达观数据
DATA GRAND

达观数据NLP特刊

——从原理到实践——

文本挖掘引擎实践篇　垂直搜索引擎实践篇　智能推荐引擎实践篇

前　言

本刊旨在为广大技术爱好者分享一线的 NLP 实战经验。

本刊以文本挖掘作为“前锋内容”，从达观的工作实践出发，从原理到实践，进一步阐述中文分词技术、半监督学习技术以及如何打造中文 NER 系统，从而引出其上游技术——推荐引擎和搜索引擎。

垂直搜索引擎实践篇以科普性质的倒排索引解读开启了搜索的实践之谈，着重讲述了垂直搜索引擎应用中的搜索排序、搜索词自动纠错以及目前应用火热的智能问答系统。

推荐系统实践篇作为压轴内容，究其原因是文本挖掘和搜索引擎都是其良好应用的关键一环。该篇首先深度分析并讲述达观数据如何攻克推荐系统较难的冷启动环节，详实有效。而后对基于用户历史行为的推荐应用进行深入浅出的讲解，之后强调了推荐系统在应用过程中多模型融合的方法和价值，最终以推荐系统的实践和优化来进行全部内容的收尾，内容环环相扣，缺一不可。

"Per Aspera ad Astra" —— 循此苦旅，以达天际。达观相信术业有专攻，专注于文本技术，也乐于同各位依然奋斗在技术研究与实践道路上的同道者们，分享经验，共同进步。达观数据在此与诸君共勉。

关于达观数据

Q1：我们是谁？

达观数据是领先的文本智能处理专家，是一家专注于企业知识管理和文字语义理解的国家高新技术企业。达观数据为企业提供完善的文本挖掘、知识图谱、搜索引擎和个性化推荐等文本智能处理技术服务，是一家将自动语义分析技术应用于企业数据化运营的人工智能公司。

Q2：我们的实践团队如何？

达观数据核心团队在 AI 领域享有声望。前腾讯文学高级总监陈运文博士担任 CEO，核心骨干来自盛大、腾讯等著名互联网公司，由拥有复旦、上交大、中科大等名校文本挖掘相关专业博士及硕士研究生学历的顶尖人才组建，技术团队历经十余年的科研深耕，在产品研发方面有丰富的经验和积累，多次斩获国际数据挖掘最高级别竞赛 ACM KDD 和 CIKM 的世界冠亚军大奖，拥有 30 多项国家发明专利，并与复旦大学建有技术联合实验室，是中国知名的 AI 技术团队。

Q3：我们的业务涉及哪些行业和客户？

现已积累华为、招商银行、中国平安、中兴、京东、顺丰、中国移动、和讯财经等数百家企业客户的成功服务经验，覆盖金融、科技、制造、法律、电商、视频、传媒等行业，通过提升企业文本的自动化处理能力，加快企业智能化转型速度、有效提升企业运营效率和经营业绩。

希望更深入了解达观数据或自然语言处理相关问题咨询，请移步知乎（ID: 达观数据 / 陈运文），由专业的技术人员为您解答。更多公司详细信息，敬请访问达观数据公司主页 www.datagrand.com 或关注达观数据微信公众号 (Datagrand_)。

目　录

【文本挖掘引擎实践篇】

【垂直搜索引擎实践篇】

【智能推荐引擎实践篇】

文本挖掘引擎实践篇

利用自然语言处理技术，让计算机具备文字阅读能力，帮助客户自动化处理海量文本数据，提升文字处理效率和文本挖掘深度，降低人工成本

【本篇关键词】NLP；中文分词；文本过滤；指纹算法；命名实体识别

达观数据基于 Deep Learning 的中文分词尝试

高翔　达观数据联合创始人

1. 现有分词方法介绍

自然语言处理（NLP，Natural Language Processing）是一个信息时代最重要的技术之一，简单来讲，就是让计算机能够理解人类语言的一种技术。在其中，分词技术是一种比较基础的模块。对于英文等拉丁语系的语言而言，由于词之间有空格作为词边际表示，词语一般情况下都能简单且准确地提取出来。而中文日文等文字，除了标点符号之外，字之间紧密相连，没有明显的词边界，因此很难将词提取出来。

分词的意义非常大，在中文中，单字作为最基本的语义单位，虽然也有自己的意义，但表意能力较差，意义较分散。而词的表意能力更强，能更加准确地描述一个事物，因此在自然语言处理中，通常情况下词（包括单字成词）是最基本的处理单位。

在具体的应用上，比如在常用的搜索引擎中，term 如果是词粒度的话，不仅能够减少每个 term 的倒排列表长度，提升系统性能，并且召回的结果相关性高更准确。比如搜索 query“的确”，如果是单字切分的话，则有可能召回“你讲的确实在理”这样的 doc。

分词方法大致分为两种：基于词典的机械切分，基于统计模型的序列标注切分两种方式。

1.1 基于词典的方法

基于词典的方法本质上就是字符串匹配的方法，将一串文本中的文字片段和已有的词典进行匹配，如果匹配到，则此文字片段就作为一个分词结果。但是基于词典的机械切分会遇到多种问题，最为常见的包括歧义切分问题和未登录词问题。

1.1.1 歧义切分

歧义切分指的是通过词典匹配给出的切词结果和原来语句所要表达的意思不相符或差别较大，在机械切分中比较常见。比如下面的例子：“结婚的和尚未结婚的人”，通过机械切分的方式，会有两种切分结果：1.“结婚 / 的 / 和 / 尚未 / 结婚 / 的 / 人”；2.“结婚 / 的 / 和尚 / 未 / 结婚 / 的 / 人”。可以明显看出，

第二种切分是有歧义的，单纯的机械切分很难避免这样的问题。

1.1.2 未登录词识别

未登录词识别也称作新词发现，指的是词没有在词典中出现，比如一些新的网络词汇，如“网红”，“走你”；一些未登录的人名，地名；一些外语音译过来的词等等。基于词典的方式较难解决未登录词的问题，简单的 case 可以通过加词典解决，但是随着字典的增大，可能会引入新的 bad case，并且系统的运算复杂度也会增加。

1.1.3 基于词典的机械分词改进方法

为了解决歧义切分的问题，在中文分词上有很多优化的方法，常见的包括正向最大匹配，逆向最大匹配，最少分词结果，全切分后选择路径等多种算法。

1.1.4 最大匹配方法

正向最大匹配指的是从左到右对一个字符串进行匹配，所匹配的词越长越好，比如“中国科学院计算研究所”，按照词典中最长匹配原则的切分结果是：“中国科学院 / 计算研究所”，而不是“中国 / 科学院 / 计算 / 研究所”。但是正向最大匹配也会存在一些 bad case，常见的例子如：“他从东经过我家”，使用正向最大匹配会得到错误的结果：“他 / 从 / 东经 / 过 / 我 / 家”。

逆向最大匹配的顺序是从右向左倒着匹配，如果能匹配到更长的词，则优先选择，上面的例子“他从东经过我家”逆向最大匹配能够得到正确的结果“他 / 从 / 东 / 经过 / 我 / 家”。但是逆向最大匹配同样存在 bad case：“他们昨日本应该回来”，逆向匹配会得到错误的结果“他们 / 昨 / 日本 / 应该 / 回来”。

针对正向逆向匹配的问题，将双向切分的结果进行比较，选择切分词语数量最少的结果。但是最少切分结果同样有 bad case，比如“他将来上海”，正确的切分结果是“他 / 将 / 来 / 上海”，有 4 个词，而最少切分结果“他 / 将来 / 中国”只有 3 个词。

1.1.5 全切分路径选择方法

全切分方法就是将所有可能的切分组合全部列出来，并从中选择最佳的一条切分路径。关于路径的选择方式，一般有 n 最短路径方法，基于词的 n 元语法模型方法等。

n 最短路径方法的基本思想就是将所有的切分结果组成有向无环图，每个

切词结果作为一个节点，词之间的边赋予一个权重，最终找到权重和最小的一条路径作为分词结果。

基于词的 n 元语法模型可以看作是 n 最短路径方法的一种优化，不同的是，根据 n 元语法模型，路径构成时会考虑词的上下文关系，根据语料库的统计结果，找出构成句子最大模型概率。一般情况下，使用 unigram 和 bigram 的 n 元语法模型的情况较多。

1.2 基于序列标注的分词方法

针对基于词典的机械切分所面对的问题，尤其是未登录词识别，使用基于统计模型的分词方式能够取得更好的效果。基于统计模型的分词方法，简单来讲就是一个序列标注问题。

在一段文字中，我们可以将每个字按照他们在词中的位置进行标注，常用的标记有以下四个 label：B，Begin，表示这个字是一个词的首字；M，Middle，表示这是一个词中间的字；E，End，表示这是一个词的尾字；S，Single，表示这是单字成词。分词的过程就是将一段字符输入模型，然后得到相应的标记序列，再根据标记序列进行分词。举例来说：“达观数据是文本智能处理专家”，经过模型后得到的理想标注序列是：“BMMESBEBEBEBE”，最终还原的分词结果是“达观数据 / 是 / 文本 / 智能 / 处理 / 专家”。

在 NLP 领域中，解决序列标注问题的常见模型主要有 HMM 和 CRF。

1.2.1 HMM

HMM（Hidden Markov Model）隐马尔科夫模型应用非常广泛，基本的思想就是根据观测值序列找到真正的隐藏状态值序列。在中文分词中，一段文字的每个字符可以看作是一个观测值，而这个字符的词位置 label（BEMS）可以看作是隐藏的状态。使用 HMM 的分词，通过对切分语料库进行统计，可以得到模型中 5 大要要素：起始概率矩阵，转移概率矩阵，发射概率矩阵，观察值集合，状态值集合。在概率矩阵中，起始概率矩阵表示序列第一个状态值的概率，在中文分词中，理论上 M 和 E 的概率为 0。转移概率表示状态间的概率，比如 B->M 的概率，E->S 的概率等。而发射概率是一个条件概率，表示当前这个状态下，出现某个字的概率，比如 p(人 |B) 表示在状态为 B 的情况下人字的概率。

有了三个矩阵和两个集合后，HMM 问题最终转化成求解隐藏状态序列最大值的问题，求解这个问题最长使用的是 Viterbi 算法，这是一种动态规划算法，

具体的算法可以参考维基百科词条，在此不详细展开。

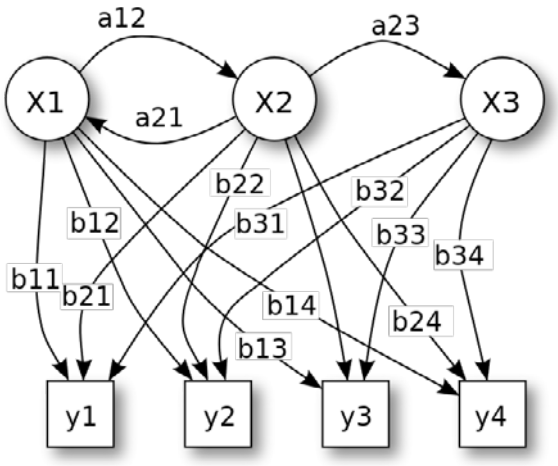


图 1.1 HMM 模型示意图

1.2.2 CRF

CRF（Conditional random field，条件随机场）是用来标注和划分结构数据的概率化结构模型，通常使用在模式识别和机器学习中，在自然语言处理和图像处理等领域中得到广泛应用。和 HMM 类似，当对于给定的输入观测序列 X 和输出序列 Y，CRF 通过定义条件概率 P(Y|X)，而不是联合概率分布 P(X，Y) 来描述模型。CRF 算法的具体算法可以参考维基百科词条。

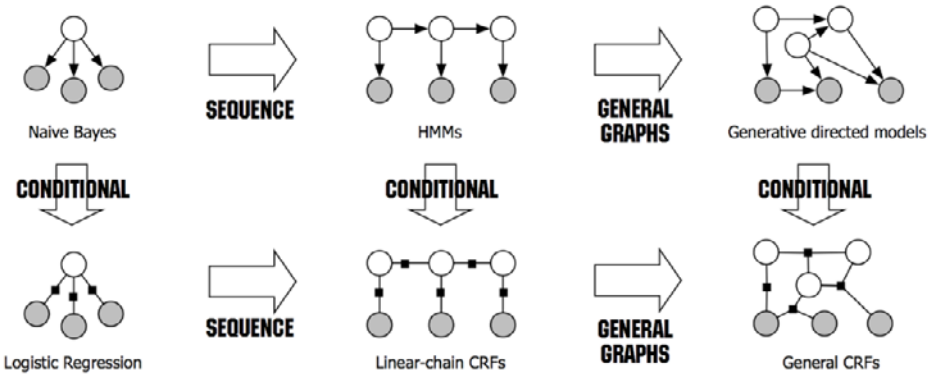


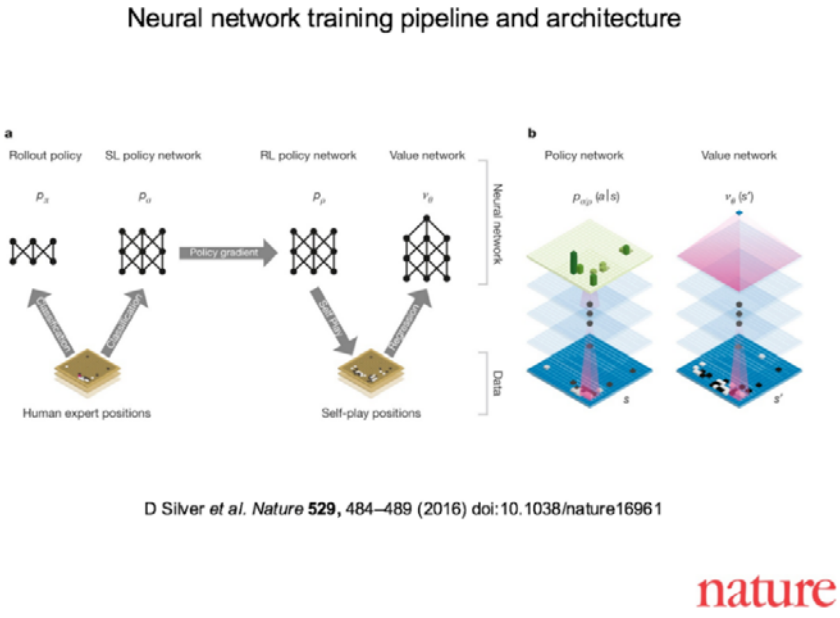
图 1.2 不同概率模型之间的关系及演化图

在实际应用中有很多工具包可以使用，比如 CRF++，CRFsuite，SGD，Wapiti 等，其中 CRF++ 的准确度较高。在分词中使用 CRF++ 时，主要的工作是特征模板的配置。CRF++ 支持 unigram，bigram 两种特征，分别以 U 和

B 开头。举例来讲 U00：%x[-2，0] 表示第一个特征，特征取值是当前字的前方第二个字，U01：%x[-1，0] 表示第二个特征，特征取值当前字前一个字，U02：%x[0，0] 表示第三个特征，取当前字，以此类推。特征模板可以支持多种特征，CRF++ 会根据特征模板提取特征函数，用于模型的建立和使用。特征模板的设计对分词效果及训练时间影响较大，需要分析尝试找到适用的特征模板。

2. 深度学习介绍

随着 AlphaGo 的大显神威，Deep Learning （深度学习）的热度进一步提高。深度学习来源于传统的神经网络模型。传统的神经网络一般由输入层，隐藏层，输出层组成，其中隐藏层的数目按需确定。深度学习可以简单的理解为多层神经网络，但是深度学习的却不仅仅是神经网络。深度模型将每一层的输出作为下一层的输入特征，通过将底层的简单特征组合成为高层的更抽象的特征来进行学习。在训练过程中，通常采用贪婪算法，一层一层地训练，比如在训练第 k 层时，固定训练好的前 k−1 层的参数进行训练，训练好第 k 层之后的以此类推进行一层层训练。



D Silver *et al. Nature* **529,** 484–489 (2016) doi:10.1038/nature16961

nature

图 2.1 AlphaGo 的神经网络模型的训练过程及架构

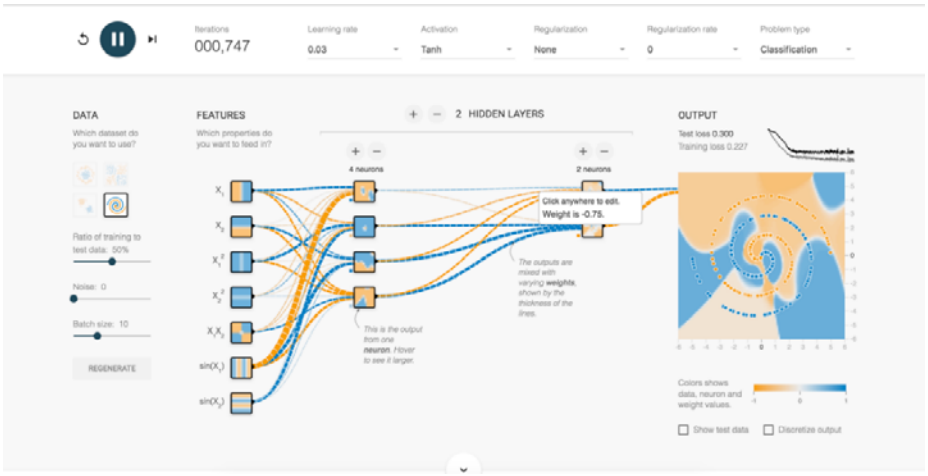


图 2.2 Google Tensorflow 官网的神经网络演示示意图

深度学习在很多领域都有所应用，在图像和语音识别领域中已经取得巨大的成功。从 2012 年开始，LSVRC（Large Scale Visual Recognition Challenge）比赛中，基于 Deep Learning 的计算框架一直处于领先地位。2015 年 LSVRC（http：//www.image-net.org/challenges/LSVRC/2015/results）的比赛中，微软亚洲研究院（MSRA）在图像检测（Object detection），图像分类定位（Object Classification+localization）上夺冠，他们使用的神经网络深达 152 层。

2.1 深度学习在 NLP 中的应用

在自然语言处理上，深度学习在机器翻译、自动问答、文本分类、情感分析、信息抽取、序列标注、语法解析等领域都有广泛的应用。2013 年末 google 发布的 word2vec 工具，可以看做是深度学习在 NLP 领域的一个重要应用，虽然 word2vec 只有三层神经网络，但是已经取得非常好的效果。通过 word-2vec，可以将一个词表示为词向量，将文字数字化，更好的让计算机理解。使 word2vec 模型，我们可以方便地找到同义词或联系紧密的词，或者意义相反的词等。

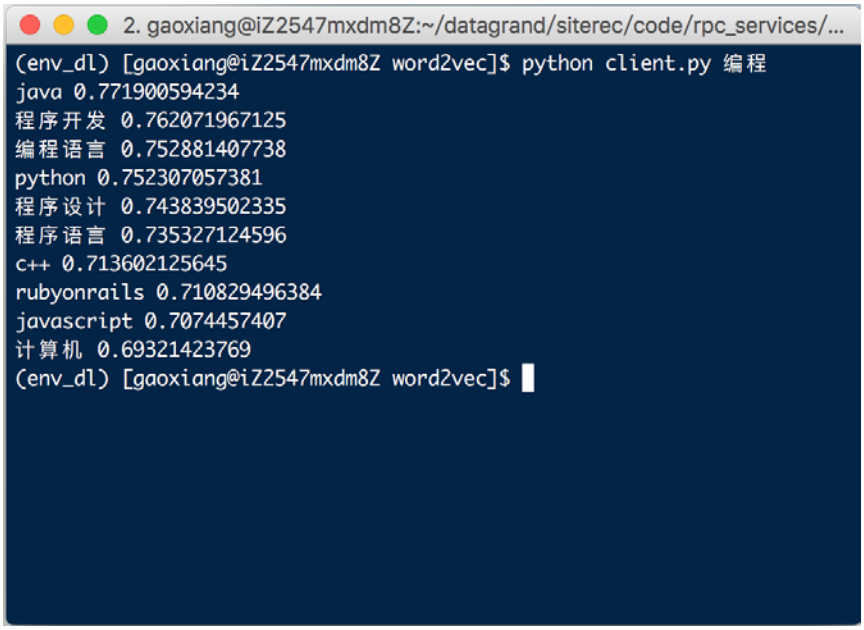


图 2.3 基于微信数据制作的 word2vec 模型测试：编程

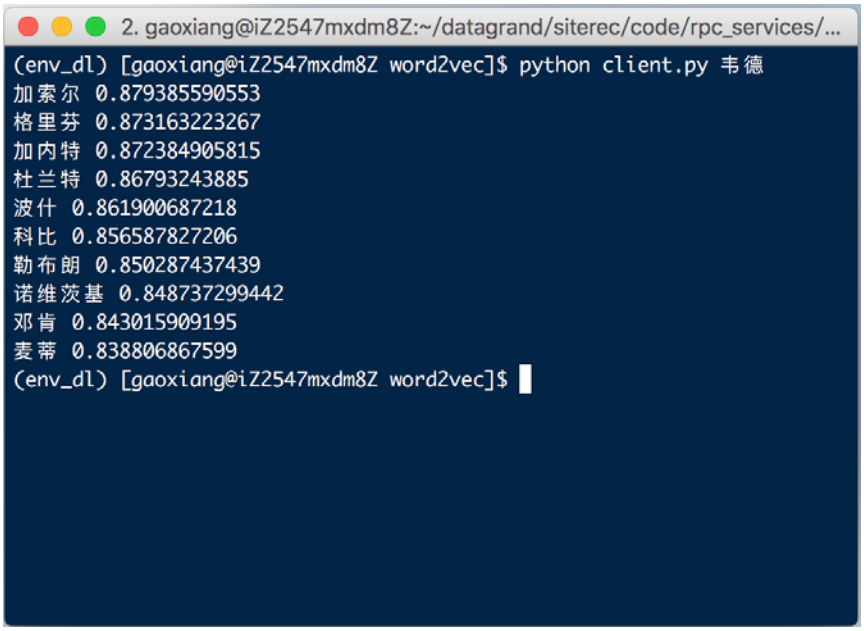


图 2.4 基于微信数据制作的 word2vec 模型测试：韦德

2.2 词向量介绍

词向量的意思就是通过一个数字组成的向量来表示一个词，这个向量的构成可以有很多种。最简单的方式就是所谓的 one-hot 向量。假设在一个语料集合中，一共有 n 个不同的词，则可以使用一个长度为 n 的向量，对于第 i 个词（i=0…n-1），向量 index=i 处值为 1 外，向量其他位置的值都为 0，这样就可

以唯一的通过一个 [0，0，1，…，0，0] 形式的向量表示一个词。one-hot 向量比较简单也容易理解，但是有很多问题。比如当加入新词时，整个向量的长度会改变，并且存在维数过高难以计算的问题，以及向量的表示方法很难体现两个词之间的关系，因此一般情况下 one-hot 向量较少的使用。

如果考虑到词和词之间的联系，就要考虑词的共现问题。最简单的是使用基于文档的向量表示方法来给出词向量。基本思想也很简单，假设有 n 篇文档，如果某些词经常成对出现在多篇相同的文档中，我们则认为这两个词联系非常紧密。对于文档集合，可以将文档按顺编号（i=0…n-1），将文档编导作为向量索引，这样就有一个 n 维的向量。当一个词出现在某个文档 i 中时，向量 i 处值为 1，这样就可以通过一个类似 [0，1，0，…，1，0] 形式的向量表示一个词。基于文档的词向量能够很好地表示词之间的关系，但是向量的长度和语料库的大小相关，同样会存在维度变化问题。

考虑用一个固定窗口大小的文本片段来解决维度变化问题，如果在这样的片段中，两个词出现了，就认为这两个词有关。举例来讲，有以下三句话：“我 \ 喜欢 \ 你”，“我 \ 爱 \ 运动”，“我 \ 爱 \ 摄影”，如果考虑窗口的大小为 1，也就是认为一个词只和它前面和后面的词有关，通过统计共现次数，我们能够得到下面的矩阵。

*	我	喜欢	你	爱	运动	摄影
我	0	1	0	2	0	0
喜欢	1	0	1	0	0	0
你	0	1	0	0	0	0
爱	2	0	0	0	1	1
运动	0	0	0	1	0	0
摄影	0	0	0	1	0	0

图 2.5 基于文本窗口共现统计出来的矩阵

可以看到这是一个 n*n 的对称矩阵 X，这个矩阵的维数会随着词典数量的增加而增大，通过 SVD（Singular Value Decomposition，奇异值分解），我们可以将矩阵维度降低，但仍存在一些问题： 矩阵 X 维度经常改变，并且由

于大部分词并不是共现而导致的稀疏性，矩阵维度过高计算复杂度高等问题。

Word2vec 是一个多层的神经网络，同样可以将词向量化。在 Word2vec 中最重要的两个模型是 CBOW（Continuous Bag-of-Word）模型和 Skip-gram（Continuous Skip-gram）模型，两个模型都包含三层： 输入层，投影层，输出层。CBOW 模型的作用是已知当前词 Wt 的上下文环境（Wt-2，Wt-1，Wt+1，Wt+2）来预测当前词，Skip-gram 模型的作用是根据当前词 Wt 来预测上下文（Wt-2，Wt-1，Wt+1，Wt+2）。在模型求解中，和一般的机器学习方法类似，也是定义不同的损失函数，使用梯度下降法寻找最优值。Word2vec 模型求解中，使用了 Hierarchical Softmax 方法和 Negative Sampling 两种方法。通过使用 Word2vec，我们可以方便地将词转化成向量表示，让计算机和理解图像中的每个点一样，数字化词的表现。

2.3 LSTM 模型介绍

深度学习有很多种不同类型的网络，在图像识别领域，CNN（Convolutional Neural Network，卷积神经网络）使用的较多，而在 NLP 领域，考虑到上下文的 RNN（Recurrent Neural Networks，循环神经网络）取得了巨大的成功。

在传统的神经网络中，从输入层到隐藏层到输出层，层之间是全连接的，但是每层内部的节点之间是无连接的。因为这样的原因，传统的神经网络不能利用上下文关系， 而在自然语言处理中，上下文关系非常重要，一个句子中前后词并不独立，不同的组合会有不同的意义，比如”优秀”这个词，如果前面是”不”字，则意义完全相反。

RNN 则考虑到网络前一时刻的输出对当前输出的影响，将隐藏层内部的节点也连接起来，即当前时刻一个节点的输入除了上一层的输出外，还包括上一时刻隐藏层的输出。RNN 在理论上可以储存任意长度的转态序列，但是在不同的场景中这个长度可能不同。比如在词的预测例子中： 1，“他是亿万富翁，他很？”；2，“他的房子每平米物业费 40 元，并且像这样的房子他有十几套，他很？”。从这两个句子中我们已经能猜到？代表“有钱”或其他类似的词汇，但是明显，第一句话预测最后一个词时的上线文序列很短，而第二段话较长。如果预测一个词汇需要较长的上下文，随着这个距离的增长，RNN 将很难学到这些长距离的信息依赖，虽然这对我们人类相对容易。在实践中，已被证明使用最广泛的模型是 LSTM（Long Short-Term Memory，长短时记忆）很好的解决了这个问题。

LSTM 最早由 Hochreiter 及 Schmidhuber 在 1997 年的论文中提出。首先 LSTM 也是一种 RNN，不同的是 LSTM 能够学会远距离的上下文依赖，能够存储较远距离上下文对当前时间节点的影响。

所有的 RNN 都有一串重复的神经网络模块。对于标准的 RNN 这个模块都比较简单，比如使用单独的 tanh 层。LSTM 拥有类似的结构，但是不同的是，LSTM 的每个模块拥有更复杂的神经网络结构： 4 层相互影响的神经网络。在 LSTM 每个单元中，因为门结构的存在，对于每个单元的转态，使得 LSTM 拥有增加或减少信息的能力。

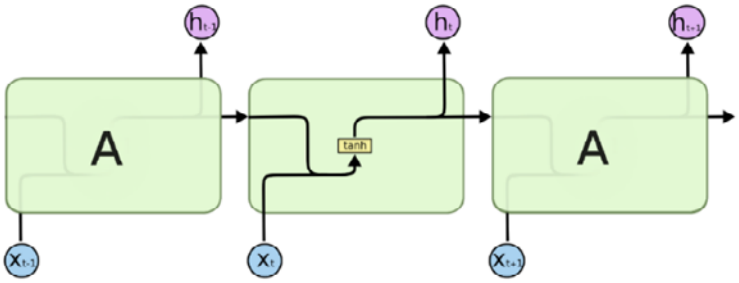


图 2.6 标准 RNN 模型中的重复模块包括 1 层结构

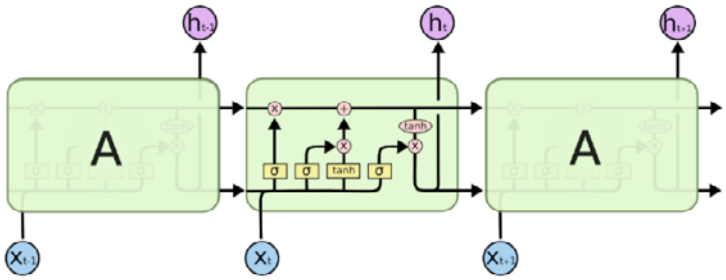


图 2.7 LSTM 模型中的重复模块包括 4 层结构

3. 深度学习库 keras 介绍

Keras（http://keras.io）是一个非常易用的深度学习框架，使用 python 语言编写，是一个高度模块化的神经网络库，后端同时支持 Theano 和 TensorFlow。而 Theano 和 TensorFlow 支持 GPU，因此使用 keras 可以使用 GPU 加速模型训练。

Keras 中包括了构建模型常用的模块，如 Optimizers 优化方法模块，Activations 激活函数模块，Initializations 初始化模块，Layers 多种网络层模块等，可以非常方便快速的搭建一个网络模型，使得开发人员可以快速上

手，并将精力放在模型设计而不是具体实现上。常见的神经网络模型如 CNN，RNN 等，使用 keras 都可以很快搭建出来，开发人员只需要将数据准备成 keras 需要的格式丢进网络训练即可。如果对 keras 中自带的 layer 有更多的需求，keras 还可以自己定制所需的 layer。

3.1 Keras 在 NLP 中的应用

Keras 项目中的 example 自带了多个示例，包括经典的 mnist 手写识别测试等，其中和 NLP 相关的示例有很多，比如基于 imdb 数据的情感分析、文本分类、序列标注等。其中 lstm_text_generation.py 示例可以用来参考设计序列标注问题，这个示例试图通过 LSTM 学习尼采的作品，通过序列标注的思想来训练一个文本生成器模型。下面着重看一下两个关键点：模型数据格式及模型设计。

3.1.1 训练数据准备

```
maxlen = 40
step = 3
sentences = []
next_chars = []
for i in range(0, len(text) - maxlen, step):
    sentences.append(text[i: i + maxlen])
    next_chars.append(text[i + maxlen])
print('nb sequences:', len(sentences))
```

这段代码是数据准备的情况。将尼采全文进行数据切割，每 40 个字符为一个片段，将紧接这个片段的字符作为预测值，来进行训练。字符片段的间隔为 3。

3.1.2 模型设计

```
model = Sequential()
model.add(LSTM(512, return_sequences=True, input_shape=
(maxlen, len(chars))))
model.add(Dropout(0.2))
model.add(LSTM(512, return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(len(chars)))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop')
```

在模型设计上，主要是使用了两层 LSTM，每层的输出维度为 512，并在每层 LSTM 后面加入了 Dropout 层，来防止过拟合。整个模型的输入维度是字符类别的个数，输入字符串长度是 40，模型的输出维度也是字符类别长度。整个模型表达的意思是每输入 40 个字符，就会从模型中输出一个预测的字符。因

为 LSTM 的对长依赖 term 的记忆性，因此在上下文很长（40 个字符）的情况下也可以表现的很好。

4. 基于深度学习方式的分词尝试

基于上面的知识，可以考虑使用深度学习的方法进行中文分词。分词的基础思想还是使用序列标注问题，将一个句子中的每个字标记成 BEMS 四种 label。模型整的输入是字符序列，输出是一个标注序列，因此这是一个标准的 sequence to sequence 问题。因为一个句子中每个字的上下文对这个字的 label 类型影响很大，因此考虑使用 RNN 模型来解决。

4.1 环境介绍

测试硬件是 Macbook Pro 2014 Mid 高配版，带 Nvidia GT 750M GPU，虽然 GPU 性能有限，但通过测试性能还是强过 mac 自带的 i7 CPU。使用 GPU 进行模型运算，需要安装 Nvidia 的 cuda 相关程序及 cuDNN 库，会有较大的性能提升。软件方面使用 python2.7，安装好了 keras，theano 及相关库。关于 keras 使用 GPU 训练的环境搭建问题，可以参考这篇文章（Run Keras on Mac OS with GPU，http：//blog.wenhaolee.com/run-keras-on-mac-os-with-gpu/）。

4.2 模型训练

模型训练使用的是经典的 bakeoff2005 中的微软研究院的切分语料，将其中的 train 部分拿过来做训练，将 test 作为最终的测试。

4.2.1 训练数据准备

首先，将训练样本中出现的所有字符全部映射成对应的数字，将文本数字化，形成一个字符到数据的映射。在分词中，一个词的 label 受上下文影响很大，因此参考之前提到的 lstm_text_generation.py 示例，我们将一个长度为 n 个字符的输入文本处理成 n 个长度为 k 的向量，k 为奇数。举例来说，当 k=7 时，表示考虑了一个字前 3 个字和后三个字的上下文，将这个七个字作为一个输入，输出就是这个字的 label 类型（BEMS）。

4.2.2 基础模型建立

参考 lstm_text_generation.py 中的模型搭建方式，我们采用一层的 LSTM

构建网络，代码如下：

```
def get_lstm_model(input_dim, input_length, nb_classes,
                   hidden_node=100):
    model = Sequential()
    model.add(Embedding(input_dim, hidden_node,
                        input_length=input_length))
    model.add(LSTM(hidden_node))
    model.add(Dropout(0.5))
    model.add(Dense(nb_classes))
    model.add(Activation('softmax'))
    model.compile(loss='categorical_crossentropy',
                  optimizer='rmsprop',
                  metrics=["accuracy"])
    return model
```

其中，输入的维度 input_dim 是字符类别总数，hidden_node 是隐藏层的结点个数。在上面的模型中，第一层输入层 Embedding 的作用是将输入的整数向量化。在现在这个模型中，输入是一个一维向量，里面每个值是字符对应的整数，Embedding 层就可以将这些整数向量化，简单来讲就是生成了每个字的字向量。接下来紧跟着一层是 LSTM，它输出维度也是隐藏层的结点个数。Dropout 层的作用是让一些神经节点随机不工作，来防止过拟合现象。Dense 层是最后的输出，这里 nb_classes 的数目是 4，代表一个字符的 label。模型建立好后开始训练，重复 20 次，训练的结果如下：

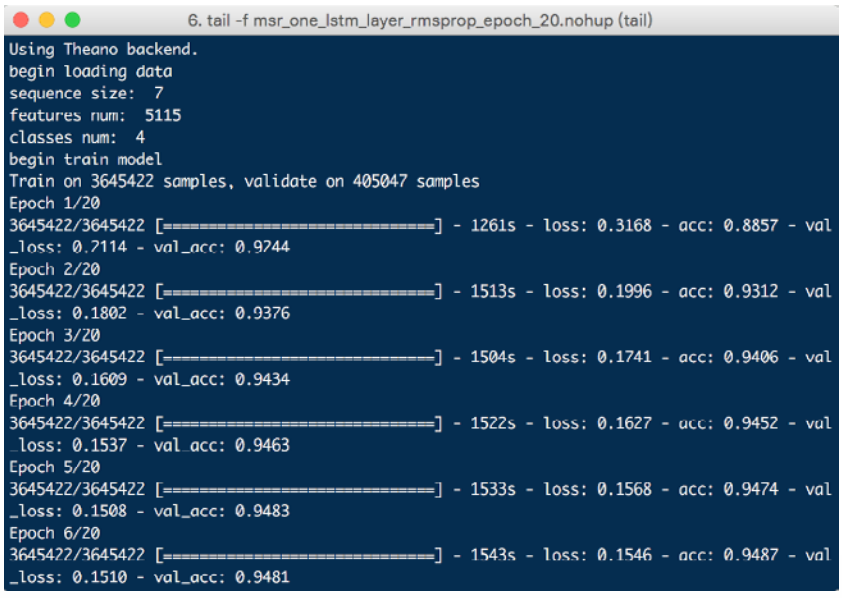


图 4.1 基础模型（1 层 LSTM 优化 RMSprop）训练 20 次

训练好后，我们使用 msr_test 的测试数据进行分词，并将最终的分词结

果使用 icwb2 自带的脚本进行测试，结果如下

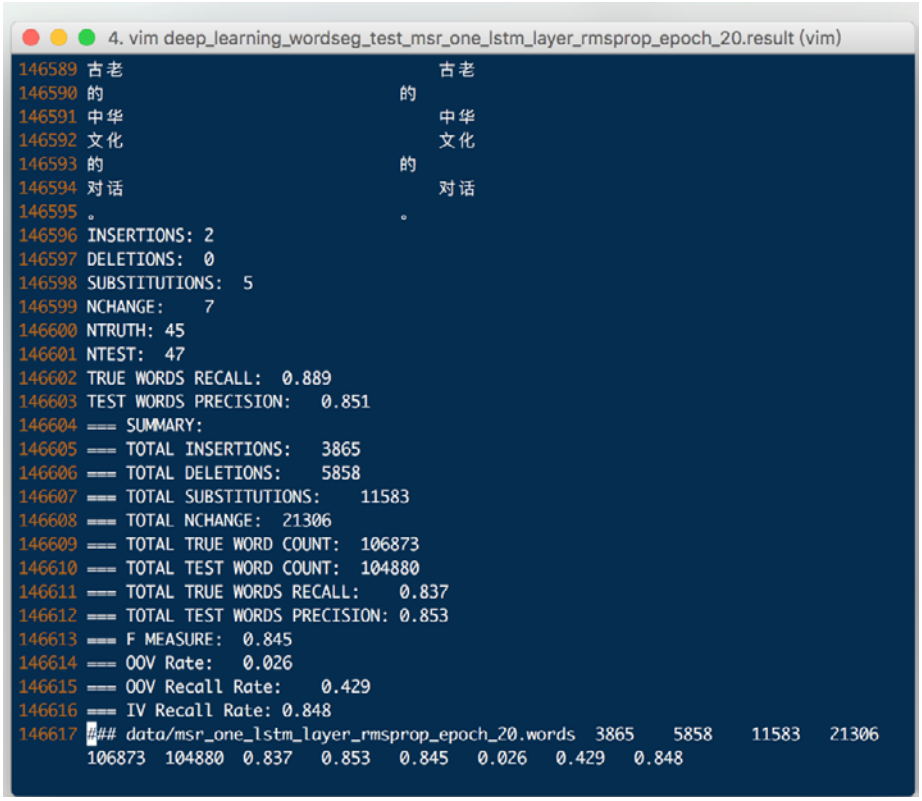


图 4.2 基础模型 F Score： 0.845

可以看到基础模型的 F 值一般，比传统的 CRF 效果差的较多，因此考虑优化模型。

4.2.3 效果改进

1) 模型参数调整

首先想到的是模型参数的调整。Keras 官方文档中提到，RMSprop 优化方法在 RNN 网络中通常是一个好的选择，但是在尝试了其他的优化器后，比如 Adam，发现可以取得更好的效果。

