# Report of Deep learning for Natural language Processing

唐宗润 ZY2303211

# Abstract

阅读 Entropy Of English，计算中文(分别以词和字为单位) 的平均信息熵

# Introduction

信息熵是信息论中的一个重要概念，用于衡量一组数据的不确定性或混乱程度。它最初由香农（Claude Shannon）在 1948 年引入，用于描述信息的平均编码长度以及信息传输的效率。

信息熵的计算公式如下：

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

其中，H 是信息熵，$p_i$ 是事件 i 发生的概率，n 是事件的总数。

信息熵的直观解释是，对于一个具有多种可能结果的系统，信息熵越大，表示系统的不确定性越高，即我们需要更多的信息来描述这个系统。反之，信息熵越小，表示系统的确定性越高，即我们需要较少的信息来描述这个系统。

信息熵在各个领域都有着重要的应用，例如：

通信和信息理论：在通信系统中，信息熵用于衡量消息的平均信息量，从而确定有效的编码方案。较低的信息熵意味着更高的压缩率和更高的传输效率。

数据压缩：信息熵可以用于设计高效的数据压缩算法，例如 Huffman 编码和 Arithmetic 编码等。

机器学习和数据挖掘：信息熵常用于决策树算法中，作为评估特征的重要度和划分数据集的依据。

密码学：在密码学中，信息熵用于衡量密码的随机性和安全性，从而评估密码系统的强度。

总之，信息熵是一个十分重要的概念，它提供了一种量化描述数据不确定性的方法，广泛应用于各种领域，对于理解和分析信息系统具有重要意义。

# Methodology

首先读取停词表并进行数据预处理

```
# 读取停用词表
with open('./cn_stopwords.txt', 'r', encoding='utf-8') as f:
    stopwords = set([line.strip() for line in f])
#读取中文文档库中的所有文档
corpus_path = 'D:/zongruntang/nlp/jyxstxtqj_downcc.com'
filenames = os.listdir(corpus_path)
docs = []
for filename in filenames:
    with open(os.path.join(corpus_path, filename), 'r', encoding='ansi') as f:
        text = f.read()
        # 分词并过滤停用词
        words = [word for word in jieba.cut(text) if word not in stopwords]
        docs.append(words)
# 将所有文档的分词结果合并为一个列表
all_words = []
for doc in docs:
    all_words.extend(doc)
# 去重
unique_words = set(all_words)
```

其次，定义并计算一元信息熵函数

```
# 定义计算汉字信息熵的函数
def calc_chinese_entropy(text):
    # 统计每个汉字出现的次数
    char_count = {}
    for char in text:
        if '\u4e00' <= char <= '\u9fa5':
            char_count[char] = char_count.get(char, 0) + 1

    # 计算每个汉字出现的概率
    char_prob = {}
    for char in char_count:
        char_prob[char] = char_count[char] / len(text)

    # 计算信息熵
    entropy = 0
    for prob in char_prob.values():
        entropy -= prob * math.log(prob, base: 2)

    return entropy
```

定义并计算二元信息熵函数

```python
# 定义计算汉字二元词组信息熵的函数
def calc_chinese_bigram_entropy(text):
    # 统计每个汉字二元词组出现的次数
    bigram_count = {}
    bigram_len = 0
    char_count = {}
    for char in text:
        if '\u4e00' <= char <= '\u9fa5':
            char_count[char] = char_count.get(char, 0) + 1

    for i in range(len(text) - 1):
        if '\u4e00' <= text[i] <= '\u9fa5' and '\u4e00' <= text[i + 1] <= '\u9fa5':
            bigram_count[(text[i], text[i + 1])] = bigram_count.get((text[i], text[i + 1]), 0) + 1
            bigram_len+=1

    # 计算每个汉字二元词组出现的概率
    bigram_prob = {}
    entropy = []
    for bigram in bigram_count.items():
        #bigram_prob[bigram] = bigram_count[bigram] / (bigram_len - 1)
        jp_xy = bigram[1] / bigram_len  # 计算联合概率p(x,y)
        cp_xy = bigram[1] / char_count[bigram[0][0]]  # 计算条件概率p(x|y)
        entropy.append(-jp_xy * math.log(cp_xy, base: 2))  # 计算二元模型的信息熵
```

