# 一．概述

用饭馆吃饭比喻推荐系统的分类：

推荐系统是推荐商品的，但是它如何推荐的呢?以饭馆吃饭来举一波例子。

原始阶段：你到一家饭馆吃饭，没有推荐系统的时候，你就默默看着菜单，凭着自己的感觉点菜。

1.0版本：你再去吃饭的时候，老板来了一句，本店特色菜是××××，你有兴趣吗? 这种就是专家推荐系统，由专家(老板)负责筛选内容，然后统一的推送给所有用户，例如热门推荐文章或者商品

2.0版本：你又去吃饭，老板来了一句“喜欢川菜还是粤菜啊?”，你来了一句“川菜”，然后老板说“本店川菜的辣子鸡和夫妻肺片很好吃，要不要试一下呢?”这就是标签化用户，然后根据用户标签进行推荐，例如Quora问答社区，你刚进入的时候，让你选择感兴趣的话题，然后进行推荐。

3.0版本：你又去吃饭，已经是熟客了，这个时候老板说“你平常很喜欢吃川菜，本店川菜系列新增了水煮肉片，要不要尝试一下?”这是基于内容的推荐系统，将所有的内容进行分类，然后推荐给经常消费该类的用户。

4.0版本：你又去吃饭，老板说“和你一样的口味的人，很喜欢我们最近推出的宫保鸡丁，你要不要试一试?”这就是基于用户行为的推荐系统，根据用户行为对用户进行分类，然后将某一类用户均常用的东西，推荐给这一类的其他用户。

目前的推荐系统基本包含在上述四个版本之中，常常是结合起来用。例如当你一开始进入某个网站，热门内容是1.0版本，然后通过收集你点击了哪些内容，逐步刻画精准的用户画像，进阶到3.0或者4.0版本。

## Memory-based    （启发式）

基于内容的推荐

基于内容的推荐（Content Based）应该算最早被使用的推荐方法，它根据用户过去喜欢的产品（本文统称为 item），为用户推荐和他过去喜欢的产品相似的产品。例如，一个推荐饭店的系统可以依据某个用户之前喜欢很多的烤肉店而为他推荐烤肉店。 CB最早主要是应用在信息检索系统当中，所以很多信息检索及信息过滤里的方法都能用于CB中。

CB的过程一般包括以下三步：

1) Item Representation：为每个item抽取出一些特征（也就是item的content了）来表示此item；

2) Profile Learning：利用一个用户过去喜欢（及不喜欢）的item的特征数据，来学习出此用户的喜好特征（profile）；

3) Recommendation Generation：通过比较上一步得到的用户profile与候选item的特征，为此用户推荐一组相关性最大的item。

举个例子说明前面的三个步骤。对于个性化阅读来说，一个item就是一篇文章。根据上面的第一步，我们首先要从文章内容中抽取出代表它们的属性。常用的方法就是利用出现在一篇文章中词来代表这篇文章，而每个词对应的权重往往使用信息检索中的tf-idf来计算。比如对于本文来说，词“CB”、“推荐”和“喜好”的权重会比较大，而“烤肉”这个词的权重会比较低。利用这种方法，一篇抽象的文章就可以使用具体的一个向量来表示了。第二步就是根据用户过去喜欢什么文章来产生刻画此用户喜好的 profile了，最简单的方法可以把用户所有喜欢的文章对应的向量的平均值作为此用户的profile。比如某个用户经常关注与推荐系统有关的文章，那么他的profile中“CB”、“CF”和“推荐”对应的权重值就会较高。在获得了一个用户的profile后，CB就可以利用所有item与此用户profile的相关度对他进行推荐文章了。一个常用的相关度计算方法是cosine。最终把候选item里与此用户最相关（cosine值最大）的N个item作为推荐返回给此用户。

