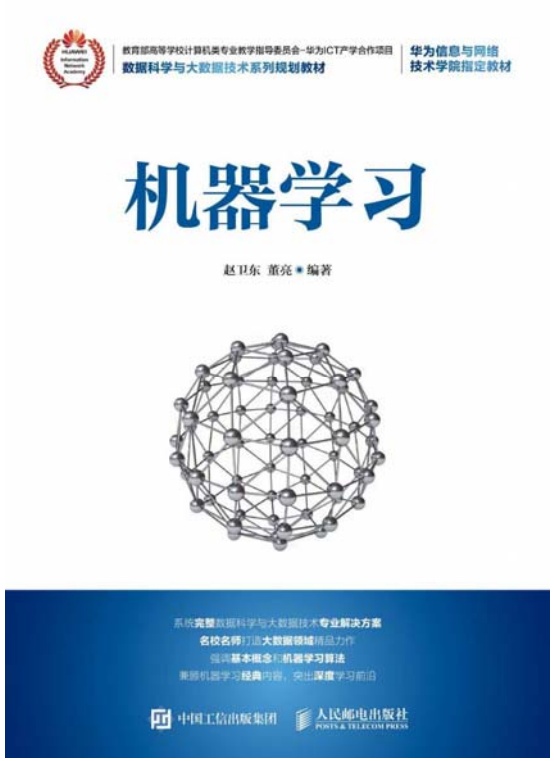


机器学习
文本分析

复旦大学 **赵卫东** 博士

wdzhao@fudan.edu.cn

章节介绍

- 文本分析是机器学习领域重要的应用之，也称之为文本挖掘。通过对文本内部特征提取，获取隐含的语义信息或概括性主题，从而产生高质量的结构化信息，合理的文本分析技术能够获取作者的真实意图。典型的文本挖掘方法包括文本分类、文本聚类、实体挖掘、观点分析、文档摘要和实体关系提取等，常应用于论文查重、垃圾邮件过滤、情感分析、智能机器和信息抽取等方面
- 本章首先介绍文本分析基础知识，然后对文本特征选取与表示、知识图谱、语法分析、语义分析等常见文本处理技术详细说明，最后介绍文本分析应用

章节结构

- 文本分析介绍
- 文本特征提取及表示
  - TF-IDF
  - 信息增益
  - 互信息
  - 卡方统计量
  - 词嵌入
  - 语言模型
  - 向量空间模型
- 知识图谱
  - 知识图谱相关概念
  - 知识图谱的存储
  - 知识图谱挖掘与计算
  - 知识图谱的构建过程

章节结构

- 词法分析
  - 文本分词
  - 命名实体识别
  - 词义消歧
- 句法分析
- 语义分析
- 文本分析应用
  - 文本分类
  - 信息抽取
  - 问答系统
  - 情感分析
  - 自动摘要

文本分析介绍

- 文本分析的过程从文本获取开始，一般经过分词、文本特征提取与表示、特征选择、知识或信息挖掘和具体应用等步骤

文本特征提取及表示

- 文本的特征表示是文本分析的基本问题，将文本中抽取出的特征词进行向量化表示，将非结构化的文本转化为结构化的计算机可以识别处理的信息，然后才可以建立文本的数学模型，从而实现对文本的计算、识别、分类等操作。通常采用向量空间模型(Vector Space Model, VSM)来描述文本向量，在保证原文含义的基础上,找出最具代表性的文本特征，与之相关的有TF-IDF、信息增益(Information Gain)和互信息(MI)等

TF-IDF

- TF-IDF (Term Frequency- Inverse Document Frequency)是一种文本统计方法，主要用来评估文本中的一个词对语料库中一篇文档的重要程度，其中Term Frequency指词频，即某一个给定的词语在该文件中出现的频率：

  假设词频为 $tf(w, d)$，w 为该词语，d 为既定文档，则 $tf(w, d)=count(w, d)/size(d)$，其中 $count(w, d)$表示词 w 在文档 d 中出现的词数，$size(d)$为该文档的总词数。

- Inverse Document Frequency指的是逆文档频率：

  文档总数 n 与词 w 所出现的文件数 $docs(w, D)$比值的对数，其中 D 表示语料库中的文件集，$idf=\log[n/docs(w, D)]$。

- 基本思想是：字词的重要性与它在当前文档中出现的次数(词频)成正比，与它在整个语料库中出现的频率成反比。例如，某个词在当前这篇文章中出现的词频较高，并且在其他文章中很少出现，则认为此词具有很好的类别区分能力，适合作为当前文章的特征词

信息增益

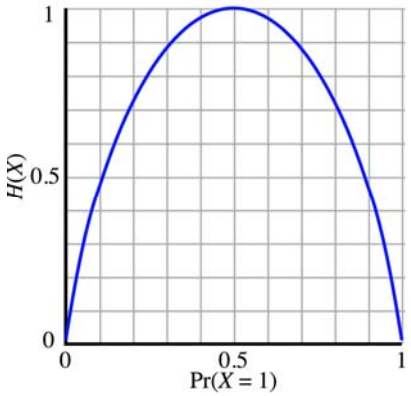
- 信息增益表示了某一个特征项的存在与否对类别预测的影响，定义为考虑某一特征项在文本中出现前后的信息熵之差
- 信息熵是信息论中对信息量多少的衡量指标，是对随机变量不确定性的度量，假如有变量X,它它可能的取值有n种，分别是$x_1, x_2, \ldots x_i, \ldots x_n$，每一种取到的概率分别是$p_1, p_2, \ldots p_i, \ldots p_n$，那么X的熵就定义为：

$$H(X) = -\sum_{i \in k} p_i * \log_2 p_i$$

- 一个变量可能的取值越多，它携带的信息量就越大，即熵与值的种类多少以及发生概率有关

信息增益

- 信息熵在分类问题时其输出就表示文本属于哪个类别的值
- 信息增益是信息论中比较重要的一个计算方法，估算系统中新引入的特征所带来的信息量，即信息的增加量
- 信息增益表示在其引入特征的情况下，信息的不确定性减少的程度，用于度量特征的重要性。可以通过计算信息增益来选择使用哪个特征作为文本表示



互信息

- 互信息(MI)表示两个变量x与y是否有关系，以及关系的强弱，可用于文本分类。MI计算的公式如下：

$$MI(t, C_i) = \sum_i p(C_i) \log \frac{p(t, C_i)}{p(t)p(C_i)}$$

- 从互信息的定义可见，某个特征词在某个类别C_i出现频率高，但在其他类别出现频率比较低，则它与该类C_i的互信息就会比较大。用互信息作为特征词和类别之间的测度，如果特征词属于该类，它们的互信息量最大。由于该方法为统计方法，不需要对特征词和类别之间关系的性质做任何假设，因此适合于文本特征和类别的匹配检验

