# 基于深度学习的xxxxx

# 绪论

## 深度学习时代推荐系统

毫无疑问，推荐系统从来没有像现在这样影响着我们的生活。想上网购物，天猫、京东的推荐系统会帮你挑选商品；想了解资讯，头条、知乎的推荐系统会为你准备感兴趣的新闻和知识；想消遣放松，抖音、快手的推荐系统会为你奉上让你欲罢不能的短视频。而驱动这些巨头进行推荐服务的，都是基于深度学习的推荐模型。

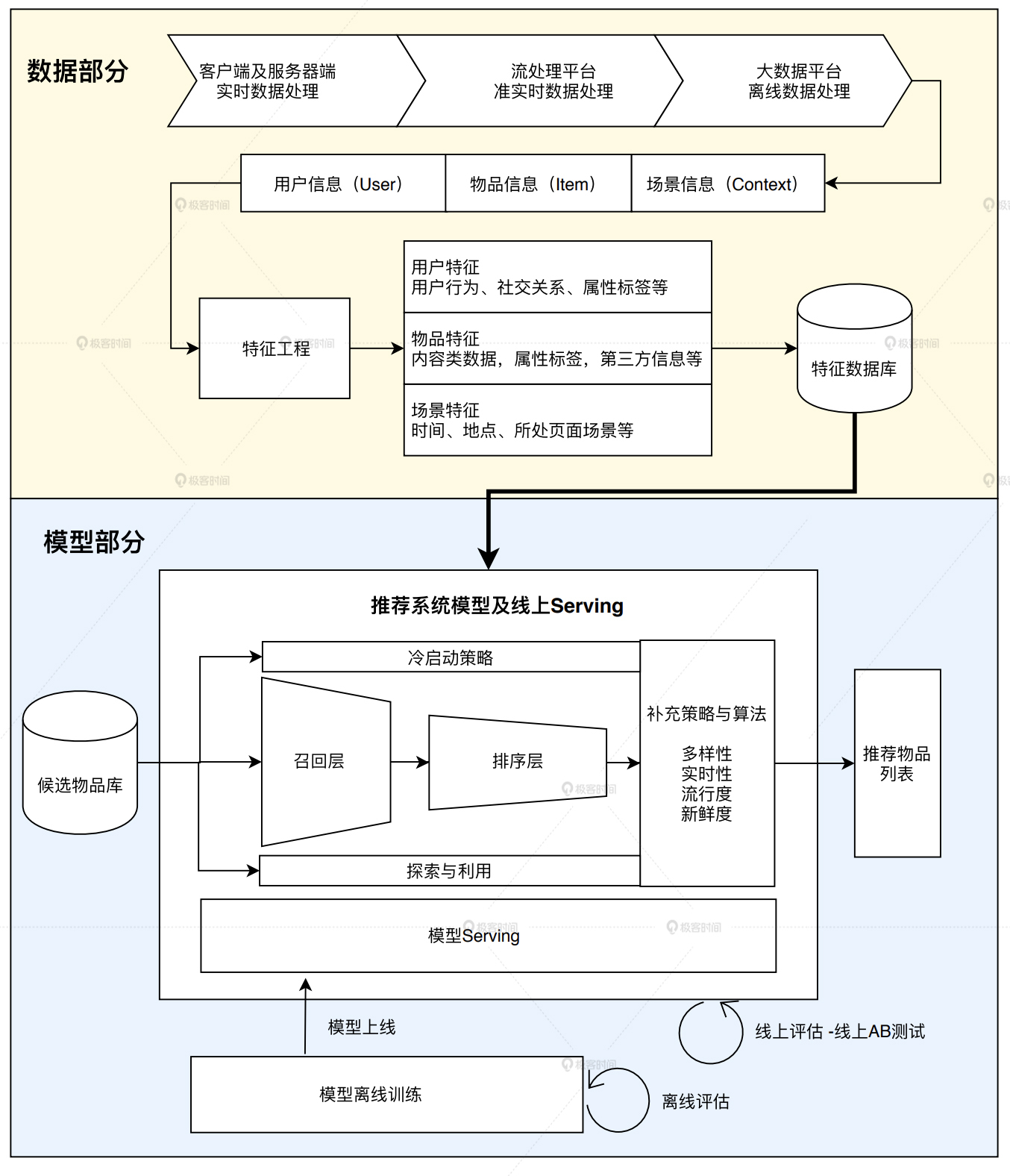
2013 年，百度率先在广告系统中应用了深度学习，2015 到 2020 年，阿里提出并应用了从 MLR 到 DIEN 等一系列的深度学习模型。国外的互联网巨头也不逞多让，从最早的 Google 的 Word2vec，到 2015 年 YouTube 的深度学习推荐系统，再到之后的 Facebook、Amazon、微软等等，几乎所有头部公司的成功应用，让深度学习如后浪般席卷了推荐系统业界，将传统的推荐模型彻底取代。

