**用深度学习解决大规模文本分类问题**

**摘要：**近年来，深度学习在图像和语音处理领域已经取得显著成就。本文基于深度学习的文本分类技术为研究背景，介绍其分类的过程，文本的分布式表示，以及介绍了几种基于深度学习神经网络文本分类的模型。是研究结果表示，在中英文不用的测试数据集上，基于深度学习的文本分类方法，均可以得到83%以上的准确率，具有很好的分类效果。

关键词：文本分类，自然语言处理，深度学习。

**Abstract:**Recently deep learning has made great development in the fields of image and voice processing.This paper takes deep learning based text classification techniques as the research background to introduce its classification process, text distributed representation, text classification network models based on the deep learning neural network. The experiments show that, for both Chinese and English data sets, deep learning based text classification method can achieve >83% accuracy, which can result in good performances.

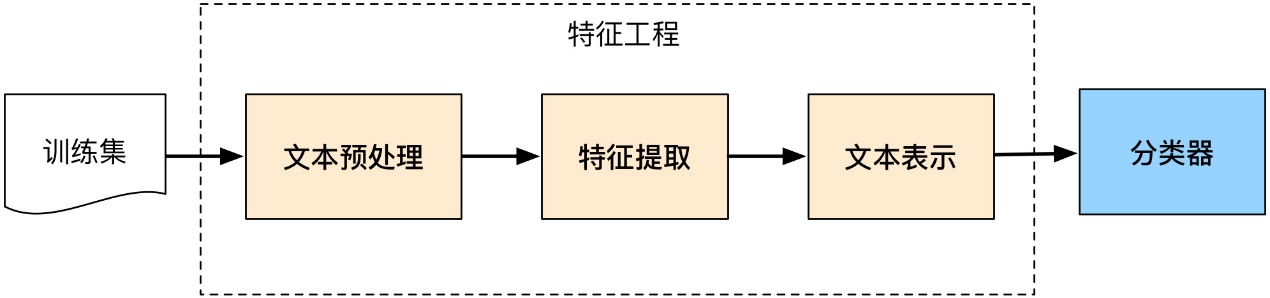
**Keywords:** text classification, natural language processing, deep learning,

深度学习 (Deep learning) 通过建立深层神经网络, 模拟人脑的机制进行解释并分析学习图像、语音及文本等数据, 是目前机器学习研究中的一个热点领域. 传统机器学习工作的有效性,很大程度上依赖于人工设计的数据表示和输入特征的有效性; 机器学习方法在这个过程中的作用仅仅是优化学习权重以便最终输出最优的学习结果. 与传统机器学习方法不同的是, 深度学习试图自动完成数据表示和 特征提取工作; 并且深度学习更强调, 通过学习过程提取出不同水平、不同维度的有效表示, 以便提高同抽象层次上对数据的解释能力. 从认知科学角度来看, 这个思路与人类学习机理非常吻合. 在面对大量感知数据的处理过程中, 人脑对其中的重要信息有着特殊的敏感性. 例如即使是四岁孩童, 放学时间站在校门口观望大量的接送家长, 总 是比较容易快速准确地发现家人熟悉的身影, 欣喜 地扑进家人的怀抱. 因此, 在人工智能研究领域中, 对于如何模仿人脑开展高效的复杂数据处理, 引发 了研究者的极大兴趣. 其中, 从仿生学角度开展的人 脑生理结构研究, 以及从人脑应用角度开展的功能 研究, 是两个典型的研究方向. 前者体现研究对象的 结构特征, 后者体现研究对象的功能特征. 两类研究又是互相渗透, 相互支撑. 例如, 在对哺乳类动物开 展的解剖研究中发现, 大脑皮质存在着层次化的系列区域; 在此基础上, 神经科学研究人员又通过测试 视觉信号输入人脑视网膜后经大脑前额皮质层到达 运动神经的时间, 推断发现大脑皮质层的主要功能 在于将视觉信号通过复杂的多层网络模型后加以提取观测信息, 而并未直接对视觉信号进行特征处理. 这就说明, 人脑在识别物体过程中, 并未直接通过视 网膜投影的外部世界进行感知, 而是需要依靠经过 某种聚集和分解处理后的信息才能识别得到物体. 这一过程中, 视皮层的功能主要是开展对视觉信号 的特征提取和计算, 而非简单重现视网膜图像. 这 种具有明确层次结构的人类视觉感知系统在大大降 低了视觉感知处理数据量的同时, 还能够保留被感 知物体关键的结构信息. 大脑这种分层次结构启发 了研究人员开展多层次神经网络的研究. 最早出现 的多层网络训练算法是采用初始值随机选定及梯度 下降优化策略的 BP (Back-propagation) 神经网络. 但是这种多层结构的主要缺陷在于输入与输出间存 在的非线性映射导致能量函数或网络误差函数空间 含有多个局部极小点, 同时采用的又是使能量或误 差单一减小的搜索方向, 容易导致局部收敛最小而 非全局最优. 相关实验及理论发现, 局部收敛 最优的情况会随着网络层数的增加而变得越来越严重, 似乎表明 BP 算法在向多层深度结构方向发展 上并无优势可言, 这在一定程度上影响了深度学习 的发展. 浅层学习结构的共同特点是仅含一种将单个原始输入信号映射到特定问题空间的简单特征结构, 基本上可以认为这类模型带有一层或没有隐层节点. 常见的此类结构有条件随机场 (Conditional randomﬁeld, CRF)、隐马尔科夫模型(HiddenMarkov model, HMM)、支持向量机 (Support vector machine, SVM)、多层感知器 (Multilayer perceptron, MLP) 及最大熵模型 (Maximum entropy, ME) 等. 这些模型大多应用在传统信号处理技术及机器学习 研究中, 存在着对复杂函数表示能力有限、对复杂问 题泛化处理能力不足的局限性. 这种情况直到 2006 年才出现转机. Hinton 等 利用深度可信网络 (Deep belief network, DBN) 结构, 对组成 DBN 的每一层受限玻尔兹曼机 (Restricted Boltzmann machine, RBM) 结构进行无 监督学习训练, 并将其用于 MNIST1 手写数字识 别任务中, 取得了错误率仅为 1.2% 的最好成绩. 不久之后, Bengio 等也提出了一种基于自动编码 器 (Auto-encoders) 的相关算法, 同样取得了较好 结果. 这些算法尽管形式不同, 但他们都遵循 相同的原理: 即在每一层局部使用无监督的训练 算法, 以引导完成特征中间表示层的训练目标. 此后, 其他一些非RBM或非 Auto-encoders 结构的 深度学习算法也陆续提出. 自 2006 年以来, 这些深度学习方法不仅在分类任务上取得显著结果, 而且在时序预测、高维降秩、 纹理建模、运动建模、对象分割、 信息抽取及自然语言处理领域都有不俗表现. 此外, 尽管上述深度模型中, 普遍采用 Auto-encoders、RBM 和DBN 结构, 能够以无监督的方式从未标注数据中学习到良好的结果, 但在面对特定任务领域时, 有监督反馈算法用来初始化深 度结构的方式也有成功应用. 尽管当前深度学习还未有完备的理论体系支撑, 但并不妨碍在图像识别和语音识别等应用领域率先 结出累累硕果. 2012 年, 一种称为“深度神经网络 (Deep neural network, DNN)”的机器学习模型在 图像识别领域的 ImageNet 评测上被采用, 把识别 错误率从26% 降到15%, 是图像识别领域近年来的 最好结果. 而在此之前的 2011 年, 同样类似的 DNN 技术在语音识别领域也取得惊人效果, 降低语音识 别错误率达 20%∼30%, 从而大大推进了应用技术 产品的开发. 比如基于 DNN 技术的微软全自动同 声传译系统, 在 2012 年 11 月中国天津的一次公开 活动中流畅地实现了自动语音识别、英文到中文的 机器翻译以及合成中文语音输出的整个过程, 效果 震惊全场. 尽管深度学习已经在上述图像和语音处理领域 取得显著进展, 但是在同属人类认知范畴的自然语 言处理任务中, 应用还未有重大突破. 本文重点分析 了当前面向自然语言处理的深度学习研究进展, 并探讨了深度学习在自然语言处理领域的可能发展空 间, 以图抛砖引玉.

