方法中，我们将 PYMK 框架化为一个排序问题，并使用点对点 LTR 来对用户进行排序。正如图 11.3 所示，在点对点 LTR 中，我们采用一个二元分类模型，该模型以两个用户为输入，并输出给定用户对形成联系的概率。

然而，这种方法有一个主要缺点；由于模型的输入是两个不同的用户，它没有考虑到可用的社交上下文。虽然这样简化了问题，但忽略用户连接信息可能会使预测不够准确。

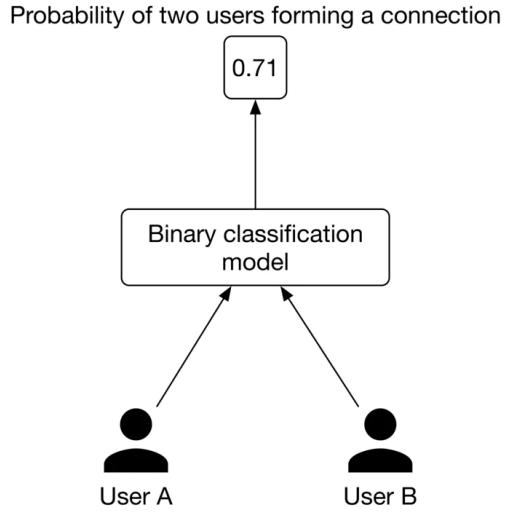


图 11.3: 具有两个输入用户的二元分类

让我们通过一个例子来分析社交上下文如何提供非常重要的见解。假设我们希望预测（用户 A, 用户 B）是否是潜在的联系。

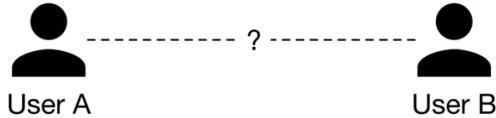


图 11.4: 用户 A 和用户 B 能否形成潜在联系？

通过查看他们的一跳邻域（用户 A 或用户 B 的联系人），我们可以获得更多的信息来确定（用户 A, 用户 B）是否是潜在联系。如图 11.5 所示，考虑两种不同的场景。

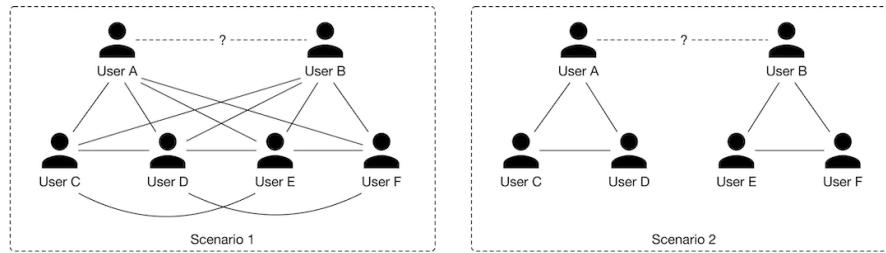


图 11.5: 两种不同的一跳邻域场景

在场景 1 中，用户 A 和用户 B 各自有四个共同的联系人，并且 C, D, E 和 F 之间也有共同的连接。

在场景 2 中，用户 A 和用户 B 各自有两个朋友，且用户 A 和用户 B 的联系人之间没有任何连接。

通过查看他们的一跳邻域，你可能会认为在场景 1 中（用户 A, 用户 B）更有可能形成联系，而不是在场景 2 中。在实践中，我们甚至可以利用两跳或三跳邻域，以从社交上下文中捕捉到更多有用的信息。

在讨论第二种方法之前，让我们了解图如何存储结构化数据（如社交上下文）以及在图上可以执行的机器学习任务。

一般来说，图表示实体（节点）集合之间的关系（边）。整个社交上下文可以表示为一个图，其中每个节点表示一个用户，两个节点之间的边表示两个用户之间的已建立联系。图 11.6 显示了一个具有四个节点和三个边的简单图。

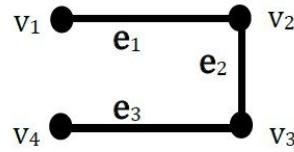


图 11.6: 一个简单的图

可以在图上表示的结构化数据上执行三种常见的预测任务：

- **图级预测**。例如，给定一个化合物作为图，预测该化合物是否为酶。
- **节点级预测**。例如，给定一个社交网络图，预测特定用户（节点）是否为垃圾信息发送者。
- **边级预测**。预测两个节点之间是否存在边。例如，给定一个社交网络图，预测两个用户是否有可能建立联系。

让我们来看看用于构建 PYMK 系统的边预测方法。

11.3.3.2 边预测

在这种方法中，我们为模型提供图信息。这使得模型能够依赖从社交图中提取的额外知识，以预测两个节点之间是否存在边。

更正式地说，我们使用一个模型，该模型以整个社交图为输入，预测两个特定节点之间存在边的概率。为了对用户 A 推荐潜在联系，我们计算用户 A 与其他用户之间的边概率，并使用这些概率作为排序标准。

