[深度学习] 自然语言处理 --- NLP入门指南_小墨鱼的专栏-程序员宅基地

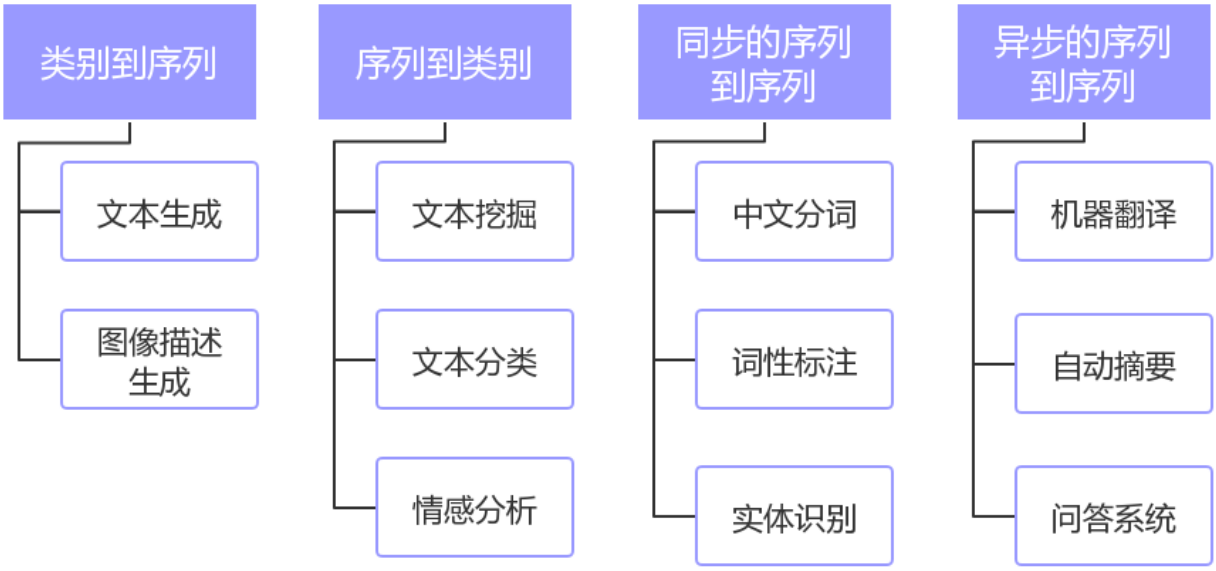
技术标签： NLP (/searchArticle?qc=NLP&page=1) 深度学习 (/searchArticle?qc=深度学习&page=1)

NLP的全称是**N**atuarl **L**anguage **P**rocessing，中文意思是自然语言处理，是人工智能领域的一个重要方向

自然语言处理（NLP）的一个最伟大的方面是跨越多个领域的计算研究，从人工智能到计算语言学的多个计算研究领域都在研究计算机与人类语言之间的相互作用。它主要关注计算机如何准确并快速地处理大量的自然语言语料库。什么是自然语言语料库？它是用现实世界语言表达的语言学习，是从文本和语言与另一种语言的关系中理解一组抽象规则的综合方法。

人类语言是抽象的信息符号，其中蕴含着丰富的语义信息，人类可以很轻松地理解其中的含义。而计算机只能处理数值化的信息，无法直接理解人类语言，所以需要将人类语言进行数值化转换。不仅如此，人类间的沟通交流是有上下文信息的，这对于计算机也是巨大的挑战。

