

# NLP-4

---

罗志钊 pt2100565

## NLP-4

[问题描述](#)

[预备知识](#)

[word2vec基本思想](#)

[K-Means](#)

[问题分析](#)

[实验过程](#)

[实验结果](#)

[心得](#)

[参考内容](#)

[附录](#)

[实验代码](#)

[实验结果（完整版）](#)

## 问题描述

---

利用给定语料库（或者自选语料库），利用神经语言模型（如：Word2Vec，GloVe等模型）来训练词向量，通过对词向量的聚类或者其他方法来验证词向量的有效性。

## 预备知识

---

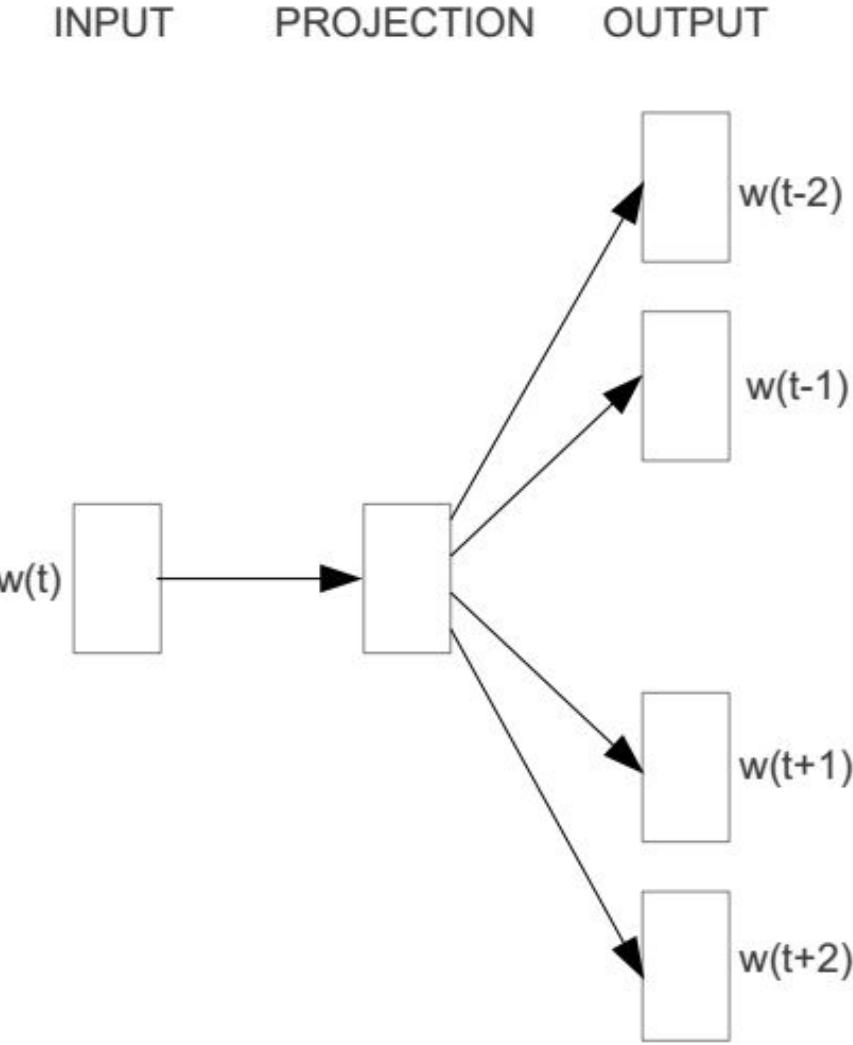
### word2vec基本思想

用来词预测词，准确的来说，word2vec是一种编码方式，将一个个的词给编码成向量，但是被他编码而成的向量不是随便生成的，而是能够体现这些单词之间的关系。

word2vec主要包含两个模型：

- 跳字模型：用当前词来预测上下文

skip-gram模型的输入是一个单词 $w_i$ ，它的输出是 $w_i$ 的上下文 $w_{-O}, w_{-O+1}, \dots, w_{-O+C}$ ，上下文的窗口大小为 $C$ 。举个例子，这里有个句子“*I drive my car to the store*”。我们如果把“car”作为训练输入数据，单词组{“I”，“drive”，“my”，“to”，“the”，“store”}就是输出。所有这些单词，我们会进行one-hot编码。skip-gram模型图如下所示：



