


浅谈智能搜索和对话式 OS



我偏笑_NS Nirvana (/u/2293f85dc197)  [+ 关注](#)

2017.05.22 20:11* 字数 21617 阅读 1826 评论 1 喜欢 25

(/u/2293f85dc197)

前面的文章主要从理论的角度介绍了**自然语言人机对话系统**所可能涉及到的多个领域的经典模型和基础知识。这篇文章，甚至之后的文章，会更贴近业务的角度来写，侧重于介绍一些与自然语言问答业务密切相关，且已经商业化，已被证明可行的技术。

本篇文章准备从技术的角度从头捋一下百度度秘事业部 CTO 朱凯华做的题为《AI 赋能的搜索和对话交互》的报告，介绍智能搜索和对话式 OS 的发展趋势以及一些经典问题的解决方案。里面提到的技术会包含，里面没提到但是我有联想到的内容，也会补充进去。

智能搜索

根据朱凯华的原话，现代搜索引擎面临两个重要挑战：

- 更好的建模搜索结果的语义相关性
- 更直接地给用户答案

看到这两个挑战，第一个让我想到的是检索模型与 LDA 等模型的关系，传统检索模型是在文档包含查询词的前提下进行相似度的计算与排序，而 LDA 以及类似的其他模型可以利用词向量（或者讲，语义相关性）发现两篇完全没有重合词的相似文档。

第二个让我想到的是知识图谱，知识图谱使知识对计算机而言成为了可计算的，在知识图谱的基础上，计算机可以通过简单的推理直接给出一些问题的答案，而不再是提供与查询词相关的网页集合，能够更直接的解决用户的问题。



智能搜索这部分就从**语义相关性**和**知识推理**两个方面来讲。

语义相关性

我们所希望实现的，是不论查询词是否包含在相应文档中，只要查询词与文档拥有语义相关性，就能将其找到。

精确命中

这个方面也还是从传统检索模型开始讲起，也即在文档包含查询词的情况下，或者讲查询词精确命中文档的前提下，如何计算相似度，如何对内容进行排序。

在《浅谈搜索引擎基础（上）》(<https://www.jianshu.com/p/c51416cf2849>)中，我提到过，BM25 模型是目前最成功的内容排序模型，也对其进行了详细的介绍，这里结合朱凯华的PPT再简单说一下：

BM25: representation capability baseline

Keyword hits:

命中的关键词对语义相关性
有正向贡献

Term weighting:

通过TF-IDF来决定命中词的贡献值

Okapi weighting based document score: [23]

$$\sum_{t \in Q, t \in D} \left[\frac{N - df + 0.5}{df + 0.5} \cdot \frac{(k_1 + 1)df}{(k_1(1 - b) + b \frac{N}{|D|}) + df} \cdot \frac{(k_2 + 1)df}{k_2 + df} \right]$$

k_1 (between 1.0–2.0), b (usually 0.75), and k_2 (between 0–1000) are constants.

Pivoted normalization weighted longest document score: [30]

$$\sum_{t \in Q, t \in D} \frac{1 - \beta + \beta \ln \left(\frac{q_t}{q_{t, \max}} \right)}{(1 - \beta) + \beta \ln \left(\frac{q_t}{q_{t, \max}} \right)} \cdot q_t \cdot \ln \frac{N - 1}{df}$$

β is a constant (usually 0.37).

Doc length normalization:

文档中未命中词对语义相关性都有负向贡献

<https://sinabat.info/docsc8201.pdf>

Baidu 百度

(/apps/download?
utm_source=sbc)



这是我文章中的图：

$$\sum_{i \in Q} \log \underbrace{\frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)}}_{\text{二元独立模型}} \cdot \underbrace{\frac{(k_1 + 1)f_i}{K + f_i}}_{\text{查询词文档位置}} \cdot \underbrace{\frac{(k_2 + 1)qf_i}{k_2 + qf_i}}_{\text{查询权重}}$$

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

(/apps/download?utm_source=sbc) ×

BM25 模型计算公式融合了 4 个考虑因素：**IDF 因子**、**文档词频**、**文档长度因子**和**查询词频**，并利用 3 个自由调节因子（k1、k2 和 b）对各种因子的权值进行调整组合。

与他在图中展示的相对应：BM25 模型计算公式中的求和符号，表示命中的关键词对语义相关性有正向贡献；二元独立模型（当前条件下等价于**IDF 因子**）和**文档词频**（TF）决定了命中词的贡献值；**文档长度因子**使得文档未命中词对语义相关性有着负向贡献；他的图中并没有直接的提到查询词频，但实际上他式子中的 q_{tf} 就是**查询词频**。

精确命中的查询词对整体语义相关性的正/负向贡献，是语义表达能力的基础。

接下来我们考虑另一种情况：

信息检索的进化史：语义表达能力的提升

● Q: A B C D E

D: X A B Y C D' Z

(B和C属于精确命中的词：BM25算法已经可以解决)

也即如果查询词并没有精确命中，那该怎样计算语义相关性？图中指出了四个方向：归一化命中、同义词命中、主题命中、反义主题命中，后两个是我造的：)

归一化命中

先来说归一化命中，这个是最简单的：

信息检索的进化史：语义表达能力的提升

● Q: A B C D E

D: X A B Y C D' Z

(B和C属于精确命中的词：BM25算法已经可以解决)

● 如何建模“未命中的词”的语义相关性，是信息检索理论演进的核心

■ 归一化命中 (A -> (1.0) A): 词干提取、拼写纠错、繁体转换、数字格式归一化

✧ 词干提取: biking -> bike; apples -> apple; tested -> test

✧ 拼写纠错: briteny -> britney

✧ 繁体转换: 怎么 -> 怎么; 天气 -> 天气

✧ 数字格式归一化: 二月一日 -> 2月1日; 三岁 -> 3岁

图中包含四个类别：词干提取、拼写纠错、简繁体转换、数字格式归一化。

我还能想到的，比如全角半角归一化、标点符号处理、大小写归一化、汉语的错别字纠正等等。

归一化命中，就是解决同一个词多种形态的问题。

其实如果强行分的话，归一化命中也可以分作两种问题：一种是计算机背锅的问题，比如词干提取、简繁体转换、数字格式归一化、全角半角归一化、标点符号处理、大小写归一化；另一种就是人背锅的问题，比如英文拼写纠错以及汉语错别字纠正。

同义词命中

信息检索的进化史：语义表达能力的提升

- Q: A B C D E
D: X A B Y C D' Z
(B和C属于精准命中的词：BM25算法已经可以解决)
- 如何构建“未命中的词”的语义相关性，是信息检索理论演进的核心
 - 归一化命中 (A \rightarrow (1.0) A)
 - 同义词命中 (D \rightarrow (0.8) D'): 具有相同概念的词互换
 - Car \rightarrow Auto
 - 上下文无关的同义词: 宝宝 \rightarrow (0.9) 小孩
 - 上下文相关的同义词: 初中2013级学业水平测试: 级 \rightarrow (0.85) 年级; 年; 期
 - 多term的同义词: 苹果手机 \rightarrow (0.7) iPhone; 多少 \rightarrow (0.8) 报价 | 价格

Baidu 百度

同义词命中，自然就是解决同一含义多个同义词的问题。

(/apps/download?
utm_source=sbc)

具体的例子图中也给出了，对于汉语而言，主要的类别包括：**上下文无关的同义词**、**上下文相关的同义词**、**多term的同义词**（**实体消歧**和**共指消解**？）。

上下文无关的同义词可以利用一些在线的同义词典来解决；对于上下文相关的同义词，猜想一下，依照自身业务，通过人工定义规则可以解决一部分，另一部分可以通过筛选后面的多term同义词以及主题命中的结果来得到。

多term的同义词让我想起了知识图谱构建三个层次之一的**知识融合**，知识融合其中有一部分关键内容叫做**实体链接**，这在《浅谈知识图谱基础》(<https://www.jianshu.com/p/4f09043e22ea>)中详细的讲过，实体链接的意义恰巧在于通过**实体消歧**和**共指消解**为不同的实体名称/别名找到真正对应的实体。

实体消歧是在同一个名称可能对应多个实体时根据上下文找到正确的实体，共指消解主要用于解决多个指称对应同一实体对象的问题。

利用知识图谱中的实体链接技术，就能找到哪些些实体名称/别名对应的是同一个实体。

主题命中

主题命中是相当重要的一部分内容，这里准备花大量篇幅去拓展开来讲。

(/apps/download?
utm_source=sbc)



信息检索的进化史：语义表达能力的提升

图中讲，建模词与词之间『爱』的关系，这个『爱』我没明白，这里我按照拥有相同主题来理解，比如『苹果』和『乔布斯』和『iPad』、比如『星巴克』和『咖啡』等。

建模词与词之间“爱”的关系

统计机器翻译模型SMT(Microsoft: Gao et al. 2010, 2011)

x	$P(q s)$	q	$P(q s)$
rank	0.56213	Vista	0.80575
slap	0.01383	Windows	0.02344
movie	0.01222	Download	0.00728
pictures	0.01211	slimark	0.00571
stink	0.00697	ap	0.00555
bars	0.00684	microsoft	0.00542
phone	0.00533	it	0.00536
rose	0.00447	compatible	0.00270
people	0.00441	premium	0.00264
surveys	0.00263	hus	0.00211
$s = \text{thentic}$		$s = \text{virts}$	
x	$P(q s)$	q	$P(q s)$
crisis	0.52826	postiff	0.11288
oil	0.02872	post	0.09851
mount	0.02117	playground	0.01729
dates	0.00959	valve	0.01053
person	0.00953	burles	0.01051
sumali	0.00903	current	0.01712
elizabeth	0.00454	quorum	0.01373
cost	0.00446	weyno	0.01372
stair	0.00441	john	0.01004
height	0.00297	screen	0.01001
$s = \text{crisis}$		$s = \text{postiff}$	

使用SMT模型计算词
语间“相关”的关系



在 PPT 中，百度用 SMT 模型来计算词与词之间的相关关系。SMT 先放一放，我们先说词相关的问题。既然说到了词相关的问题，就不得不说到词向量了，说到词向量又不得不提 word2vec，这部分就从 word2vec 开始讲起。

word2vec

word2vec 是 Google 于2013年开源推出的一个用于获取词向量的工具包。

先提一下什么是 Huffman 树和 Huffman 编码，简单一句话，词频越大的词离根节点越近，词频越大的词的编码长度越短。

然后再说一下语言模型，毕竟词向量和语言模型有着密切的关系。

(/apps/download?
utm_source=sbc)



根据牛津大学的《Deep Learning for Natural Language Processing》课程，至少存在三种可行的语言模型，分别是：**Count based N-Gram Language Models (N-Gram 语言模型)**、**Neural N-Gram Language Models (神经概率语言模型)**、**Recurrent Neural Network Language Models (RNN 语言模型)**。

这里先说前两种，其实第一种在我的《浅谈自然语言处理基础（上）》(<https://www.jianshu.com/p/c123c7534500>)中，已经详细讲过了，是一种利用 n-1 阶马尔科夫假设的简单语言模型。

然后这里着重介绍一下神经概率语言模型，神经概率语言模型由 Bengio 等人在文《A neural probabilistic language model. Journal of Machine Learning Research》提出，该模型中用到了词向量。

那什么是词向量呢，一种最简单的词向量是 **one-hot representation (稀疏向量)**，就是用很长的向量来表示一个词，向量的长度为词典的大小，向量的分量只有一个 1，其它全为 0，1 的位置对应该词在词典中的索引。

在《浅谈机器学习基础（上）》(<https://www.jianshu.com/p/ed9ae5385b89>)中讲朴素贝叶斯算法时就提到了**词集模型 (Set Of Words, SOW)**和**词袋模型 (Bag Of Words, BOW)**，其实词集模型就是文档中不同词的 one-hot representation 之和；词袋模型与词集模型的区别在于，词集模型对一个词的 one-hot representation 只加一次，而词袋模型全部累加起来，记录词在文档中的出现次数。

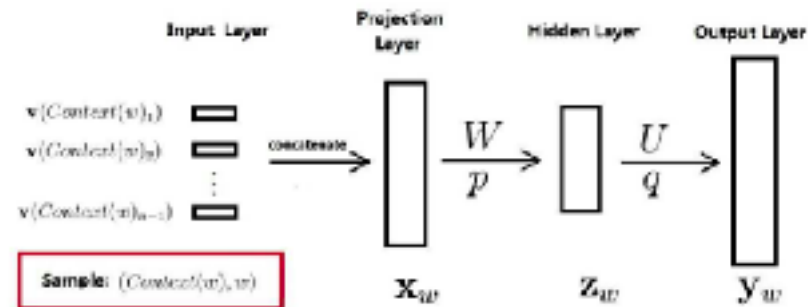
在《浅谈数据挖掘基础》(<https://www.jianshu.com/p/a83a52da14a2>)中，我提到过**维数灾难**，维数灾难的一个表现是，在高维空间下，几乎所有的点对之间的距离都差不多相等；另一个表现是，集合任意的两个向量之间都是近似正交的。这会让我们欧氏距离和余弦距离计算方式失效。在词典大小过大的情况下，使用 one-hot representation 表示文档，并通过《浅谈搜索引擎基础（上）》(<https://www.jianshu.com/p/c51416cf2849>)



中提到的空间向量法等计算相似度，就会触发维数灾难。并且如果利用 one-hot representation，那两个词的词向量之间就永远是正交关系，我们无法利用它来刻画词与词之间的相似性。

然后还有另一种词向量，叫做 **Distributed Representation (密集向量/稠密向量)**，它最早由 Hinton 于 1986 年提出的，其基本想法是：通过训练将某种语言中的每一个词映射成一个固定长度的短向量（这里的短是相对于 one-hot representation 来说的），所有这些向量构成一个词向量空间，而每一向量视为该空间中的一个点，在这个空间上引入『距离』，就可以根据词之间的距离来判断它们之间的（词法、语义上的）相似性了。神经概率语言模型和 word2vec 采用的都是这种 Distributed Representation 的词向量。得到词语的 Distributed Representation 的词向量的过程，可以看做是 Word Embedding。