我们首先来看看NLP的任务类型，如下图所示：

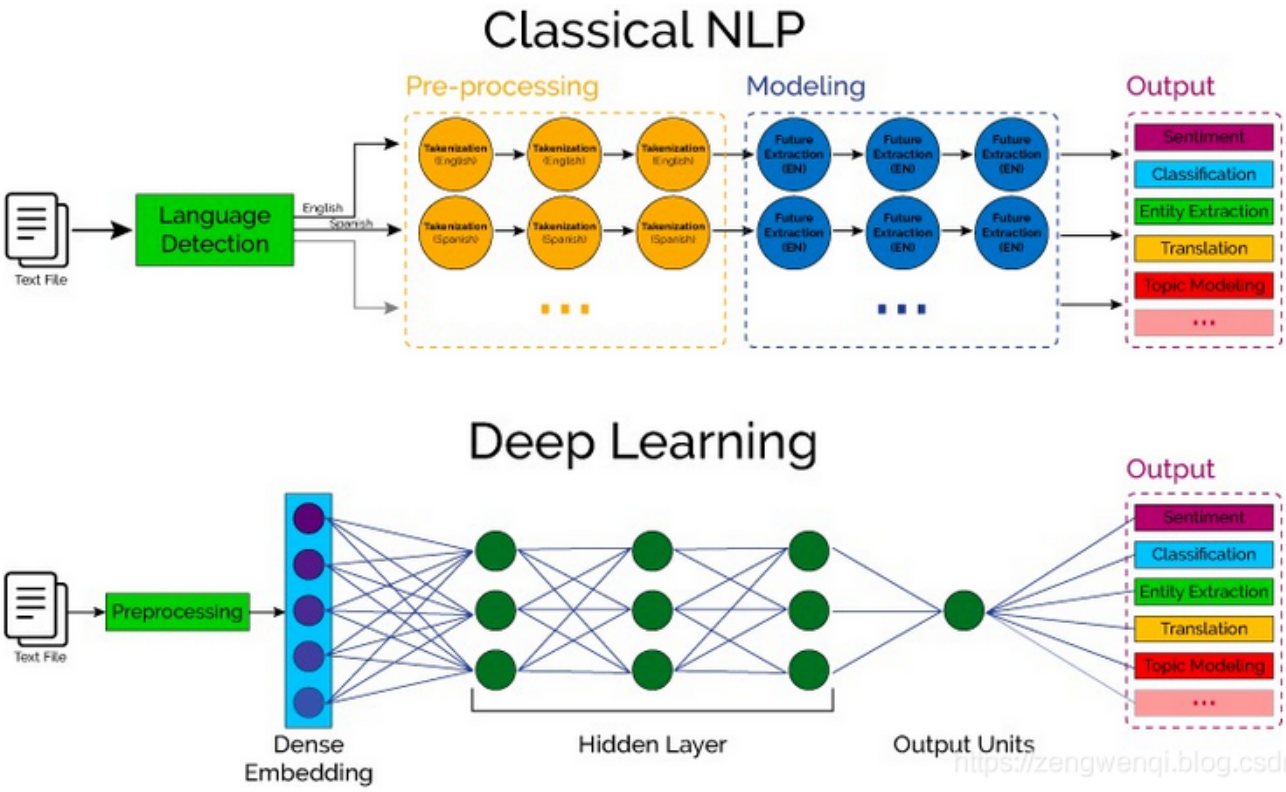


主要划分为了四大类：

- 类别到序列
- 序列到类别
- 同步的序列到序列
- 异步的序列到序列

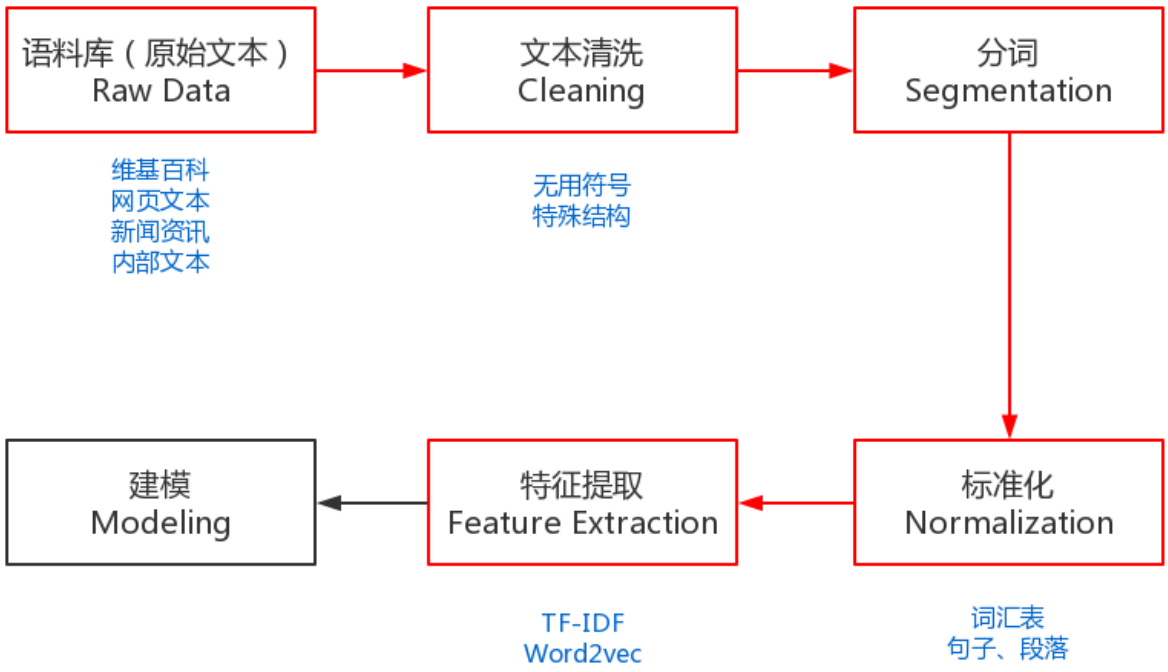
其中“类别”可以理解为是标签或者分类，而“序列”可以理解为是一段文本或者一个数组。简单概况NLP的任务就是从一种数据类型转换成另一种数据类型的过程，这与绝大多数的机器学习模型相同或者类似，所以掌握了NLP的技术栈就等于掌握了机器学习的技术栈。

传统方式和深度学习方式 NLP 对比



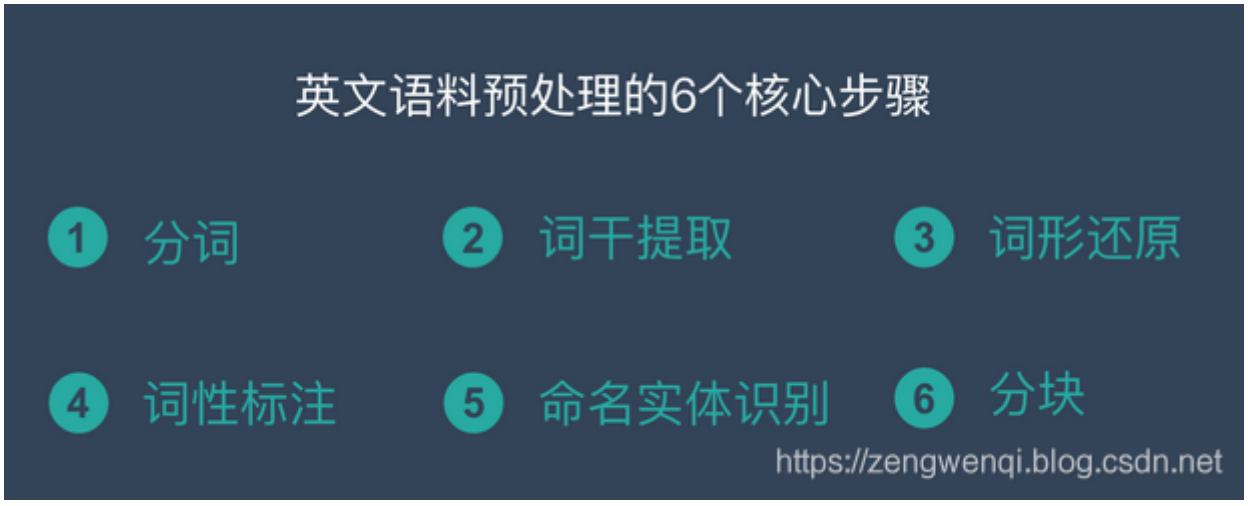
NLP的预处理

为了能够完成上述的NLP任务，我们需要一些预处理，是NLP任务的基本流程。预处理包括：收集语料库、文本清洗、分词、去掉停用词（可选）、标准化和特征提取等。



图中红色的部分就是NLP任务的预处理流程，有别于其它机器学习任务的流程

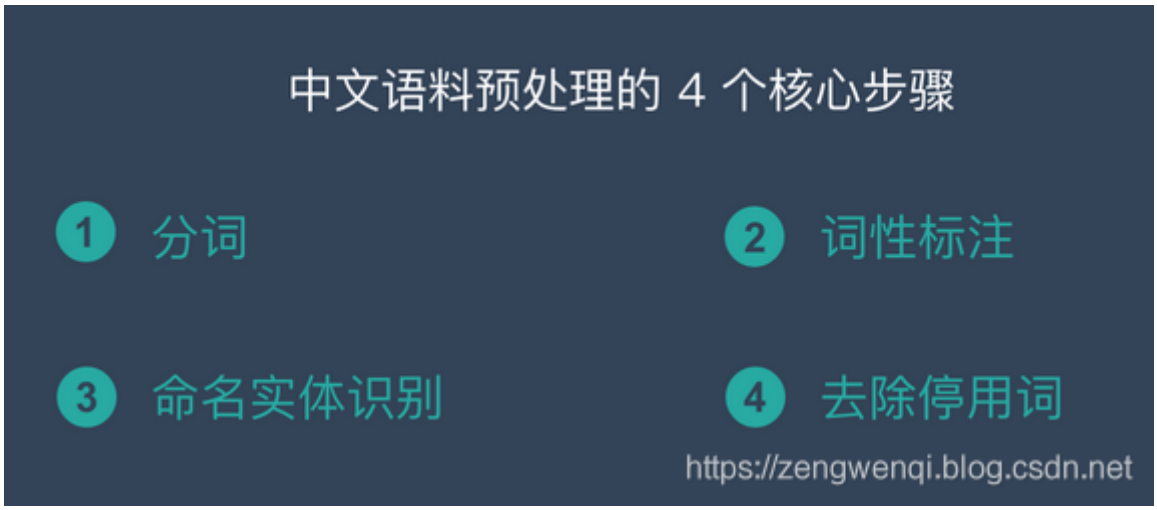
英文 NLP 语料预处理的 6 个步骤



1. 分词 – Tokenization (https://easyai.tech/ai-definition/tokenization/)