对于以上三个步骤，具体的操作：

**1) Item Representation：**

真实应用中的item往往都会有一些可以描述它的属性。这些属性通常可以分为两种：结构化的（structured）属性与非结构化的（unstructured）属性。所谓结构化的属性就是这个属性的意义比较明确，其取值限定在某个范围；而非结构化的属性往往其意义不太明确，取值也没什么限制，不好直接使用。比如在交友网站上，item就是人，一个item会有结构化属性如身高、学历、籍贯等，也会有非结构化属性（如item自己写的交友宣言，博客内容等等）。对于结构化数据，我们自然可以拿来就用；但对于非结构化数据（如文章），我们往往要先把它转化为结构化数据后才能在模型里加以使用。真实场景中碰到最多的非结构化数据可能就是文章了（如个性化阅读中）。下面我们就详细介绍下如何把非结构化的一篇文章结构化。

如何代表一篇文章在信息检索中已经被研究了很多年了，下面介绍的表示技术其来源也是信息检索，其名称为向量空间模型（Vector Space Model，简称VSM）。

记我们要表示的所有文章集合为

，而所有文章中出现的词（对于中文文章，首先得对所有文章进行分词）的集合（也称为词典）为

。也是说，我们有N篇要处理的文章，而这些文章里包含了n个不同的词。我们最终要使用一个向量来表示一篇文章，比如第j篇文章被表示为

，其中

表示第1个词

在文章j中的权重，值越大表示越重要；

中其他向量的解释类似。所以，为了表示第j篇文章，现在关键的就是如何计算

各分量的值了。例如，我们可以选取

为1，如果词

出现在第 j 篇文章中；选取为0，如果

未出现在第j篇文章中。我们也可以选取

为词

出现在第 j 篇文章中的次数（frequency）。但是用的最多的计算方法还是信息检索中常用的词频-逆文档频率（term frequency–inverse document frequency，简称tf-idf）。第j篇文章中与词典里第k个词对应的tf-idf为：

其中

是第k个词在文章j中出现的次数，而

是所有文章中包括第k个词的文章数量。

      最终第k个词在文章j中的权重由下面的公式获得：

做归一化的好处是不同文章之间的表示向量被归一到一个量级上，便于下面步骤的操作。

**2) Profile Learning**

假设用户u已经对一些item给出了他的喜好判断，喜欢其中的一部分item，不喜欢其中的另一部分。那么，这一步要做的就是通过用户u过去的这些喜好判断，为他产生一个模型。有了这个模型，我们就可以根据此模型来判断用户u是否会喜欢一个新的item。所以，我们要解决的是一个典型的有监督分类问题，理论上机器学习里的分类算法都可以照搬进这里。

下面我们简单介绍下CB里常用的学习算法——KNN：

对于一个新的item，最近邻方法首先找用户u已经评判过并与此新item最相似的k个item，然后依据用户u对这k个item的喜好程度来判断其对此新item的喜好程度。这种做法和CF中的item-based kNN很相似，差别在于这里的item相似度是根据item的属性向量计算得到，而CF中是根据所有用户对item的评分计算得到。

对于这个方法，比较关键的可能就是如何通过item的属性向量计算item之间的两两相似度。[2]中建议对于结构化数据，相似度计算使用欧几里得距离；而如果使用向量空间模型（VSM）来表示item的话，则相似度计算可以使用cosine。

**3) Recommendation Generation**

通过上一步的学习，会得到一个推荐列表，我们直接把这个列表中与用户属性最相关的n个item作为推荐返回给用户即可。

## Memory-based 协同过滤（CF）

俗话说“物以类聚、人以群分”，继续拿看电影这个例子来说，如果你喜欢《蝙蝠侠》、《碟中谍》、《星际穿越》、《源代码》等电影，另外有个人也都喜欢这些电影，而且他还喜欢《钢铁侠》，则很有可能你也喜欢《钢铁侠》这部电影。

所以说，当一个用户 A 需要个性化推荐时，可以先找到和他兴趣相似的用户群体G，然后把 G 喜欢的、并且 A 没有听说过的物品推荐给 A，这就是基于用户的系统过滤算法。