互信息

- 点互信息(Pointwise Mutual Information, PMI) 可用于度量事物之间的相关性，在文本分析领域，可用其计算词语间的语义相似度，基本思想是统计两个词语同时出现的概率，如果概率越大，其相关性就越大，关联度越高。两个词语x与y的PMI值计算公式如下

$$PMI(x,y) = \log_2 \frac{p(x,y)}{p(x)p(y)}$$

- 互信息在文本处理方面的主要缺点是没有将词与词之间的顺序、句法和语义等信息考虑进去，这限制了某些领域的应用效果

卡方统计量

- 卡方x^2统计量在统计学中是用来检测两个事件的独立性。在文本特征选择中，两个事件分别指词项的出现与类别的归属，用于统计词项和类别之间的独立性。卡方的计算公式如下:

$$\sum_{i=1}^{n} \frac{(x_i - E)^2}{E}$$

- 卡方是基于显著性统计来选择特征的，它统计文档中是否出现词t，却不管t在该文档中出现了几次，这会使得它对低频词有所偏袒。特征选择时可能选出更多的少见词项，而漏掉出现次数较多的关键词，少见词项对分类并无用处，这就是所谓的“低频词缺陷”。因此卡方检验也经常与其他因素(如词频)综合来扬长避短
- 卡方和互信息的出发点不同，但它们的准确性却差不多，且都主要用于监督式文本分类，在非监督式分类中，般使用 TF-IDF 作为特征词的选取方法

词嵌入

- 词嵌入是将词转化为向量表示，即使用低维、稠密、实值的词向量来表示每个词， 从而使计算词语相关度成为可能。两个词具有语义相关或相似，则它们所对应的词向量之间距离相近。度量向量之间的距离可以使用经典的欧拉距离和余弦相似度等
- 在向量空间中，每一个词用1和0组成的向量表示（如[0,0,0,0,...,0,1,0,...,0,0,0]），有多少个词语就有多少维向量，这就是独热（one-hot）表示方法。如果要表示句子，则用句中的多个词构成一个向量矩阵。很明显，某种语言的词汇数量越多，词向量就越大，而句子的向量矩阵就会越大。但是，one-hot表示方法存在“词汇鸿沟” 问题，即词与词之间没有同义、词序、搭配等关联信息，仅从词的向量中看不出两个词之间关系。为了解决这一问题，就需要对词向量进行训练，建立词向量之间的关系。训练方法是通过大量的现有语料句子传入神经网络模型中，用模型的参数来表示各个词向量之间的关系

词嵌入

- 训练词向量的典型工具有Word2Vec和GloVe等
- Word2Vec认为经常在一个句子中出现的 词语相似度是比较高的，即对于一个中心词， 最大化周边单词的概率。Word2Vec采用三层网络进行训练，最后一层采用霍夫曼树(Huffman) 来预测
- GloVe是通过共现计数来实现的:首先，构建一个词汇的共现矩阵，每一行是一个词， 每列是句子。 通过共现矩阵计算每个词在每个句子中出现的频率。由于句子是多种词汇的组合，其维度非常大，需要降维，即对共现矩阵进行降维
- Word2Vec和Glove比较容易且快速地融合新的句子加入词汇表进行模型训练
- Glove在并行化处理上更有优势，处理速度较快

词嵌入

- gensim是一款开源的Python工具包，用于从文本中无监督地学习文本隐层的向量表示，并提供了相似度计算、信息检索等API接口。以下是gensim官网上训练和使用Word2Vec模型的demo代码

```
from gensim.models import Word2Vec
from gensim.models.word2vec import LineSentence
sentences = LineSentence('sentence_list.txt')
model = Word2Vec(sentences, size=128, window=5,
min_count=5, workers=4)
items = model.most_similar('学习')
for item in items:
    print item[0], item[1]
model.similarity('英语', '数学')
```

语言模型

- 语言模型(Language Model)是通过概率分布的方式来计算句子完整性的模型，广泛应用于各种自然语言处理问题，例如语音识别、机器翻译、分词、词性标注等。例如，用于确定哪个词语序列的可能性更大，即句子的合法性判断。或者用于给定若干个词，预测下一个最可能出现的词话。也可用于计算某句子中词语搭配是否合理
- 对于一个由词语组成的的句子$S = word_1, word_2, \ldots\ldots, word_n$,它的概率表示为

$$p(S) = p(word_1, word_2 \ldots\ldots word_k)$$
$$= p(word_1)p(word_2|word_1) \ldots p(word_k|word_1, word_2, \ldots, word_{k-1})$$

- 由于上式中的参数过多，计算复杂度过高，需要近似的计算方法。最常用n-gram模型方法，此外还有决策树、最大熵、马尔科夫模型和条件随机域等方法

语言模型

- n-gram模型也称为n-1阶马尔科夫模型，它是一个有限历史假设，即当前词的出现概率仅仅与前面n-1个词相关。当n取1、2、3时，n-gram模型分别称为unigram、bigram和trigram语言模型。n越大，模型越准确，也越复杂，需要的计算量就越大。最常用的是bigram，其次是unigram和trigram，n取≥4的情况较少
- 一般使用困惑度进行语言模型评测
- 训练工具有SRILM和rnnlm

语言模型

- 基于SRILM工具，可以用如下命令生成语言模型：

  ngram-count -text input.txt -lm output.lm

- 其中，input.txt是经过分词后的语料文本，每一行是一个句子。生成词频统计和语言模型保存在count.lm文件中。
- 执行如下命令可以基于语言模型来生成测试语句的困惑度：

  ngram -ppl test.txt -lm output.lm -debug 2 > test_result.ppl

- 其中test.txt是待测试的文本句子，每行是一个经过分词的句子。通过-lm指定在上步中训练好的语言模型。检测结果储存在test_result.ppl中，示例如下

语言模型

```
拥有 全新 骁龙 660 移动 平台 搭配 6G 运存 让 数据处理 高效
        p( 拥有 | <s> )      = [2gram] 0.01793821 [ -1.746221 ]
        p( 全新 | 拥有 ...)  = [2gram] 0.001913622 [ -2.718144 ]
        p( 骁龙 | 全新 ...)  = [1gram] 0.000736711 [ -3.132703 ]
        p( 660 | 骁龙 ...)   = [2gram] 0.02556118 [ -1.592419 ]
        p( 移动 | 660 ...)   = [1gram] 0.0001365131 [ -3.864826 ]
        p( 平台 | 移动 ...)  = [2gram] 0.0196641 [ -1.706326 ]
        p( 搭配 | 平台 ...)  = [1gram] 0.001986997 [ -2.701803 ]
        p( 6G | 搭配 ...)    = [2gram] 0.01205386 [ -1.918874 ]
        p( 运存 | 6G ...)    = [3gram] 0.3261201 [ -0.4866224 ]
        p( 让 | 运存 ...)    = [1gram] 0.005246758 [ -2.280109 ]
        p( 数据处理 | 让 ...)       = [1gram] 1.354035e-05 [ -4.86837 ]
        p( 高效 | 数据处理 ...)     = [1gram] 0.0005092599 [ -3.293061 ]
        p( </s> | 高效 ...)  = [2gram] 0.05939064 [ -1.226282 ]
1 sentences, 12 words, 0 OOVs
0 zeroprobs, logprob= -31.53576 ppl= 266.58 ppl1= 424.5999
```