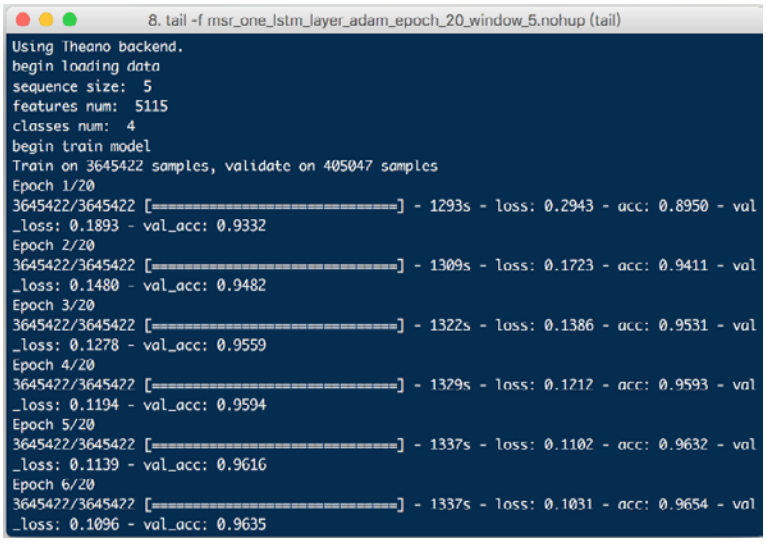


图 4.3 1 层 LSTM 优化器 Adam 训练 20 次

可以看到，Adam 在训练过程中的精度就已经高于 RMSprop，使用 icwb2 的测试结果为：

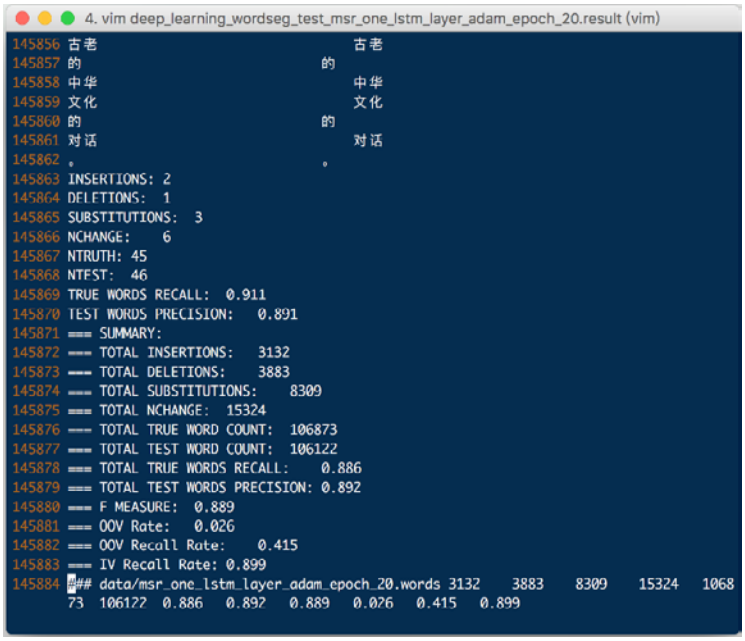


图 4.4 修改优化器 Adam 后的模型 F Score：0.889

2) 模型结构改变

现在网络结构较简单，只有一层 LSTM，参考文档示例中的模型设计，考虑使用两层的 LSTM 来进行测试，修改后的代码如下：

```
def get_lstm_model(input_dim, input_length, nb_classes,
                   hidden_node=100):
    model = Sequential()
    model.add(Embedding(input_dim, hidden_node,
                        input_length=input_length))
    model.add(LSTM(hidden_node, return_sequences =True))
    model.add(LSTM(hidden_node))
    model.add(Dropout(0.5))
    model.add(Dense(nb_classes))
    model.add(Activation('softmax'))
    model.compile(loss='categorical_crossentropy',
                  optimizer='adam', metrics=["accuracy"])
    return model
```

注意，第一层 LSTM 有个 return_sequences =True 可以将最后一个结果出入到输出序列，保证输出的 tensor 是 3D 的，因为 LSTM 的输入要求是 3D 的 tensor。

两层 LSTM 模型训练过程如下：

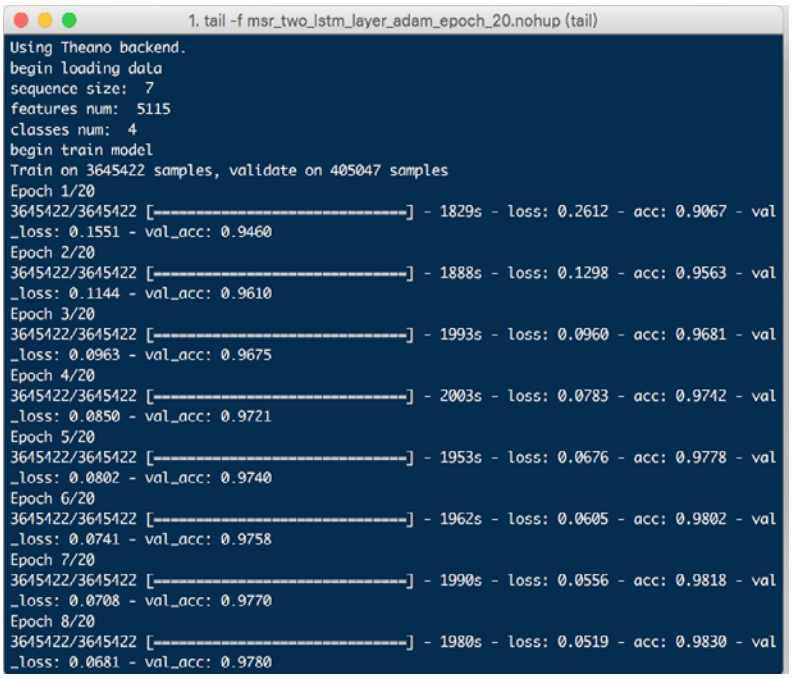


图 4.5 2 层 LSTM 优化器 Adam 训练 20 次的模型

可以看到，两层 LSTM 使得模型更加复杂，训练时常也增加不少。模型训练后，使用 icwb2 的测试结果为：

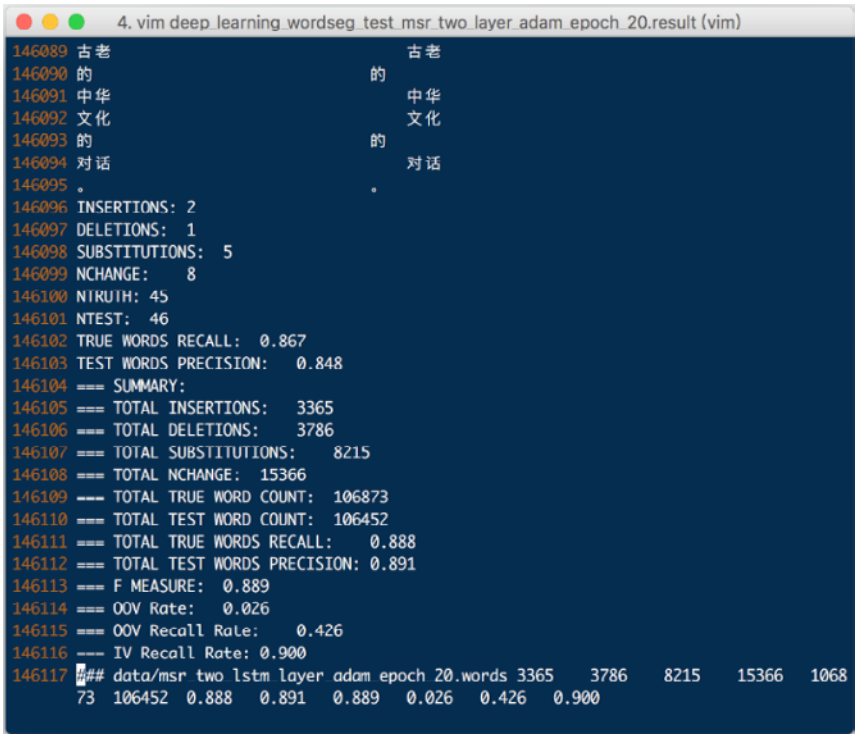


图 4.6 两层 LSTM 的模型 F Score：0.889

可以看到，随着模型的复杂，虽然 F Score 无提升，但是其他的指标有一定的提升。一般来说，神经网络在大量训练数据下也会有更好的效果，后续会继续尝试更大数据集更复杂模型的效果。

5. 总结和展望

深度学习技术给 NLP 技术中的中文分词技术带来了新鲜血液，改变了传统的思路。深度神经网络的优点是可以自动发现特征，大大减少了特征工程的工作量。随着技术的进一步发展，深度学习在 NLP 领域将会发挥更大的作用。达观数据将在已有成熟的 NLP 算法及模型基础上，逐渐融合基于深度神经网络的 NLP 模型，在文本分类、序列标注、情感分析、语义分析等功能上进一步优化提升效果，来更好为客户服务。

【本文作者简介】

高翔，达观数据联合创始人，自然语言处理技术专家，达观数据前端项目组、文本语义理解组负责人，上海交通大学通信专业硕士，曾代表达观数据参加 2016 青年互联网创业大赛并赢得全国总冠军荣誉、第五届中国创新创业大赛优秀企业奖、中国电子 i+ 创新创效创意大赛总决赛一等奖。曾就职于腾讯文学，盛大文学，盛大创新院，负责文本阅读类产品、搜索引擎、文本挖掘及大数据调度系统的开发工作。在自然语言处理和机器学习等技术方向有着丰富的经验。

扫描二维码申请试用
达观数据 NLP 产品

基于半监督学习技术的达观数据文本过滤系统

张健　达观数据架构师

社交、直播、论坛、电商等各类平台每天都会产生海量 UGC(User Generated Content)，其中不可避免地混杂有大量垃圾文本。这些内容不但严重影响用户体验，而且还可能引发违规的运营风险。面对这些迫切需要，达观数据提供了垃圾信息过滤的服务，精准定位并剔除不良信息。

通常垃圾信息过滤的问题可以看作分类问题，即判断一个评论是属于正常评论这个分类，还是属于垃圾信息这个分类。文本分类的研究已经经历了很长时间的发展，传统的垃圾信息过滤方法一般是监督的，但是为了确保分类器有良好的泛化能力，这些方法的使用都必须以存在大量标注语料作为前提条件。而在垃圾信息过滤的场景下，标注工作是一件极为困难的事情，达观的审核系统在开发阶段初期就面临标注样本不足的挑战。一方面，用户活跃的平台每天都能产生大量新的评论，而且垃圾信息所占的比重会很高，标注成本非常高；另一方面，垃圾信息发布用户会想方设法把自己“隐藏”在其他正常评论中，只凭语义信息可能难以确定是否为垃圾信息。

为了克服标注样本不足的难题，垃圾信息过滤可以引入半监督学习方法来增强信息处理的能力。半监督学习方法的优势是能够在只有少量标注数据的条件下，综合利用已标注数据和未标注数据的信息，达到较好的过滤效果。达观的文本挖掘系统在多个模块里面都使用到了半监督学习的方法，主要方式是通过外部知识来对训练样本进行语义扩展，然后结合数量较多的未标注样本选取预测置信度高的子集作为新样本加入训练集进行模型训练。

下面我介绍一下最近阅读过的采用半监督学习来进行垃圾信息过滤的两篇论文：NetSpam 和 SPEAGLE。

1. NetSpam: a Network-based Spam Detection Framework for Reviews in Online Social Media

论文链接：http://ieeexplore.ieee.org/document/7865975/

NetSpam 论文的基本思想是使用了异构信息网络方法来对用户评论进行建

模，比较新颖的是在文本分类过程中，利用到了异构网络中不同的边类型的信息来提升分类效果。另外，使用无监督的方法能够在没有标注样本的情况下，根据评论数据的统计信息，获得各种特征对应的重要性。

1.1 特征类型

这篇文章提到了问题解决使用到的几方面特征，主要是从基于用户 / 评论和行为 / 语言特征两个维度去刻画，具体特征示例参考表 1.1。

	基于用户	基于评论
行为特征	**突发性**：垃圾信息发布用户会经常在很短的时间内发布大量的垃圾评论，一方面由于这样更容易影响阅读者，另外一方面是由于他们只是些临时用户。特征量化表示为： $x_{BST}(i)=\begin{cases}0 & (L_i-F_i)\notin(0,\tau)\\ 1-\frac{L_i-F_i}{\tau} & (L_i-F_i)\in(0,\tau)\end{cases}$ 其中 Li-Fi 表示最近和最早的评论的时间间隔。 负面比率：垃圾信息发布用户为了打击商业对手，会发布负面诽谤言论或者对一些产品打分极低。	早期时帧：为了将垃圾信息保留在评论顶部位置，它的发布需越快越好。特征量化表示 $x_{ETF}(i)=\begin{cases}0 & (T_i-F_i)\notin(0,\delta)\\ 1-\frac{T_i-F_i}{\delta} & (T_i-F_i)\in(0,\delta)\end{cases}$ 其中 Ti － Fi 表示特定评论的最近和最早发布的时间间隔。 带阈值评分偏差率：为了促进商品推广，垃圾信息发布者通常会打很高的评分，一般和该产品平均评分有较大的偏差，特征量化表示为： $x_{DEV}(i)=\begin{cases}0 & otherwise\\ 1-\frac{r_{ij}-avg_{e\in E_{*j}}r(e)}{4} & >\beta_1\end{cases}$ 其中 rij 表示用户对商品的评分，β 1 表示阈值。
语言特征	内容相似度：垃圾发布用户不会花精力去原创内容，只会经常在相同的评论模板上去修改，所以会有大量的相似评论。	第一人称数目、感叹号比例等：研究发现垃圾信息包含一些语言特性，譬如说第二人称使用频率远高第一人称，以及为了吸引眼球符号使用不同正常习惯。

表 1.1 NetSpam 特征类型

1.2 异构信息网络

首先展示异构信息网络（Heterogeneous Information Network，HIN）定义，信息网络可以用一个有向图 G = (V, E) 来表示，其中 V 代表节点，E 代表边。并且用映射函数 φ:V → A 来表示每一个节点 v ∈ V 属于节点类型集合 A: φ(v) ∈ A，用映射函数 ψ:E → R. 来表示每条边 e ∈ E 属于边的类型集合 R: ϕ (e) ∈ R。而在异构信息网络里面，节点和边有多重类型，每个节点或者每个边

都有固定的类型，一个异构信息网络如图 1.1。

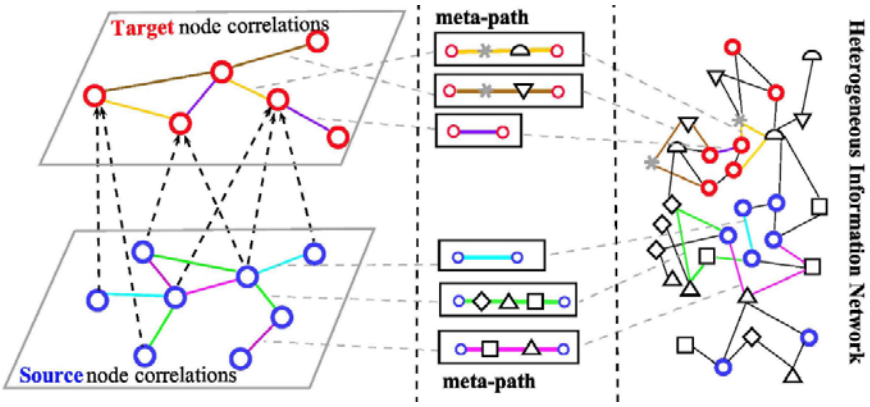


图 1.1 异构信息网络

元路径 P 是定义在网络模式 TG = (A, R) 上的，如$A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$表示了从 A_1 到 A_{l1} 的复杂的关系，$R = R_1 \circ R_2 \circ \cdots \circ R_l$。元路径 P 的长度即为关系 R 的个数。不同元路径代表着不同的物理意义，元路径所所蕴含的丰富的语义特征，是 HIN 的一大非常重要的特征。

异构信息网络上分类问题的形式化定义是：对于网络 G = (V, E)，V’ 是 V 中需要进行分类的目标节点子集。目标节点的分类信息包含 $C_1 \ldots C_k$。C 在该场景里，我们在 V’ 中已经有部分已标注的节点，分类任务的目标就是预测 V’ 中所有未标注的节点。

1.3 算法模型

首先计算先验概率，设定评论 u 的初始概率记作为 y_u。模型方案包括了半监督学习和无监督学习两种。在半监督学习方案中，初始概率 y_u 定义为：

$$y_u = \begin{cases} 1, & \text{标记为垃圾信息} \\ 0, & \text{其他} \end{cases}$$

在无监督学习方案中，初始概率 y_u 定义为：

$$y_u = \left(\frac{1}{L}\right)\sum_{l=1}^{L} f(x_{lu})$$

其中$f(x_{lu})$表示评论 u 在特征 l 上的概率，L 是所使用特征的总数。

异构网络网络架构的元素基于多个特征生成：负面比率（NR），平均内容

相似度（ACS），第一人称数目（1PP）和早期时帧（ETF）等，见图 1.2。

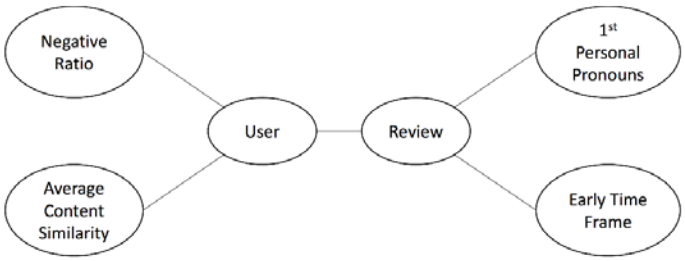


图 1.2 网络架构

基于该网络架构形成的评论到评论的元路径可表示为表格 1.2。

标记	类型	元路径	关联语义信息
R-DEV-R	RB	评论-带阈值偏差率-评论	具有同等偏差率的评论
R-U-NR-U-R	UB	评论-用户-负面比率-用户-评论	被具有相似的负面比率的不同用户发表的评论
R-ETF-R	RB	评论-早期时帧-评论	同一时间间隔发布的评论
R-U-BST-U-R	UB	评论-用户-突发性-用户-评论	同样突发性的用户发布的评论
R-RES-R	RL	评论-感叹号比例-评论	具有相同感叹号比例的评论
R-PP1-R	RL	评论-第一人称数目-评论	具有相同第一人称数目的评论
R-U-ACS-U-R	UL	评论-用户-平均内容相似度-用户-评论	具有相同平均内容相似度的不同用户发布的评论
R-U-MCS-U-R	UL	评论-用户-最大内容相似度-用户-评论	具有相同最大内容相似度的不同用户发布的评论

表格 1.2 元路径列表

给定了元路径的设定后，论文扩展了异构网络的定义，在元路径上具有相同的值的两个评论是相互连通的。给定评论 u，u 在元路径 pl 上的值的计算方式为：

$$\bar{m}_u^{p_l} = \frac{\lfloor s \times f(x_{lu}) \rfloor}{s}$$

其中 s 表示指定的元路径对垃圾评论相关的确定性的级别。如果对于两个评论 u 和 v，如果满足

$$m_u^{p_l} = m_v^{p_l}$$

那么在评论网络中就把这两个评论连通起来。

1.4 分类过程

NetSpam 的分类过程包括两个步骤：计算每个特征的影响权重；计算每条评论的最终概率并且进行标记垃圾 / 非垃圾信息。（达观数据 张健）NetSpam认为节点的分类是基于评论网络中该节点与其他节点的关系完成的，关联的两个节点会有较高的概率带有同样的标签。在此过程中，元路径的权重会帮助我们去理解评论网络中各种影响因子的重要性，合理设计权重的计算方式对分类效果有直接影响。该论文提出元路径权重的计算方式为：

$$W_{p_i} = \frac{\sum_{r=1}^{n}\sum_{s=1}^{n} mp_{r,s}^{p_i} \times y_r \times y_s}{\sum_{r=1}^{n}\sum_{s=1}^{n} mp_{r,s}^{p_i}}$$

标记过程就比较简单，假设 $Pr_{u,v}$ 是和垃圾评论 v 有连通关系的未标注评论 u 可能是垃圾评论的概率，它的计算方式为：

$$Pr_{u,v} = 1 - \Pi_{i=1}^{L} 1 - mp_{u,v}^{p_i} \times W_{p_i}$$

而评论 u 最终为垃圾评论的概率 Pr_u 计算方式为：

$$Pr_u = avg(Pr_{u,1}, Pr_{u,2}, ..., Pr_{u,n}) \quad 。$$

图 1.3 展示了对构建的异构信息网络进行分类处理的计算过程：

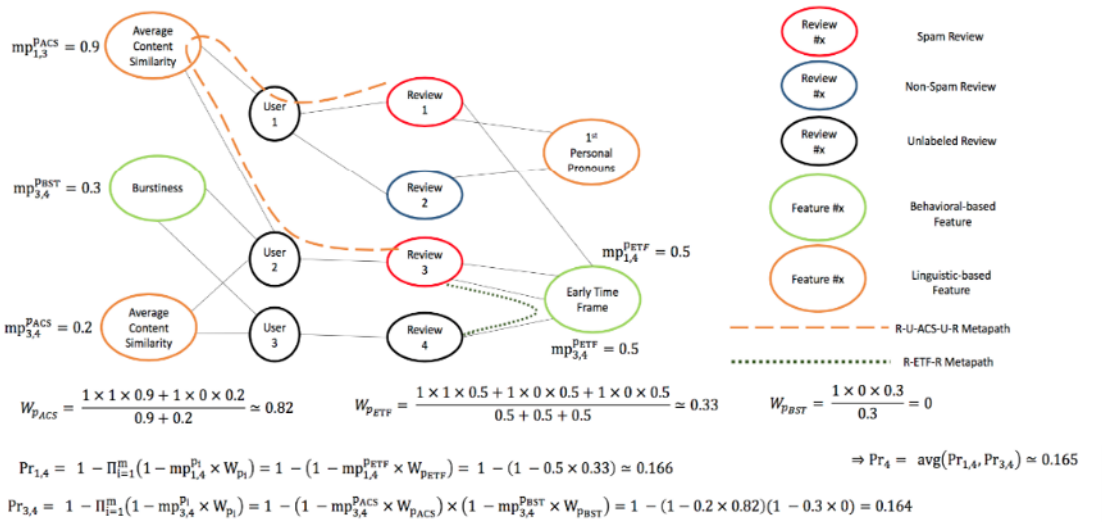


Fig. 2: An example of a review network and different steps of proposed framework.

图 1.3 NetSpam 分类处理流程

1.4 小结

基于异构信息网络对用户评论进行建模，从全局上充分地收集了评论和用户，评论和评论，用户和用户之间的多元关系信息，在行为和语言特征两个维度上进行评估，能在不依赖于专家知识的基础上，自动地学习到用户和评论的分类属性，具有较强的鲁棒性。

2. Collective Opinion Spam Detection: Bridging Review Networks and Metadata

论文链接：http://dl.acm.org/citation.cfm?id=2783370

SpEagle 论文认为垃圾信息过滤需要充分用到包括文本、时间戳和评分在内的元数据和评论网络，并且需要将这它们融合到一个体系内。如图 4 所示，SpEagle 利用了元数据、评论网络以及评论标签的信息，完成了识别出垃圾内容发布者、虚假评论和虚假内容目标商品三者的任务，分类过程是通过评论 – 产品和评论 – 用户的关系构建马尔科夫随机场模型实现。

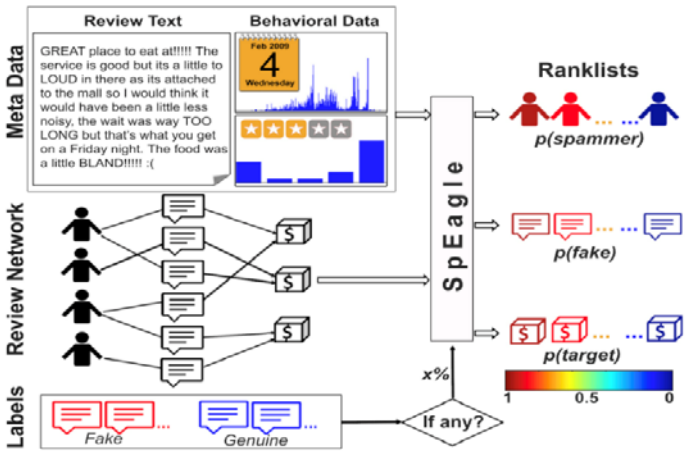


图 2.1 SPEAGLE 系统框架

2.1 特征类型

SPEAGLE 用到的特征和 NetSpam 论文相似，如表格 3 和表格 4（其中第三列的 H/L 表示和垃圾内容的关联度是高 / 低）：

用户&商品特征			
行为特征	MNR	H	一天内发布的最大评论数
	PR	H	正面评价的比例（4-5 星）
	NR	H	负面评价的比例（1-2 星）
	avgRD	H	平均偏差率
	WRD	H	带权重的偏差率
	BST	H	突发性
	ERD	L	用户评价分数的分布的熵
	ETG	L	评论间隔分布的熵
语言特征	RL	L	平均评论字数长度
	ACS	H	平均内容相似度
	MCS	H	最大内容相似度

表 2.1 SPEAGLE 用户 & 商品特征类型

评论特征			
行为特征	Rank	L	在所有商品评论中的次序
	RD	H	评分偏差率的绝对值
	EXT	H	评分极端性：评分值仅在{4，5}，$x_{EXT}=1$；否则 $x_{EXT}=0$。
	DEV	H	带阈值的评分偏差率：
	ETF	H	早期时帧
	ISR	H	是否是单例。$x_{ISR}=1$ 仅当这是用户唯一发布的评论，否则为 0。
语言特征	PCW	H	字母全大写单词所占比例
	PC	H	大写字母所占比例
	L	L	评论字数长度
	PP1	L	第一人称的比例
	RES	H	感叹号比例
	SW	H	主观单词比例（见 sentiWordNet）
	OW	L	客观单词比例（见 sentiWordNet）
	F	H	评论的频率
	DL_u	L	基于单个词的描述长度
	DL_b	L	基于二元词组的描述长度

表 2.2 SPEAGLE 评论特征

2.2 模型定义

利用评论 – 产品和评论 – 用户的关系，构建出了一个二分图 G=(V,E)，节点集合 $V = U \cup P \cup R$，包括了评论、用户和产品三种类型的节点。在该二分图上进行分类的目的是对每一个节点都分配一个标签，评论类型节点标签的值域 L_R={ 真实，虚假 }，用户类型节点标签的值域 L_U={ 正常用户，垃圾内容发布用户 }，产品类型节点标签的值域 L_P={ 垃圾内容目标产品，非目标产品 }。该分类问题可以形式化转化为成对马尔科夫随机场模型（MRF）。MRF 模型

包含了一个无向图，无向图的每个节点都和一个随机变量 Yi 关联，作为它的状态（状态数目有限）。而在成对 MRF 中，一个节点的标签可看做只依赖它的邻居和与图中其他所有节点独立。

标签的联合概率可以写作：$P(\mathbf{y}) = \frac{1}{Z} \prod_{Y_i \in V} \phi_i(y_i) \prod_{(Y_i, Y_j, s) \in E^{\pm}} \psi_{ij}^{s}(y_i, y_j)$

其中 y 表示对所有节点的一种标签标注方法，y_i 是节点分配的标签，Z 是一个标准化常量。独立因子$\phi: \mathcal{L} \to \mathbb{R}^+$是节点的势函数，表示每个节点的初始分类概率，$\phi: \mathcal{L}_U \times \mathcal{L}_P \to \mathbb{R}^+$是边的势函数，表示带有标签 yi 的节点通过边 s 连通到带有标签 yj 的节点的概率。

SPEAGLE 算法面向两种类型的边的通用势函数ψ^t为：

	User ($\psi^{t='write'}$)		($\psi^{t='belong'}$) Product	
Review	*benign*	*spammer*	*non-target*	*target*
genuine	1	0	$1-\epsilon$	ϵ
fake	0	1	ϵ	$1-\epsilon$

从上可看出，我们假定了垃圾内容发布用户的评论都是虚假的，正常用户发布的评论都是真实的；不过虚假的评论也有可能是一个非目标产品，真实评论也可能和虚假评论在目标产品中共存。

节点的先验势函数通过 SPEAGLE 的特征类型进行响应的抽取计算来获取：$\phi_i \leftarrow \{1 - S_i, S_i\}$

2.3 算法过程

SPEAGLE 的算法过程将用户 - 评论 - 产品的图 G，通用势函数ψ^t，元数据中抽取出的特征以及已标注节点的标签（包括评论、产品和用户的标签）作为输入，输出为所有未标注节点对应分类的概率。算法执行过程中，先对所有节点进行分类概率的初始化，如果是已标注节点，则根据ψ^t进行设定；如果是未标注节点，则需要抽取这些节点对应的特征，然后计算出特征的垃圾内容权重 S_i，然后设置势函数为$\phi_i \leftarrow \{1 - S_i, S_i\}$。

然后主要步骤是通过 Loopy Belief Propagation（LBP）算法来计算条件边缘概率。LBP 算法是基于网络中的每个节点通过和邻近节点交换信息对自身的概率状况进行评估，而且这个过程是迭代进行，对于每一次迭代，消息 m_{i-j} 从节点传递到节点 j，其中$T_i, T_j \in \{U, R, P\}$，表示了节点 i 和节点 j 的类型。消息 m_{i-j} 表示节点对节点 j 的置信度，即 i 认为的 j 的分类分布。对于分类分布的，是基于连通节点 i 和节点 j 的边的权重，以及节点 i 的领域中不包含去 j 的其他节点中接收到的消息来进行。消息传递迭代进行，直到小于阈值到达稳定状态。

当消息稳定之后，计算出边缘概率 bi(yi) 见图。对于分类问题，节点可以根据$max_{y_i} b_i(y_i)$来进行标记；对于排序问题，则可以按照$_{y_i} b_i(y_i)$来进行排序。

Algorithm 1: SpEagle

Input: User–Review–Product graph $G = (V, E)$, compatibility potentials ψ^t (Table 1), review metada (ratings, timestamps, text), labeled node set L
Output: Class probabilities for each node $i \in V \backslash L$
foreach $i \in V$ **do** // *compute/initialize priors*
  if $i \in L$ **then**
    if *i is positive (spam) class* **then** $\phi_i \leftarrow \{\epsilon, 1-\epsilon\}$
    else $\phi_i \leftarrow \{1-\epsilon, \epsilon\}$
  else
    Extract corresponding features in Table 2
    Compute spam score S_i using Eqn. (2)
    $\phi_i \leftarrow \{1-S_i, S_i\}$
foreach $(Y_i^{T_i}, Y_j^{T_j}, t) \in E$ **do** // *initialize all msg.s*
  foreach $y_j \in \mathcal{L}_{T_j}$ **do**
    $m_{i\to j}(y_j) \leftarrow 1$
repeat // *iterative message passing*
  foreach $(Y_i^{T_i}, Y_j^{T_j}, t) \in E$ **do**
    foreach $y_j \in \mathcal{L}_{T_j}$ **do**

$$m_{i\to j}(y_j) = \alpha \sum_{y_i \in \mathcal{L}_{T_i}} \left(\phi_i(y_i)\, \psi_{ij}^t(y_i, y_j) \prod_{Y_k \in \mathcal{Y}_{\mathcal{N}_i} \backslash Y_j} m_{k\to i}(y_i) \right)$$

until messages stop changing within a δ threshold
foreach $Y_i^{T_i} \in \mathcal{Y}_{V\backslash L}$ **do** // *compute final beliefs*
  foreach $y_i \in \mathcal{L}_{T_i}$ **do**

$$b_i(y_i) = \beta\, \phi_i(y_i) \prod_{Y_j \in \mathcal{Y}_{\mathcal{N}_i}} m_{j\to i}(y_i)$$

图 2.2 SPEAGLE 算法伪代码

2.4 小结

SPEAGLE 采用了基于评论网络来完成分类任务，将评论、用户和产品三者置于统一一致的框架内，而且对于不同的对象类型都使用了统一化的分类方法，这个点比较新鲜。

3. 达观数据垃圾信息过滤工程实践

达观的文本挖掘系统在多个模块里面都使用到了半监督学习的方法，主要方式是通过外部知识来对训练样本进行语义扩展，然后结合数量较多的未标注样本选取预测置信度高的子集作为新样本加入训练集进行模型训练。

从上面两篇论文中的特征类型选择中可以看到，里面的语义特征抽取过程是在英文文本上进行的。到了中文环境下，语义特征抽取的过程会变得复杂很多，主要是由汉语的语言特性造成。具体到垃圾信息过滤这个场景中，变形识别问题是有效进行语义特征抽取亟需解决的重要问题。

3.1 变形识别问题

我们在浏览像贴吧、论坛、新闻媒体等各种平台中，会时常看到变形的敏感词。人脑的思维方式让我们能够非常自然地发现这些变形词，因为这些变形词在句子中是“异常”的部分，这种“异常”的感觉会将我们的注意力聚集到这一区域，进而逐渐发现完整的变形词。而机器在直接面对这些变形词（包括间杂特殊符号，同音变换，形近变换，简繁转换，偏旁拆分等）时就显得稍微力不从心，变形词识别是解决中文垃圾内容过滤的一个重要问题。

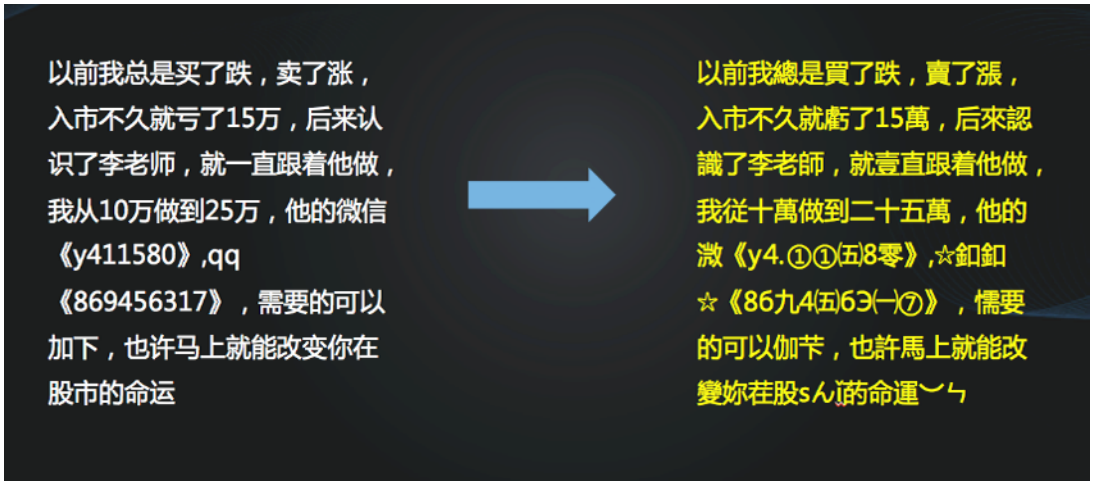


图 3.1 变形识别问题

3.2 变形词自动化生成

如果关键词词库通过人工配置的话，不仅成本大，而且扩展比较困难，面对新类型的垃圾内容出现反应时间也相对较慢。

为了解决变形词识别的问题，达观数据变形词采用了自动化生成的方法，具体步骤包括：

1) 获取关键词词库的字作为种子集合；

2) 构建变形词关联网络（结合拼音相似度、字形相似度、字频、共同出度、共同入度构建关联边的权重）；

3) 生成关键字的相似度大于阈值的变形词；

4) 已针对已有关键词词库构建变形词词库。

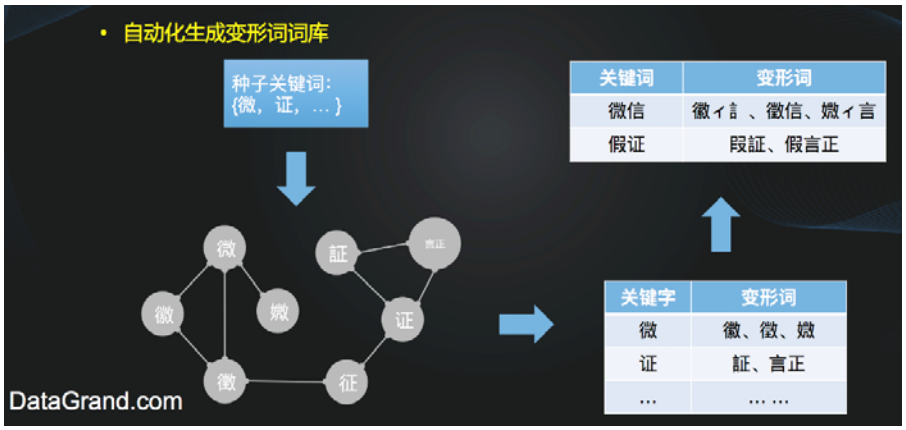


图 3.2 自动化生成变形词词库

3.3 变形词检测

而在正文预测进行变形识别时，如果单纯依靠词库不结合语境的话，很有可能将正常词语错误识别为变形词。譬如根据同音转换的原则进行变形识别时，“Esports 海涛解说视频专题”识别出变形关键词“海淘”，实际上普通读者一眼可以看出来这其实是一段正常文本，“海涛”并非“海淘”的变形词。达观审核系统在解决变形词识别时，使用了下面的方法来进行变形词检测。