## Skip-gram

- 连续词袋模型（本次使用的模型）：通过上下文来预测当前值。

学习目标是最大化对数似然函数

$$L = \sum_{w \in C} \log p(w | \text{Context}(w))$$

其中， $w$ 表示语料库C中任意一个词

网络计算的步骤：

- 输入层：上下文单词的onehot。（假设单词向量空间dim为V，上下文单词个数为C）
- 所有onehot分别乘以共享的输入权重矩阵W（V\*N矩阵，N为自己设定的数，初始化权重矩阵W）
- 所得的向量（注意onehot向量乘以矩阵的结果）相加求平均作为隐层向量，size为1\*N.
- 乘以输出权重矩阵W
- 得到向量 V 激活函数处理得到V-dim概率分布 {PS: 因为是onehot嘛，其中的每一维都代表着一个单词}，概率最大的index所指示的单词为预测出的中间词（target word）

- 与true label的onehot做比较，误差越小越好。loss function (一般为交叉熵代价函数)

## K-Means

K-Means算法是无监督的聚类算法

- 从n个数据中随机选择 k 个对象作为初始聚类中心；
- 根据每个聚类对象的均值（中心对象），计算每个数据点与这些中心对象的距离；并根据最小距离准则，重新对数据进行划分；
- 重新计算每个有变化的聚类簇的均值，选择与均值距离最小的数据作为中心对象；
- 循环步骤2和3，直到每个聚类簇不再发生变化为止。

## 问题分析

我把问题拆解为三部分。第一部分是预处理，由于是聚类分析，依旧需要去除大量的stop words，这一步骤和实验三类似。第二部分是之后利用word2vec进行训练，得到分词模型。第三部分是关联验证，最后用gensim库中的similarity用于处理相关词的关联，用于得到某个词相关度较高的一下词及其关联度；用k-means算法进行处理，得到聚类结果并绘图，即验证词向量的有效性。本次数据集选用射雕英雄传。

## 实验过程

### 1. 数据预处理

利用jieba对文本进行处理，去除标点符号以及stop words，核心代码如下。

Ps.data1是原数据文件夹，data2存放的是处理的分词和射雕vec，用于模型训练和得到聚类图表。

```
def dispose(data1, data2): # 读取语料内容
    content = []
    # names = os.listdir(path_in)
    sw = [ "的", "了", "在", "是", "我", "有", "和", "就",
           "不", "人", "都", "一", "一个", "上", "也", "很", "到", "说", "要", "去",
           "你",
           "会", "着", "没有", "看", "好", "自己", "这", "罢", "这", '在', '又', '在',
           '得', '那', '他', '她', '不', '而', '道', '与', '之',
           '见', '却', '问', '可', '但',
           '没', '啦', '给', '来', '既', '叫', '只', '中', '么', '便',
           '听', '为', '跟', '个', '甚', '下', '还', '过', '向', '如此',
           '已', '位', '对', '如何', '将', '岂', '哪', '似', '以免', '均',
           '虽然', '即', '由', '再', '使', '从', '麼', '其实', '阿',
           '被', '当', '里', '时', '虽', '远', '轻', '凑', '兄' ]

    for line in open(data1, 'r', encoding='UTF-8', errors='ignore'):
        line.strip('\n')
        line = re.sub('\s', ' ', line)
        line = re.sub('[\u0000-\u4DFF]', ' ', line)
        line = re.sub('[\u9FA6-\uFFFF]', ' ', line)
        line = re.sub('[\u9FA6-\uFFFF]', ' ', line)
        for a in sw:
            line = re.sub(a, ' ', line)
```

```

if len(line) == 0:
    continue
seg_list = list(jieba.cut(line, cut_all=False))
line_seg = ""
for term in seg_list:
    line_seg += term + " "
con = jieba.cut(line, cut_all=False) # 结巴分词
# content.append(con)
content.append(" ".join(con))
with open(data2, "w", encoding='UTF-8', errors='ignore') as f:
    f.writelines(content)
return content