2019 年，阿里著名的千人千面系统驱动了天猫“双 11”2684 亿元的成交额。更让人感叹的是，字节跳动在 2020 年 1 月 5 日发布的一份《抖音数据报告》中宣布，抖音日活用户突破了 4 亿。要知道，这距离抖音 2016 年 9 月上线仅过去了 3 年多一点的时间。作为一个几乎完全由推荐系统驱动的应用，这样的增长速度是惊人的，是前所未有的。而字节跳动技术团队曾经披露的，深度学习在推荐算法、视频内容理解、自然语言处理等方向上的应用，则又一次向我们印证了深度学习的强大实力。

# 基础架构

## 经典技术架构

推荐系统要解决的问题用一句话总结就是，在“信息过载”的情况下，用户如何高效获取感兴趣的信息。



## 深度学习基础

神经元：输入--激活函数--输出

神经网络

网络训练

深度学习与神经网络关系

深度学习在推荐系统中应用

# 特征工程

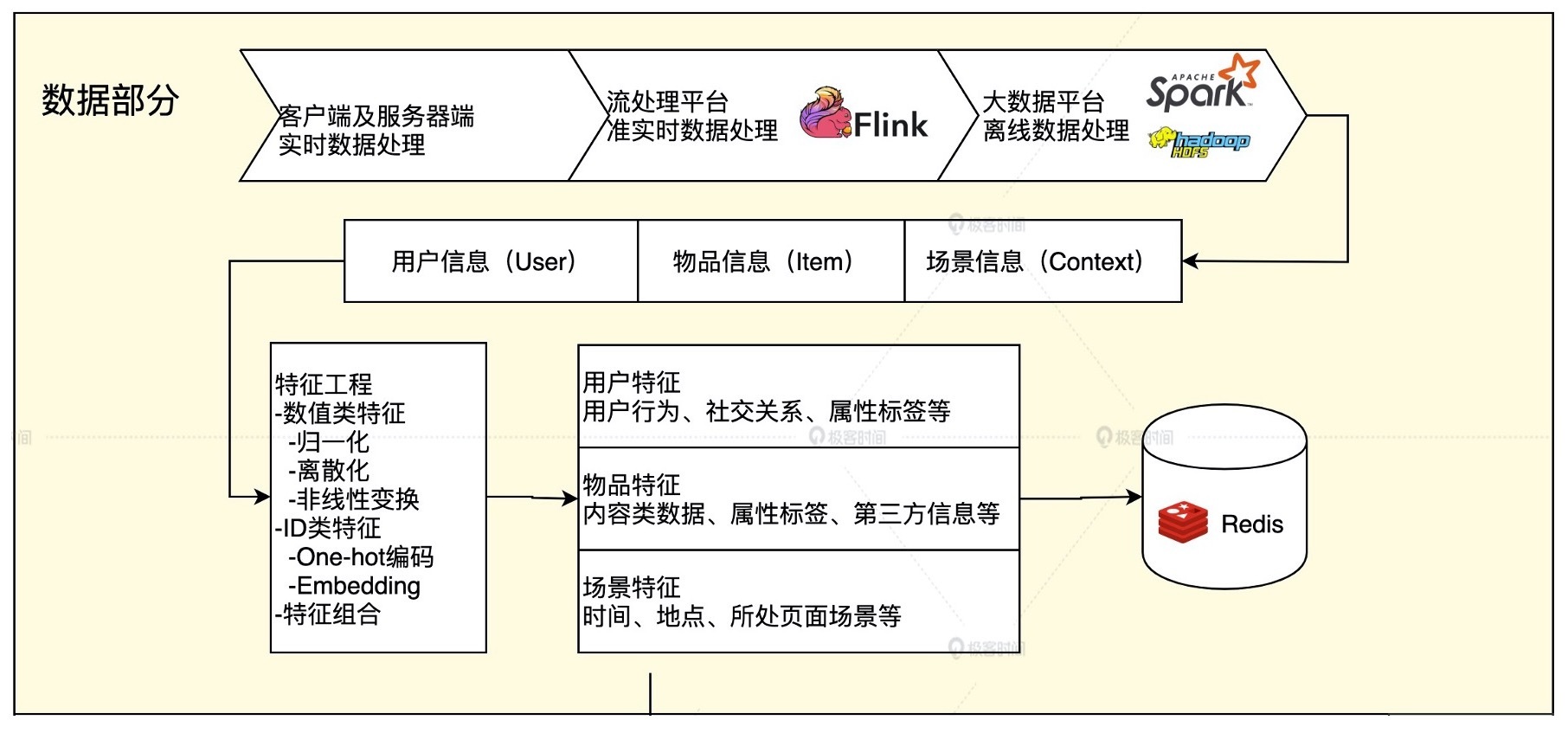


图1 特征工程部分在推荐系统中的位置

在这个系统之中，特征工程就是利用工程手段从“用户信息”“物品信息”“场景信息”中提取特征的过程。这个过程说起来容易，但实际做起来其实困难重重。

## 特征工程原则

这其实也是我们构建推荐系统特征工程的原则：尽可能地让特征工程抽取出的一组特征，能够保留推荐环境及用户行为过程中的所有“有用“信息，并且尽量摒弃冗余信息。

## 常用特征

### 用户行为数据

用户行为在推荐系统中一般分为显性反馈行为（Explicit Feedback）和隐性反馈行为（Implicit Feedback）两种，在不同的业务场景中，它们会以不同的形式体现。

### 用户关系数据

用户关系数据也可以分为“显性”和“隐性”两种，或者称为“强关系”和“弱关系”。如图 4 所示，用户与用户之间可以通过“关注”“好友关系”等连接建立“强关系”，也可以通过“互相点赞”“同处一个社区”，甚至“同看一部电影”建立“弱关系”。

