1. 实验内容与完成程度

自动词性标注就是用计算机来自动地给文本中的词标注词类。任务是根据一个词在某个特定句子中的上下文，为这个词标注正确的词性，要解决的主要问题是词性兼类歧义和未登录词词性的确定问题。词性标注的正确率直接影响到文本的后续工作，因为词义消歧和句法分析都以经过词性标注的句子为基础。

本实验中我选择了维特比算法实现中文分词。

本程序较好的完成了维特比算法，但是对未登陆词的处理不是很好。

2. 实验原理

2.1 HMM 模型

从概率角度上看，词性标注问题可描述为：给定一个含有词语序列 w1，w2…，wm 的输入句子 W，确定最有可能的词性标记序列 T=t1，t2…，tm，使得条件概率 P（T|W）最大。

根据 Bayes 公式，P（T|W）= P（T）P（W |T）/ P（W），由于词性标注中，W 是给定的，P（W）不依赖于 T，所以 P（T|W）≈P（T）P（W |T），其中 P（T）= P（t1|t0）P（t2|t1,t0）…P（ti|ti-1,ti-2,…），根据一阶 HMM 假设可得 P（T）≈ P（t1|t0）P（t2|t1）…P（ti|ti-1），P（W |T）是已知词性标记串产生词串的条件概率，P（W |T）= P（w1|t1）P（w2|t2,t1,w1）…P（wi|ti,ti-1,…t1,w i-1,…w1），根据一阶 HMM 假设可得 P（W |T）≈ P（w1|t1）P（w2|t2）…P（wi|ti）。

2.2 维特比算法

维特比算法是一种动态规划算法用于寻找最有可能产生观测事件序列的-维特比路径-隐含状态序列，特别是在马尔可夫信息源上下文和隐马尔可夫模型中。维特比算法的基础可以概括成下面三点：

1.如果概率最大的路径 p(或者说最短路径)经过某个点，比如途中的 X22，那么这条路径上的起始点 S 到 X22 的这段子路径 Q，一定是 S 到 X22 之间的最短路径。否则，用 S 到 X22 的最短路径 R 替代 Q，便构成一条比 P 更短的路径，这显然是矛盾的。证明了满足最优性原理。

2.从 S 到 E 的路径必定经过第 i 个时刻的某个状态，假定第 i 个时刻有 k 个状态，那么如果记录了从 S 到第 i 个状态的所有 k 个节点的最短路径，最终的最短路径必经过其中一条，这样，在任意时刻，只要考虑非常有限的最短路即可。

3. 结合以上两点，假定当我们从状态 i 进入状态 i+1 时，从 S 到状态 i 上各个节的最短路径已经找到，并且记录在这些节点上，那么在计算从起点 S 到第 i+1 状态的某个节点 Xi+1 的最短路径时，只要考虑从 S 到前一个状态 i 所有的 k 个节点的最短路径，以及从这个节点到 Xi+1，j 的距离即可。

2.3 维特比算法处理词性标注

中文中，每个词会有多种词性（比如"希望"即是名字又是动词），给出一个句子后，我们需要给这个句子的每个词确定一个唯一的词性，实际上也就是在若干词性组合中选择一个合适的组合。动词、名词等词类的搭配是具有规律性的，比如动词+名词的形式是大量存在的，当我们看到句子"存在希望"，如果确定了"存在"是动词，那么由于动名词组合的概率较大，我们就会认定"希望"是名词。viterbi 算法的原理就是基于此。我们需要计算所有的名词+动词，名词+名词，动词+形容词。。。。。等各种种词性搭配的出现概率，然后从中选出最大概率的组合。设一句

话有三个词组成分，分别为 A、B、C，则该句的概率值可以使用如下公式表示：

P(A 为 sxA1)*P(sxA1 位于句首)*P(B 为 sxB1)*P(sxA1->sxB1)*P(C 为 sxC1)*P(sxB1->sxC1)