3.3.1 贝叶斯分析方法

贝叶斯分析方法统计变形词在正常文本上下文中出现概率，计算当前文本上下文中变形词的后验概率。像“徽亻言”这样的词语，在正常文本中出现的概率几乎为 0，所以可以判别为变形词；而对于出现在“Esports 海涛解说视频专题”的关键词“海涛”，在计算出了当前文本上下文的后验概率之后，可判别为正常词语。

3.3.2 词嵌入方法

词嵌入方法将单词转化为词向量，计算上下文语义重心，计算单词的词向量与上下文文本语义重心向量的相似度。正常文本里面的词语跟上下文文本语义接近，所以对应的词向量在空间上也是比较接近的。通过计算它与上下文语义重心的相似度，可以判别出来该词语是否处于正常语境中，从而识别出来是否是变形词。

【本文作者简介】

张健，达观数据联合创始人，文本挖掘组文本审核总负责人，包括文本审核系统的架构设计、开发和日常维护升级，文本挖掘功能开发。复旦大学计算机软件与理论硕士，曾在盛大创新院负责相关推荐模块，在盛大文学数据中心负责任务调度平台系统和集群维护管理，数据平台维护管理和开发智能审核系统。对大数据技术、机器学习算法有较深入的理解和实践经验。

扫描二维码申请试用
达观数据 NLP 产品

达观文本指纹算法和系统简述

文辉　达观数据科学家

1. 文本指纹介绍

互联网网页存在大量的重复内容网页，无论对于搜索引擎的网页去重和过滤、新闻小说等内容网站的内容反盗版和追踪，还是社交媒体等文本去重和聚类，都需要对网页或者文本进行去重和过滤。最简单的文本相似性计算方法可以利用空间向量模型，计算分词后的文本的特征向量的相似性，这种方法存在效率的严重弊端，无法针对海量的文本进行两两的相似性判断。模仿生物学指纹的特点，对每个文本构造一个指纹，来作为该文本的标识，从形式上来看指纹一般为固定长度较短的字符串，相同指纹的文本可以认为是相同文本。

最简单的指纹构造方式就是计算文本的 md5 或者 sha 哈希值，除非输入相同的文本，否则会发生“雪崩效应”，极小的文本差异通过 md5 或者 sha 计算出来的指纹就会不同（发生冲撞的概率极低），那么对于稍加改动的文本，计算出来的指纹也是不一样。因此，一个好的指纹应该具备如下特点：

1）指纹是确定性的，相同的文本的指纹是相同的；

2）指纹越相似，文本相似性就越高；

3）指纹生成和匹配效率高。

业界关于文本指纹去重的算法众多，如 k-shingle 算法、google 提出的 simhash 算法、Minhash 算法、top k 最长句子签名算法等等，本文接下来将简单介绍各个算法以及达观指纹系统的基本架构和思路。

2. 常用的指纹算法

2.1 k-shingle 算法

shingle 在英文中表示相互覆盖的瓦片。对于一段文本，分词向量为 [w1, w2, w3, w4, … wn]，设 k=3，那么该文本的 shingle 向量表示为 [(w1,w2,w3), (w2,w3,w4), (w3,w4,w5), …… (wn−2,wn−1,wn)]，可以通过计算两个文本的 shingle 向量的相似度（jarccard 系数）来判断文本是否重复。由于 k-shingle 算法的 shingle 向量空间巨大（特别是 k 特别大时），相比 vsm 更加耗费资源，一般业界很少采用这类算法。

2.2 Simhash 算法

Simhash 是 google 用来处理海量文本去重的算法，同时也是一种基于 LSH(locality sensitive hashing) 的算法。简单来说，和 md5 和 sha 哈希算法所不同，局部敏感哈希可以将相似的字符串 hash 得到相似的 hash 值，使得相似项会比不相似项更可能的 hash 到一个桶中，hash 到同一个桶中的文档间成为候选对。这样就可以以接近线性的时间去解决相似性判断和去重问题。

simhash 算法通过计算每个特征（关键词）的哈希值，并最终合并成一个特征值，即指纹。

2.2.1 simhash 算法流程

1）首先基于传统的 IR 方法，将文章转换为一组加权的特征值构成的向量；

2）初始化一个 f 维的向量 V，其中每一个元素初始值为 0；

3）对于文章的特征向量集中的每一个特征。

做如下计算：

a) 利用传统的 hash 算法映射到一个 f-bit（一般设成 32 位或者 64 位）的签名。对于这个 f- bit 的签名，如果签名的第 i 位上为 1，则对向量 V 中第 i 维加上这个特征的权值，否则对向量的第 i 维减去该特征的权值；

b）整个特征向量集合迭代上述运算后，根据 V 中每一维向量的符号来确定生成的 f-bit 指纹的值，如果 V 的第 i 维为正数，则生成 f-bit 指纹的第 i 维为 1，否则为 0。

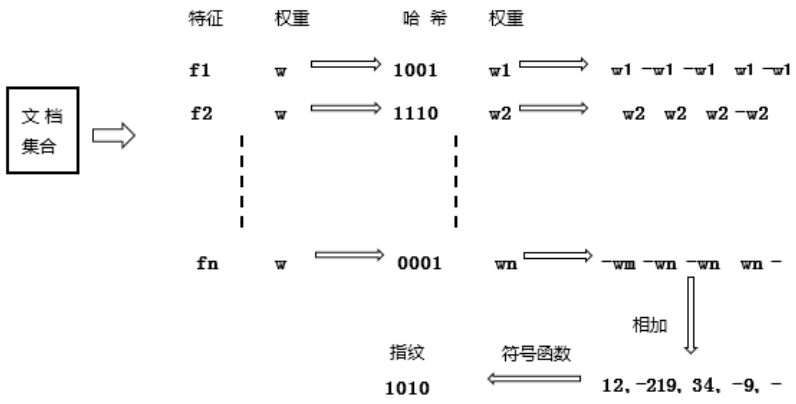


图 2.1 simhash 算法示意图

2.2.2 Simhash 指纹匹配过程

经过 simhash 指纹生成算法生成的指纹是一个 f 位的二进制字符串，如一

个 32 位的指纹，‘10100111110001101010001101101l’。对于两个文本的 f 位 0-1 字符串，simhash 算法采用 hamming distance 来计算两个指纹之间的相似度，但是对于海量文本，如何从千万级别（甚至更多）的指纹集合中，找出最多只有 k 位不同的指纹呢?

一个简单的思想就是以空间换时间，对于一个 32 位的指纹来说，将该指纹划分成 4 段，即 4 个区间，每个区间 8 位，如果两个指纹至多存在 3（设 k=3）位差异，那么至少有一段的 8 位是完全相同的，因此可以考虑利用分段来建立索引，来减少需要匹配的候选指纹数量。

2.2.3 Simhash 指纹匹配算法

1）首先对于指纹集合 Q 构建多个表 T1，T2…Tt，每一个表都是采用对应的置换函数 π(i) 将 32-bit 的 fingerprint 中的某 p(i) 位序列置换换到整个序列的最前面。即每个表存储都是整个 Q 的 fingerprint 的复制置换。

2）对于给定的 F，在每个 Ti 中进行匹配，寻找所有前 pi 位与 F 经过 π(i) 置换后的前 pi 位相同的 fingerprint。

3）对于所有在上一步中匹配到的置换后的 fingerprint，计算其是否与 π(i)(F) 至多有 k-bit 不同。

Simhash 算法比较高效，比较适用于对于长文本。

2.3 Minhash 算法

Minhash 也是一种 LSH 算法，同时也是一种降维的方法。Minhash 算法的基本思想是使用一个随机的 hash 函数 h(x) 对集合 A 和 B 中的每个元素进行 hash，hmin(A)、hmin(B) 分别表示 hash 后集合 A 和集合 B 的最小值，那么 P(hmin(A) == hmin(B)) = Jaccard(A, B)。这是 minhash 算法的核心，其中 hmin(A) 为哈希函数 h(x) 对集合 A 的最小哈希值。

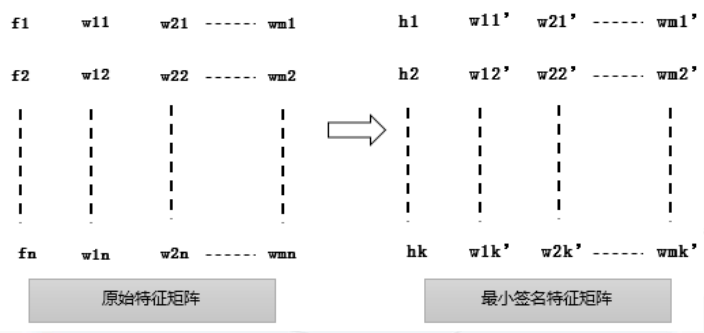


图 2.2 最小签名矩阵生成示意图

Minhash 算法采用最小哈希函数族（一组随机的最小哈希函数）来构建文档的最小哈希签名。文档的最小哈希签名矩阵是对原始特征矩阵降维的结果。应用过程中，可以使用 k 个最小函数分别计算出集合的哈希最小值。设 hi 表示第 i 个最小 hash 函数，最小签名矩阵中列向量为样本 si 的最小签名向量，其中 wij 表示第 j 个最小 hash 函数对样本 i 的最小哈希值。

当 k 小于原始集合的长度（k << n）时，就相当于对数据降维，类比 PCA 等降维方法，minhash 避免了复杂的矩阵运算。由于最小签名矩阵中，样本 i，j 的某一行或某几行的子向量的相似度于样本 i，j 的 jarcarrd 距离相等，因此可以对最小签名矩阵运行行条化策略，经矩阵平均分为 b 个行条，每个行条由 r 条组成，当两个样本在任意一个行条中的向量相等，即是一个相似性候选对，并检查文档是否真正相似或者相等。

关于 minhash 的原理和推导，以及在大量文本及高维特征下如何快速进行最小签名矩阵的构建操作可以参考 https://en.wikipedia.org/wiki/MinHash 及《大数据互联网大规模数据挖掘与分布式处理》，数学的奥妙就在于此。

经过 minhash 降维后的文本向量，从概率上保证了两个向量的相似度和降维前是一样的，结合 LSH 技术构建候选对可以大大减少空间规模，加快查找速度。

3. 内容型网页文本指纹算法

本节将给出我们在对内容型网页（小说、新闻等）去重任务中总结出来的算法和实践经验，特别在当前内容版权日益受到重视和保护的背景下，对于内容版权方来说，如何从网络上发现和追踪侵权和盗版行为日益重要。

从前文可以看出，指纹识别算法是实现指纹识别的关键，它直接决定了识别率的高低，是指纹识别技术的核心。特别是类似新闻类、小说类网页在转载或者盗版过程中，文字的个数、顺序上一般都保持一致，当然不排除个别字错误或者少一个字的情况。

指纹生成的过程主要包括将文本全部转换成拼音、截取每个字拼音的首字母、统计该粒度内字母的频率分布、通过和参考系比较将结果进行归一化、按字母序将数字表征转换成数字。

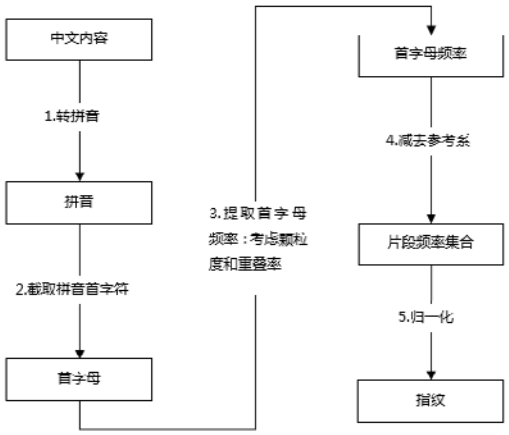


图 3.1 指纹生成算法

算法描述：

1）转拼音：可以解决字符集编码不一致的问题，可以利用成熟的英文指纹算法，减小分布空间，同时可以解决同音字替代问题；

2）截取拼音首字：减小存储长度和分布空间（26 个字母）；

3）提取首字母频率：选择多少字来计算指纹，统计频率分布。需要设置颗粒度的大小（分段大小）以及重叠率。大粒度容错性高，但是匹配率低；小粒度容错性低，但是误报率高且敏感度高。重叠率是设置指纹计算片段移动的窗口大小：

假设拼音内容长为 2n，颗粒长度为 n，重叠率为 50%，则需要计算的指纹片段分别为 [1−n]，[n/2,3*n/2]，[n,2n]；

4）减去参考系：频率减去参考系；

5）归一化：将每个字母的数字特征归一化到一个闭区间内，如 [0,9], 按照字母顺序连接数字特征，变成一个数字，即指纹。

- 若空间为 [0,9], 即一个 20 位的整数，2^64，需要 8 byte
- 若空间为 [0,7], 可用一个 20 位的 8 进制数，8^20, 需要 8 byte
- 若空间为 [0,3], 只需要 4^20, 共 40 bit, 5 byte
- 若空间为 [0,1], 需要 2^20,20 bit,3 byte

归一化过程的算法步骤如下，假设颗粒长度为 m：

```
输入：片段频率集合 S:[s1,s2,s3,…sn]
参数：指纹集合 dnas:[]
计算基数 radix:=pow(2, log(m)/log(2) )
FOR 片段频率 s IN S
    修正频率,每个频率值:=max(频率，基数)
    指纹 dna:=空串
FOR tmp IN s[m-5:m]
        将 tmp 转换成整数，基数为 radix
        将 tmp 转换成字符串，基数为 radix
        dna:=dna 连接 tmp
    dnas:=dnas 添加 dna
END

输出：指纹集合 dnas
```

4. 达观指纹系统结构

4.1 基本架构

达观指纹追踪系统主要由爬虫系统、指纹生成系统、指纹存储、指纹查询和比对、数据分析、后台管理系统等几个主要模块构成，如图 4 所示。其中存储层包括匹配结果信息库、网页库以及指纹库。

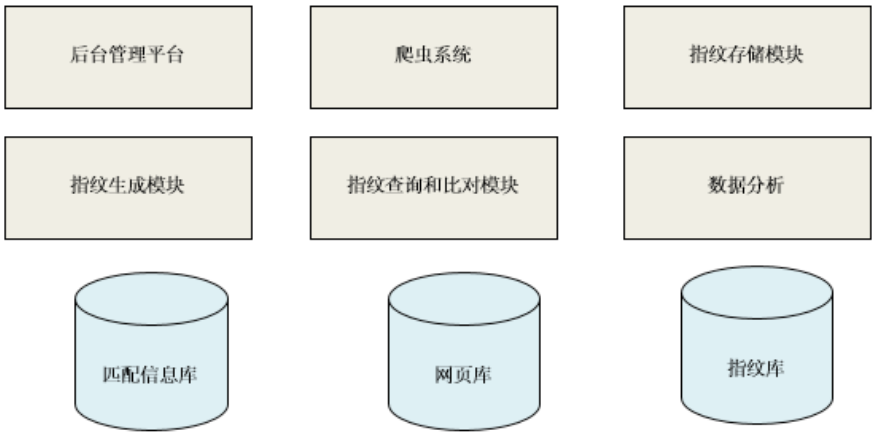


图 4.1 指纹追踪系统模块图

1) 爬虫系统

爬虫系统从目的上看主要在于抓取互联网上的特定领域的网页（如新闻类网页），爬虫系统是原始数据的唯一来源，只有通过爬虫系统才能从浩瀚的互

联网中抓取相似的网页内容。爬虫系统需要拥有较高的抓取能力和反爬取能力，为整个系统提供大量的待检测页面。

2) 指纹存储模块

指纹存储模块计算母体（海量文本）的指纹，指纹可以理解为一行文本的向量表示，本系统的指纹存储系统采用 mongo DB 进行存储。

3) 指纹生成模块

指纹生成模块的输入是一行文本，其输出为该文本的指纹表示，为了达到较高的对比准确率，一个好的指纹生成系统至关重要。

4) 指纹查询和比对模块

指纹库中存储着大量的母体指纹，对于某一文本，指纹查询和比对模块要快速的判断该文本是否在母体库中存在重复。

5) 数据分析

数据分析系统需要对大量的文本及其对比结果进行统计数据分析。

6) 后台管理平台

提供数据分析的展示，并提供用户使用查询和输出分析报告等。

4.2 数据存储模块

1) 网页库

主要存放爬虫系统抓取的网页信息、站点信息，本系统网页库采用 mongo DB。

2) 指纹库

主要存放母体指纹，本系统采用 mongo DB 存放指纹。为了加快指纹的查询和比对，本系统采用 redis 来对指纹建立索引，加快匹配速度。

3) 匹配信息库

存储指纹匹配结果，包括待匹配的两个指纹、原始网页 id、匹配相似度等。

4.3 系统架构

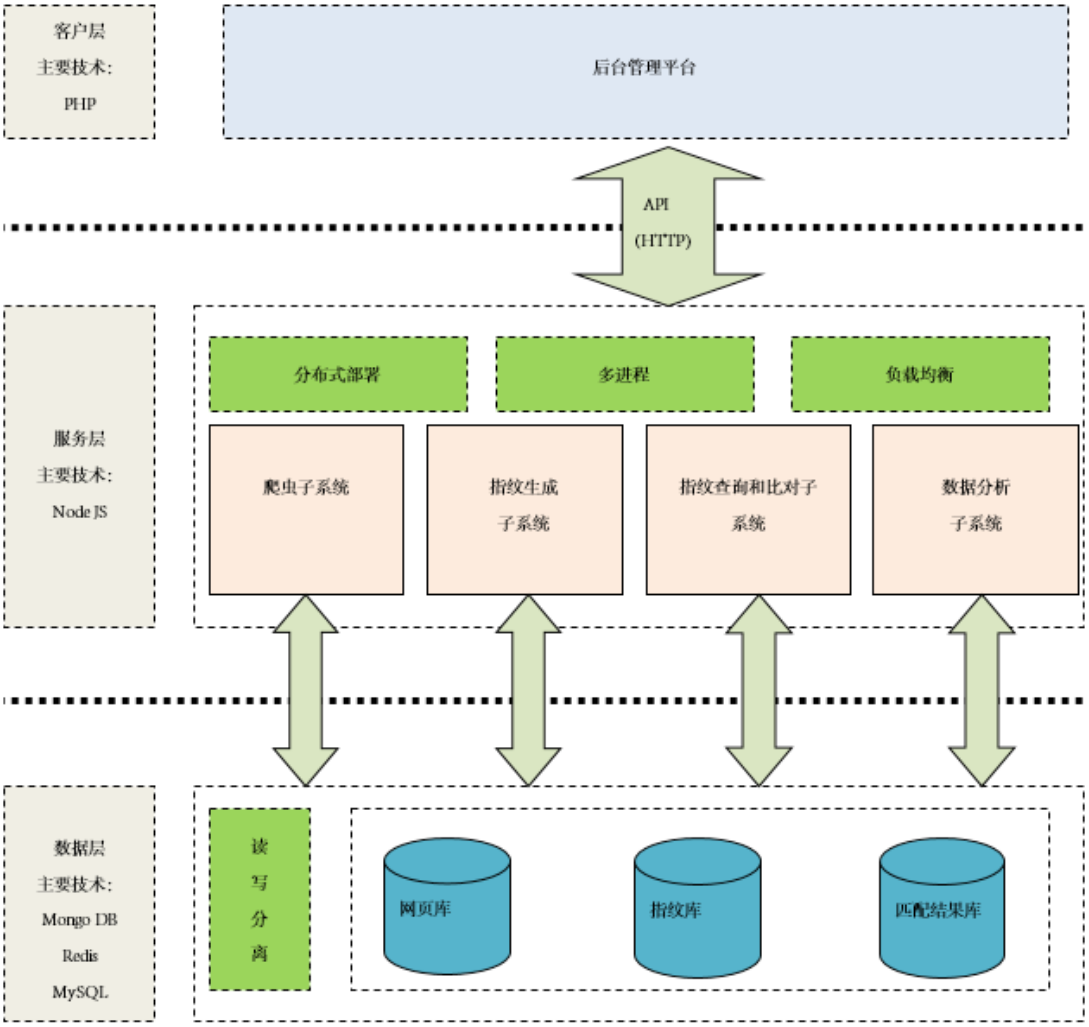


图 4.2 系统架构图

4.4 系统处理流程

本系统的处理流程如图 6 所示，系统支持每天自动化从母体库中调度新的任务进行去重操作。

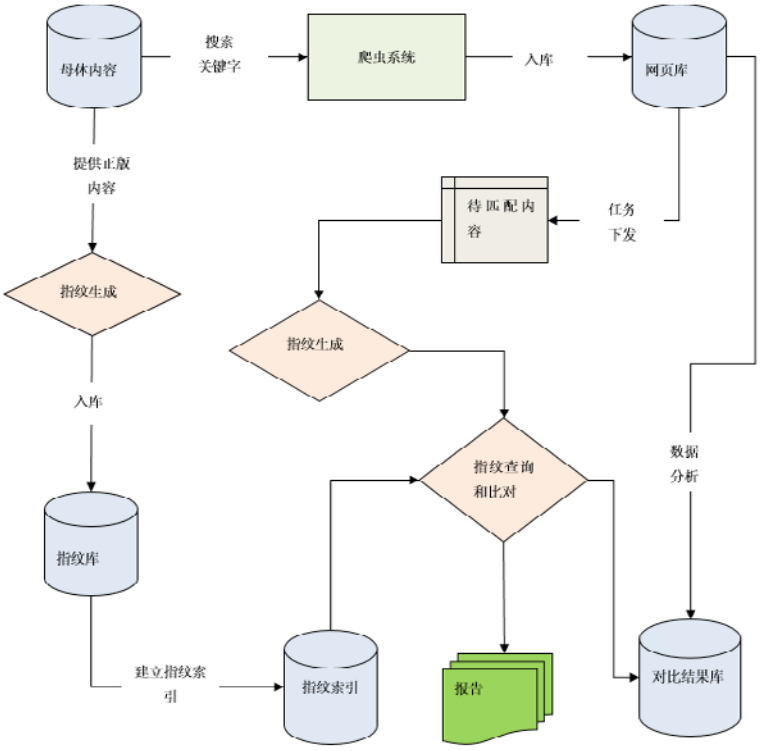


图 4.3 系统流程图

5. 总结

对于网页去重、内容盗版追踪、内容聚类等应用来说，指纹模块都是极其重要的模块。本文介绍了一些比较常用的指纹算法，包括 k-shingle、sim-hash、minhash；同时介绍了达观数据自主开发的指纹追踪系统及其关键算法，达观数据（www.datagrand.com）在指纹系统构建和算法方面积累了丰富的经验，没有最好的算法，只有合适的算法，在实际的使用过程中，需要根据具体业务场景，确定架构和算法。

【本文作者简介】

文辉，达观数据联合创始人，主要负责大观数据爬虫系统、推荐系统等主要系统的研究和开发。同济大学计算机应用技术专业硕士，曾就职于盛大文学数据中心部门，负责爬虫系统、推荐系统、数据挖掘和分析等大数据系统的研发工作，在爬虫系统、Hadoop/Hive、数据挖掘等方面具备充足的研发和实践经验。

扫描二维码申请试用
达观数据 NLP 产品

达观数据如何打造一个中文 NER 系统

高翔　达观数据联合创始人

1. NER 简介

NER（Named Entity Recognition，命名实体识别）又称作专名识别，是自然语言处理中常见的一项任务，使用的范围非常广。命名实体通常指的是文本中具有特别意义或者指代性非常强的实体，通常包括人名、地名、机构名、时间、专有名词等。NER 系统就是从非结构化的文本中抽取出上述实体，并且可以按照业务需求识别出更多类别的实体，比如产品名称、型号、价格等。因此实体这个概念可以很广，只要是业务需要的特殊文本片段都可以称为实体。以下将详细介绍达观数据在文本语义理解过程中是如何构建中文 NER 系统的。

2. NER 问题分解

NER 问题的目标是从文本抽取出特定需求实体的文本片段。针对这个任务，通常使用基于规则的方法和基于模型的方法。

2.1 基于规则的方法

基于规则进行实体抽取是较容易想到的方式。针对有特殊上下文的实体，或实体本身有很多特征的文本，使用规则的方法简单且有效。比如，抽取文本中物品价格，如果文本中所有商品价格都是“数字 + 元”的形式，则可以通过正则表达式”\d*\.?\d+ 元”进行抽取。但是如果待抽取文本中价格的表达方式多种多样，例如“一千八百万”、“伍佰贰拾圆”、“2000 万元”，这个时候就要修改规则来满足所有可能的情况。随着语料数量的增加，面对的情况也越来越复杂，规则之间也可能发生冲突，整个系统也可能变得不可维护。因此基于规则的方式比较适合半结构化或比较规范的文本中的进行抽取任务，结合业务需求能够达到一定的效果。总结一下基于规则的实体抽取方式，优点：简单，快速；缺点：适用性差，维护成本高后期甚至不能维护。

2.2 基于模型的方法

从模型的角度来看，命名实体识别问题实际上是序列标注问题。序列标注问题指的是模型的输入是一个序列，包括文字、时间等，输出也是一个序列。针对输入序列的每一个单元，输出一个特定的标签。以中文分词任务进行举例，

例如输入序列是一串文字：“我是中国人”，输出序列是一串标签：“SSBME”，其中“BMES”组成了一种中文分词的标签体系，B 表示这个字是词的开始 Begin，M 表示词的中间 Middle，E 表示词的结尾 End，S 表示单字成词。因此我们可以根据输出序列“SSBME”进行解码，得到分词结果“我 \ 是 \ 中国人“。序列标注问题涵盖了自然语言处理中的很多任务，包括语音识别、中文分词、机器翻译、命名实体识别等，而常见的序列标注模型包括 HMM，CRF，RNN 等模型。

2.2.1 HMM

HMM（Hidden Markov Model，隐马尔可夫模型）是使用非常广泛经典的一个统计模型，作为一个生成式模型，HMM 用来描述一个含有隐含未知参数的马尔可夫过程。简单来讲，HMM 模型包括两个序列三个矩阵：观察序列、隐藏序列、初始状态概率矩阵、状态转移概率矩阵、发射概率矩阵。通常情况下，我们要根据观察序列和三个矩阵，来得到隐藏序列。

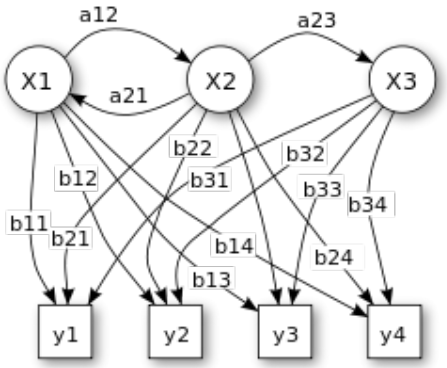


图 2.1 HMM 模型，其中 X 表示隐藏序列，y 表示观察序列，
a 表示状态转移概率，b 表示发射概率

以中文分词任务举例，使用“BMES”标签体系，HMM 模型就是从切分好的语料中统计出初始状态概率矩阵、状态转移概率矩阵、发射概率矩阵这三个矩阵的概率参数。初始状态矩阵指的是序列第一个字符是 BMES 的概率，显然字符是 M 和 E 的概率为 0。状态转移概率矩阵是 BMES 四种状态间转移的概率，显然 B-->S，M-->S，M-->B 等状态的转移概率为 0。发射概率矩阵指的是一个字符是 BMES 四种状态其中一种的概率，比如“中 -->B：0.3“、“中 -->E：0.4“等。可以看到，HMM 模型只需按照模型要求，统计出上述概率矩阵即可，因此 HMM 的优点是模型简单训练快，但因为马尔可夫假设的原因，模型效果相对较差。

2.2.2 CRF

CRF（Conditional random field，条件随机场）是一种判别式模型。条件随机场是给定随机变量 X 的情况下，随机变量 Y 的马尔科夫随机场。马尔科夫随机场是概率无向图模型，满足成对、局部及全局马尔可夫性。对于序列标注问题，一般使用线性链条件随机场。

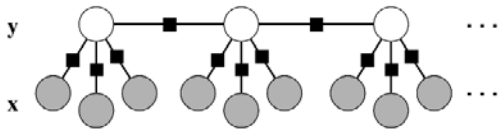


图 2.2 一种线性条件随机场

对于条件随机场的模型训练，通常使用基于 BFGS、SGD 等算法的优化算法，不同软件包的实现上也有所区别。理论上 CRF 算法性能要优于 HMM，因为 CRF 可以使用更多的特征，但同时，特征选择对于模型的性能有一定的影响，除此之外，相对于 HMM，CRF 模型的训练也更加复杂，时间相对较长。

2.2.3 RNN

随着深度学习的兴起，RNN、LSTM、BILSTM 等模型已经被证明在 NLP 任务上有着良好的表现。相比传统模型，RNN 能够考虑长远的上下文信息，并且能够解决 CRF 特征选择的问题，可以将主要的精力花在网络设计和参数调优上，但 RNN 一般需要较大的训练数据，在小规模数据集上，CRF 表现较好。在学术界，目前比较流行的做法是将 BILISTM 和 CRF 进行结合，借鉴两个模型各自的优点，来达到更好的效果。

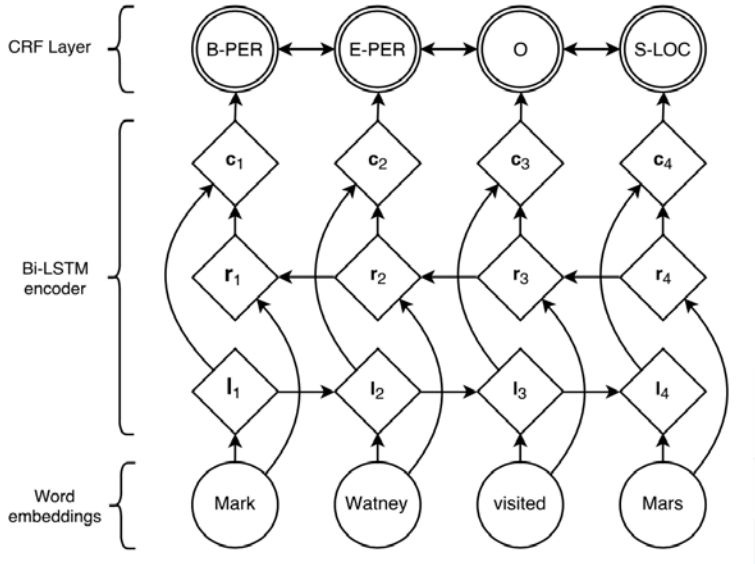


图 2.3 BILSTM+CRF 标注模型

3. 基于 CRF 模型打造中文 NER 系统

上面介绍了用于序列标注不同模型的特点。虽然深度学习有着广阔的前景，并且在机器翻译等任务上表现优异，但对于序列标注任务而言，CRF 模型老而弥坚且比较成熟，在工业界中被广泛使用，因此本章使用 CRF 模型打造一个中文 NER 系统。

3.1 明确标注任务

前文讲过，NER 可以根据业务需求标注各种不同类型的实体，因此首先要明确需要抽取的实体类型。一般通用场景下，最常提取时间、人物、地点及组织机构名，因此本任务提取 TIME、PERSON、LOCATION、ORGANIZATION 四种实体。

3.2 数据及工具准备

明确任务后就需要训练数据和模型工具。对于训练数据，我们使用经典的人民日报 1998 中文标注语料库，其中包括了分词和词性标注结果，下载地址为：http://icl.pku.edu.cn/icl_groups/corpus/dwldform1.asp。 对 于 CRF，有很多开源的工具包可供选择，在此使用 CRF++ 进行训练。CRF++ 官方主页为 https://taku910.github.io/crfpp/，包括下载及使用等说明。

3.3 数据预处理

人民日报 1998 语料库下载完毕后，解压打开“199801.txt”这个文件（注意编码转换成 UTF-8),可以看到内容是由 word/pos 组成，中间以两个空格隔开。我们需要的提取的实体是时间、人名、地名、组织机构名，根据 1998 语料库的词性标记说明，对应的词性依次为 t、nr、ns、nt。通过观察语料库数据，需要注意四点：1，1998 语料库标注人名时，将姓和名分开标注，因此需要合并姓名；2，中括号括起来的几个词表示大粒度分词，表意能力更强，需要将括号内内容合并；3，时间合并，例如将”1997 年 /t 3 月 /t” 合并成”1997 年 3 月 /t”；4，全角字符统一转为半角字符，尤其是数字的表示。

通过脚本将语料库数据进行处理，处理前后的结果如图 3.1 和图 3.2 所示。

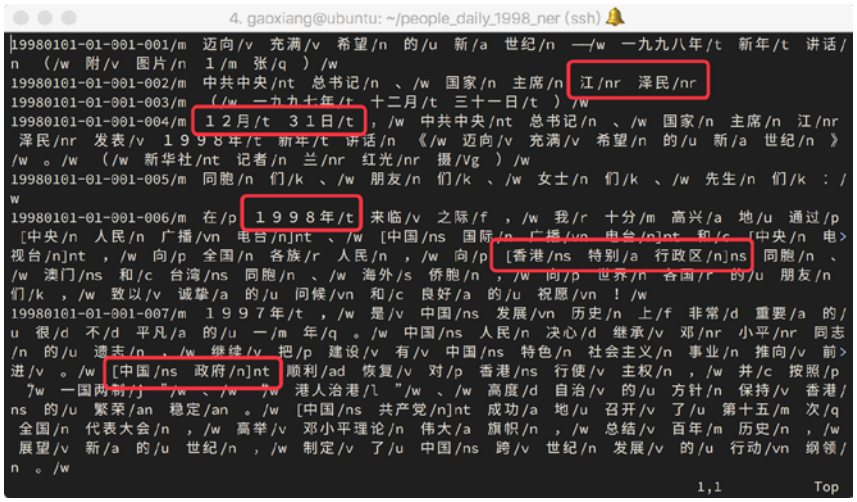


图 3.1 人民日报 1998 标注语料数据处理前

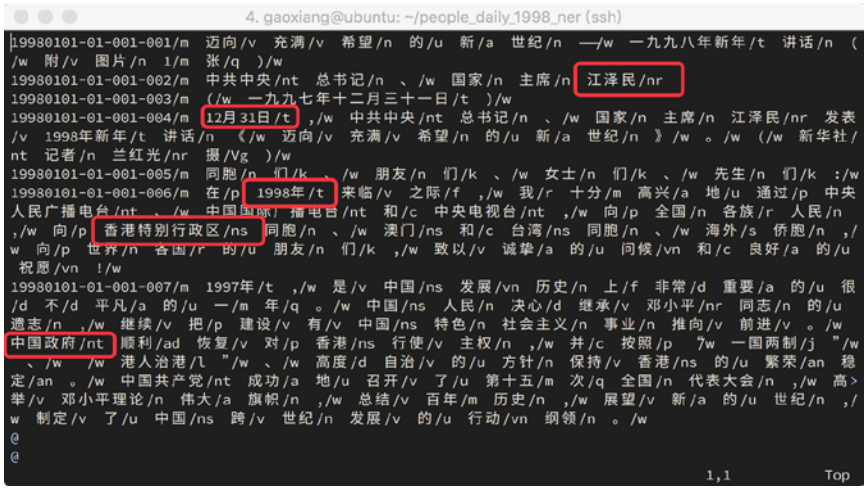


图 3.2 人民日报 1998 标注语料数据处理后

3.4 模型训练

根据我们的 NER 任务需求及 CRF++ 的训练要求，模型训练需要 4 个步骤：1) 确定标签体系；2) 确定特征模板文件；3) 处理训练数据文件；4) 模型训练。

3.4.1 确定标签体系

对于 NER 任务，常见的标签体系包括 IO、BIO、BMEWO、BMEWO+。下面举例说明不同标签体系的区别。

Tokens	IO	BIO	BMEWO	BMEWO+
昨	O	O	O	O
天	O	O	O	O
，	O	O	O	O_PERSON
李	I_PERSON	B_PERSON	B_PERSON	B_PERSON
晓	I_PERSON	I_PERSON	M_PERSON	M_PERSON
明	I_PERSON	I_PERSON	E_PERSON	E_PERSON
前	O	O	O	PERSON_O
往	O	O	O	O_LOCATION
上	I_LOCATION	B_LOCATION	B_LOCATION	B_LOCATION
海	I_LOCATION	I_LOCATION	E_LOCATION	E_LOCATION
。	O	O	O	LOCATION_O

表 3.1 不同标签体系的标注示例

大部分情况下，标签体系越复杂，准确度也越高，但相应的训练时间也会增加。因此需要根据实际情况选择合适的标签体系。本文选择和分词系统类似的 BMEWO 标签体系。

3.4.2 特征模版设计

特征模版是一个文本文件，其内容如图 3.3 所示，其中每行表示一个特征。图 3.3 使用了 unigram 特征，并且仅以字符本身作为特征而不考虑其他特征。除当前字符外，还使用了其前后 3 个字，以及上下文的组合作为特征。CRF++ 会根据特征模版生成相关的特征函数。关于特征模版的详细解释可以查看官网文档，并且对于特征的选择和设计可以灵活配置，图 3.3 仅作为参考。