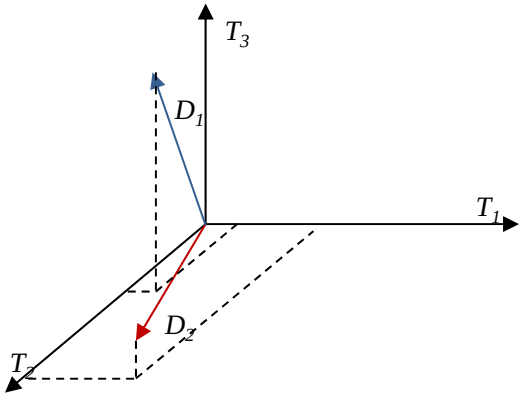
- 检测结果最后一行是评分基本情况，其中logprob是整个句子的概率，它由各词条件概率值相加得到的。ppl、ppl1均为困惑度指标，它们的值越小，句子质量越高

向量空间模型

- 向量空间模型能把文本表示成由多维特征构成的向量空间中的点，从而通过计算向量之间的距离来判定文档和查询关键词之间的相似程度
- 对于任一文档$d_j \in D$， D为文档数据集，可将其表示成如下n维向量的形式：$d_j = (w_{1j}, w_{2j}, \dots, w_{nj})$。其中$w_{tj}$为第$n$个特征词在文档$d_j$中的权重，$n$为文档$d_j$的总特征词数。则此时文档数据集$D$可以看做$n$维空间下的一定数量的向量集合，TF-IDF值也常常被选作为特征词的权重。而文档之间的相似度指两个文档内容相关程度的大小，当文档以向量来表示时，则可以使用文档内积或夹角余弦值来表示，两者夹角越小说明相似度越高
- 用VSM将文档表示成向量形式后，在基于向量的文本相似度计算中，常用的相似度计算方案有内积、Dice系数、Jaccard系数和夹角余弦值

向量空间模型

- VSM在文本检索处理中所具有的优势主要表现在以下几个方面
  - 对特征词的权重计算进行了改进，权重的计算通过对文本特征项的出现频次统计实现(TF-IDF值)，使问题的复杂性大为降低，改进了检索效果
  - 将文档简化为特征词及其权重集合的向量表示，对文档内容处理简化为VSM中向量运算
  - 根据文档和查询之间的相似度对检索结果进行排序，使对检索结果数量的控制与调整具有相当的弹性与自由度，有效地提高了检索效率



向量空间模型

- 向量空间模型理论也存在着一定的缺陷，主要包括以下几个方面
  - 各特征词之间的关系做了相互独立的前提假定，并且没有考虑词的顺序，这会失掉大量的文本结构信息，降低了语义的准确性
  - 相似度的计算量较大，当有新文档加入时， 必须重新计算特征词的权重
  - 在特征词权重的计算中，只考虑其出现次数等统计信息(如TF-IDF算法)，而以该信息来反映特征词的重要性，不免全面

词法分析

- 词法分析包括分词、词性标注、命名实体识别和词义消歧。其中命名实体识别是识别句子中的人名、地点、时间、产品名称和机构名称等命名实体，每个命名实体都是由一个或多个词语构成。词义消歧是根据句子上下文语境来判断某个词语的真实意思

文本分词

- 中文文本分词主要分为基于词典的分词方法、基于统计的分词方法和基于规则的分词方法
- 基于词典的分词方法
  - 最大匹配法、最小分词方法
  - 简单，分词效率高，但难以适应开放的大规模文本的分词处理
- 基于统计的分词
  - 实用性好
- 基于规则的分词
  - 较难，还在试验阶段

文本分词

- 常用的中文分词系统
  - SCWS-简单中文分词系统
  - IK Analyzer
  - ICCTCLAS
  - 结巴分词
  - 斯坦福分词
  - HanLP分词
  - 哈工大分词器
  - THULAC

文本分词

- 英文分词3S步骤
  - 根据空格拆分单词（Split）
  - 去除停用词（Stop word）
  - 提取词干（Stemming）

命名实体识别

- 命名实体识别(Named EntitiesRecognition, NER) 是自然语言处理的基础任务，目标是识别人名、地名和组织机构名等命名实体，一般包括实体类、时间类和数字类3大类，以及人名、地名、机构名、时间、日期、货币和百分比等7小类。由于这些命名实体数量不断增加，通常无法通过词典的方法识别，一般需要从词语形态或结构上进行分析传统命名实体识别算法是基于统计的方法，包括隐马尔可夫模型、最大熵(MaxmiumEntropy,ME)、支持向量机(SVM)、条件随机场(CRF)等
- 基于统计的方法对特征选取的要求较高，对语料库的依赖也比较大，需要从文本中选择对该项任务有影响的各种特征，面可用的大规模通用语料库又比较少
- 目前中英文通用命名实体识别(人名、地名、机构名)的F1值都能达到90%以上。命名实体识别的主要难点是表达不规律、缺乏训练语料的开放域命名实体识别，如电影、歌曲名、网名等

词义消歧

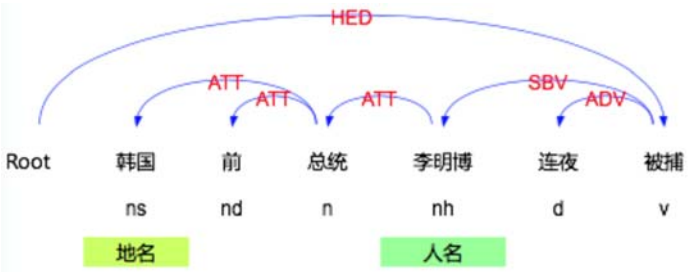
- 词义消歧(WSD)是自然语言理解中核心的问题，在词义、句义、篇章含义层次都会出现不同的上下文(Context) 下语义不同的现象。消歧就是根据上下文来确定对象的真实语义
- 词义消歧是信息检索、观点挖掘、文本理解和自然语言生成、推理等自然语言处理的基础模块
- 根据所使用的资源类型不同，可以将词义消歧方法分为以三类
  - 基于词典的词义消歧
  - 有监督词义消歧
  - 无监督和半监督词义消歧

句法分析

- 句法分析(Parsing) 是将输入句子从序列形式变成树状结构，从而可以捕捉到句子内部词语之间的搭配或者修饰关系。句法分析在机器翻译、问答、文本挖掘、信息检索等方面广泛应用。目前两种主流的句法分析方法是短语结构句法分析和依存结构句法体分析。其中后者已经成为研究句法分析的热点

依存结构句法体分析

- 依存句法(Dependency Parsing, DP) 认为句法结构本质上包含词和词之间的依存关系，依存关系是指词与词之间存在修饰关系。通过分析语言单位成分之间的依存关系揭示其句法结构，将输入的文本从序列形式转化为树状结构，从而刻画句子内部词语之间的句法关系



依存结构句法体分析

- 目前主要是数据驱动的依存句法分析，通过对大规模语料进行训练得到模型。这种方式生成的模型比较容易跨领域和语言环境。比较常见的是基于图(graph-based) 的分析方法和基于转移(transition- based) 的分析方法