除了模型利用的典型特征外，模型还依赖于从社交图中提取的额外知识，以预测两个节点之间是否存在边。

11.4 数据准备

11.4.1 数据工程

在本节中，我们讨论可用的原始数据：

- 用户
- 连接
- 互动

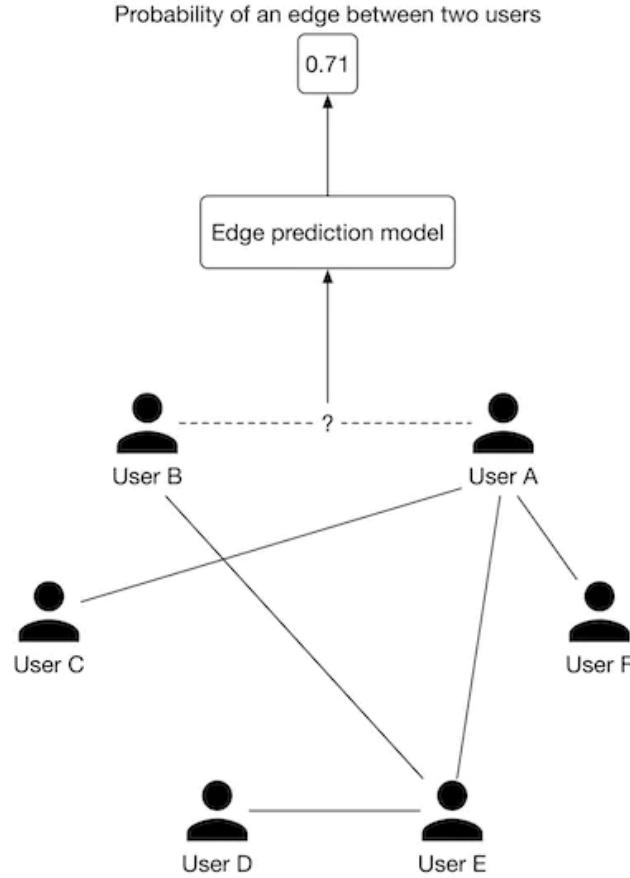


图 11.7: 具有图输入的二元分类

11.4.1.1 用户

除了用户的基本信息外，我们还拥有他们的教育和工作背景、技能等信息。表 11.1 显示了用户教育背景数据的一个示例。可能还有类似的表格用于存储工作经验、技能等。

表 11.1: 用户教育背景数据

用户 ID	学校名称	专业	毕业年份
1	X 大学	计算机科学	2020
2	Y 大学	CS	2019

处理此类原始数据的一个挑战是，特定属性可以以不同的形式表示。例如，“计算机科学”和“CS”具有相同的含义，但文本不同。因此，在数据工程步骤中标准化原始数据很重要，以避免将单个属性的不同形式视为不同属性。标准化原始数据的方法有多种，例如：

- 强制用户从预定义列表中选择属性。
- 使用启发式方法对属性的不同表示进行分组。
- 使用基于机器学习的方法，如聚类 1 或语言模型对相似属性进行分组。

11.4.1.2 连接

表 11.2 显示了连接数据的简化示例。每行代表两个用户之间的一次连接及其形成的时间。

表 11.2: 连接数据

用户 ID 1	用户 ID 2	连接形成时间
1	2	2022-01-01
3	4	2021-05-10

11.4.1.3 互动

互动类型多种多样，例如用户发送连接请求、接受请求、关注其他用户、搜索实体、查看个人资料、点赞或对帖子进行回应等。需要注意的是，实际操作中，可能会将互动数据存储在不同的数据库中，但为了简化，这里我们将所有数据放在一个表中。

表 11.3: 互动数据

用户 ID	互动对象 ID	互动类型	互动内容	时间戳
1	2	查看个人资料	-	2022-01-10
2	1	发送请求	-	2022-01-11

11.4.2 特征工程

为了确定用户（例如，用户 A）的潜在连接，模型需要利用用户 A 的信息，如年龄、性别等。此外，用户 A 与其他用户之间的亲和度也很有用。在本节中，我们讨论了一些最重要的特征。

11.4.2.1 用户特征

人口统计信息：年龄、性别、城市、国家等 人口统计信息有助于判断两个用户是否有可能建立联系。用户通常倾向于与具有相似人口统计特征的人建立联系。

人口统计数据中常常会有缺失值。要了解如何处理缺失值，请参考“简介和概述”章节。

连接数、关注者数、关注数和待处理请求数 这些信息很重要，因为用户更可能与拥有大量关注者或连接的人建立联系，而不是与连接较少的用户。

账户年龄 最近创建的账户不如存在时间更长的账户可靠。例如，如果一个账户是在昨天创建的，它更有可能是垃圾账户。因此，推荐给用户时需要谨慎。

收到的互动数量 这些是表示在一定时间（如一周内）内收到的互动总数的数值，例如点赞、分享和评论。用户往往更愿意与平台上更活跃的用户（即收到更多互动的用户）建立联系。