在推荐系统中，利用用户关系数据的方式也是多种多样的，比如可以将用户关系作为召回层的一种物品召回方式；也可以通过用户关系建立关系图，使用 Graph Embedding 的方法生成用户和物品的 Embedding；还可以直接利用关系数据，通过“好友”的特征为用户添加新的属性特征；甚至可以利用用户关系数据直接建立社会化推荐系统。

### 属性、标签类数据

把属性类数据（Attribute Data）和标签类数据（Label Data）归为一组进行讨论，是因为它们本质上都是直接描述用户或者物品的特征。属性和标签的主体可以是用户，也可以是物品。

在推荐系统中使用属性、标签类数据，一般是通过 Multi-hot 编码的方式将其转换成特征向量，一些重要的属性标签类特征也可以先转换成 Embedding，比如业界最新的做法是将标签属性类数据与其描述主体一起构建成知识图谱（Knowledge Graph），在其上施以 Graph Embedding 或者 GNN（Graph Neural Network，图神经网络）生成各节点的 Embedding，再输入推荐模型。

### 内容类数据

内容类数据（Content Data）可以看作属性标签型特征的延伸，同样是描述物品或用户的数据，但相比标签类特征，内容类数据往往是大段的描述型文字、图片，甚至视频。一般来说，内容类数据无法直接转换成推荐系统可以“消化”的特征，需要通过自然语言处理、计算机视觉等技术手段提取关键内容特征，再输入推荐系统。

### 场景信息（上下文信息）

场景信息，或称为上下文信息（Context Information），它是描述推荐行为产生的场景的信息。最常用的上下文信息是“时间”和通过 GPS、IP 地址获得的“地点”信息。根据推荐场景的不同，上下文信息的范围极广，除了我们上面提到的时间和地点，还包括“当前所处推荐页面”“季节”“月份”“是否节假日”“天气”“空气质量”“社会大事件”等等。