然后回到神经概率语言模型，既然是神经概率语言模型，自然要用到神经网络，这里给出这个神经网络的结构示意图，出自《word2vec 的数学原理详解》：



它包括四个层：**输入层**、**投影层 (Projection Layer)**、**隐藏层**和**输出层**。有人也将输入层与投影层结合，统一看做输入层，这样就变成了三层神经网络结构。

然后来详细讲一下这个图，对于语料 C 中的任意一个词 w ，将 $\text{Context}(w)$ 取为其前面的 $n-1$ 个词（类似于 N-Gram），如果其前面的词不足 $n-1$ 个，则可以人为的添加一个或几个填充向量，它们也参与训练过程。这样二元对 $(\text{Context}(w), w)$ 就是一个训练样本了。

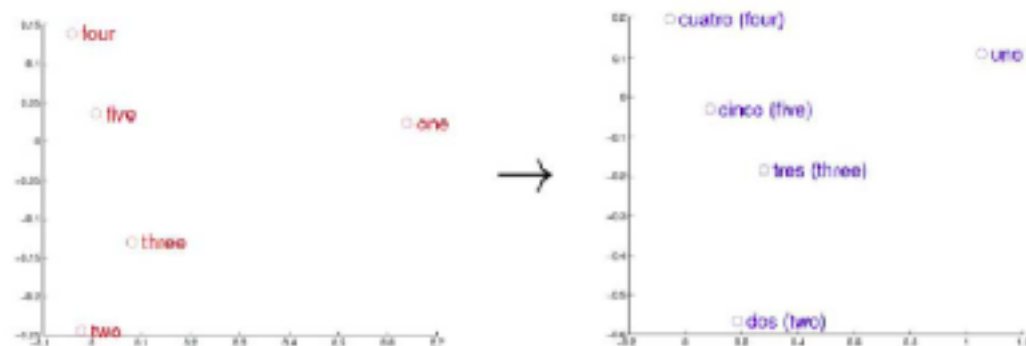
(/apps/download?utm_source=sbc) ×

先来说输入层和投影层，如果词向量长度为 m ，则投影层的规模是 $(n-1)m$ 。因为输入层包含 $\text{Context}(w)$ 中 $n-1$ 个词的词向量，而投影层的向量由输入层的 $n-1$ 个词的词向量按顺序首尾相接地拼起来形成一个长向量。

然后是输出层，输出层的规模与词典大小的规模相同，而隐藏层的规模是可调参数，由用户指定。

通过语料训练，我们得到了词典中不同单词 w 的词向量以及填充向量，和这个神经网络的参数。对于神经概率语言模型，训练时，词向量是用来帮助构造目标函数的辅助参数，训练完成后，它也只是语言模型的一个副产品，不过是很重要的副产品。

根据词向量就可以度量词之间的相似性，包括同一种语言内的词的相似性，甚至还可以度量不同语言之间词的相似性，如下图所示：



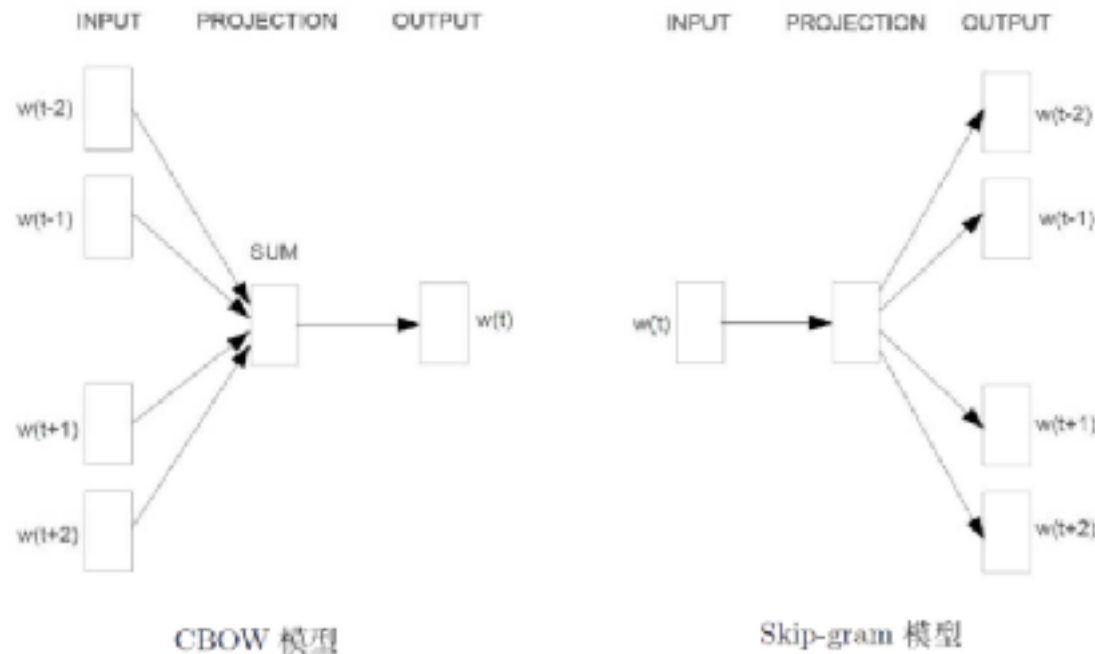
词向量只是针对『词』来提的，事实上，我们可以针对更细粒度和更粗粒度进行推广，比如**字向量**、**句子向量**和**文档向量**，它们能为字、句子、文档等单元提供更好的表示。

正因为有了词向量，神经概率语言模型与传统的 N-Gram 语言模型相比，可以更好的建模一些原训练语料中没有出现过的搭配。比如语料中存在『A cute cat』，但却不存在『A cute dog』。如果在 N-Gram 语言模型下，『A cute cat』的生成概率会很高，而『A cute dog』的生成概率就近乎为零，但神经概率语言模型却可以发现『cat』与『dog』的相似性，进而为『A cute dog』提供一个更恰当的生成概率。

(/apps/download?utm_source=sbc) ×

接下来正式开始介绍 word2vec，word2vec 并不是直接基于前面说的神经概率语言模型，而是用到了两个特殊的模型：**CBOW模型 (Continuous Bag-of-Words Model)** 和 **Skip-gram 模型 (Continuous Skip-gram Model)**。

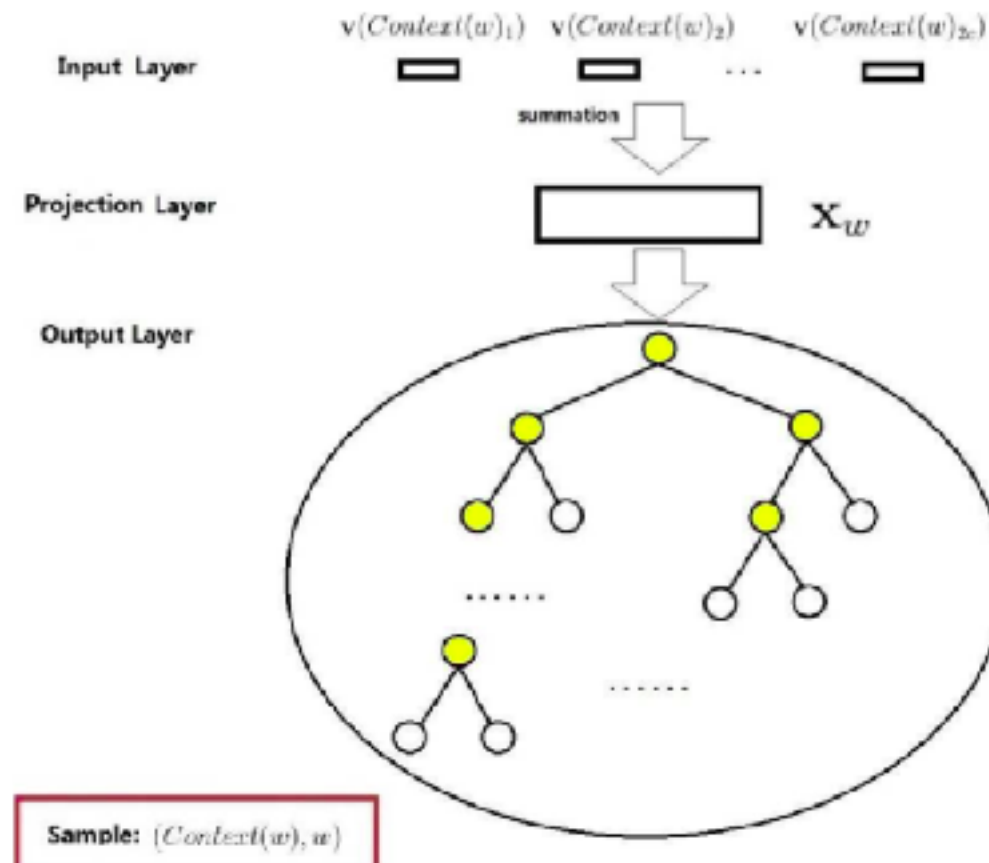
两个模型都包含三层：**输入层、投影层和输出层**。前者是在已知当前词 w 上下文的前提下预测当前词；而后者恰恰相反，是在已知当前词 w 的前提下，预测其上下文：



对于 CBOW 和 Skip-gram 两个模型，word2vec 给出了两套框架，它们分别基于 Hierarchical Softmax 和 Negative Sampling（简称为 NEG），NEG 是 NCE（Noise Contrastive Estimation）的一个简化版本，目的是用来提高训练速度并改善所得词向量的质量。与 Hierarchical Softmax 相比，NEG 不再使用复杂的 Huffman 树，而是利用相对简单的**随机负采样**，能大幅提高性能。

先来说基于 Hierarchical Softmax 的 CBOW 模型：

(/apps/download?utm_source=sbc) ×



(/apps/download?
utm_source=sbc)

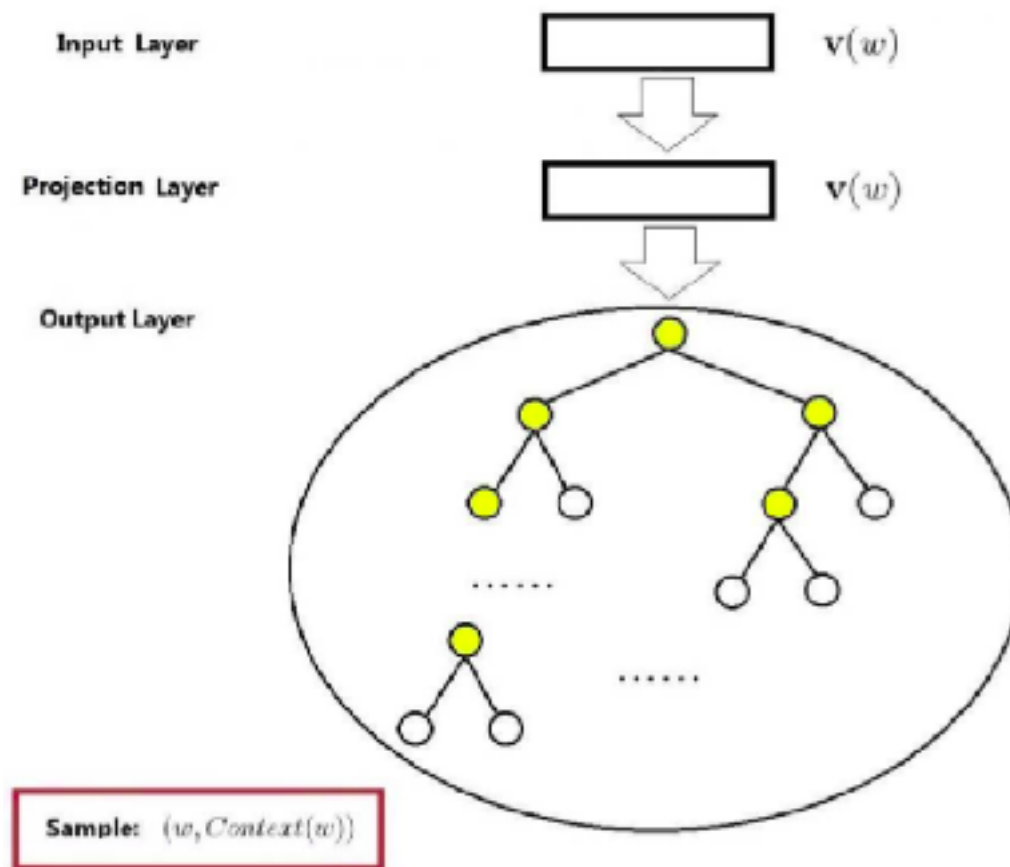
与前面讲过的神经概率语言模型相比，主要存在三处不同：

- （从输出层到投影层的操作）神经概率语言模型是通过拼接，而 CBOW 是通过**累加求和**
- （隐藏层）神经概率语言模型有隐藏层，而 CBOW **没有隐藏层**
- （输出层）神经概率语言模型是线性结构，而 CBOW 采用 **Huffman 树**，当然基于 Hierarchical Softmax 采用的都是 Huffman 树。

那如何利用 Huffman 树来得到我们所需的 $p(w|\text{Context}(w))$ 呢？对于词典 D 中的任意词 w ，Huffman 树中必存在一条从根节点到词 w 对应节点的路径 p_w （且这条路径是唯一的）。路径 p_w 上存在 l_w-1 个分支，将每个分支看做一次二分类，每一次分类就产生一

个概率，将这些概率乘起来，就是所需的 $p(w|\text{Context}(w))$ 。

然后是基于 Hierarchical Softmax 的 Skip-gram 模型：



(/apps/download?utm_source=sbc) ×

前面说过，Skip-gram 是在已知当前词 w 的前提下，预测其上下文，所以其输入就只包含当前样本的中心词 w ；图中的投影层其实是多余的，只是为了方便比较，因为只输入词 w ，所以投影层和输入层恒等；与 CBOW 模型一样，输出也是一棵 Huffman 树。