短语结构句法分析

- 短语结构句法分析的研究基于上下文无关文法（CFG），CFG主要是对句子成分结构进行建模。一个CFG由一系列规则组成，每个规则给出了语言中的符号可被组织或排列的方法，以及符号和单词构成的字典
- 与CFG相关的一个操作就是语法解析（Parsing）。语法解析是指在给定词串的情况下，根据某种给定的形式文法分析并确定其语法结构的一种过程。例如“一只小猫抓老鼠”这句话，通过斯坦福大学的NLP开源框架Stanford CoreNLP来识别其语法结构。代码如下

```
from stanfordcorenlp import StanfordCoreNLP
nlp = StanfordCoreNLP(model_path,lang='zh',memory='8g')
sentence = '一只小猫抓老鼠'
pos_tag = nlp.pos_tag(sentence)
print("pos_tag:",pos_tag)
parse_result = nlp.parse(sentence)
print("parse:",parse_result)
```

短语结构句法分析

- 运行之后将结果打印输出，其中词性标记（pos_result）的结果：
- [('一', 'CD'), ('只', 'M'), ('小', 'JJ'), ('猫', 'NN'), ('抓', 'VV'), ('老鼠', 'NN')]
- 解析树的结果（parse_result）如下：

```
(ROOT
  (IP
    (NP
      (QP (CD 一)
        (CLP (M 只)))
      (ADJP (JJ 小))
      (NP (NN 猫)))
    (VP (VV 抓)
      (NP (NN 老鼠)))))
```

语义分析

- 语义分析的目标是理解句子表达的真实意思。但是语义表示形式尚未完全统一。 语义角色标注是比较成熟的浅层语义分析技术。给定句子中的个谓词， 语义角 色标注的任务就是从句子中标注出这个谓词的施事、受事、时间、地点等参数。语义角色标注一般都在句法分析的基础上完成，句法结构对于语义角色标注的性能至关重要
- 语义依存分析与句法依存分析的重要区别是语义依存分析不受句法结构的影响。语义依存关系分为三类:语义角色，每一种语义角色对应存在一个嵌套关系和反关系;事件关系，描述两个事件间的关系;语义依附标记，标记说话者语气等依附性信息

语义分析

- 语义依存与语义角色标注之间也存在关联，语义角色标注只关注句子主要谓词的论元以及谓词与论元之间的关系，而语义依存不仅关注谓词与论元的关系，还关注谓词之间、论元之间、论元内部的语义关系。论元是指句子中名词性的词或短语，并且与谓词搭配语义依存对句子语义信息的刻画更加完整全面
- 语义角色标注（SRL）通过标记与句子中谓词相关联的论元的角色，为每个语义角色被赋予一定的语义。语义角色标注系统已经处于NLP系统的末端，其精度和效率都受到前面分词、词性标记、句法分析模块的影响

语义分析

- 对“一只小猫抓老鼠”进行语义依存分析，只需要一行代码

```
dep =nlp.dependency_parse(sentence)
```

- 得到分析结果：

```
[('ROOT', 0, 5), ('nummod', 4, 1), ('mark:clf', 1, 2), ('amod', 4, 3), ('nsubj', 5, 4), ('dobj', 5, 6)]
```

- 其中的数字表示的是词语的序号，与分词列表进行匹配对应后重到结果如下：

```
nummod 猫 一
mark:clf 一 只
amod 猫 小
nsubj 抓 猫
dobj 抓 老鼠
```

语义分析

- 基于哈工大pyltp开源程序对句子进行语义角色标记

```
from pyltp import *
MODELDIR = "ltp_data_v3.4.0/"
self.segmentor = Segmentor()
m_path = os.path.join(MODELDIR,"cws.model")
dict_path = str(os.path.join(MODELDIR, "dict.txt"))
self.segmentor.load_with_lexicon(m_path,dict_path)
self.postagger = Postagger()
self.postagger.load(os.path.join(MODELDIR, "pos.model"))
self.parser = Parser()
self.parser.load(os.path.join(MODELDIR, "parser.model"))
#语义角色标注构建
self.labeller = SementicRoleLabeller()
self.labeller.load(os.path.join(MODELDIR, "pisrl.model"))
words = self.segmentor.segment(sentence)
wordlist = list(words)
postags = self.postagger.postag(words)
arcs = self.parser.parse(words, postags)
#语义角色标注结果
roles = self.labeller.label(words, postags, arcs)
```

文本分析应用

- 文本分析在实际应用中涉及的范围较广，本节主要介绍文本分类、信息抽取、问答系统、情感分析和内容摘要，在具体领域内的应用一般是上述技术的组合运用

文本分类

- 文本分类(Text Classification)的任务是根据给定文档的内容或主题，自动分配预先定义的类别标签。对文档进行分类，先将文本表示成特征向量形式，然后采用各种分类或聚类模型，通过训练分类器或进行聚类，实现文本分类
- 一般可以直接使用经典的模型或算法解决文本分类或聚类问题。例如，对于文本分类， 可以选用朴素贝叶斯、决策树、KNN、逻辑回归、支持向量机等分类模型。对于文本聚类，可以选用k-均值、层次聚类或谱聚类等聚类算法。但是，在算法应用过程中会面临很多与文本特征相关的问题，例如，文本的向量是稀疏的，以及文本具有定的序列特点， 并且同一语义对应多种表述方式等，这些问题都是构建文本分类模型所面临的关键问题
- 文本分类分为基于规则的分类模型、基于机器学习的分类模型和基于神经网络的模型等

基于Spark框架下逻辑回归方法进行文本分类

- 数据集来自AG新闻语料库，新闻类别包括全球、体育、商业、科技等类别
- 首先，引入Python语言的pyspark库及相应模块包，并使用Spark中SQLContext读取csv格式的训练文本，代码如下

```
from pyspark.sql import SQLContext
from pyspark import SparkContext
from pyspark.sql.functions import col
from pyspark.ml.feature import RegexTokenizer, StopWordsRemover, CountVectorizer
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import OneHotEncoder, StringIndexer, VectorAssembler
from pyspark.ml import Pipeline
from pyspark.ml.feature import HashingTF, IDF
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
sc =SparkContext()
sqlContext = SQLContext(sc)
data=sqlContext.read.format('com.databricks.spark.csv').options(header='true',inferschema='true').
load('train_text.csv')
data.show(10)
```

基于Spark框架下逻辑回归方法进行文本分类

- 每条新闻由类别（Category）、标题（Title）、内容详情(Detail)三部分组成
- 训练集的10条新闻内容

Category	Title	Detail
3	Wall St. Bears Cl...	Reuters - Short-s...
3	Carlyle Looks Tow...	Reuters - Private...
3	Oil and Economy C...	Reuters - Soaring...
3	Iraq Halts Oil Ex...	Reuters - Authori...
3	Oil prices soar t...	AFP - Tearaway wo...
3	Stocks End Up, Bu...	Reuters - Stocks ...
3	Money Funds Fell ...	AP - Assets of th...
3	Fed minutes show ...	USATODAY.com - Re...
3	Safety Net (Forbe...	"Forbes.com - Aft...
3	Wall St. Bears Cl...	NEW YORK (Reuter...