11.4.2.2 用户-用户亲和度

两个用户之间的亲和度是预测他们是否会建立联系的一个重要信号。让我们看看一些可以捕捉用户-用户亲和度的重要特征。

教育和工作亲和度

- **共同的学校**: 用户倾向于与曾就读于同一所学校的人建立联系。
- **在校同期**: 在校期间的重叠年份增加了两个用户建立联系的可能性。例如，用户可能希望与在同一时间就读于 X 学校的人建立联系。
- **相同的专业**: 一个二元特征，表示两个用户是否在学校中有相同的专业。
- **共同的公司数**: 用户可能会与曾在相同公司工作的人建立联系。
- **相同的行业**: 一个二元特征，表示两个用户是否在相同的行业工作。

社交亲和度

- **个人资料访问**: 用户查看另一个用户个人资料的次数。
- **共同连接数（即共同好友）**: 如果两个用户有很多共同的连接，他们更有可能建立联系。这个特征是最重要的预测特征之一²。
- **时间折扣的共同好友**: 该特征根据连接存在的时间对共同连接进行加权。让我们通过一个例子来理解这个特征背后的原因。

假设我们想确定用户 B 是否是用户 A 的潜在连接。考虑两种情况：在场景 1 中，用户 A 的连接是最近形成的，而在场景 2 中，这些连接早已形成。如图 11.8 所示。

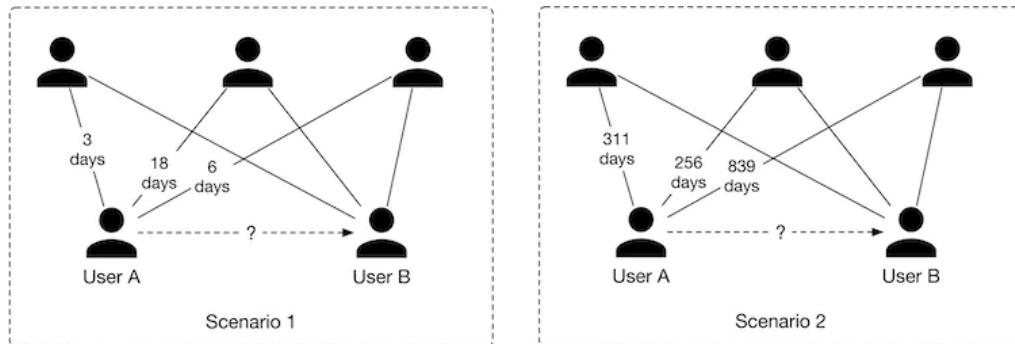


图 11.8: 比较最近的连接与旧连接

在场景 1 中，用户 A 的网络最近增长，这意味着用户 A 更有可能与用户 B 建立联系。而在场景 2 中，用户 A 可能已经知道用户 B，但选择不与其建立联系。

11.5 模型开发

11.5.1 模型选择

前面我们将 PYMK 问题定义为边预测任务，即模型以社交图为输入，预测两个用户之间是否存在边。为了处理边预测任务，我们选择可以处理图输入的模型。图神经网络 (GNNs) 专为图数据而设计。让我们更详细地了解它。

11.5.2 GNNs

GNNs 是可以直接应用于图的神经网络。它们提供了一种便捷的方法来执行图级、节点级和边级的预测任务。

如图 11.9 所示，GNN 以图作为输入。该输入图包含与节点和边相关的属性。例如，节点可以存储年龄、性别等信息，而边可以存储用户之间的特征，如共同的学校和工作地点数量、连接时间等。给定输入图及相关属性，GNN 为每个节点生成节点嵌入。

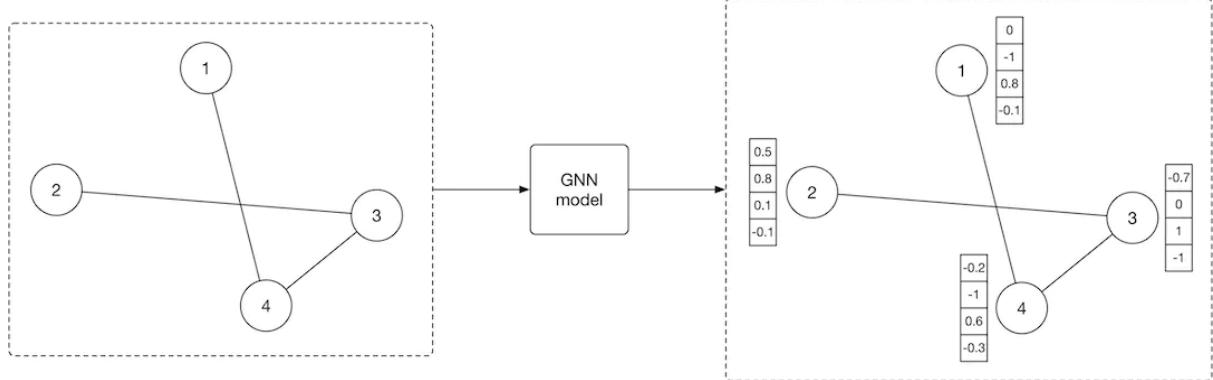


图 11.9: GNN 模型为每个图节点生成节点嵌入

一旦生成了节点嵌入，就可以使用相似性度量（例如点积）来预测两个节点之间形成连接的可能性。例如，如图 11.10 所示，我们计算节点 2 和节点 4 的嵌入之间的点积，以预测它们之间是否存在边。