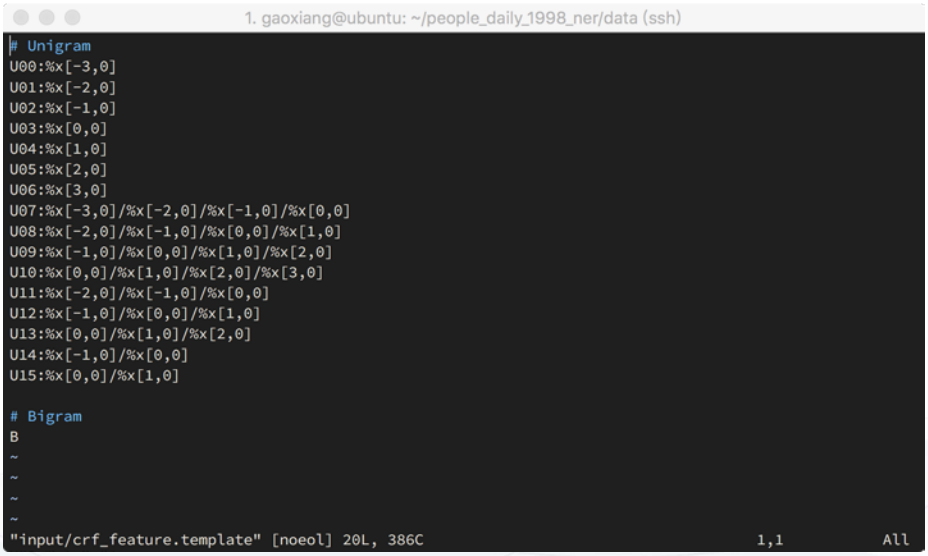


图 3.3 特征模板设计

3.4.3 训练数据生成

CRF 模型的训练数据是一行一个 token，一句话由多行 token 组成。每一行可以分为多列，除最后一列外，其他列表示特征。本文所描述的 NER 系统，单字表示 token，并且仅使用字符这一种特征，因此可以根据语料库中每个字在词中的位置和词性，以及所选的标签系统，生成 CRF++ 的训练数据。生成的训练数据如图 3.4 所示。

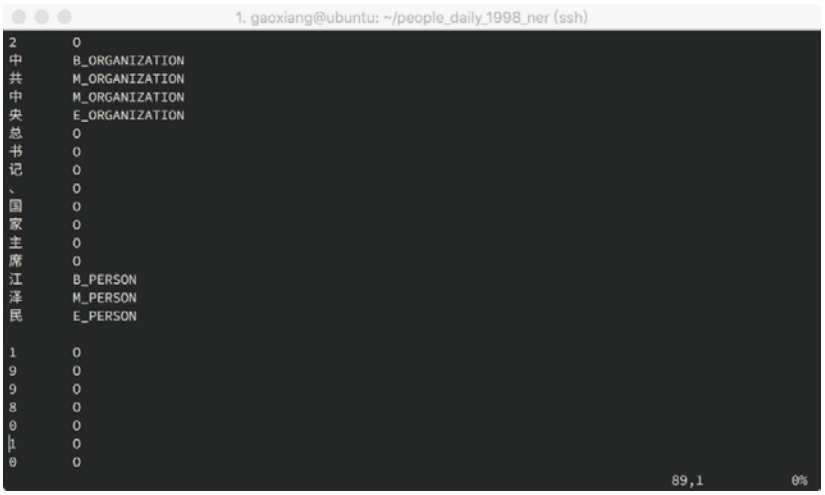


图 3.4 CRF++ 训练数据示例

3.4.4 模型训练

准备好特征模版和训练数据后就可以进行模型训练，如图 3.5 所示。使用 crf_learn 命令，指定模版文件、训练数据文件和输出模型文件就可以进行训练。参数 -f 1 表示过滤频次低于 1 的特征，在这里不进行特征过滤，-c 1.0 用来调节 CRFs 的超参数，c 值越大越容易过拟合。除此之外，还有 -a 等其他参数进行控制调整。图 3.6 展示了训练完毕的相关数据。

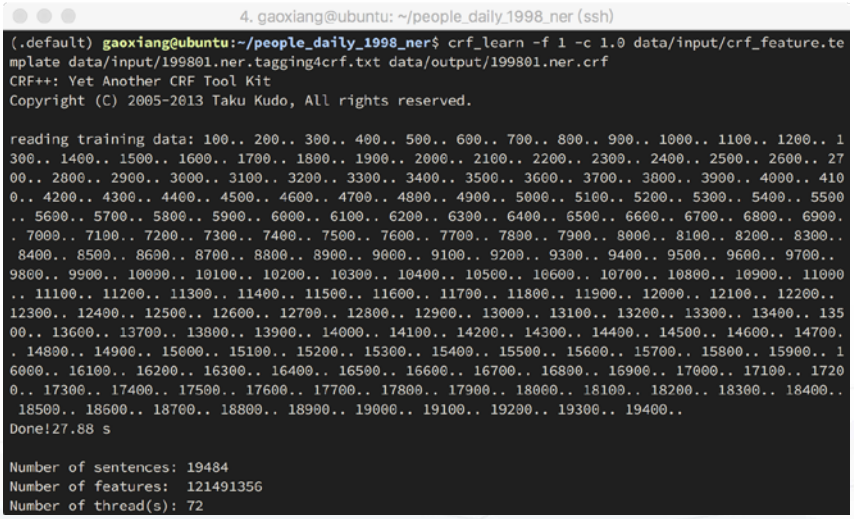


图 3.5 CRF++ 训练过程

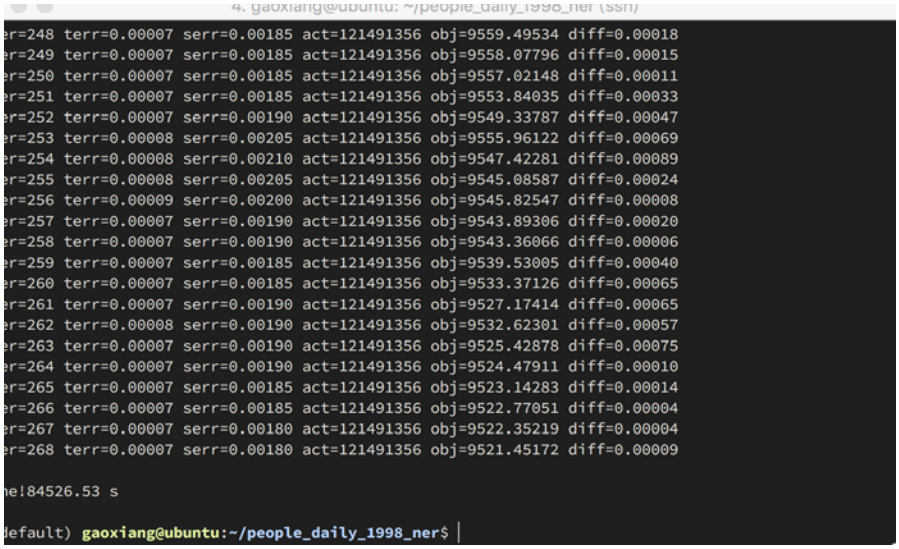


图 3.6 CRF++ 训练结果

3.5 模型预测及使用

模型训练完毕后就可以进行预测。CRF++ 提供 crf_test 命令进行测试，我们使用文本“北京市委组织部长姜志刚调任宁夏副书记“进行测试，测试文件中每字一行，每句话使用空行隔开。测试结果如图 3.7 所示。

图 3.7 CRF++ 测试结果

从图 3.7 的结果我们可以看到，CRF 模型能够对输入文字序列输出相应的标签从而完成 NER 任务。在模型预测时，CRF++ 主要使用了维特比算法进行 nbest 输出。在模型训练时，可以指定 -t 参数输出文本格式的模型，方便

debug 或编写自己的模型加载及解码程序。

对于一个完整的 NER 过程，除了得到序列标签外，还要对标签序列进行解码得到最终的结果。CRF++ 同时提供了 python 接口，可以方便的在 python 程序中进行模型的调用得到标签序列，然后通过标签解码得到最终的结果。图 3.8 展示了一个完整的 NER 预测结果。

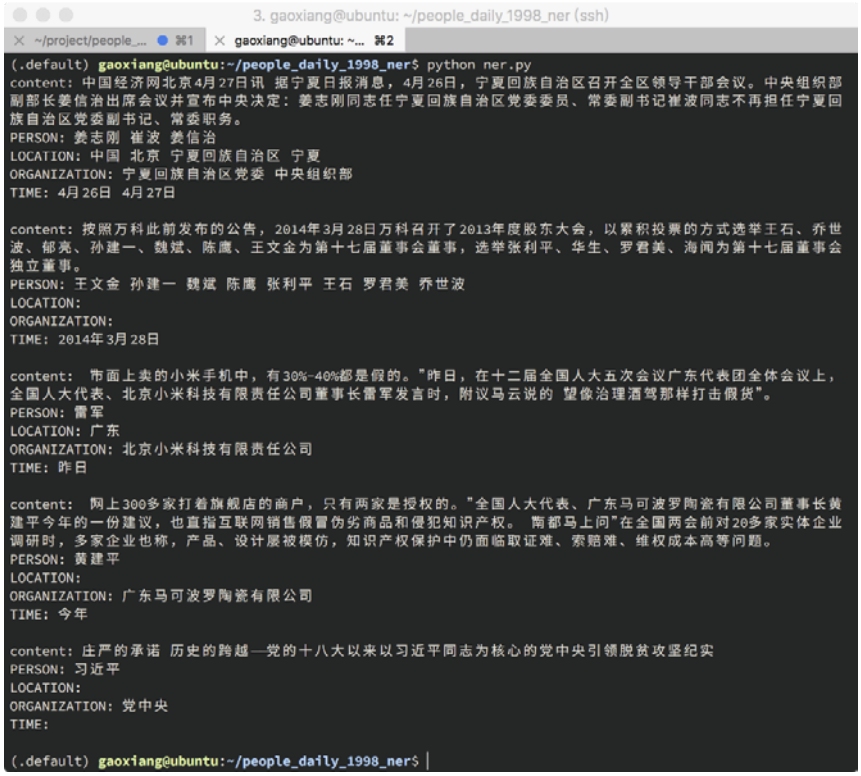


图 3.8 使用 python 代码进行 NER 的预测结果

图 3.8 展示了较好的结果，能够识别出诸如“北京小米科技有限责任公司”这样的组织机构名。通过使用 1998 年的数据识别出了 2010 年才成立的公司，这就是模型算法的力量。当然模型也有瑕疵，诸如“郁亮”、“海闻”这样的人名没有识别出来，这除了和模型特征选择相关外，也和语料库的规模和标注有关，因此语料库的建设和积累更加重要。

4. 总结

本文讲述了 NER 任务的基本概念及方法，并使用业界成熟的语料和工具开发了一个简单能 work 的基于 CRF 模型的中文 NER 系统，而实际提供线上服务的 NER 系统要比这个复杂的多。在自然语言处理的实际工作中，除了不同模型、算法、工具的使用和参数调优外，语料库的选择和积累也非常重要。对于中文文本语义分析技术，达观数据拥有多年的技术积累并紧跟行业潮流，对已有成熟技术进行深挖，对新兴技术进行研究集成。同时，针对不同行业及任务积累了丰富的文本语料，并源源不断地使用新数据对语料模型进行升级更新，保证分析结果的准确性和实时性，为客户提供高品质服务。

【本文作者简介】

高翔，达观数据联合创始人，达观数据前端项目组、文本语义理解组负责人，自然语言处理技术专家，上海交通大学通信专业硕士，曾代表达观数据参加 2016 青年互联网创业大赛并赢得全国总冠军荣誉、第五届中国创新创业大赛优秀企业奖、中国电子 i+ 创新创效创意大赛总决赛一等奖。曾就职于腾讯文学，盛大文学，盛大创新院，负责文本阅读类产品、搜索引擎、文本挖掘及大数据调度系统的开发工作。在自然语言处理和机器学习等技术方向有着丰富的经验。

垂直搜索引擎实践篇

垂直搜索引擎利用知识图谱和语义分析技术，帮助用户从海量信息中快速准确搜索到目标内容，为客户搭建出高效精准的智能搜索系统

【本篇关键词】倒排索引；搜索排序；QUERY自动纠错；智能问答

搜索引擎之倒排索引解读

冯仁杰　达观数据软件工程师

互联网时代，信息纷繁海量，人们通过搜索引擎直达“心中所想”已是常态。那么搜索引擎到底是如何高效查找目标内容呢？本文主要介绍搜索引擎里一个比较重要的结构——倒排索引。

1. 倒排索引简介

倒排索引（**Inverted Index**），是一种索引方法，常被用于全文检索系统中的一种单词文档映射结构。现代搜索引擎绝大多数的索引都是基于倒排索引来进行构建的，这源于在实际应用当中，用户在使用搜索引擎查找信息时往往只输入信息中的某个属性关键字，如一些用户不记得歌名，会输入歌词来查找歌名；输入某个节目内容片段来查找该节目等等。

面对海量的信息数据，为满足用户需求，顺应信息时代快速获取信息的趋势，聪明的开发者们在进行搜索引擎开发时对这些信息数据进行逆向运算，研发了“关键词——文档”形式的一种映射结构，实现了通过物品属性信息对物品进行映射，可以帮助用户快速定位到目标信息，极大地降低了信息获取难度。倒排索引又叫反向索引，它是一种逆向思维运算，是现代信息检索领域里面最有效的一种索引结构。

2. 倒排索引 &FAQ

从用户请求到结果返回，许多朋友会对倒排索引在检索系统中的工作过程产生好奇，本小节就倒排索引的一些常规认识，有如下问题：

Q1 何为索引？倒排索引又是什么？

索引，是为了加快信息查找过程，基于目标信息内容预先创建的一种储存结构。例如：一本书，没有目录，理论上也是可读的，只是当你合上当前在读的内容时，下次再翻开书本去查找，就比较耗费时间了。如果增加几页目录，我们可以快速地了解书本的大体内容分布，以及每一个章节页面位置的分布情况，这样我们查询内容的效率自然就会提高。书的目录，就是书本内容一种简单索引。

倒排索引，是索引技术中的一种，它是基于信息主体的关键属性值进行构建的。

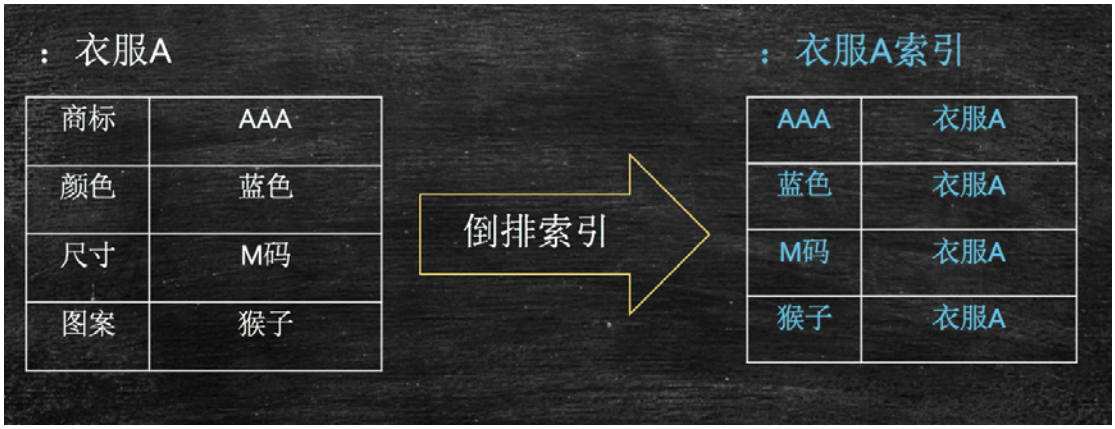


图 2.1 倒排索引概念示例图

假设检索系统中只有一个商品——衣服 A，基于该商品构建其倒排索引结构之后，会产生上图右表中的索引结构，这样用户可以通过搜“AAA”，“蓝色”，“M 码”，“猴子”，均可找到该商品，加快了检索速度，扩大了检索范围。

Q2 当接受到用户查询请求时，倒排索引中发生了什么？

一般地，当接受到用户查询请求时，进入到倒排索引进行检索时，在返回结果的过程中，主要有以下几个步骤：

Step1：在分词系统对用户请求等原始 Query 进行分析，产生对应的 terms；

Step2：terms 在倒排索引中的词项列表中查找对应的 terms 的结果列表；

Step3：对结果列表数据进行微运算，如：计算文档静态分，文档相关性等；

Step4：基于上述运算得分对文档进行综合排序，最后返回结果给用户。

上述该过程是较为简洁的一个检索过程。事实上，在生产环境中因为业务环境的繁杂，会使得索引的设计模式变得复杂且繁多。前文主要通过概念图来介绍倒排索引的架构体系，一个成熟的检索系统往往拥有一套较为稳定的算法体系，用于处理生产环境中的每一处细节技术需求。上述步骤中涉及了大量相关的数据储存技术、查找算法、排序算法、文本处理技术甚至 I/O 技术等等。

3. 倒排索引技术剖析

构建倒排索引是搜索引擎里面至关重要的一个步骤，从技术层面去分析，对于构造一个倒排索引，主要分为两部分：1）Doc2term 词项构造；2）倒排记录表的构建。

3.1 term 词项构造

词项构造是在构建索引过程中必不可或缺的一个步骤，词项构造效果的好坏往往会直接影响到用户的搜索体验，以及搜索结果的召回。该过程主要是利用分词系统将文档中的各项属性的文本信息拆分成一些表意较强且重要的词汇，便于用户查找。

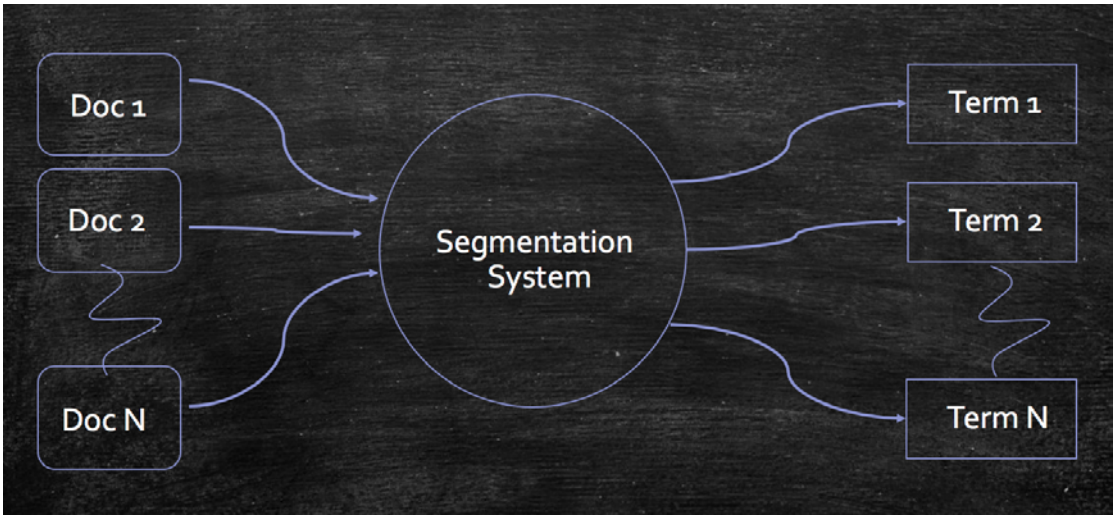


图 3.1 词项构造概念图

在词项构造的过程中，利用分词系统对文本进行处理时往往涉及到很多方面的问题，而且对于不同语种，会有不同的处理机制。

下面主要介绍在处理文本时涉及到的几个问题：

1）文本词条化

一段文本信息，它本身是一个由语言组成的字符串系列。本项技术点的主要任务是将一段连续的文本序列信息拆分成多个子序列。它与语言本身相关，面对不同的语言，处理文本的方式往往会不一样。

对于中文，由于其语言多歧义且表意紧凑的特性，在实际应用中，一般需要借助 NLP 的相关技术对内容进行特征抽取，甚至人工标注等，生成对应的词典，随后再基于词典利用分词器进行分词，才能看到较好的文本词条效果。

而对于英文，普遍的英文句子，段落内容，它会以空格符作为单词之间的分隔符，所以一般情况下，以空格符对英文内容进行拆分，已经可以取得比较好的效果。不过英文中也会存在一些特殊模式，如带上撇号的格式——“Teacher’s office”，连字符格式——“English-speaking”，也需要进行对应的处理，把单词提取出来。

2）停用词过滤

停用词是指在文档列表中出现的频数较高且价值不大的词。以英文为例，在英文文档中出现次数较多的停用词如：”is”、”the”、”I”、“and”、”me”等等；这一类词语在往往出现在所有文档中，若以此类词语为 term 进行索引构建，则会产生多个全量文档索引列表。停用词过滤的使用往往依赖于实际使用场景，关键字查询使用得较为频繁的场景如某一个电商品牌的垂直型搜索引擎，一个合适的停用词表显得尤为重要；而对于 Web 搜索引擎如百度、Google 等，该类型的搜索引擎面向的查询场景较多，通用性较强，往往不需要停用词过滤。

3）词条归一化

基于上述两点，将文档内容转换成一个或多个 term 后，在查询时，最理想的情况是用户输入的关键字刚好与 term 完全匹配。实际上，很多时候用户输入的 query 与词条之间往往不会完全匹配，而用户们还是希望 query 能与词条进行匹配，比如用户在查询“color”时，用户肯定也希望能看到关于“colour”的返回结果。

词条归一化的任务就是将一些看起来不完全一致的词条划分为一个等价类，比如英式单词 colour 和美式单词 color 归为一类、Air-conditioner 和 air-conditioner 归为一类等等；这样，用户在查询时，只要对等价类中的任意单词进行搜索，都会返回包含等价类中的任意一个单词的文档。

4）词干提取、词形还原

这是词条规范化的两种重要方式，用于扩展检索范围。词干提取的主要思想是“缩减”，将词条转化为词干，如：将“beaches”处理成“beach”，将“bananas”处理成“banana”等；词形还原的主要思想是“转换”，如：将“doing”、“done”、“did”转化成原型“do”，将“given”、“gave”转化成原型“give”等；词干提取的实现方法一般是基于规则对词条后缀进行缩减，至于词形还原，其实现方法需要词典来进行词形变化的映射；基于在此

结合词条归一化技术，对扩展检索范围会产生一定的正向作用。

3.2 倒排记录表的构建

倒排记录表的构建过程面向的是海量的文档数据集合，在大小规模上它比词项集合要大得多，无法完全存放在内存当中，需要写入磁盘。因此，在构建倒排记录表时我们有必要为内存的使用作考虑。

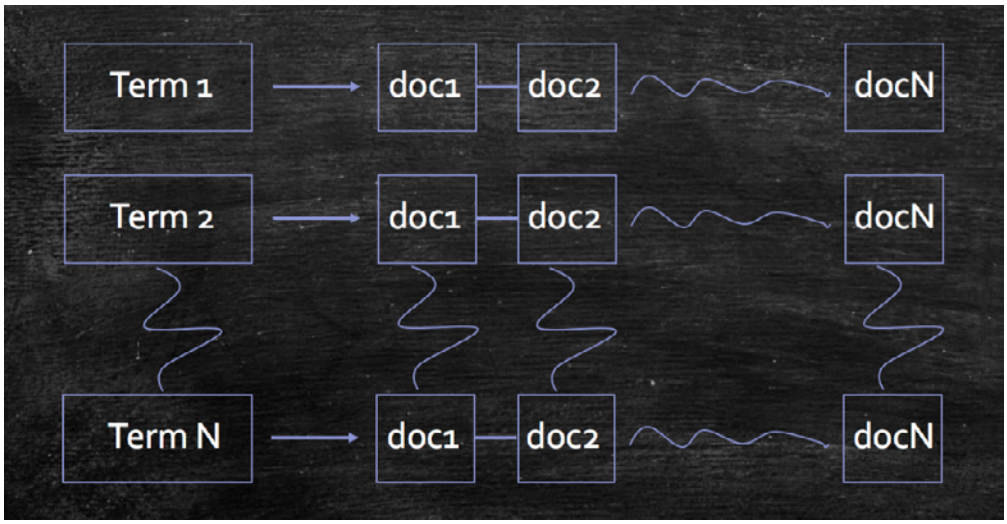


图 3.2 倒排索引概念图

在无法全内存的情况下，倒排记录表的主要构建思想是“分割”，亦即基于一定的处理逻辑对全量文档集合进行等份的批量处理。对于不同的业务需求，构建倒排记录表的方法往往会不一样。

基本的构建方法如下：

1) 通过一系列的处理将文档集合转化为“词项 ID—文档 ID”对；

2) 对词项 ID、文档 ID 进行排序，将具有相同词项对文档 ID 归并到该词项所对应的倒排记录表中，效果如图 3 所示；

3) 将上述步骤产生的倒排索引写入磁盘，生成中间文件；

4) 将上述所有的中间文件合并成最终的倒排索引。

从业务应用场景的角度出发，倒排记录表的构建方法主要有：单遍扫描和多遍扫描；从工程角度出发，倒排记录表的构建方法主要有：分布式构建和动态构建。

3.2.1 单遍扫描构建

顾名思义， 单遍扫描指的是仅对文档集合进行一次遍历，即可完成倒排索引的构建。由于内存开销问题，会将全量文档集进行分割，转换成几个内存大小相同的文档集合，然后依次执行前文中提及到的构建方法。该方法能快速构建一个简单可行的倒排索引，帮助用户通过关键字匹配快速找到目标文档。

3.2.2 多遍扫描构建

多遍扫描主要用于构建索引时获取关于文档的更多相关信息，如一些词项 TF-IDF 指标、词频、文档内容关系等，以丰富倒排记录表的内容，为搜索引擎进行功能扩充；在工业流水线上，单遍扫描构建索引由于其查询类型的丰富度不够，显然已经不能满足广大用户的需求了。搜索用户的需求并不止于关键字查询，像短语查询、模糊查询、精确筛选、模糊筛选、排序、聚合统计等等需求。这意味着我们在构建倒排列表时要尽可能获取文档的更多信息，便于查询时的微运算、重排序、相关性分析等技术需求。

3.2.3 分布式构建

对于一些大型搜索引擎如 Web 搜索引擎，单台机器已无法支撑其索引构建，需要多台机器组成集群对其进行分布式处理，将构建成的倒排索引进行分割，分布在多台机器上，每台机器各自形成独立的索引结构，当用户发出请求时，会有多台机器响应，并且根据用户的搜索需求在各自的索引结构进行查询，返回相关结果，再将所有结果在内存中进行集中处理，最后把处理过的最优结果返回给用户。在具体的实现过程中，工程师们往往更钟情于一些通用的面向大规模机器计算的分布式架构如 Hadoop 中的 MapReduce、Java 中的 Fork/join 架构等，极大地提高了软件开发效率。

3.2.4 动态构建

该方法中的文档集合是变化的，这要求在对文档集进行索引构建时也要对文档的更新进行自适应。此问题常见于电商领域里，如商品的上下架、商品内容的更新等，都会引发索引的动态更新问题。于此，我们常采取一些策略型方法来解决该类型的问题，提高索引的实时性。

常见的策略如下两种：

1）周期性对文档进行全量重建索引；

2）基于主索引的前提下，构建辅助索引，用于储存新文档，维护于内存中，当辅助索引达到一定的内存占用时，写入磁盘与主索引进行合并；

策略 1）是最简单直接、且有效的索引更新策略，对于数量级较大的搜索引擎来说处理简单便捷，由于动态索引计算的复杂性，使用其它策略会使得索引难维护，甚至引发严重的性能问题。所以大型搜索引擎往往更倾向于周期性重建索引，不过这会涉及到索引热切换的问题，大量的文档经常会产生持续性的文档更新情况，这对于索引热切换时会造成一定的困难，处理不好会导致数据丢失，用户查不到新文档等问题。

策略 2）中在进行主辅索引合并时会遇到比较大的储存开销，由于文档量较大，这意味着在进行合并操作时会涉及到大量倒排文件的读写操作，要想将该过程高效化，目前能处理该问题的文件系统极其稀少，所以该策略在生产环境中往往可用性并不高。

4. 总结

在实际生产环境中，由于业务的繁杂，倒排索引的技术体系会比本文所阐述的技术点要复杂得多。本文主要讲解了倒排索引的作用、索引构建方法、用户行为分析以及索引的应用场景，从整体出发，向大家介绍现代倒排索引大致的技术体系，帮助大家了解倒排索引的概念，了解搜索引擎。可能本文阐述的技术点、架构体系会因为笔者个人的理解偏差而存在一些不足或欠缺。如有疑问，欢迎交流。

【本文作者简介】

冯仁杰，达观数据搜索组研发技术人员，负责搜索离线数据统计及相关服务模块的日常维护管理，参与搜索引擎架构的工程设计、搜索集群健康状况监控模块的开发及维护等。

搜索引擎排序实践

桂洪冠　达观数据技术副总裁

随着互联网的深入发展，人类已然进入大数据时代。如何在浩瀚的数据海洋里高速有效地获取有价值的信息，正是促使大数据技术走向众多企业的关键。搜索引擎作为获取信息的有效入口，已然经历了 20 多年的发展，并一直试图理解用户搜索意图以及提升搜索的精准性。

Google 是全球性的搜索引擎，看似简单的搜索框背后隐藏的是极其复杂的系统架构和搜索算法，其中排序（以下统称 Ranking）的架构和算法更是关键部分。Google 正是通过 PageRank 算法深刻改变搜索排序而一举击败众多竞争对手。

Ranking 是搜索引擎的核心技术，本文以搜索引擎的 Ranking 技术为切入点，从搜索引擎架构、检索模型、机器学习算法、点击模型、搜索效果评估等方面将达观数据（htttp://www.datagrand.com）在搜索引擎 Ranking 的构建与优化过程中的一些实践经验与大家做分享。

1. 经典搜索排序架构

通常在线搜索引擎要求实时响应（毫秒级）用户的搜索请求，使得在线对每个文档进行基于模型的 Ranking 复杂计算不太现实，因而搜索的过程被分成两个阶段。

阶段一是使用相对简单的常用检索模型对用户 query 从索引中快速检索出 Top-k 候选结果集。常用检索模型主要有向量空间模型 (Vector Space Model)、布尔模型 (Boolean Model)、概率检索模型 BM25 等，通常 Top-k 的候选集选取还结合离线计算质量分高的文档以排除掉文本相关但质量分太低的文档；

阶段二则使用计算相对复杂的机器学习排序模型对 Top-k 候选结果集进行精确的重排序，因为 Top-K 的候选结果集数据量级一般不会很大，这一步计算可控。

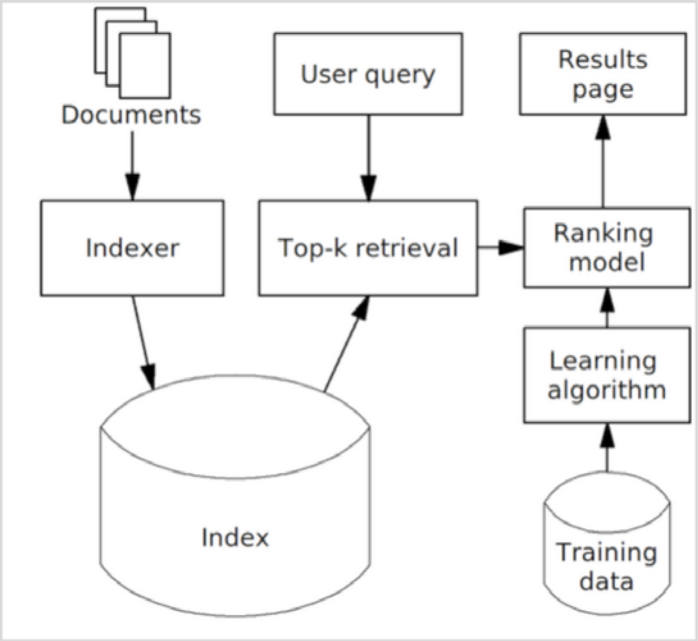


图 1.1 一个经典的搜索引擎排序架构

Ranking 模型的训练数据主要由 query、文档以及 query 与文档的相关度组成，相关度可以标记成好、不好两个级别或细粒度更高的 Perfect、Excellent、Good、Fair、Bad 五个级别。训练数据主要有两种获取方式：方式一是由搜索评测人员标记 query 与每个文档的相关度，进行手工的评测整理；方式二是通过自动分析搜索点击日志生成。

显然，对于大规模机器学习排序模型的训练数据，人工标注的成本过高，而且人工也无法对模型进行相对实时的更新。达观数据主要通过方式二生成训练数据，自动分析搜索点击日志，分析用户在同一个搜索 session 内对 query 的各种变换、对搜索结果中不同位置的文档的点击行为以及后继的筛选、翻页等行为，综合计算出一个可以标记训练数据的搜索满意度得分。

达观搜索的实践表明，通过分析搜索点击日志可以实现模型训练数据的自动生成和实时更新，同时也可以达到比较满意的搜索效果。

2. 达观搜索引擎架构

图 2.1 达观搜索引擎架构

达观搜索引擎架构从底往上分别是分布式数据存储层、索引构建与模型训练层、索引数据与模型数据分发层、搜索核心层、开放接口层，同时系统架构还支持搜索引擎的索引配置、Ranking 策略配置、搜索分析以及效果评估。

搜索核心层是由 query 分析引擎、索引引擎、Ranking 引擎构成。其中 query 分析引擎（QUERY ANALYSIS ENGINE）负责对用户的 query 进行语义分析和意图识别，包括 query 分词、中心词提取、query 纠错、query 自动提示、query 扩展等。

索引引擎（INDEX ENGINE）执行 Top-k 候选结果选取，这里我们综合考虑了检索模型的搜索相关性评分和文档的静态质量分（离线计算），另外在这一层还根据用户的筛选条件以及业务层面的搜索结果配置进行了搜索结果的筛选和融合。

排序引擎（RANKING ENGINE）利用机器学习模型对 Top-k 的候选集执行第二轮的精确排序。RANKING ENGINE 内置一个算法插件框架，可以根据用户配置的搜索排序策略加载相应的排序算法插件以及排序算法模型，同时还支持用户使用对搜索流量划分到不同的排序算法插件，以实现多个算法策略的同时在线 A/B testing 对比。

3. 检索模型的选择

常见的检索模型主要有布尔模型 (Boolean Model)、向量空间模型 (Vector Space Model)、概率检索模型 BM25 与 BM25F。

3.1 布尔模型

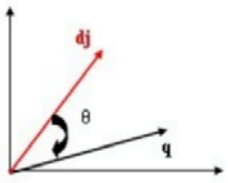
布尔（Boolean）模型是基于集合论和布尔代数的一种简单检索模型。它的特点是查找那些对于某个查询词返回为“真”的文档。在该模型中，一个查询词就是一个布尔表达式，包括关键词以及逻辑运算符。通过布尔表达式，可以表达用户希望文档所具有的特征。由于集合的定义是非常直观的，Boolean 模型提供了一个信息检索系统用户容易掌握的框架。查询串通常以语义精确的布尔表达式的方式输入。

布尔模型的主要优点是直观和简单，缺陷在于完全匹配会导致被返回的结果文档太多或者太少。

3.2 向量空间模型 (Vector Space Model，VSM)

VSM 概念简单，即把对文本内容的处理简化为向量空间中的向量运算，并且它以空间上的相似度表达语义的相似度，直观易懂。当文档被表示为文档空间的向量，就可以通过计算向量之间的相似性来度量文档间的相似性。文本处理中最常用的相似性度量方式是余弦距离。

$$Cosine(d_j,q)=\frac{d_j\cdot q}{|d_j|*|q|}=\frac{\sum_{i=1}^{t} w_{ij}*w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{ij}^2}*\sqrt{\sum_{i=1}^{t} w_{iq}^2}}$$



向量空间模型中通常采用 TF* IDF 的方式计算权重。Wij = TFij * IDFij 表示 termi 在文档 dj 的权重，Wiq = TFiq * IDFiq 表示 termi 在 query q 中的权重。

VSM 的优点：

1) 对 term 的权重的计算可以通过对 term 出现频率的统计方法自动完成，使问题的复杂性大为降；

2) 支持部分匹配和近似匹配，并可以根据 query 和文档之间的相似度对结果进行排序。

VSM 缺点：

1) 基于 term 之间的独立性假设，也即权重计算没有考虑 term 之间的位置关系，也没有考虑 term 的长度对权重的影响；

2) 计算量大。新文档加入需要重新计算 term 的权重。

3.3 概率检索模型

概率统计检索模型 (Probabilistic Retrieval Model) 是另一种普遍使用的信息检索算法模型，它应用文档与查询相关的概率来计算文档与查询的相似度。

3.3.1 二元独立模型 (BIM)

词汇独立性假设 : 文档里面出现的词没有任何关联，这样一个文档的出现就可以转为各个单词出现概率的乘积。

对于同时出现查询 qi 以及文档 di 的时候，对 qi 在 di 中出现的单词进行“相关文档 / 不相关文档”统计，即可得到查询与文档的相关性估计值

$$\sum_{q_i=d_i=1} log\frac{(r_i+0.5)((N-R)-(n_i-r_i)+0.5)}{(n_i-r_i+0.5)(R-r_i+0.5)}$$

其中：

N 表示是文档集中总的文档数；

R 表示与 query 相关的文档数；

ri 表示与 query 相关的文档中含有的第 i 个 term 文档个数；

ni 表示含有的第 i 个 term 文档总数；

0.5 是平滑因子，避免出现 log(0)。

3.3.2 BM25 模型

BM25 模型在 BIM 模型的基础上考虑了查询词在 Query 以及 Doc 中的权重，并通过实验引入了一些经验参数。BM25 模型是目前最成功的内容排序模型。

改进之后的 BM25 模型的拟合公式如下 :

$$\sum_{i\in Q} log\frac{(r_i+0.5)((N-R)-(n_i-r_i)+0.5)}{(n_i-r_i+0.5)(R-r_i+0.5)} \cdot \frac{(k_1+1)f_i}{K+f_i} \cdot \frac{(k_2+1)qf_i}{k_2+qf_i}$$