基于Spark框架下逻辑回归方法进行文本分类

- 在数据预处理时，首先应用正则化方法（RegexTokenizer方法实现）对详情字段分词，新增名为words的列，使用RegexTokenizer去掉words列中的停用词，增加名为filtered的列。采用TF-IDF提取新闻内容特征（作用于filtered列），其中词语在IDF最少要出现3次，输出的列名为features。用管道（Pipeline）按顺序执行前述的分词、去停用词、特征提取和类型转换等阶段（Stage），使用pipeline.fit()和pipeline.transform()方法执行各阶段（Stage）的原始DataFrame处理和转换。例如RegexTokenizer.transform()方法将实现分词，并增加words列到原始的dataframe上。HashingTF.transform()方法将单词列转化为特征向量，给dataframe增加一个带有特征向量的列

基于Spark框架下逻辑回归方法进行文本分类

```
regexTokenizer = RegexTokenizer(inputCol="Detail", outputCol="words", pattern="\\W")
add_stopwords=["this","that","rt","t","c","the","me","he","it","a","an","is","has","had"]
stopwordsRemover=StopWordsRemover(inputCol="words",outputCol="filtered")
    .setStopWords(add_stopwords)
label_stringIdx = StringIndexer(inputCol = "Category", outputCol = "label")
hashingTF = HashingTF(inputCol="filtered", outputCol="rawFeatures", numFeatures=100000)
idf = IDF(inputCol="rawFeatures", outputCol="features", minDocFreq=3)
pipeline = Pipeline(stages=[regexTokenizer, stopwordsRemover, hashingTF, idf, label_stringIdx])
pipelineFit = pipeline.fit(data)
dataset = pipelineFit.transform(data)
```

基于Spark框架下逻辑回归方法进行文本分类

- 生成的数据集dataset中，选择10条新闻记录进行显示，包括字段为类型、filtered列、详情特征列行标签

```
+--------+--------------------+--------------------+-----+
|Category|            filtered|            features|label|
+--------+--------------------+--------------------+-----+
|       3|[reuters, short, ...|(100000,[20625,24...|  2.0|
|       3|[reuters, private...|(100000,[8894,913...|  2.0|
|       3|[reuters, soaring...|(100000,[277,8273...|  2.0|
|       3|[reuters, authori...|(100000,[11637,20...|  2.0|
|       3|[afp, tearaway, w...|(100000,[2580,382...|  2.0|
|       3|[reuters, stocks,...|(100000,[3206,815...|  2.0|
|       3|[ap, assets, of, ...|(100000,[1466,322...|  2.0|
|       3|[usatoday, com, r...|(100000,[1466,109...|  2.0|
|       3|[forbes, com, aft...|(100000,[273,3223...|  2.0|
|       3|[new, york, reute...|(100000,[6039,160...|  2.0|
+--------+--------------------+--------------------+-----+
```

基于Spark框架下逻辑回归方法进行文本分类

- 将数据集按照7:3比例随机分为训练集和测试集，应用逻辑回归算法进行训练，最多训练迭代次数为20次。对模型lr训练后，使用测试集评估分类效果

```
(trainingData, testData) = dataset.randomSplit([0.7, 0.3], seed = 42)
# 训练模型
lr = LogisticRegression(maxIter=20, regParam=0.3, elasticNetParam=0)
lrModel = lr.fit(trainingData)
predictions = lrModel.transform(testData)
predictions.select("Detail","Category","probability","label","prediction")
    .show(n=10,truncate = 30)
```

```
+------------------------------+--------+------------------------------+-----+----------+
|                        Detail|Category|                   probability|label|prediction|
+------------------------------+--------+------------------------------+-----+----------+
|A spate of bombings which k...|       1|[0.8550385207988529,0.04817...|  0.0|       0.0|
|Moscow - Recorders from the...|       1|[0.5035406814668905,0.11417...|  0.0|       0.0|
|A German court acquitted th...|       1|[0.7916381408919737,0.08029...|  0.0|       0.0|
|Thousands of Bangladesh opp...|       1|[0.8587088308896439,0.04190...|  0.0|       0.0|
|"AFP - Australia's Greens -...|       1|[0.8043426140228166,0.03678...|  0.0|       0.0|
|"LOS ANGELES - Movie-goers ...|       1|[0.08343286087334675,0.1057...|  0.0|       3.0|
|Scientists develop an elect...|       1|[0.131141873568267,0.156367...|  0.0|       3.0|
|"SPRINGFIELD, Ore. - While ...|       1|[0.21580455859343545,0.4033...|  0.0|       1.0|
|Baroness Thatcher returns h...|       1|[0.591110723850574,0.137972...|  0.0|       0.0|
|Shiite Muslim cleric Moqtad...|       1|[0.8441148817422035,0.03901...|  0.0|       0.0|
+------------------------------+--------+------------------------------+-----+----------+
```

基于Spark框架下逻辑回归方法进行文本分类

- 应用多类别分类评估器（MulticlassClassificationEvaluator）对分类效果评估，代码如下。

```
eval = MulticlassClassificationEvaluator(predictionCol="prediction")
print("accuracy:",eval.evaluate(predictions,{eval.metricName:"accuracy"}))
```

- 将评估结果输出
- accuracy: 0.9037890353920889

信息抽取

- 信息抽取(Information Extraction) 是指从文本中提取指定类型的信息，如实体、属性、关系、事件等，并通过信息归并、冗余消除和冲突消解等手段将非结构化文本转换为结构化信息
- 每一句文本所蕴含的意思可以描述为其中实体之间的关联，因此文本实体和及其之间的语义关系也就成为理解文本意义的基础，信息抽取可以通过抽取实体之间的语义关系，或其语义角色关系，并基于这些信息进行计算和推理，有效理解文本的语义
- 关系抽取是识别文本中实体之间的语义关系，关系抽取的输出通常是一个三元组(实体A，关系类别，实体B)来表示实体A和实体B之间存在特定类别的语义关系
- 事件抽取是从非结构化文本中抽取事件

问答系统

- 自动问答(Question Answering, QA) 是指计算机自动回答用户所提出的问题。问答系统是信息服务的一种高级形式， 不同于现有搜索引擎，系统返回用户的结果不再是基于关键同匹配排序的文档列表，而是精准的自然语言答案
- 问答系统在回答用户问题时，首先需要正确理解用户所提的自然语言问题，并抽取其中的关键语义信息，然后在已有语料库、知识库或问答库中通过检索、匹配、推理的手段获取答案并返回给用户。上述过程涉及词法分析、句法分析、语义分析、信息检索、推理、知识工程、内容生成等多项关键技术
- 自动问答的主要研究问题包括
  - 问句理解
  - 文本信息抽取
  - 知识推理