根据上述基本原理，我们可以将基于用户的协同过滤推荐算法拆分为两个步骤：

**1) 发现兴趣相似的用户**

通常用 Jaccard 公式或者余弦相似度计算两个用户之间的相似度。设 N(u) 为用户 u 喜欢的物品集合，N(v) 为用户 v 喜欢的物品集合，那么 u 和 v 的相似度是多少呢：

Jaccard 公式：

      余弦相似度：

假设目前共有4个用户： A、B、C、D；共有5个物品：a、b、c、d、e。用户与物品的关系（用户喜欢物品）如下图所示：

 如何一下子计算所有用户之间的相似度呢？为计算方便，通常首先需要建立“物品—用户”的倒排表，如下图所示：

然后对于每个物品，喜欢他的用户，两两之间相同物品加1。例如喜欢物品 a 的用户有 A 和 B，那么在矩阵中他们两两加1。如下图所示：

计算用户两两之间的相似度，上面的矩阵仅仅代表的是公式的分子部分。以余弦相似度为例，对上图进行进一步计算：

到此，计算用户相似度就大功告成，可以很直观的找到与目标用户兴趣较相似的用户。

**2) 推荐物品**

首先需要从矩阵中找出与目标用户 u 最相似的 K 个用户，用集合 S(u, K) 表示，将 S 中用户喜欢的物品全部提取出来，并去除 u 已经喜欢的物品。对于每个候选物品 i ，用户 u 对它感兴趣的程度用如下公式计算：

      其中 rvi 表示用户 v 对 i 的喜欢程度，在本例中都是为 1，在一些需要用户给予评分的推荐系统中，则要代入用户评分。

      举个例子，假设我们要给 A 推荐物品，选取 K = 3 个相似用户，相似用户则是：B、C、D，那么他们喜欢过并且 A 没有喜欢过的物品有：c、e，那么分别计算 p(A, c) 和 p(A, e)：

看样子用户 A 对 c 和 e 的喜欢程度可能是一样的，在真实的推荐系统中，只要按得分排序，取前几个物品就可以了。

商业案例

第四范式 - 先荐 -偏向文本推荐

<https://cloud.tencent.com/info/ec6072b72366431b032b3abe13b104e0.html> 通俗的讲解了推荐系统每个流程的必要性

# 2. 商品推荐

## 1. 机器学习方法

早年的推荐算法主要是各种单模型，例如逻辑回归、协同过滤、矩阵分解等等。后来推荐算法演化成了混合模型，例如 GBDT + LR , GBDT + FM 等。而随着深度学习的崛起，深度神经网络越来越深刻地影响了推荐系统领域的发展。

**标签库**

标签是联系用户与物品、内容以及物品、内容之间的纽带，也是反应用户兴趣的重要数据源。标签库的最终用途在于对用户进行行为、属性标记。是将其他实体转换为计算机可以理解的语言关键的一步。

标签库则是对标签进行聚合的系统，包括对标签的管理、更新等。

一般来说，标签是以层级的形式组织的。可以有一级维度、二级维度等。

标签的来源主要有：

* 已有内容的标签
* 网络抓取流行标签
* 对运营的内容进行关键词提取

对于内容的关键词提取，使用[结巴分词](https://github.com/fxsjy/jieba) + [TFIDF](http://www.ruanyifeng.com/blog/2013/03/tf-idf.html)即可。此外，也可以使用[TextRank](http://www.tuicool.com/articles/UZ77Z3)来提取内容关键词。

这里需要注意的一点是对于关联标签的处理，比如用户的标签是足球，而内容的标签是德甲、英超，那么用户和内容是无法联系在一起的。最简单的方式是人工设置关联标签，此外也可以使用word2vec一类工具对标签做聚类处理，构建主题模型，将德甲、英超聚类到**足球**下面。

**内容特征化**

也就是给内容打标签，目前两种方式：人工打标签，机器自动打标签。

针对机器自动打标签，需要采取机器学习方式实现，即针对一系列给定的标签，给内容选取其中匹配度最高的几个标签。这不同于通常的分类是聚类算法。采用分词+Word2vec来实现，过程如下：