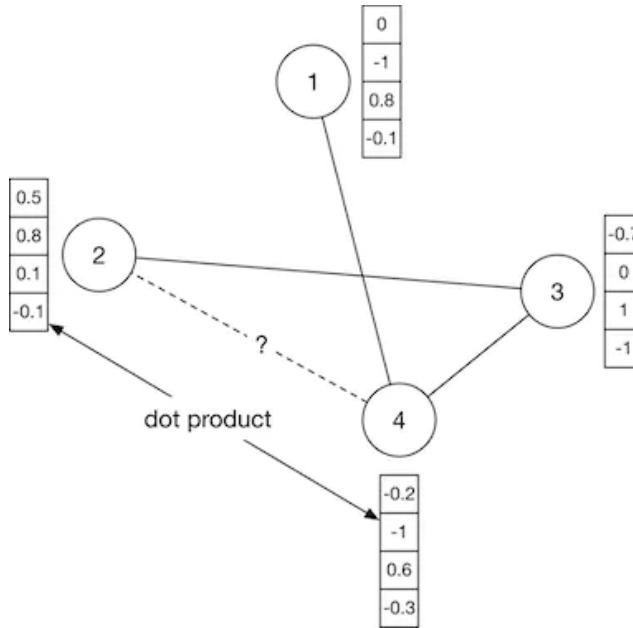


图 11.10: 预测节点 2 和节点 4 之间形成边的可能性

近年来，已经开发出多种基于 GNN 的架构，如 GCN 3、GraphSAGE 4、GAT 5 和 GIT 6。这些变体具有不同的架构和复杂性。要确定哪个架构效果最好，需要进行大量实验。要深入了解基于 GNN 的架构，请参考 7。

11.5.3 模型训练

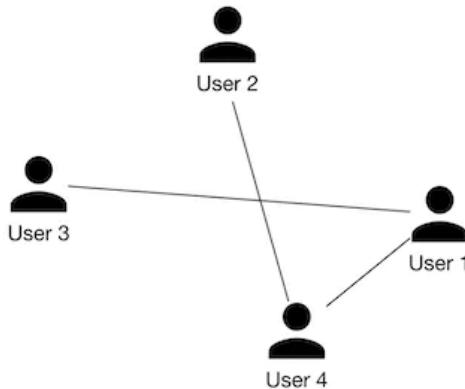
为了训练 GNN 模型，我们将社交图在时间 t 的快照提供给模型，模型预测将在时间 $t + 1$ 时形成的连接。让我们看看如何构建训练数据。

11.5.3.1 构建数据集

构建数据集的步骤如下：

1. 创建时间 t 的图快照
2. 计算图的初始节点特征和边特征
3. 创建标签

1. 创建时间 t 的图快照。 构建训练数据的第一步是为模型创建输入。由于 GNN 模型期望以社交图作为输入，我们使用可用的原始数据在时间 t 创建社交图的快照。图 11.11 显示了时间 t 的图的一个示例。



A snapshot of the social graph at time t

图 11.11: 时间 t 的社交图快照

2. 计算图的初始节点特征和边特征。 如图 11.12 所示，我们提取用户的特征，如年龄、性别、账户年龄、连接数等。这些特征被用作节点的初始特征向量。

类似地，我们提取用户之间的亲和度特征，并将它们用作边的初始特征向量。如图 11.13 所示，在用户 2 和用户 4 之间存在一条边。 $E_{2,4}$ 表示捕捉共同好友数量、个人资料访问次数、在共同学校的重叠时间等信息的初始特征向量。

3. 创建标签。 在这一步中，我们创建模型需要预测的标签。我们使用时间 $t + 1$ 时的图快照 (graph snapshot) 来确定正标签或负标签。让我们看一个具体的例子。

如图 11.14 所示，正标签和负标签是根据是否在 $t + 1$ 时形成新边来创建的。特别地，我们在 $t + 1$ 时建立连接的节点对标记为正标签。否则，它们标记为负标签。

