最近因为项目需要，加上个人兴趣，就找了几篇有关自动文本摘要的综述论文，简单做了些调研、学习，这里做个小结。

# 0、概述

自动文本摘要，顾名思义，就是自动的从文本中生成摘要。

其实这方面的研究最早在1950年代就开始了，但一直得不到大众的关注，在国际上尚且如此，在国内的情况就更加尴尬了。基于中文文本的自动摘要，在国内能找到的资料、工具并不多。

# 1、应用场景

自动文本摘要能用在哪些地方呢？从个人习惯来说，我倾向于把日常接触到的文字内容分为“水货”和“干货”两种。

“水货”是指哪些呢？就是那些看完之后没有太多实际收获的内容，比如很多公众号里的鸡汤文、鸡血文、狗血文等等，对于很多人而言，小说类侧重故事性的文章，也都是“水货”。总的来说，我们发现，其实自动文本摘要并不适用于这一类内容，因为读者接触这些内容是为了来体验、来感受的，而不是抱着功利心来的。

自动文本摘要主要还是应用于“干货”上，新闻文章、科学文献、股市研报等等，人们从这些内容中，想要的是新的信息收获。

那沿着这个思路，我们就可以想到自动摘要的应用场景都有哪些了。

用在新闻文章上，我们可以看到新闻快报、头条，像我几乎每天上班路上都会打开的虎嗅早报、36氪早报，以及微博上的新闻推送，其实是能通过自动摘要来生成的。

用在科学文献上，我们更快速的看论文，涨知识。

用在股市研报上，我们看一眼就能知道A股又要跌多少。

这是第一层次的思维，我们不妨进一步的想象：如果同时用在多篇新闻上，或者多篇论文上，或者多篇研报上，我们能看到些什么？

用在特定事件的多篇新闻报道上，我们能更完整的看到事件的全貌。

用在特定领域的论文上，我们能看到这个领域的分支、进展。

用在股市研报上，我们能看到市场对特定股票的态度。

# 2、任务拆解

从数据源来看，我们发现自动文本摘要可以分为两大块方向：单文档摘要和多文档摘要。

可能有些同学会想，多文档的情况下，直接拼在一起然后用单文档摘要的技术去处理不就可以了吗？确实这是一种思路，但这种思路存在的问题是，虽然有着相似的主题，但不同文档的侧重点不同，而如果直接当做一篇文档来处理，可能会去掉一些相对小众的主题信息。

从结果来看，自动摘要还能分为两大块方向：抽取式(extraction)和生成式(abstraction)。

抽取式指的是从原始文本中抽取关键的词句，来组成一篇摘要；而生成式则是从原始文本中通过一些较为高级、复杂的技术，生成出一篇摘要来，会出现一些原文中完全没出现过的词句。

简单思考后，我们可以发现，抽取式可能更适用于长文本，因为长文本中主题信息分布得比较散，也能有比较多的句子可以用来抽取。然而当面对短文本时，比如两三百词的文本，要做成一两句话的摘要，抽取式可能就很难有较好的效果了，因为这两三百词可能是由八九个句子组成的，而这八九个句子每个都有不同的重要信息，简单从中抽取一两个句子，肯定会有信息遗漏。

不难想到，单文档摘要比多文档摘要简单一些，抽取式比生成式要简单一些。也正因如此，很多人在写文本摘要的算法博客以及工业实用时，都只讲TextRank，因为这是最好实现的抽取式文本摘要，而且后面我们会提到，这个算法连训练集都不需要！

# 3、效果评估

在展开来谈算法之前，有必要先提一下自动摘要的效果评估方式。一方面是没有效果评估就没法谈优化、没法谈项目贡献，在公司中，这样的项目可能根本就无法通过立项；另一方面也是怕同学们在看了后面一大堆算法之后就不想看这块内容了（笑）。

最简单粗暴的方式当然就是人工评估啦。这没太多可说的，打分呗。一个分不够就多维度综合打分呗。

但人力成本那么高，自然还是得想办法出一些指标来自动评估了。

在对自动摘要进行评估时，我们主要是在想两个问题：

1、对于文档D来说，摘要A是不是足够好?

足够好意味着：摘要应该包含原文尽可能多的信息，且尽可能短，而且读起来通顺。

于是我们可以针对信息量给出一个打分，针对长度给出一个打分，针对可读性给出一个打分。假如三个打分都达到了我们所要的水平，就能认为特定摘要足够好。

2、同样基于文档D，摘要A是不是比摘要B好?