1. 将文本语料进行分词，以空格,tab隔开都可以，使用结巴分词。
2. 使用word2vec训练词的相似度模型。
3. 使用tfidf提取内容的关键词A,B,C。
4. 遍历每一个标签，计算关键词与此标签的相似度之和。
5. 取出TopN相似度最高的标签即为此内容的标签。

此外，可以使用文本主题挖掘相关技术，对内容进行特征化。这也分为两种情况:

通用情况下，只是为了效果优化的特征提取，那么可以使用非监督学习的主题模型算法。如LSA、PLSI和GaP模型或者LDA模型。

在和业务强相关时，需要在业务特定的标签体系下给内容打上适合的标签。这时候需要使用的是监督学习的主题模型。如sLDA、HSLDA等。

**用户特征化**

用户特征化即为用户打标签，通过用户的行为日志和算法模型得到用户的每个标签的权重。

* 用户对内容的行为：点赞、不感兴趣、点击、浏览。对用户的反馈行为如点赞赋予权值1，不感兴趣赋予-1；对于用户的浏览行为，则可使用点击/浏览作为权值。
* 对内容发生的行为可以认为对此内容所带的标签的行为。
* 用户的兴趣是时间衰减的，即离当前时间越远的兴趣比重越低。时间衰减函数使用**1/[log(t)+1]**, t为事件发生的时间距离当前时间的大小。
* 要考虑到热门内容会干预用户的标签，需要对热门内容进行降权。使用click/pv作为用户浏览行为权值即可达到此目的。
* 此外，还需要考虑噪声的干扰，如标题党等。

**隐语义推荐**

有了内容特征和用户特征，可以使用[隐语义模型](http://blog.csdn.net/harryhuang1990/article/details/9924377)进行推荐。这里可以使用其简化形式，以达到实时计算的目的。

用户对于某一个内容的兴趣度(可以认为是CTR)：

其中i=1…N是内容c具有的标签，m(ci)指的内容c和标签i的关联度(可以简单认为是1),n(ui)指的是用户u的标签i的权重值,当用户不具有此标签时n(ui)=0，q©指的是内容c的质量，可以使用点击率(click/pv)表示。

<http://www.rowkey.me/blog/2016/04/07/up-recommend/?spm=a2c4e.11153940.blogcont465265.7.48647e09R9vlB5>  基于内容和用户画像的个性化推荐

<https://blog.csdn.net/harryhuang1990/article/details/9924377> LFM计算兴趣度，没理解

## 2. 深度学习

Huifeng Guo 等中国国内学者在 IJCAI 2017 发表了一篇题为《DeepFM: A Factorization-Machine based Neural Network for CTR Prediction 》的论文，讲解了如何将 FM 和深度学习模型进行融合之后进行推荐的算法。

DeepFM 主要有以下三个优点：

1. DeepFM 可以对低阶特征交互和高阶特征交互进行建模，不需要进行特征工程。
2. DeepFM 可以高效的进行训练，因为模型宽的部分和深的部分，不仅共享输入，也共享嵌入式向量。
3. 实验数据表明 DeepFM 可以在点击率预估问题上取得优秀的效果。

DeepFM 的输入数据为点击率预估常见的 (X, y) 元组，其中 X 是表示用户和物品的特征向量，可能包括非数值数据，y 是点击数据标签，y = 1 表示用户点击了物品，y = 0 表示用户没有点击物品。