在 Skip-gram 模型中，我们需要的是 $p(\text{Context}(w)|w)$ ， $p(\text{Context}(w)|w)$ 需要通过计算 $p(u|w)$ 的累乘积来得到（ u 是所有包含在 $\text{Context}(w)$ 中的词）。而每个 $p(u|w)$ 的计算方式，与 CBOW 中的思想类似。

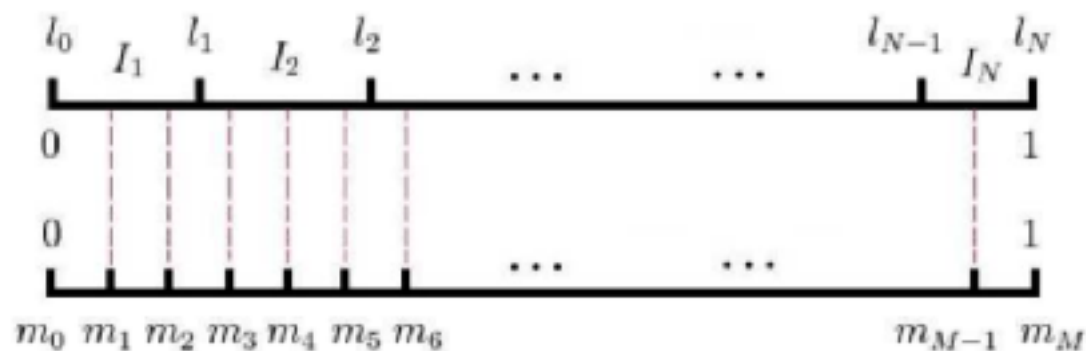
前面说，NEG 以随机负采样代替 Huffman 树，其实严格来说，这么讲不对，一种方法怎么代替一种结构呢，只是不再使用 Huffman 树，转而利用负采样近似取得相同的结果，同时大幅提高了性能。

其实 NEG 本质采用的方法还是最大似然估计，类似逻辑回归（Logistic Regression），增大正样本的概率同时降低负样本的概率。

比如对于基于 NEG 的 CBOW 模型，已知 w 的上下文 $\text{Context}(w)$ ，需要预测 w ，因此，对于给定的 $\text{Context}(w)$ ， w 就是一个正样本，其他词就是负样本了，那负样本那么多，该如何挑选呢？

接下来介绍负采样算法，这本质上是一个**带权采样问题**，词典 D 中的词在语料 C 中出现的次数有高有低，对于那些高频词，被选为负样本的概率就应该比较大，反之，对于那些低频词，其被选中的概率就应该比较小。

接下来再谈谈 word2vec 中的具体做法，其实与我在《浅谈深度学习基础（上）》
(<https://www.jianshu.com/p/df9a4473d6d4>)中提到的**吉布斯采样**类似，如下图所示：



(/apps/download?utm_source=sbc) ×

假设词典 D 中共有 N 个词，每个词对应一段长度，这个长度与其在语料 C 中的出现频率正相关，也即图中的上半部分，一个有 N 个划分区间的**非等距划分**。假设存在 $M \gg N$ ，图的下半部分是对同样总长度的**等距划分**，拥有 M 个划分区间。

这样，对于正样本 w ，每当我们需要抽取一个负样本，就从 M 个划分区间随机抽取一个，找到其对应的非等距划分区间所对应的单词，作为负样本。这样，在语料中词频越高的单词就越容易被选中。另外，如果负样本采样时正好抽到了我们的正样本 w ，那就跳过去继续抽取。

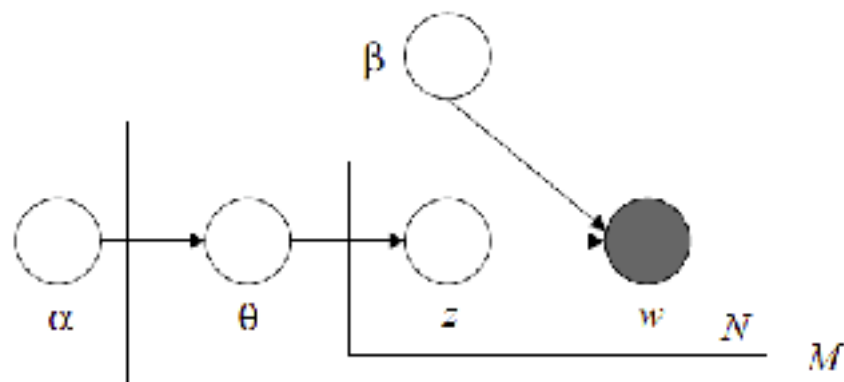
(/apps/download?utm_source=sbc) ×

LSA、pLSA、LDA

除了 word2vec 所采用的 CBOW 和 Skip-gram 以及神经概率语言模型之外，还有很多不同模型可以用来估计词向量，其中就包括在《浅谈推荐系统基础》(<https://www.jianshu.com/p/c8711ff27eb0>)详细讲过的**LDA (Latent Dirichlet Allocation)**，以及与其类似的主题模型 **LSA (Latent Semantic Analysis)** 和 **pLSA (probabilistic Latent Semantic Analysis)**。

通常 LDA 的效果要好于 LSA 和 pLSA。

先简单回顾一下 LDA，LDA 模型是这样引入的：



假设我们需要生成 M 篇文章，每篇文章包含 N 个单词。LDA 模型包含了 $\alpha \beta \theta z w$ 五个参数，一个生成任务对应一对 $\alpha \beta$ ， α 决定主题向量 θ 的分布，每篇文章对应一个主题向量 θ ，主题向量 θ 决定主题 z 的分布，文章中的每个单词对应一个主题 z ，主题 z 与 β 一起决定每个单词的分布。

(/apps/download?utm_source=sbc) ×

LDA 的联合概率分布如下：

$$p(\theta, z, w | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$$

LDA 可以合理的将单词归类到不同的隐含主题之中；并且通过计算两篇文章主题向量 θ 的 KL 散度（相对熵），可以得到两篇文章的相似性。

LSA 与 pLSA 这里也简单介绍一下原理。

LSA 的基本思想其实就是将文章从稀疏的高维空间（基于 one-hot representation 的词向量）映射到一个低维的向量空间（基于 Distributed Representation 的词向量），我们称之为隐含语义空间（Latent Semantic Space），那如何做到的呢。

在《浅谈机器学习基础（下）》(<https://www.jianshu.com/p/0359e3b7bb1b>)中，我介绍过 SVD 的这样一个用法：SVD 可以应用在自然语言处理中，利用矩阵分解的思想估计出同义词类、文章主题以及同义词类与文章主题之间的关系。

和 PCA 采用特征值分解的思想类似，LSA 就采用了 SVD 的方式来求解 Latent Semantic Space。

pLSA 是对 LSA 的优化，引入了概率模型，但并不完善。LDA 在 pLSA 的基础上进一步优化了概率模型，自从诞生之后便蓬勃的发展，广泛应用于个性化推荐、社交网络、广告预测等各个领域。

SMT、DSSM、Sent2Vec

然后回到朱凯华 PPT 中提到的 SMT 模型：

(/apps/download?
utm_source=cbc)



建模词与词之间“爱”的关系

统计机器翻译模型SMT(Microsoft: Gao et al. 2010, 2011)

α	$P(\alpha w)$	β	$P(\beta w)$
rank	0.56213	Vista	0.80075
slaps	0.01283	Windows	0.02344
movie	0.01222	Download	0.00728
pictures	0.01211	slimane	0.00471
stink	0.00697	ap	0.00345
fees	0.00681	microsoft	0.00342
photos	0.00533	it	0.00336
rose	0.00449	compatible	0.00270
people	0.00441	premium	0.00264
survivors	0.00362	rhin	0.00211
$w = \text{rhin}$		$w = \text{Vista}$	
α	$P(\alpha w)$	β	$P(\beta w)$
crimes	0.52826	point	0.11238
oil	0.10672	pop	0.09861
mount	0.02117	playground	0.01029
dance	0.00953	active	0.00695
person	0.00933	burden	0.00551
summit	0.00903	current	0.00712
elkshing	0.00454	quarries	0.00373
cost	0.00446	synce	0.00372
visit	0.00441	inm	0.00364
height	0.00292	newcom	0.00351
$w = \text{crimes}$		$w = \text{point}$	

使用SMT模型计算词
语间“相关”的关系



我找到了 PPT 中图的出处，出自 Jianfeng Gao 的这篇论文 Clickthrough-Based Translation Models for Web Search: from Word Models to Phrase Models (<https://link.jianshu.com?t=https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/cikm1108-draft.pdf>)

这篇论文的主要意思是，查询语句与文档中所使用的词汇以及语言风格的差异，会给搜索引擎带来挑战。所以 Jianfeng Gao 等提出了将翻译模型引入传统检索模型的想法。文章中尝试训练了两个不同的翻译模型，一个基于词的翻译模型用于学习单个词之间的翻译概率，而另一个基于短语的翻译模型用于学习多词短语之间的翻译概率，这两个模型被集成到了传统检索模型中，发现可以得到更优的搜索质量。并且一些标准的统计机器翻译技术，如字对齐，双语短语提取等，也可以被用来构建更好的网页文献检索系统。模型的训练集来源于用户的点击日志，文章认为用户的查询语句与用户最终点击的文档标题具有一定的语义相关性，这就是题目中 Clickthrough-Based 的来源。



朱凯华的 PPT 中所提到的统计机器翻译模型，指的就是这个。既然 SMT 模型就是为了解决词汇以及语言风格之间的差异问题，那它能够通过计算词之间的相似性以提高搜索质量就是理所应当的了。

在查询 SMT 模型相关文献的同时，我发现了同样出自微软之手的 **DSSM (Deep Structured Semantic Model)**，出自论文 Learning Deep Structured Semantic Models for Web Search using Clickthrough Data (https://link.jianshu.com?t=https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/cikm2013_DSSM_fullversion.pdf)

DSSM 的提出背景与 SMT 模型基本相同，同样是致力于利用语义匹配理解文档与查询语句间词汇使用以及语言风格的差异，以进一步提高搜索质量。DSSM 同样基于用户点击数据来训练模型。

DSSM 具有三个特点：

- 采用基于**三字母 (tri-letter)**的词哈希方式来表示长短不同的单词
- 采用深度神经网络来抽象高层次的语义表示
- 采用用户点击数据来有监督的训练模型

举个例子，什么叫做基于三字母 (tri-letter) 的词哈希方式。比如词『#cat#』，它的三字母表示方式就是『#-c-a』、『c-a-t』、『a-t-#』。这与在《浅谈语音识别基础》(<https://www.jianshu.com/p/a0e01b682e8a>)中所提到的**三音素**异曲同工。

关于 DSSM 模型的效果，文中有这样一张图：

(/apps/download?utm_source=sbc) ×

NDCG results on a real-world Web search task

Models	NDCG@1	NDCG@3	NDCG@10
BM25	30.8	37.3	45.5
Previous Shallow/Deep Semantic Models, trained on doc collection (unsupervised)			

(/apps/download?
utm_source=sbc)

其中，NDCG (Normalized Discounted Cumulative Gain) 是用来衡量搜索质量的指标。其计算公式如下：

$$N(n) = \frac{1}{Z_n} \sum_{i=1}^n \frac{(2^{r(i)} - 1)}{\log(1 + j)}$$

Normalization Cumulating Gain Position discount

Gain 定义了每条结果的质量，Position discount 使得排名越靠前的结果权重越大，NDCG 通过累积计算前 k 条结果的质量来定义搜索质量。

然后重新回到图中，图中讲基于点击日志的 DSSM 模型，要优于同样基于点击日志的 Word Translation Model (即百度 PPT 中提到的 SMT 模型) 等，也优于基于文档集的 LSA、pLSA 和 Deep Auto-Encoder (在《浅谈深度学习基础(上)》(<https://www.jianshu.com/p/df9a4473d6d4>)中有详细讲解)，也优于传统的 BM25 模型。只是不知道为什么没有与 LDA 比较。

基于 DSSM，微软开放下载了 **Sent2Vec**。Sent2Vec (<https://link.jianshu.com?t=https://www.microsoft.com/en-us/download/details.aspx?id=52365>) 将一对短字符串 (比如查询语句与用户点击的文档标题) 映射到一个连续低维空间，这样就可以利用余弦相似度来计算短字符串对之间的语义相似性。

简单总结一下，word2vec 可以通过词向量估计词之间的相似性，Sent2Vec 可以通过字符串向量估计短字符串对之间的相似性，LDA 在估计长篇文档之间的相似性时，效果较好。

反义主题命中

然后我们重新回到 PPT 中来：

(/apps/download?utm_source=sbc) ×

信息检索的进化史：语义表达能力的提升

- Q: A B C D E
D: X A B Y C D' Z
(B 和 C 属于精确命中的词：BM25 算法已经可以解决)
- 如何建模“未命中的词”的语义相关性，是信息检索理论演进的核心
 - 归一化命中 (A \rightarrow (1.0) A)
 - 同义词命中 (D \rightarrow (0.8) D')
 - 主题词与词之间“爱”的关系 (E \rightarrow (+0.2) X; E \rightarrow (+0.05) Y; E \rightarrow (+0.03) Z)

本质相同：每个词对应一个相关词列表，每个相关词有对应的权重
该建模方式的限制：相关词列表不能太长

Baidu

其实归一化命中、同义词命中、主题命中都是对任何一个查询语句中的词，建立一个相关词表，每个相关词配上一个权重，如果文档中出现了任意一个相关词，都会得到大小不等的正向贡献。比如归一化命中贡献为1，同义词命中贡献为 0.8，主题命中的权重更小。