## 一、传统文本分类方法

文本分类问题算是自然语言处理领域中一个非常经典的问题了，相关研究最早可以追溯到上世纪50年代，当时是通过专家规则（Pattern）进行分类，甚至在80年代初一度发展到利用知识工程建立专家系统，这样做的好处是短平快的解决top问题，但显然天花板非常低，不仅费时费力，覆盖的范围和准确率都非常有限。

后来伴随着统计学习方法的发展，特别是90年代后互联网在线文本数量增长和机器学习学科的兴起，逐渐形成了一套解决大规模文本分类问题的经典玩法，这个阶段的主要套路是人工特征工程+浅层分类模型。训练文本分类器过程见下图：



整个文本分类问题就拆分成了特征工程和分类器两部分，玩机器学习的同学对此自然再熟悉不过了

**1.1 特征工程**

特征工程在机器学习中往往是最耗时耗力的，但却极其的重要。抽象来讲，机器学习问题是把数据转换成信息再提炼到知识的过程，特征是“数据-->信息”的过程，决定了结果的上限，而分类器是“信息-->知识”的过程，则是去逼近这个上限。然而特征工程不同于分类器模型，不具备很强的通用性，往往需要结合对特征任务的理解。

文本分类问题所在的自然语言领域自然也有其特有的特征处理逻辑，传统分本分类任务大部分工作也在此处。文本特征工程分位文本预处理、特征提取、文本表示三个部分，最终目的是把文本转换成计算机可理解的格式，并封装足够用于分类的信息，即很强的特征表达能力。

**1）文本预处理**

文本预处理过程是在文本中提取关键词表示文本的过程，中文文本处理中主要包括文本分词和去停用词两个阶段。之所以进行分词，是因为很多研究表明特征粒度为词粒度远好于字粒度，其实很好理解，因为大部分分类算法不考虑词序信息，基于字粒度显然损失了过多“n-gram”信息。

具体到中文分 词，不同于英文有天然的空格间隔，需要设计复杂的分词算法。传统算法主要有基于字符串匹配的正向/逆向/双向最大匹配；基于理解的句法和语义分析消歧；基于统计的互信息/CRF方法。近年来随着深度学习的应用，WordEmbedding + Bi-LSTM+CRF方法逐渐成为主流，本文重点在文本分类，就不展开了。而停止词是文本中一些高频的代词连词介词等对文本分类无意义的词，通常维护一个停用词表，特征提取过程中删除停用表中出现的词，本质上属于特征选择的一部分。

经过文本分词和去停止词之后淘宝商品标题变成了下图“ / ”分割的一个个关键词的形式：

夏装 / 雪纺 / 条纹 / 短袖 / t恤 / 女 / 春 / 半袖 / 衣服 / 夏天 / 中长款 / 大码 / 胖mm / 显瘦 / 上衣 / 夏

**2）文本表示和特征提取**

**文本表示：**

文本表示的目的是把文本预处理后的转换成计算机可理解的方式，是决定文本分类质量最重要的部分。传统做法常用词袋模型（BOW, Bag Of Words）或向量空间模型（Vector Space Model），最大的不足是忽略文本上下文关系，每个词之间彼此独立，并且无法表征语义信息。词袋模型的示例如下：

( 0, 0, 0, 0, .... , 1, ... 0, 0, 0, 0)

一般来说词库量至少都是百万级别，因此词袋模型有个两个最大的问题：高纬度、高稀疏性。词袋模型是向量空间模型的基础，因此向量空间模型通过特征项选择降低维度，通过特征权重计算增加稠密性。

**特征提取：**

向量空间模型的文本表示方法的特征提取对应特征项的选择和特征权重计算两部分。特征选择的基本思路是根据某个评价指标独立的对原始特征项（词项）进行评分排序，从中选择得分最高的一些特征项，过滤掉其余的特征项。常用的评价有文档频率、互信息、信息增益、χ²统计量等。