我们拿到了信息量、长度、可读性三个打分，然后在判断对两篇摘要孰优孰劣的时候，需要将这些打分通过某种方式整合成一个打分，才能做比较。

这就意味着，我们需要在信息量、长度、可读性之间做一些权衡，也就是计算机学科里常说的 tradeoff 。

如何基于信息量打分、如何基于长度打分、如何基于可读性打分、如何权衡三个打分，对于这些问题给出不同解决方案，就是不同评估方式之间的本质区别。

## ROUGE

目前比较常用的是ROUGE类的指标，ROUGE=Recall-Oriented Understudy for Gisting Evaluation，字面意思是：用于要点评估的召回导向的替代（原谅我翻译渣...）。

我们假设有一组参考摘要 $R = \{r_1, r_2, \ldots, r_m\}$； $s$ 作为自动生成的摘要； $\Phi_n(d)$ 表示特定文档的n-gram 0-1向量，长度为所有可能的n-gram数量。

于是我们常见的**ROUGE-N**是这样定义的：

$$ROUGE_n(s) = \frac{\sum_{r \in R}(\Phi_n(r) * \Phi_n(s))}{\sum_{r \in R}(\Phi_n(r) * \Phi_n(r))}$$

简单来说，就是(重叠的N-gram数)/(参考摘要中的N-gram数)。好了，熟悉机器学习的同学们请坐下，这确实就是Recall。

ROUGE-N是基于N-grams来计算的，后来有人想出来用最长共同子串（LCS），同时改Recall为F-measure，这就是**ROUGE-L**：

$$ROUGE_L(s) = \frac{(1+\beta^2)R_{LCS}P_{LCS}}{R_{LCS} + \beta^2 P_{LCS}}$$

其中：

$$R_{LCS}(s) = \frac{\sum LCS(r_i, s)}{\sum |r_i|}; P_{LCS}(s) = \frac{\sum LCS(r_i, s)}{|s|}$$

**ROUGE-W**: 从ROUGE-L改进而来，加了权重。

**ROUGE-S**: Skip-bigram based co-occurrence statistics. Skip-bigram is any pair of words in their sentence order.

**ROUGE-SU**: Skip-bigram plus unigram-based co-occurrence statistics.

# 信息论

这一类评估方法，主要思想是看两个分布的散度。主要有两种：

**KL divergence**

$$KL(p^{\theta_A}||p^{\theta_R}) = \sum_{i=1}^{m} p_i^{\theta_A} log \frac{p_i^{\theta_A}}{p_i^{\theta_R}}$$

**Jensen-Shannon divergence**

$$JS(p^{\theta_A}||p^{\theta_R}) = (KL(p^{\theta_A}||r) + KL(p^{\theta_A}||r))/2 = H(r) - H(p^{\theta_A})/2 - H(p^{\theta_R})/2$$

# 4、算法

接下来就是最核心的部分了——算法。

正如前面所说的，自动摘要主要分抽取式和生成式两种。基于这两种方式，算法难度上有很大的不同。

## 4.1 抽取式

抽取式自动摘要的核心思路是，对文档中的所有句子打分，最终挑出若干个权重高的句子来组成摘要。

最简单粗暴（往往也很靠谱）的做法自然是根据人为规则来挑选句子了，这也是1950年代人们刚开始做自动摘要的思路——利用一些简单的统计特征，例如：句子所包含的单词或短语的数量，句子在文档中的位置，句子是否包含重要单词或短语，句子跟文档标题的词语重叠程度等等，来衡量不同句子的重要性，进而组成摘要。

当然了，随着发展，算法变得越来越“高级”，效果也越来越好。

抽取式自动摘要存在两个重要的基础技术：

1、句子的特征向量表示

常见的有以下几种：

基于统计的：我们用一个长度跟语料库（假设$D$由若干个句子组成$s_1, s_2, \ldots, s_M$）相等的向量$W = [w_1, w_2, \ldots, w_N]$来表示，这个向量中的一个值对应语料库中一个词的重要性。这个值可以是简单的0或1，表示该词语是否存在于特定句子中。又或者是更常见的，TFIDF值：

$$TFIDF(s, w) = TF(s, w) * IDF(D, w) = (\frac{count(w \in s)}{len(s)}) * log(\frac{len(D)}{len(D|w \in D) + 1})$$

TF表示词语在该句子中出现的频率；IDF则是通过看该词语是否在很多句子里都存在来评估词语是否“过于常见”。

基于潜在语义的：这种出来的特征向量长度不一，基本是可以人为控制的，旨在找到隐藏在词句背后的潜在语义特征，比如LDA，word2vec等等，一般缺少明确的可解释性。