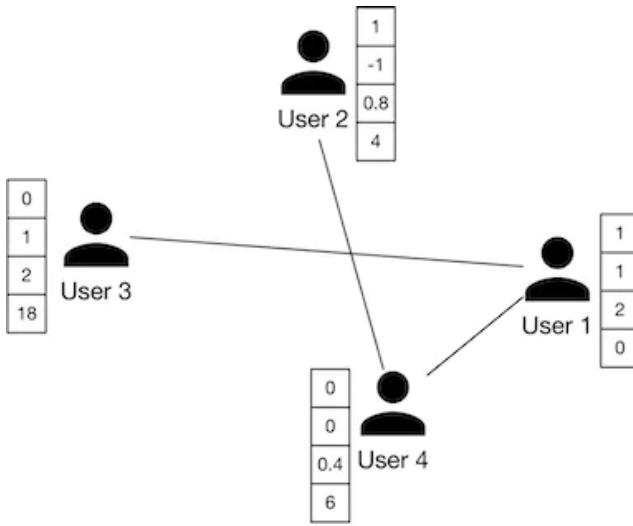


图 11.12: 初始边特征

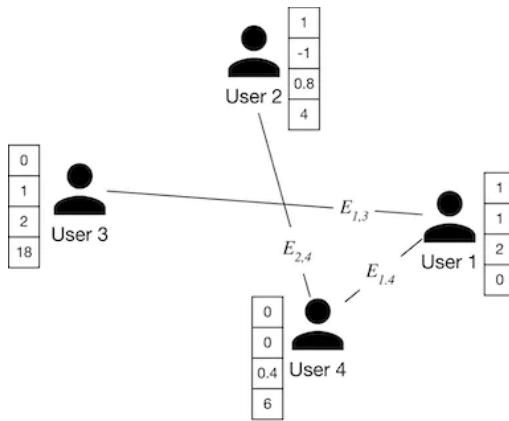


图 11.13: 初始节点特征

11.5.3.2 选择损失函数

一旦创建了输入图和标签，我们就可以训练 GNN 模型了。关于 GNN 训练的详细解释以及选择哪些损失函数超出了本书的范围。要了解更多相关内容，请参见 [7](#)。

11.6 评估

11.6.1 离线指标

在离线评估中，我们评估 GNN 模型和 PYMK 系统的性能。

GNN 模型 由于 GNN 模型预测边的存在，可以将其视为二元分类模型。使用 ROC-AUC 指标来衡量模型的性能。

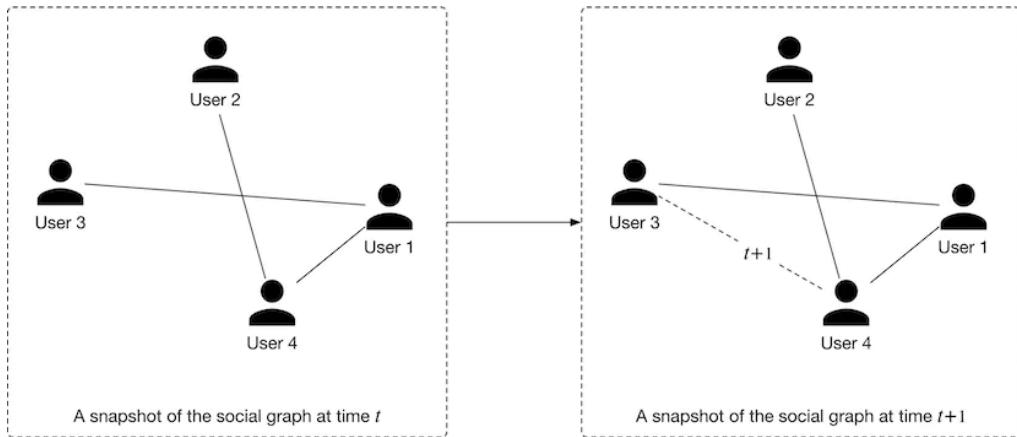
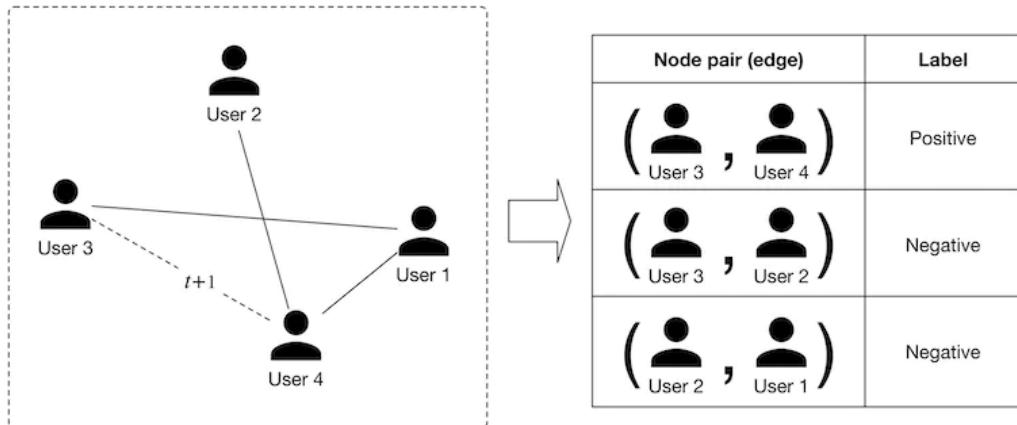
图 11.14: 从时间 t 到 $t+1$ 形成的新边