2. 词干提取 (https://easyai.tech/ai-definition/stemming-lemmatisation/) – Stemming (https://easyai.tech/ai-definition/stemming-lemmatisation/)
3. 词形还原 (https://easyai.tech/ai-definition/stemming-lemmatisation/) – Lemmatization
4. 词性标注 – Parts of Speech (https://easyai.tech/ai-definition/part-of-speech/)
5. 命名实体识别 – NER (https://easyai.tech/ai-definition/ner/)
6. 分块 – Chunking

中文 NLP 语料预处理的 4 个步骤



1. 中文分词 – Chinese Word Segmentation (https://easyai.tech/ai-definition/tokenization/)
2. 词性标注 – Parts of Speech (https://easyai.tech/ai-definition/part-of-speech/)
3. 命名实体识别 – NER (https://easyai.tech/ai-definition/ner/)
4. 去除停用词

第1步：收集您的数据---语料库

对于NLP任务来说，没有大量高质量的语料，就是巧妇难为无米之炊，是无法工作的。

而获取语料的途径有很多种，最常见的方式就是直接下载开源的语料库，如：维基百科的语料库 (https://dumps.wikimedia.org/zhwiki/)。

但这样开源的语料库一般都无法满足业务的个性化需要，所以就需要自己动手开发爬虫去抓取特定的内容，这也是一种获取语料库的途径。当然，每家互联网公司根据自身的业务，也都会有大量的语料数据，如：用户评论、电子书、商品描述等等，都是很好的语料库。

示例数据源

每个机器学习问题都从数据开始，例如电子邮件，帖子或推文列表。常见的文字信息来源包括：

- 产品评论（在亚马逊，Yelp和各种应用商店）
- 用户生成的内容（推文，Facebook帖子，StackOverflow问题）
- 故障排除（客户请求，支持服务单，聊天记录）

现在，数据对于互联网公司来说就是石油，其中蕴含着巨大的商业价值。所以，小伙伴们在日常工作中一定要养成收集数据的习惯，遇到好的语料库一定要记得备份（当然是在合理合法的条件下），它将会对你解决问题提供巨大的帮助。

第2步：清理数据 --- 文本清洗

我们遵循的首要规则是："您的模型将永远与您的数据一样好。"

数据科学家的关键技能之一是了解下一步是应该对模型还是数据进行处理。一个好的经验法则是首先查看数据然后进行清理。**一个干净的数据集将允许模型学习有意义的功能，而不是过度匹配无关的噪音。**

我们通过不同的途径获取到了想要的语料库之后，接下来就需要对其进行清洗。因为很多的语料数据是无法直接使用的，其中包含了大量的无用符号、特殊的文本结构。

> 数据类型分为：
>
> - 结构化数据：关系型数据、json等
> - 半结构化数据：XML、HTML等
> - 非结构化数据：Word、PDF、文本、日志等

需要将原始的语料数据转化成易于处理的格式，一般在处理HTML、XML时，会使用Python的lxml (https://lxml.de/)库，功能非常丰富且易于使用。对一些日志或者纯文本的数据，我们可以使用正则表达式进行处理。

正则表达式是使用单个字符串来描述、匹配一系列符合某个句法规则的字符串。Python的示例代码如下：

```
import re

# 定义中文字符的正则表达式
re_han_default = re.compile("([\u4E00-\u9FD5]+)", re.U)

sentence = "我/爱/自/然/语/言/处/理"

# 根据正则表达式进行切分
blocks= re_han_default.split(sentence)

for blk in blocks:
    # 校验单个字符是否符合正则表达式
    if blk and re_han_default.match(blk):
        print(blk)
```

输出：

```
我
爱
自
然
语
言
处
理
```

除了上述的内容之外，我们还需要注意中文的编码问题，在windows平台下中文的默认编码是GBK（gb2312），而在linux平台下中文的默认编码是UTF-8。在执行NLP任务之前，我们需要统一不同来源语料的编码，避免各种莫名其妙的问题。

如果大家事前无法判断语料的编码，那么我推荐大家可以使用Python的chardet (https://github.com/chardet/chardet)库来检测编码，简单易用。既支持命令行： `chardetect somefile` ，也支持代码开发。

以下是用于清理数据的清单:

1. 删除所有不相关的字符，例如任何非字母数字字符
2. 令牌化通过将其分割成单个的单词文本
3. 删除不相关的单词，例如“@”twitter提及或网址
4. 将所有字符转换为小写，以便将诸如“hello”，“Hello”和“HELLO”之类的单词视为相同
5. 考虑将拼写错误或交替拼写的单词组合成单个表示（例如“cool”/“kewl”/“cooool”）
6. 考虑词开还原（将诸如“am”，“are”和“is”之类的词语简化为诸如“be”之类的常见形式）

按照这些步骤并检查其他错误后，我们可以开始使用干净的标记数据来训练模型！

第3步：分词

中英文分词的3个典型区别

区别1：分词方式不同，中文更难

英文有天然的空格作为分隔符，但是中文没有。所以如何切分是一个难点，再加上中文里一词多意的情况非常多，导致很容易出现歧义。下文中难点部分会详细说明。

区别2：英文单词有多种形态

英文单词存在丰富的变形变换。为了应对这些复杂的变换，英文NLP相比中文存在一些独特的处理步骤，我们称为词形还原（Lemmatization）和词干提取(Stemming)。中文则不需要