公式的第 1 部分同 BIM 独立模型，公式的第 2 部分是查询词的 term 在 Doc 中的权重，第 3 部分是查询词的 term 在查询本身的权重。

fi 表示 term 在 D 中的词频，K 因子表示文档长度的考虑，其计算公式为：

$$K = k_1((1-b) + b \cdot \frac{dl}{avdl})$$

其中：

k1 为经验参数， k1 一般设置为 1.2；

b 为调节因子，将 b 设为 0 时，文档长度因素将不起作用，经验表明一般 b=0.75；

dl 代表当前文档的长度；

avdl 代表所有文档的平均长度；

qfi 表示在查询中的词频，k2 也为调节因子，因为在短查询下这部分一般为 1，为了放大这部分的差异，k2 一般取值为 0~1000。

综上所述，BM25 模型结合了 BIM 因子、文档长度、文档词频和查询词频进行公式融合，并利用 k1，k2，b 对各种因子进行权重的调整。

3.3.3 BM25F 模型

BM25F 模型对 BM25 模型的改进之处在于考虑了文档不同区域的加权统计，例如文档的标题和描述被赋予了不同的区域权重，在各个不同区域分别统计词频。

BM25F 模型的计算公式为：

$$\sum_{i:q_i=d_i=1} log\frac{(r_i+0.5)((N-R)-(n_i-r_i)+0.5)}{(n_i-r_i+0.5)(R-r_i+0.5)} \times \frac{f_i^u}{k_1+f_i^u}$$

$$f_i^u = \sum_{k=1}^{u} w_k \times \frac{f_{ui}}{B_u}$$

$$B_u = ((1-b_u) + b_u \times \frac{ul_u}{uvul_u})$$

其中：

文档 D 来自不同的 u 个域；

各个域对应的全总为 Wk；

f_{ui} 表示词频；

B_u 表示各个域的长度；

ul_u 为域的实际长度，$uvul_u$ 表示域的平均长度；

b_u 为各个域长度的调节因子。

3.4 检索模型总结

每种检索模型各有千秋，适用于不同的场景和应用。布尔模型、空间向量模型、概率模型等传统检索模型的排序方法一般通过构造相关性函数实现，然后按照相关性进行排序。检索模型尤其概率模型比较适用于内容相关性排序，但内容相关性一般仅考虑 query 和 doc 的 tf，idf，dl，avdl 等因素，很难融合点击反馈、文档质量分、点击模型等更多的排序因素。一个大型搜索引擎排序因子往往多达数十个乃至上百个（Google 搜索排序因子超过 200 个），如果模型中参数过多，调参会变得非常困难，也很容易导致过拟合现象。

但正如前文所述，搜索引擎需要快速响应用户搜索请求，无法在毫秒级时间内对每一个召回结果进行精确的机器学习排序。业界的主流的做法是首先进行第一轮的 Top-k 选取再对 Top-k 结果进行第二轮的精确重排序。传统检索模型尤其概率模型比较适用于文本内容相关性排序，能够满足快速获取 Top-k 候选结果集的需求。达观数据（www.datagrand.com）搜索在第一轮 Top-k 选取中选用的是 BM25F 检索模型。BM25F 模型相比 BM25 模型考虑了文档不同区域的加权统计，可以获得更好的文本相关性，是目前最优的文本检索模型。

机器学习排序（Machine Learning to rank）方法很容易融合多种特征，且有成熟深厚的数学理论基础，通过迭代优化参数，对于数据稀疏、过拟合等问题也有比较成熟的理论和实践。

4. 机器学习排序（Machine Learning to rank）

机器学习排序系统一般分为离线学习系统和在线预测排序系统。离线系统的设计需要靠特征的选择、训练集的标注、MLR 方法的选定、确定损失函数、以最小化损失函数为目标进行优化，以获取排序模型的相关参数。在线预测排序系统将待预测结果输入到机器学习得到的排序模型，即可得到结果的相关性得分，进而依据相关性得分得到搜素结果的最终排序。

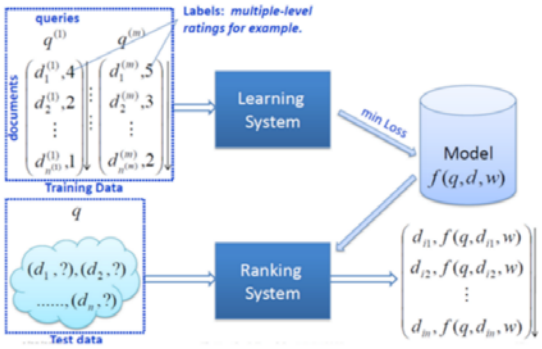


图 4.1 机器学习排序系统框架

排序模型的选择直接影响在线预测的效果。在类似电商时效性强的应用场景中，业务上经常需要根据商品库存、价格等变化及时调整排序结果，由于排序模型的高度复杂性，人工干预只能做局部小范围的调整，更多的还是要对模型进行实时的自动化更新。

对于这个问题，达观数据（www.datagrand.com）在实践中总结出了一个在线 - 近线 - 离线的三层系统架构，即 Online-Nearline-Offline（在线 - 近线 - 离线）三层混合机制。离线系统负责 day 级全量训练数据的学习、近线系统负责 hour 级模型的学习与更新、在线系统负责 minut 级的准实时反馈数据的学习与模型的更新。

4.1 特征选取与特征工程

特征是算法、模型的养料之源。特征选择的好坏直接关系到算法训练学习出的模型的效果。与传统的文本分类不同，MLR 输出的是给定 query 的文档集合的排序，不仅要考虑文档自身的特征，还要考虑 query 与文档关联关系的特征。

综合来说，MLR 需要考虑三个方面的特征：

1）文档本身的静态特征，包括文档的文本特征，如带权重的词向量，文档不同域（主标题、段落标题、描述内容、锚文本、URL 链接等）的 TF、IDF、BM25 和其他语言模型得分，也包括文档的质量分、网页文档的 PageRank 等重要性得分。关于文档的质量分，达观搜索根据不同的业务场景有不同的计算指标，比如电商相关的商品的质量分计算除了要考虑商品本身的文本与图片丰富度，更多的还要考虑商品的各种业务指标如销量变化、收藏、价格、库存、类别、上架时间、评论、商家信誉等级、是否作弊等，而媒体相关的文章的则需要考虑阅读数、转发数、赞数、收藏、评论、发文时间、主题类型等。

2) 文档和 query 关联的特征，比如 query 对应文档的 TD-IDF score， BM25 score 等。

3) query 本身的特征，比如文本特征，带权重的词向量，query 长度，query 所述的分类或主题，query 的 BM25 的 sum/avg/min/max/median 分数，query 上个月的热度等。

在 query 与文档的特征工程中，除了从词法上分析，还需要从“被阐述”的词法所“真正想表达”的语义即概念上进行分析提取。比如一词多义，同义词和近义词，不同的场景下同一个词表达不同的意思，不同场景下不同的词也可能表达相同的意思。LSA（隐语义分析）是处理这类问题的著名技术，其主要思想是映射高维向量空间到低维的潜在语义空间或概念空间，也即进行降维。

具体做法是将词项文档矩阵做奇异值分解（SVD）

$$C = U\Sigma V^T$$

其中 C 是以文档为行，词项 terms 为列的矩阵（假设 M x N），元素为 term 的 tf-idf 值。C 被分解成 3 个小矩阵相乘；U 的每一列表示一个主题，其中的每个非零元素表示一个主题与一篇文章的相关性，数值越大越相关；V 表示 keyword 与所有 term 的相关性；Σ 表示文章主题和 keyword 之间的相关性。

4.2 MLR 算法的选择

MLR 一般来说有三类方法：单文档方法（Pointwise），文档对方法（Pairwise），文档列表方法（Listwise）。

4.2.1 Pointwise 方法

Pointwise 把文档当成单个的点分别进行计算，实际上把一个 Ranking 问题转化成二值分类问题、回归问题或多值分类问题。Query 与文档之间的相关度作为 label，label 一般划分为 : {Perfect, Excellent, Good, Fair, Bad} 。

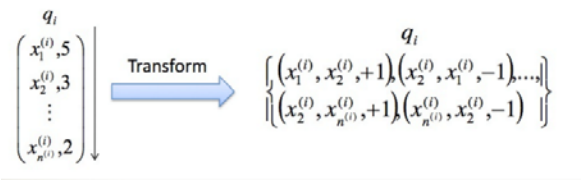
Pointwise 方法主要包括：Pranking (NIPS 2002)， OAP-BPM (EMCL 2003)， Ranking with Large Margin Principles (NIPS 2002)， Constraint Ordinal Regression (ICML 2005)

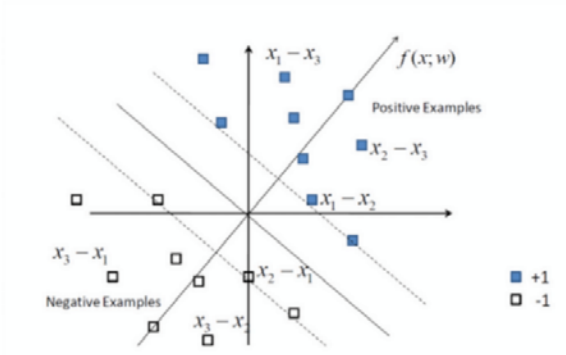
Pointwise 的不足之处：

Pointwise 使用传统的分类，回归或者 Ordinal Regression 来对给定 query 下的单个文档的相关度进行建模，没有文档位置对排序结果的影响，而回归和分类的损失函数会尽量拟合所有的数据，算法为了整体损失最小，有可能把排在前面的文档的损失变得更大，或者把排在后面的文档的损失变得更小，从而导致排序难以取得良好的效果。

4.2.2 Pairwise 方法

在 Pairwise 中 query 与文档对 <di， dj> 结合，假设在同一 Query 下，di 的相关性大于 dj，那么我们可以把 di-dj 标记为 +1，dj-di 标记为 -1，从而可以把原问题转换为一个分类或回归问题。





Pairwise 方法主要包括：Ranking SVM (ICANN 1999)， RankBoost (JMLR 2003)， LDM (SIGIR 2005)， RankNet (ICML 2005)， Frank (SIGIR 2007)， GBRank (SIGIR 2007)， QBRank (NIPS 2007)， MPRank (ICML 2007)， IRSVM (SIGIR 2006) 。

Pairwise 的不足：

1） 文档较多时，pair 的数目是平方级增长的，计算量太大；

2） Pair 对不同级别之间的区分度一致对待，没有对排在前面的结果作更好的区分。对于搜索引擎而言，用户更倾向于点击前几页的结果；

3）相关文档集大小带来模型的偏置。如果一个 query 下文档远多于另一 query， 支持向量就会向该 query 偏置，导致分类器对后者区分不好。

4.2.3 Listwise 方法

Listwise 的输入是 query 对应的一个文档列表，计算每个 query 对应的文档列表的得分。

Listwise 有一种基于文档排列的概率分布进行训练的方法，通过对训练实例的训练找到一个最优的打分函数 f， 使得 f 对 query 的打分结果的概率分布与训练数据的实际排序尽可能相同。损失是按照训练数据的实际排序概率分布与模型输出的概率分布之间的 KL 距离来度量的。

$$P(\pi)=\prod_{i=1}^{n}\frac{s_{\pi(i)}}{\sum_{j=i}^{n}s_{\pi(j)}}$$

$$P(\mathrm{ABC})=\frac{s_A}{s_A+s_B+s_C}\cdot\frac{s_B}{s_B+s_C}\cdot\frac{s_C}{s_C}$$

P(A ranked No.1)

P(B ranked No.2 | A ranked No.1)

P(C ranked No.3 | A ranked No.1, B ranked No.2)

Listwise 算法主要包括：LambdaRank (NIPS 2006)，AdaRank (SIGIR 2007)， SVM-MAP (SIGIR 2007)， SoftRank (LR4IR 2007)， GPRank (LR4IR 2007)， CCA (SIGIR 2007)， RankCosine (IP&M 2007)， List-Net (ICML 2007)， ListMLE (ICML 2008)，p-ListMLE 。

相比于 Pointwise 和 Pairwise 方法，Listwise 方法直接优化给定查询下整个文档集合的序列，所以比较好地解决了以上算法的缺陷。Listwise 方法中的 LambdaMART(是对 RankNet 和 LambdaRank 的改进) 在 Yahoo Learning to Rank Challenge 表现出最好的性能。

达观数据（www.datagrand.com）在搜索排序中使用了一种 position-aware ListMLE(p-ListMLE) 的算法，ListMLE 考虑了排序位置信息，但没有对不同位置的重要程度进行区分。达观数据搜索的实践显示同样的条件下 p-ListMLE 的搜索效果指标 nDCG 要优于 ListMLE。

5. 点击模型

我们在排序实践中还发现MLR无法充分利用用户对搜索结果的点击反馈。俗话说群众的眼睛是雪亮的，用户对不同位置的搜索结果的点击行为直接反应了搜索结果的好坏。我们根据用户的历史点击记录生成了点击模型，通过点击模型对 MLR 的结果再进行一次调整。

点击模型又称为点击调权，搜索引擎根据用户对搜索结果的点击，可以挖掘出哪些结果更符合查询的需求。

点击模型基于如下基本假设：

1) 用户的浏览顺序是从上至下的；

2) 需求满足好的结果，整体点击率一定高；

3) 同一个 query 下，用户点击的最后一个结果之后的结果，可以假设用户已经不会去查看了（一定程度上减弱了位置偏见）；

4) 用户进行了翻页操作，或者有效的 query 变换，则可以认为前页的结果用户都浏览过，并且不太满意；

5) 用户点击的结果，如果引发后继的转化行为（比如电商搜索中的加购物车），则更有可能是用户满意的结果。

点击模型日志：

图 5.1 点击模型（日志收集）

达观数据搜索中 MLR 算法优化 + 点击模型对结果调权后，搜索效果的显著提升。

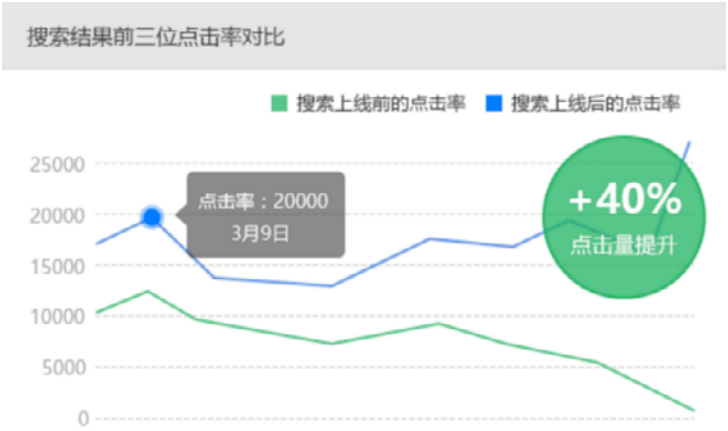


图 5.2 达观数据搜索上线前后的效果对比

6. 搜索排序效果评估

搜索引擎的排序是一个复杂的过程，特征的选择、算法的变化、模型的更新都会导致排序结果的变化。那么如何衡量一个排序结果的好坏呢?

MLR 是用机器学习的方法来进行排序，所以评价 MLR 效果的指标就是评价排序的指标，主要包括一下几种：

1） WTA(Winners take all) 对于给定的查询 q，如果模型返回的结果列表中，第一个文档是相关的，则 WTA(q)=1，否则为 0。

2） MRR(Mean Reciprocal Rank) 对于给定查询 q，如果第一个相关的文档位置是 R(q)，则 MRR(q)=1/R(q)。

3） MAP(Mean Average Precision) 对于每个真实相关的文档 d，考虑其在模型排序结果中的位置 P(d)，统计该位置之前文档集合的分类准确率，取所有这些准确率的平均值。

4）NDCG(Normalized Discounted Cumulative Gain) 是一种综合考虑模型排序结果和真实序列之间关系的一种指标，也是最常用的衡量排序结果指标，详见 Wikipedia。

6.1 评价指标的使用

使用评价指标主要有手工标注答案和自动化评估两种。手工标注方式既费时费力，又无法及时进行评估效果反馈。自动化评估方式对提高评估效率十分重要。最常用的自动评估方法是 A/B testing 系统。

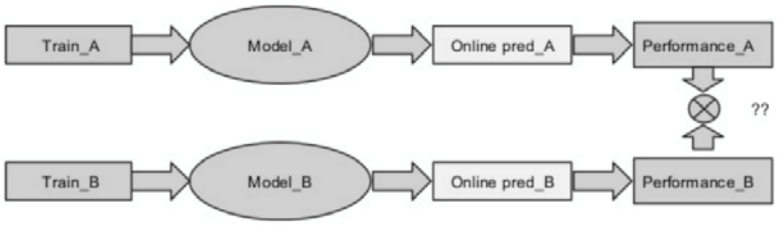


图 6.1 A/B testing 系统

A/B testing 系统将用户的流量在算法模型 A/B 之间进行分配，即将通过用户的分组号 (bucket id) 将用户流量分别导入不同的算法分支，用户在不同算法分支的行为连同分组号被记录下来，后台分析系统分析这些行为数据可以生成一系列对比指标，通过这些指标可以直观的分析算法模型优劣。

7. 总结

本文从搜索引擎排序的架构、检索模型、机器学习排序模型与算法到搜索效果评估，全面介绍了达观搜索引擎排序实践方面的一些经验。达观数据搜索团队长期致力于基于大数据的搜索算法优化，经过多年的积极探索，目前在开放搜索引擎的系统研发和效果提升方面已经积累了丰富的经验。随着 DT 时代的到来和深度学习兴起，达观数据技术团队将在基于大数据的深度挖掘方面不断探索和尝试以给用户带来更好的产品和服务。

【本文作者简介】

桂洪冠，达观数据联合创始人，中国计算机学会 CCF 会员，大数据技术专家。在参与创办达观数据前，曾在腾讯文学、阿里巴巴、新浪微博等知名企业担任大数据挖掘高级技术管理工作。桂洪冠在数据技术领域拥有 6 项国家发明专利，中国科学技术大学计算机硕士学位。在大数据架构与核心算法以及文本智能处理等领域有深厚的积累和丰富的实战经验。

扫描二维码申请试用
达观数据 NLP 产品

搜索引擎的 Query 自动纠错技术和架构

高翔　达观数据联合创始人

1. 背景

如今，搜索引擎是人们的获取信息最重要的方式之一，在搜索页面小小的输入框中，只需输入几个关键字，就能找到你感兴趣问题的相关网页。搜索巨头 Google，甚至已经使 Google 这个创造出来的单词成为动词，有问题 Google 一下就可以。在国内，百度也同样成为一个动词。除了通用搜索需求外，很多垂直细分领域的搜索需求也很旺盛，比如电商网站的产品搜索，文学网站的小说搜索等。面对这些需求，达观数据作为国内提供中文云搜索服务的高科技公司，为合作伙伴提供高质量的搜索技术服务，并进行搜索服务的统计分析等功能。

搜索引擎系统最基本最核心的功能是信息检索，找到含有关键字的网页或文档，然后按照一定排序将结果给出。在此基础之上，搜索引擎能够提供更多更复杂的功能来提升用户体验。对于一个成熟的搜索引擎系统，用户看似简单的搜索过程，需要在系统中经过多个环节，多个模块协同工作，才能提供一个让人满意的搜索结果。其中拼写纠错（Error Correction，以下简称 EC）是用户比较容易感知的一个功能，比如百度的纠错功能如下图所示：

图 1.1 百度纠错功能示例

EC 其实是属于 Query Rewrite（以下简称 QR）模块中的一个功能，QR 模块包括拼写纠错，同义改写，关联 query 等多个功能。QR 模块对于提升用户体验有着巨大的帮助，对于搜索质量不佳的 query 进行改写后能返回更好的搜索结果。QR 模块内容较多，以下着重介绍 EC 功能。

在搜索引擎中，我们将用户输入的关键字查询叫做 query。用户希望得到和输入 query 相关的质量较好的网页或文档，这个“好”字定义有多种衡量方式，最简单的标准就是那些对用户帮助最大最具吸引力的结果能够排到前列，搜索工程师们也在努力通过各种算法的提升来达到这个目的。

但是往往出于各种原因，用户输入的 query 本身质量不高或是错误的，如果搜索引擎不对这种错误进行修正弥补，会导致召回错误的结果，或者结果数少甚至没有结果。当用户看到搜索结果较差较少时，如果能意识到自己的 query 错误，对 query 进行修正再次检索，也许能找到想要的结果。但有时用户也不知道自己的 query 错在哪里，这个时候就会非常着急。

笔者之前从事搜索相关工作时，刚开始搜索系统不支持纠错功能，结果收到用户大量的吐槽和投诉，说明没有纠错功能的搜索系统会大大降低用户体验，不仅如此，这些错误 query 检索还浪费大量的流量。当开发完毕并在搜索系统中使用 EC 模块后，纠错成功的流量占到总流量的 2%，不仅提升了用户体验，还能够挽回流量损失，提升用户粘度。

2. EC 常见错误

EC 应该怎么做？首先我们看一下常见的 query 错误都有哪些。

对于英文，最基本的语义元素是单词，因此拼写错误主要分为两种，一种是 Non-word Error，指单词本身就是拼错的，比如将“happy”拼成“hbppy”，“hbppy”本身不是一个词。另外一种是 Real-word Error，指单词虽拼写正确但是结合上下文语境确是错误的，比如“two eyes”写成“too eyes”，“too”在这里是明显错误的拼写。

而对于中文，最小的语义单元是字，往往不会出现错字的情况，因为现在每个汉字几乎都是通过输入法输入设备，不像手写汉字也许会出错。虽然汉字可以单字成词，但是两个或以上的汉字组合成的词却是更常见的语义元素，这种组合带来了类似英文的 Non-word Error，比如“洗衣机”写成“洗一鸡”，虽然每个字是对的，但是整体却不是一个词，也就是所谓的别字。汉字也有类似 Real-word Error 的问题，比如加薪圣旨，加薪和圣旨都是正确的词，但是

两个连在一起确有问题，因此很多情况下汉语 query 纠错实际上是短语纠错问题。Query 除了纯汉字外，现在还会出现中英文混拼错误，中文拼音混拼等错误。

下图是笔者在搜索日志中找到的一些常见错误 query：

```
EC succeeded, orginal keyword is:[梦魂天帝], new keyword is:[梦魂天地], ec strategy is:0
EC succeeded, orginal keyword is:[yanyujiangnan], new keyword is:[烟雨江南], ec strategy is:0
EC succeeded, orginal keyword is:[tidu], new keyword is:[提督], ec strategy is:0
EC succeeded, orginal keyword is:[箭神], new keyword is:[剑神], ec strategy is:0
EC succeeded, orginal keyword is:[魔獸世界], new keyword is:[魔兽世界], ec strategy is:0
EC succeeded, orginal keyword is:[都世小农民], new keyword is:[都市小农民], ec strategy is:0
EC succeeded, orginal keyword is:[竣工吧主], new keyword is:[军工霸主], ec strategy is:0
EC succeeded, orginal keyword is:[流忙教师], new keyword is:[流氓教师], ec strategy is:0
EC succeeded, orginal keyword is:[shengyi], new keyword is:[圣衣], ec strategy is:0
EC succeeded, orginal keyword is:[荒龙诀], new keyword is:[皇龙诀], ec strategy is:0
EC succeeded, orginal keyword is:[综满之空夜], new keyword is:[综漫之空夜], ec strategy is:0
EC succeeded, orginal keyword is:[爆力电子业], new keyword is:[暴利电子业], ec strategy is:0
EC succeeded, orginal keyword is:[清未洋], new keyword is:[情未央], ec strategy is:0
EC succeeded, orginal keyword is:[zhutianzhijie], new keyword is:[诸天之劫], ec strategy is:0
EC succeeded, orginal keyword is:[进身保镖], new keyword is:[近身保镖], ec strategy is:0
EC succeeded, orginal keyword is:[wangyouzhibashishentou], new keyword is:[网游之霸世神偷], ec str
EC succeeded, orginal keyword is:[十香デッドエンド], new keyword is:[十香], ec strategy is:0
EC succeeded, orginal keyword is:[moshoushijie], new keyword is:[魔兽世界], ec strategy is:0
EC succeeded, orginal keyword is:[世上最牛召唤], new keyword is:[史上最牛召唤], ec strategy is:0
EC succeeded, orginal keyword is:[不良闲妻], new keyword is:[不良贤妻], ec strategy is:0
EC succeeded, orginal keyword is:[都世巨灵神], new keyword is:[都市巨灵神], ec strategy is:0
EC succeeded, orginal keyword is:[饶雪慢], new keyword is:[饶雪漫], ec strategy is:0
EC succeeded, orginal keyword is:[chongsheng], new keyword is:[重生], ec strategy is:0
EC succeeded, orginal keyword is:[超级抽检], new keyword is:[超级抽奖], ec strategy is:1
EC succeeded, orginal keyword is:[权cai], new keyword is:[权财], ec strategy is:0
EC succeeded, orginal keyword is:[斗渔], new keyword is:[斗鱼], ec strategy is:0
EC succeeded, orginal keyword is:[aogu], new keyword is:[傲骨], ec strategy is:0
EC succeeded, orginal keyword is:[zhangxiaohua], new keyword is:[张小花], ec strategy is:0
EC succeeded, orginal keyword is:[xiao'yao], new keyword is:[逍遥], ec strategy is:0
```

图 2.1 搜索日志中的错误 query

从上图可以看出，中文搜索中常见的错误 query 主要包括别字，纯拼音，模糊音，拼音汉字混合，拼音其他符号混合等多种问题。

3. Query 出错的原因分析

目前最普遍的中文输入方式是拼音输入法，用户输入拼音，输入法给出候选词，但是由于用户误选或无需要候选词时，query 就有可能出错。虽然相较之前智能输入法现在已经足够强大，但仍有一些新的产品、小说、影视作品，输入法可能会覆盖不到。比如一些新奇网络词汇的出现，传统的词典已经无法包括这些词。还有一些较为陌生的词，比如“芈月传”，很多人都是听朋友介绍很好看，结果去搜索引擎中搜索相关信息，很多人只知道第一个字发音是“mi”，但实际是哪个字却不确定。

除此之外，用户搜索时也会从网页或其他文档上复制粘贴文字来搜索，导致搜索 query 不完整或带入其他字符，甚至打字太快也是错误 query 输入的原因。

4. Query 纠错方案

英文拼写纠错已经有较长的历史，对于英文纠错的研究较多。英文纠错是中文纠错的重要基础，其中很多算法思想同样适用于中文。因此首先介绍一下英文纠错问题。在介绍具体纠错方案前，先介绍两个重要的的概念：编辑距离，n 元语法模型。

4.1 基础概念

4.1.1 编辑距离

编辑距离是两个字符串之间，由一个转换成另外一个所需要的最少操作次数，允许的操作包括字符替换，增加字符，减少字符，颠倒字符。举例来讲，apple 和 apply 的编辑距离是 1，access 和 actress 的编辑距离是 2，arrow 和 brown 的编辑距离是 3，编辑距离的计算操作过程如下图所示。

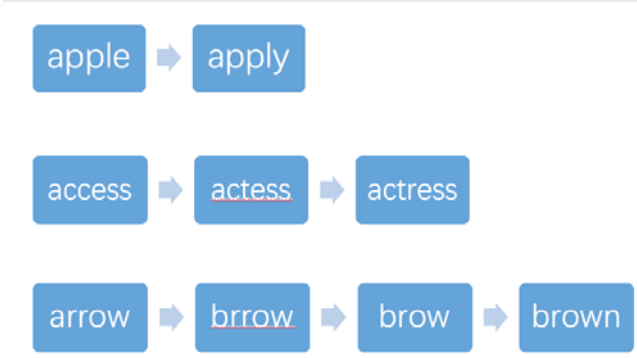


图 4.1 编辑距离计算过程

4.1.2 n 元语法模型

语言模型（language mode）在基于统计模型的语音识别，机器翻译，汉语自动分词和句法分析中有着广泛的应用，目前采用的主要是 n 元语法模型（n-gram model）。一个语言模型构建字符串的概率分布 p(W)，假设 p(W) 是字符串作为句子的概率，则概率由下边的公式计算：

$$P(w_1w_2\dots w_n)=\prod_i P(w_i\mid w_1w_2\dots w_{i-1})$$

其中 w1 表示第一个词，w2 表示第二个词，以此类推。p(w4|w1w2w3) 表示前面三个词是 w1w2w3 的情况下第四个词是 w4 的概率。

w1w2...wi-1 称作历史，如果 w 共有 5000 个不同的词，i=3 的时候就有 1250 亿个组合，但是训练数据或已有语料库数据不可能有这么多组合，并且绝大多数的组合不会出现，所以可以将 w1w2...wi−1 根据规则映射到等价类，最简单的方式就是取 wi 之前 n−1 个历史，根据马尔科夫假设，一个词只和他前面 n−1 个词相关性最高，这就是 n 元语法模型。

$$P(w_1w_2\dots w_n)\approx\prod_i P(w_i\mid w_{i-k}\dots w_{i-1})$$

通常用的 n 元语法模型包括 unigram，bigram，trigram，其中 unigram 表示这个词和前面的词无关，彼此独立，计算公式如下：

$$P(w_1w_2\dots w_n)\approx\prod_i P(w_i)$$

Bigram 表示一个词只和它前面一个词有关，计算公式如下：

$$P(w_i\mid w_1w_2\dots w_{i-1})\approx P(w_i\mid w_{i-1})$$

4.2 英文纠错

4.2.1 Non-word 纠错

纠错首先要检测出错误。检测错误的方法有很多种，对于 Non−word 错误可以使用语料库字典，如果输入词不在字典中，即可以判定为错词。

纠错过程就是找出和错词最相似的一些候选词，然后从中选出正确的纠错词。候选词可以使用上面介绍的编辑距离从语料库中找出。统计指出，80% 的错误词的编辑距离是 1，并且几乎所有的错误的编辑距离在 2 以内。

在候选词中找到最终的纠错词，比较简单的方法可以根据候选词的权重进行排序，给出权重最高的词作为纠错词，这个权重可以是人工标注的结果，也可以是语料库统计的词频或其他方式。相对复杂的候选词选择方法可以使用统计模型计算，比如噪声信道模型。

噪声信道模型（Noisy Channel Model）最早是香农为了模型化信道的通信问题，在信息熵概念上提出的模型，目标是优化噪声信道中信号传输的吞吐

量和准确率。对于自然语言处理而言，信道噪声模型如下图，其中I表示输入，O 表示经过噪声信道后的输出，I'表示经过解码后最有可能的输入。

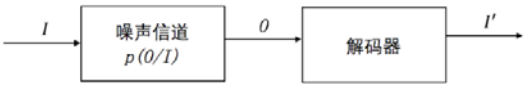


图 4.2 信道噪声模型框图

在自然语言处理中，很多问题都可以归结为给定输出 O（有可能包括错误信息）I，在所有可能的输入中找到最可能的那一个作为输入I'。

自然语言处理中的机器翻译，词性标注，语音识别等多个问题都可以使用信道噪声模型来解决，对于纠错问题也可以使用信道噪声模型来解决，相应的求解问题可以用公式表达：

$$\begin{aligned}\hat{w} &= \underset{w \in V}{\operatorname{argmax}} P(w \mid x) \\ &= \underset{w \in V}{\operatorname{argmax}} \frac{P(x \mid w) P(w)}{P(x)} \\ &= \underset{w \in V}{\operatorname{argmax}} P(x \mid w) P(w)\end{aligned}$$

其中 p(x|w) 是正确的词编辑成为错误词 x 的转移概率，包括删除（deletion）、增加（insertion）、替换（substitution）和颠倒（transposition）四种转移矩阵，这个转移矩阵的概率可以通过统计大量的正确词和错误词对来得到，转移矩阵的计算公式如下：

```
del[x,y]:    count(xy typed as x)
ins[x,y]:    count(x typed as xy)
sub[x,y]:    count(x typed as y)
trans[x,y]:  count(xy typed as yx)
```

$$P(x|w) = \begin{cases} \frac{\text{del}[w_{i-1}, w_i]}{\text{count}[w_{i-1} w_i]}, & \text{if deletion} \\ \frac{\text{ins}[w_{i-1}, x_i]}{\text{count}[w_{i-1}]}, & \text{if insertion} \\ \frac{\text{sub}[x_i, w_i]}{\text{count}[w_i]}, & \text{if substitution} \\ \frac{\text{trans}[w_i, w_{i+1}]}{\text{count}[w_i w_{i+1}]}, & \text{if transposition} \end{cases}$$

将转移矩阵计算公式代入公式 5 的噪声信道模型公式中，根据不同候选词和纠错词之间的变换关系选择转移矩阵类型，就能得到概率最大的候选词。

4.2.2 Real-word 纠错

有研究报告指出有 40%~45% 的错误属于 Real-word Error 问题。Real-word 问题中，每个词都是正确的，但是组合在一起成为短语或句子时意思却不对。因此纠错策略和 Non-word 有些不同。首先是候选词集合的生成，对于句子或短语中每个词生成候选集合，这个集合包括：1. 这个词本身；2. 所有和这个词编辑距离为 1 的词；3. 同音词。集合选定后，选择最佳候选对象或组合时，可以使用的方法有噪声信道模型及特殊分类器。

噪声信道模型和 Non-word 纠错类似，只是计算目标从某个候选词的最大概率变成不同位置候选词组合形成的句子 p(s) 的最大概率，这个问题可以使用 HMM（Hidden Markov model，隐马尔可夫模型）求解。

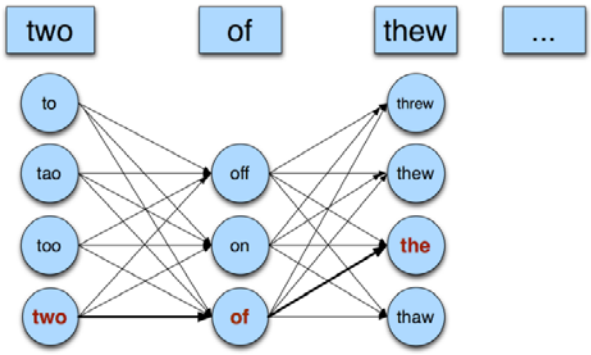


图 4.3 噪声信道模型纠错 Real-word Error 问题

上图中，每一数列是这个位置词的候选词集合，其中每个词的状态转移概率可以通过语言模型在语料库中统计求得。

基于分类方法纠错，分类器将会根据多个特征训练出一对 Real-word 之间的转移模型，常见的分类器包括 SVM（Support Vector Machine，支持向量机）或者是基于规则的分类器，特征可以选择每个 word 的 unigram，bigram 概率等。

4.3 中文纠错

中文纠错以英文纠错为基础但却有所不同。在中文中，一般情况下错误词和正确词的长度是相同的，只是指定位置上的某一个字有误，因此状态转移矩阵只有替换一种。其次是中文词语往往较短，即使编辑距离只有 1，就会有大量的候选词，存在较大的转义风险。中文使用拼音作为文字的发音，每个字都有固定的发音（多音字除外），而拼音输入法占据中文输入方式的主导地位，导致错误 query 中的别字同音但字形错误。因此中文纠错采取以拼音为基础，编辑距离等其他方式为辅的策略。

4.3.1 候选词集合的获取

对于错误的词的候选词集合，可以通过数据自动挖掘来生成。英文候选词集合一般通过编辑距离来获得，而中文候选词集合使用和错误词有相同的拼音的词组成。比如“嗒衣”这个错词的拼音是“dayi”，则可以通过事先挖掘好的拼音是“dayi”的词组成候选集，比如 <dayi: 大一，大衣，大意，大姨，搭衣，答疑 ..>。

4.3.2 候选词的选择

对于纠错候选词的选择就是一个对候选词进行排序，按照一定的排序规则，把排名最高的候选词作为最佳纠错结果返回。排序规则可以使用词频等多种特征，候选词按照这些特征规则进行排序，返回权重较高的词。

对于一个无上下文关系的词进行纠错，候选词的选择会比较困难，比如上面“嗒衣”这个错词的候选词有很多，无论按照哪一种方式进行排序，都存在较为严重的转义风险，这时可以使用编辑距离等其他方式辅助选择。

相对于单独一个词的 query，由多个词组成的 query 纠错相对更加精准，每个词存在上下文关系约束，整个 query 的意图更加明确。通过对 query 分词，查找每个词的候选词集合，然后使用和英文 Real-word 纠错类似的方式纠错。

除了搜索日志 query 和语料库的统计挖掘，搜索系统中的 session 分析和点击模型提供的数据也能够为 query 纠错服务。搜索 session 指的是用户在某一个时间段内的搜索行为，如果把搜索日志按照时间排序，对于某一个用户的搜索日志来说，可以看到用户的搜索行为是分段的，每段之间往往有较为明显的间隔，每一段我们称为一个搜索 session。一般来说，用户在一个 session 内的搜索行为都是为了解决一个问题，因此在此 session 内用户输入的 query 往往都是相关的。

点击模型中的一些统计数据可以判断一个搜索 query 质量的高低，质量高的 query 往往会给出较好的结果，用户点击的欲望更高。举例来说，“度假”（正确）“渡假”（错误）这两个 query，假设用户输入较多的是错误的“渡假”，系统给出结果会比较差。下图例子中“渡假”的搜索结果都没有命中标题，而标题往往是用户最为关注的信息，如果标题中不含有搜索 query 关键字，用户点击的欲望也会较低。

图 4.4 错误 query“渡假”的结果少，质量差

图 4.5 正确 query“度假”的结果多，质量好

在这种情况下，虽然“渡假”的搜索次数更多，但是点击模型给出 query 分数会比较低，而候选词“度假”的 query 得分就会高一些，可以辅助其他纠错方式完成纠错。

4.4 存在的问题

搜索系统许多功能的召回率和准确率是矛盾的，但是在 query 纠错问题中，准确率往往要求更高。拼音 query 到汉字 query 的纠错，往往会存在较大的转义风险，不同的类型的拼音转换方式（全拼，模糊全拼，简拼，混拼）有着不同程度的转义风险，召回越大则准确率降低，因此使用全拼较为稳妥。

5. 达观数据搜索系统 query 纠错技术介绍

达观数据（http://www.datagrand.com）在搜索引擎等大数据技术上有着深厚的积累，搜索引擎提供多种功能及服务，其中纠错模块是比较重要的功能之一。