这样的建模方式有一个根本性的限制，就是相关词列表不能太大，否则模型大小会爆炸，无法实用。

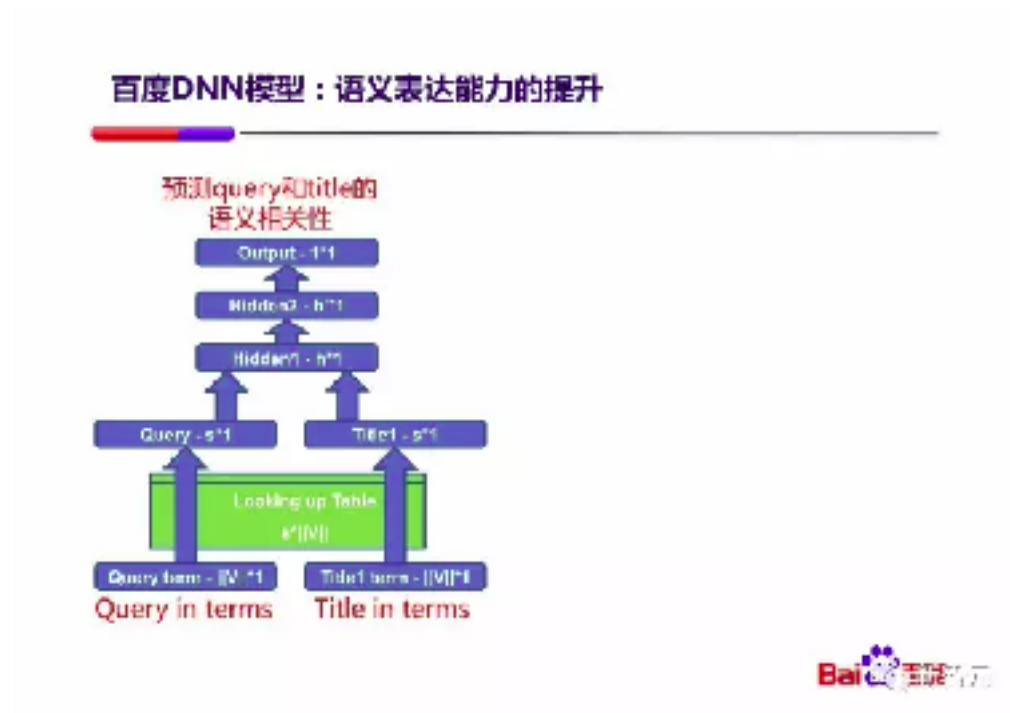
在这种方式的基础上我们还可以更进一步，就是建模词与词之间的反义关系，比如『爱/恨』，这样当某个词在查询中出现，那包含其反义词的文档，理论上应该进行适当降权。

建模词反义关系的做法与主题命中中的方法类似，任意两个向量都可以计算距离，而距离有正有负。

百度 DNN 模型

之后在 PPT 中，朱凯华更进一步讲了百度的做法，下图为百度 DNN 模型初版结构：

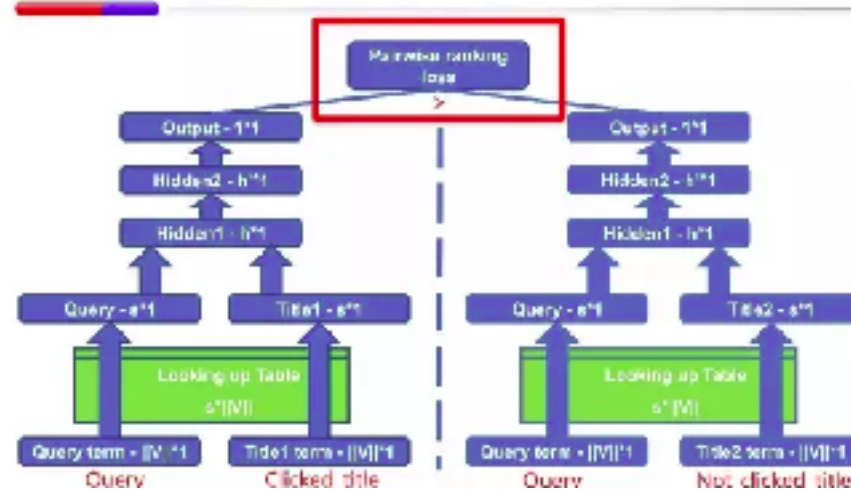
(/apps/download?
utm_source=sbc)



通过一个 DNN 模型来计算查询 Query 与用户最终点击的文档 title 之间的语义相关性。中间绿色的 Looking up Table 是 Word Embedding 层。

然后是进行 Pairwise loss 训练的 DNN 模型结构：

百度DNN模型：语义表达能力的提升



(/apps/download?utm_source=sbc)

前面说 SMT 模型（Word Translation Model）和 DSSM 都是基于用户点击数据进行训练的，这里 PPT 中给出了一种可行的利用用户点击数据的训练方式，其目的是建模同一个 Query 下有点击和无点击的文档标题之间的语义差异。

其实在《浅谈搜索引擎基础（上）》(<https://www.jianshu.com/p/c51416cf2849>)中，已经介绍过了机器学习排序算法下三种可行的训练方式，分别为：**单文档方法**、**文档对方法**、**文档列表方法**。而这三种方法的英文名称，就分别为 **Pointwise**、**Pairwise**、**Listwise**。

Pointwise 方法的主要思想是将排序问题转化为多类分类问题或者回归问题。也即预测查询语句与文档标题之间的相似度（回归问题），或预测查询语句与文档标题之间的相似级别（多类分类问题），然后根据相似度/相似级别进行排序。

Pairwise 方法是目前比较流行的方法，效果也非常不错。它的主要思想是将 Ranking 问题形式化为二元分类问题。也即处理的对象为文档对 $\langle \text{Doc1}, \text{Doc2} \rangle$ ，我们利用训练好的模型为 $\langle \text{Doc1}, \text{Doc2} \rangle$ 做二分类，一类是 Doc1 排序前于 Doc2，另一类则相反，依次来得到不同文档之间的排序。在训练时，就如 PPT 中所说的，以同一个 Query 下用

户点击与否作为训练数据的来源。举个例子，比如在同一个 Query 下，用户点击了 Doc1，那 $\langle \text{Doc1}, \text{Doc2} \rangle$ 的二分类结果就应当为 Doc1 排序前于 Doc2，由此我们就得到了带标注（labeled）的训练数据，也可以依此来做监督学习训练我们的 DNN 模型。

Listwise 方法相对于前两种（Pointwise，Pairwise）而言，不再将 Ranking 问题直接形式化为一个分类或者回归问题，而是直接对文档的排序结果（List）进行优化。至于优化方法，我们比较自然想到的是直接利用排序评价指标，比如《浅谈搜索引擎基础（上）》(<https://www.jianshu.com/p/c51416cf2849>)中讲过的 P@10 和 MAP，或者是前面提到的 NDCG，但是问题在于 NDCG 这样的评价指标通常是非平滑（连续）的，而通用的目标函数优化方法针对的都是连续函数。针对排序问题的优化函数有：

- **RankCosine**：使用正确排序与预测排序的分值向量之间的 Cosine 相似度（夹角）来表示损失函数
- **ListNet**：使用正确排序与预测排序的排列概率分布之间的 KL 距离（相对熵）作为损失函数

接下来 PPT 中以『ghibli车头如何放置车牌』为例讲解的 DNN 模型对搜索结果的影响：

(/apps/download?
utm_source=sbc)



在上线 DNN 模型之前，搜索上述 Query，搜索引擎完全不能理解 Query 的含义，毕竟 BM25 模型只考虑了 IDF 因子、文档词频、文档长度因子和查询词频，如果文档与 Query 中的词不重合，那 BM25 模型也没有办法。

(/apps/download?
utm_source=sbc) ×

而在上线了 DNN 模型之后，DNN模型成功捕获了 Query 中『车头』和 Document 中『前』、『前面』的语义关系；捕获了 Query 中『如何放置』和 Document 中『放哪里』、『怎么装』、『咋挂』的语义关系。

朱凯华讲，其实这是在训练的100亿样本中，有一些样本体现了这样的语义关系，被 DNN 模型学习到了。两个例子如上图，那是两个用户点击过的 Query/Document 对。

然后是进一步分析的例子：

(/apps/download?
utm_source=sbc) ×

如果 Query 和 Document1 都完全相同是『三岁小孩感冒怎么办』的时候，DNN 的预测分数是-15；如果我们在 Document1 后面加『宝宝树』，预测得分立刻大幅提高了（升到-13），虽然『宝宝树』这个词没有出现在 Query 中，但是DNN模型认为这个词和 Query其他部分是有语义相关关系的，所以给出了正向的贡献。如果我们在 Document1 后面加了『搜房网』，预测得分立刻下降了（降到-15.8），因为 DNN 模型认为这个『搜房网』的出现和 Query 其他部分语义更不相关，所以给出了负向的贡献。这个例子中，其实D2、D3 和 D1 的区别都是增加了一个和 Query 没关系的 Term，在传统的 Information Retrieval（信息检索）中是一定会给 D2、D3 一样的相关性得分，而区别不出两者的差异。

后面两个例子类似，DNN 能够区分『宝宝』和『小宝宝』类似，而和『狗宝宝』语义不相关。『励志格言』与『格言』类似，而『幼儿教师』和『幼儿』语义不相关。

然后朱凯华提到了 DNN 模型的发展方向：

(/apps/download?
utm_source=sbc) ×

至于图中提到的 SNE，是一种高维数据可视化工具：

(/apps/download?
utm_source=sbc) ×

百度 CNN 模型

将 CNN 模型应用在 NLP 中能够更好的建模**短距离依赖关系**，PPT 中给出了 CNN 模型与 BOW-DNN 模型的比较：

BOW-DNN 模型无法考虑词序，所以不能区分 Query 中的『葫芦岛北到北京西』和『北京西到葫芦岛北』，给出了一样的得分；而 CNN 能够知道哪个才是语义相关的。

第二个例子也是同理，CNN 能够区分『我被好友删除』和『好友被我删除』之间的语义差别。

CNN for NLP

在《浅谈深度学习基础（下）》(<https://www.jianshu.com/p/3d1ddfce1563>)中，其实已经详细的介绍了 CNN 的结构，但文中只是将 CNN 看做计算机视觉系统的核心结构，并没有提到 CNN 在 NLP 领域的应用。

这里我阅读了《UNDERSTANDING CONVOLUTIONAL NEURAL NETWORKS FOR NLP》(<https://link.jianshu.com?t=http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/>)以及李航老师等的 Convolutional Neural Network Architectures for Matching Natural Language Sentences (<https://link.jianshu.com?t=https://arxiv.org/pdf/1503.03244.pdf>)，对 CNN 在 NLP 领域的应用有了一个基本的了解。

(/apps/download?
utm_source=sbc)



(/apps/download?
utm_source=sbc)



在 NLP 中应用 CNN，CNN 的输入不再是图像像素，而是由句子或文档表示而成的矩阵。通常情况下，表示方法如上图所示，每一行代表一个词的词向量，这个词向量可以通过 word2vec 等得到的 Distributed Representation 的词向量，甚至也可以是利用 one-hot representation 的词向量。这个矩阵，或者说『图像』的行数是单词的个数，而列数等于词向量的维数。

接下来是卷积过程，图中所用的六个卷积核宽度均与矩阵的宽度相同，否则会将同一个词的词向量割裂，这是与图像识别中的卷积核不同的地方；另外，它们的高度分别为 4 个词、3 个词和 2 个词，它们的卷积步长为 1 个词。由此得到 6 张不同的 feature maps。

之后对 feature maps 进行最大池化，记录下每张 feature map 中的最大值；随后将其连接起来，组成一个特征向量；最后对这个特征向量进行处理，得到我们所需的分类结果。

在 CV（Computer Vision，计算机视觉）中使用 CNN，低层抽象与高层抽象的物理意义是较为明显的，一般低层抽象发现图像的边缘等特征，高层抽象是低层抽象的组合，抽象程度更高，所记录的特征也更复杂。而在 NLP 中，这样的物理意义似乎并不太明显，因为具有相关关系的词并不一定紧邻着出现在句子中，而图像中同一个物体一般会由相邻的像素来表示。

CNN 在 NLP 领域有一些常见应用，比如情感分析、垃圾信息监测、主题分类、关系抽取和关系分类等。

李航博士的论文之中提到了 CNN 在语义匹配中的应用，提出了**卷积语句模型（Convolutional Sentence Model）**和两个**卷积匹配模型（Convolutional Matching Models）**：ARC-1 和 ARC-2。

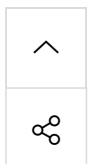


(/apps/download?
utm_source=sbc) ×

其中，卷积语句模型的示例如上图，其实与前文提到的 CNN 在 NLP 中的基本用法相类似，将句子表示成以词数为行数、词向量维数为列数的矩阵，对其进行多次卷积和池化，最终得到固定长度的特征向量。

需要特别注意的是，图中第一次卷积的结果，一竖列为一张 feature map，竖列的个数为不同卷积核的个数。De 为词向量的维数，F1 为不同卷积核的个数。

卷积匹配模型 ARC-1 其实原理很简单，就是拿两个句子分别过一下上面的卷积语句模型，得到特征向量，再将这两个特征向量输入到一个 MLP（multi-layer perceptron，多层感知机）中，去计算匹配度，其示意图如下：



不过有人实验后提出，如果直接像图中一样连接两个向量作为输入送给 MLP，效果并不好，因为会丢失边界信息，MLP 不知道 a 与 b 两个特征向量的分界在哪，更优的方式是将 [a, b, aTWb] 整体作为输入给到 MLP，一定要有 aTWb 这项。

(/apps/download?
utm_source=sbc) ×

相比较而言，ARC-2 的结构反而与 CNN 在 CV 中的应用更相似一些，其示意图如下：

ARC-2 尝试让两个句子在得到像 ARC-1 结果那样的高层抽象表示之前就进行相互作用，不再先分别通过 CNN 结构得到各自高层抽象表示。

可以这样理解，ARC-2 将两个句子看做了二维图像的两个维度，两个一维的句子结合成了一张二维的『图片』，然后将这张特别的『图片』作为 CNN 结构的输入。

这导致在 ARC-1 模型中，一张 feature map 仅仅是一个列向量，或者说是一个一维矩阵，若干个列向量并在一起形成了 ARC-1 示意图中的模样（二维），而在 ARC-2 中，一张 feature map 成为了一个二维矩阵，若干个二维矩阵叠在一起形成了 ARC-2 示意图中的模样（三维）。