DeepFM 的预测函数如下：

 , 其实是利用 sigmoid 函数对基于 FM 的预测和基于 DNN 的预测进行了融合。算法模型中的 FM 部分算法架构如下图所示：

DNN 部分的算法架构如下图所示：

在混合模型中，FM 和 DNN 共享同一个特征嵌入层。嵌入层的结构如下图所示：

作者随后在测试数据集合上对近年来的几种不同的点击率预估算法进行了测评，选择的测评标准包括 AUC 和 Log-loss ，测评结果如下图所示：

与 DeepFM 相比，其他几种点击率预估算法有以下缺点：

1.    FNN : FNN 是一个由 FM 初始化的前向神经网络。FM 预训练策略有如下两个问题：1). 嵌入层参数受 FM 影响较大 2). FM 预训练对算法效率有影响。另外，FM 只包含了高阶特征组合。

2.    PNN : PNN 及其变种 IPNN 和 OPNN 忽略了低阶特征的组合。

3.    宽深网络：宽身网络的 FM 部分需要人工处理特征工程。

DeepFM 设计思路简单，源于 2016 年 Google 的宽深网络方法但是效果出众。自推荐系统诞生以来，人们便设计了各种不同的模型融合方法。从修改主题模型的 Collaborative Topic Regression ，到基于 blending 的 GBDT + LR 再到后面的宽深网络方法， 给我们设计算法提供了不同的思路。算法本身是一个既要考虑模型，又要考虑数据和参数的学科，DeepFM 在如何设计模型层面给我们展示了很好的范例。

## 在线学习

增量学习和在线学习

增量学习是指在学得模型后，再接收到训练样例时，仅需根据新样例对模型进行更新，不必重新训练整个模型，并且先前学得的有效信息不会失效 .在线学习是指每获得一个新样本就进行一次模型史新显然，在线学习是增量学习的特例，而增量学习可视为"批模式"(batch-mode) 的在线学习（周志华）。

 增量学习主要表现在两个方面：一方面由于其无需保存历史数据，从而减少存储空间的占用；另一方面增量学习在当前的样本训练中充分利用了历史的训练结果，从而显著地减少了后续训练的时间。

         增量学习主要有两方面的应用:一是用于数据库非常大的情形,例如Web日志记录;二是用于流数据,因为这些数据随着时间在不断的变化,例如股票交易数据.另外在增量学习中,现有的增量学习算法大多采用决策树和神经网络算法实现的,它们在不同程度上具有以下两方面的缺点:一方面由于缺乏对整个样本集期望风险的控制,算法易于对训练数据产生过量匹配;另一方面,由于缺乏对训练数据有选择的遗忘淘汰机制,在很大程度上影响了分类精度。

最简单的方法之一是增量学习存储的所有数据允许重新训练。在另一个极端是数据的训练、一个实例接一个实例,一个流行的在线学习。采用在线学习算法的方法已经实现增量学习但没有考虑所有的学习问题,特别是学习的新类。

    随着人工智能和机器学习的发展,人们开发了很多机器学习算法。这些算法大部分都是批量学习(Batch Learning)模式,即假设在训练之前所有训练样本一次都可以得到,学习这些样本之后,学习过程就终止了,不再学习新的知识。然而在实际应用中,训练样本通常不可能一次全部得到,而是随着时间逐步得到的,并且样本反映的信息也可能随着时间产生了变化。如果新样本到达后要重新学习全部数据,需要消耗大量时间和空间,因此批量学习的算法不能满足这种需求。只有增量学习算法可以渐进的进行知识更新,且能修正和加强以前的知识,使得更新后的知识能适应新到达的数据,而不必重新对全部数据进行学习。增量学习降低了对时间和空间的需求,更能满足实际要求。虽然增量学习的研究已经有一定的历史,但由于不同的作者对其理解不同,至今增量学习也没有统一的定义。许多作者甚至将增量学习等同于在线学习(Online Learning)。这里,引用Robipolikar对增量学习算法的定义,即一个增量学习算法应同时具有以下特点:

1)可以从新数据中学习新知识;

2)以前已经处理过的数据不需要重复处理;

3)每次只有一个训练观测样本被看到和学习;

4)学习新知识的同时能保存以前学习到的大部分知识;

5)—旦学习完成后训练观测样本被丢弃;

6)学习系统没有关于整个训练样本的先验知识;

开展增量学习的研究具有以下两方面非常重要的意义:

1)随着数据库以及互联网技术的快速发展和广泛应用,社会各部门积累了海量数据,而且这些数据量每天都在快速增加。如何从这些数据中获取有用信息以及对数据进行分析和处理是一项艰苦的工作。而传统的批量学习方式是不能适应这种需求的,只有通过增量学习的方式才能有效解决这种需求。