情感分析

- 情感分析，也称为意见挖掘或观点挖掘。是对带有感情色彩的主观性文本进行分析、处理、归纳和推理的过程。一般是在句子级别或段落级别上进行极性分类，判断出此文字中表述的观点是积极的、消极的、还是中性的情绪。更高级的情感分析情绪状态较丰富，例如喜欢、高兴、惊讶、愤怒、伤心、害怕等
- 情感分析应用广泛，可应用于舆情监控、信息预测、产品评价等，作为文本处理的基础模块可以用于自然语言生成，例如对选择或生成的句子进行一次情感检测， 防止生成消极的文章
- 文本情感分析方法有基于词典的方法、机器学习方法、概念级技术等几类

基于机器学习的文本情感分析-SnowNLP

- SnowNLP是一个Python编写的开源情感分析程序，设计思路与英文情感分析框架TextBlob相似，不同之处在于SnowNLP可以处理中文并且不需要依赖NLTK等第三方库，本质上是贝叶斯分类。通过pip3 install snownlp就可以安装，其调用方法如下代码所示

```
from snownlp import SnowNLP
s = SnowNLP('这个件衣服看起来很漂亮')
print(s.words)
print(s.sentiments)
输出结果为:
['这个', '件', '衣服', '看', '起来', '很', '漂亮']
0.959781206245387
```

自动摘要

- 自动文摘是指通过自动分析给定的篇或多篇文档，提炼，总结其中的要点信息，最终输出篇长度较短、可读性良好的摘要。该摘要中的句子可直接出自原文，也可利用内容生成技术生成，即通过对原文本进行压缩、提炼，生成简明扼要的文字描述
- 自动文摘方法分为抽取式摘要和生成式摘要。抽取式方法相对比较简单，通常利用不同方法对文档结构单元(句子、段落等)进行评价，对每个结构单元赋予一定权重，然后选择最重要的结构单元组成摘要。而生成式方法通常需要利用自然语言理解技术对文本进行语法、语义分析，对信息进行融合，利用自然语言生成技术生成新的摘要句子。目前的自动文摘方法主要基于抽取式方法，其优点是易于实现，能保证摘要中的每个句子具有良好的可读性

自动摘要

- 也可以利用拓展新将强的贝叶斯话题模型，对话题相关性概率进行建模。加权频数的定义可以有多种，如信息检索中常用的TF- IDF权重。还可以利用隐语义分析(LDA)得到低维隐含语义表示并加以利用。在多文档摘要任务中，重要的句子可能和更多其他句子较为相似，所以可以用相似度作为节点之间的边权，通过迭代求解基于图的排序算法来得到句子的重要性得分
- 另一方面，目前已经具备一定数量的公开摘要标记数据集， 可用于监督训练模型。 例如利用回归模型或排序学习模型进行有监督学习，得到句子或概念对应的得分。因为文档内容描述具有结构性，因此也可用隐马尔科夫模型(HMD)、 条件随机场(CRF)、 结构化支持向量机(Structural SVM)等算法进行监督训练。对应的特征包括所在位置、包含词汇、与邻句的相似度等。内容选择方法包括贪心选择和全局优化

知识图谱

- 知识图谱是结构化的语义知识库，用于以符号形式描述物理世界中的概念及其相互关系，由实体之间通过关系相互连接，构成网状的知识结构
- 知识图谱的目标是为了让机器能够理解文本背后的含义。为此，需要对可描述的事物(实体)进行建模，填充它的属性，拓展它和其他实体的联系，即构建机器的先验知识。此外，还涉及知识提取、表达、存储和检索一系列技术
- 知识图谱首先是由Google于2012 年提出，目的是为了提升搜索结果的质量和提高检索效率，有知识图谱作为辅助，搜索引擎能够理解用户查询背后的语义信息，获取字符串背后隐含的对象或事物，这样返回的结果更为精准。此后，各个机构也开始着手打造各种知识库，比较知名的有DBPedia、NELL、OpenIE、Freebase、Google KG、BabeNet、WordNet和Yago等

知识图谱

- 知识图谱的应用非常广泛，特别适合于智能客服、金融、公安、航空和医疗等“知识密集型”领域
- 很多金融公司构建了金融知识库对金融知识进行集成与管理，并辅助金融专家进行风控控制和欺诈识别等
- 生物医学专家通过集成和分析大规模的生物医学知识图谱，辅助其进行药物发现
- 在公安领城中，对人员、位置、事件和社交关系等信息应用知识图请可以及时发现热点事件的发展、传播与关键点，提早做出感知和识别，从而实现预防犯罪

知识图谱相关概念

- 本体这个术语来自哲学概念，用于描述实体和实体间的关系。例如，“描述”都是“本体”的外在符号，人们所看到的图像、说的语言、对事物的感觉都是符号到本体的某种映射，所以它只可意会不可言传。在信息科学中的本体指的是语义层面的意思。在人工智能领域中，本体就是用详细的描述方法定义出来的概念或者概念体系，建立本体的过程就是一个定义概念的过程
- Tom Gruber把本体定义为概念及其关系的形式化描述。本体类似于数据库中的表结构，主要用来定义类和关系，以及类层次和关系层次等。最常用的本体描述语言有RDF和网络本体语言(Ontology Web Language, OWL)等， 可以用于定义同义词、反义词，以及对属性的值域施加约束等。本体通常被用来为知识图谱定义图谱结构，个本体库是由类、属性和实例组成，在OWL里统称为实体(entity)

知识图谱相关概念

- 知识库(Knowledge Base)是人工智能的经典概念之一。 最早作为专家系统(Expert System)的组成部分，用于实现决策推理。知识库中的知识有很多种不同的形式，例如本体知识、关联性知识、规则库和案例知识等
- 链接数据(Linked Data)是由Tim Berners Lee 于2006年提出，为了强调语义互联网的目的建立数据之间的链接，而非仅仅把结构化的数据发布到网上。链接数据最接近于知识图谱的概念
- 语义网络(Semantic Network) 最早是1960年由认知科学家Allan M. Collins 作为知识表示的一种方法提出。其中WordNet是最典型的语义网络。与知识图谱相比，早期的语义网络更加侧重描述概念及其之间的关系，而知识图谱更加强调数据或事物之间的链接

知识图谱的存储

- 按照存储方式的不同，知识图谱的存储可分为基于表结构的存储和基于图结构的存储。
- 基于表结构的存储采用二维数据表的方式存储数据，例如三元组表、类型表以及关系数据库
- 基于图结构的存储可以使用图数据库

知识图谱挖掘与计算

- 知识图谱的挖掘主要是基于图运算的理论，从海量结点中寻找权威节点（重要节点），与目标节点最近（路径最短）且最权威的节点
- 最短路径路径算法有Dijkstra算法和Floyed算法
- Dijkstra算法步骤：
  - 初始时，S只包含原点v,距离为0。用U表示与S对立的顶点集合。
  - 从U中选取一个距离v最小的顶点k，把k加入S集合。
  - 以k为另一个原点，对U中每个顶点修改到原点的最短距离，若到k的距离小于到v的距离，则将原有的距离修改为更小的值。
  - 重复2、3步骤，直到所有顶点都加入S集合

- Floyd是一种动态规划算法，用于求每对顶点之间的最短路径，其算法步骤如下：
  - 初始化某一个图的邻接矩阵D，矩阵中每个元素为$d_{ij}^{(0)}$有：

$$d_{ij}^{(0)} = \begin{cases} l_{ij}，i和j连通 \\ 0，i = j \\ \infty，i和j不连通 \end{cases}$$