词性还原：does，done，doing，did 需要通过词性还原恢复成 do。

词干提取：cities，children，teeth 这些词，需要转换为 city，child，tooth”这些基本形态

区别3：中文分词需要考虑粒度问题

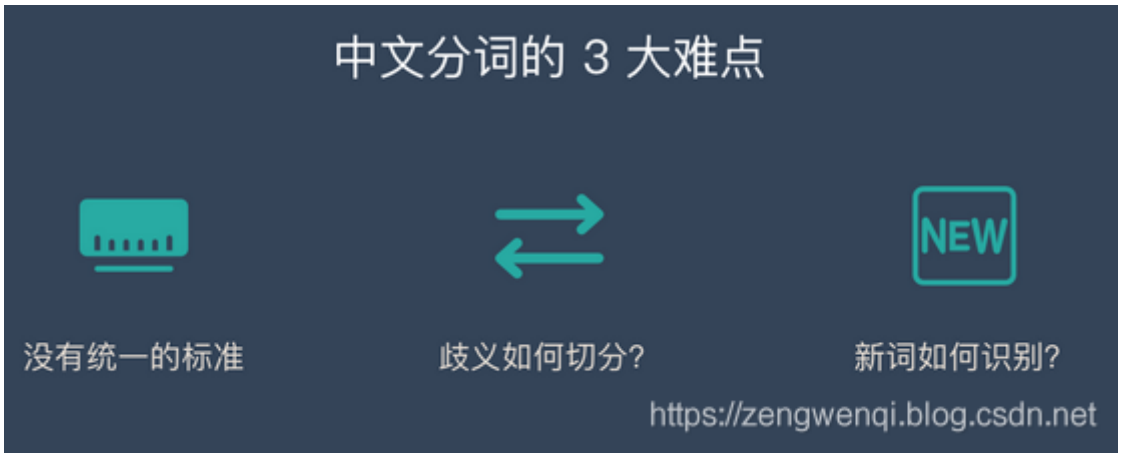
例如「中国科学技术大学」就有很多种分法：

- 中国科学技术大学
- 中国 \ 科学技术 \ 大学
- 中国 \ 科学 \ 技术 \ 大学

粒度越大，表达的意思就越准确，但是也会导致召回比较少。所以中文需要不同的场景和要求选择不同的粒度。这个在英文中是没有的。

中文分词是一个比较大的课题，相关的知识点和技术栈非常丰富，可以说搞懂了中文分词就等于搞懂了大半个NLP。

中文分词的3大难点



难点 1：没有统一的标准

目前中文分词没有统一的标准，也没有公认的规范。不同的公司和组织各有各的方法和规则。

难点 2：歧义词如何切分

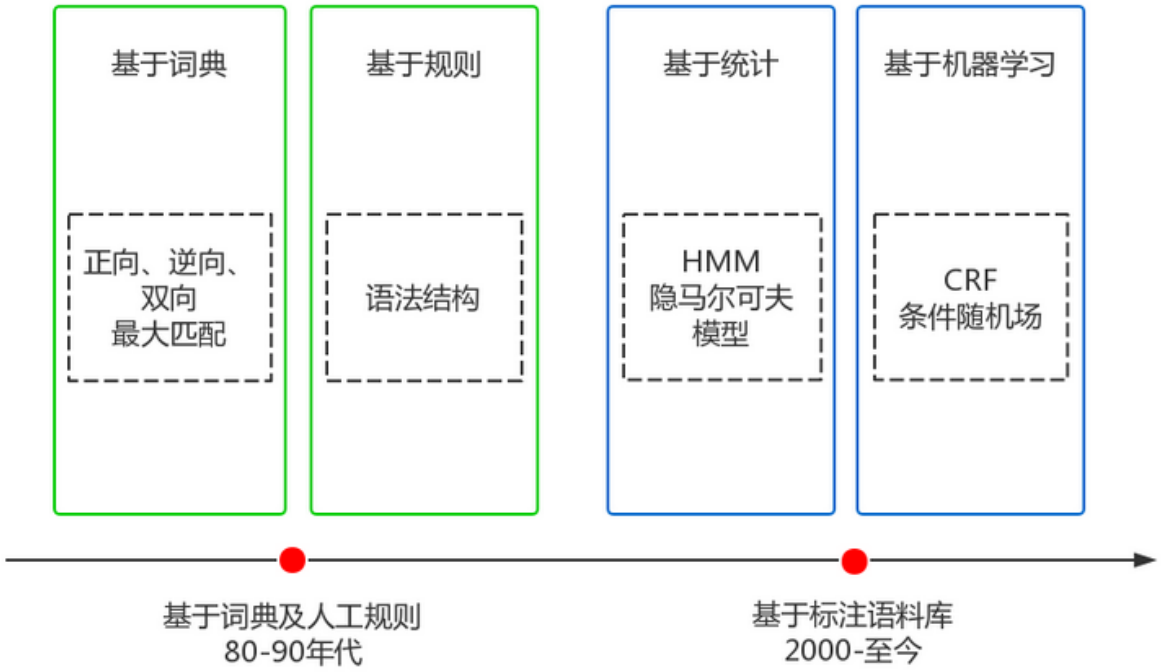
例如「兵乓球拍卖完了」就有2种分词方式表达了2种不同的含义：

- 乒乓球 \ 拍卖 \ 完了
- 乒乓 \ 球拍 \ 卖 \ 完了

难点 3：新词的识别

信息爆炸的时代，三天两头就会冒出来一堆新词，如何快速的识别出这些新词是一大难点。比如当年「蓝瘦香菇」大火，就需要快速识别。

中文分词经历了20多年的发展，克服了重重困难，取得了巨大的进步，大体可以划分成两个阶段，如下图所示：



词典匹配与规则

优点：速度快、成本低

缺点：适应性不强，不同领域效果差异大

基本思想是基于词典匹配，将待分词的中文文本根据一定规则切分和调整，然后跟词典中的词语进行匹配，匹配成功则按照词典的词分词，匹配失败通过调整或者重新选择，如此反复循环即可。代表方法有基于正向最大匹配和基于逆向最大匹配及双向匹配法。

基于统计与机器学习

优点：适应性较强

缺点：成本较高，速度较慢

这类目前常用的是算法是**HMM**、**CRF**等算法，比如stanford、Hanlp分词工具是基于CRF算法。以CRF为例，基本思路是对汉字进行标注训练，不仅考虑了词语出现的频率，还考虑上下文，具备较好的学习能力，因此其对歧义词和未登录词的识别都具有良好的效果。

常见的分词器都是使用机器学习算法和词典相结合，一方面能够提高分词准确率，另一方面能够改善领域适应性。

目前，主流的中文分词技术采用的都是基于词典最大概率路径+未登录词识别（HMM）的方案，其中典型的代表就是 `jieba` 分词，一个热门的多语言中文分词包。

中文分词工具

下面排名根据 GitHub 上的 star 数排名：

1. Hanlp (https://github.com/hankcs/HanLP)
2. Stanford 分词 (https://github.com/stanfordnlp/CoreNLP)
3. ansj 分词器 (https://github.com/NLPchina/ansj_seg)
4. 哈工大 LTP (https://github.com/HIT-SCIR/ltp)
5. KCWS分词器 (https://github.com/koth/kcws)
6. jieba (https://github.com/yanyiwu/cppjieba)
7. IK (https://github.com/wks/ik-analyzer)
8. 清华大学THULAC (https://github.com/thunlp/THULAC)
9. ICTCLAS (https://github.com/thunlp/THULAC)

英文分词工具

1. Keras (https://github.com/keras-team/keras)
2. Spacy (https://github.com/explosion/spaCy)
3. Gensim (https://github.com/RaRe-Technologies/gensim)
4. NLTK (https://github.com/nltk/nltk)

第4步：标准化

标准化是为了给后续的处理提供一些必要的基础数据，包括：去掉停用词、词汇表、训练数据等等。

当我们完成了分词之后，可以去掉停用词，如：“其中”、“况且”、“什么”等等，但这一步不是必须的，要根据实际业务进行选择，像关键词挖掘就需要去掉停用词，而像训练词向量就不需要。

词汇表是为语料库建立一个所有不重复词的列表，每个词对应一个索引值，并索引值不可以改变。词汇表的最大作用就是可以将词转化成一个向量，即One-Hot编码。

假设我们有这样一个词汇表：

```
我
爱
自然
语言
处理
```

那么，我们就可以得到如下的One-Hot编码：

```
我：   [1, 0, 0, 0, 0]
爱：   [0, 1, 0, 0, 0]
自然：[0, 0, 1, 0, 0]
语言：[0, 0, 0, 1, 0]
处理：[0, 0, 0, 0, 1]
```