再之后的卷积、池化过程就与 CV 中 CNN 的卷积、池化过程类似了。

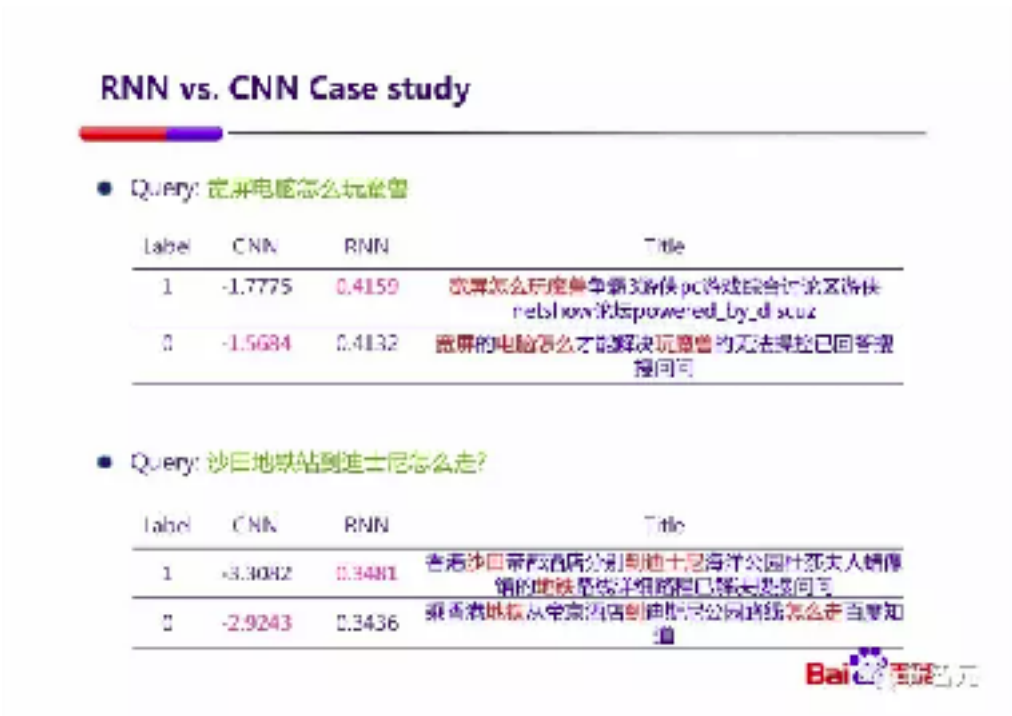


需要注意的地方，第一个仍然是，在 1D 卷积层中，涉及到将两个词向量连接，直接连接仍然会存在丢失边界信息的问题，采用前文提到的方案可以得到更优结果。另外，直接用传统 Word Embedding 方法得到的词向量来表示句子作为 CNN 的输入可能不是最佳方案，利用已经过了 LSTM 的 hidden state 效果可能会更好。

(/apps/download?utm_source=sbc)

百度 RNN 模型

通过 RNN 可以建模长距离的依赖关系：



RNN Language Model

在《浅谈深度学习基础（下）》(<https://www.jianshu.com/p/3d1ddfce1563>)中，简单提及了 RNN 并对 LSTM 的结构进行了较为详细的介绍，但是对 RNN 的应用提及不多，这里主要参考这篇《The Unreasonable Effectiveness of Recurrent Neural Networks》

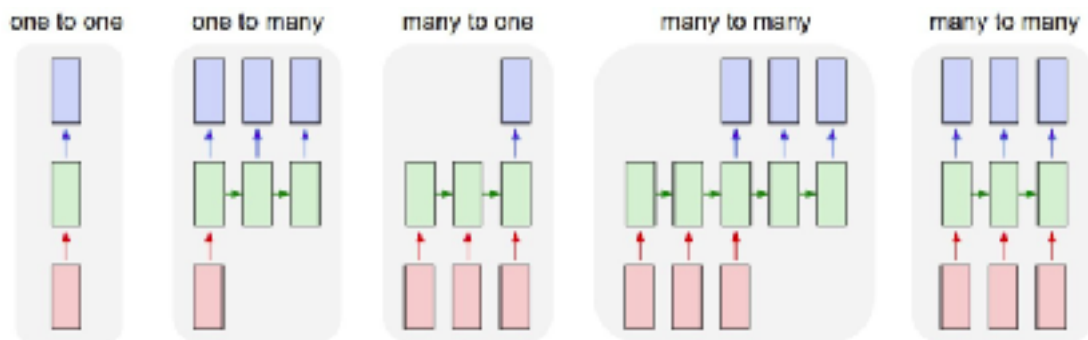
^

🔗

(<https://link.jianshu.com?t=http://karpathy.github.io/2015/05/21/rnn-effectiveness/>)，以一个基于 RNN 的简单的字符级语言模型（Character-Level Language Models）为例来介绍 RNN 语言模型。

对于 RNN，我们需要知道，RNN 的当前输出并不只取决于当前输入，前一个隐层状态与当前输入共同决定了当前隐层状态，而当前隐层状态决定了当前输出。而且 RNN 的输入输出均为序列：

(/apps/download?utm_source=sbc) ×



我们希望能利用 RNN 搭建一个简单的语言模型，这个语言模型拥有一个只包含四个字符『h、e、l、o』的单词表。我们希望能通过训练使得该模型拥有输出『hello』的能力，具体来讲，也即当上文为『h』时输出『e』，上文为『he』时输出『l』，上文为『hel』时输出『l』，上文为『hell』时输出『o』。

下图为这个 RNN 语言模型的示意：



(/apps/download?
utm_source=sbc)

RNN 的输入输出采用的是 one-hot representation 的词向量，最开始的隐层状态被初始化为零向量，上一隐层状态与 W_{hh} 相乘后与当前输入向量与 W_{xh} 相乘后的结果相加后过一个 tanh 激活函数，再与 W_{hy} 相乘得到输出向量。输出向量中拥有最大权值的分量所对应的字符被视为输出字符。

我们希望通过训练使得输出向量中绿色加粗的权值能够尽可能大，而红色权值能够尽可能小。

另外，从图中可以发现，当输出为『e』时，hidden layer 向量中第三分量最大，而输出为『l』时，第一分量最大，输出『o』时，第二分量最大。所以 RNN 语言模型同样可以产生类似于神经概率语言模型词向量的 Distributed Representation 的词向量，并且，前文中也确实提到过，如果将 CNN 的输入替换为 LSTM 的 hidden state 可能会得到更好的结果。

还有，我考虑这样一个问题，如果 RNN 语言模型的输入输出用的不是 one-hot representation 而是 Distributed Representation，然后输出就是一个 Word Embedding 的结果，我们找到词向量空间中与输出向量最接近的 k 个词，然后为其分配一定的概率来决定输出字符，这样会不会有更好的结果？（我并没有找到对应的论文，也不知道这个想法是否已经被实践了，只是顺着思考下去）

《The Unreasonable Effectiveness of Recurrent Neural Networks》

(<https://link.jianshu.com?t=http://karpathy.github.io/2015/05/21/rnn-effectiveness/>)原文中另外采用莎士比亚的文集训练了 RNN 语言模型，也得到了不错的结果。

知识推理

知识图谱

我们希望通过引入『知识』，利用推理能力，能够更直接的给出用户答案，而不是给出用户相关网页的集合。

为了实现这样的目的，第一个方向就是采用知识图谱推理：

(/apps/download?
utm_source=sbc)



《浅谈知识图谱基础》(<https://www.jianshu.com/p/4f09043e22ea>)对知识图谱的结构以及相关技术架构做了一个粗略的介绍，这里再简单回顾一下：

知识图谱的逻辑结构分为两个层次：**数据层**、**模式层**



数据层通常以『实体-关系-实体』或者『实体-属性-值』三元组作为事实的基本表达方式。

模式层通常采用本体库来管理，存储的是经过提炼的知识。

知识图谱的构建技术分为三个步骤：**信息抽取、知识融合、知识加工**

信息抽取是一种自动化地从半结构化和无结构数据中抽取实体、关系以及实体属性等结构化信息的技术。涉及的关键技术包括：**命名实体识别、关系抽取和属性抽取**。

知识融合对数据进行清理和整合，包含 2 部分内容：**实体链接和知识合并**。实体链接的主要技术为**实体消歧和共指消解**。知识合并指从第三方知识库产品或已有结构化数据获取知识输入，常见的知识合并需求有两个，一个是**合并外部知识库**，另一个是**合并关系数据库**。

知识加工用于获得结构化，网络化的知识体系，主要包括3方面内容：**本体构建、知识推理和质量评估**。

本体构建包含 3 个阶段：**实体并列关系相似度计算、实体上下位关系抽取以及本体的生成**。

知识的推理方法可以分为2大类：**基于逻辑的推理和基于图的推理**。基于逻辑的推理主要包括**一阶逻辑谓词、描述逻辑**以及**基于规则的推理**。基于图的推理主要基于**神经网络模型或 Path Ranking 算法**。

质量评估可以对知识的可信度进行量化，通过舍弃置信度较低的知识，可以保障知识库的质量。

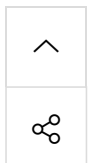
PPT 中显示百度支持不同种类的基于知识图谱的推理，比如『主语+谓词』，比如『刘德华的老婆』：

(/apps/download?
utm_source=sbc)



(/apps/download?
utm_source=sbc) ×

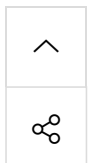
支持『谓词+取值范围』方式的推理，比如『180cm以上的男明星』：



支持『谓词+排名』方式的推理，比如『世界第五/第八/第十高峰』：

(/apps/download?
utm_source=sbc) ×

还有多步推理，比如『谢霆锋的爸爸的儿子的前妻的年龄』：



(/apps/download?
utm_source=sbc) ×

另外，需要注意的是，有些问句需要先对其进行句法分析，才能确定如何查询知识图谱，比如：

百度采用了依存语法对用户问句进行句法分析。

依存句法分析与语义依存分析

《浅谈自然语言处理基础（下）》(<https://www.jianshu.com/p/872a72e861fd>)对依存句法分析进行过较为详细的介绍，这里结合哈工大语言云 (https://link.jianshu.com?t=http://www.ltp-cloud.com/intro/#sdp_how)再回顾一下。

****依存句法分析 (Dependency Parsing, DP) ****通过分析语言单位内成分之间的依存关系揭示其句法结构。直观来讲，依存句法分析识别句子中的『主谓宾』、『定状补』这些语法成分，并分析各成分之间的关系。示意图如下：

在查找文献的过程中，我发现了比依存语法分析更进一步的语义依存分析，语义依存分析是在我之前的文章中没有提到过的。

语义依存分析 (Semantic Dependency Parsing, SDP)，分析句子各个语言单位之间的语义关联，并将语义关联以依存结构呈现。使用语义依存刻画句子语义，好处在于不需要去抽象词汇本身，而是通过词汇所承受的语义框架来描述该词汇，而论元的数目相对词汇来说数量总是少了很多的。语义依存分析目标是跨越句子表层句法结构的束缚，直接获取深层的语义信息。其示意图如下：

(/apps/download?
utm_source=sbc) ×

与依存句法分析相对比，语义依存分析所得到的结果更能反映句子的语义。依存语法分析识别句子中的『主谓宾』、『定状补』这些语法成分，而语义依存分析着重寻找的是：**主要语义角色、事件关系、语义依附标记**。

常见的语义依存关系如下：

(/apps/download?
utm_source=sbc) ×

知识图谱推理的两大限制及三种解决方案

基于知识图谱推理有一些关键的限制，限制之一是知识图谱有**盲区（Blindness）**，图中是一个 bush wives 的例子：

(/apps/download?
utm_source=sbc) ×

一旦某概念不在知识图谱中，知识图谱会做一个它自己尽可能好的解释，但它可能和真正的解释相差很多。比如『从百草园到三味书屋』，确实有地方叫百草园，也有地方叫三味书屋，一个只关注地理位置 POI 的知识图谱就会觉得他是一个完美匹配导航需求的 Query。但放大来看，大家都知道他是鲁迅的一篇散文。

基于知识图谱推理的第二个限制是静态的图谱很难描述**用户意图的分布以及变化**：



(/apps/download?
utm_source=sbc) ×

举例来说，『天龙八部』这四个字，可以是小说，游戏，电视剧，电影。并且如果当前有一部电影叫做『天龙八部』上映了，那搜索结果也应该做适当的调整。

只用知识图谱向用户提高答案是不够的，通常只能覆盖 1%~2% 的 Query。

而搜索能够帮助知识图谱解决盲区和意图分布的问题：



(/apps/download?
utm_source=sbc) ×

搜索通过为知识图谱提供上下文，帮助其解决盲区所带来的问题，并可以更好的理解用户的真实意图。而知识图谱能够帮助搜索引擎理解搜索的 Query 和结果，能够更直接的给予用户答案。

知识图谱与搜索结合的常见方式有三种：**知识图谱+搜索+长答案**、**知识图谱+搜索+结果聚合**、**知识图谱+搜索+知识实体**。

知识图谱+搜索+长答案

朱凯华讲这是自动识别**最佳答案段落**（Paragraph Ranking）的问题，其实与**自动摘要**（automatic summarization/abstracting）技术的原理类似。

这里要区分一下自动摘要与前面提到过的信息抽取，自动摘要是利用计算机自动实现文本分析、内容归纳和摘要自动生成技术，而信息抽取则是从自然语言文本中自动抽取指定类型的实体、关系事件等事实信息的应用技术。

自动摘要常见的方法有两种，一种是**基于图的方法**，另一种是**基于特征的方法**。

基于图的方法将文档的每句话作为节点，句子之间的相似度作为边权值构建图模型，用 Page Rank 算法进行求解，得到每个句子的得分。边的权重隐式定义了不同句子间的游走概率。这些方法把做摘要的问题看成随机游走来找出稳态分布（Stable Distribution）下的高概率（重要）的句子集。基于图的自动摘要算法的代表有 TextRank 和 LexRank。