A、B、C 有多个属性，对每个属性都用上式计算，去概率最大值为该句子的次序标注。

3. 训练语料库与开放环境

实验选用人民日报 98 语料库；
编程语言为 C++;
编程环境为 vs2017(windows)
评测数据 自己拟订

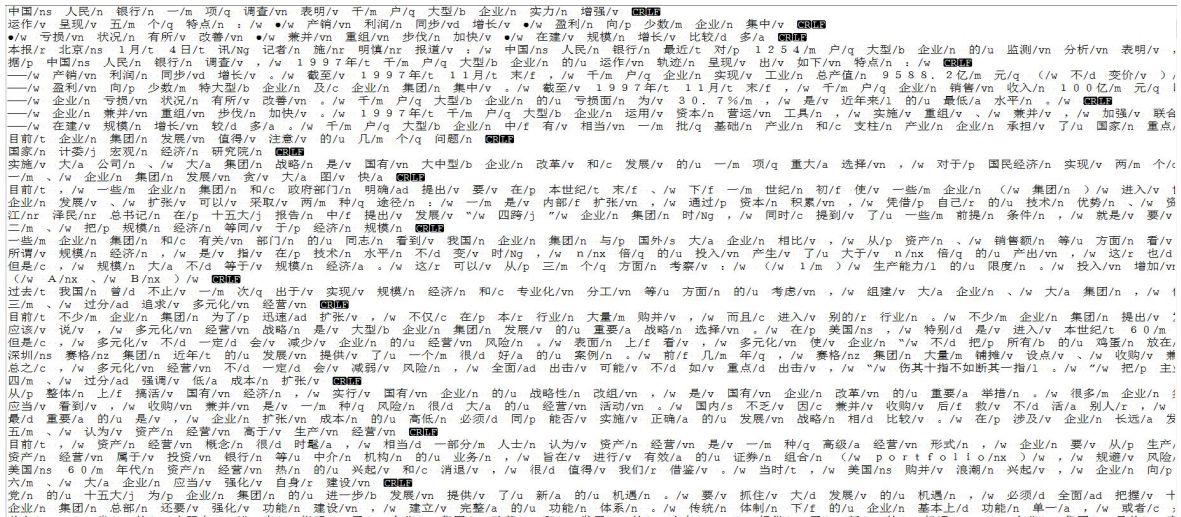
注：

读取训练文件的语句 fin.open("F://train_half.txt");
读取测试文件的语句 fin1.open("F://test_file.txt");

4. 实验设计与实现

4.1 语料库的准备

本实验采用人民日报 98 年语料库



4.2 读取并记录数据

对语料库的内容进行统计。需要得到以下数据。

（1）所有可能的词性。

（2）所有出现的词语。

（3）每个词语以不同词性出现的次数。

（4）记录句首词为不同词性的次数。

（5）记录不同词性间转换的次数。（如遇到："看电影"这个句子，则有[动词][名词]的值加一。）

4.3 获取概率结果

（1）计算每类词性作为句首出现的比例（比如:动词为句首，占所有不同词性为句首中的比例），记录到 double head_sx[sx_num];。

（2）计算前词固定为词性[n]时，后词为词性[x]占总情况的比例（如：前词固定为[动词]时，后词[名词]出现的次数占所有[x][动词]的比例），记录到 double bet[sx_num][sx_num];

（3）计算每一个词作为不同类词性出现的次数，占所有该类词出现总数的比例(如："震惊"作为动词出现的次数占所有名词的比例），记录到 double sx[sx_num];

4.4 标注

由以上得到了 head_sx[],bet[][],sx[]数组。先将测试的一条语句全部读入，求得其词组数目 len_test。然后使用函数 head_handle()，处理第一个词组 A。对 A 的所有属性 sx，计算

pro = P(sx)*P(sx 在句首)，调用函数 body_handle(int last,double probability, int level)。解释一下该函数的几个参数的含义：last->上一个词组的属性，probability->到上一个词组的概率积，level->目前的深度（head_handle()中 level = 0）。body_handle()处理词组 B，对 B 的所有属性 sx，计算 new_pro =pro* P(sx)*bet[A][B]，递归调用函数 body_handle(int last,double probability, int level)。在该过程中，记录每一个 level 所取得的属性值。