LDA是在构建了TFIDF矩阵后，运用SVD将其拆分成是三个子矩阵，并将其中两个子矩阵提出来相乘得到句子-主题的一个向量。

word2vec则是通过神经网络进行学习而得。

当然了，你也可以把上面几种特征向量都拼在一起来用。

2、句子间的相似度计算

在有了两个句子的特征向量后，我们需要基于这两个向量来做相似度计算。

最常用的自然是余弦相似度了：

$$CosSim(s_1, s_2) = \frac{s_1 * s_2}{|s_1| * |s_2|}$$

也就是计算这两个向量间角度的余弦值。

偶尔可能还会用到Jaccard系数：

$$J(v_i, v_j) = \frac{\sum_k min(v_{ik}, v_{jk})}{\sum_k max(v_{ik}, v_{jk})}$$

再其他的就很少见了。

## 4.1.1 图

大名鼎鼎的TextRank和LexRank就是这一类算法。主要都是参考了PageRank的思想。

**TextRank**的好处太多：易于实现、不需要训练集、效果还不错，总而言之，特别适合伸手党。

就PageRank具体而言，我们先构造一个图 $G = \{V, E\}$ 。把网页作为一个点，而如果 $V_j$ 后跟随着 $V_i$ ，我们则认为存在一条 $V_j => V_i$ 的边。先对图中所有的点赋初始值，然后按照以下公式持续迭代：

$$WS(V_i) = (1-d) + d * \sum_{V_j \in IN(V_i)} \frac{w_{ji}}{\sum_{V_k \in OUT(V_j)} w_{jk}} * WS(V_j)$$

$d$ 称为阻尼系数，一般取0.85。从迭代的公式中，可以看到，对于特定一个点，分值取决于指向它的其它点的分值，该公式既考虑了这个点有多少入边，还考虑了指向它的点本身有多少出边。

TextRank基本就是把PageRank直接搬了过来：将句子作为点，取所有的点构造全连接图，两个点之间的边的权重则用句子间相似度来计算。

**LexRank**与TextRank基本相似。（好吧，暂时查不到这两者的区别）

### 4.1.2 聚类

这个方向的思路是：在获取了句子的特征向量之后进行聚类，从而把文章能分成几块主题。最后只要从每个主题里面提取一些句子出来，组成摘要就可以了。

聚类的通用算法自然可以往上套了，比如K-Means，OBSCAN等等。

### 4.1.3 分类

另外一种思路是通过训练一个分类模型来判断特定句子是否应该放在摘要中。既然转成了二分类问题，那所有分类模型都能往上面套了。

比如贝叶斯、决策树、SVM、HMM、CRF、神经网络等等。

### 4.1.4 深层次自然语言分析

除了机器学习，有一些人在尝试着通过一些较复杂的自然语言分析处理技术来进行文本摘要。总的来说，人们尝试对文本进行解构。

整体的流程是：分割文档、识别词汇链、通过强词汇链来找到重要语句。

### 4.1.5 基于群体智能

太高级了，没看懂……

particle swarm optimization

cuckoo optimization

bacterial foraging optimization

## 4.2 生成式

生成式文本摘要 https://zhuanlan.zhihu.com/p/30559757

1、基于RNN

A Deep Reinforced Model for Abstractive Summarization

目前最好的基于RNN的Seq2Seq生成式文本摘要模型之一来自Salesforce，在基本的模型架构上，使用了注意力机制（attention mechanism）和强化学习（reinforcement learning）。

2、基于CNN

Convolutional Sequence to Sequence Learning

基于卷积神经网络的自动文本摘要模型中最具代表性的是由Facebook提出的ConvS2S模型，它的编码器和解码器都由CNN实现，同时也加入了注意力机制

http://blog.ilibrary.me/2017/05/15/sumy-textsum%E5%92%8Cfairsqe

sumy https://github.com/miso-belica/sumy

tensorflow-textsum https://github.com/tensorflow/models/tree/master/research/textsum

# 5、中文文本

关于中文自然语言处理，HanLP 是个相当全面的工具，但这是用Java开发的，虽然有提供Python接口（pyhanlp），但不好改动（太久没用Java，懒得重温...），所以后来我主要还是用 TextRank4ZH，粗略来看效果也略好一些。

# 6、结语

通过这段时间的了解，发现这块领域还是很有意思的，但现成的工具比较少，主要也就TextRank了。

就个人而言，对生成式自动摘要的兴趣更强一些，因为目前项目涉及到的主要是短文本。

# 7、引用

综述：

Text Summarization Techniques: A Brief Survey

A survey on Automatic Text Summarization

A Survey on Automatic Text Summarization, Dipanjan Das

A survey on Automatic Text Summarization, N.Nazari

生成式自动摘要：

A Deep Reinforced Model for Abstractive Summarization

Convolutional Sequence to Sequence Learning