图 11.15: 创建正标签和负标签

PYMK 系统 在之前的章节中，我们详细讨论了如何为排名和推荐系统选择合适的离线指标，因此这里不再赘述。在我们的系统中，用户要么与推荐的连接建立关系，要么忽略它。由于这种二元性质（连接或不连接），mAP 是一个很好的选择。

11.6.2 在线指标

在实践中，公司会跟踪大量的在线指标来衡量 PYMK 系统的影响。让我们来探讨两个最重要的指标：

- 最近 XXX 天内发送的连接请求总数
- 最近 XXX 天内接受的连接请求总数

最近 XXX 天内发送的连接请求总数 该指标有助于我们了解模型是否增加或减少了发送的连接请求数量。例如，如果一个模型导致发送的连接请求总数增加了 5%，我们可以认为该模型对业务目标有正面影响。

然而，该指标存在一个主要缺点。只有当接收方接受连接请求时，两个用户之间才会形成新的连接。例如，一个用户可能发送了 1000 个连接请求，但接收方只接受了一小部分。这一指标可能无法准确反映用户网络的实际增长情况。现在，让我们用下一个指标来解决这个缺点。

最近 XXX 天内接受的连接请求总数 由于只有当接收方接受发送方的请求时，新的连接才会形成，因此该指标能够准确反映用户网络的实际增长情况。

11.7 服务

在服务时，PYMK 系统需要高效地向给定用户推荐一系列潜在的连接。在本节中，我们解释为什么需要进行速度优化，并介绍一些提高 PYMK 效率的技术。然后，我们提出一种设计，其中不同的组件协同工作来处理请求。

11.7.1 效率

正如需求收集部分所述，平台上用户总数为 10 亿，这意味着我们需要在 10 亿个嵌入中进行排序，以找到单个用户的潜在连接。更具挑战性的是，需要为每个用户运行该算法。毫不意外，这在我们的规模上是不可行的。为了解决这个问题，通常使用两种常见技术：(1) 利用朋友的朋友 (FoF) 和 (2) 预计算 PYMK。

11.7.1.1 利用朋友的朋友 (FoF)

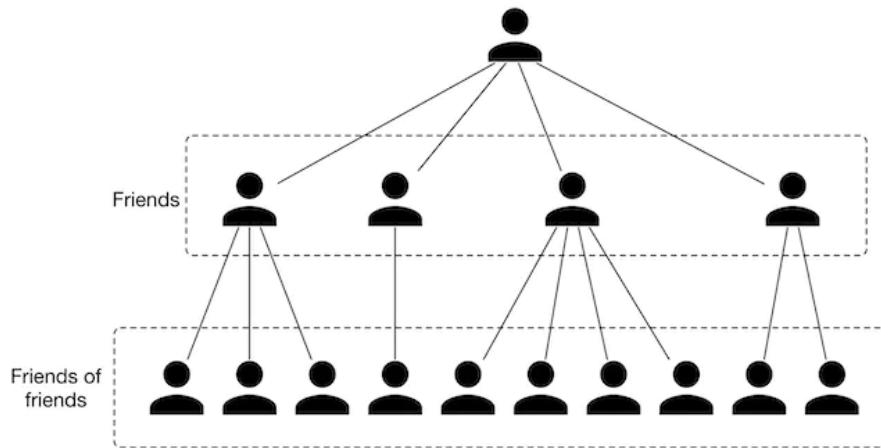


图 11.16: 用户的朋友的朋友 (FoF)

根据 Meta 的一项研究²，92% 的新友谊是通过朋友的朋友形成的。该技术利用用户的 FoF 来缩小搜索范围。

如前所述，一个用户平均有 1000 个朋友。这意味着用户平均拥有 100 万个 (1000×1000) FoF。这将搜索范围从 10 亿减少到 100 万。

11.7.1.2 预计算 PYMK

让我们退一步，考虑采用在线或批量预测 (Batch prediction)。

在线预测 在 PYMK 中，**在线预测**是指在用户加载主页时实时生成潜在的连接。在这种方法中，我们不会为不活跃的用户生成推荐。由于推荐是“即时”计算的，如果计算推荐所需时间过长，会带来不佳的用户体验。

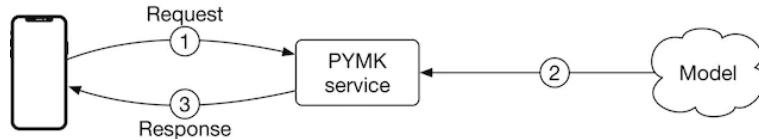


图 11.17: PYMK 中的在线预测

批量预测 (Batch prediction) 批量预测意味着系统为所有用户预先计算潜在连接并将其存储在数据库中。当用户加载主页时，我们直接获取预先计算的推荐，因此从最终用户的角度来看，推荐是即时的。批量预测的缺点是可能会产生不必要的计算。假设 20% 的用户每天登录。如果我们每天为每个用户生成推荐，那么用于生成 80% 推荐的计算能力将被浪费。