这样我们就可以简单的将词转化成了计算机可以直接处理的数值化数据了。虽然One-Hot编码可以较好的完成部分NLP任务，但它的问题还是不少的。

当词汇表的维度特别大的时候，就会导致经过One-Hot编码后的词向量非常稀疏，同时One-Hot编码也缺少词的语义信息。由于这些问题，才有了后面大名鼎鼎的**Word2vec**，以及Word2vec的升级版**BERT**。

除了词汇表之外，我们在训练模型时，还需要提供训练数据。模型的学习可以大体分为两类：

- 监督学习，在已知答案的标注数据集上，模型给出的预测结果尽可能接近真实答案，适合预测任务
- 非监督学习，学习没有标注的数据，是要揭示关于数据隐藏结构的一些规律，适合描述任务

根据不同的学习任务，我们需要提供不同的标准化数据。一般情况下，标注数据的获取成本非常昂贵，非监督学习虽然不需要花费这样的成本，但在实际问题的解决上，主流的方式还选择监督学习，因为效果更好。

带标注的训练数据大概如下所示（情感分析的训练数据）：

```
距离 川沙 公路 较近 公交 指示 蔡陆线 麻烦 建议 路线 房间 较为简单         __label__1
商务 大床 房 房间 很大 床有 2M 宽 整体 感觉 经济 实惠 不错 !    __label__1
半夜 没 暖气 住 !          __label__0
```

其中每一行就是一条训练样本， `__label__0` 和 `__label__1` 是分类信息，其余的部分就是分词后的文本数据。

第5步：特征提取

为了能够更好的训练模型，我们需要将文本的原始特征转化成具体特征，转化的方式主要有两种：**统计**和**Embedding**。

> 原始特征：需要人类或者机器进行转化，如：文本、图像。
>
> 具体特征：已经被人类进行整理和分析，可以直接使用，如：物体的重要、大小。

NLP表示方式

目前常用的文本表示方式分为：

1. 离散式表示（*Discrete Representation*）；
2. 分布式表示（*Distributed Representation*）；

离散式表示（*Discrete Representation*）

One-Hot

One-Hot 编码又称为“独热编码”或“哑编码”，是最传统、最基础的词（或字）特征表示方法。这种编码将词（或字）表示成一个向量，该向量的维度是词典（或字典）的长度（该词典是通过语料库生成的），该向量中，当前词的位置的值为1，其余的位置为0。

文本使用*one-hot* 编码步骤：

1. 根据语料库创建 词典（*vocabulary*），并创建词和索引的 映射（*stoi，itos*）；
2. 将句子转换为用索引表示；
3. 创建*OneHot* 编码器；
4. 使用*OneHot* 编码器对句子进行编码；

***One-Hot* 编码的特点如下：**

1. 词向量长度是词典长度；
2. 在向量中，该单词的索引位置的值为 1 ，其余的值都是 0
3. 使用*One-Hot* 进行编码的文本，得到的矩阵是**稀疏矩阵**

缺点：

1. 不同词的向量表示互相正交，无法衡量不同词之间的关系；
2. 该编码只能反映某个词是否在句中出现，无法衡量不同词的重要程度；
3. 使用*One-Hot* 对文本进行编码后得到的是高维稀疏矩阵，会浪费计算和存储资源；

词袋模型（Bag Of Word，BOW）

例句：

1. *Jane wants to go to Shenzhen.*
2. *Bob wants to go to Shanghai.*

在词袋模型中不考虑语序和词法的信息，每个单词都是相互独立的，将词语放入一个“袋子”里，统计每个单词出现的频率。

词袋模型编码特点：

1. 词袋模型是对文本（而不是字或词）进行编码；
2. 编码后的向量长度是词典的长度；
3. 该编码忽略词出现的次序；
4. 在向量中，该单词的索引位置的值为单词在文本中出现的次数；如果索引位置的单词没有在文本中出现，则该值为 0；

缺点

1. 该编码忽略词的位置信息，位置信息在文本中是一个很重要信息，词的位置不一样语义会有很大的差别（如 “猫爱吃老鼠” 和 “老鼠爱吃猫” 的编码一样）；
2. 该编码方式虽然统计了词在文本中出现的次数，但仅仅通过“出现次数”这个属性无法区分常用词（如：“我”、“是”、“的”等）和关键词（如：“自然语言处理”、“*NLP*”等）在文本中的重要程度；

TF-IDF（词频-逆文档频率）

为了解决词袋模型无法区分常用词（如：“是”、“的”等）和专有名词（如：“自然语言处理”、“*NLP*”等）对文本的重要性的问题，*TF-IDF* 算法应运而生。

TF-IDF 全称是：*term frequency–inverse document frequency* 又称 词频-逆文本频率。其中：

统计的方式主要是计算词的词频（TF）和逆向文件频率（IDF）：

1. ***TF***（***Term Frequency***）：某个词在当前文本中出现的频率，频率高的词语或者是重要的词（如：“自然语言处理”）或者是常用词（如：“我”、“是”、“的”等）；
2. ***IDF***（***Inverse Document frequency***）：逆文本频率。文本频率是指：含有某个词的文本在整个语料库中所占的比例。逆文本频率是文本频率的倒数；

那么，每个词都会得到一个TF-IDF值，用来衡量它的重要程度，计算公式如下：

$$TF = \frac{某个词在文章中出现的总次数}{文章包含的总词数}$$

$$IDF = log(\frac{语料库的文本总数}{包含某词的文本数量+1})$$

$$TF-IDF = TF * IDF$$

优点

1. 实现简单，算法容易理解且解释性较强；
2. 从*IDF*的计算方法可以看出常用词（如：“我”、“是”、“的”等）在语料库中的很多文章都会出现，故IDF的值会很小；而关键词（如：“自然语言处理”、“*NLP*”等）只会在某领域的文章出现，*IDF*的值会比较大；故：***TF-IDF*在保留文章的重要词的同时可以过滤掉一些常见的、无关紧要的词；**

缺点

1. 不能反映词的位置信息，在对关键词进行提取时，词的位置信息（如：标题、句首、句尾的词应该赋予更高的权重）；
2. *IDF*是一种试图抑制噪声的加权，本身倾向于文本中频率比较小的词，这使得*IDF*的精度不高；
3. *TF-IDF*严重依赖于语料库（尤其在训练同类语料库时，往往会掩盖一些同类型的关键词；如：在进行*TF-IDF*训练时，语料库中的 娱乐 新闻较多，则与 娱乐 相关的关键词的权重就会偏低 ），因此需要选取质量高的语料库进行训练；

分布式表示（*Distributed Representation*

理论基础：

- 1954年，*Harris*提出分布式假说（*distributional hypothesis*）奠定了这种方法的理论基础：***A word's meaning is given by the words that frequently appear close-by***（上下文相似的词，其语义也相似）；
- 1957年，*Firth*对分布式假说做出进一步的阐述和明确：***A word is characterized by the company it keeps***（词的语义由其上下文决定）；

n-gram

n-gram 是一种 语言模型(Language Model, LM)。语言模型是一种基于概率的判别式模型，该模型的输入是一句话（单词的序列），输出的是这句话的概率，也就是这些单词的联合概率（*joint probability*）。（备注：语言模型就是判断一句话是不是正常人说的。）