特征权重主要是经典的TF-IDF方法及其扩展方法，主要思路是一个词的重要度与在类别内的词频成正比，与所有类别出现的次数成反比。

**3）基于语义的文本表示**

传统做法在文本表示方面除了向量空间模型，还有基于语义的文本表示方法，比如LDA主题模型、LSI/PLSI概率潜在语义索引等方法，一般认为这些方法得到的文本表示可以认为文档的深层表示，而word embedding文本分布式表示方法则是深度学习方法的重要基础，下文会展现 。

**1.2 分类器**

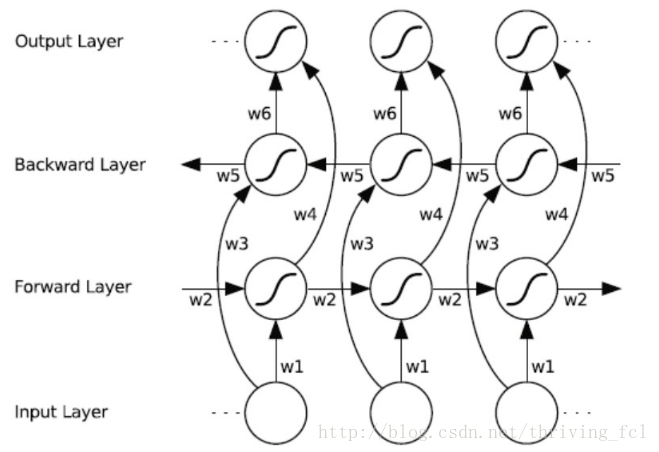
分类器基本都是统计分类方法了，基本上大部分机器学习方法都在文本分类领域有所应用，比如朴素贝叶斯分类算法（Naïve Bayes）、KNN、SVM、最大熵和神经网络等等，传统分类模型不是本文重点，在这里就不展开了。

**深度学习的方法**

一、Attention 的作用

在RNN的文本分类模型中，可以把RNN看成一个encoder，将需要被分类的文本表示为一个dense vector，再使用全连接层与softmax输出各类别的概率。

在具体的文本的表示上，可以将RNN最后一个时刻的输出作为文本的表示，也可以综合考虑每个时刻的的输出，将它们合并为一个向量。在tagging与classication的任务中常用双向RNN(下文写作BIRNN)，每个时刻的输出向量可以理解为这个时刻的输入词在上下文的语境中对当前任务的一个贡献。BIRNN如下图所示



根据人类的阅读习惯进行思考，我们在阅读的时候，注意力通常不会平均分配在文本中的每个词。再回到上面的文本表示，如果直接将每个时刻的输出向量相加再平均，就等于认为每个输入词对于文本表示的贡献是相等的，但实际情况往往不是这样，比如在情感分析中，文本中地名、人名这些词应该占有更小的权重，而情感类词汇应该享有更大的权重。

所以在合并这些输出向量时，希望可以将注意力集中在那些对当前任务更重要的向量上。也就是给他们都分配一个权值，将所有的输出向量加权平均。假设输出向量为htht，权值为αtαt，则合并后的表示为

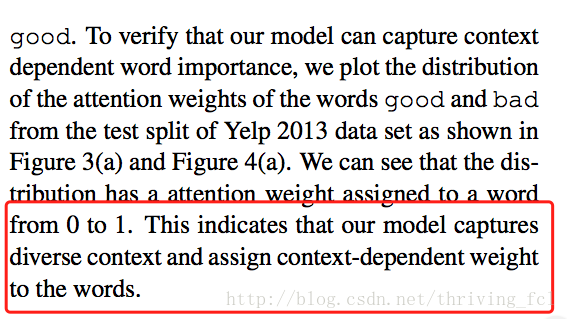
s=∑tαtht

上文所说的为BIRNN的每个输出向量分配不同权值，使得模型可以将注意力集中在重点几个词，降低其他无关词的作用的机制就是Attention机制。使用了Attention机制可以使得文本表示的结果在当前的任务中更合理。

使用Attention的两个好处：

1.可以更好的表征文本，使训练出的模型具有更好的表现（更高的精度）。

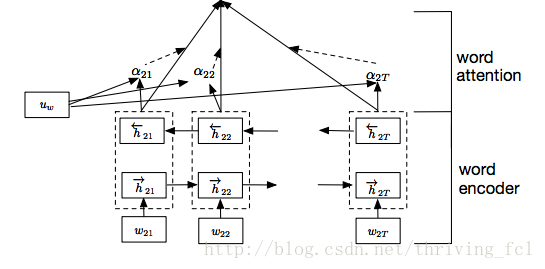
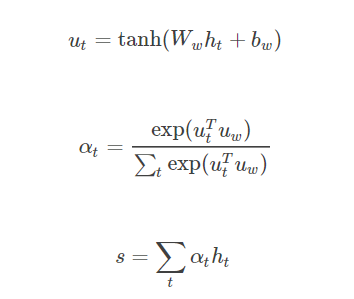
2.为模型提供了更好的解释性，如直观的看出在文本分类中哪些词或句子更重要， 如果训练出的模型不理想，根据这些权值分析badcase也非常方便。



Hierarchical Attention Networks for Document Classification 这篇论文中提到他们的模型可以学习到词的上下文信息，并且分配与上下文相关的词的权重。我觉得这并不是由于Attention这个机制的作用，而是RNN本身就具有这个能力。

二、Attention 原理

上文说到需要给BIRNN的每个输出分配权重，如何分配就是Attention的原理，用一张结构图加三个公式应该就可以解释清楚了。



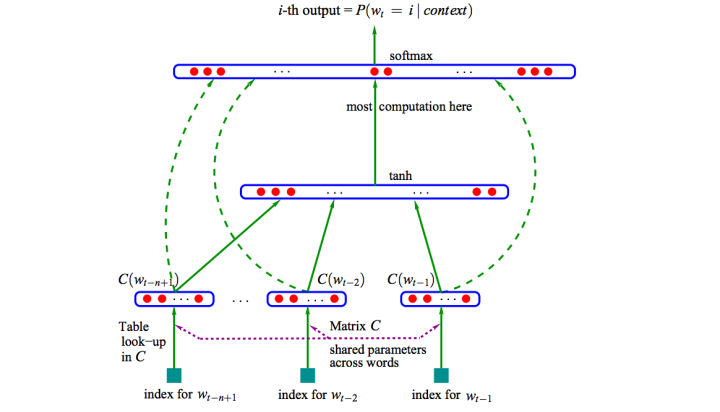
公式(1)中的WwWw与bwbw为Attention的权重与bias，在实现的时候也要设置attention的size，不过也可以简单的令它们等于BIRNN的输出向量的size。