基于特征的方法通过考虑句子长度、句子位置、句子是否包含标题词等特征来发现文档中的关键句，代表算法是 TextTeaser。

但前面的方法往往都只考虑了相关性而没考虑多样性，排序后得到结果的缺点之一便是无法避免选出来的句子之间相似度极高的现象，可能排名靠前的几句话表达的都是相似的意思。

针对句子过于相似的现象，有人提出引入惩罚因子，这就是 **MMR（Maximal Marginal Relevance，最大边缘相关模型）** 算法。基于 MMR 的变种，有人提出了玻森新闻自动摘要算法 (<https://link.jianshu.com?t=https://segmentfault.com/a/1190000004265070>)，可以方便的平衡句子的相关性和多样性，并提供了可供调用的 新闻自动摘要 API (<https://link.jianshu.com?t=http://docs.bosonnlp.com/summary.html>)。

知识图谱+搜索+结果聚合

(/apps/download?
utm_source=sbc)



(/apps/download?
utm_source=sbc) ×

结果聚合这里用到了**情感分析（Sentiment Analysis）**的技术。因为有些问题可能目前并没有一个权威的答案，众说纷纭。我们就可以通过情感分析了解，有多大比例的人持赞同意见，以提供给用户参考。

情感分析由易到难分为三个层次：

- 分辨积极/消极情绪（PPT 中的做法）
- 为情绪评级，由消极到积极
- 检测出情绪的主体、客体以及复杂的情绪类型

情感文本分类可以大致分为**有监督学习方法**、**半监督学习方法**和**无监督学习方法**三类。

有监督学习方法利用大量标注样本，并结合各种特征和特征权值选择策略，使用机器学习方法进行情感分类。半监督学习方法通过少量标注样本进行初步训练，之后利用大量未标注样本进行学习，以此构建分类模型。无监督学习方法仅使用非标注样本，比如可以利用一些情感分类标注中的种子词集来实现无监督分类。

知识图谱+搜索+知识实体



(/apps/download?
utm_source=sbc) ×

将搜索与知识图谱结合可以处理一些对于知识图谱而言难度很高的问题。比如『王刚第二任妻子是谁』这个 Query 其实对知识图谱理解的要求很高：要理解这个第二，不是年龄第二，不是身高第二，而是结婚时间排序第二。

利用语义分析可以得出『是谁』询问的是一个人；利用搜索可以找到与该 Query 相关的上下文内容；利用知识图谱可以找到上下文中哪些实体是人。

朱凯华讲，百度可以直接回答用户 7% 的 Query：



对话式OS

概述

我一直持有一个观点：对话交互是人机交互的未来，因为对话交互就是当前人与人沟通最重要的方式。

我们创造了计算器，可以类比于发现了一个语言不通的原始部落，我们需要先去学习他们的语言，这是沟通的基础。而当我们学会了他们的语言之后，就要开始逐步教会他们我们的语言，因为作为一个高级文明，我们语言的沟通效率要更高，并且对于人类来讲，沟通门槛更低，也更自然。

而且实在难以想象，我们未来对家里的台灯、洗衣机、扫地机器人等智能设备下命令时，还要走到它身边按按钮。

阿里小蜜的分析文章将对话系统分成了两层，分别是：**意图识别层**、**问答匹配层**。

意图识别层识别语言的真实意图，将意图进行分类并进行意图属性抽取。意图识别层是一个结合上下文数据模型与领域数据模型不断对意图进行明确和推理的过程，决定了后续的领域识别流程。

问答匹配层对问题进行匹配识别及生成答案，经典的问题类型有三种：

- **问答型**：例如『密码忘记怎么办？』
- **任务型**：例如『我想订一张明天从杭州到北京的机票』
- **闲聊型**：例如『我心情不好』

特别的，我个人把这类虚拟个人助理（Virtual Personal Assistant，VPA）执行任务的过程分为三个阶段：

- **理解阶段**：即理解人类的需求，常用技术有语音识别、图像识别、自然语言理解等。
- **思考阶段**：即寻找需求的解决方案，常用技术有搜索引擎、推荐系统、知识图谱等。
- **执行阶段**：即完成人类的需求，常用的技术有自然语言生成、智能家居/物联网以及目前被用来满足需求的其它技术。

并且，不同种类的任务对于不同阶段的侧重程度也不同，比如『明天下午三点提醒我去开会』、『后天杭州冷吗，会不会下雨』侧重于理解；『什么样的衬衫比较适合我』、『第30任美国总统是谁』侧重于思考；『开始扫地/洗衣服/烧水』、『开灯/关灯』侧重于执行。

对于『对话式OS』这部分，下文准备先讲解一些问答领域的关键技术，再针对上文提到的经典问题类型分场景来讨论解决方案。

DuerOS 与 POMDP 框架

(/apps/download?
utm_source=sbc)



PPT 中提到了 POMDP 框架，**POMDP (Partially Observable Markov Decision Processes, 部分可观测马尔科夫决策过程)** 是 **MDP (Markov Decision Processes, 马尔科夫决策过程)** 的扩展，而 POMDP、MDP 与浅谈自然语言处理基础（上）(<https://www.jianshu.com/p/c123c7534500>)中讲过的 MC (马尔可夫链) 和 HMM (隐马尔科夫模型) 又有一定相关性，示意图如下：

(/apps/download?
utm_source=sbc)



我们容易理解 MC 到 HMM 是状态由可见变为了不可见；由 MC 到 MDP 多考虑了动作，我们可以以下棋为例，每个状态到下个状态的转移不仅和当前状态有关，还和当前状态我们采取的动作有关；由 MDP 到 POMDP 的过程同样是状态由可见变为不可见的过程，也同样举个例子，MDP 中，在明确当前状态的前提下，与动作结合后，确定下一个状态，而 POMDP 中，我们并不明确知道当前状态，只知道当前状态不同可能性的概率，这样我们的决策过程就要考虑到当前状态的不同可能性。

下文将会讲到的强化学习还会重新提到 MDP 这部分内容。

朱凯华提到，从整个对话系统来说，DuerOS 的整体是符合 POMDP 的框架的。采用 POMDP 的框架有很大的好处：

- 整个系统可以整体建模每个环节的出错可能性
- 整个系统可以整体建模几轮对话后用户最终给予的反馈

PPT 中还提到了整个『对话式 OS』系统的基本组成部分，从 ASR (自动语言识别) 到 NLU (自然语言理解) 到 DST (对话状态跟踪) 到 DM (对话管理)，之后选择直接执行动作或者进行 NLG (自然语言生成) 和 TTS (语音合成)。



按照我之前的任务过程的三个阶段来划分，图中的 ASR、NLU、DST 是理解阶段，DM 是思考阶段，Action、NLG、TTS 是执行阶段，也是完全符合我的构想的：)

ASR 与语音纠错

(/apps/download?
utm_source=sbc) ×

PPT 中还介绍了百度的语音纠错技术，利用了RNN的序列标注能力，来找到语义上可能的替换点，然后通过上文提到过的 Skip-gram 检索最终产生正确的替换。

但是作为 PM，我个人对这个功能持保留意见，这个功能用起来很蛋疼：

- 一是整个用户群体对这种功能的认知程度不高，不了解这个功能
- 即便了解这个功能，还需要会拆字，对用户来说太麻烦，学习成本高
- 即便会拆字，怎么让用户相信他拆字说的那一大段话你都能识别正确，心理成本太大



- 而且用户更正的方式确实是千奇百怪多种多样的，理论上是个无限集，怎么保证识别率

所以这样的情况理论上式存在的：用户说了一句话，出现了识别错误，用户语音纠错，结果纠错语句被识别错误，用户继续语音纠错，结果越来越错，从今往后用户再也不会用这个功能了。

对于这类 VPA，你能做到什么功能并不是最重要的事情，最重要的事情是让用户相信，通过你来满足这个需求的预期心理成本要低于他之前的做法，这样他才会来用你。而这种预期心理成本又包括了解成本、学习成本、信任成本以及你产品功能未达预期的损失成本等等。

综上，这部分技术我不再做详细介绍。

NLU 与 Slot Filling

(/apps/download?
utm_source=sbc) ✕



Slot Filling 是完成之前提到的任务型问题的重要技术，百度采用一个字符级的双向 LSTM 和 CRF 层来做序列标注，再利用序列标注的结果做填槽。

这里先来说一下**双向 LSTM**，LSTM 结构在浅谈深度学习基础（下）(<https://www.jianshu.com/p/3d1ddfce1563>) 中有过详细的讲解，这里就不重复了，这里的重点主要放在**双向**上。

我们知道 RNN 的一个突出优点是能够解决长距离的依赖关系，这对于自然语言处理是非常重要的，因为语言中一词多义的情况很多，很多时候没有上下文就无法对词的真实含义进行准确的判断。但是普通的 LSTM 存在一个问题，就是只能利用上文中的信息，而如果能像访问过去的上下文信息一样，访问未来的上下文，这样对于许多序列标注任务是非常有益，这就是双向 LSTM 的基本思想。

(/apps/download?utm_source=sbc) ×

上图为双向 RNN (BRNN) 的结构示意图，与前文中的单向 RNN 的示意图比较，可以发现，双向 RNN 中的输出不再只取决于 Forward Layer 中的隐层状态，而是同时取决于 Forward Layer 与 Backward Layer 中的两个隐层状态；当前输入也同时影响 Forward Layer 与 Backward Layer 对应位置的隐层状态。

而双向 LSTM，就是将 BRNN 中的普通 RNN 单元更换为 LSTM 单元。

百度的做法如下图所示：

这里举了一个例子『王乐君伪装者观看』，先经过**自动分词**（浅谈自然语言处理基础（中）(<https://www.jianshu.com/p/3bb214a200ca>)中有详细介绍）拿到划词边界，并以 0/1 的方式存储。

随后将每个字进行 Word Embedding，拿到字向量，作为双向 LSTM 的输入，而从图中来看，双向 LSTM 的输出是一个具有两个分量的向量。

(/apps/download?
utm_source=sbc)



仔细观察的话可以发现，CRF layer 中，对『剧名』和『演员』的标注是以 IOB 方式进行的：I 为词中、B 为词首、O 为词外。实际上在浅谈自然语言处理基础（下）(<https://www.jianshu.com/p/872a72e861fd>)中，就提到，基于语块的语义角色标注方法利用 IOB 等方式将语义角色标注作为一个序列标注问题来解决。而且基于语块的语义角色标注方法一般没有论元剪除这个过程，因为 O 相当于已经剪除了大量非 base NP，也即不可能为论元的内容。论元辨识通常也不需要了，base NP就可以认为是论元。

看起来，在 CRF Layer 中，序列标注是这样进行的。首先有多少个/种类的槽需要填写，LSTM 的输出向量中就存在多少个分量，然后在 CRF Layer 中，将向量进行拆解，不同输出向量中相同位置的分量被放在一起处理，作为该特定槽位所需的序列标注素材。

至于 CRF（条件随机场），在浅谈自然语言处理基础（中）

(<https://www.jianshu.com/p/3bb214a200ca>)中有过详细介绍，这里就不重复了。

DST（对话状态跟踪）、上下文替换、主体补全

(/apps/download?
utm_source=sbc)



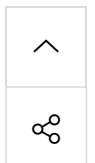
对话状态跟踪就是根据多轮的对话来确定用户当前的目标（user goal）到底是什么的过程。

在封闭域的任务型问题中，对话状态跟踪以槽位的填写为核心，下文会有讲解。这里主要说一下开放域中的状态跟踪：

(/apps/download?
utm_source=sbc) ×

开放域中状态追踪的核心是**上下文替换**、**主体补全**。从图中来看，上下文替换将上文中的动作加给当前句子中的主体，而主体补全为上文中的主体执行当前句子中的动作；上下文替换是去上文中找动作，主体补全是去上文中找主体。

这两个经典问题很适合用来测试开放域 chatbot 的多轮对话能力，我测试了一下图灵机器人：



(/apps/download?
utm_source=sbc) ×

发现它在上下文替换方面做过优化，而主体补全做的就不好了：

回答完全不知所云。

这里还要与知识图谱中的实体消歧和共指消解做一下区分，实体消歧是区分同名实体，共指消解是为实体的代称找到对应的实体，比如『他/它/她/这』是什么；而这里的上下文替换和主体补全的场景是对话，而不是文档，很多事情代词也会被省略，仅通过共指

消解无法起到这里主体补全的作用。当然也并不是一点作用都起不到，只是上下文替换和主体补全更适合对话的场景。

这里重新放一下百度的 PPT：

(/apps/download?
utm_source=sbc)



这里采用了 BIE 的标注方式，与前文中的 IOB 相比，多了对词尾的标注 E。

百度通过 RNN + CRF 来做到的状态更新：用户顺序说Q1，Q2，百度看到Q2会直接更新理解为Q3。从图中来看，上下文替换/主体补全的基础和填槽一样，仍然是序列标注，BIE 标注识别主体，如果句子中没有主体就从上文找；如果句子中只有主体，就代入到上一句，替换掉上一句中的主体。

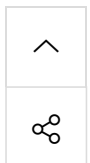
seq2seq、Encoder-Decoder、Attention Model



既然谈到了开放域的对话问题，就不得不说到 seq2seq (<https://link.jianshu.com?t=https://github.com/google/seq2seq>) 了。seq2seq 是 Google 开源推出的一个具体实现了 **Encoder-Decoder** 框架，并结合了 **Attention Model** 的一个序列转换模型，可以用于机器翻译、文本总结、会话建模、图像字幕等场景，其示意图如下：

(/apps/download?
utm_source=sbc) ×

先说什么是 Encoder-Decoder 框架，Encoder-Decoder 框架可以看作是一种文本处理领域的研究模式，其最抽象的一种表示如下图所示：



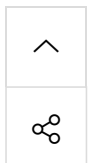
如果采用 RNN 来实现 Encoder-Decoder 框架，一种常见的模型配置如下图所示：

(/apps/download?
utm_source=sbc) ×

上图所示的模型客观上存在两个问题：

- 第一个是，输入序列的所有信息都需要靠图中 C 的记忆模块来存储，C 的记忆模块实际上成为了整个网络性能的瓶颈
- 第二个是，以翻译场景为例，将『Tom chase Jerry』翻译为『汤姆追逐杰瑞』，在翻译『汤姆』时，『Tom』和『chase』、『Jerry』的权重相同，这显然是不应该的