```

## 2. word2vec训练

使用开源的Gensim库提供的接口来训练Word2vec模型，本次使用连续词袋模型。

```

model = Word2Vec(sentences=LineSentence(book), hs=1, min_count=10, window=5,
vector_size=200, sg=0, epochs=200)

```

## 3. similarity函数和k-means处理

这里分析三部分：姓名、武功和势力进行聚类分析。具体内容如下

```

word_name = ['郭靖', '黄蓉', '杨康']
word_Kungfu = ['降龙十八掌', '九阴真经', '蛤蟆功']
word_group= ['丐帮', '桃花岛', '蒙古']

```

- 利用similarity()得到各词之间关联度最高的十个词，篇幅有限，核心代码如下

```

for i in range(0,3):
    print()
    print("人物名字: "+ word_name[i])
    for result in model.wv.similar_by_word(word_name[i], topn=10):
        print(result[0], result[1])
for i in range(0, 3):
    print()
    print("武功: "+ word_Kungfu[i])
    for result in model.wv.similar_by_word(word_Kungfu[i], topn=10):
        print(result[0], result[1])
for i in range(0, 3):
    print()
    print("特征: "+ word_group[i])
    for result in model.wv.similar_by_word(word_group[i], topn=10):
        print(result[0], result[1])

```

这里展示部分内容

人物名字: 郭靖
黄蓉 0.4566042721271515
周伯通 0.34426403045654297
洪七公 0.3359399437904358
欧阳克 0.32554471492767334
老毒物 0.28453922271728516
七 0.2688484489917755
傻姑 0.2544213533401489
黄药师 0.2490149587392807
始终 0.2465125471353531
陆冠英 0.2427675575017929

这里很清晰得到，郭靖和黄蓉以及几位师傅之间关联最高，是符合射雕里人物成长历程的。

特征: 蒙古
军官 0.2392040193080902
土地 0.232521653175354
蒲儿帖 0.22722633183002472
昂然 0.22293338179588318
急 0.22075749933719635
藏身 0.21864937245845795
知些 0.2177640199661255
无 0.19529566168785095
番话 0.19274704158306122
王重阳 0.19180046021938324

从蒙古势力中得到军官，土地还有王重阳这些词有较高关联度，王重阳早年去蒙古，而蒙古在射雕系列中连年征战，夺取大量土地，因而是符合的。

- k-means得到词之间的关联图

根据以上九个词的部分相关词进行拓展，进行k-means聚类图分析，包括拓展在内到所有词如下

```
[ '郭靖', '黄蓉', '杨康', '洪七公', '周伯通', '欧阳锋', '降龙十八掌', '九阴真经', '蛤蟆功',
'丐帮', '桃花岛', '蒙古' ]
```

### 核心代码

```
def cluster(vector):
    with open('./data2/射雕vec.txt', 'rb') as f:
        vec_dist = pkl.load(f)
    vec = []
    for d in vector:
        vec.append(vec_dist[d])
    center, label, inertia = k_means(vec, n_clusters=3)
    vec = PCA(n_components=2).fit_transform(vec)
```

```
#plt.rcParams['font.sans-serif'] = ['Arial Unicode MS']
plt.rcParams['font.family'] = ['Arial Unicode MS']
plt.rcParams['axes.unicode_minus'] = False
plt.scatter(vec[:, 0], vec[:, 1], cmap='viridis', c=label, alpha=0.5)
for i, w in enumerate(vector):
    plt.annotate(text=w, xy=(vec[:, 0][i], vec[:, 1][i]),
                 xytext=(vec[:, 0][i] + 0.01, vec[:, 1][i] + 0.01))
plt.colorbar()
plt.show()
```