公式(2)中的uwuw也是需要设置的权重，公式(2)其实也就是对所有uTtuwutTuw结果的softmax。

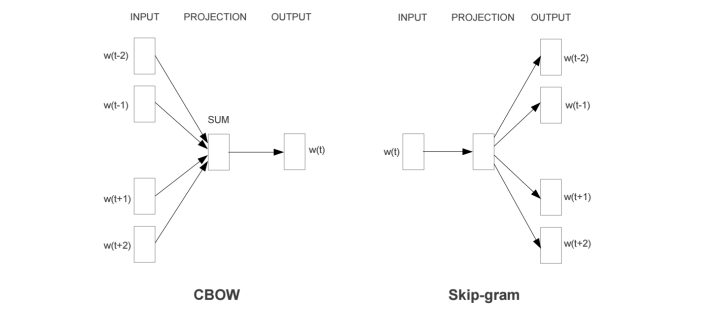
公式(3)即是将计算出的αtαt 作为各时刻输出的权值，对它们加权求和表示为一个向量。

**2.1 文本的分布式表示：词向量（word embedding）**

分布式表示（Distributed Representation）其实Hinton 最早在1986年就提出了，基本思想是将每个词表达成 n 维稠密、连续的实数向量，与之相对的one-hot encoding向量空间只有一个维度是1，其余都是0。分布式表示最大的优点是具备非常powerful的特征表达能力，比如 n 维向量每维 k 个值，可以表征 个概念。事实上，不管是神经网络的隐层，还是多个潜在变量的概率主题模型，都是应用分布式表示。下图是03年Bengio在 [A Neural Probabilistic Language Model](https://link.zhihu.com/?target=http%3A//www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf) 的网络结构：

这篇文章提出的神经网络语言模型（NNLM，Neural Probabilistic Language Model）采用的是文本分布式表示，即每个词表示为稠密的实数向量。NNLM模型的目标是构建语言模型：

词的分布式表示即词向量（word embedding）是训练语言模型的一个附加产物，即图中的Matrix C。

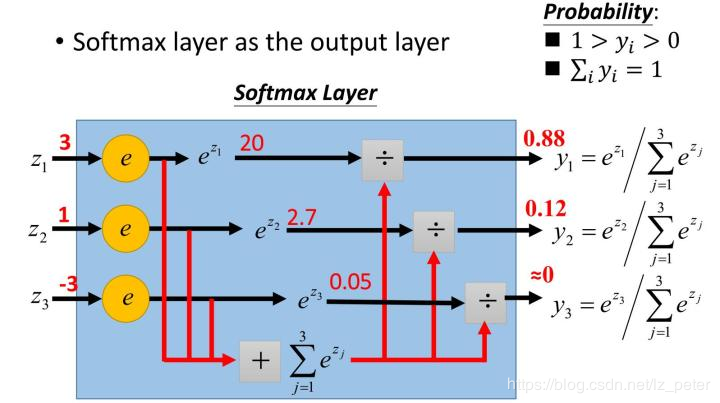
尽管Hinton 86年就提出了词的分布式表示，Bengio 03年便提出了NNLM，词向量真正火起来是google Mikolov 13年发表的两篇word2vec的文章 [Efficient Estimation of Word Representations in Vector Space](https://link.zhihu.com/?target=http%3A//ttic.uchicago.edu/~haotang/speech/1301.3781.pdf) 和 [Distributed Representations of Words and Phrases and their Compositionality](https://link.zhihu.com/?target=https%3A//papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf)，更重要的是发布了简单好用的[word2vec工具包](https://link.zhihu.com/?target=https%3A//code.google.com/archive/p/word2vec/)，在语义维度上得到了很好的验证，极大的推进了文本分析的进程。下图是文中提出的CBOW 和 Skip-Gram两个模型的结构，基本类似于NNLM，不同的是模型去掉了非线性隐层，预测目标不同，CBOW是上下文词预测当前词，Skip-Gram则相反。

除此之外，提出了Hierarchical Softmax 和 Negative Sample两个方法，很好的解决了计算有效性，事实上这两个方法都没有严格的理论证明，有些trick之处，非常的实用主义。额外多提一点，实际上word2vec学习的向量和真正语义还有差距，更多学到的是具备相似上下文的词，比如“good”“bad”相似度也很高，反而是文本分类任务输入有监督的语义能够学到更好的语义表示。

至此，文本的表示通过词向量的表示方式，把文本数据从高纬度高稀疏的神经网络难处理的方式，变成了类似图像、语音的的连续稠密数据。深度学习算法本身有很强的数据迁移性，很多之前在图像领域很适用的深度学习算法比如CNN等也可以很好的迁移到文本领域了，下一小节具体阐述下文本分类领域深度学习的方法。

**2.2 softmax函数**

softmax函数，又称归一化指数函数。它是二分类函数sigmoid在多分类上的推广，目的是将多分类的结果以概率的形式展现出来。下图展示了softmax的计算方法：