当 level>=len_test，即该句子处理完毕。比较此时的概率与最大概率的值，若此时的概率大于最大概率，则记录此时每一个 level 对应的属性值。

最后 head_handle()返回，即得到了当前语句的最优词性标注。

继续处理下一句，直至文本结束。

4.5 部分代码展示

读取训练文本函数 read_head()

```
bool vtb1::read_head()
{
    //c = ' ';
    while (c == ' ' || c == '\n' || c == '\r')
        c = fin.get();

    if (fin.eof())
        return false;

    char temp_value[50];
    int len_temp_value = 0;
    while (c != '/')                    //读取词组
    {
        if (c != ' ' && c != '\n' && c != '\r')
            temp_value[len_temp_value++] = c;

        c = fin.get();
    }
    temp_value[len_temp_value] = '\0';

    cout << temp_value << " ";

    /*********
    *处理head*
    *********/
    count_head++;       //首词组总数加1
    int pos = word_pos(temp_value);     //if(pos == len_w)->新词组，将改词添加后还需要对w[len_w]做初始化

                                                /*新词组处理*/
    if (pos == len_w)
    {
        strcpy_s(w[len_w].value, temp_value);
        w[len_w].total_value = 1;         //出现总数为1
        for (int i = 0; i < sx_num; i++)
        {
            w[len_w].count_sx[i] = 0;
            w[len_w].sx[i] = 0;
        }
        len_w++;
    }

    /*已有词处理*/
    else
```

读取训练文本函数 read_czsx()

```
bool vtb1::read_czsx()
{
    int sentence_end = 0;
    while (c == ' ' || c == '\r')
        c = fin.get();
    if (fin.eof())
        return false;
    char temp_value[50];
    int len_temp_value = 0;
    while (c != '/')
    {
        if (c == '\n')
            return false;

        temp_value[len_temp_value++] = c;
        c = fin.get();
    }
    temp_value[len_temp_value] = '\0';
    cout << temp_value << " ";

    /*********
    *句子终止处理*
    *********/
    if (strcmp(temp_value, "。") == 0 || strcmp(temp_value, "？") == 0 || strcmp(temp_value, "！") == 0 || strcmp(temp_value, "；") == 0 || strcmp(temp_value, "：") == 0)
        sentence_end = 1;

    /*********
    *处理w*
    *********/
    //count_head++;      //首词组总数加1
    int pos = word_pos(temp_value);     //if(pos == len_w)->新词组，将改词添加后还需要对w[len_w]做初始化

                                            /*新词组处理*/
    if (pos == len_w)
    {

        strcpy_s(w[len_w].value, temp_value);
        w[pos].total_value = 1;         //出现总数为1
        for (int i = 0; i < sx_num; i++)
        {
            w[pos].count_sx[i] = 0;
            w[pos].sx[i] = 0;
        }
```

处理统计数据，获取相应概率

```
void vtb1::train()
{
    int end_flag = read_head();
    //cout << "over0";
    while (end_flag)
    {
        read_rest();
        end_flag = read_head();
    }
    //cout << "over1";
    /*训练结果汇总*/

    //head_sx[]
    for (int i = 0; i < sx_num; i++)
    {
        head_sx[i] = count_head_sx[i] / count_head;
    }

    //w[]
    for (int i = 0; i < len_w; i++)
    {
        for (int j = 0; j < sx_num; j++)
        {
            w[i].sx[j] = w[i].count_sx[j] / w[i].total_value;
        }
    }

    //bet[][]
    for (int i = 0; i < sx_num; i++)
    {
        for (int j = 0; j < sx_num; j++)
        {
            bet[i][j] = count_bet[i][j] / total_bet;
        }
    }
```

head_handle()

```
void vtb1::head_handle()
{
    int pos = word_pos(test[0]);                //句首词组在w[]中的标号
    int level = 0;            //当前属性数组长度(层数)记为0
    max_pro = 0;          //最大属性置0

    int i;
    int exist = 0;
    int exist1 = 0;
    for (i = 0; i < sx_num; i++)
    {
        if (w[pos].sx[i] != 0 && head_sx[i] != 0)
        {
            exist = 1;
            pre_sx[level] = i;
            double pro = w[pos].sx[i] * head_sx[i];
            body_handle(i,pro,level+1);
        }
    }
    if (exist == 0)           //不存在两个都满足
    {
        for (i = 0; i < sx_num; i++)
        {
            if (w[pos].sx[i] != 0)
            {
                exist1 = 1;
                pre_sx[level] = i;
                //cout << "i = " << i << endl;
                double pro = w[pos].sx[i];
                body_handle(i, pro,level+1);
            }
        }
        if (exist1 == 0)     //未登陆词
        {
            pre_sx[level] = 16;                 //名词
            double pro = 1.0;
            body_handle(16, pro, level + 1);
        }
    }
}
```