2)通过对增量学习模型的研究,能够使我们从系统层面上更好地理解和模仿人脑的学习方式和生物神经网络的构成机制,为开发新计算模型和有效学习算法提供技术基础。

## 4. 评测方式

以图书推荐系统为例：

首先推荐系统要满足用户的需求，要尽可能地覆盖各种图书，要能收集到高质量的用户反馈，增加用户和图书网站的交互，提高网站的收入。要能够准确预测用户的行为，还要扩展用户的视野，帮助用户发现那些他们可能会感兴趣的但却不那么容易发现的东西。本文主要从用户，网站，内容提供方提出不同的指标。

三种推荐系统实验方法

1、离线实验（offline experiment）

实施步骤：

（1）将从日志系统收集来的用户行为数据，生成为一个标准的数据集；

（2）按照一定的规则，将数据集分成训练集和测试集两个部分；

（3）在训练集上训练用户兴趣模型，在测试集上进行测试；

（4）利用定义好的离线指标评测算法，在测试集上进行结果预测。

优点：所有实验在数据集（从系统日志中提取而来）上完成，对实际系统和用户参与依赖度低，方便快捷；

缺点：对商业上关注的指标获取能力弱；

2、用户调查（user study）

用户调查，是一种通过分析被调查用户（真实的用户），在被测推荐系统上完成任务时的行为和回答问题的情况，来了解测试系统性能的一种实验方法。它旨在为上线测试提供准备工作，以防范上线测试所潜在的降低用户满意度的问题。

优点：在离线测试解决不了的“用户主观感受相关指标”的获取上，有着优越的性能；风险易控。

缺点：实验成本高，难组织大规模测试；双盲实验设计困难，影响测评结果。

3、在线实验（online experiment）

这里的在线实验方法，偏指AB测试方法。

实施步骤：

（1）通过一定规则将用户随机分组；

（2）对不同组的用户采用不同算法；

（3）统计不同组用户的不同评测指标，以比较不同的算法。

优点：公平获得不同算法实际在线的性能指标，包括商业上关注的指标。

缺点：试验周期较长；设计AB测试系统的工程量大，且流量切分设计一般必不可少。

评测指标：

1、用户满意度：

基本在实际操作中不可用，首先这是一个相对主观的指标，一般依靠用户调研获取，而用户基本也不知道自己想要什么，最多谈一个表面上的感受，参考价值不大。其次最关心推荐系统好坏的往往是这个项目的PM，用户满意度的概念很容易被偷换成PM满意度或老板满意度。以前有个朋友在游戏公司，结果各个需求都以老板的满意为准，最后倒闭了。所以在实际情况下，会用衡量准确度的客观指标来参考，比如通过点击率的统计看用户对推出内容的满意程度。

2、预测准确度：

应用中，按场景可以将推荐系统分为TopN推荐和评分推荐两种，评分推荐一般用RMSE（ 均方根误差）和MAE(绝对平均)误差计算。其中RMSE加大了对预测不准的项的惩罚，评测更加严格。而对于另一种TopN推荐的预测准确率，一般用Recall（召回率）和Precision（准确率）来评测，需要的时候还可以计算多对准确率和召回率，然后画出PR曲线进行评测。

3、覆盖率：

覆盖率用于描述系统对于长尾物品的发掘能力，简单说就是对所有用户推荐的物品能够包括的物品种类越多，覆盖率越大，这样就引出了覆盖率最简单的一种定义方式：系统能够推荐的物品占总物品集合的比例。但是这样的计算方法没有考虑推荐列表中每种物品出现的频率，如果列表中不但出现的比例大，而且每种物品出现的频率也相近，那么对长尾的挖掘能力越好。通过物品在推荐列表中出现次数定义覆盖率的方式有信息熵和基尼系数两种。这两个指标的计算又会涉及到流行度的计算，一种商品的流行度就是它和多少用户发生了用户行为。