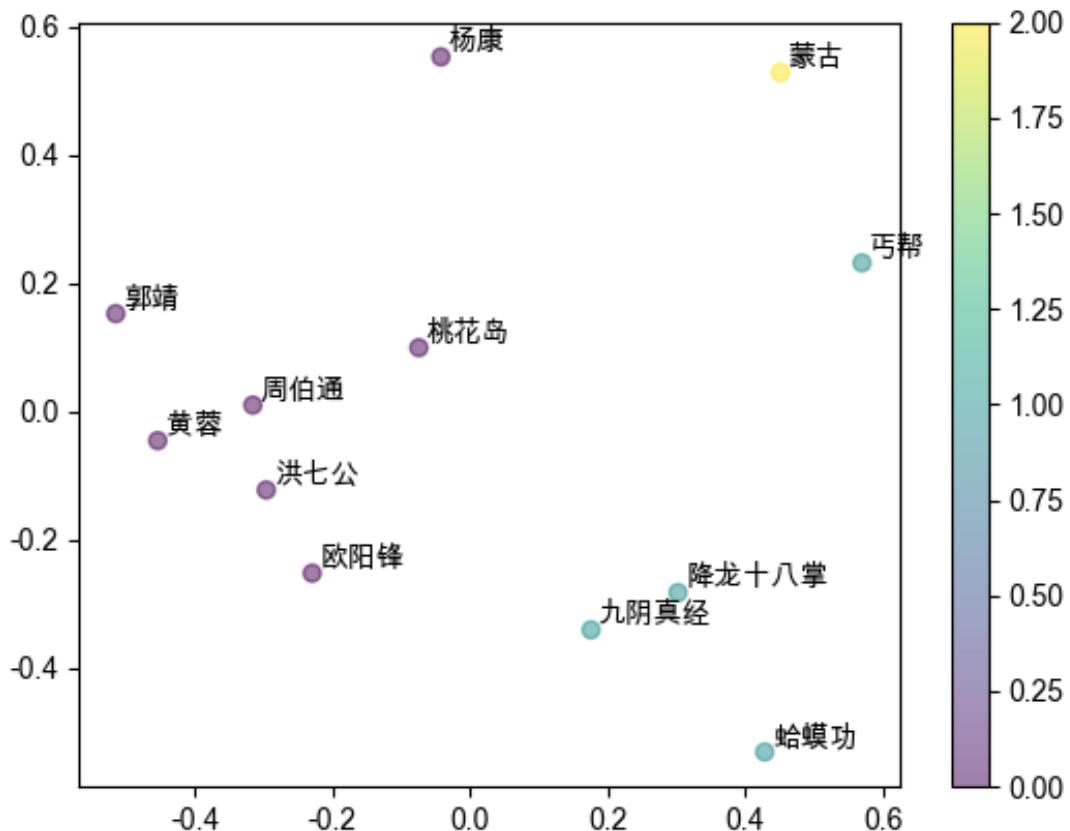
## 实验结果

---

1. 关于郭靖、黄蓉、杨康、九阴真经、降龙十八掌、蛤蟆功、丐帮、桃花岛和蒙古这九个词相关十个词分析如下，完整的实验结果放入附录中

人物名字: 郭靖	人物名字: 黄蓉	人物名字: 杨康
黄蓉 0.4566042721271515	郭靖 0.4566042721271515	活 0.2687632441520691
周伯通 0.34426403045654	洪七公 0.319037199020385	杨康心 0.23546013236045
洪七公 0.33593994379043	黄蓉笑 0.313349723815917	自身 0.2227800935506826
欧阳克 0.32554471492767	周伯通 0.306428104639053	遗物 0.2226532399654388
老毒物 0.28453922271728	海滩 0.2639518082141876	右肩 0.2226406931877136
七 0.2688484489917755	欧阳锋 0.252117812633514	薄 0.21840925514698029
傻姑 0.2544213533401489	葫芦 0.2515917718410492	棒 0.20497046411037445
黄药师 0.24901495873928	著 0.24856530129909515	身世 0.2010780572891235
始终 0.2465125471353531	老顽童 0.246609985828399	酒 0.1985650360584259
陆冠英 0.24276755750179	悄声 0.2456432431936264	忙 0.1972237080335617
武功: 降龙十八掌	武功: 九阴真经	武功: 蛤蟆功
剑法 0.2659899294376373	葫芦 0.2555567622184753	百倍 0.2208737730979919
满天 0.2613832950592041	空明拳 0.25142940878868	空明拳 0.21466831862926
急 0.24910227954387665	真经 0.2463907897472381	三分 0.2088710218667984
凭 0.241050124168396	洪七公 0.24136364459991	钹 0.20452967286109924
终身 0.2385780811309814	耳边 0.2401223629713058	反正 0.2039959132671356
掌 0.23788118362426758	周伯通 0.23412020504474	两条 0.2012603580951696
大事 0.2367507666349411	周大哥 0.22377838194370	额头 0.1972479224205017
五行 0.2348045706748962	所 0.22186201810836792	彷徨 0.1942408978939056
解释 0.2282589077949524	参仙 0.2111066281795501	啸声 0.1922751069068908
梁子翁 0.225610882043838	以 0.2088114321231842	臂力 0.1899264454841613
特征: 丐帮	特征: 桃花岛	特征: 蒙古
污衣 0.251438885927200	想想 0.281222403049469	军官 0.2392040193080902