5.1 纠错过程

对于搜索中的 query 纠错功能，纠错过程主要分为以下 3 个过程：

1）Query 纠错判断。对于常见错误，例如常见的拼写错误，使用事先挖掘好的错误 query 字典，当 query 在此字典中时纠错。如果用户输入的 query 查询无结果或结果较少于一定阈值时，尝试纠错，可以根据不同领域的策略和容忍度，配置最少结果数阈值。

2）不同策略独立纠错。达观数据使用多种纠错策略，主要使用拼音纠错和编辑距离纠错，并辅助模糊音形近字二次纠错等其他纠错策略。同音策略是用户输入的错误 query 和候选纠错 query 有相同的拼音。编辑距离策略就是错误 query 和候选 query 之间编辑距离小于一定阈值，并配合其他条件进行过滤。

3）候选词结果选择。因为每个策略比较独立，不同策略会给出不同的候选词，因此对于候选词的选取，每个策略有所不同。不同策略之间，不同策略内部需要使用不同的评估方式，来选择最优结果。

达观搜索系统的纠错模块包括上述多个策略，每个策略独立运行，针对不同的领域和业务情况，策略优先级和权重可配置，纠错松紧度可调节。

5.2 系统设计

达观数据 EC 系统主要分为三部分：数据模块，离线建库端及在线检索端。

图 5.1 EC 系统模块构成

5.2.1 数据模块

数据模块的主要作用是为后面的离线建库端和在线检索端提供数据。

数据模块对搜索 log 定期进行抽取和统计，对 query 进行归一化后给出 query 频次词典。对数据库信息整理给出自定义词典。通过爬虫系统爬取优质词条词典。

5.2.2 离线建库端

离线建库端使用数据模块准备好的各种词典生就纠错词典，包括拼音纠错词典，编辑距离纠错词典等。根据配置，对频次词典中对超出一定长度 query 上述操作不处理。

5.2.3 在线检索端

在线检索端负责 query 实时纠错，根据 5.1 节的三个步骤进行。如果第一次纠错 query 查询结果较差，则使用扩大召回的方式，比如二次纠错、片段纠错等扩大召回重新纠错，进行二次查询并返回质量较高的查询结果。

5.3 纠错效果评估

纠错效果的好坏从微观上来讲，可以查看搜索日志中无结果或结果少的纠错 query 以及点击模型中点击较少的纠错 query 等方式发现 bad case，在针对这些 bad case 出现的原因进行分类总结，后续改进算法。

在宏观上可以关注搜索效果评估系统中的 MAP 和 MRR 分数，使用 AB test，查看使用纠错模块后或纠错算法升级后的带来的效果提升。

6. 结语

一个完善的搜索引擎系统中，纠错功能是重要的一环，对提升用户体验及用户满意度有很大的帮助，亦能补救大量错误 query 所带来的流量损失。达观数据在搜索引擎服务上有着丰富的行业经验，能够为合作企业提供高质量的搜索服务，充分挖掘企业的数据价值。

【本文作者简介】

高翔，达观数据联合创始人，自然语言处理技术专家，达观数据前端项目组、文本语义理解组负责人，上海交通大学通信专业硕士，曾代表达观数据参加 2016 青年互联网创业大赛并赢得全国总冠军荣誉、第五届中国创新创业大赛优秀企业奖、中国电子 i+ 创新创效创意大赛总决赛一等奖。曾就职于腾讯文学，盛大文学，盛大创新院，负责文本阅读类产品、搜索引擎、文本挖掘及大数据调度系统的开发工作。在自然语言处理和机器学习等技术方向有着丰富的经验。

达观数据智能问答技术研究

江永青　达观数据架构师

在机器人围棋大胜李世石、柯洁之后，人工智能越来越火，智能问答也是人工智能中必不可少的一环。智能问答一般用于解决企业客服、智能资讯等应用场景，实现的方式多种多样，包括简单的规则实现，也可以基于检索实现，还可以通过 encoder-decoder 框架生成，本文通过几种常见的问答技术，概要介绍了达观数据智能问答相关原理研究。

1. 基于规则的智能问答

基于规则的智能问答通常是预先设置了一系列的问答规则，在用户输入一个问题时，去规则库里匹配，看是否满足某项规则，如果满足了就返回该规则对应的结果。如规则库里设置“* 你好 *”->“你好啊！”，那么用户在输入“你好”时，机器人会自动返回“你好啊！”。如果规则库非常庞大，达到了海量的级别库，则可对规则建立倒排索引，在用户新输入一个问题时，先去倒排索引中查找命中的规则集合，再通过这个集合中的规则进行匹配返回。

使用规则库的智能问答优点是简单方便，准确率也较高；缺点是规则库要经常维护扩展，而且覆盖的范围小，不能对新出现的问题进行回答。

2. 基于检索的智能问答

基于检索的智能问答很像一个搜索引擎，但又和搜索引擎不同，相比搜索引擎而言，智能问答更侧重于对用户意图和语义的理解。它基于历史的问答语料库构建索引，索引信息包括问题、答案、问题特征、答案特征等。用户问问题时，会将问题到索引库中匹配，首先进行关键字和语义的粗排检索，召回大量可能符合答案的问答对；然后通过语义和其他更丰富的算法进行精排计算，返回最好的一个或几个结果。

2.1 粗排策略

粗排策略跟一般的搜索引擎非常类似，主要基于的技术包括粗细粒度分词、词重要性计算、核心词识别、命名实体识别、语义归一等相关技术，主要是为了在粗排阶段尽可能地把相关问题进行召回。

1）词重要性计算：通过计算重要性，越能表示问题的词汇权重越高，在召回时

命中这些词汇的候选集越有可能被召回。如：“靠谱的英语培训机构有哪些？”，在这个问题中，“英语”、“培训”、“机构”是高权重的词，“靠谱”是较高权重的词，“哪些”是较低权重的词；因此越符合“英语培训机构”的答案越有可能被召回。

2) 核心词识别：核心词就是候选集中必须相关的词。如“北京住宿多少钱？”，核心词是“北京”、“住宿”，如果候选集中没有这两个相关的词，如“上海住宿多少钱？”，“北京吃饭多少钱”，都是不符合问题需求的。

3) 命名实体识别：通过命名实体识别，能协助识别出问题答案中的核心词，也可以对核心专有名词进行重要性加权，辅助搜索引擎提升召回效果。

4) 语义归一也是扩大召回的重要手段，同一个问题可能有很多种问法，不同的问法如果答案不同，或者召回的结果数目不同，就会很让人烦恼了，比如“刘德华生日是哪天？”、“刘德华出生在哪一天？”，如果不作语义归一的话，有可能某一个问题都不会召回结果。

2.2 精排策略

通过粗排，搜索引擎已经返回了一大批可能相关的结果，比如 500 个，如何从这 500 个问题中找到最符合问题的一个或者几个，非常考验算法精度。一般基于检索的问答系统都会通过语义或者深度学习的方法寻找最匹配的答案。

2.2.1 基于句子相似度的算法。

基于句子相似度的算法有很多种，效果比较好的有基于 word2vec 的句子相似度计算和基于 sentence2vec 的句子相似度计算。基于 word2vec 计算两个句子的相似度，就是以词向量的角度计算第一个句子转换到第一个句子的代价。

词向量有个有趣的特性，通过两个词向量的减法能够计算出两个词的差异，这些差异性可以应用到语义表达中。如：vec(Berlin) – vec(Germany) = vec(Paris) – vec(France)；通过这个特性能够用用来计算句子的相似度。假设两个词 xi, xj 之间的距离为$c(i,j) = \|x_i - x_j\|_2$，这可以认为是 xi 转换到 xj 的代价。可以将句子用词袋模型$d \in R^n$表示，模型中某个词 i 的权重为$d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}$，其中 ci 是词 i 在该句子中出现的次数。设置 $T \in R^{n*n}$为一个转换矩阵，Tij 表示句子 d 中词 i 有多少权重转换成句子 d’ 中的词 j，如果要将句子 d 完全转换成句子 d’，所花费的代价计算如下：

$$\min_{\mathbf{T}\geq 0} \sum_{i,j=1}^{n} \mathbf{T}_{ij} c(i,j)$$
$$\text{subject to: } \sum_{j=1}^{n} \mathbf{T}_{ij} = d_i \quad \forall i \in \{1,\ldots,n\} \qquad (1)$$
$$\sum_{i=1}^{n} \mathbf{T}_{ij} = d'_j \quad \forall j \in \{1,\ldots,n\}.$$

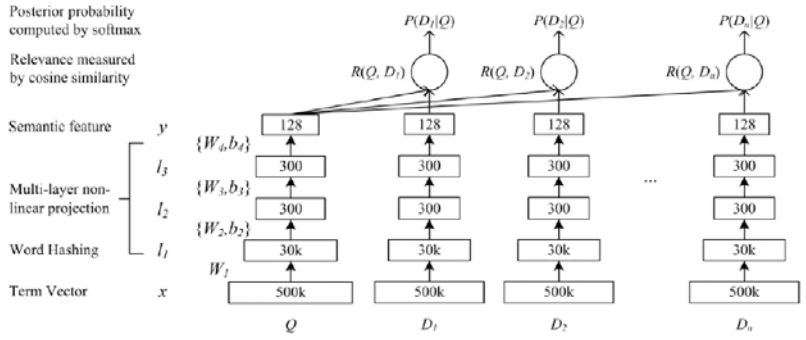
如果用 Xd 表示句子中的词向量通过权重 di 进行加权平均的句向量，可以推导出，句子转换代价的下限是两个句向量的欧式距离。

$$\sum_{i,j=1}^{n} \mathbf{T}_{ij} c(i,j) = \sum_{i,j=1}^{n} \mathbf{T}_{ij} \|\mathbf{x}_i - \mathbf{x}'_j\|_2$$
$$= \sum_{i,j=1}^{n} \|\mathbf{T}_{ij}(\mathbf{x}_i - \mathbf{x}'_j)\|_2 \geq \Big\| \sum_{i,j=1}^{n} \mathbf{T}_{ij}(\mathbf{x}_i - \mathbf{x}'_j) \Big\|_2$$
$$= \Big\| \sum_{i=1}^{n} \Big(\sum_{j=1}^{n} \mathbf{T}_{ij} \Big) \mathbf{x}_i - \sum_{j=1}^{n} \Big(\sum_{i=1}^{n} \mathbf{T}_{ij} \Big) \mathbf{x}'_j \Big\|_2$$
$$= \Big\| \sum_{i=1}^{n} d_i \mathbf{x}_i - \sum_{j=1}^{n} d'_j \mathbf{x}'_j \Big\|_2 = \|\mathbf{X}\mathbf{d} - \mathbf{X}\mathbf{d}'\|_2.$$

一般这个下限表示两个短句子相似的程度已经足够了，如果需要通过完全最优化的方法计算 $\min_{T\geq 0} \sum_{i,j=1}^{n} T_{ij} c(i,j)$ 的值，可以通过 EMD solver 算法计算。

2.2.2 基于深度学习计算问答匹配程度

基于句向量的距离计算句子相似度，可以 cover 大部分的 case，但在句子表面相似，但含义完全不同的情况下就会出现一些问题，比如“我喜欢冰淇淋”和“我不喜欢冰淇淋”，分词为“我”，“不”，“喜欢”，“冰淇淋”，两个句子的相似度是很高的，仅一字“不”字不同，导致两个句子意思完全相反。要处理这种情况，需要使用深度模型抓住句子的局部特征进行语义识别。

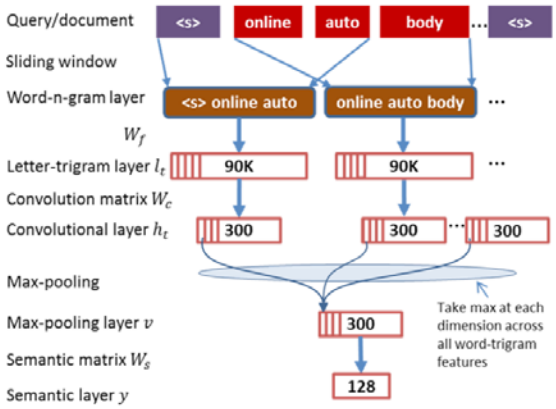


如图所示，Q 是用户的问题，D 是返回的各个答案。对于某一个问答句子，首先将它映射到 500k 大小的 BOW 向量 Term Vector 里。因为 Term Vector 是

稀疏矩阵，可以使用 Word Hashing 或者其他 Embedding 的方法将其映射到 30k 大小的词向量空间里。接下来的 l1, l2, l3 层就是传统的 MLP 网络，通过神经网络得到 query 和 document 的语义向量。计算出（D，Q）的 cosine similarity 后，用 softmax 做归一化得到的概率值是整个模型的最终输出，该值作为监督信号进行有监督训练。模型通过挖掘搜索点击日志构造的 query 和对应的正负 document 样本（点击 / 不点击），输入 DSSM 进行训练。

2.2.3 基于卷积神经网络计算问答匹配程度

句子中的每个词，单独来看有单独的某个意思，结合上下文时可能意思不同；比如“Microsoft office”和“I sat in the office”，这两句话里的 office 意思就完全不一样。通过基于卷积神经网络的隐语义模型，我们能够捕捉到这类上下文信息。



如图所示，先通过滑窗构造出 query 或 document 中的一系列 n-gram terms，比如图中是 Word-n-gram layer 中的 trigram；然后通过 word-hashing 或者 embedding 将 trigram terms 表示为 90k 的向量；通过卷积向量 Convolution matrix Wc 对每个 letter-trigram 向量作卷积，可以得到 300 维的卷积层 Convolutional layer；最后通过 max-pooling 取每个维度在 Convolutional layer 中的最大值，作为文本的隐语义向量。模型也是通过挖掘搜索日志进行有监督训练。

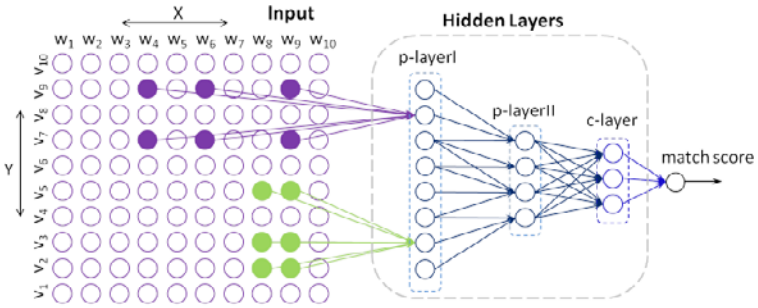
通过卷积神经网络，能得到句子中最重要的信息。如下面一些句子，高亮的部分是卷积神经识别的核心词，它们是在 300 维的 Max-pooling 层向量里的 5 个最大神经元激活值，回溯找到原始句子中的词组。

microsoft office excel could allow remote code execution

welcome to the apartment office

3. 基于主题模型计算问答匹配程度

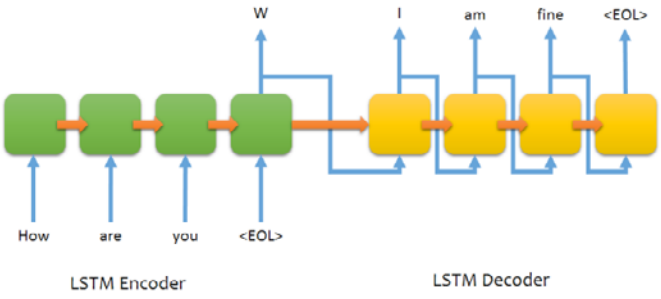
短文本一般词语比较稀疏，如果直接通过共现词进行匹配，效果可能会不理想。华为诺亚方舟实验室针对短文本匹配问题，提出一个 DeepMatch 的神经网络语义匹配模型，通过（Q, A）语料训练 LDA 主题模型，得到其 topic words，这些主题词用来检测两个文本是否有语义相关。该模型还通过训练不同“分辨率”的主题模型，得到不同抽象层级的语义匹配（“分辨率”即指定 topic 个数，高分辨率模型的 topic words 通常更加具体，低分辨率的 topic words 通常更加抽象）。在高分辨率层级无共现关系的文本，可能在低分辨率存在更抽象的语义关联。DeepMatch 模型借助主题模型反映词的共现关系，可以避免短文本词稀疏带来的问题，并且能得到不同的抽象层级的语义相关性。



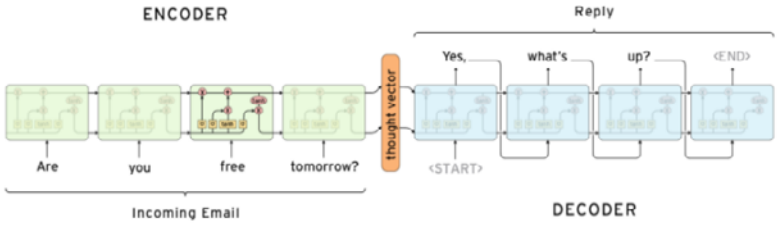
如图所示，绿色和紫色块分别表示在同一个分辨率下不同的主题在 X 和 Y 文本中命中的主题词块，与上一层分辨率（p-layerII）的主题的关联通过是否与上一层的主题词块有重叠得到。如此通过多层的主题，能够构建出神经网络，并使用有监督的方式对相关权重进行训练。

4. 基于产生式的智能问答

基于产生式的智能问答系统，主要是通过 seq2seq 的方式，通过一个翻译模型的方式进行智能回答，其中问题是翻译模型的原语言，答案是翻译模型的目标语言。Seq2seq 模型包含两个 RNN，一个是 Encoder，一个是 Decoder。Encoder 将一个句子作为输入序列，每一个时间片处理一个字符。Decoder 通过 Encoder 生成的上下文向量，使用时间序列生成翻译（回答）内容。



在 Encoder 中，每一个隐藏的状态影响到下一个隐藏状态，并且最后一个隐藏状态可以被认为是序列的总结信息。最后这个状态代表了序列的意图，也就是序列的上下文。通过上下文信息，Decoder 会生成另一个结果序列，每一个时间片段，根据上下文和之前生成的字符，Decoder 都会生成一个翻译字符。



这个模型有一些不足：首先是这个模型不能处理变长的字符序列，而一般的翻译模型和问答模型中的序列长度都是不定的。另外一个是仅通过一个 context 变量，并不足以完全表示输入序列的信息。在序列变得很长之后，大量的信息会被丢弃，因此需要多个 context 变量及注意力机制进行处理。

4.1 Padding

通过 Padding 方式，可以将问答字符串固定为定长的序列，比如使用如下几个序列进行 Padding：

EOS : 序列的结束

PAD : Padding 字符

GO : 开始 Decode 的字符

UNK : 不存在字典中的字符

对于问答对：

Q : 你过得怎样?

A : 过得很好。

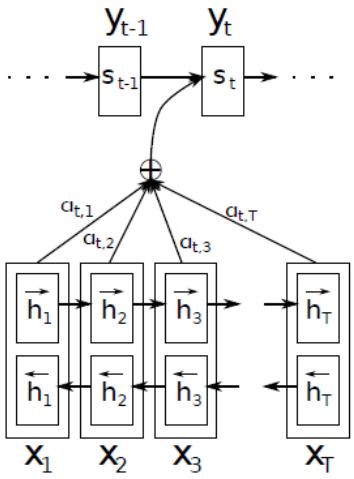
通过 padding 将生成固定的如下字符串：

Q : [PAD, PAD, PAD, PAD, “？”，“样”，“怎”，“得”，“过”，“你”]

A : [GO, “过”，“得”，“很”，“好”，“。”, PAD, EOS, PAD, PAD]

4.2 注意力机制

Seq2Seq 的一个限制是输入序列的所有信息只能编码到一个定长的数组 context 里，如果输入序列变长的话，我们很容易会丢失信息，因此 Seq2Seq 模型对长输入序列效果很差。通过引入一个注意力机制，可以让 Decoder 在翻译（回答）时按权重选择性地提取输入的各个隐层变量进行计算，这样就能在产生输出时考虑到输入的关键信息了。



如图中，每个输入的状态 yi 除了与前一个状态 yi−1，输入的隐藏状态 si 有关之外，还与注意力向量 ci 有关。

$$p(y_i|y_1,\dots,y_{i-1},x) = g(y_{i-1},s_i,c_i)$$

$$s_i = f(s_{i-1},y_{i-1},c_i)$$

这里的注意力向量不再是单独的一个 c，而是对于每一个输入 yi 都有对应的 ci，它的计算方式如下：

$$c_i = \sum_{j=1}^{T_x} a_{ij}h_j$$

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x}\exp(e_{ik})}$$

$$e_{ij} = a(s_{i-1},h_j)$$

eij 是一个对准模型，它计算了位置为 i 的输入和位置为 j 的输出之间关系的比重。它通过输出的隐藏状态 si – 1（用来计算 si 及输出状态 yi）和输入序列的第 j 个隐藏变量 hj 计算而得。

通过将问答对以翻译模型进行训练（问题作为原语言，答案作为目标语言），在新来一个问题时，可以在没有任何历史相似问题的情况下得到较接近的答案。然而因为目前的翻译模型还不够成熟，经常翻译出莫名奇妙的答案，让人摸不着头脑，如下所示：

你好 -> 你好

很高兴认识你 -> 我也是

你是谁 -> 我是说，我们是朋友

我们是朋友吗 -> 不是

我们为什么不是朋友 -> 我们不知道，我们都不知道你们在哪里

达观数据很厉害 -> 我们知道

通过产生式的翻译模型，即使基于大量的语料训练，很有可能效果还是不够理想，因此经常作为基于规则和检索的问答系统的辅助方式。

5. 小结

本文介绍了智能问答技术的相关原理，讲述了实现智能问答的一般方法。借助于深度学习技术，智能问答效果比传统机器学习提升 20% 左右。达观数据拥有多年的自然语言处理技术积累，并且紧跟行业潮流，在文本语义、信息检索、智能问答方面不断深耕，助力各个企业享受文本挖掘技术的成果。

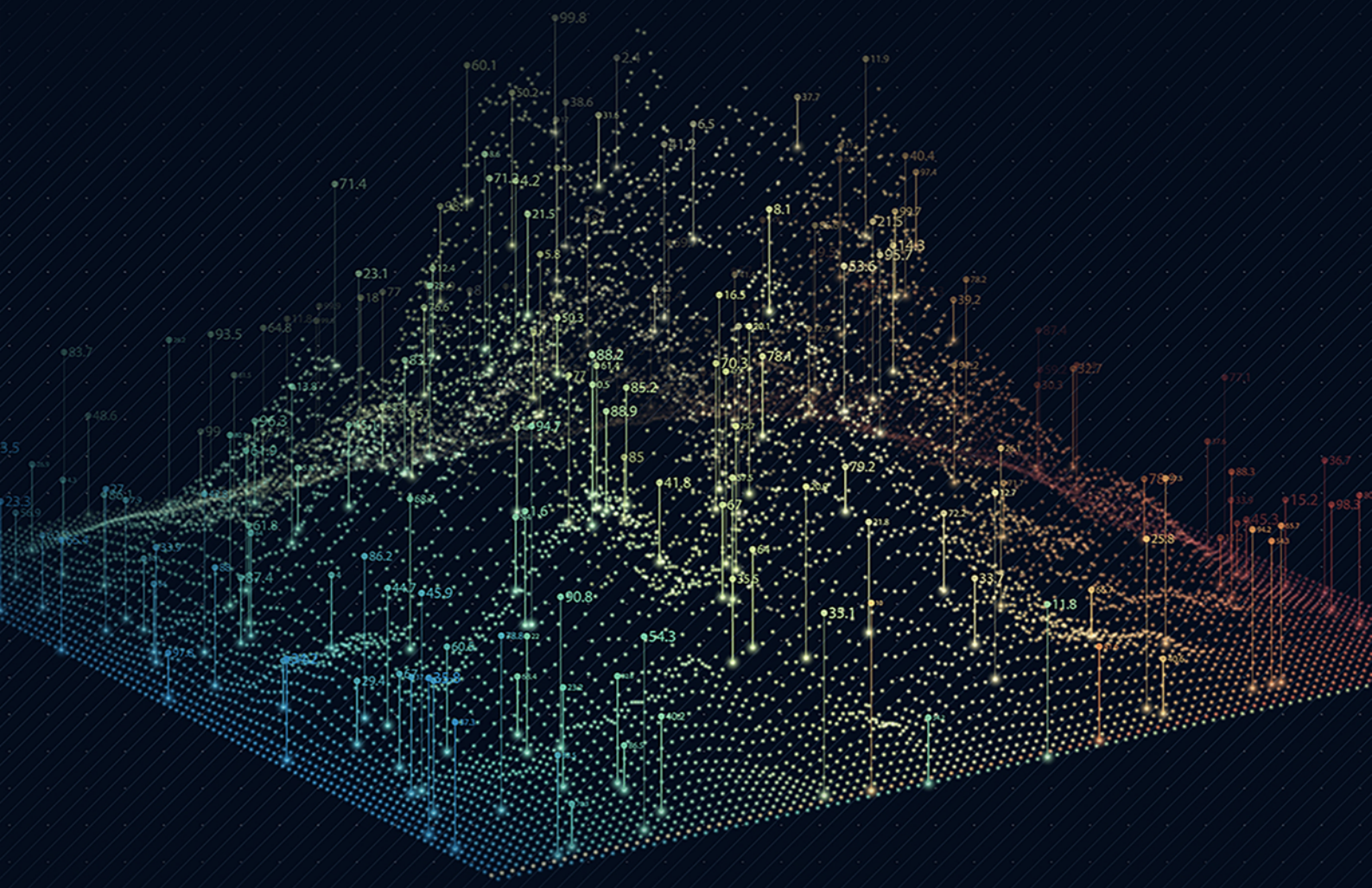
【本文作者简介】

江永青，达观数据联合创始人，搜索引擎组负责人，包括搜索系统的架构设计和开发、搜索效果优化以及日常管理等。浙江大学软件工程专业硕士，曾在盛大创新院搜索组负责索引和检索模块，盛大文学部分负责数据平台架构和智能审核系统模块，对搜索系统，大数据平台，机器学习有较丰富的实践经验。

智能推荐引擎实践篇

智能推荐引擎，根据不同用户的喜好挖掘生成用户画像，为每位用户提供“千人千面”的个性化推荐内容，帮助传媒、电商等行业有效提升点击率、转化率及用户粘性，极大地增加客户的经营效益

【本篇关键词】冷启动；用户画像；推荐系统优化；融合推荐算法



推荐系统中的冷启动和探索利用问题探讨

文辉　达观数据科学家

1. 前言

互联网技术和大数据技术的迅猛发展正在时刻改变我们的生活，视频网站、资讯 app、电商网站对于推荐系统而言，每天都有大量的活跃用户在不断的产生海量的用户行为，同时，每天又都产生大量的新增 PGC 或者 UGC 内容（如小说、资讯文章、短视频等）。

从推荐系统的角度来看，系统每时每刻都面临大量的新旧用户、新旧物品和大量的用户行为数据。对于用户，我们需要对要用户进行建模，去刻画用户的肖像和兴趣。然而我们常常面对的情况是用户的行为是稀疏的，而且可能存在比例不一的新用户，如何给新用户推荐，是推荐系统中的一个著名问题，即冷启动问题。

给新用户展示哪些 item 决定了用户的第一感和体验，同时在推荐过程中，我们需要考虑给新 item 展示的机会，也能也需要给一个喜欢科幻电影的 user 推荐一些非科幻类型的电影，而这就是推荐系统中另外一个问题，即探索和利用的问题。

2. 冷启动和 EE 问题

推荐系统需要根据历史的用户行为和兴趣偏好预测用户未来的行为和兴趣，因此历史用户行为某种程度上成为推荐的重要先决条件。实际过程中，我们面对大量的新用户，这些用户我们并不知道他们的 profile，对于这些用户，常用的冷启动的算法包括根据已有的个人静态信息（年龄、性别、地理位置、移动设备型号等）为用户进行推荐。然而实际情况下，我们很难在系统中直接获取这些用户信息。给新用户推荐的 item，由于成本较高（用户不感兴趣就再也不来了），我们需要保证 item 要足够热门，要保证足够的多样性，同时尽量保证可区分。

与用户的冷启动相对应的，则是 item 的冷启动，当一个新物品加入站内，如何快速的展现的用户。对于 CF 算法来说，无论是基于领域还是基于模型，如果想要这个新物品被推荐出来，显然我们需要用户对这个物品的行为。特别是在某些场景下，推荐列表是给用户展示的唯一列表，那么显而易见，只能在

推荐列表中尝试给用户推荐新物品。一个最简单的做法就是在推荐列表中随机给用户展示新物品，但是这样显然不太个性化，所以一个相对较好的做法是将新物品推荐给曾经喜欢过与新物品相似的物品的用户。由于新 item 没有用户行为，因此物品相似度只能从 item 自身出发，包括根据 item 的内容信息挖掘出 item 的向量表示，通过向量相似度来刻画物品相似度，还可以利用 topic model 挖掘出 item 的主题分布等等。

推荐系统需要对用户兴趣的不断探索和细化，同时也需要尽可能的扩大展示物品的多样性和宽度。比如极端情况下给用户展示物品的场景只有推荐列表，同时我们需要尽可能的优化的 ctr，那么面对冷启动用户我们该如何选择物品，如何快速的探测出用户的兴趣，比如对于一个热爱足球的足球迷我们该如何选择给他推荐的物品呢?

比较简单的方式我们可以根据 ctr 排序，给冷启动用户推荐最热门点击率最高的物品，给足球迷推荐点击率最高的足球相关物品。显然这样做会保证我们推荐结果的 ctr 会比较高，但是这样做会减少我们推荐结果的覆盖率，降低推荐结果的多样性，以致于推荐物品候选集也会收敛，出现反复推荐。比如对于资讯来说，用户看到的资讯都是点击率较高的搞笑类，但是显然搞笑并不能刻画的兴趣。而这也就是我们得不断探索用户兴趣，扩大推荐结果多样性的原因。简单来看这其实是一个选择问题，即探索（exploration）和利用（exploitation）问题。接下来本文将详述 EE 问题和某些已有算法。

3. 多臂老虎机模型和 UCB 算法

当你走进一家赌场，面对 20 个一模一样的老虎机，你并不知道它们吐钱的概率，假设你的成本是 1000 元，每摇一次的成本是 2 元，那么你在总共 500 次摇臂的机会下，该如何最大化你的收益呢？ 这就是多臂老虎机问题 (Multi-armed bandit problem, K-armed bandit problem, MAB)。

一个简单的做法就是每台老虎机我们都摇 10 次，余下的 300 次都选择成功率最高的那台。但是显然我们耗费了 200 次机会来探索，而且我们仍然无法保证实验成功率最高的那台老虎机就是真实成功率最高的。大家也可以猜到，如果我们有足够多的探索机会，那么我们几乎可以选择出成功率最高的老虎机。很遗憾，我们没有足够的探索机会，对应到我们的推荐问题中就是任何的用户展示 pv 都是珍贵的，况且实际情况的“老虎机”远远不止 20 台，而且还存在不断新加入的情况，这就导致获取每个 item 收益率的成本太大。

我们使用累计遗憾（collect regret）来衡量策略的优劣：

$$\rho = Tu^{*} - \sum_{t=1}^{T} \hat{r}_t$$

t 表示当前轮数，u^*表示平均最大收益，$\hat{r}_t$ 表示第 t 轮的实际收益。累计遗憾公式表明了实际累计收益和理想最佳收益的差值。为了简单起见，我们假设每台机器的收益为伯努利收益，即收益要么是0，要么是1。对应到推荐系统中，老虎机即对应我们的物品（item），每次摇臂即认为是该物品的一次展示，我们可以用是否点击来表示收益，点击 (win) 的收益就是 1，没有点击 (lose) 的收益就是 0。

解决 bandit 问题的算法众多，一般分为基于 semi-uniform 的策略、probability matching 策略、pricing 策略等。这里只列举若干个策略，具体大家可以参考（https://en.wikipedia.org/wiki/Multi-armed_bandit）。

Epsilon-greedy 策略：每次试验都以 $1-\epsilon$ 的概率选择前面试验中平均收益最佳的 item，以 ϵ 的概率等概率随机选择其他 item，该策略简单，而且可以通过 ϵ 控制探索和利用的比率。

Epsilon-first 策略：该策略探索和利用交叉选择，总试验次数为 ϵ，探索次数为ϵ *N，探索阶段也是等概率随机选择所有 item，利用阶段也是选择平均收益最好的机器。

Epsilon-decreasing 策略：与 Epsilon-greedy 策略近似，不同地方在于 ϵ 随着试验的进行会不断减少。

3.1 UCB（Upper Confidence Bound）算法

在推荐系统中，通常量化一个物品的收益率（或者说点击率）是使用点击数 / 展示数。例如点击为 10，展示数为 8，则估计的点击率为 80%，在展示数达到 10000 后，其表现 ctr 是否还能达到 80% 呢？显然是不可能的。而这就是统计学中的置信度问题，计算点击率的置信区间的方法也有很多，比如威尔逊置信空间（https://en.wikipedia.org/wiki/Binomial_proportion_confidence_interval#Wilson_score_interval）。

UCB 算法步骤包括：首先对所有 item 的尝试一下，然后每次选择以下值最大的那个 item: $\bar{x}_j(t)+\sqrt{\frac{2lnt}{T_{j,t}}}$

其中 $\bar{x}_j(t)$ 是物品x_j到目前的收益均值，$\sqrt{\frac{2lnt}{T_{j,t}}}$本质上是均值的标准差。t 是目前的试验次数，$T_{j,t}$ 是这个 item 被试的次数。

这个公式表明随着每个物品试验次数的增加，其置信区间就越窄，收益概率就越能确定。如果收益均值越大，则被选中的机会就越大（exploit），如果收益均值越小，其被选中的概率也就越少，同时那些被选次数较少的 item 也会得到试验机会，起到了 explore 的作用。

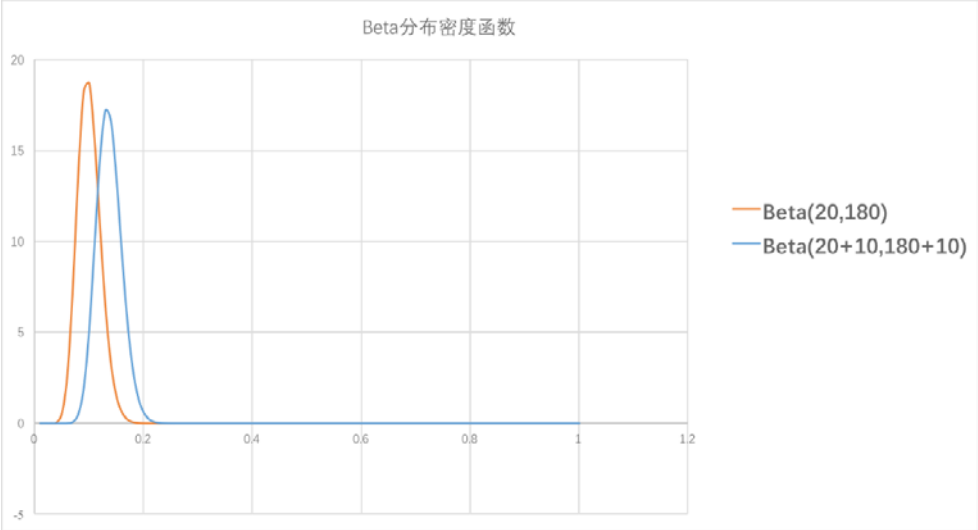
Probability-matching 策略表明一台机器的选择次数应当与它是最佳收益 item 的概率相同。其中 Thompson 采样就是一种 Probability-matching 策略算法，该算法也是一个的在线学习算法，即通过不断的观察数据来更新模型参数。

3.2 Thompson 采样

Thompson 采样假设每个 item 的收益率为 p，那么如何来估计一个 item 的收益概率 p 呢？直接用试验结果的收益概率 p 是否合理呢？ 比如一个 item 给用户展示了 20 次，用户点击了 16 次，那么我们可以认为这个 item 的收益率是 80% 吗？而这显然是不合理的，因为我们首先需要考虑的就是这个收益率的置信度，20 次试验得出的结果置信度小于试验了 10000 次试验的置信度，其次可能我们的经验表明所有 item 的平均收益率只有 10%。

Thompson 采样使用 Beta 分布来描述收益率的分布（分布的分布：https://en.wikipedia.org/wiki/Beta_distribution），我们通过不断的试验来估计一个置信度较高的基于概率 p 的概率分布。假设概率 p 的概率分布符合 Beta(wins, lose)，beta 分布有两个参数 wins 和 lose，每个 item 都维护了 beta 分布的参数，每次试验都选择一个 item，有　点击则 wins 增加 1，否则 lose 增加 1。每次选择 item 的方式则是：用每个 item 的 beta 分布产生一个随机数，选择所有 item 产生的随机数中的最大的那个 item。Beta 分布概率密度函数如下：

$$\mathrm{Beta}(x;\alpha,\beta) = \frac{x^{\alpha-1} * (1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1} * (1-u)^{\beta-1} du} = \frac{x^{\alpha-1} * (1-x)^{\beta-1}}{B(\alpha,\beta)}$$



举例来说，推荐系统需要试探新用户的兴趣，假设我们用内容类别来表示每个用户的兴趣，通过几次展示和反馈来获取用户的兴趣。针对一个新用户，使用 Thompson 算法为每一个类别采样一个随机数，排序后，输出采样值 top N 的推荐 item。获取用户的反馈，比如点击。没有反馈则更新对应类别的 lose 值，点击了则更新对应类别的 wins 值。

4. LinUCB 算法

回到推荐列表的场景，推荐系统为用户推荐物品。user 和 item 都可以用一系列特征表示。用户特征包括用户的统计历史行为、人口学属性信息；物品特征包括描述信息、类别信息等等。在这种场景下，探索和利用也必须是个体用户级别上实施，因为不同用户看到相同的物品的反馈差异较大。

LinUCB 算法是一种基于上下文特征（用户特征、物品特征）的 UCB 算法，基于特征进行探索和利用。该算法结合上下文特征，选择给用户的推荐物品，同时利用用户反馈及时修正选择策略，以达到最大化收益（提升点击率）的目标。