长尾效应如下图：

长尾效应，英文名称Long Tail Effect。“头”（head）和“尾”（tail）是两个统计学名词。正态曲线中间的突起部分叫“头”；两边相对平缓的部分叫“尾”。从人们需求的角度来看，大多数的需求会集中在头部，而这部分我们可以称之为流行，而分布在尾部的需求是个性化的，零散的小量的需求。而这部分差异化的、少量的需求会在需求曲线上面形成一条长长的“尾巴”，而所谓长尾效应就在于它的数量上，将所有非流行的市场累加起来就会形成一个比流行市场还大的市场。

简单来说就是热门的商品和 冷门商品贫富差距太大

4、多样性：

推荐列表中物品的两两不相似性，可以用相似度来定义，相似度越高，多样性就越低。

5、新颖性：

推出那些用户之前没有接触过的内容。

6、惊喜度：

推荐一个和用户兴趣一点关系没有但是用户觉得还很不错的内容。

7、信任度：

要让用户认同推荐系统的推荐结果和推荐理由。

8、实时性：

产生了新的用户行为之后能不能实时更新推荐里列表

加入了新的物品能不能立即推荐给用户（物品的冷启动问题）

9、健壮性：

又称鲁棒性，抗击作弊的能力

可以通过模拟\*\*\*进行评测

提高健壮性的方法：

1、设计推荐系统是使用代价高的用户行为

2、使用数据训练模型前进行\*\*\*检测，对数据进行清理

# 文本推荐

理论

文本推荐是推荐系统的一个应用场景，不仅可以用来进行新闻推荐，也可以应用在问答推荐等其他产品。国外有一些公司甚至将文本推荐最为主营业务，比如一家总部位于美国的公司Outbrain，就专门为各大网站提供文本服务。

1. 谷歌新闻推荐系统

2010 年 Jiahui Liu 等人发表的论文 Personalized News Recommendation based on Click Behavior 是文本推荐领域的经典之作。作者用非常简单的方式设计了 Google 新闻推荐系统。该系统采用的方法是简单的协同过滤和 SVD 分解，外加对于用户个人和群体喜好的历史统计信息。这种方式一方面解决了推荐本身这个问题，另一方面解决了新闻中存在的马太效应问题。

1. 纽约时报推荐系统

纽约时报的推荐系统源于普林斯顿大学和微软的研究人员对于文本推荐的研究工作。2011 年 Chong Wang 等人在论文 Collaborative Topic Modeling for Recommending Scientific Articles提出了 Collaborative Topic Regression 模型。该模型结合了协同过滤和主题模型，概率图模型如图1所示。

图1. Collaborative Topic Regression概率图模型

Collaborative Topic Regression 及其后续的研发工作成就了纽约时报的新闻推荐系统。

1. 百度知道问答推荐系统

2012年和2014年年百度分别在 ACM RecSys 和 ACM SAC 上发表了两篇论文，描述了百度知道问答推荐系统是如何构建的。百度知道的推荐系统采用了混合模型，结合了线性模型和非线性模型，利用了自然语言处理中关键词提取等技术。

图2. 百度知道问答推荐系统架构

1. 协同深度学习

2015年Hao Wang等人在数据挖掘顶级会议KDD上发表文章Collaborative Deep Learning for Recommender Systems。作者结合了协同过滤和stacked denoising autoencoder设计了文本推荐的混合模型。混合模型的概率图模型如下图：

                     协同深度学习的概率图模型

1. 基于GRU的深度文本模型

2016年Trapit Bansal等人在ACM RecSys会议上发表论文Ask the GRU:Muti-task Learning for Deep Text Recommendation，提出基于GRU的文本推荐模型。如下图：

基于 GRU文本推荐模型的深度神经网络结构

文本推荐作为推荐系统的一个重要分支，依托于大量的机器学习和自然语言处理技术，广泛应用于新闻和问答等领域。而近年来深度学习的崛起，也对文本推荐起到了推波助澜的作用。

工程：

<https://www.kaggle.com/c/outbrain-click-prediction> kaggle - outbrain公司文本点击预测比赛

二．国内外研究

国内外研究(知网):

电商用户数据分析研究