共现矩阵（*Co-Occurrence Matrix*）

首先指定窗口大小，然后统计窗口（和对称窗口）内词语共同出现的次数作为词的向量（*vector (https://easyai.tech/ai-definition/vector/)*）。

语料库：

1. *I like deep learning.*
2. *I like NLP.*
3. *I enjoy flying.*

备注： 指定窗口大小为1（即：左右的 window_length=1，相当于 bi-gram）统计数据如下：*（I, like）， （like, deep）， （deep, learning）， （learning, .）， （I, like）， （like, NLP）， （NLP, .）， （I, enjoy）， （enjoy, flying）， （flying, .）*。则语料库的共现矩阵如下表所示：

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learing	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

从以上的共现矩阵可以看出，单词 like 和 enjoy 都在单词 I 附件出现且统计数目大概相等，则它们在 语义 和 语法 上的含义大概相同。

优点

1. 考虑了句子中词的顺序；

缺点

1. 词表的长度很大，导致词的向量长度也很大；
2. 共现矩阵也是稀疏矩阵（可以使用 *SVD*、*PCA* 等算法进行降维，但是计算量很大）；

Word2Vec

word2vec 模型是*Google*团队在2013年发布的 word representation 方法。该方法一出让 预训练词向量 的使用在 *NLP* 领域遍地开花。

word2vec模型

word2vec有两种模型：***CBOW*** 和 ***SKIP-GRAM***；

- CBOW：利用上下文的词预测中心词；

- SKIP-GRAM：利用中心词预测上下文的词；

优点

1. 考虑到词语的上下文，学习到了语义和语法的信息；
2. 得到的词向量维度小，节省存储和计算资源；
3. 通用性强，可以应用到各种*NLP*任务中；

缺点

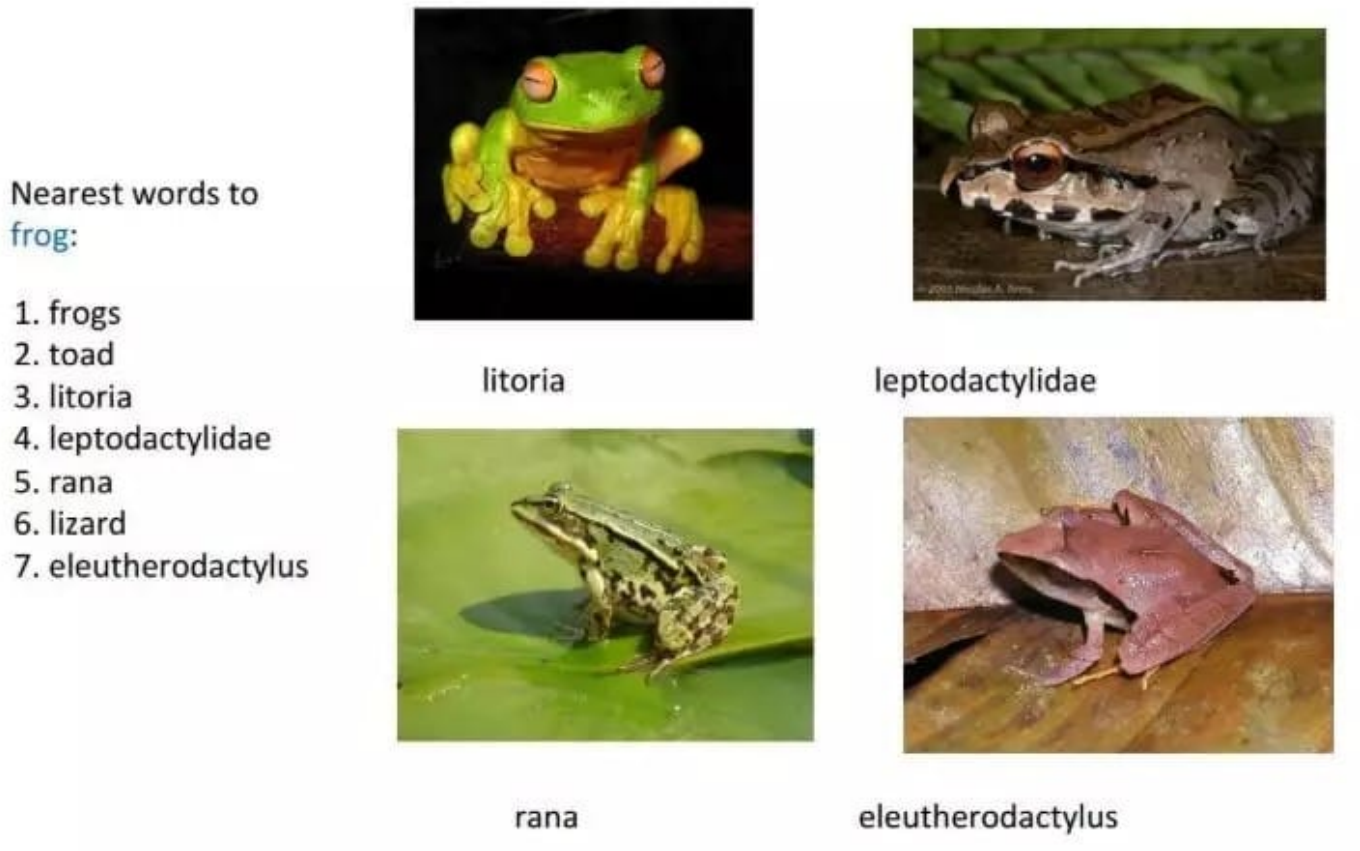
1. 词和向量是一对一的关系，无法解决多义词的问题；
2. word2vec是一种静态的模型，虽然通用性强，但无法真的特定的任务做动态优化；

GloVe

GloVe 是斯坦福大学*Jeffrey*、*Richard* 等提供的一种词向量表示算法，*GloVe* 的全称是*Global Vectors for Word Representation*，是一个基于全局词频统计（*count-based & overall staticstics*）的词表征（*word representation*）算法。该算法综合了global matrix factorization（全局矩阵分解） 和 local context window（局部上下文窗口） 两种方法的优点。

备注：*Glove*模型的推导公式比较复杂，在这里不做详细推导，具体可以查看官网（https://nlp.stanford.edu/projects/glove/）。

效果



优点

1. 考虑到词语的上下文、和全局语料库的信息，学习到了语义和语法的信息；
2. 得到的词向量维度小，节省存储和计算资源；
3. 通用性强，可以应用到各种*NLP* 任务中；