使用互斥线性模型的 LinUCB

LinUCB 算法假设推荐 item 的每次展现收益（是否点击）是和上下文特征成线性关系的，即：

$$E[r_{t,a}|x_{t,a}] = x_{t,a}^T \theta_a^*$$

其中 $x_{t,a}$ 表示用户特征和物品特征的合集，$r_{t,a}$表示第 t 次尝试的收益，a 表示 item，θ_a^*表示物品 a 的位置系数向量。可以看出各个 item 的模型参数是相互独立的（互斥）。

设 D_a（d*m）表示为 m 个训练上下文，c_a表示每个上下文的实际收益，对训练数据(D_a, c_a)使用岭回归训练出的物品 a 的参数为：

$$\hat{\theta}_a = (D_a^T D_a + \mathrm{I}_d)^{-1} * D_a^T * c_a$$

其中I_d表示 d*d 的单位矩阵。其中在置信度$1-\delta$下，模型收益与期望收益满足：

$$\left|x_{t,a}^T \hat{\theta}_a - E[r_{t,a}|x_{t,a}]\right| \leq \alpha\sqrt{x_{t,a}^T (D_a^T D_a + \mathrm{I}_d)^{-1} x_{t,a}}$$

其中 $\delta > 0$，$\alpha = 1 + \sqrt{\ln\left(\frac{2}{\delta}\right)/2}$ 。

上述等式给出了物品 a 期望收益的一个 UCB，因此也就引申出了 UCB 的选择策略，对于第 t 次试验，选择以下式中最大值的物品，

$$a_t \stackrel{\text{def}}{=} arg\max_{a \in A}(x_{t,a}^T \hat{\theta}_a + \alpha\sqrt{x_{t,a}^T (A_a)^{-1} x_{t,a}})$$

其中$A_a \stackrel{\text{def}}{=} D_a^T D_a + \mathrm{I}_d$。上述模型中预期收益 $x_{t,a}^T \theta_a^*$ 的方差为 $x_{t,a}^T (A_a)^{-1} x_{t,a}$

即 $\sqrt{x_{t,a}^T (A_a)^{-1} x_{t,a}}$ 为标准差。

Algorithm 1 LinUCB with disjoint linear models.

```
0:  Inputs: α ∈ ℝ+
1:  for t = 1, 2, 3, ..., T do
2:      Observe features of all arms a ∈ At: x_{t,a} ∈ ℝ^d
3:      for all a ∈ At do
4:          if a is new then
5:              A_a ← I_d (d-dimensional identity matrix)
6:              b_a ← 0_{d×1} (d-dimensional zero vector)
7:          end if
8:          θ̂_a ← A_a^{-1} b_a
9:          p_{t,a} ← θ̂_a^⊤ x_{t,a} + α sqrt(x_{t,a}^⊤ A_a^{-1} x_{t,a})
10:     end for
11:     Choose arm a_t = arg max_{a∈At} p_{t,a} with ties broken arbi-
        trarily, and observe a real-valued payoff r_t
12:     A_{a_t} ← A_{a_t} + x_{t,a_t} x_{t,a_t}^⊤
13:     b_{a_t} ← b_{a_t} + r_t x_{t,a_t}
14: end for
```

以上为互斥线性模型 LinUCB 的基本算法流程，其中结合上述内容，第一行α参数控制了 explore 的程度，即α越大，置信区间上限也就越大，也就加大了 explore 的程度；4-7 行对于新物品，使用单位阵和 0l 向量进行参数初始化；

8-9 行计算 item a 的置信区间上限；11 行选择最优 item；12-13 行更新选择 item 的模型参数。

思想上 LinUCB 算法类似于对召回结果重排序的方法，也是考虑用户和 item 的特征，来计算出收益最大的 item，不同的是 LinUCB 借鉴了 UCB 的置信区间的方法来平衡 exploit 和 explore 问题，同时从 LinUCB 算法是一个在线的学习算法，与一般离线算法需要离线训练不同，LinUCB 随着每次展示和反馈会不断优化我们的模型参数和收益。

关于 LinUCB 算法的介绍请参考论文 http://rob.schapire.net/papers/www10.pdf 。

5. CLUB 算法

CLUB(online clustering bandits) 算法假设将全部用户划分成若干个用户群，每个用户群对相同推荐内容的反馈是一致的，同时自适应的调整用户群。与 liner bandit 一样，CLUB 算法也是根据特征计算收益，不同的是 CLUB 算法中相同群体用户共享相同的参数向量，即第 i 个用户对 item a 的收益为：

$$a_i(x) = u_{j(i)}^T x + \epsilon_{j(i)}(x)$$

其中表示第 i 个 user，$j(i)$表示第 i 个 user 所属的用户群编号，u表示每小个用户群的参数向量，x 表示上文下特征，ϵ表示噪声项。

该算法在时刻 t，对于用户 i，维护一个向量u_i的估计值$w_{i,t}$。与 liner bandit 算法相似，$w_{i,t}$根据收益反馈不断更新。与 LinUCB 算法类似的，$w_{i,t-1}$可以根据协方差矩阵$M_{i,t-1}$（d*d 维，初始化为单位阵）的逆和向量$b_{i,t-1}$（d 维向量，初始化为 0 向量）计算得出。除此之外，算法需要维护一个无向图 $G_t = (V, E_t)$，每个节点表示一个 user。算法首先从完全图开始，根据$w_{i,t}$的演化逐步移除节点之间的边。定义第 t 时刻的用户群个数为m_t，$\hat{v}_{1,t}, \hat{v}_{2,t} \cdots\cdots \hat{v}_{1,mt}$ 表示时刻 t 的用户划分群。显然初始状态下：m_1=1，$\hat{v}_{1,1} = V$（全部用户）。

在每个时刻 t（1，2，……），用户 i_t ，相关上下文特征向量集合为 $(x_{t,1}, x_{t,2}, \ldots\ldots x_{t,ct})$ ，用户 i_t所属的群为$\hat{v}_{\hat{j}t,t}$ 。CLUB 算法根据收益选择 item 的式如下：

$$k_t = \underset{k=1,\ldots ct}{\operatorname{argmax}} \left(\overline{w}_{\hat{j}t,t-1}^T * x_{t,k} + CB_{\hat{j}t,t-1} * x_{t,k}\right)$$

其中$\overline{w}_{\hat{j}t,t-1}$是通过同用户群内各个节点通过最小方差逼近拟合计算得出的聚合权重向量，CB 为$\overline{w}_{\hat{j}t,t-1}$向量的置信区间上限。

CLUB 算法观察到 item 的收益a_t后，更新协方差矩阵$M_{i_t,t-1} \to M_{i_t,t}$，更新$b_{i_t,t-1} \to b_{i_t,t}$；虽然不会更新其他节点的 M 和 b，但是会隐式地影响下一轮的聚合权重向量$\bar{w}_{\hat{j}_t,t+1}$；接下来判断节点i_t与相邻节点的参数向量（$\bar{w}_{\hat{j}_t,t-1}$）距离，如果足够大，则将该边移除。

CLUB 算法的完整流程如下：

Input: Exploration parameter $\alpha > 0$; edge deletion parameter $\alpha_2 > 0$

Init:

- $\boldsymbol{b}_{i,0} = \mathbf{0} \in \mathbb{R}^d$ and $M_{i,0} = I \in \mathbb{R}^{d\times d}$, $i = 1, \ldots n$;
- Clusters $\hat{V}_{1,1} = V$, number of clusters $m_1 = 1$;
- Graph $G_1 = (V, E_1)$, G_1 is connected over V.

for $t = 1, 2, \ldots, T$ **do**

Set $\boldsymbol{w}_{i,t-1} = M_{i,t-1}^{-1}\boldsymbol{b}_{i,t-1}, \quad i = 1, \ldots, n$;

Receive $i_t \in V$, and get context $C_{i_t} = \{\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,c_t}\}$;

Determine $\hat{j}_t \in \{1, \ldots, m_t\}$ such that $i_t \in \hat{V}_{\hat{j}_t,t}$, and set

$$\bar{M}_{\hat{j}_t,t-1} = I + \sum_{i\in\hat{V}_{\hat{j}_t,t}} (M_{i,t-1} - I),$$

$$\bar{\boldsymbol{b}}_{\hat{j}_t,t-1} = \sum_{i\in\hat{V}_{\hat{j}_t,t}} \boldsymbol{b}_{i,t-1},$$

$$\bar{\boldsymbol{w}}_{\hat{j}_t,t-1} = \bar{M}_{\hat{j}_t,t-1}^{-1}\bar{\boldsymbol{b}}_{\hat{j}_t,t-1};$$

$$\text{Set } k_t = \operatorname*{argmax}_{k=1,\ldots,c_t} \left(\bar{\boldsymbol{w}}_{\hat{j}_t,t-1}^{\top}\boldsymbol{x}_{t,k} + \mathrm{CB}_{\hat{j}_t,t-1}(\boldsymbol{x}_{t,k})\right),$$

$$\mathrm{CB}_{j,t-1}(\boldsymbol{x}) = \alpha\sqrt{\boldsymbol{x}^{\top}\bar{M}_{j,t-1}^{-1}\boldsymbol{x}\,\log(t+1)},$$

$$\bar{M}_{j,t-1} = I + \sum_{i\in\hat{V}_{j,t}} (M_{i,t-1} - I), \quad j = 1, \ldots, m_t.$$

Observe payoff $a_t \in [-1, 1]$;

Update weights:

- $M_{i_t,t} = M_{i_t,t-1} + \bar{\boldsymbol{x}}_t\bar{\boldsymbol{x}}_t^{\top}$,
- $\boldsymbol{b}_{i_t,t} = \boldsymbol{b}_{i_t,t-1} + a_t\bar{\boldsymbol{x}}_t$,
- Set $M_{i,t} = M_{i,t-1}$, $\boldsymbol{b}_{i,t} = \boldsymbol{b}_{i,t-1}$ for all $i \neq i_t$;

Update clusters:

- Delete from E_t all (i_t, ℓ) such that

$$||\boldsymbol{w}_{i_t,t-1} - \boldsymbol{w}_{\ell,t-1}|| > \widetilde{\mathrm{CB}}_{i_t,t-1} + \widetilde{\mathrm{CB}}_{\ell,t-1},$$

$$\widetilde{\mathrm{CB}}_{i,t-1} = \alpha_2\sqrt{\frac{1+\log(1+T_{i,t-1})}{1+T_{i,t-1}}},$$

$$T_{i,t-1} = |\{s \le t-1 : i_s = i\}|, \qquad i \in V;$$

- Let E_{t+1} be the resulting set of edges, set $G_{t+1} = (V, E_{t+1})$, and compute associated clusters $\hat{V}_{1,t+1}, \hat{V}_{2,t+1}, \ldots, \hat{V}_{m_{t+1},t+1}$.

end for

其中α控制探索的程度，α_2是用户关系是否删除的控制参数。上述算法流程

包含了删除关系边的条件，其中$T_{i,t-1}$是 t 时刻前历史上 i 用户被选中的次数。

CLUB 算法首先提出了基于协同概念的 bandit 算法，即每次用户预测对 item 收益是由这个所属的群体的聚合权重向量参数所决定的，同时根据个人反馈更新个人参数，个人参数又隐式地影响群体参数和用户群体的划分。据 CLUB 算法论文介绍，在一些公共数据集中，取得了比 LinUCB 更好的效果，关于 CLUB 算法的更多细节请参考 https://arxiv.org/abs/1401.8257。

6. 结束语

本文简单介绍了推荐系统中一直存在的两大问题：冷启动和 EE 问题，并简单阐述了业界解决这两大问题的一些常见解决方法和算法。正如前文所说，EE 问题某种程度中一直以矛盾共同体存在，在实际场景中，需要平衡两者。

【本文作者简介】

文辉，达观数据联合创始人，主要负责大观数据爬虫系统、推荐系统等主要系统的研究和开发。同济大学计算机应用技术专业硕士，曾就职于盛大文学数据中心部门，负责爬虫系统、推荐系统、数据挖掘和分析等大数据系统的研发工作，在爬虫系统、Hadoop/Hive、数据挖掘等方面具备充足的研发和实践经验。

如何基于用户历史行为进行精准个性化推荐

于敬　达观数据科学家

在DT(data technology) 时代，网上购物、观看视频、聆听音乐、阅读新闻等各个领域无不充斥着各种推荐，个性化推荐已经完全融入人们的日常生活当中。个性化推荐根据用户的历史行为数据进行深层兴趣点挖掘，将用户最感兴趣的物品推荐给用户，从而做到千人千面，不仅满足了用户本质的信息诉求，也最大化了企业的自身利益，所以个性化推荐蕴含着无限商机。

号称“推荐系统之王”的电子商务网站亚马逊曾宣称，亚马逊有20% ~ 30% 的销售来自于推荐系统。其最大优势就在于个性化推荐系统，该系统让每个用户都能有一个属于自己的在线商店，并且在商店中能找到自己最感兴趣的商品。美国著名视频网站 Netflix 曾举办推荐系统比赛，悬赏 100 万美元，希望能将其推荐算法的预测准确度提升 10%。美国最大的视频网站 YouTube 曾做过实验比较个性化推荐和热门视频的点击率，结果显示个性化推荐的点击率是后者的两倍。

达观数据拥有雄厚的研发推荐系统的技术积累，曾在 ACM、CIKM、KDD、Hackathon 等国际竞赛的获奖，在内容推荐，文本挖掘、广告系统等方面申请有超过三十项国家发明专利。本文从数据处理、用户行为建模到个性化推荐，分享达观数据在个性化推荐系统方面积累的一些经验。

1. 数据收集及预处理

推荐系统的本质其实就是通过一定的方式将用户和喜欢的物品联系起来。物品和用户自身拥有众多属性信息进行标识。

1.1 物品属性

物品表示推荐系统的客体，在不同的应用场景下，物品指代不同的待推荐事物。比如，在书籍推荐中，物品表示书籍；在电商推荐中，物品表示商品；在电影推荐中，物品表示电影；在社交网络推荐中，物品表示人。商品有多种属性标识自己是什么。

图 1.1 商品属性

1.2 用户属性

用户表示推荐系统的主体，自身属性包括人口统计学信息以及从用户行为数据中挖掘分析得到的偏好。

图 1.2 用户属性

1.3 用户行为

用户的每一次的行为操作无不反应用户内心的本质需求，包括页面浏览、点击、收藏、购物、搜索、打分、评论等，这些数据是个性化推荐系统的最重要的数据。根据用户自身独有的行为数据，可以为每一个用户生成特有的画像。

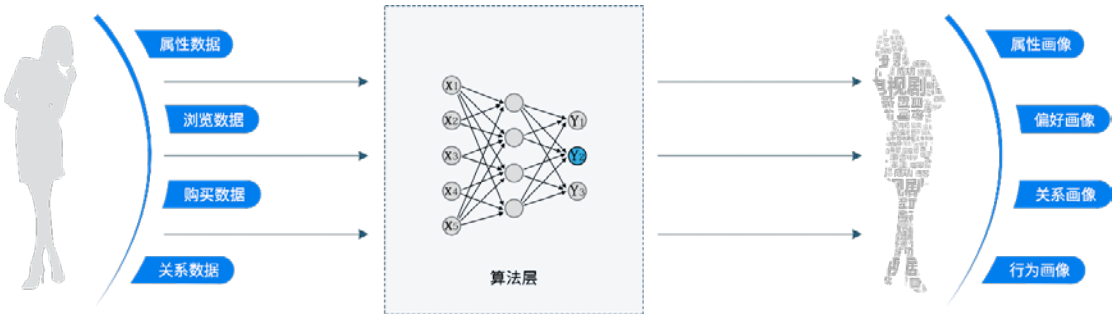


图 1.3 用户行为分析

1.4 数据处理

在数据采集的过程中，难免会出现一些脏数据，在使用数据前需要进行清洗。过滤掉关键字段为空、数值异常、类型异常等数据；用户 id 包括 cookie、手机号、email、注册 id 等，需要进行映射得到用户唯一 id；以及数据去重等操作。另外，还有“人为”的脏数据，如作弊、刷单等行为，这些数据也需要清除，否则会严重影响后续算法的效果。达观数据在反作弊方面也做了很多工作，可有效筛选各种行为上的作弊情况。

2. 用户行为建模

基于用户历史行为进行挖掘分析，会得到刻画用户本质需求的一组属性集合，即得到用户模型。个性化推荐的准确性很大程度上依赖于对用户属性刻画的准确性。达观数据采用了多种方式进行量化，主要包括显式用户偏好分析和隐式用户兴趣点挖掘。

2.1 显式用户偏好分析

结合用户历史行为和物品信息，可以得到每种行为下的用户偏好数据，包括偏好的维度及偏好程度，如偏好的物品、品牌、类别、标签等。再将各种行为的偏好数据合并，最终得到用户在物品、品牌、类别、标签等各个维度上的偏好程度。合并不同维度的数据时，需要考虑到不同的行为类型反映的用户偏

好程度是不同的。比如购买行为比点击行为更能反映用户的偏好，则由购买行为计算得到的偏好数据在合并时赋予的权重要高一些。要保证各种行为的各个维度的数据具有可比性，需要进行归一化，而且同纬度的要采用相同的归一化方法。

2.2 隐式用户兴趣点挖掘

除了结合物品信息进行分析计算得到的显式偏好外，还有一部分隐式兴趣点需要挖掘，这部分主要用于细分用户群体，进行有针对性的进行更有效的推荐。划分群体的准则要根据具体的业务需求而定，比如是否是高价值用户、是否价格敏感、是否对大牌情有独钟、大神用户和小白用户的区分、喜欢热门流行还是偏小众的等等。借助机器学习中的分类（如 SVM）和聚类（如 k-means）算法可有效解决用户群体的划分问题，牵涉到的训练和测试数据需要先根据一些规则粗略得到候选集，再结合人工标记的进行筛选。除了可以从行为数据中抽取特征外，也可以从物品和用户的属性数据中抽取特征。经过模型的训练、预测和后处理，从而将用户划分到不同的群体。

2.3 协同过滤的基石

在个性化推荐中，应用很广泛的是基于用户的协同过滤算法。这个算法最重要一点是相似用户的计算。

序号	计算方法	Score
1	皮尔森相关度	0.03826165210969024
2	欧式距离相似度	0.051503212529481694
3	余弦相似度	0.05212762106590356
4	Spearman秩相关系数	0.038273240558099296
5	曼哈顿距离	0.16820003126162392
6	Tanimoto系数	0.057949432508708404
7	对数似然相似度	0.05200320727068626

图 2.1 相似度计算方法对推荐效果的影响

相似度的计算很多种方法，如余弦相似度、皮尔逊相关度等，曾经使用 mahout 做过的一个不同相似度度量方法下的对比测试结果，测试中 score 的计算使用的是绝对差值的平均，越小越好。本次测试结果表明，在基于用户的协同过滤中，使用皮尔逊相关度的计算方法，推荐效果最好。

其实不同的相似度计算方法有各自的优缺点，适用不同的应用场景，可以

通过对比测试进行选取。在实际业务中，相似度的计算方法都有很多变种，比如是否考虑去除冷门物品和热门物品的影响。毕竟过于冷门和过于热门的物品对衡量用户间的相似度时区分度不好，这时就需要进行剪枝。这种基于 K 近邻的选取相似用户的方法，相似度的阈值设置对结果影响很大，太大的话召回物品过多，准确度会有下降。

2.4 时间维度上的考量

在处理各个维度的偏好数据时，需要考虑用户行为的有价值程度是随时间衰减的，即行为发生时间距当前的时间越近，得到的数据越能表征用户将来的行为。毕竟用户的口味随着时间的推移是会变化的，所以时间越近权重越高。

另外，还需要考虑偏好和兴趣点数据的在时间上的持续和变化过程，即需要刻画用户的口味呈现的时间规律。为了解决这个问题，我们根据不同的时间间隔来界定，分长期、短期、近期和实时四个时间维度。长期的覆盖了用户几乎一直不变的兴趣；短期的覆盖了用户变化中的兴趣；而近期则反映了用户的“尝鲜”的特点。这三种兴趣是离线计算的，还要考虑用户的实时兴趣，我们通过很短的时间间隔进行近线挖掘分析，从而快速适应用户当前的信息需求。通过上述过程，最终就为每个用户生成了各个维度上的偏好和兴趣点数据。

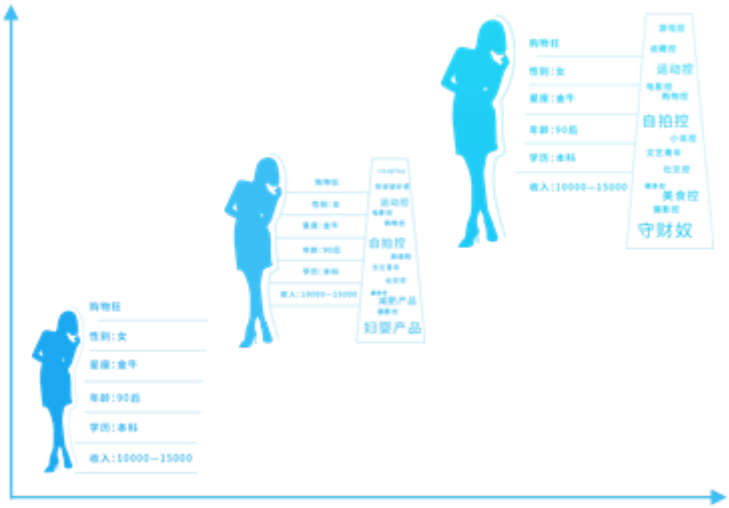


图 2.2 用户画像实时更新

3. 个性化推荐的实践经验

以用户模型和物品属性数据为载体，结合多种推荐算法和效果优化策略，个性化推荐系统将用户最感兴趣的物品精准推荐给当前用户。不同算法有自己

的应用场景，所以根据业务需要、数据的丰富程度、效果衡量指标等选择合适的推荐算法，然后根据推荐结果进行不断迭代，最终完成符合预期效果的个性化推荐系统。

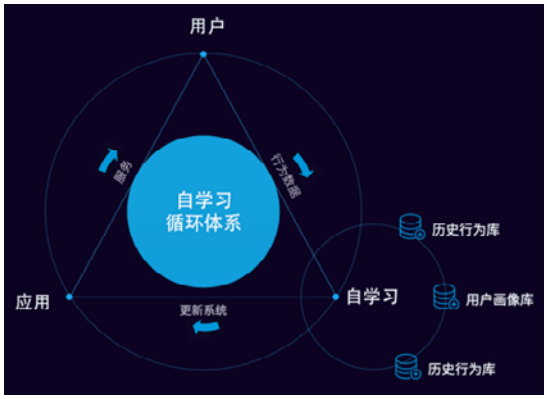


图 3.1 个性化推荐流程

3.1 基于内容的推荐

主要过程是将用户的信息特征和物品对象的特征相匹配的过程，从而得到待推荐的物品集合。通过用户模型中的类别、标签、品牌等各维度的偏好数据，在全量物品列表中寻找与之匹配的用户感兴趣的物品列表，并给出用户感兴趣的程度。根据挖掘的兴趣点，对部分用户进行有针对性的推荐，为其“量身定制”推荐结果，满足其特有的需求。基于内容的推荐方法，优点是能保证推荐内容的相关性，并且根据内容特征可以解释推荐结果，而且对新物品的推荐是也能有很好的考量。缺点是由于内容高度匹配，导致推荐结果的惊喜度较差，而且对新用户不能提供可靠的推荐结果。

3.2 基于协同过滤的推荐

协同过滤方法主要基于群体智慧，认为相似的用户对新物品的喜好也是相似的，相似的物品对于同一用户来说，喜好程度也是相似的。这种方法克服了基于内容方法的一些弊端，最重要的是可以推荐一些内容上差异较大但是又是用户感兴趣的物品。

方法大致分为两类：基于近邻的方法和基于模型的方法。前者在数据预测中直接使用已有数据进行预测，将用户的所有数据加载到内存中进行运算。基于模型的方法则是通过数据进行模型训练，然后为用户预测新的物品，主要包括：pLSA(Probabilistic Latent Semantic Analysis)、LDA(Latent Dirichlet Allocation)、SVM(Support Vector Machines)、SVD(Singular Value Decomposition)等。

基于用户模型中的相似用户列表和偏好的物品列表，分别使用基于用户的和基于物品的协同过滤，将相似用户喜欢的物品和相似的物品加入到推荐的候选集当中。同时，推荐权重的计算会考量相似度的大小及物品自身的质量分。

3.3 基于知识的推荐

当用户的行为数据较少同时又有明确的需求时，协同过滤和基于内容的推荐效果不尽人意，但是基于知识的推荐可以帮助我们解决这类问题。这种方法不需要用户行为数据就能推荐，所以不存在冷启动问题。推荐结果主要依赖两种形式，一是用户需求跟物品之间相似度，一种是明确的推荐规则。实际应用主要是以强规则为主。

3.4 补足策略

当用户历史数据比较局限或者在冷启动的时候，导致待推荐物品的数量不足没有达到预定要求时，根据用户模型的数据，结合挖掘的各种榜单进行补足，如全局热门、分类热门等。

3.5 多算法融合

单一算法有各自的优缺点，并不能满足实际的线上需求。为了提供最优质的个性化推荐服务，保证推荐结果的多样性、新颖性和惊喜度，需要融合多个推荐算法，进行混合推荐。常见的混合方法有以下几种：

1）加权式混合

主要是对每个算法赋予不同的权重，通过将多个推荐算法的结果进行加权组合在一起，最后排序得到推荐结果。

$$rec_{weighted}(u,i) = \sum_{k=1}^{n} \beta_k * rec_k(u,i)$$

不同推荐算法的结果需要归一化在相同的范围内，并且各个算法的权重之和为 1。

2）交叉式混合

主要是直接将不同的推荐算法的结果组合在一起推荐给用户，从而每个推荐算法的优质结果都会被展示给用户。

$$rec_{mixed}(u,i) = \bigcup_{k=1}^{n} \langle rec_k(u,i), k \rangle$$

3）切换式混合

主要是根据不同应用场景决定使用哪一种推荐算法，应用场景改变的话则切换推荐算法。例如在新闻推荐时，首先使用基于内容的推荐，当找不到合适的内容时，接着使用协同过滤算法进行跨内容的推荐，最后使用朴素贝叶斯分类器找到与用户长期兴趣匹配的结果。

5）串联混合

主要是将不同的推荐算法进行排序，后面的推荐算法对前面的不断优化，最终得到一个多级优化下的推荐结果。

6）分级混合

主要是先界定不同的算法的好坏，优先使用好算法的推荐结果，得不到结果时再使用次好的，依次类推。

达观数据在实践中充分利用了各种混合方法来提高推荐效果，并取得了优异的成效。例如基于加权式和分级混合的流程是，首先通过权重的大小来衡量每种推荐算法结果的好坏，产生待推荐的物品集合，在合并的时候，将优先使用好的推荐算法的结果。实践中则是各种指标综合权衡，整个过程也要复杂很多。

3.6 重排序

排序学习 (Learning To Rank, LTR) 一直是机器学习中的热门研究领域。由于排序过程牵涉到各种维度的参数数据，导致调参费时费力，而且很可能会出现过拟合现象。而机器学习方法不仅有成熟的理论基础，而且很容易融合多种特征，通过不断的迭代来进行参数优化，可有效解决数据稀疏、过拟合等问题。著名的 Netflix 公司就在他们的推荐系统中全面应用了 LTR 技术。

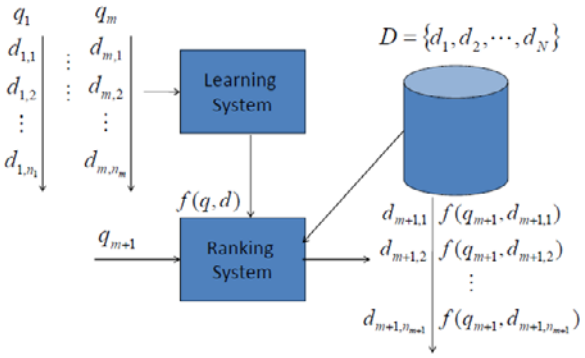


图 3.2 排序学习的流程

对于已标注的训练集，首先选定 LTR 方法，确定损失函数，以最小化损失函数为目标进行优化即可得到排序模型的相关参数，这就是学习过程。预测过程将待预测结果输入学习得到的排序模型中，即可得到结果的相关得分，利用该得分进行排序即可得到待预测结果的最终顺序。LTR 分按点 (pointwise)、按对 (pairwise) 和按表 (listwise) 三种方法，涉及到的常见模型有 LR(Logistic Regression)、SVM、DT(Decision Tree)。

关于排序模型的选择，LR 算法主要适用于特征数很多、样本量很大的情况。如果是样本量很大，但是特征比较少的情况时，建议使用 DT 的算法。主要是因为在特征数较少时，对应的问题往往是非线性的，此 DT 算法可以发挥自身的优势。另外，SVM 在解决非线性分类问题是效果也非常好。相对于另外两种方法，按表的方法往往更加直接，它专注于自己的目标和任务，直接优化排序结果，因此往往效果也是最好的。

经过多个推荐算法的处理，最终得到待推荐物品的结合，使用少量维度的特征进行排序过于简单，效果也大打折扣。基于推荐算法得到的相关特征，结合物品和用户的特征进行组合，可以得到各种特征，并且有些特征是正相关有些是负相关，需要不断优化。借助机器学习方法得到了最终的物品排序，呈现给用户。

4. 结束语

本文从构建用户模型到个性化推荐，介绍了达观数据的一些实践经验。个性化推荐系统能有效解决信息过载和长尾物品两个方面的问题，不仅提供了极佳的用户体检，满足了用户的信息需求，也帮助了企业挖掘其中蕴含的无限商机。

【本文作者简介】

于敬，达观数据联合创始人，中国计算机学会（CCF）会员，第 23 届 ACM CIKM Competition 竞赛国际冠军，达观数据个性化推荐组总负责人，工作包括推荐系统的架构设计和开发、推荐效果优化等。同济大学计算机应用技术专业硕士，承担公司重大紧急项目的架构设计和个性化推荐研发管理工作，所开发的个性化推荐系统曾创造了上线后效果提升 300% 的记录。先后在盛大创新院和盛大文学数据中心从事用户行为建模、个性化推荐、大数据处理、数据挖掘和分析、文本智能审核、反作弊和广告投放引擎相关工作，对智能推荐、数据挖掘、大数据技术和广告引擎有较深入的理解和多年实践经验。

多模型融合推荐算法在达观数据的运用

纪达麒　达观数据 CTO

1. 研发背景

互联网时代也是信息爆炸的时代，内容太多，而用户的时间太少，如何选择成了难题。电商平台里的商品、媒体网站里的新闻、小说网站里的作品、招聘网站里的职位……当数量超过用户可以遍历的上限时，用户就无所适从了。对海量信息进行筛选、过滤，将用户最关注最感兴趣的信息展现在用户面前，能大大增加这些内容的转化率，对各类应用系统都有非常巨大的价值。

搜索引擎的出现在一定程度上解决了信息筛选问题，但还远远不够，其存在两个主要弊端：

第一，搜索引擎需要用户主动提供关键词来对海量信息进行筛选。当用户无法准确描述自己的需求时，搜索引擎的筛选效果将大打折扣，而用户将自己的需求和意图转化成关键词的过程有时非常困难（例如“找家附近步行不太远就可以到的餐厅，别太辣的”）。更何况用户是懒惰的，很多时候都不愿意打字；

第二，搜索结果往往会照顾大多数用户的点击习惯，以热门结果为主，很难充分体现出个性化需求。

解决这个问题的最好工具就是——推荐系统（Recommendation System）。

推荐系统的效果好坏，体现在推荐结果的用户满意度上，按不同的应用场景，其量化的评价指标包括点击率、成交转化率、停留时间增幅等。为了实现优秀的推荐效果，众多的推荐算法被提出，并在业界使用。但是其中一类方法非常特殊，我们称为多模型融合算法。融合算法的意思是，将多个推荐算法通过特定的方式组合的方法。融合在推荐系统中扮演着极为重要的作用，本文结合达观数据的实践经验为大家进行系统性的介绍。

2. 为什么需要融合推荐算法

推荐系统需要面对的应用场景往往存在非常大的差异，例如热门 / 冷门的内容、新 / 老用户，时效性强 / 弱的结果等，这些不同的上下文环境中，不同推荐算法往往都存在不同的适用场景。不存在一个推荐算法，在所有情况下都胜过其他的算法。融合方法的思想就自然而然出现了，就是充分运用不同分类

算法的各种优势，取长补短，组合形成一个强大的推荐框架。俗话说就叫“三个臭皮匠顶个诸葛亮”。

在介绍融合方法前，首先简单介绍几类常见推荐算法的优缺点。

基于物品的协同过滤（Item-based Collaborative Filtering）是推荐系统中知名度最高的方法，由亚马逊（Amazon）公司最早提出并在电商行业内被广泛使用。

$$s_{ij} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \quad p_{ui} = \sum_{j \in S(i,k) \cap N(u)} s_{ij} p_{uj}$$

基于物品的协同过滤在面对物品冷启动（例如新上架物品），或行为数据稀疏的情况下效果急剧下降。另外，基于物品的协同过滤倾向于为用户推荐曾购买过的类似商品，通常会出现多样性不足、推荐惊喜度低的问题。

基于用户的协同过滤（User-based Collaborative Filtering），其公式略有不同：

$$s_{uv} = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} \quad p_{ui} = \sum_{v \in S(u,k) \cap N(i)} s_{uv} p_{vi}$$

基于用户的协同过滤在推荐结果的新颖性方面有一定的优势，但是推荐结果的相关性较弱，而且容易受潮流影响，推荐大众热门物品。同时新用户或低活跃用户也会遇到用户冷启动的棘手问题。

还有一类方法称为基于模型的方法。常见的有隐语义与矩阵分解模型（Latent Factor Model），LFM 对评分矩阵通过迭代的方法进行矩阵分解，原来评分矩阵中的 missing value 可以通过分解后的矩阵求得。

在达观数据的实践经验里，LFM 通常是推荐精度较好的一类计算模型。但当数据规模大时其运算性能会明显降低，同时计算依赖全局信息，因而很难作增量更新，导致实际工程中会遇到不少困难。而且隐语义模型还存在调整困难、可解释性差等问题。

基于内容的推荐算法（Content-based Recommendation）是最直观的推荐算法，这个方法实现简单方便，不存在冷启动问题，应对的场景丰富，属于“万金油”型打法。例如按同类别、同标签等进行推荐。但在一些算法公开评测中，基于内容的方法效果都是效果较差的。原因是基于内容的方法缺少用户行为的分析，存在“结果相关但是不是用户想要的”这样难以克服的问题。同时该算法往往受限于对文本、图像或音视频内容分析的技术深度，很难准确把握住用户真正关注的“内容点”。

基于统计思想的一些方法，例如 Slope One，关联规则 (Association Rules)，或者分类热门推荐等，计算速度快，但是对用户个性化偏好的描述能力弱，实际应用时也存在各种各样的问题，在此不多赘述。

即使相同的算法，当使用不同数据源时也会产生不同的推荐结果。比如协同过滤，使用浏览数据和使用交易数据得到的结果就不一样。使用浏览数据的覆盖面比较广，而使用交易数据的偏好精度比较高。

3. 常见的多模型融合算法

经过达观数据的众多实践发现，多模型融合算法可以带来比单一模型算法有极为明显的效果提升。但是怎样进行有效的融合，充分发挥各个算法的长处？这里总结一些常见的融合方法。

3.1 线性加权融合法

线性加权是最简单易用的融合算法，工程实现非常方便，只需要汇总单一模型的结果，然后按不同算法赋予不同的权重，将多个推荐算法的结果进行加权，即可得到结果：

$$Score(u,i)=\sum_{k=1}^{n}\beta_k * rec_k(u,i)$$

Score(u,i) 是给用户（user）推荐商品（item）的得分，β k 是算法 K 的权重，reck(u,i) 是算法 k 得到的用户（user）对商品 item 的推荐得分。这种融合方式实现简单，但效果较差。因为线性加权的参数是固定的，实践中参数的选取通常依赖对全局结果升降的总结，一旦设定后，无法灵活的按照不同的推荐场景来自动变换。比如如果某个场景用算法 A 效果较好，另外一种场景用算法 B 效果较好，线性融合的方式在这种情况下不能取得好的效果。为了解决这个问题，达观数据进行了改进，通过引入动态参数的机制，通过训练用户对推荐结果的评价、与系统的预测是否相符生成加权模型。动态的调整权重使得效果大幅提升。

3.2 交叉融合法

交叉融合常被称为 Blending 方法，其思路是在推荐结果中，穿插不同推荐模型的结果，以确保结果的多样性。

这种方式将不同算法的结果组合在一起推荐给用户 $rec(u)=\bigcup_{k=1}^{n}(rec_k(u))$

交叉融合法的思路是“各花入各眼”，不同算法的结果着眼点不同，能满足不同用户的需求，直接穿插在一起进行展示。这种融合方式适用于同时能够展示较多条结果的推荐场景，并且往往用于算法间区别较大，如分别基于用户长期兴趣和短期兴趣计算获得的结果。

3.3 瀑布融合法

瀑布型（Waterfall Model）融合方法采用了将多个模型串联的方法。每个推荐算法被视为一个过滤器，通过将不同粒度的过滤器前后衔接的方法来进行。

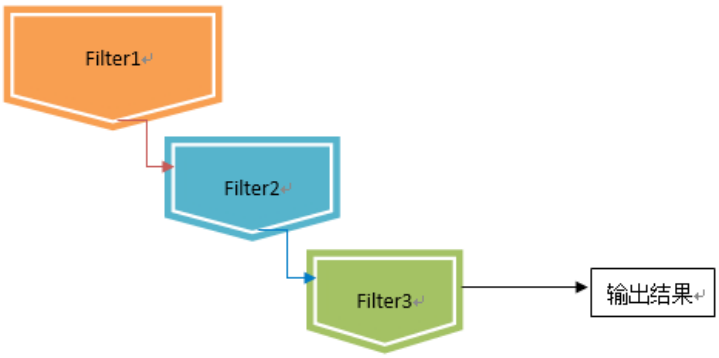


图 3.1 过滤器衔接

在瀑布型混合技术中，前一个推荐方法过滤的结果，将作为后一个推荐方法的候选集合输入，层层递进，候选结果在此过程中会被逐步遴选，最终得到一个量少质高的结果集合。这样设计通常用于存在大量候选集合的推荐场景上。