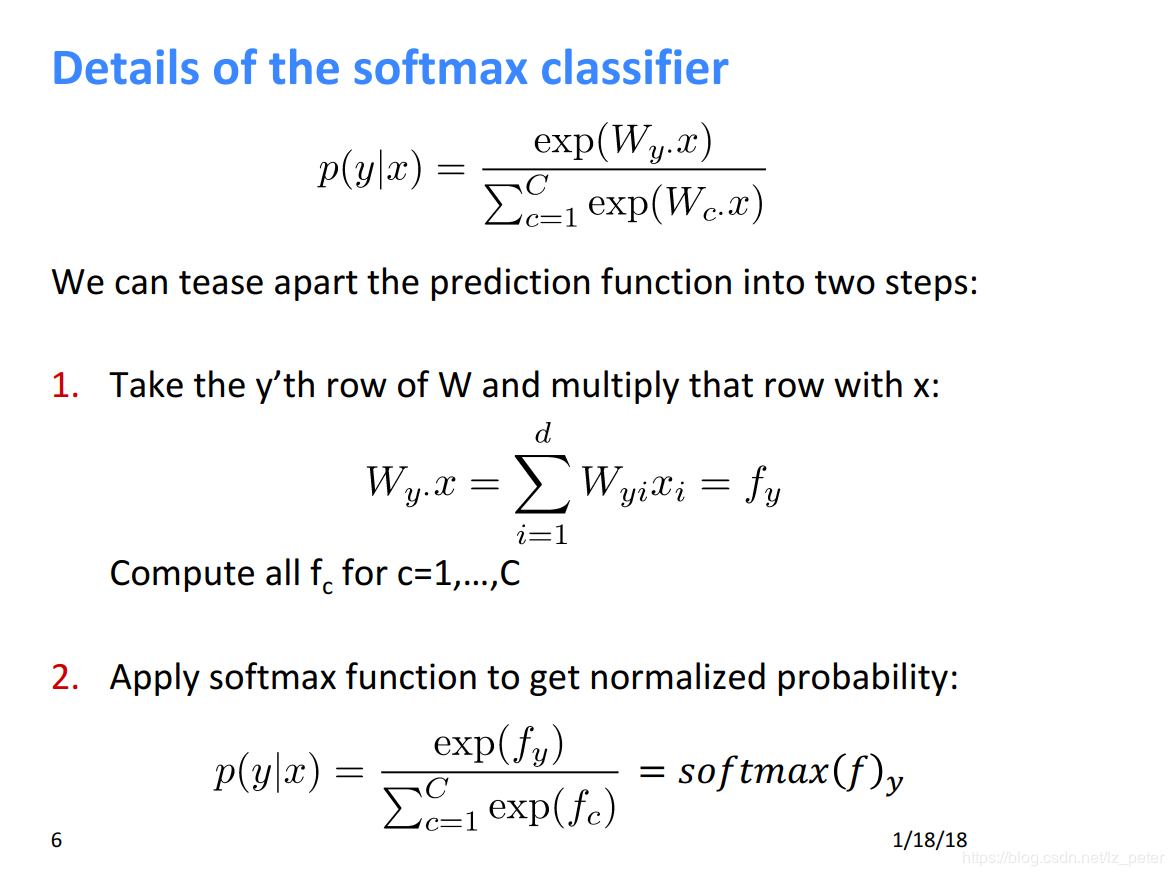
我们知道指数函数的值域取值范围是零到正无穷。与概率取值相似的地方是它们都是非负实数。那么我们可以1）利用指数函数将多分类结果映射到零到正无穷；2）然后进行归一化处理，便得到了近似的概率。

总结一下softmax如何将多分类输出转换为概率，可以分为两步：

1）分子：通过指数函数，将实数输出映射到零到正无穷。

2）分母：将所有结果相加，进行归一化。

下图为斯坦福大学CS224n课程中最softmax的解释：



## 2.3 N-Gram模型详解

既然要做语言模型，基于统计概率来说，我们需要计算句子的概率大小 ，这个也就是最终要求的一句话的概率了，概率大，说明更合理，概率小，说明不合理。

因为是不能直接计算，所以我们先应用条件概率得到

中间插入下条件概率： P(B|A)：A 条件下 B 发生的概率。从一个大的空间进入到一个子空间（切片），计算在子空间中的占比。



然而，如果直接算条件概率转化后的式子的话，对每个词要考虑它前面的所有词，这在实际中意义不大，显然并不好算。那这个时候我们可以添加什么假设来简化吗？可以的，我们可以基于马尔科夫假设来做简化。

**什么是马尔科夫假设？**

马尔科夫假设是指，每个词出现的概率只跟它前面的少数几个词有关。比如，二阶马尔科夫假设只考虑前面两个词，相应的语言模型是三元模型。引入了马尔科夫假设的语言模型，也可以叫做马尔科夫模型。  
  
马尔可夫链（Markov chain）为状态空间中经过从一个状态到另一个状态的转换的随机过程。该过程要求具备“无记忆”的性质：下一状态的概率分布只能由当前状态决定，在时间序列中它前面的事件均与之无关。

也就是说，应用了这个假设表明了当前这个词仅仅跟前面几个有限的词相关，因此也就不必追溯到最开始的那个词，这样便可以大幅缩减上述算式的长度。即式子变成了这样：

注：这里的m表示前m个词相关

然后，我们就可以设置m=1，2，3，....得到相应的一元模型，二元模型，三元模型了，关于

当 m=1, 一个一元模型（unigram model)即为 ：



当 m=2, 一个二元模型（bigram model)即为 ：



当 m=3, 一个三元模型（trigram model)即为



而N-Gram模型也就是这样，当m=1，叫1-gram或者unigram ；m=2，叫2-gram或者bigram ；当 m=3叫3-gram或者trigram ；当m=N时，就表示的是N-gram啦。

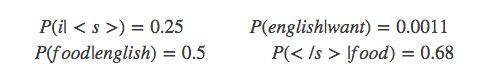
说明了什么是N-Gram模型之后，下面说说N-Gram经典应用，同时更深入的理解下：

## 利用N-Gram模型评估语句是否合理

假设现在有一个语料库，我们统计了下面的一些词出现的数量

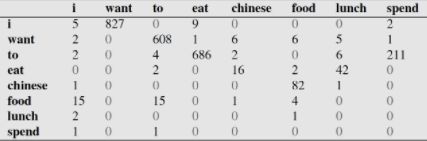


下面的这些概率值作为已知条件：

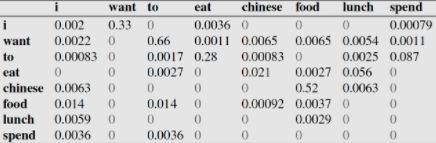


p(want|<s>) = 0.25

下面这个表给出的是基于Bigram模型进行计数之结果



例如，其中第一行，第二列 表示给定前一个词是 “i” 时，当前词为“want”的情况一共出现了827次。据此，我们便可以算得相应的频率分布表如下。



比如说，我们就以表中的p(eat|i)=0.0036这个概率值讲解，从表一得出“i”一共出现了2533次，而其后出现eat的次数一共有9次，p(eat|i)=p(eat,i)/p(i)=count(eat,i)/count(i)=9/2533 = 0.0036