  - 在原路径中增加一个新节点，如果产生的新路径比原路径短，则将原路径值修改为更小值，这样会依次产生多个矩阵，$D^{(1)}$，$D^{(2)}$，…，$D^{(n)}$，对第k个矩阵$D^{(k)} = \left\{d_{ij}^{(k)}\right\}$

$$d_{ij}^{(k)} = min\left\{d_{ij}^{(k-1)}，d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right\}$$

k从1开始进行循环，i，j分别从1到n开始遍历，直到k等于n结束

知识图谱挖掘与计算

- **权威结点分析**是从知识图谱中分析结点的权威性，从中发现权威结点。权威结点分析常用于社交网络权威人物或权威机构的发现。权威结点分析主要采用互投票方法的方式，其思想来源于PageRank思想
- PageRank是指被越多的优质网页所指向的网页，具有更高的优质概率。如果两个网页存在链接指向，说明这两个网页是存在关联，因此可采用一个相关性的参数来衡量。页面的质量是一个累计值，由所有指向此页面的链接通过递归算法计算得到。一 个页面拥有越多的被指向页面，那么它的优质度就更高，反之，网页优质度就越低
- 如果知识图谱的数据量非常庞大，为了降低算法开销，可采用分块式的方式来实现算法，先计算每个分块图的PageRank,根据各数据块之间的相关性，得到新图的PageRank, 再反复迭代，分析权威节点
- 权威节点分析还可采用基于结点属性及结点间关系的多特征方法，将节点属性和关系综合分析

知识图谱挖掘与计算

- 相似节点发现是指从知识图谱海量节结中，寻找与已知节点相似的结点，可基于结点进行属性计算以及关系计算。常常应用于企业寻找潜在客户、专利检索等
- 假定一个无向图G= (V,E,M)，V中结点总数为N，$M = \{a_1, .., a_m\}$，其中a_i是结点关联属性的m个取值。在原始图G中加入属性结点和属性边构造属性扩展图，针对属性扩展图G'=(V', E',M),使用基于结构情境的相似度计算方法,计算每个结构结点的结构相似度，属性边的加入会使得具有同一属性的结点之间的相似度增大，对于每个属性结点，计算其到所有与之相连的结构结点的转移概率，并将此转移概率与结点的结构相似度相结合计算出最终的结点相似度，最后使用改进的K-means聚类算法在结点相似度的基础上对结点进行聚类，求得最终结构。其中聚类初始中心点的选取遵循最大最小原则。具体步骤如下：

- 当V不为空时，读入原始图G=（V，E，M），根据属性与结点的从属关系构造属性扩展图G'=(V', E',M)，得到G'中结点的邻接矩阵；
- 计算得到属性扩展图G'中结点的结构相似度矩阵；
- 当属性结点与其他它结点相邻时，用属性结点的转移概率矩阵更新结点的结构相似度，得到结点相似度矩阵；
- 使用改进的k-means算法对图G'中的结点聚类，根据最大最小原则确定K个初始聚类中心点，即首先选择度数最大的结点k加入初始簇心中，然后采用最小相似度原则将与已入簇结点的平均相似度s最小的结点作为新的初始聚类中心点加入初始簇心集合。其中，为了排除孤立点的影响，选取结点度数平均值作为候选簇心的筛选阀值；
- 更新簇，将除初始聚类中心点以外的其它结点划分到相似度较高的簇心所在的簇中，确定此次迭代后结构；
- 更新簇心集合。计算结点在已划分好的图内部的度数，将度数最大的结点作为新的簇心。若新的簇心与上次迭代的簇心不同，则重复步骤5；若相同，则停止迭代，输出最终结构。

知识图谱的构建过程

- 知识图谱的逻辑结构分为数据层和模式层。数据层为知识图谱的数据存储，模式层在数据层之上，存储的是经过提炼的知识，通常采用本体库来管理知识图谱的模式层，规范实体、关系以及实体的类型和属性等对象之间的联系。本体库在知识图谱中相当于模具，拥有本体库的知识库一般冗余知识会比较少
- 技术结构方面，知识图谱的构建过程包括知识获取、数据融合，其中知识获取是从非结构化、半结构化以及结构化数据中获取知识，并对知识进行分词、词性标注、提取实体、语义消歧和关系挖掘等基本文本处理。数据融合将不同来源的知识进行融合构建数据之间的关联

知识图谱的构建过程

- 知识图谱有自顶向下和自成向上两种构建方式。自顶向下的方式需要专家手工编辑形成数据模式，而自底向上的构建，则是借助一定的技术手段，基于行业现有标准，从公开采集的数据中提取出资源模式进行映射，选择其中置信度较高的新模式，经人工审核之后，加入到知识库中。这个过程需要随时间不断更新循环，根据知识获取的逻辑，这步骤包含三个阶段：信息抽取、知识融合以及知识加工

知识图谱的构建过程

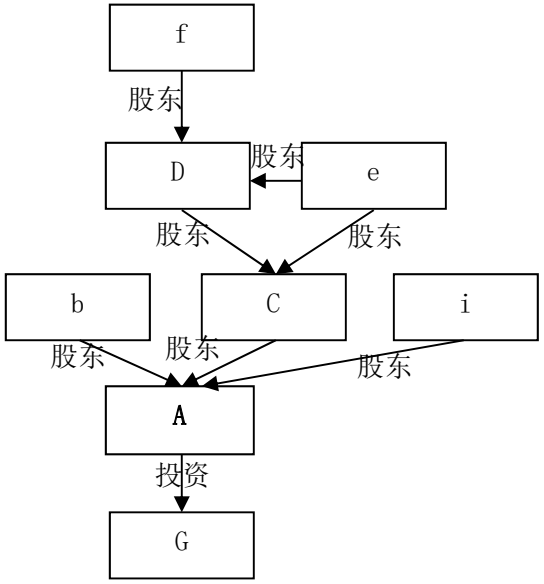
- 下面以电商领域知识图谱构建为例，介绍知识图谱的一般构建过程。
  - （1） 确定领域本体，一个本体描述的是一个特定的领域。例如要描述的领域是“电商”。
  - （2）列举领域内的术语集合，指定领域中的一组重要概念。例如，要描述“电商”这个领域，可以列举出“商品”、“卖家”、“买家”、“厂家”等概念。
  - （3）确认基本术语之间的关系，包括分类、类间层次结构和属性等。即确定概念之后，再确定这些概念之间的关系，例如并列关系、包含关系和关联关系等，“平台”与“卖家”是包含关系。
  - （4）添加约束规则，包括属性约束（例如商品品牌、大小和重量等）、值约束（例如，只有卖家才可以发布商品）等。
  - （5）定义实例，将具体的实例信息导入到之前建立的结构中，形成知识库
  - （6）检查和验证，通过对本体自身的不一致和置入本体的实例集进行一致性检查