设计瀑布型混合系统中，通常会将运算速度快、区分度低的算法排在前列，逐步过渡为重量级的算法，让宝贵的运算资源集中在少量较高候选结果的运算上。在面对候选推荐对象（Item）数量庞大，而可曝光的推荐结果较少，要求精度较高且运算时间有限的场景下，往往非常适用。

3.4 特征融合法

不同的原始数据质量，对推荐计算的结果有很大的影响。以用户兴趣模型为例，我们既可以从用户的实际购买行为中，挖掘出用户的“显式”兴趣，又可以从用户的点击行为中，挖掘用户“隐式”兴趣；另外从用户分类、人口统计学分析中，也可以推测用户偏好；如果有用户的社交网络，那么也可以了解周围用户对该用户兴趣的影响。

所以通过使用不同的数据来源，抽取不同的特征，输入到推荐模型中进行训练，然后将结果合并。这种思路能解决现实中经常遇到的数据缺失的问题，

因为并非所有用户都有齐全的各类数据，例如有些用户就缺少交易信息，有些则没有社交关系数据等。通过特征融合的方法能确保模型不挑食，扩大适用面。

3.5 预测融合法

推荐算法也可以被视为一种“预测算法”，即我们为每个用户来预测他接下来最有可能喜欢的商品。而预测融合法的思想是，我们可以对每个预测算法再进行一次预测，即不同的算法的预测结果，我们可以训练第二层的预测算法去再次进行预测，并生成最终的预测结果。

如下图 3.2 所示，我们把各个推荐算法的预测结果作为特征，将用户对商品的反馈数据作为训练样本，形成了第二层预测模型的训练集合，具体流程如下。

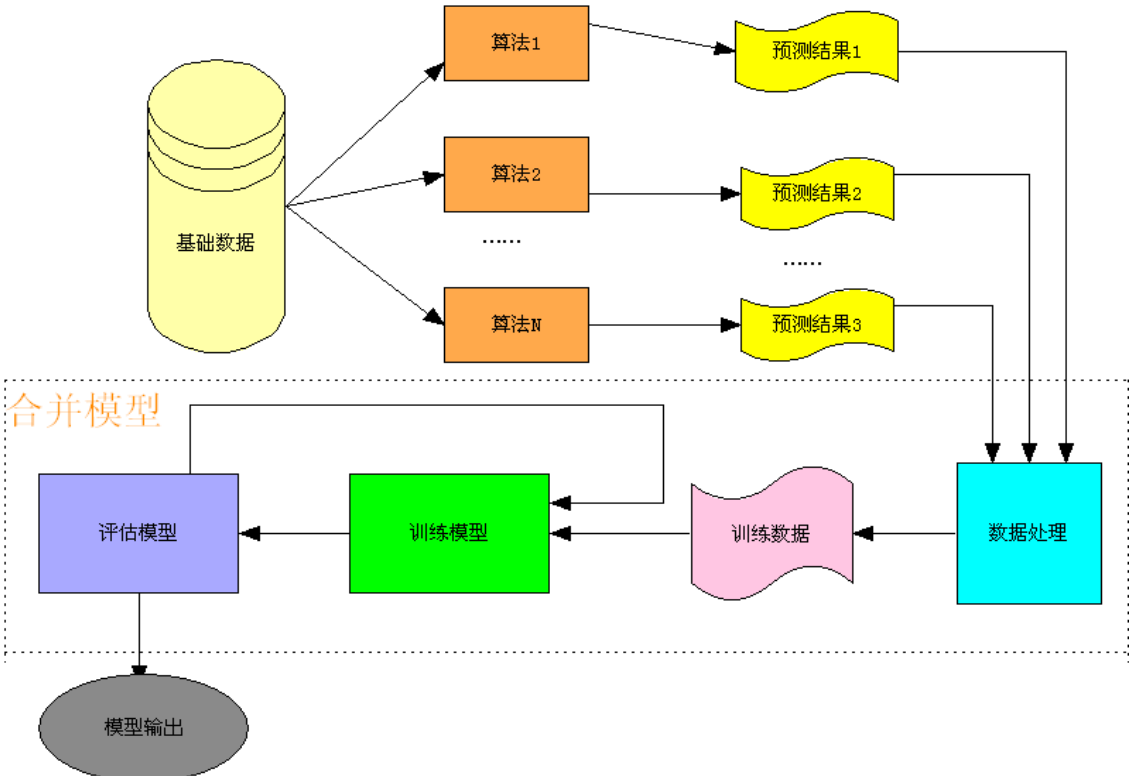


图 3.2 模型融合

图 3.2 中的二层预测模型可以使用常用的分类算法，如 SVM、随机森林、最大熵等，但达观实践中，融合效果较好的是 GBDT(Gradient Boosting Decision Tree) 方法。

3.6 分类器 Boosting 思想

推荐问题有时也可以转化为模式分类(Pattern Classification)问题去看待，我们将候选集合是否值得推荐划分为几个不同的集合，然后通过设计分类器的

方法去解决。这样一来我们就可以用到分类算法中的 Boosting 思想，即将若干个弱分类器，组合成一个强分类器的方法。

Boosting 的核心思想是每轮训练后对预测错误的样本赋以较大的权重，加入后续训练集合，也就是让学习算法在后续的训练集中对较难的判例进行强化学习，从而得到一个带权重的预测函数序列 h，预测效果好的预测函数权重较大，反之较小。最终的预测函数 H 对分类问题采用有权重的投票方式，对回归问题采用加权平均的方法对新示例进行判别。算法的流程如下（参考自 treeBoost 论文）。

Algorithm 6: L_K_TreeBoost

$F_{k0}(\mathbf{x}) = 0, \quad k = 1, K$

For $m = 1$ to M do:

$p_k(\mathbf{x}) = \exp(F_k(\mathbf{x})) / \sum_{l=1}^{K} \exp(F_l(\mathbf{x})), \quad k = 1, K$

For $k = 1$ to K do:

$\tilde{y}_{ik} = y_{ik} - p_k(\mathbf{x}_i), \quad i = 1, N$

$\{R_{jkm}\}_{j=1}^{J} = J$ terminal node $tree(\{\tilde{y}_{ik}, \mathbf{x}_i\}_1^N)$

$\gamma_{jkm} = \frac{K-1}{K} \frac{\sum_{\mathbf{x}_i \in R_{jkm}} \tilde{y}_{ik}}{\sum_{\mathbf{x}_i \in R_{jkm}} |\tilde{y}_{ik}| (1 - |\tilde{y}_{ik}|)}, \quad j = 1, J$

$F_{km}(\mathbf{x}) = F_{k,m-1}(\mathbf{x}) + \sum_{j=1}^{J} \gamma_{jkm} 1(\mathbf{x} \in R_{jkm})$

endFor

endFor

end Algorithm

图 3.3 算法流程

通过模型进行融合往往效果最好，但实现代价和计算开销也比较大。

4. 达观的多级融合技术

在达观数据（www.datagrand.com）的实践中，采用的多级融合架构如下：

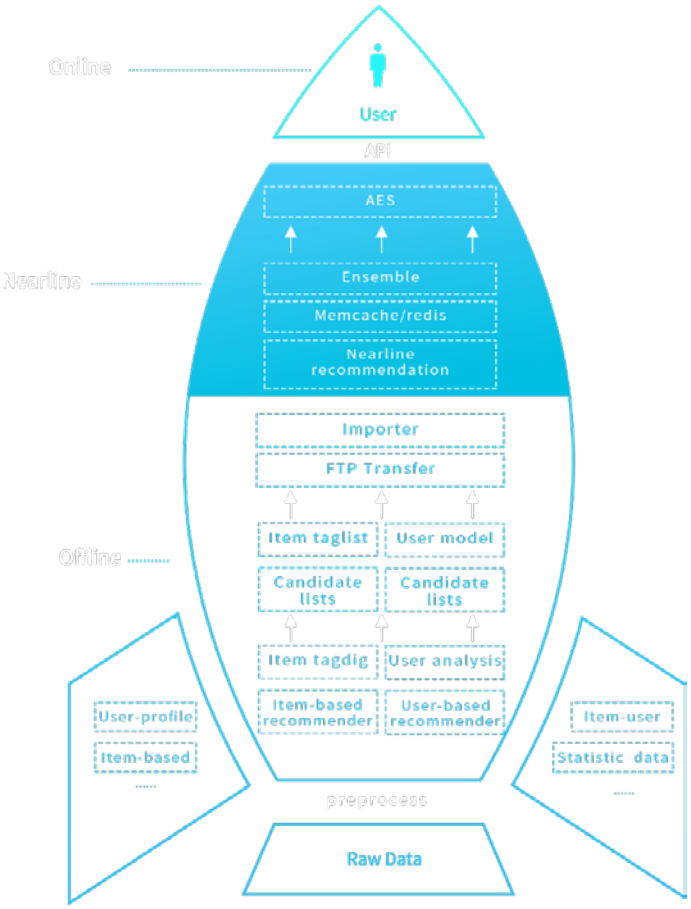


图 4.1 达观数据三级架构模型

4.1 Online 系统

直接面向用户，是一个高性能和高可用性的推荐服务，其中的 Online Ensemble 模块会融合 Nearline 计算的推荐结果以及基于 content Base 的推荐结果。Online 系统往往请求压力比较大，需要在较短的时间内返回结果，所以这里往往使用最简单的优先级融合算法。

4.2 Nearline 系统

这个系统部署在服务端，一方面会接收 User Behavior Log，根据用户最新的动作行为，生成推荐结果，并且和 Offline Model 进行融合，达观这边使用通过点击反馈进行调整的线性融合方法，具体方法如下。

1) Nearline 获取用户的展现日志和点击日志。展现日志包括了用户展现的哪些 item，以及这些 item 是通过什么算法推荐出来，推荐的位置，以及对应的权重。

2) 如果是展现日志，则减小推荐出 item 对应策略的权重，更新方式如下：

$$\beta'_k = \beta_k * (1 - \lambda * Ctr_i) * (1 - \varepsilon * \frac{Score_k}{Score_{item}})$$

β ‘k 是更新后的权重，Ctri 是展现位置 i 的平均点击率， Scorek 是算法 K 对该 item 的得分。Scoreitem 是该 item 的总得分。λ 是位置点击率的衰减常数、 ε 是算法点击率的衰减常数，可以根据具体的业务场景设置不同的值。

3) 如果是点击日志，则增加推荐出 item 的对应策略的权重，更新方式如下：

$$\beta'_k = \beta_k * (1 + \partial * \frac{Score_k}{Score_{item}})$$

β ‘k 是更新后的权重，Scorek 是算法 K 对该 item 的得分，Scoreitem 是该 item 的总得分 ,∂ 是点击衰减常数。

4) 根据更新后的权重，重新计算该用户的推荐结果。

通过这种融合方式，会为每个用户生成一个加权线性融合算法的 Model，根据这个 Model 计算出对应的推荐结果。

4.3 Offline 系统

挖掘长期的、海量的用户行为日志。以优化点击率为例，我们可以把用户的展现过的 item，以及是否点击形成训练数据，我们就需要生成一个是否点击的分类模型。我们的分类器分为两个 Level: L1 层和 L2 层。L1 层是基础分类器，可以使用协同过滤、矩阵分解、contentbase 等基础算法；L2 层基于 L1 层，将 L1 层的分类结果形成特征向量，再组合一些其他的特征后，形成 L2 层分类器（如 GBDT）的输入。Ensemble 的训练过程稍微复杂，因为 L1 层模型和 L2 层模型要分别进行训练后再组合。实践中我们将训练样本按照特定比例切分开，分别简称为 Train pig 和 Test Pig。

基于划分后的样本，整个训练过程步骤如下：

1）使用 Train pig 抽取特征，形成特征向量后训练 L1 层模型；

2）使用训练好的 L1 层模型，预测 Test pig，将预测结果形成 L2 层的输入特征向量；

3)结合其他特征后，形成 L2 层的特征向量，并使用 Test pig 训练 L2 层模型;

4）使用全部训练样本（Tain pig + Test pig）重新训练 L1 层模型；

5）将待测样本 Test 抽取特征后先后使用上述训练好的 L1 层模型生成预测结果，再把这些结果通过 L2 层 Ensemble 模型来生成最终的预测结果。

4.4 使用心得

1）算法融合的特征除了算法预测值之外，也可以加入场景特征。比如在场景 1 算法 A 效果较好，场景 2 中算法 B 效果较好，当我们把场景也作为特征进行融合，就可以学习出这种特征。

2）数据切分一定要干净。往往容易犯的错误是基础算法用的一些词典使用了全部的数据，这会使得融合算法效果大打折扣，因为相当于基础算法已经提前获知了融合算法的测试数据。

3）基础算法的区分度越好，融合算法的效果越好，比较不容易出现过拟合。

4）L2 层也一样可能出现过拟合（Overfitting），所以也可以加交叉验证，L2 层示意图如下图所示。

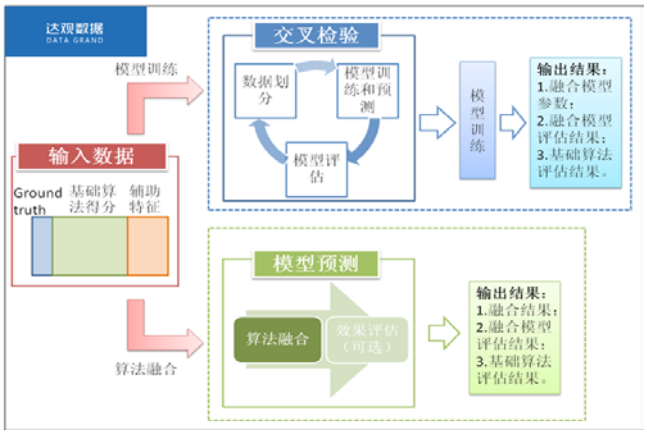


图 4.2 EM 算法

5. 总结和展望

推荐系统中的融合技术是非常重要的一个环节，在实战中，灵活运用融合技术可以发挥各个算法的长处，满足多样的用户需求，大大提升推荐结果的质量，达观数据在此方面将不懈努力，探索出更多更好的应用。

【本文作者简介】

纪达麒，达观数据首席技术官（CTO），研发团队总负责人，中国计算机学会（CCF）会员。拥有 10 年技术团队管理经验，在加入达观前，曾担任腾讯文学数据中心高级研究员、盛大文学技术总监，搜狗广告系统高级研发工程师，百度工程师等职务，擅长数据挖掘以及实时服务系统架构设计工作。所开发的个性化推荐系统曾创造了上线后效果提升 300% 的记录，曾代表公司多次参加国际数据挖掘竞赛，是 ACM KDD-Cup，CIKM Competition 等世界一流数据挖掘竞赛获胜队伍的核心成员。

扫描二维码申请试用
达观数据 NLP 产品

个性化推荐系统的实践与优化

于敬　达观数据科学家

在当今 DT 时代，每天都在产生着海量的数据，移动互联网的兴起更是让我们体验到获取信息是如此的简单和方便。同时，更多的选择也带来更多的困扰，面对层出不穷的信息和服务带来的困扰，使得个性推荐迅速崛起，并且大放异彩，在金融、电商、视频、资讯、直播、招聘、旅游等各个领域都能看到推荐系统的存在。

达观数据凭借多年在推荐系统方面的技术积累和优质的大数据服务，已经有数百家公司接入达观推荐系统，覆盖多个行业，实现企业经营业绩的大幅提升。本次分享结合达观数据个性化推荐引擎在各个行业的从业经验，围绕以下内容展开：

1）个性化推荐应用场景和价值；

2）用户画像和个性化推荐算法；

3）推荐系统优化方法。

1. 个性化推荐应用场景和价值

首先，我们来说说个性化推荐应用场景和价值。

图 1.1 企业面临的问题

个性化推荐产生的初衷是为了解决信息过载和物品长尾的问题。信息过载就是个人接受的信息超过了个人所能处理或有效利用的范围，导致人出现无所适从的问题。同时，如此多样丰富的信息中，大部分是属于冷门而没有曝光的机会。对于处于移动互联网的今天，这些问题尤其突出。

对用户而言，每天面对海量的资讯、商品、视频、音乐等各种服务时，如何快速找到自己感兴趣的内容确实是件耗费时间和精力的事情，尤其是在没有明确意图的情况下。

对于企业而言，手握海量资源，却只有一小部分内容曝光在用户面前，大部分都石沉大海，不仅造成资源浪费，还留不住用户。所以在当前各种产品同质化的今天，如何讨好并留住用户，挖掘数据中存在的价值，对企业也是一种极大的挑战。

越来越多的事实证明，个性化推荐系统是解决上述问题的有效工具。

美国最大的视频网站 YouTube 曾做过实验比较个性化推荐和热门视频的点击率，结果显示个性化推荐的点击率是后者的两倍。美国著名视频网站 Net-flix 曾举办过全球的推荐系统比赛，悬赏 100 万美元，希望参赛选手能将其推荐算法的预测准确度提升至少 10%。号称“推荐系统之王”的电子商务网站亚马逊曾宣称，亚马逊有 35% 的销售来自于推荐系统。其最大优势就在于个性化推荐系统，该系统让每个用户都能有一个属于自己的在线商店，并且在商店中能招到自己最感兴趣的商品。

日常生活当中，当我们打开各种各样的 app 和网页，首先进入视野的很多都是个性化推荐。对于企业而言，推荐系统可以让更多的资源得到曝光、改善用户体验、增加用户的停留时长和粘性，最终提高用户转化。

最后总结下，推荐系统的核心价值主要包括：

1）描述物品的特点，并与用户的个性化偏好进行匹配，帮用户便捷的筛选出感兴趣的内容；

2）进行有效的信息过滤以解决用户的过载问题，面对陌生领域时提供参考意见；

3）根据用户反馈迅速捕捉用户的兴趣，以及兴趣的变化，需求不明确时，作用户的“贴心助手”；

4）选择合适的场景、时机、表现方式进行推荐，满足用户的好奇心。

2. 用户画像和个性化推荐算法

个性化推荐的两个关键点：用户画像和个性化推荐算法。

目前普遍存在的两种个性化推荐结果生成方法。

一是依靠人工编辑进行推荐。这种方式不仅需要大量的人力成本且费时费力，最终推荐出来的结果是千篇一律，没有考虑用户个性化的差异，更没有考虑用户反馈。

二是通过一定逻辑生成的热门榜单。这种方式极易导致马太效应，一些热门物品会会一直霸占榜单，也容易造成刷单，毕竟占据着更好的流量入口，需要引入反作弊机制才能保证推荐结果的公平公正。

高质量的个性化推荐系统其实包括三大部分：

1）基于海量用户行为数据，挖掘多种多样的高质量推荐候选集；

2）对用户实时兴趣进行精准定位，秒级更新结果满足个性化需求；

3）基于高性能分布式计算框架，快速迭代算法生成多维度用户画像进行千人千面的推荐。

影响个性化推荐精准性的至关重要的因素是生成用户画像。用户画像通过对海量的用户行为数据进行深入的分析和挖掘，从多个维度来描述用户的基础属性、标签及兴趣点等，清晰并且准确地勾勒出用户的轮廓概貌。这些数据帮助应用方更好了解用户行为路径，明确用户流失情况和原因，为应用方的产品功能优化决策提供可靠参考依据。

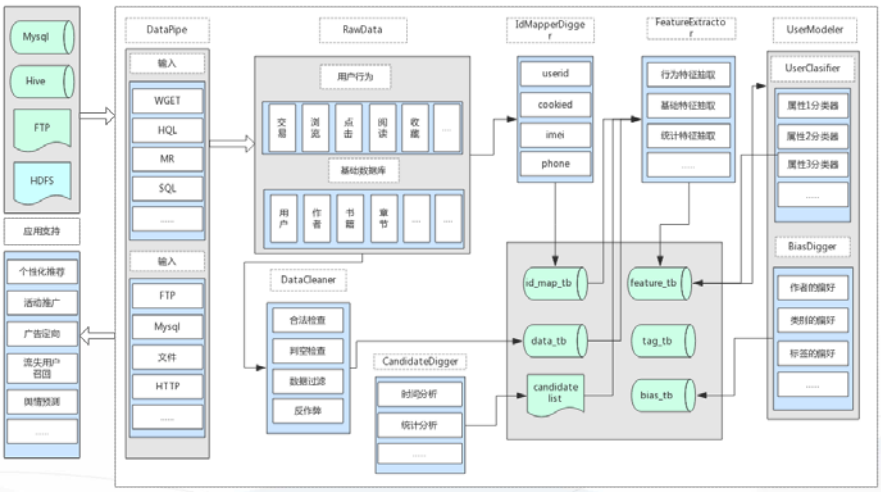


图 2.1 小说类客户的用户画像生成流程

通过多种方式收集到用户数据，包括用户各种行为数据，结合书籍、用户、作者等基础数据，首先进行数据预处理和用户归一化处理，然后进行数据统计与特征抽取，最后基于机器学习中的分类、聚类等方法进行智能挖掘分析，形成了用户各个维度的属性信息。

有了用户画像之后，接下来就交给个性化推荐算法了，这里主要说明基于内容的推荐和协同过滤。

2.1 基于内容的推荐

这个算法适合于待推荐物品拥有丰富语义信息的场景，如标题、标签、类别、作者等信息。但是像直播这个行业，直播内容和主播当前的播放状态紧密相关的，而且内容变化也较频繁，就不适宜使用此算法。在资讯媒体、视频等行业效果还是客观的。

基于内容的推荐主要过程是将推荐物品的信息特征和待推荐对象的特征相匹配的过程，从而得到待推荐的物品集合。匹配算法很多是借鉴了信息检索领域中的技术，如 K 最近邻 KNN 和 Rocchio 的相关性反馈方法，主要是以含有相同标签的其它物品、同类别的其它物品等形式出现。

这种方法能保证推荐内容的相关性，并且根据内容特征可以解释推荐结果。缺点是由于内容高度匹配，导致推荐结果的惊喜度较差，另外用户的反馈数据也没有使用。

2.2 协同过滤算法

主要思想是基于群体智慧，利用已有大量用户群过去行为数据来预测当前用户最可能感兴趣的东西。这种方法克服了基于内容方法的一些弊端，最重要的是可以挖掘物品之间隐含的相关性，推荐一些内容上差异较大但又是用户感兴趣的物品。

对于基于用户的协同过滤，首先计算用户之间的距离，得到与当前用户距离最近的 N 个用户，将这些用户喜欢的 item 进行合并和评分预测，得到推荐结果。基于物品的协同过滤则是计算物品间的距离进行评分预测得到推荐结果。

基于领域的方法重点关注物品之间的关系或者用户之间的关系，基于物品的方法是根据用户对和他感兴趣的物品相似的物品评分，来对该用户的偏好物品建立模型。

隐语义模型采用的是另外一种方法，把物品和用户映射到相同的隐语义空间。这个空间试图通过描述物品和用户两种实体在潜在因子上的特征来解释评分，而这些因子是根据用户的反馈自动判断出来的。

用隐语义模型来进行协同过滤的目标是揭示隐藏的特征，这些特征能解释观测到的评分。该模型包括 pLSA（Probability Latent Semantic Analysis）模型、神经网络模型、LDA（Latent Dirichlet Allocation）模型，以及由用户 - 物品评分矩阵的因子分解推导出的模型（也叫基于 SVD 的模型，Singular Value Decomposition）。

计算用户 - 用户距离和物品 - 物品距离有很多方法，在实际业务中会有很多的变形，比如对热门物品的降权，不然会引入一些噪声数据，因为相对一些不那么热门的物品更能表征用户的偏好信息。

由于矩阵因子分解技术在线上业务的准确性和稳定性的突出表现，它已经成为协同过滤算法的首选。

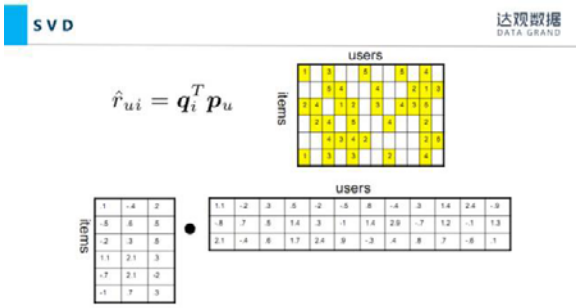


图 2.2 矩阵因子分解技术（一）

首先，对于 user−item 关系矩阵，也叫评分矩阵，表示用户对每个 item 的评分，比如 1 表示 1 分，5 表示 5 分，分数越高就表示越喜欢。

通过用户的操作行为数据我们就得了这样一个矩阵，通过矩阵分解的方式就得到了两个矩阵，分别是物品 – 潜在因子矩阵和潜在因子 – 用户矩阵，我们的目标是预测用户对未打分的物品的喜好程度，也就是图中除了黄色格子之外的数据。

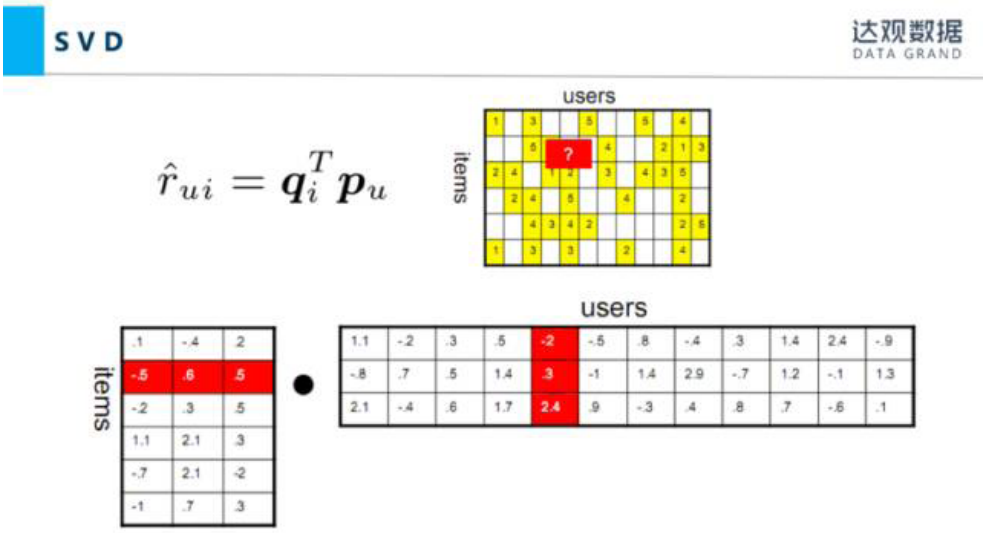


图 2.3 矩阵分解因子（二）

对于未知的评分，可以使用分解后两个矩阵相乘，就得到图中空白处的评分数据，进行排序和过滤，最终就可以得到用户对物品的喜好程度，也就得到最终的推荐结果。

原始的 SVD 并没有考虑到用户和物品自身的差异（bias），进行升级，我们来看看 SVD++ 是怎么的形式。

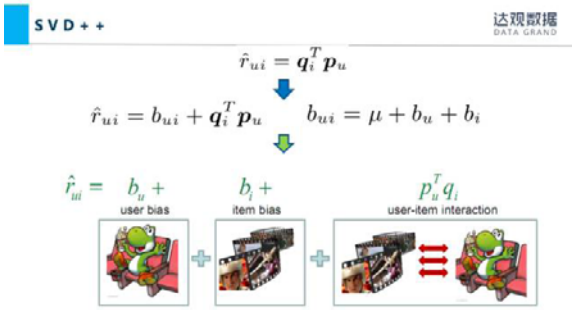


图 2.4 矩阵分解因子 ++

相对于之前的方式这里引入了 b ui，也就是用户本身的 bias 和物品本身的 bias。对于用户的 bias，有的用户倾向于打高分，有的倾向于打低分。物品的 bias，比如有的电影出自于大导演、大公司等倾向于得到高分，有的比较小众容易得低分。在计算中，这些 bias 信息都需要包含进来。

通过对算法的升级，就可以使用各种各样的用户、物品的 bias 信息，包括用户和物品的 profile，同时各种属性也可以进行组合，如用户性别、年龄信息。最终的模型求解问题就转化为求解最优化问题，这个最小二乘法问题可以通过随机梯度下降算法有效地解决。

图 2.5 达观数据推荐系统架构图

这里是达观数据推荐系统的架构图，从基础层的数据传输、存储和运算，到模型层的用户画像、物品画像等的挖掘生成，然后到多种推荐算法的计算得到部分初选的推荐候选集，最后交由组合层的机器学习模型进行重排序，生成最终的推荐结果返回给用户。整个流程在上百家客户上都取得了卓越的推荐效果，覆盖资讯、视频、直播、电商等多个行业。

接下来说说个性化推荐的优化，推荐系统的优化方法有很多种，今天挑选了三个，都和当前火热的深度学习有或多或少的关系，用于性能和效果的优化。

3. 推荐系统优化方法

达观数据接入了上百家客户，数据量的规模之大可想而知，对于性能的要求非常高，尤其在高并发的推荐场景中。基于硬件成本和性能的综合考虑，达观个性化推荐引擎除了使用内存和 redis 作为缓存之外，也引入了 LevelDB。

LevelDB 是 Google 的两位大神 Jeff Dean 和 Sanjay Ghemawat 发起的开源项目，简而言之，LevelDb 是能够处理十亿级别规模 Key-Value 型

数据持久性存储的 C++ 程序库。LevelDb 是一个持久化存储的 KV 系统，和 Redis 这种内存型的 KV 系统不同，LevelDb 不会像 Redis 一样狂吃内存，而是将大部分数据存储到磁盘上。

LevelDB 在随机写，顺序读 / 写方面具有很高的性能，但是随机读的性能很一般。换句话说，LevelDB 很适合应用在查询较少，而写很多的场景。

个性化推荐引擎需要尽可能快的响应用户的每一次操作，以适应用户短期兴趣的变化，进而提高推荐效果的精准性。各种推荐算法生成候选集、多算法融合、返回结果的时候，牵涉到频繁的读取操作。

在高并发量的场景下，当内存或者 redis 不足以完全支撑线上业务时，使用 LevelDB 将会对性能有不错的提升。毕竟 LevelDb 在写的时候对内存要求不高，读的时候则根据性能要求的不同需要对应的内存。

谈到效果优化，业界都达成了一个共识：case-by-case 查看推荐结果，也就是说需要具体问题具体分析。只有通过这种方式发现问题，才能更好地优化推荐效果。下面分析几个常见的优化方法。

3.1 推荐结果多样化的优化

在实际的推荐场景中，抓住了用户的喜好，但推荐出来满屏的“相似”结果会带来极差的用户体验。如何在保证用户兴趣的前提下又能让推荐结果的多样性更好呢?

物品信息中很重要的一个特征是标签。好的推荐系统不仅体现在精准性，还有多样性和惊喜度方面的要求。对物品信息进行深层度的挖掘分析，进而对标签进行拓展，也是一种实现上述要求行之有效的方法。其实使用 word2vec 就可以解决这类问题。

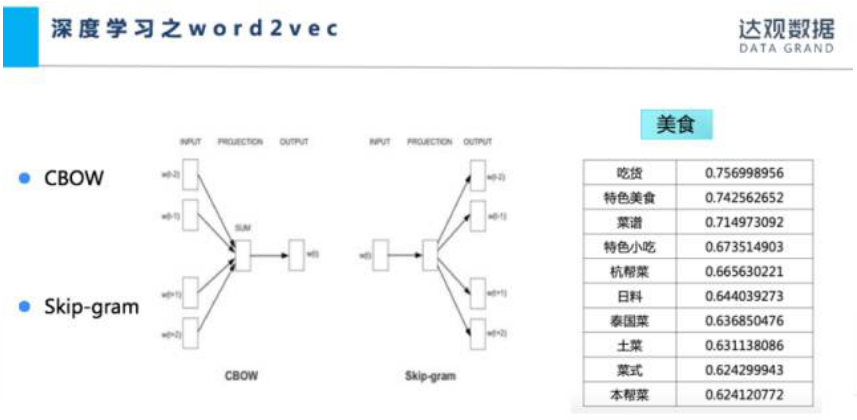


图 3.1 深度学习之 word2vec

Google 于 2013 年开源推出了一个用于获取 word vector 的工具包 word2vec，它包含了对两种模型的训练，如下图。在训练每种模型的时候又分 HS 和 NEG 两种方法。在 Word2Vec 的训练过程中，每个 word vectors 都被要求为相邻上下文中的 word 的出现作预测，所以即使随机初始化 Word vectors，但是这些 vectors 最终仍然能通过预测行为捕获到 word 之间的语义关系，从而训练到较好的 word vectors。

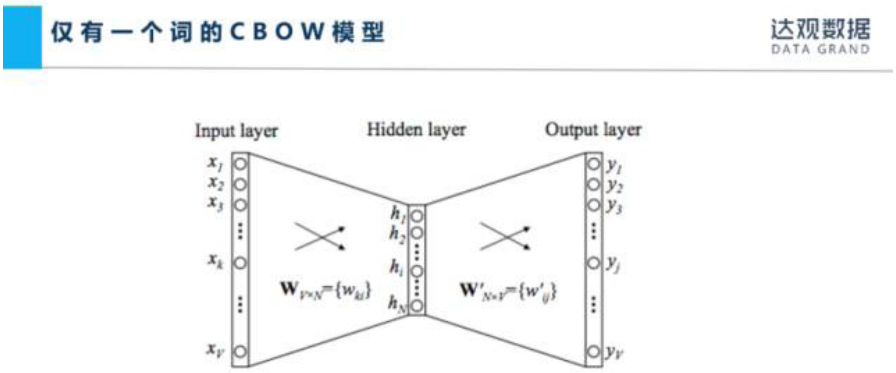


图 3.2 仅有一个词的 CBOW 模型

这是仅有一个词的 CBOW 模型。Word2Vec 尽量让具有相同上下文的 word 向量相似，从而获得较好的 vector representation 相似性。这种相似性有时候是线性的，临近的结果会与相似，即 Word2vec 可以学习到词与词之间语义上的联系。

另外，由于 Word2Vec 采用了非常多的方法简化网络结构，简化训练流程，导致 Word2Vec 可以很轻易地训练超大的训练集。一个优化后的单机实现版的 Word2Vec 算法可以在一天时间内训练 100 bililion words。word2vec 可以把对文本内容的处理简化为向量空间中的向量运算，计算出向量空间上的相似度，来表示文本语义上的相似度。基于得到的向量，也就得到了可以扩展的词。

从图 3.1 中可以看出，基于 word2vec 训练好的模型，输入“美食”，返回了相似程度最高的十个词及相似权重，从结果上看都是和美食相关的。基于这些相似词召回相关推荐结果，不仅可以保证语义上的相关性，也可以大大改善推荐结果的多样性。

3.2 item embedding

2016 年 Oren Barkan 以及 Noam Koenigstein 借鉴 word2vec 的思想，提出 item2vec，通过浅层的神经网络结合 SGNS(skip-gram with negative

sampling) 训练之后，将 item 映射到固定维度的向量空间中，通过向量的运算来衡量 item 之间的相似性。

词的上下文即为邻近词的序列，很容易想到，词的序列其实等价于一系列连续操作的 item 序列，因此，训练语料只需将句子改为连续操作的 item 序列即可，item 间的共现为正样本，并按照 item 的频率分布进行负样本采样。

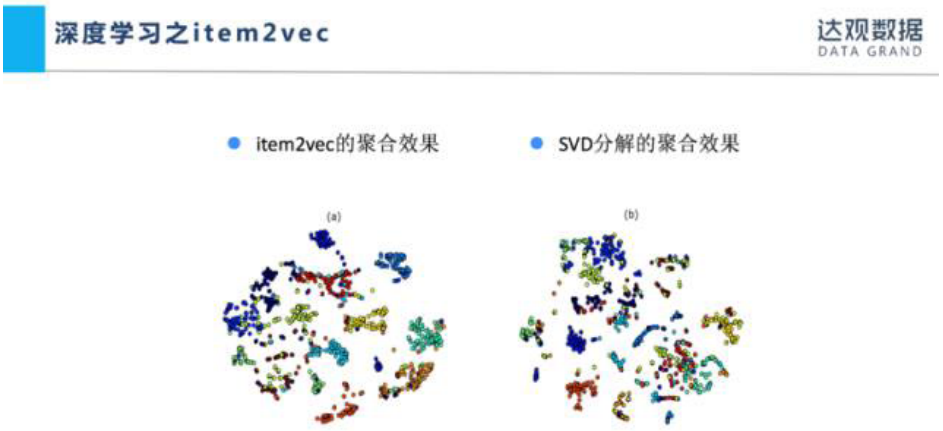


图 3.3 item2vec

Oren Barkan and Noam Koenigstein 以 SVD 作为 baseline，SVD 的隐类以及 item2vec 的维度都取 40，用 Microsoft Xbox Music service 收集的 user-artists 数据集，对结果进行聚类，同一个颜色的节点表示相同类型的音乐人。

图 a 是 item2vec 的聚合效果，图 b 是 SVD 分解的聚合效果，很显然 item2vec 的聚合效果更胜一筹。

【本文作者简介】

于敬，达观数据联合创始人，中国计算机学会（CCF）会员，第 23 届 ACM CIKM Competition 竞赛国际冠军，达观数据个性化推荐组总负责人，工作包括推荐系统的架构设计和开发、推荐效果优化等。同济大学计算机应用技术专业硕士，承担公司重大紧急项目的架构设计和个性化推荐研发管理工作，所开发的个性化推荐系统曾创造了上线后效果提升 300% 的记录。先后在盛大创新院和盛大文学数据中心从事用户行为建模、个性化推荐、大数据处理、数据挖掘和分析、文本智能审核、反作弊和广告投放引擎相关工作，对智能推荐、数据挖掘、大数据技术和广告引擎有较深入的理解和多年实践经验。

声　明

关注“达观数据”

立即试用

联系我们

邮箱：contact@datagrand.com
电话：400-175-9889
官网：www.datagrand.com
地址：上海市浦东新区亮秀路 112 号浦东软件园 Y1 座 515 室
北京市朝阳区西坝河西里 18 号正通创意中心 6 号楼
深圳市南山区粤海街道科文路 1 号华富洋大厦 4 层思微 B62