body_handle()

```
void vtb1::body_handle(int last,double pro, int level)
{
    //level++;

    if (level >= len_test)
    {

        //比较当前的pro与最大pro，判断是否需要记录此时的属性数组
        if (pro > max_pro)
        {

            for (len_best_sx = 0; len_best_sx < level; len_best_sx++)
            {
                best_sx[len_best_sx] = pre_sx[len_best_sx];
            }
        }
        max_pro = pro;
        //level--;
        return;
    }
    int pos = word_pos(test[level]);

    //cout << "go into 1.5" << endl;

    int i;
    int exist = 0;                          //不存在满足if (w[pos].sx[i] != 0 && bet[last][i] != 0)的i
    int exist1 = 0;
    for (i = 0; i < sx_num; i++)
    {
        if (w[pos].sx[i] != 0 && bet[last][i] != 0)
        {
            //cout << "go into 1.75" << endl;
            exist = 1;
            //last = i;
            pre_sx[level] = i;
            double new_pro = pro * w[pos].sx[i] * bet[last][i];
            //cout << "go into 2" << endl;
            body_handle(i,new_pro,level+1);
            //cout << "go into 3" << endl;
            //level--;
        }
    }
```

输出标注结果

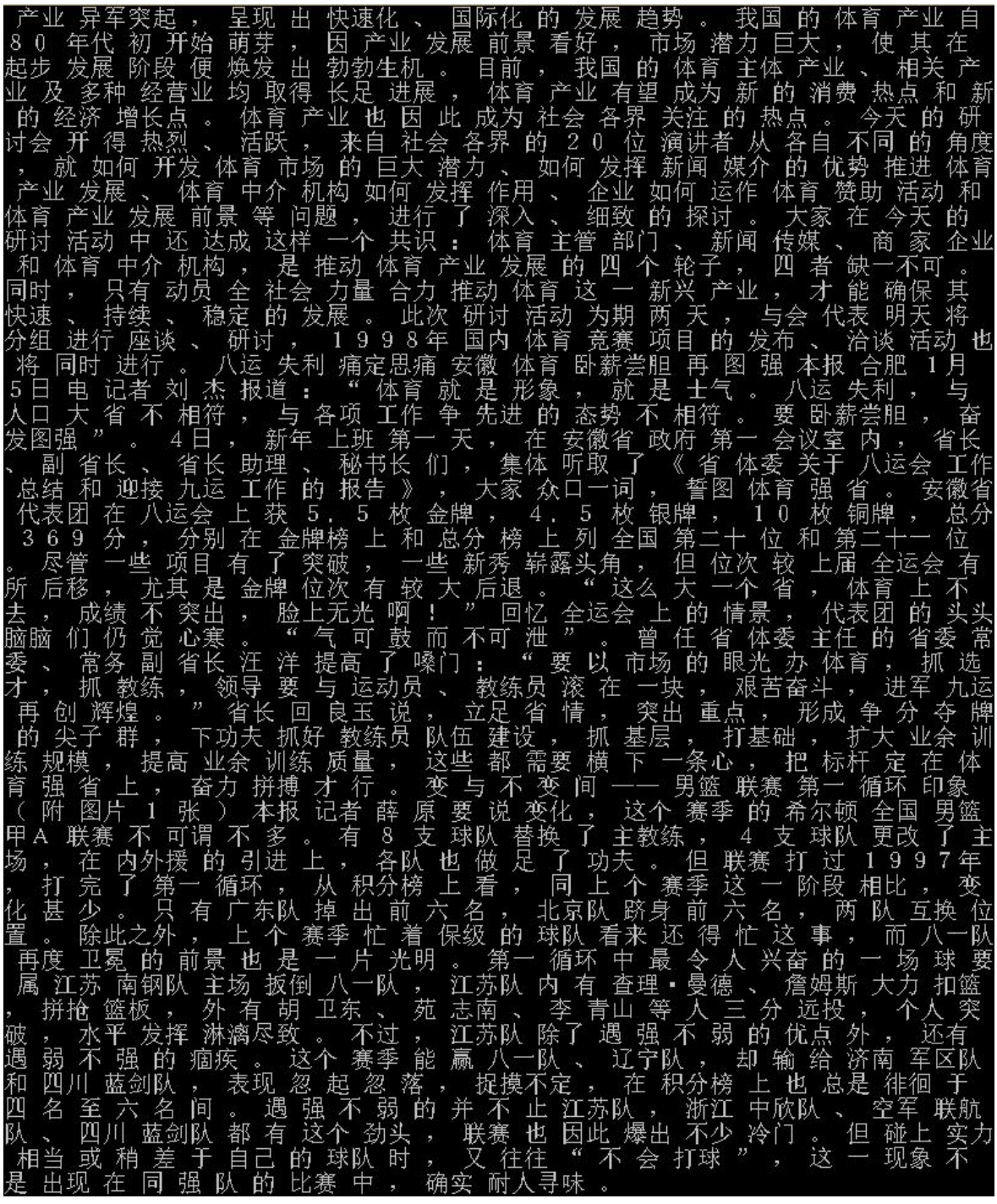
```
void vtb1::analyze()
{
    cout << endl << endl << endl << "------------------测试结果------------------" << endl << endl;
    int flag = read_a_sentence();

    //cout <<endl<< "analyze start" << endl;
    //cout << "flag = " << flag;
    while (flag == true)
    {

        head_handle();
        int i;
        for (i = 0; i < len_test; i++)
        {
            fout << test[i] << "/"<<int_to_char(best_sx[i]) << " ";
            cout << test[i] << "/"<<int_to_char(best_sx[i]) << " ";
        }
        cout << endl;
        flag = read_a_sentence();
    }
}
```