下面我们通过基于这个语料库来判断s1=“<s> i want english food</s>” 与s2 = "<s> want i english food</s>"哪个句子更合理：

首先来判断p(s1)

P(s1)=P(i|<s>)P(want|i)P(english|want)P(food|english)P(</s>|food)

=0.25×0.33×0.0011×0.5×0.68=0.000031

再来求p(s2)？

P(s2)=P(want|<s>)P(i|want)P(english|want)P(food|english)P(</s>|food)

=0.25\*0.0022\*0.0011\*0.5\*0.68 = 0.00000002057

通过比较我们可以明显发现0.00000002057<0.000031,也就是说s1= "i want english food</s>"更合理。

当然，以上是对于**二元语言模型（bigram model）**的，大家也可以算下三元，或者1元语言模型的概率，不过结果都应该是一样的，

再深层次的分析，我们可以发现这两个句子的概率的不同，主要是由于顺序i want还是want i的问题，根据我们的直觉和常用搭配语法，i want要比want i出现的几率要大很多。所以两者的差异，第一个概率大，第二个概率小，也就能说的通了。

## n-gram 模型其他应用举例

n-gram模型也有其他很多应用，以下一一举例：

1.**研究人类文明：**n-gram模型催生了一门新学科([Culturomics](https://link.zhihu.com/?target=https%3A//en.wikipedia.org/wiki/Culturomics))的成立，通过数字化的文本，来研究人类行为和文化趋势。

2.**搜索引擎：**当你在谷歌或者百度的时候，输入一个或几个词，搜索框通常会以下拉菜单的形式给出几个像下图一样的备选，这些备选其实是在猜想你想要搜索的那个词串。如下图：



实际上这些都是根据语言模型得出。比如使用的是二元语言模型预测下一个单词：排序的过程就是：

p(”不一样“|"我们")>p(”的爱“|"我们")>p(”相爱吧“|"我们")>.......>p("这一家"|”我们“)，这些概率值的求法和上面提到的完全一样，数据的来源可以是用户搜索的log。

3.**输入法：**比如输入“zhongguo”，可能的输出有：中国，种过，中过等等....这背后的技术就要用到n-gram语言模型了。item就是每一个拼音对应的可能的字。

.......(还有很多，只有有关语言模型，都可以应用)

**2.4 深度学习文本分类模型**

词向量解决了文本表示的问题，该部分介绍的文本分类模型则是利用CNN/RNN等深度学习网络及其变体解决自动特征提取（即特征表达）的问题。

1）fastText

f fastText是Facebook于2016年开源的一个词向量计算和文本分类工具，在学术上并没有太大创新。但是它的优点也非常明显，在文本分类任务中，fastText（浅层网络）往往能取得和深度网络相媲美的精度，却在训练时间上比深度网络快许多数量级。在标准的多核CPU上， 能够训练10亿词级别语料库的词向量在10分钟之内。可以看出fastText有两个主要的特点：

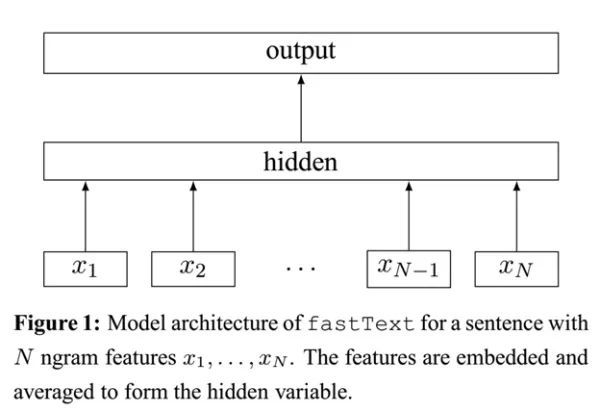
1. 速度很快
2. 在速度的基础上精度较高 。

对应的解决办法就是：

1. 层级简单 + embedding叠加 + 分层Softmax
2. 字符级别的n-gram

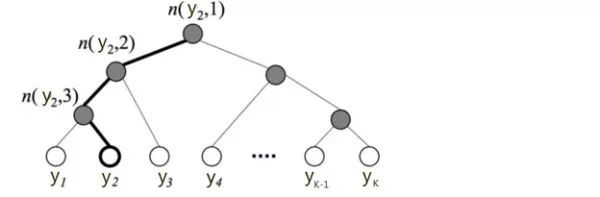
### 解释

* 快的原因：

1. 层级简单:[[](http://5b0988e595225.cdn.sohucs.com/images/20180126/ff57de198e894003a949e94202feff7b.jpeg)](http://5b0988e595225.cdn.sohucs.com/images/20180126/ff57de198e894003a949e94202feff7b.jpeg)
2. 单词的embedding叠加获得的文档向量. 全连接参数由 n *L*1024 变成 1 *L*1024
3. 在输出时，fastText采用了分层Softmax，大大降低了模型训练时间：

标准的Softmax回归中，要计算y=j时的Softmax概率：，我们需要对所有的K个概率做归一化，这在|y|很大时非常耗时。于是，分层Softmax诞生了，它的基本思想是使用树的层级结构替代扁平化的标准Softmax，使得在计算时，只需计算一条路径上的所有节点的概率值，无需在意其它的节点。

下图是一个分层Softmax示例：

[[](http://5b0988e595225.cdn.sohucs.com/images/20180126/4de02dd88a944fc38aa89dd2bb1a653d.jpeg)](http://5b0988e595225.cdn.sohucs.com/images/20180126/4de02dd88a944fc38aa89dd2bb1a653d.jpeg)

树的结构是根据类标的频数构造的霍夫曼树。K个不同的类标组成所有的叶子节点，K-1个内部节点作为内部参数，从根节点到某个叶子节点经过的节点和边形成一条路径。从根节点走到叶子节点，实际上是在做了3次二分类的逻辑回归。通过分层的Softmax，计算复杂度一下从|K|降低到log|K|。

* 准的原因：字符级别的n-gram：

word2vec把语料库中的每个单词当成原子的，它会为每个单词生成一个向量。这忽略了单词内部的形态特征，比如：“apple” 和“apples”，“达观数据”和“达观”，这两个例子中，两个单词都有较多公共字符，即它们的内部形态类似，但是在传统的word2vec中，这种单词内部形态信息因为它们被转换成不同的id丢失了。