招呼 0.2503372430801391	桃花 0.2547352612018585	土地 0.232521653175354
大小 0.2503257393836975	互 0.24399301409721375	蒲儿帖 0.227226331830024
二丐 0.2446162700653076	死活 0.2419787794351577	昂然 0.2229333817958831
攻击 0.2360644787549971	忽然 0.2406994700431823	急 0.22075749933719635
皇帝 0.2313179522752761	研习 0.2364480197429657	藏身 0.2186493724584579
黄二 0.2291396558284759	老毒物 0.235091730952262	知些 0.2177640199661255
死伤 0.2099188417196273	老化 0.2341057658195495	无 0.19529566168785095
继承 0.2052703648805618	陈玄风 0.227550148963928	番话 0.1927470415830612
大事 0.2046264111995691	牵 0.22553607821464539	王重阳 0.191800460219383

2. k-means词相关



## 心得

- 在去除stop words后，依然有大量我不需要的词参入，比如，杨康里的“活”、“薄”，不明其意，效果不好。理想的情况应该是郭靖一词的相关词，以完整的名词为主。通过对stop words表进行添加，去除这些单个的词。但是对于取得名词，我并没有好的想法，涉及到对词性进行分析，相对麻烦。
- 在郭靖一词的分析中，出现了诸如黄蓉、洪七公、欧阳克等人名字，以及对欧阳锋的特有称呼“老毒物”，这些人和郭靖在小说中的相关度确实很高；黄蓉和郭靖在小说中前中期相关度很高，基本关联词也类似；“九阴真经”一词关联度高的有洪七公周伯通等人，周伯通精通九阴真经，在桃花岛把武功传给了靖儿，诸如周大哥、周伯通、空明拳在九阴真经相关词出现较高便有了合适的解释。后期七公武功被欧阳锋所废，借用九阴真经功力恢复，所以和洪七公关联度也较高；“桃花岛”一词相关度较高的有老毒物，陈玄风，桃花，老毒物可能是欧阳锋在岛上的时候，郭靖黄蓉洪七公对他的称呼比较固定，所以和桃花岛相关度比较高。“降龙十八掌”“杨康”“蛤蟆功”“丐帮”这些词结果不明确，可能的原因是因为这些词在小说中出现的频率较低，分析结果相对较差
- 通过对‘郭靖’, ‘黄蓉’, ‘杨康’, ‘洪七公’, ‘周伯通’, ‘欧阳锋’, ‘降龙十八掌’, ‘九阴真经’, ‘蛤蟆功’, ‘丐帮’, ‘桃花岛’, ‘蒙古’这些词进行聚类分析，得到发现黄蓉、周伯通、洪七公、欧阳锋这些人关联度较高；而九阴真经、降龙十八掌和蛤蟆功关联度较高，而丐帮、蒙古这些词没什么关联度，这也符合我的预期猜想。
- 通过学习word2vec模型和k-means，对word embedding模型进行训练，得到了较为准确的聚类结果。然而其中对于其有效性的判断是建立在我比较了解射雕英雄传这个小说的基础来进行的，这种分析虽然可行，但却是以验证性去验证实验结果，我觉得还是有一定的局限性。

## 参考内容

- <https://blog.csdn.net/u010665216/article/details/78721354>
- [https://blog.csdn.net/weixin\\_46474921/article/details/123783987](https://blog.csdn.net/weixin_46474921/article/details/123783987)