## 特征处理

特征处理的原则，“特征处理没有标准答案，需要根据模型效果实践出真知”。

一句话总结 Spark 的计算过程：Stage 内部数据高效并行计算，Stage 边界处进行消耗资源的 shuffle 操作或者最终的 reduce 操作。

### 利用 One-hot 编码处理类别型特征

我们使用 Spark 的机器学习库 MLlib 来完成 One-hot 特征的处理。

最主要的步骤是，我们先创建一个负责 One-hot 编码的转换器，OneHotEncoderEstimator，然后通过它的 fit 函数完成指定特征的预处理，并利用 transform 函数将原始特征转换成 One-hot 特征。

One-hot 编码也可以自然衍生成 Multi-hot 编码（多热编码）。

### 数值型特征的处理 - 归一化和分桶

主要讨论两方面问题，**一是特征的尺度，二是特征的分布**。

Fr这个特征由于波动范围高出 fs 几个量级，可能会完全掩盖 fs 作用，这当然是我们不愿意看到的。为此我们希望把两个特征的尺度拉平到一个区域内，通常是[0,1]范围，这就是所谓归一化。归一化虽然能够解决特征取值范围不统一的问题，但无法改变特征值的分布。

我们经常会用分桶的方式来解决特征值分布极不均匀的问题。所谓“分桶（Bucketing）”，就是将样本按照某特征的值从高到低排序，然后按照桶的数量找到分位数，将样本分到各自的桶中，再用桶 ID 作为特征值。

在 Spark MLlib 中，分别提供了两个转换器 MinMaxScaler 和 QuantileDiscretizer，来进行归一化和分桶的特征处理。它们的使用方法和之前介绍的 OneHotEncoderEstimator 一样，都是先用 fit 函数进行数据预处理，再用 transform 函数完成特征转换。

**无论是平方还是开方操作，改变的还是这个特征值的分布**，这些操作与分桶操作一样，都是希望通过改变特征的分布，让模型能够更好地学习到特征内包含的有价值信息。但由于我们没法通过人工的经验判断哪种特征处理方式更好，所以索性把它们都输入模型，让模型来做选择。

## Embedding

简单来说，Embedding 就是用一个数值向量“表示”一个对象（Object）的方法，我这里说的对象可以是一个词、一个物品，也可以是一部电影等等。

### 序列数据的embedding

Word2vec

Word2vec 是由谷歌于 2013 年正式提出的，其实它并不完全是原创性的，学术界对词向量的研究可以追溯到 2003 年，甚至更早的时期。但正是谷歌对 Word2vec 的成功应用，让词向量的技术得以在业界迅速推广，进而使 Embedding 这一研究话题成为热点。毫不夸张地说，Word2vec 对深度学习时代 Embedding 方向的研究具有奠基性的意义。从另一个角度来看，Word2vec 的研究中提出的模型结构、目标函数、负采样方法、负采样中的目标函数在后续的研究中被重复使用并被屡次优化。掌握 Word2vec 中的每一个细节成了研究 Embedding 的基础。

Item2vec

在 Word2vec 诞生之后，Embedding 的思想迅速从自然语言处理领域扩散到几乎所有机器学习领域，推荐系统也不例外。既然 Word2vec 可以对词“序列”中的词进行 Embedding，那么对于用户购买“序列”中的一个商品，用户观看“序列”中的一个电影，也应该存在相应的 Embedding 方法。

于是，微软于 2015 年提出了 Item2Vec 方法，它是对 Word2vec 方法的推广，使 Embedding 方法适用于几乎所有的序列数据。Item2Vec 模型的技术细节几乎和 Word2vec 完全一致，只要能够用序列数据的形式把我们要表达的对象表示出来，再把序列数据“喂”给 Word2vec 模型，我们就能够得到任意物品的 Embedding 了。

### 图结构数据embedding

# 线上服务

# 推荐模型

# 模型评估

# 前沿拓展

# 参考文献

1.王[喆](https://book.douban.com/search/%E7%8E%8B%E5%96%86) 《深度学习推荐系统》2020.3 电子工业出版社