5. 实验结果展示

5.1 词性分类

```
a; 形容词
Ag; 形容词语素
an; 名形词 //具有名词功能的形容词
b; 区别词
c; 连词
d; 副词
Dg; 副词语素
e; 叹词
Eg; 叹词语素
f; 方位词
g; 语素
h; 前接成分
i; 成语
j; 缩略语
k; 后接成分
l; 习用语
m;数词
n; 名词
Ng; 名词语素
ns; 名词-表处所
nt; 名词-表时间
nx; 名词-英文字母
nz; 名词-其他专有名词
o; 像声词
p; 介词
q; 量词
Qg;量词语素
r; 代词
Rg; 代词语素
s; 处所词
t; 时间词
Tg; 时间词语素
u; 助词
Ug; 助词语素
v; 动词
Vg; 动词语素
vd; 副动词
w; 标点
x; 字
y; 语气词
Yg; 语气词语素
z; 状态词
```

5.2 训练过程（读取数据并统计）



产业异军突起，呈现出快速化、国际化的发展趋势。我国的体育产业自80年代初开始萌芽，因产业发展前景看好，市场潜力巨大，使其在起步发展阶段便焕发出勃勃生机。目前，我国的体育主体产业、相关产业及多种经营业均取得长足进展，体育产业有望成为新的消费热点和新的经济增长点。体育产业也因此成为社会各界关注的热点。今天的研讨会开得热烈、活跃，来自社会各界的20位演讲者从各自不同的角度，就如何开发体育市场的巨大潜力、如何发挥新闻媒介的优势推进体育产业发展、体育中介机构如何发挥作用、企业如何运作体育赞助活动和体育产业发展前景等问题，进行了深入、细致的探讨。大家在今天的研讨活动中还达成这样一个共识：体育主管部门、新闻传媒、商家企业和体育中介机构，是推动体育产业发展的四个轮子，四者缺一不可。同时，只有动员全社会力量合力推动体育这一新兴产业，才能确保其快速、持续、稳定的发展。此次研讨活动为期两天，与会代表明天将分组进行座谈、研讨，1998年国内体育竞赛项目的发布、洽谈活动也将同时进行。八运失利痛定思痛安徽体育卧薪尝胆再图强本报合肥1月5日电记者刘杰报道：“体育就是形象，就是士气。八运失利，与人口大省不相符，与各项工作争先进的态势不相符。要卧薪尝胆，奋发图强”。4日，新年上班第一天，在安徽省政府第一会议室内，省长、副省长、省长助理、秘书长们，集体听取了《省体委关于八运会工作总结和迎接九运工作的报告》，大家众口一词，誓图体育强省。安徽省代表团在八运会上获5.5枚金牌，4.5枚银牌，10枚铜牌，总分369分，分别在金牌榜上和总分榜上列全国第二十位和第二十一位。尽管一些项目有了突破，一些新秀崭露头角，但位次较上届全运会有所后移，尤其是金牌位次有较大后退。“这么大一个省，体育上不去，成绩不突出，脸上无光啊！”回忆全运会上的情景，代表团的头头脑脑们仍觉心寒。“气可鼓而不可泄”。曾任省体委主任的省委常委、常务副省长汪洋提高了嗓门：“要以市场的眼光办体育，抓选才，抓教练，领导要与运动员、教练员滚在一块，艰苦奋斗，进军九运再创辉煌。”省长回良玉说，立足省情，突出重点，形成争分夺牌的尖子群，下功夫抓好教练员队伍建设，抓基层，打基础，扩大业余训练规模，提高业余训练质量，这些都需要横下一条心，把标杆定在体育强省上，奋力拼搏才行。变与不变间——男篮联赛第一循环印象（附图片1张）本报记者薛原要说变化，这个赛季的希尔顿全国男篮甲A联赛不可谓不多。有8支球队替换了主教练，4支球队更改了主场，在内外援的引进上，各队也做足了功夫。但联赛打过1997年，打完了第一循环，从积分榜上看，同上个赛季这一阶段相比，变化甚少。只有广东队掉出前六名，北京队跻身前六名，两队互换位置。除此之外，上个赛季忙着保级的球队看来还得忙这事，而八一队再度卫冕的前景也是一片光明。第一循环中最令人兴奋的一场球要属江苏南钢队主场扳倒八一队，江苏队内有查理·曼德、詹姆斯大力扣篮，拼抢篮板，外有胡卫东、苑志南、李青山等人三分远投，个人突破，水平发挥淋漓尽致。不过，江苏队除了遇强不弱的优点外，还有遇弱不强的痼疾。这个赛季能赢八一队、辽宁队，却输给济南军区队和四川蓝剑队，表现忽起忽落，捉摸不定，在积分榜上也总是徘徊于四名至六名间。遇强不弱的并不止江苏队，浙江中欣队、空军联航队、四川蓝剑队都有这个劲头，联赛也因此爆出不少冷门。但碰上实力相当或稍差于自己的球队时，又往往“不会打球”，这一现象不是出现在同强队的比赛中，确实耐人寻味。