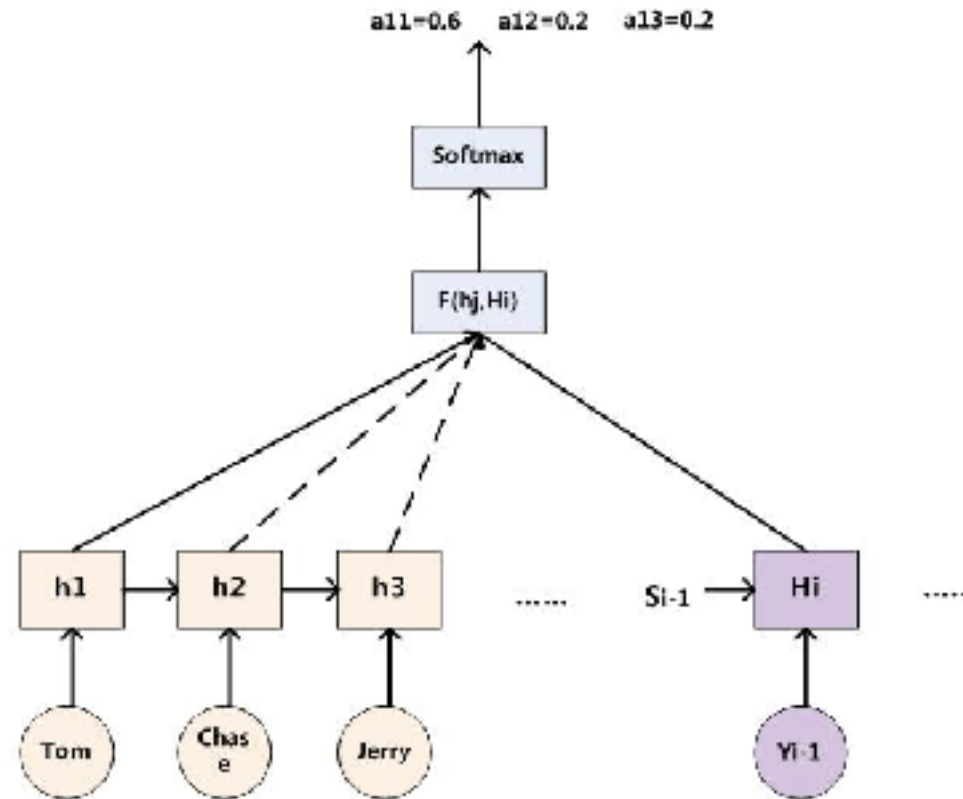
依照这样的思路，Attention Model 被提出了，其示意图如下：



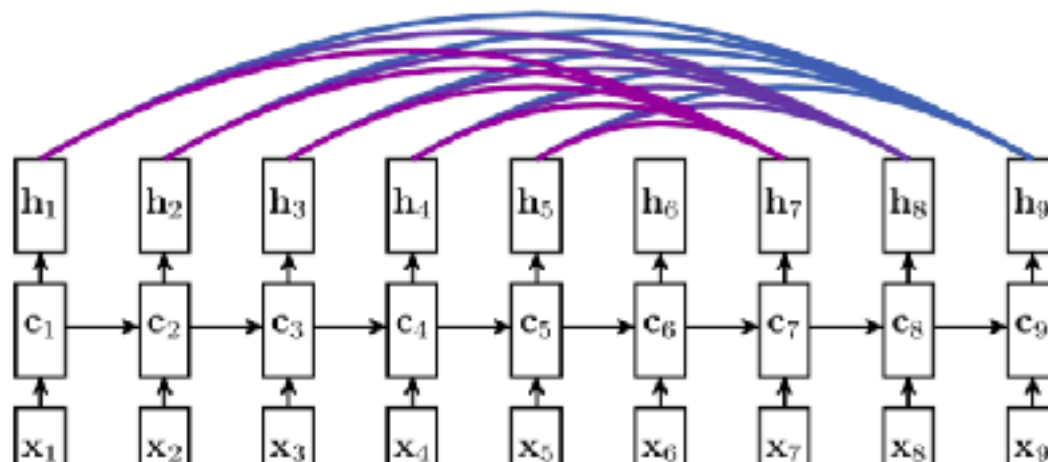


以 RNN 为例，将其细化，如下图：

(/apps/download?utm_source=sbc) ×

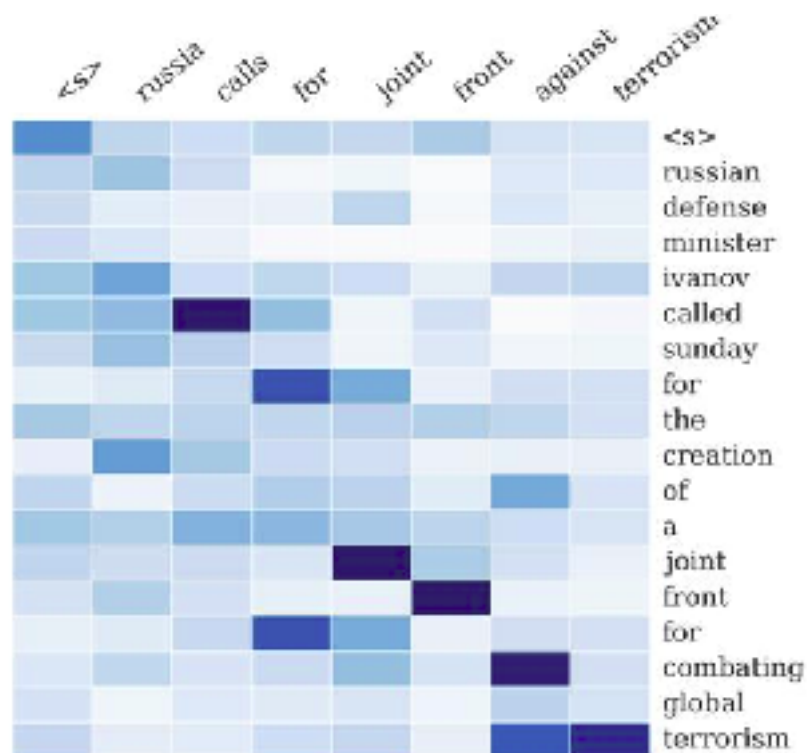


输出序列中的每一个字符产生时，不仅考虑了当前输入（实际上通常为输出序列的上一个输出字符）和上个隐层状态，还为输入序列的输出（下图中的 h_1 到 h_5 ）分配了不同权重，一并考虑进来：



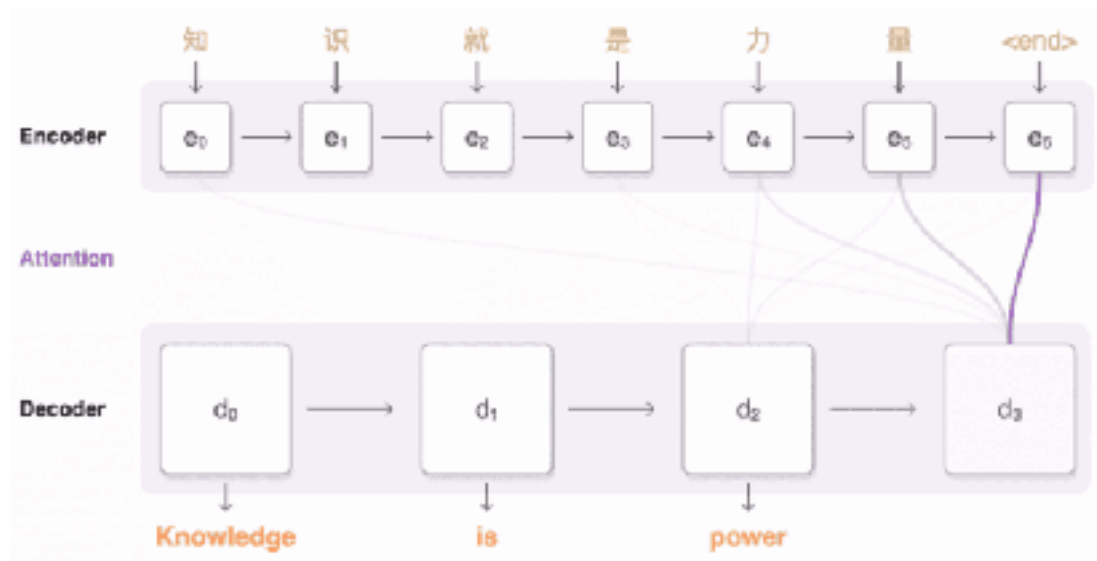
(/apps/download?utm_source=sbc) ×

一般文献中会把 Attention Model 看做单词对齐模型，这在机器翻译的场景下表现的尤为突出：



输入序列中的不同字符对输出序列产生某一特定字符所起到的作用显然应该是有区别的，就像读英文篇章做阅读理解，一道题目的答案通常也出自篇章中特定的几句话，将注意力平均分配给所有内容是不应该的。

(/apps/download?utm_source=sbc) ×



在 Google 提供的 seq2seq 的示意图中，紫色的粗细代表了输入序列中的不同字符对当前输出序列产生字符的影响大小。

DM（对话状态管理）与 Reinforcement Learning（强化学习）



(/apps/download?
utm_source=sbc) ×

在对话状态管理阶段，主要是系统需要选择合适的 Action 来动作。

PPT 中举了一个度秘高考的例子，通过与考生对话，引导考生进一步了解他关心学校的信息，来最终做出决定。

百度把这个 Dialogue management 的问题构建成一个 MDP 问题（由于在实际产品中并不知道考生最终填报了哪个志愿，Reward 被设计成每次交互都会给出显示的奖惩），放在经典的 Reinforcement Learning 的框架中来求解。

接下来介绍 Reinforcement Learning，针对强化学习，南京大学的俞扬博士有过两篇文章：强化学习前沿（上）(<https://link.jianshu.com?t=http://www.leiphone.com/news/201705/NITc7oObBqh116Z5.html>)、强化学习前沿（下）(<https://link.jianshu.com?t=http://www.leiphone.com/news/201705/uO8nd09EnR77NBRP.html>)，非常详尽，我这里只是概要的介绍一下。

其实我在第一篇文章浅谈机器学习基础（上）(<https://www.jianshu.com/p/ed9ae5385b89>)中就提到过强化学习。可以这样理解，强化学习与训练动物的过程类似，动物面对一个未知环境，驯兽师发出指令后，如果动物做出了正确的行为，就会得到奖励。强化学习也是同理，是一个让机器学习正确动作并最大化总奖励的过程。

强化学习与监督学习不同，如下图所示：

(/apps/download?utm_source=sbc) ×



监督学习是一个**开环**的学习过程：

- 通常，监督学习任务会从环境中收集一批数据
- 接着我们用监督学习算法从数据中产生模型
- 最后就可以用这个模型来做预测了

而强化学习，是一个**闭环**学习过程：

- 首先，也是从环境中产生数据；
- 用强化学习的算法从数据中产生模型；
- 还要把模型放回到环境中运行，接着又会产生新的数据出来，再重复以上步骤。

总结起来，两者最核心的区别，在于**强化学习需考虑自身对环境的影响**。

强化学习的历史非常悠久，其中，早期的强化学习和它的一个数学模型 MDP 有很大关系，也就是前文中提到过的马尔科夫决策过程，其示意图如下：

(/apps/download?
utm_source=sbc)



(/apps/download?
utm_source=sbc) ×

马尔科夫决策过程里有一个四元组，即**状态、动作、奖赏、转移**；每个状态可能做不同的动作，所以转移概率也不同。

早期的强化学习是完全以 MDP 为数学基础的，对它来说也是要找一个策略，这个策略就是选择不同动作会有不同的概率，或者是确定性策略，在一个状态就输出一个动作。

求解马尔科夫决策过程上的最优策略，就是找到马尔科夫决策过程中总回报最大的策略。

在计算回报时，会涉及到 **V 值函数**和 **Q 值函数**。

V 值的含义是从当前状态出发，经过足够长时间之后，所估算出的该步的回报是多少。但是如果只计算 V 值，从中导出策略并不方便，它表示的是总的回报，而我们需要选择动作，也即知道哪个动作所导致的下个状态的 V 值更好。

为了避免麻烦，我们引入 Q 值函数，Q 值函数比 V 值函数多了一个动作输入，它要估计的是在当前做了动作 a 以后，再跟着这个策略 π 一直做下去，它的回报是多少。有了 Q 值函数，看到状态 s 后，把每个 a 带进去，看哪个 a 出来的 Q 值大，就用哪个 a。所以这样就可以在当前状态直接决定用哪个动作了。

Q 值和 V 值是有直接的对应关系的，如果按照策略来选择动作，平均的 Q 值就是 V 值。

寻找最优策略通常有两种方法：

- 第一种方法：首先评估给定一个策略的 V 值，然后找一个方向来提高这个策略，重新计算 V 值来做策略迭代

- 第二种方法：直接通过V值来更新V值，这一方法称为值迭代

上面讲了 MDP，但是 MDP 并不是强化学习，因为它的四元组都已给出，特别是奖赏和转移。你任给它一个状态和动作，都可以算出奖赏值；转移也是，输入一个状态、动作以后，它会告诉你转移出来的状态是什么。

而在强化学习任务中，奖赏和转移都是未知的，需要通过学习得出。具体解决办法有两个，一个是**基于模型（Model-Based）**的方法，另一个是**免模型学习**。

基于模型方法就是构建出强化学习所对应的 MDP 模型，再在 MDP 模型的基础上去求解策略。从随机策略开始，把策略放到环境中运行，从运行的序列数据中把 MDP 抽象出来。因为序列数据可以提供环境转移和奖赏的监督信息，简单的做一个回归，就能知道一个状态做了一个动作下面会转移到哪儿，以及能得到的奖赏是多少。但是基于模型的方法计算起来往往比较复杂，现在的工作大多专注在免模型学习上。

免模型学习和之前讲到的策略迭代的思路很像，首先，评估当前策略怎么样；第二，提高当前策略。主要方法有两种：一种叫做**蒙特卡罗采样方法（Monte-Carlo method）**，一种是**时序差分法（Temporal Difference method）**。