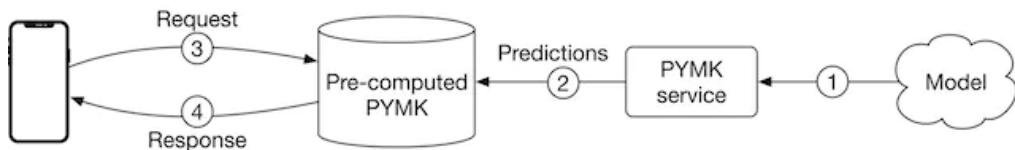


图 11.18: PYMK 中的批量预测

我们选择在线还是批量？ 我们推荐批量预测有两个原因。首先，根据收集的需求，每天有 3 亿活跃用户。即时计算所有 3 亿用户的 PYMK 可能过于缓慢，影响用户体验。

其次，由于 PYMK 中的社交图不会迅速变化，预计算的推荐在较长时间内仍然具有相关性。例如，我们可以保留 PYMK 推荐七天，然后重新计算它们。对于新用户，可以缩短时间窗口（例如一天），因为他们的网络通常增长得更快。

在社交网络中，用户可能不希望反复看到相同的一组推荐连接。为支持这一点，我们可以预先计算比所需更多的连接，只显示用户尚未见过的推荐。

11.7.2 机器学习系统设计

图 11.19 显示了 PYMK 机器学习系统设计。设计包括两个 Pipeline:

- PYMK 生成 Pipeline
- 预测 Pipeline

让我们检查每个部分。

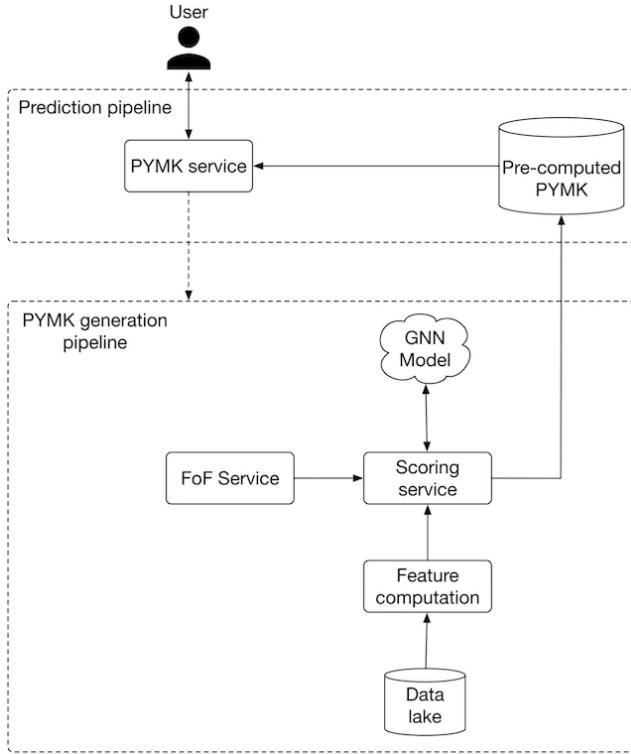


图 11.19: PYMK 机器学习系统设计

PYMK 生成 Pipeline 该 Pipeline 负责为所有用户生成 PYMK 并将结果存储在数据库中。让我们更详细地了解这个 Pipeline。

首先，对于特定用户，FoF 服务将连接缩小到一小部分候选连接（2 跳邻居）。如图 11.20 所示。

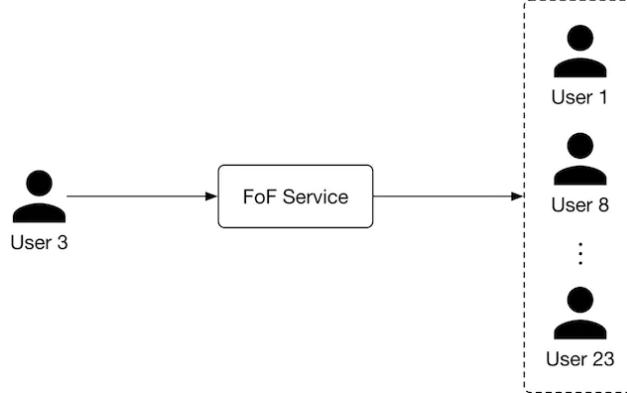


图 11.20: FoF 服务输入-输出

接下来，评分服务获取 FoF 服务生成的候选连接，使用 GNN 模型对每个连接进行评分，然后为用户生成 PYMK 的排序列表。PYMK 被存储在数据库中。当用户请求时，我们可以直接从数据库中提取其个人的 PYMK 列表。此流程如图 11.21 所示。