定义并计算三元信息熵函数

```python
    # 计算每个汉字二元词组出现的概率
    bigram_prob = {}
    entropy = []
    for bigram in bigram_count.items():
        #bigram_prob[bigram] = bigram_count[bigram] / (bigram_len - 1)
        jp_xy = bigram[1] / bigram_len  # 计算联合概率p(x,y)
        cp_xy = bigram[1] / char_count[bigram[0][0]]  # 计算条件概率p(x|y)
        entropy.append(-jp_xy * math.log(cp_xy, base: 2))  # 计算二元模型的信息熵

    # 计算信息熵
    return round(sum(entropy), 6)

# 定义计算汉字三元词组信息熵的函数
def calc_chinese_trp_entropy(text):
    # 统计每个汉字三元词组出现的次数
    bigram_count = {}
    bigram_len = 0
    trp_count = {}
    trp_len = 0
    for i in range(len(text) - 2):
        if '\u4e00' <= text[i] <= '\u9fa5' and '\u4e00' <= text[i + 1] <= '\u9fa5' and '\u4e00' <= text[i + 2] <= '\u9fa5':
```

最后分别计算金庸各个著作的信息熵，并基于此计算总文库的一元、二元、三元信息熵。

# Conclusion

文档 1 的一元信息熵为：3.2525552116075365
文档 2 的一元信息熵为：12.359217185079066

文档 3 的一元信息熵为：12.455754699791575
文档 4 的一元信息熵为：11.364784728393367
文档 5 的一元信息熵为：12.274530388085248
文档 6 的一元信息熵为：12.220157692339416
文档 7 的一元信息熵为：11.888218481776933
文档 8 的一元信息熵为：8.811267271590175
文档 9 的一元信息熵为：12.488563889114507
文档 10 的一元信息熵为：11.797105897997941
文档 11 的一元信息熵为：12.067645766923926
文档 12 的一元信息熵为：11.7971066923926
文档 13 的一元信息熵为：8.41360268955823
文档 14 的一元信息熵为：11.237631557127937
文档 15 的一元信息熵为：11.281664051368855
文档 16 的一元信息熵为：12.249904047663705
文档 17 的一元信息熵为：9.103708089338893
文档 18 的一元信息熵为：11.697109566716737
文档 1 的二元信息熵为：12.24990
文档 2 的二元信息熵为：1.812049
文档 3 的二元信息熵为：4.152879
文档 4 的二元信息熵为：4.012654
文档 5 的二元信息熵为：4.701541
文档 6 的二元信息熵为：4.848591
文档 7 的二元信息熵为：4.608831
文档 8 的二元信息熵为：2.916233
文档 9 的二元信息熵为：3.97712
文档 10 的二元信息熵为：4.791778
文档 11 的二元信息熵为：4.854165
文档 12 的二元信息熵为：4.848565
文档 13 的二元信息熵为：1.770923
文档 14 的二元信息熵为：3.60526
文档 15 的二元信息熵为：3.083038
文档 16 的二元信息熵为：4.051713
文档 17 的二元信息熵为：2.176664
文档 18 的二元信息熵为：5.024732
文档 1 的三元信息熵为：0.519064
文档 2 的三元信息熵为：0.090574
文档 3 的三元信息熵为：0.500141
文档 4 的三元信息熵为：0.519701
文档 5 的三元信息熵为：0.646318
文档 6 的三元信息熵为：0.672178
文档 7 的三元信息熵为：0.534064
文档 8 的三元信息熵为：0.359809
文档 9 的三元信息熵为：0.432002
文档 10 的三元信息熵为：0.643233

文档 11 的三元信息熵为：0.799899
文档 12 的三元信息熵为：0.643899
文档 13 的三元信息熵为：0.249675
文档 14 的三元信息熵为：0.372712
文档 15 的三元信息熵为：0.293971
文档 16 的三元信息熵为：0.464051
文档 17 的三元信息熵为：0.233856
文档 18 的三元信息熵为：0.842603
整个文库的汉字信息熵为： 12.739912482993983
整个文库的汉字二元词组信息熵为： 6.541074
整个文库的汉字三元词组信息熵为： 1.181922