蒙特卡罗采样方法的本质是利用采样来计算 Q 的期望值，比如简单的，我们可以直接采样环境中的数据，来估计 Q 的值。但是这种简单的方法存在一个问题，就是由于缺乏对环境的探索，没有足够多种类的数据，无法评估策略在所有情况下的可能性，导致 Q 值的更新可能会存在问题。

所以我们需要对环境进行探索，得到不同动作的 Q 值。我们可以探索尽可能多的次数，来确保 Q 值估算的准确性，但花费了太多机会在 Q 值低的动作上；也可以不同动作都只探索一次，然后根据这一次的 Q 值估算结果，将剩余的机会都给予这次估计中 Q 值大的动作，这就是一个寻找平衡点的问题了。

有一种理论上相当漂亮的 **UCB（Upper Confidence Bound）** 方法被用来处理这个问题，简单的讲，两个动作的平均 Q 值差距越大，探索次数就越少。

(/apps/download?
utm_source=sbc)



而且对于策略而言，存在 **On/Off Policy 策略** 的区别，简单的说 On Policy 就是策略中包含探索，而 Off Policy 的话，探索是探索，策略是策略，通过带探索的策略采样，但更新的只是策略本身。

另外，蒙特卡洛采样算法有一个很突出的缺陷：一定要拿到整个轨迹以后，才能更新模型。而时序差分法每走一步都可以更新模型。

在时序差分法中，On Policy 策略对应** SARSA 算法**，Off Policy 策略对应 **Q-Learning 算法**。

前面所讲的这两种方法，处理的都是离散的情景，但真实环境中有很多问题无法离散的来表示，为了能让强化学习方法能够处理连续场景下的问题，又有两种方法被提出了，分别是：**值函数估计（Value function approximation）、策略搜索（Policy Search）**。

策略搜索可以学出随机性策略，不同于值函数的确定性策略；并且策略搜索和监督学习的兼容性比较好；在高维数据下的表现也好一些。

阿里小蜜技术实践分析

上文已经对自然语言问答所涉及到的技术进行了概要的讲解，这里准备以阿里小蜜为例，具体的探讨不同问题场景下的技术选型，以《颠覆传统的电商智能助理-阿里小蜜技术揭秘》(<https://link.jianshu.com?t=http://www.infoq.com/cn/articles/electricity-supplier-intelligent-assistant/>)为参考。

首先，为了匹配问题场景，需要进行**意图识别**：

(/apps/download?utm_source=sbc) ×

(/apps/download?utm_source=sbc) ×

将用户相关特征、用户行为序列、Query+Context 等，表示为前文讲过的 one-hot representation 或者 Distributed Representation 的词向量，作为输入送入 DNN 模型，可以直接进行多分类，也可以进行多次二分类。多分类的性能要优于二分类，而二分类的复用性较好。

之后进入到具体问题场景的处理阶段，前文提到，经典的问题类型有三种：**问答型、任务型、闲聊型**。

而目前主流的智能匹配技术分为如下 4 种方法：

- **基于模板匹配 (Rule-Based)**
- **基于检索模型 (Retrieval Model)**
- **基于统计机器翻译模型 (SMT)**
- **基于深度学习模型 (Deep Learning)**

模板匹配就是基于规则的自然语言处理了，通常是基于上下文无关文法搭建一个解析系统，由人工编写或机器生成一些匹配句式，配合同义词典来处理用户问句。

检索模型上文提到过，BM25 就是一个典型的例子。

SMT 上文也提到过，通过将翻译模型引入搜索，来建模词与词之间的语义相近关系。

基于深度学习模型的上文也提到过很多，比如神经概率语言模型、word2vec、DSSM、基于 CNN 的、基于 RNN 的。

对于**问答型**问题，其特点是：有领域知识的概念，且知识之间的关联性高，并且对精准度要求比较高。



基于问答型场景的特点，阿里小蜜在技术选型上采用了**知识图谱构建+检索模型**相结合的方式进行核心匹配模型的设计。所谓『知识图谱构建+检索模型』，与前文百度的『知识图谱+搜索』是异曲同工的。

知识图谱的构建方法在《浅谈知识图谱基础》

(<https://www.jianshu.com/p/4f09043e22ea>)中有详细的讲解。

知识图谱在模型构建初期可能会存在数据的松散和覆盖率问题，导致匹配的覆盖率不足，也就是前文提到的『盲区』问题；原文中还提到了知识图谱的维护成本较高；原文中没有提到，但必定存在的是，仅凭知识图谱无法分辨用户具体意图，也就是前文提到的『用户意图』问题。所以除了知识图谱，还要配合传统的检索模型，阿里小蜜检索模型的基本流程如下图：

其实主要也就是搜索引擎里面的那些东西，可以参考《浅谈搜索引擎基础（上）》

(<https://www.jianshu.com/p/c51416cf2849>)和《浅谈搜索引擎基础（下）》

(<https://www.jianshu.com/p/8ac91c39ae8f>)两篇文章。

计算模块阿里小蜜用的是空间向量模型，利用余弦相似度等计算文本之间的相似度。不过改成概率检索模型 BM25，再配合前文提到的归一化命中、同义词命中、主题命中和反义主题命中，可能会得到更好的结果。

(/apps/download?
utm_source=sbc)



然后是**任务型**问题，特点是有领域的概念，每个任务负责独立的业务流程，任务之间相对互斥性强，精准度要求高。

基于任务型的特点，在技术选型上，阿里小蜜采用了**意图决策+Slot Filling**的方式进行会话匹配设计。

所谓意图决策，就是找到对应的领域本体知识场景，比如机票的场景：

(/apps/download?
utm_source=sbc)



而意图决策其实就是分类，和词性标注、词义消歧、实体消歧、意图识别这些方法类似，比如可以用上下文特征向量和余弦相似度来聚类（无监督）、用朴素贝叶斯算法（有监督生成模型）、用最大熵模型（有监督判别模型）、或者把上下文 Embedding 了送进一个 DNN 里也是可以的（深度学习）。

找到对应的领域本体知识场景之后，就要开始填槽了，做 Slot Filling 的方法前文也有涉及，百度用的是双向 LSTM+CRF。

阿里的基本流程是，在问答匹配过程中结合上下文模型和领域数据模型不断在 Query 中进行 Slot Filling，并循环进行本体意图树的不断填充和修改，直到**必选意图树填充完整**后进行输出，如下图：



(/apps/download?
utm_source=sbc) ×

核心就是把所有必填的槽位都填好，没填好就追问，这也是封闭域中的一种可行的多轮对话方式。

但是其实任务型问题远不止这么简单，就比如上文中提到的帮助用户填报高考志愿这样的任务型问题，不是通过追问填好槽位 A B C 就能输出结果的。这种任务型问题涉及到状态的转移，用户在询问的过程中不断了解，不断澄清，系统要不断选择合适的 Action 来引导用户逐步明确自己的意向，最终填好最重要的那个槽。比如『我该怎么填写高考志愿』，难道直接问用户『你想去哪所学校』、『你想报哪个专业』来填槽？这样的问题通过 Reinforcement Learning 来处理可能会是不错的方法。

然后就是**闲聊型**了，特点是非面向目标，语义意图不明确，通常期待的是语义相关性和渐进性，对准确率要求相对较低。

不过我个人有个观点，就是『不能通过图灵测试的chatbot，其价值不如个人助理』。

阿里小蜜的闲聊型问题的解决方案是**检索模型与 Deep Learning 相结合**。

检索模型就不说了，所谓 Deep Learning 的方法，就是指前文中的 seq2seq、Encoder-Decoder、Attention Model那些了。

阿里小蜜的聊天引擎，结合了两者的优势，将两个模型进行了融合。先通过传统的检索模型检索出候选集数据，然后通过 seq2seq 对候选集进行 Rerank，重排序后超过指定的阈值就进行输出，不到阈值就通过 seq2seq 进行答案生成，整体流程如下图：

(/apps/download?
utm_source=sbc) ×

后记

花了半年，看了几百万字的材料，写了将近 20 万字的文章，从机器学习、深度学习、自然语言处理、语音识别、推荐系统、搜索引擎、数据挖掘、知识图谱的经典模型和基础知识，到综合前面的所有内容来构建自然语言问答这块的技术基础，到今天，终于算是圆满完成给自己定下的毕业前的任务了。也希望我能在 3-5 年后，成为能够独当一面的人工智能产品经理：)

📖 Machine Learning Phoenix (/nb/9152053)

举报文章 © 著作权归作者所有



我偏笑_NSNirvana (/u/2293f85dc197) ★ ♂

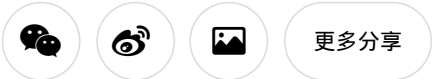
写了 253995 字，被 1335 人关注，获得了 686 个喜欢
(/u/2293f85dc197)

+ 关注

INTJ，人机对话产品经理，微博@我偏笑_NSNirvana

♡ 喜欢 (/sign_in?utm_source=desktop&utm_medium=not-signed-in-like-button) | 25





(/apps/download?utm_source=nbc)

nshu.io/notes/images/12178251/weibo/image_1
(/apps/download?
utm_source=sbc)

Deep Learning in NLP-词向量和语言模型 (/p/d46609947489?utm_campa...

Deep Learning 算法已经在图像和音频领域取得了惊人的成果，但是在 NLP 领域中尚未见到如此激动人心的结果。关于这个原因，引一条我比较赞同的微博。@王威廉：Steve Renals算了一下icassp录取文章题目中...

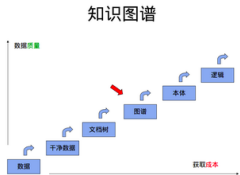
stonelin3935 (/u/6c767c086b0d?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

【刘知远】知识图谱——机器大脑中的知识库 (/p/8bf571284747?utm_cam...

作者：刘知远（清华大学）；整理：林颖（RPI）本文来自Big Data Intelligence知识就是力量。——[英]弗兰西斯·培根1 什么是知识图谱在互联网时代，搜索引擎是人们在线获取信息和知识的重要工具。当用户输入一..

墨白找 (/u/9ee413023cf1?
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/bd15e0f50eb9?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)
知识图谱技术解剖 (/p/bd15e0f50eb9?utm_campaign=maleskine&utm_c...

本体、知识库、知识图谱、知识图谱识别之间的关系？本体：领域术语集合。知识库：知识集合。知识图谱：图状具有关联性的知识集合。知识图谱本质上是语义网络，是一种基于图的数据结构，由节点(Point)...

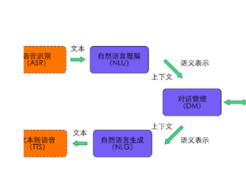




方弟 (/u/1ffaf4faed6?)

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/94d83218a7aa?)

(/apps/download?
utm_source=sbc)

utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

阿里智能对话交互技术实践与创新 (/p/94d83218a7aa?utm_campaign=mal...

姓名：周小蓬 16019110037 转载自：http://blog.csdn.net/qq_40027052/article/details/78672907 [嵌牛导读]

过去 20 多年，互联网及移动互联网将人类带到了一个全新的时代，如果用一个词来总结和概括这个时代的...



aeytifw (/u/bb736bc8eea9?)

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

机器学习(Machine Learning)&深度学习(Deep Learning)资料(Chapter 1) (...)

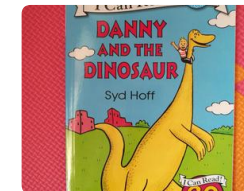
机器学习(Machine Learning)&深度学习(Deep Learning)资料(Chapter 1) 注:机器学习资料篇目一共500条,篇目二开始更新 希望转载的朋友,你可以不用联系我,但是一定要保留原文链接,因为这个项目还在继续也...



Albert陈凯 (/u/185a3c553fc6?)

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/248d3966dda5?)



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

清华百人201704赵子彧6月28日打卡 (/p/248d3966dda5?utm_campaign=m...

晨读：孩子《凉州词》妈妈《the ugle duckling》路上清华音频1a 学校组织参观湖广会馆，下午放假 午睡

后作业 读绘本《dannt and dunosaur》i want go and play outside 线上英语 reading eggs 游戏 b...



甜蜜盒子Lily (/u/b75976729f4d?)

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)



(/p/dbb0e4f01859?



(/apps/download?
utm_source=sbc)



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

7月26日晨起感恩 (/p/dbb0e4f01859?utm_campaign=maleskine&utm_co...

感恩昨天死党的聚会，满满都是爱和支持，感恩望望的精心准备 感恩读书会事情，死党们都愿意积极参与，给我很大的信心 感恩林娴姐一早来电的支持和鼓励，简单来说逐步丰富 感恩近期身边亲朋好友的不理解，...



棋予 (/u/1d6ddcb29fef?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

域名被“墙”怎么办？ (/p/53da604d7351?utm_campaign=maleskine&utm...

一、域名被墙：如果域名ping的通却打不开网站（排除服务器宕机），用代理或者使用VPN可以打开一般说明域名被封了。假如域名下的网站非法信息多，敏感，又不整改，会直接被墙掉，就是通常所说的被封锁...



Ashley_8aff (/u/719453b66194?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/1049abca56f9?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

没有雾霾的清晨 (/p/1049abca56f9?utm_campaign=maleskine&utm_cont...

清晨的闹铃还没响，一阵阵清脆婉转的鸟叫声把她从睡梦中吵醒，早起的鸟儿有虫吃，勤快的小麻雀已经迫不及待的出来觅食叫个不停。窗外的光斜斜地透了进来，撒了一屋子的金色，她的情绪被感染，睡意全无。...




杨扬雪 (/u/844ce0e50bb5?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

我反现了系列 (/p/13c4f3fc7a8d?utm_campaign=maleskine&utm_conten...



1.我发现: 微博和微信朋友圈, 这种发布碎片信息的地方不适合好文笔的存在。 理由很简单嘛:你和自己对话, 或是和别人聊天的时候, 会用大白话还是书面语? 而且自带社交属性的微信尤甚。 2.我发现: 原来24楼..

 维姬的自言自语 (/u/36e274fb959d?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation) (/apps/download?utm_source=sbc) ✕