为了克服这个问题，fastText使用了字符级别的n-grams来表示一个单词。对于单词“apple”，假设n的取值为3，则它的trigram有:

|  |  |
| --- | --- |
| 1 | “<ap”, “app”, “ppl”, “ple”, “le>” |

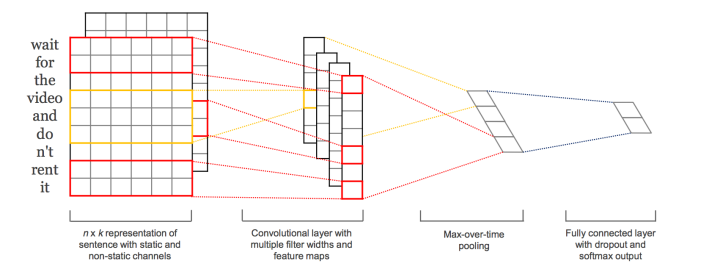
其中，<表示前缀，>表示后缀。于是，我们可以用这些trigram来表示“apple”这个单词，进一步，我们可以用这5个trigram的向量叠加来表示“apple”的词向量。

优点：

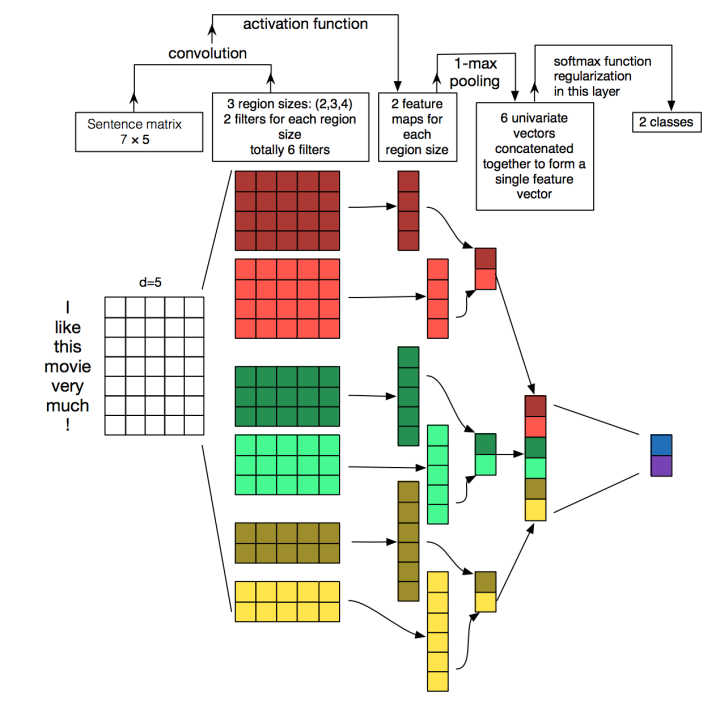
1. 对于低频词生成的词向量效果会更好。因为它们的n-gram可以和其它词共享。
2. 对于训练词库之外的单词，仍然可以构建它们的词向量。我们可以叠加它们的字符级n-gram向量。

**2）TextCNN**

fastText 中的网络结果是完全没有考虑词序信息的，而它用的 n-gram 特征 trick 恰恰说明了局部序列信息的重要意义。[卷积神经网络（CNN Convolutional Neural Network）](https://link.zhihu.com/?target=http%3A//colah.github.io/posts/2014-07-Understanding-Convolutions/)最初在图像领域取得了巨大成功，CNN原理就不讲了，核心点在于可以**捕捉局部相关性**，具体到文本分类任务中可以利用CNN来提取句子中类似 n-gram 的关键信息。



TextCNN的详细过程原理图见下：

**TextCNN详细过程**：第一层是图中最左边的7乘5的句子矩阵，每行是词向量，维度=5，这个可以类比为图像中的原始像素点了。然后经过有 filter\_size=(2,3,4) 的一维卷积层，每个filter\_size 有两个输出 channel。第三层是一个1-max pooling层，这样不同长度句子经过pooling层之后都能变成定长的表示了，最后接一层全连接的 softmax 层，输出每个类别的概率。

**特征**：这里的特征就是词向量，有静态（static）和非静态（non-static）方式。static方式采用比如word2vec预训练的词向量，训练过程不更新词向量，实质上属于迁移学习了，特别是数据量比较小的情况下，采用静态的词向量往往效果不错。non-static则是在训练过程中更新词向量。推荐的方式是 non-static 中的 fine-tunning方式，它是以预训练（pre-train）的word2vec向量初始化词向量，训练过程中调整词向量，能加速收敛，当然如果有充足的训练数据和资源，直接随机初始化词向量效果也是可以的。

**通道（Channels）**：图像中可以利用 (R, G, B) 作为不同channel，而文本的输入的channel通常是不同方式的embedding方式（比如 word2vec或Glove），实践中也有利用静态词向量和fine-tunning词向量作为不同channel的做法。

**一维卷积（conv-1d）**：图像是二维数据，经过词向量表达的文本为一维数据，因此在TextCNN卷积用的是一维卷积。一维卷积带来的问题是需要设计通过不同 filter\_size 的 filter 获取不同宽度的视野。

**Pooling层**：利用CNN解决文本分类问题的文章还是很多的，比如这篇 [A Convolutional Neural Network for Modelling Sentences](https://link.zhihu.com/?target=https%3A//arxiv.org/pdf/1404.2188.pdf) 最有意思的输入是在 pooling 改成 (dynamic) k-max pooling ，pooling阶段保留 k 个最大的信息，保留了全局的序列信息。比如在情感分析场景，举个例子：

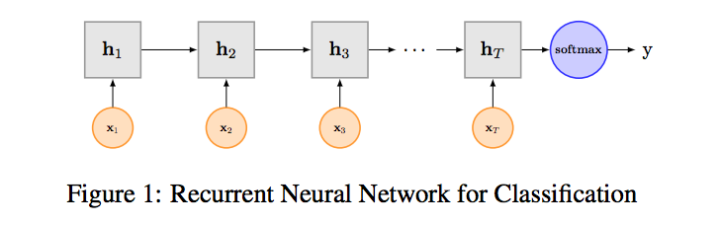
“ 我觉得这个地方景色还不错，但是人也实在太多了 ”