预测 Pipeline 当请求到达时，PYMK 服务首先查看预计算的 PYMK 是否存在推荐。如果存在，则直接获取推荐。如果没有，它会向 PYMK 生成 Pipeline 发送一次性请求。

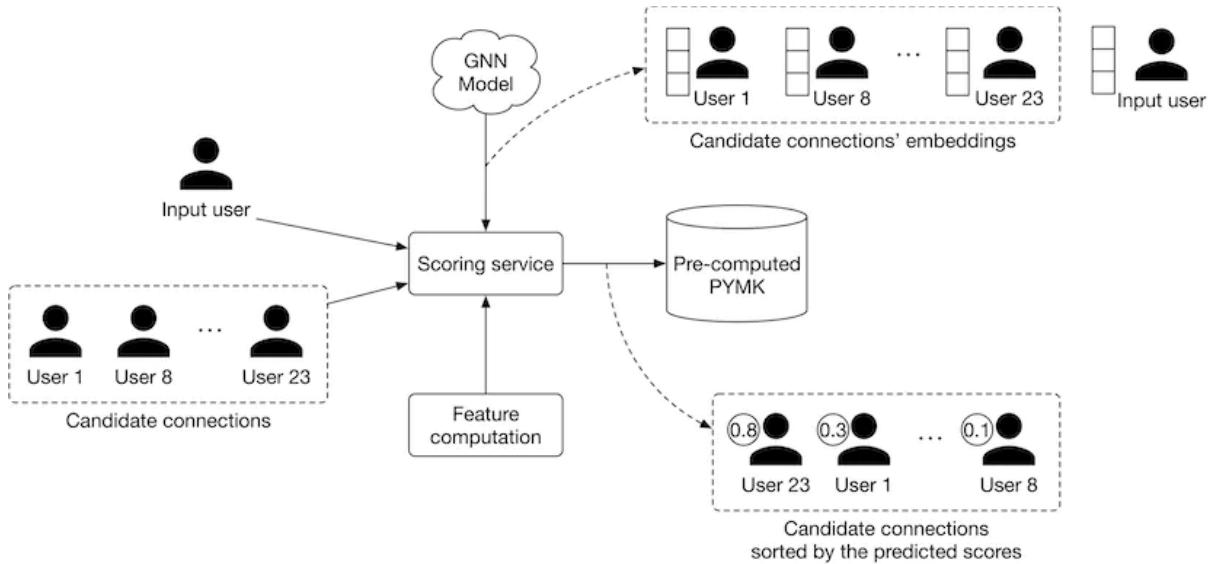


图 11.21: 评分服务输入-输出

请注意，我们提出的是一个简化的系统。如果在面试中被要求优化它，以下是一些潜在的讨论点：

- 仅为活跃用户预算 PYMK。
- 使用轻量级排序器在评分服务分配分数之前将生成的候选减少到较小的集合。
- 使用重新排序服务为最终的 PYMK 列表添加多样性。

11.8 其他讨论点

如果在面试结束时还有时间，可以考虑以下讨论点：

- 个性化随机游走 8 是另一种常用于推荐的方法。由于其高效性，是建立基线的有用方法。
- 偏差问题。频繁用户在训练数据中的代表性通常高于偶尔使用的用户。由于训练数据中代表性不均衡，模型可能会对某些群体产生偏见，而对其他群体不利。例如，在 PYMK 列表中，频繁用户可能会被推荐给其他用户的频率更高。随后，这些用户可能会建立更多的连接，使他们在训练数据中的代表性更强 9。
- 当用户多次忽略推荐的连接时，如何在未来的重新排序中考虑到这一点。理想情况下，忽略的推荐应具有较低的排名 9。
- 用户可能不会立即发送连接请求。可能需要几天或几周。那么，我们应该何时将推荐的连接标记为负面？一般来说，我们将如何处理推荐系统中的延迟反馈问题 10？

References

1. Clustering in ML. <https://developers.google.com/machine-learning/clustering/overview>.
2. PYMK on Facebook. <https://youtu.be/Xpx5RYNTQvg?t=1823>.
3. Graph convolutional neural networks. <http://tkipf.github.io/graph-convolutional-networks/>.
4. GraphSage paper. <https://cs.stanford.edu/people/jure/pubs/graphsage-nips17.pdf>.
5. Graph attention networks. <https://arxiv.org/pdf/1710.10903.pdf>.
6. Graph isomorphism network. <https://arxiv.org/pdf/1810.00826.pdf>.
7. Graph neural networks. <https://distill.pub/2021/gnn-intro/>.
8. Personalized random walk. https://www.youtube.com/watch?v=HbzQzUaJ_9I.
9. LinkedIn's PYMK system. <https://engineering.linkedin.com/blog/2021/optimizing-pymk-for-equity-in-network-creation>.
10. Addressing delayed feedback. <https://arxiv.org/pdf/1907.06558.pdf>.