3. [https://blog.csdn.net/weixin\\_50891266/article/details/116750204?spm=1001.2014.3001.5502](https://blog.csdn.net/weixin_50891266/article/details/116750204?spm=1001.2014.3001.5502)

## 附录

### 实验代码

```
# -*- coding: utf-8 -*-
import jieba
import re
import numpy as np
from sklearn.cluster import k_means
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import pickle as pkl
from gensim.models import Word2Vec
from gensim.models.word2vec import LineSentence
def dispose(data1, data2): # 读取语料内容
    content = []
    sw = [ "的", "了", "在", "是", "我", "有", "和", "就",
           "不", "人", "都", "一", "一个", "上", "也", "很", "到", "说", "要", "去", "你",
           "会", "着", "没有", "看", "好", "自己", "这", "罢", "这", "在", "又", "在", "得",
           '那', '他', '她', '不', '而', '道', '与', '之',
           '见', '却', '问', '可', '但',
           '没', '啦', '给', '来', '既', '叫', '只', '中', '么', '便',
           '听', '为', '跟', '个', '甚', '下', '还', '过', '向', '如此',
           '已', '位', '对', '如何', '将', '岂', '哪', '似', '以免', '均',
           '虽然', '即', '由', '再', '使', '从', '麼', '其实', '阿',
           '被', '当', '里', '时', '虽', '远', '轻', '凑', '兄' ]
    for line in open(data1, 'r', encoding='UTF-8', errors='ignore'):
        line.strip('\n')
        line = re.sub('\s', ' ', line)
        line = re.sub('[\u0000-\u4dff]', ' ', line)
        line = re.sub('[\u9fa6-\uffff]', ' ', line)
        line = re.sub('[\u9fa6-\uffff]', ' ', line)
        for a in sw:
            line = re.sub(a, ' ', line)
        if len(line) == 0:
            continue
        seg_list = list(jieba.cut(line, cut_all=False))
        line_seg = ""
        for term in seg_list:
            line_seg += term + " "
        con = jieba.cut(line, cut_all=False) # 结巴分词
        # content.append(con)
        content.append(" ".join(con))
    with open(data2, "w", encoding='UTF-8', errors='ignore') as f:
        f.writelines(content)
    return content
```

```

def cluster(vector):
    with open('./data2/射雕vec.txt', 'rb') as f:
        vec_dist = pkl.load(f)
    vec = []
    for d in vector:
        vec.append(vec_dist[d])
    center, label, inertia = k_means(vec, n_clusters=3)
    vec = PCA(n_components=2).fit_transform(vec)

    #plt.rcParams['font.sans-serif'] = ['Arial Unicode MS']
    plt.rcParams['font.family'] = ['Arial Unicode MS']
    plt.rcParams['axes.unicode_minus'] = False
    plt.scatter(vec[:, 0], vec[:, 1], cmap='viridis', c=label, alpha=0.5)
    for i, w in enumerate(vector):
        plt.annotate(text=w, xy=(vec[:, 0][i], vec[:, 1][i]),
                     xytext=(vec[:, 0][i] + 0.01, vec[:, 1][i] + 0.01))
    plt.colorbar()
    plt.show()

if __name__ == '__main__':
    book='./data2/射雕英雄传.txt'
    dispose("./data1/射雕英雄传.txt", "./data2/射雕英雄传.txt")
    # model = Word2Vec(data_txt, vector_size=400, window=5, min_count=5, epochs=200,
    workers=multiprocessing.cpu_count())

    word_name = ['郭靖', '黄蓉', '杨康']
    word_Kungfu = ['降龙十八掌', '九阴真经', '蛤蟆功']
    word_group= ['丐帮', '桃花岛', '蒙古']

    print(book)
    model = Word2Vec(sentences=LineSentence(book), hs=1, min_count=10, window=5,
vector_size=200, sg=0, epochs=200)
    model.wv.vectors = model.wv.vectors / (np.linalg.norm(model.wv.vectors,
axis=1).reshape(-1, 1))
    vec_dist = dict(zip(model.wv.index_to_key, model.wv.vectors))
    with open('./data2/射雕vec.txt', 'wb') as f:
        pkl.dump(vec_dist, f)

    for i in range(0,3):
        print()
        print("人物名字: "+ word_name[i])
        for result in model.wv.similar_by_word(word_name[i], topn=10):
            print(result[0], result[1])
    for i in range(0, 3):
        print()
        print("武功: "+ word_Kungfu[i])
        for result in model.wv.similar_by_word(word_Kungfu[i], topn=10):