知识图谱的构建过程

- 下面以电商领域知识图谱构建为例，介绍知识图谱的一般构建过程。
  - （1） 确定领域本体，一个本体描述的是一个特定的领域。例如要描述的领域是“电商”。
  - （2）列举领域内的术语集合，指定领域中的一组重要概念。例如，要描述“电商”这个领域，可以列举出“商品”、“卖家”、“买家”、“厂家”等概念。
  - （3）确认基本术语之间的关系，包括分类、类间层次结构和属性等。即确定概念之后，再确定这些概念之间的关系，例如并列关系、包含关系和关联关系等，“平台”与“卖家”是包含关系。
  - （4）添加约束规则，包括属性约束（例如商品品牌、大小和重量等）、值约束（例如，只有卖家才可以发布商品）等。
  - （5）定义实例，将具体的实例信息导入到之前建立的结构中，形成知识库
  - （6）检查和验证，通过对本体自身的不一致和置入本体的实例集进行一致性检查

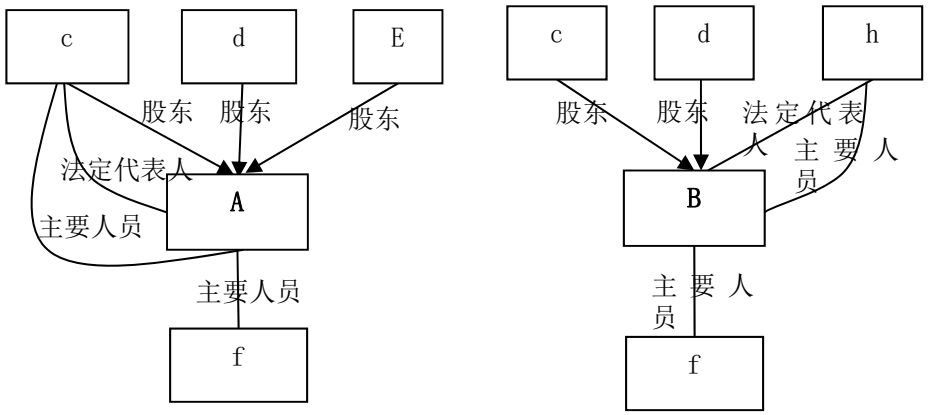
基于知识图谱的企业信息查询平台实现

- 企业A关联族谱的数据结构



基于知识图谱的企业信息查询平台实现

- A、B两家公司的直接人员关联节点图

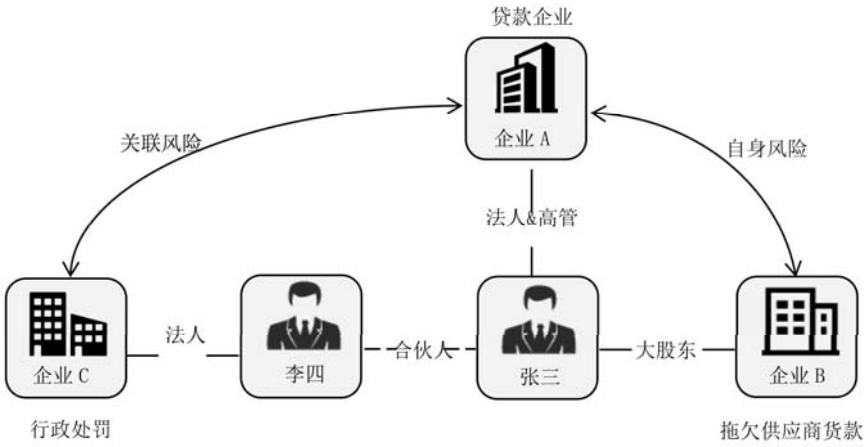


基于知识图谱的企业信息查询平台实现

- 从原始的工商信息中，公司A、B没有任何关联。计算A、B公司是否有疑似关联可以通过下面的多特征维度识别去重算法：
  - 将公司A的所有直接相邻节点去掉重复名称后得到数组A{c, d, E, f}
  - 将公司B的所有直接相邻节点去掉重复名称后得到数组B{c, d, f, h}
  - 循环数组B中的元素，将B中的元素添加进A中，如果遇到添加失败返回false，则将当前元素添加进临时数组temp中
  - 统计temp数组中元素数量，定义相似度衡量的阀值
  - 大于设定的阀值表示两家公司有疑似关联，否则没有关联

基于知识图谱的企业信息查询平台实现

- 企业A关联风险分析

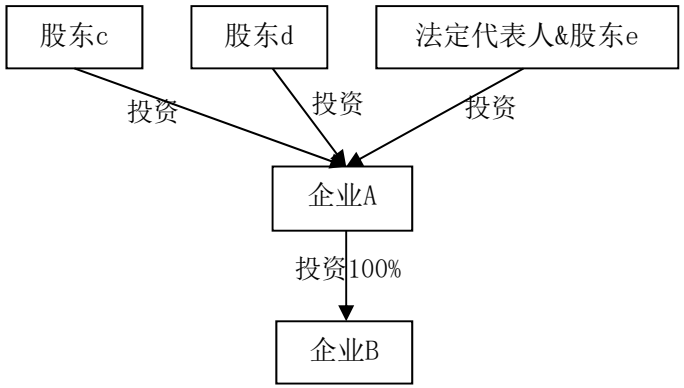


基于知识图谱的企业信息查询平台实现

- 针对目标公司的企业关联风险有4个指标衡量：
  - 通过PageRank计算出来的每家企业的PR值。
  - 受到直接影响的企业与目标评估企业相互关联层级数，第一层影响大于第二层，第二层影响大于第三层，依次递减。
  - 受到直接影响的企业对目标评估企业的持股百分比，对目标企业持股比例越大，对目标企业影响就越大。
  - 对影响事件进行风险评估分类。根据人员种类分析对公司影响，如法定代表人、股东、主要人员、普通员工。针对影响事件划分不同影响等级，如行政处罚、经营异常、失信事件、被执行人事件、法院公告等。针对正负新闻舆情进行影响等级分类。越重要的人物对企业影响越大，越重要的事件对企业影响越大，负面新闻对公司影响大。
- 经过这4个指标的综合衡量得到的风险影响因素，将对风险划分为5个等级，越大的数字表风险依次递增。其中，1为无风险，5为最严重风险，将这些风险分析后并最终显示给用户

基于知识图谱的企业信息查询平台实现

- 企业投资关系路径分析



基于知识图谱的企业信息查询平台实现

- 企业知识图谱数据存储及使用
- 在企业相关的数据维度中，以工商信息中的数据作为企业知识图谱的基础来源。工商信息主要包括工商照面信息(Company)、股东信息(Partner)、人员信息(Employee)、分支机构(Branch)和历史变更记录(Change)等
- 实体和关系在提取后，选择免费开源的图数据库Neo4j作为关联关系存储的数据库。作为一个高性能的图形数据库，Neo4j 将结构化好的数据存储在网络上而不是关系表中，基于图的搜索，具有完全事务管理功能，能很好支撑动态数据特性的应用需求。利用Neo4j 提供的Cyhper语法，开发人员可以专注业务场景，而直接使用自带的图挖掘算法


THANK
YOU