缺点

1. 词和向量是一对一的关系，无法解决多义词的问题；
2. glove也是一种静态的模型，虽然通用性强，但无法真的特定的任务做动态优化；

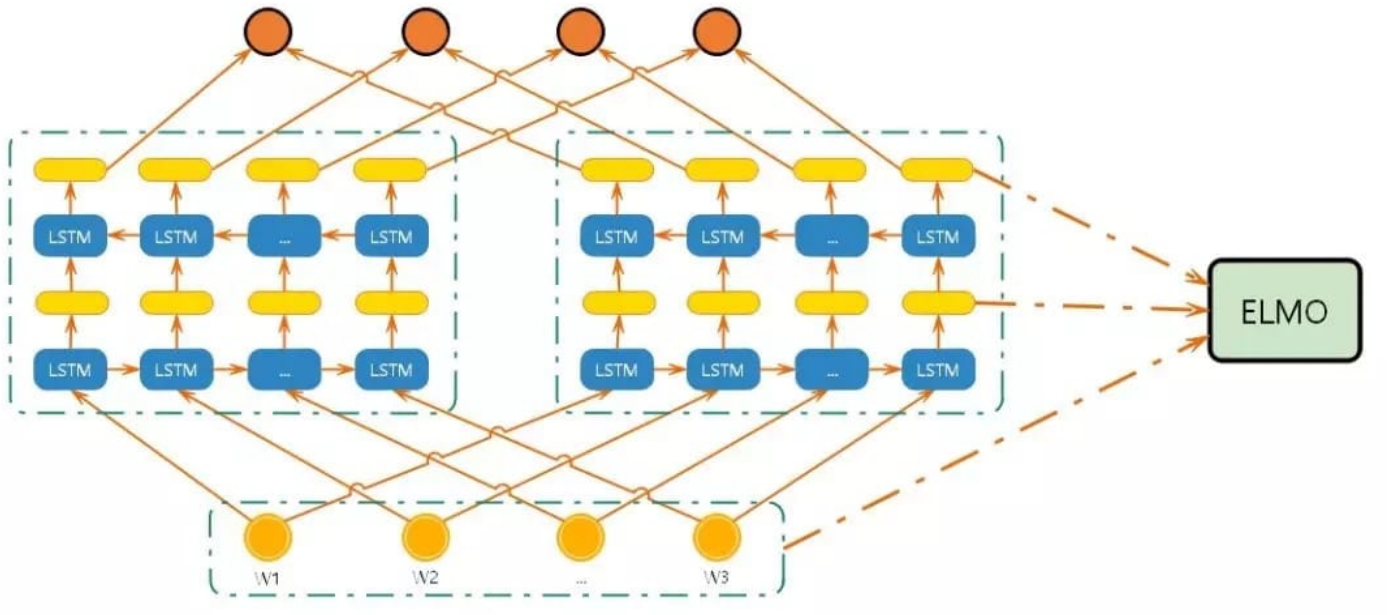
ELMO

word2vec 和 glove 算法得到的词向量都是静态词向量（静态词向量会把多义词的语义进行融合，训练结束之后不会根据上下文进行改变），静态词向量无法解决多义词的问题（如：“我今天买了7斤**苹果**” 和 “我今天买了**苹果**7” 中的 **苹果** 就是一个多义词）。而ELMO模型进行训练的词向量可以解决多义词的问题。

ELMO 的全称是“ *Embedding from Language Models* ”，这个名字不能很好的反映出该模型的特点，提出 *ELMO* 的论文题目可以更准确的表达出该算法的特点“ *Deep contextualized word representation* ”。

该算法的精髓是：用语言模型训练神经网络，在使用*word embedding (https://easyai.tech/ai-definition/word-embedding/)* 时，单词已经具备上下文信息，这个时候神经网络可以根据上下文信息对*word embedding* 进行调整，这样经过调整之后的*word embedding* 更能表达在这个上下文中的具体含义，这就解决了静态词向量无法表示多义词的问题。

网络模型



过程

1. 上图中的结构使用字符级卷积神经网络（*convolutional neural network, CNN (https://easyai.tech/ai-definition/cnn/)*）来将文本中的词转换成原始词向量（*raw word vector*）；

2. 将原始词向量输入双向语言模型中第一层；
3. 前向迭代中包含了该词以及该词之前的一些词汇或语境的信息（即上文）；
4. 后向迭代中包含了该词以及该词之后的一些词汇或语境的信息（即下文）；
5. 这两种迭代的信息组成了中间词向量（*intermediate word vector*）；
6. 中间词向量被输入到模型的下一层；
7. 最终向量就是原始词向量和两个中间词向量的加权和；

效果

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

如上图所示：

- 使用glove训练的词向量中，与 play 相近的词大多与体育相关，这是因为语料中与play相关的语料多时体育领域的有关；
- 在使用elmo训练的词向量中，当 play 取 演出 的意思时，与其相近的也是 演出 相近的句子；

NLP的业务场景

NLP 的4个典型应用

- 文本纠错：识别文本中的错别字，给出提示以及正确的建议
- 情感倾向分析：对包含主观信息的文本进行情感倾向性判断
- 评论观点抽取：分析评论关注点和观点，输出标签
- 对话情绪识别：识别会话者所表现出的情绪类别及置信度
- 文本标签：输出能够反映文章关键信息的多维度标签
- 文章分类：输出文章的主题分类及对应的置信度
- 新闻摘要：抽取关键信息并生成指定长度的新闻摘要

大家不要被这些眼花缭乱的业务场景给搞晕了，其实上面的这些业务都是基于我们之前讲的NLP预处理的输出，只是应用了不同的机器学习模型，比如：SVM、LSTM、LDA等等。

这些机器学习模型大部分是分类模型（序列标注也是一种分类模型），只有少部分是聚类模型。这些模型就是泛化的了，并不只是针对于NLP任务的。要想讲清楚这部分内容，就需要另开一个关于“机器学习入门”的主题，这里就不过多的展开了。

小结：只要大家掌握了NLP的预处理，就算入门NLP了，因为后续的处理都是一些常见的机器学习模型和方法。

结束语

NLP是一个非常有挑战性的工作，同时也是一个非常有发展空间的工作，所以大家只要克服了前期的入门门槛，那么迎接你的将是一片广阔的天地。道阻且长，行则将至。

安利一下掘金小册《深入理解NLP的中文分词：从原理到实践》(https://juejin.im/book/5d9ea8fff265da5b81794756)，它将系统的帮助你学习NLP的中文分词的相关知识，而中文分词对于NLP的重要意义，通过本文大家也应该十分清楚了。掌握中文分词的技术，不仅对于提高NLP任务的结果质量有很大的帮助，同时对于理解机器学习也有很大的促进作用。

领域里的8-种文本表示方式及优缺点 (https://easyai.tech/blog/nlp-%E9%A2%86%E5%9F%9F%E9%87%8C%E7%9A%848-%E7%A7%8D%E6%96%87%E6%9C%AC%E8%A1%A8%E7%A4%BA%E6%96%B9%E5%BC%8F%E5%8F%8A%E4%BC%98%E7%BC%BA%E7%82%B9/)

https://easyai.tech/ai-definition/nlp/ (https://easyai.tech/ai-definition/nlp/)

https://juejin.im/post/5dccbe0ff265da795315a119 (https://juejin.im/post/5dccbe0ff265da795315a119)

智能推荐

Ubuntu 15.10/16.04 上安装Caffe——确保编译好的库相互兼容_Solomon1588的博客-程序员宅基地 (/article/Solomon1558/13161510)

在Ubuntu14.04 LTS版本上编译安装Caffe的教程非常多，安装过程也较为顺利，然而在更新版本系统上编译安装Caffe的过程中，仍会遇到很多问题。其中，在make过程中遇到undefined reference to ‘xxx’，在make runtest 过程中遇到Segmentation faul这两个错误非常常见，在github的BVLC/caffe issues中有非常多类似的问题,e.g.&#1

带你重新认识指针(下)_龙跃十二的博客-程序员宅基地 (/article/qq_38646470/13161506)

指针一直是一个头疼的问题，指针灵活性很好，但是可控性很差，容易写出程序崩溃的代码。我们要怎样才能使的指针可控呢？龙叔带你重新认识指针

项目管理与人员沟通问题_weixin_30604651的博客-程序员宅基地 (/article/weixin_30604651/13161500)

项目做了一段时间了，所遇到的问题除技术问题外，很大一部分是内部沟通问题，如果有一个因明的leader，就不用多说了，但大多数情况不如我们所愿，经过了痛苦的交流，还是没有达到一个统一的认识，这样麻烦就会接踵而至，总是抱着不能得罪人得心态，以为自己在踏实做事了，就ok了，一段时间坚持下来，情况却不像我预计的那样。 做事，该怎样干就怎样干，不会说可以学着说，慢慢说，不然要磨合干什...