5.3 测试结果展示

状态转移矩阵（#表示该状态不存在）

```
a->s 1.59406e-05
a->t 3.18811e-05
a->Tg 1.59406e-05
a->u 0.00787464
a->Ug 0
a->v 0.000812969
a->Vg 0
a->w 0.00589801
a->x 0
a->y 0.000207227
a->Yg 0
a->z 0
a->ad 0
a->vn 0.00216792
a->vd 0
a->nx 0
a->an 0.000127525
a->nz 0
a-># 0
a-># 0
a-># 0
a-># 0
a-># 0
Ag->a 1.59406e-05
Ag->Ag 0
Ag->b 0
Ag->c 1.59406e-05
Ag->d 0
Ag->Dg 0
Ag->e 0
Ag->Eg 0
Ag->f 1.59406e-05
Ag->g 0
Ag->h 0
Ag->i 0
Ag->j 0
Ag->k 0
Ag->l 0
Ag->m 0
Ag->n 7.97029e-05
Ag->Ng 3.18811e-05
Ag->nr 0
Ag->ns 0
```

测试文件

6. 遇到的问题以及解决方案

1、在处理语料库文件，要注意编码方式，避免中文信息容易出现的乱码情况。Windows环境下的换行是/n/r，因此在判断换行情况的时候条件应该是 if(　==/n/r)。在获取训练集的

时候要在每个句子的结尾加上/n。

2、if (w[pos].sx[i] != 0 && head_sx[i] != 0)，A 为句首词，但 A 的所有属性都不合放句首的情况。导致无法继续调用 body_handle()。if (w[pos].sx[i] != 0 && bet[last][i] != 0)，没有考虑不存在这种条件的情况，导致递归无法继续，最后标注结果全部为/a。

解决方法是增加了对这两种情况的处理：

```
if (exist == 0)             //
{
    //cout << "go into 3.75" << endl;
    for (i = 0; i < sx_num; i++)
    {
        if (w[pos].sx[i] != 0)
        {
            //last = i;
            //cout << "i ==" << i<<endl;
            exist1 = 1;
            pre_sx[level] = i;
            double new_pro = pro * w[pos].sx[i];
            //cout << "go into 4" << endl;
            body_handle(i, new_pro,level + 1);
            //cout << "go into 5" << endl;
            //level--;
        }
    }
```

3、未登陆词词性为 a（较少出现）导致后序分析出错。解决方法：

未登陆词的属性随机选取出现频率最高的三种属性之一。

7. 心得体会

通过本次词性标注实验，对模型训练有了初步的了解。也知道了怎么处理得到训练数据，体会到了中文信息处理的乐趣。