虽然前半部分体现情感是正向的，全局文本表达的是偏负面的情感，利用 k-max pooling能够很好捕捉这类信息。

**3）TextRNN**

尽管TextCNN能够在很多任务里面能有不错的表现，但CNN有个最大问题是固定 filter\_size 的视野，一方面无法建模更长的序列信息，另一方面 filter\_size 的超参调节也很繁琐。CNN本质是做文本的特征表达工作，而自然语言处理中更常用的是递归神经网络（RNN, Recurrent Neural Network），能够更好的表达上下文信息。具体在文本分类任务中，Bi-directional RNN（实际使用的是双向LSTM）从某种意义上可以理解为可以捕获变长且双向的的 "n-gram" 信息。

RNN算是在自然语言处理领域非常一个标配网络了，在序列标注/命名体识别/seq2seq模型等很多场景都有应用，下图LSTM用于网络结构原理示意图，示例中的是利用最后一个词的结果直接接全连接层softmax输出了。

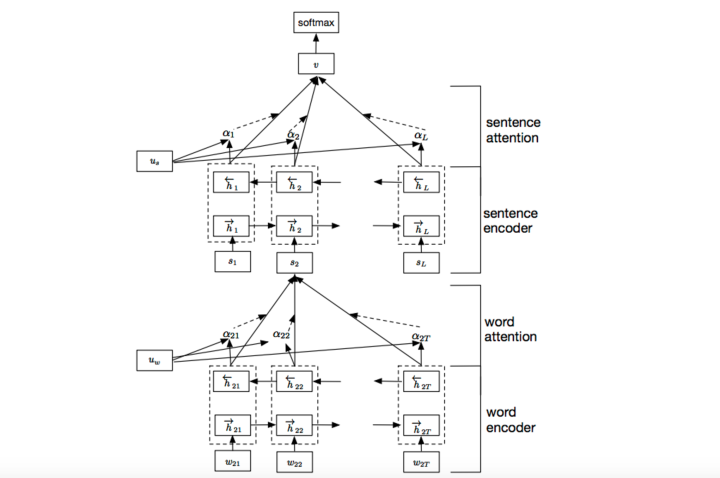


**4）TextRNN + Attention**

CNN和RNN用在文本分类任务中尽管效果显著，但都有一个不足的地方就是不够直观，可解释性不好，特别是在分析badcase时候感受尤其深刻。而注意力（Attention）机制是自然语言处理领域一个常用的建模长时间记忆机制，能够很直观的给出每个词对结果的贡献，基本成了Seq2Seq模型的标配了。实际上文本分类从某种意义上也可以理解为一种特殊的Seq2Seq，所以考虑把Attention机制引入近来，研究了下学术界果然有类似做法。

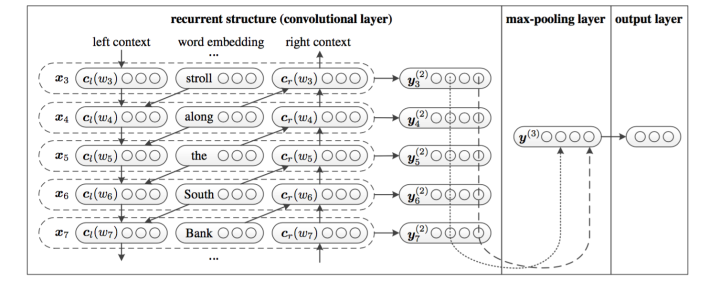
**TextRNN + Attention 模型**：

下图是模型的网络结构图，它一方面用层次化的结构保留了文档的结构，另一方面在word-level和sentence-level。淘宝标题场景只需要 word-level 这一层的 Attention 即可。

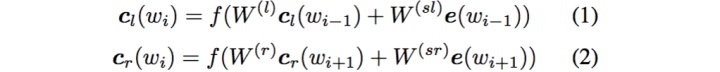
加入Attention之后最大的好处自然是能够直观的解释各个句子和词对分类类别的重要性。

**5）TextRCNN（TextRNN + CNN）**

我们参考的是中科院15年发表在AAAI上的这篇文章 Recurrent Convolutional Neural Networks for Text Classification 的结构：



利用前向和后向RNN得到每个词的前向和后向上下文的表示：



这样词的表示就变成词向量和前向后向上下文向量concat起来的形式了，即：

最后再接跟TextCNN相同卷积层，pooling层即可，唯一不同的是卷积层 filter\_size = 1就可以了，不再需要更大 filter\_size 获得更大视野，这里词的表示也可以只用双向R。

**三．结束语**

尽管基于神经网络的深度学习在文本处理的方面有着很大的进步但是，仍然存在着许多无法解决的问题，由于一词多义的存在, 使得即使采用词向量技术作为深度学习的原始输入信号, 也还是不能如图像或语音一样将所有原始信息确定地输入到深度学 习模型中.在深度学习模型分层表示原始输入信号 的不同特征时, 这种不确定性所带来的误差有可能 在不同层间被传递并局部放大.解决这种一词多义所带来的不确定性的方法, 似乎还是要结合上下文语言情境. 因此, 突破自然语 言字、词、短语、小句等局部表示的局限性, 面向包含上下文全局信息的篇章、文本来开展深层语义理 解, 如篇章分析、篇章理解等, 应该是重点发展的方向之一.

**References：**

* 1. 03年Bengio在 [A Neural Probabilistic Language Model](https://link.zhihu.com/?target=http%3A//www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf)
  2. 中科院15年发表在AAAI上的文章Recurrent Convolutional Neural Networks for Text Classification
  3. [Hierarchical Attention Networks for Document Classification](https://link.zhihu.com/?target=https%3A//www.cs.cmu.edu/~diyiy/docs/naacl16.pdf)
  4. <https://zhuanlan.zhihu.com/p/32829048>
  5. <https://www.sohu.com/a/219080991_129720>
  6. <https://zhuanlan.zhihu.com/p/25928551>
  7. <https://blog.csdn.net/thriving_fcl/article/details/73381217>
  8. <https://zhuanlan.zhihu.com/p/34219483>