---

listview滑动置顶效果_bobo89455100的博客-程序员宅基地 (/article/bobo89455100/13161499)

listview滑动置顶

---

python svm pca实践（一）_qq_43440169的博客-程序员宅基地 (/article/qq_43440169/13161497)

https://blog.csdn.net/qq_25819827/article/details/80301973

---

oracle手工收集awr报告_Oracle AWR报告生成步骤_书生夜行的博客-程序员宅基地 (/article/weixin_42297746/13161493)

11.2.0.4版本的1以oracle用户登录系统2执行sqlplus / as sysdba3运行@?/rdbms/admin/awrrpt.sql$sqlplus / as sysdbasql>@?/rdbms/admin/awrrpt.sql则会提示选择报告类型有text，html 2种，默认html，直接回车或者输入html都可。然后后提示要几天的，oracle11g默认最多保存8天...

随便推点

python什么为假_python对于真和假的定义_weixin_39639698的博客-程序员宅基地 (/article/weixin_39639698/109930285)

python对于真和假的定义1.简要说明在python中一切都是对象。我们知道，要进行逻辑运算，就要知道真和假。高中的数学就知道了，真 and 真=真，真 and 假 =假。以及真 or 假 = 真。总结起来就是，进行逻辑与运算的时候，两个必须是真才能是真，否则结果为假。 进行逻辑或运算的时候，必须是至少有一个为真则结果为真，否则为假。这就是最初的基础知识。我们不禁要问，到底什么才是真和假，在py...

Linux下的软件的安装_weixin_33913377的博客-程序员宅基地 (/article/weixin_33913377/93784562)

一、yum相关介绍及命令【1】 用来管理软件的一个命令。通过该命令完成对软件的下载、卸载等操作。【2】 命令：（必须在yum搭建成功后才能正常运行） yum clean all 清楚原有缓存 yum repolist 列出仓库信息 yum install software 安装 yum list softwar...

业务逻辑处理_ht121907的博客-程序员宅基地_逻辑处理 (/article/ht121907/79433839)

功能的实现，都是依靠业务逻辑来完成的，记得看过不能完成业务逻辑的程序员都不会成长的，确实是的，最近在完成业务逻辑的时候，程序的业务判断有很多的，所以开始接触，设计模式，看到来一些设计模式，看结合项目，确实是可以根据设计模式来改写的，so，懂得设计模式可以快速的，写好的代码的。关于函数同步和异步之间的区别，就是一个是你买煎饼的时候，需要等待，一个是你需要吃煎饼的时候，自然就来的，都有好处，也要看...

使用淘宝IP库获取用户ip地理位置_weixin_34343308的博客-程序员宅基地 (/article/weixin_34343308/91813415)

为什么80%的码农都做不了架构师？>>> ...

---

pytest文档30-功能用例与自动化用例完美对接(allure)_上海-悠悠的博客-程序员宅基地 (/article/qq_27371025/118269894)

前言做自动化做久了，经常会思考一个问题，到底别人是怎么做的自动化，跟自己的有啥不一样，看过不少书和资料，都是停留在demo的层面。真正把自动化做的好的大牛又不屑于分享自己的劳动成果，所以大部分情况就是一群菜鸡在群里互啄，停留在初级入门的demo层面上。到底自动化要达到什么样的效果呢？这里我把最近的研究成果分享下，有经验的小伙伴也可以一起交流下。功能用例我一直认为一切的自动化用例是基于功能测试用例的， 脱离了功能测试用例，你的代码写的再漂亮，那也仅仅是show代码的。面试的时候经常会遇到一个问题，

Elasticsearch集群安装_风间净琉璃的博客-程序员宅基地 (/article/u011937566/122285932)

docker安装elasticsearch集群、安装elastic-head、安装kibana。

推荐文章

百度知道推客户端APP 同时支持iOS及Android_weixin_34018202的博客-程序员宅基地 (/article/weixin_34018202/90001292)

《JavaScript 高级程序设计》第四章：变量、作用域和内存问题_weixin_30741653的博客-程序员宅基地 (/article/weixin_30741653/95942101)

ckplayer 参数设置详解_weixin_34278190的博客-程序员宅基地 (/article/weixin_34278190/92323810)

props传递对象_props传递数据_weixin_40001805的博客-程序员宅基地 (/article/weixin_40001805/111907204)

sugarcrm php版本,安装SugarCRM 6.5版本_仙夜子的博客-程序员宅基地 (/article/weixin_35765226/116281829)

设置ant 打war包部署到tomcat并运行_MissHannahDong的博客-程序员宅基地 (/article/db308950372/9225539)

Flask—02-Flask会话控制与模板引擎_badiu_30394251的博客-程序员宅基地 (/article/weixin_30394251/97756200)

OGRE 3D程序设计（1）_无敌的成长日记的博客-程序员宅基地 (/article/jimoshuicao/7940941)

热门文章

机器学习第8章（集成学习）_罗辑罗辑的博客-程序员宅基地_机器学习第八章 (/article/jinhualun911/108811731)

java web servlet jsp_《Servlet/JSP深入详解—基于Tomcat的Web开发》PDF 下_美丽的贝丽珠的博客-程序员宅基地 (/article/weixin_32413177/114168622)

springmvc 日志打印sql_Spring mvc+mybatis+Log4j控制台打印sql语句_爱新觉罗维川的博客-程序员宅基地 (/article/weixin_28775381/112819849)

总结-办事顺序_weixin_34318272的博客-程序员宅基地 (/article/weixin_34318272/94681974)

Django REST Framework——1. API和RESTful接口规范_花_城的博客-程序员宅基地_django 接口规范 (/article/qq_39330486/121894143)

用VS 2003 制作安装文档_weixin_30446197的博客-程序员宅基地 (/article/weixin_30446197/95305253)

第二章 Hadoop序列化_琉璃百般枯的博客-程序员宅基地 (/article/qq_38358499/117455833)

Web安全深度剖析第一、二章读书笔记_Moplit的博客-程序员宅基地 (/article/weixin_45118086/117535188)

相关标签

NLP (/searchArticle?qc=NLP&page=1)

深度学习 (/searchArticle?qc=深度学习&page=1)