```

```
    print(result[0], result[1])
for i in range(0, 3):
    print()
    print("特征: " + word_group[i])
    for result in model.wv.similar_by_word(word_group[i], topn=10):
        print(result[0], result[1])
cluster(['郭靖', '黄蓉', '杨康', '洪七公', '周伯通', '欧阳锋', '降龙十八掌', '九阴真经', '蛤蟆功', '丐帮', '桃花岛', '蒙古'])
```

## 实验结果（完整版）

人物名字: 郭靖

黄蓉 0.4566042721271515  
周伯通 0.34426403045654297  
洪七公 0.3359399437904358  
欧阳克 0.32554471492767334  
老毒物 0.28453922271728516  
七 0.2688484489917755  
傻姑 0.2544213533401489  
黄药师 0.2490149587392807  
始终 0.2465125471353531  
陆冠英 0.2427675575017929

人物名字: 黄蓉

郭靖 0.4566042721271515  
洪七公 0.31903719902038574  
黄蓉笑 0.31334972381591797  
周伯通 0.30642810463905334  
海滩 0.2639518082141876  
欧阳锋 0.2521178126335144  
葫芦 0.2515917718410492  
著 0.24856530129909515  
老顽童 0.24660998582839966  
悄声 0.2456432431936264

人物名字: 杨康

活 0.2687632441520691  
杨康心 0.23546013236045837  
自身 0.22278009355068207  
遗物 0.22265323996543884  
右肩 0.22264069318771362  
薄 0.21840925514698029  
棒 0.20497046411037445  
身世 0.20107805728912354  
酒 0.1985650360584259  
忙 0.1972237080335617

武功：降龙十八掌

剑法 0.26598992943763733

满天 0.2613832950592041

急 0.24910227954387665

凭 0.241050124168396

终身 0.23857808113098145

掌 0.23788118362426758

大事 0.2367507666349411

五行 0.23480457067489624

解释 0.2282589077949524

梁子翁 0.2256108820438385

武功：九阴真经

葫芦 0.25555676221847534

空明拳 0.25142940878868103

真经 0.24639078974723816

洪七公 0.24136364459991455

耳边 0.24012236297130585

周伯通 0.2341202050447464

周大哥 0.2237783819437027

所 0.22186201810836792

参仙 0.21110662817955017

以 0.2088114321231842

武功：蛤蟆功

百倍 0.22087377309799194

空明拳 0.21466831862926483

三分 0.2088710218667984

钹 0.20452967286109924

反正 0.20399591326713562

两条 0.20126035809516907

额头 0.1972479224205017

彷徨 0.19424089789390564

啸声 0.19227510690689087

臂力 0.18992644548416138

特征：丐帮

污衣 0.2514388859272003

招呼 0.25033724308013916

大小 0.2503257393836975

二丐 0.24461627006530762

攻击 0.23606447875499725

皇帝 0.23131795227527618

黄二 0.22913965582847595

死伤 0.20991884171962738

继承 0.20527036488056183

大事 0.2046264111995697

特征：桃花岛

想想 0.281222403049469  
桃花 0.2547352612018585  
互 0.24399301409721375  
死活 0.24197877943515778  
忽然 0.24069947004318237  
研习 0.2364480197429657  
老毒物 0.23509173095226288  
老化 0.23410576581954956  
陈玄风 0.22755014896392822  
牵 0.22553607821464539

特征: 蒙古

军官 0.2392040193080902  
土地 0.232521653175354  
蒲儿帖 0.22722633183002472  
昂然 0.22293338179588318  
急 0.22075749933719635  
藏身 0.21864937245845795  
知些 0.2177640199661255  
无 0.19529566168785095  
番话 0.19274704158306122  
王重阳 0.191800460219383