**Universidade de Aveiro**

**MPEI 2020/21**

**2º guião para avaliação**

**Joaquim Andrade nº93432**

**Francisco Silva nº93400**



1.a)Abaixo encontra-se o código. Fizemos também uso da função, fornecida pelo docente, crawl, a qual alteramos para não passar o último argumento para a palavra. A palavra criada foi “a”.

% Random walk on the Markov chain

% Inputs:

% H - state transition matrix

% first - initial state

% last - terminal or absorving state

function state = crawl(H, first, last)

% the sequence of states will be saved in the vector "state"

% initially, the vector contains only the initial state:

state = [first];

% keep moving from state to state until state "last" is reached:

while (1)

a=nextState(H, state(end));

if (a== last) %stops the atribution of the last state to the word

break;

end

state(end+1) = a;

end

end

m=[0 1/3 0 1/4 0; %r

1/2 0 1/2 1/4 0; %o

0 1/3 0 1/4 0; %m

1/2 0 1/2 0 0; %a

0 1/3 0 1/4 0]; %.

%cria uma matriz de transição

basedados= ['r','o','m','a',' '];%cria caracteres na mesma posição que na matriz

palavra= basedados(crawl(m,randi(4),5));%cria um caminho pela matriz e passa-o para uma palavra

1.b) Foram geradas aproximadamente 117867 palavras não repetidas, ou seja aproximadamente 9/10 das palavras são repetidas. Os resultados das cinco palavras que mais apareceram foram:

'o' 0.0832810000000000

'a' 0.0622420000000000

'ro' 0.0418120000000000

'mo' 0.0414750000000000

'ra' 0.0313570000000000

Fazemos também uso da função crawl acima aqui. O código utilizado foi o seguinte:

m=[0 1/3 0 1/4 0; %r

1/2 0 1/2 1/4 0; %o

0 1/3 0 1/4 0; %m

1/2 0 1/2 0 0; %a

0 1/3 0 1/4 0]; %.

%cria uma matriz de transição

a=cell(10e5, 1);%aloca espaço para 10e5 palavras em cell

for i=1: 10e5 %ciclo cria e aloca 10e5 palavras no cell criado anterior

a{i}=basedados(crawl(m,randi(4),5)) ;

end

pD= length(unique(a)); %número de palavras não repetidas

Mpu= unique(a);

[uc, ~, idc] = unique(a);

counts= accumarray(idc, ones(size(idc)));

M= cell(pD, 2);

for i=1:pD

M(i,1)=Mpu(i); %aloca as palavras não repetidas

M(i,2)= num2cell(counts(i)/10e5); %aloca a probabilidade de se repetirem no array original

end

f=cell(5,2);%5 palavras mais usadas e probabilidade

M=sortrows(M,2); % ordena as probabilidades

for i=1: 5

f(i,1)= M(pD-i+1,1); %guarda aas primeiras 5 palavras

f(i,2)= M(pD-i+1,2); %guarda as primeiras 5 probabilidades

end

1.c)Cálculos das probabilidades teóricas das 5 palavras mais geradas:

'o' 0,25\*1/3 = 0,083333333

'a' 0,25\*0,25 = 0,0625

'ro' 0,25\*0,5\*1/3 = 0,041666666

'mo' 0,25\*0,5\*1/3 = 0,041666666

'ra' 0,25\*0,5\*0,25 = 0,03125

Comparando estes valores com os obtidos na alínea anterior observamos que são muito idênticos e por isso podemos concluir que a simulação foi bem sucedida.

1.d)Ao código da pergunta 1.b) adicionámos o seguinte excerto de código que abre o ficheiro de palavras, interseta-o com as palavras geradas e com as probabilidades de cada palavra ser gerada. Essa probabilidade deu 0.3531.

fid=fopen('wordlist-preao-20201103.txt');

data=textscan(fid,'%s');

fclose(fid);

g= data{1}(1:end);%abrir e ler o ficheiro para uma célula

a=0;

h=intersect(g, Mpu);%interseta g com as palavras geradas

for i=1: length(Mpu)

if ismember(M(i,1), h)==1%se existir uma palavra em M

a=a+cell2mat(M(i,2)); %somamos a probabilidade dessa palavra

end

end

1.e)Esta função crawl não adiciona o ultimo estado à palavra e para quando n é alcançado.

function state = crawl2(H, first, last, n)%add n

state = [first];

d=1;

while (1)

a=nextState(H, state(end));

if (a== last || n==d)%stops the atribution of the last state to the word

break;

end

state(end+1) = a;

d=d+1;

end

end

1.f) Usando a função resolvida anteriormente, juntamente com o código também resolvido anteriormente, chegámos às seguintes conclusões:

2- Tal como na função 1.f), usámos o código já desenvolvido, e alterámos apenas a matriz de transição para a seguinte:

m=[0 0.3 0 0.3 0; %r

0.3 0 0.3 0.1 0; %o

0 0.2 0 0.2 0; %m

0.7 0 0.7 0 0; %a

0 1/2 0 0.4 0]; %.

As nossas conclusões foram as seguintes:

3-Retirámos as percentagens em relação às 10e5 palavras criadas e usámos apenas as palavras que usam as letras ‘a’, ‘o’, ‘m’, ‘r’ e que estão no dicionário e na lista de palavras geradas e repetidas (519051 palavras). Dessas:

começadas por a: 0.3546

começadas por m: 0.2598

começadas por o: 0.3128

começadas por r: 0.0728

Foi adicionado o seguinte código(depois de retirada a divisão por 10e5):

pa=0;

pm=0;

pr=0;

po=0;

for i=1: length(Mpu)

if ismember(M(i,1), h)==1 && M{i,1}(1)== 'a' %se existir uma palavra em M e a primeira letra for a

pa=pa+cell2mat(M(i,2)); %somamos o numero de palavras começadas por a

end

if ismember(M(i,1), h)==1 && M{i,1}(1)== 'm' %se existir uma palavra em M e a primeira letra for m

pm=pm+cell2mat(M(i,2)); %somamos o numero de palavras começadas por m

end

if ismember(M(i,1), h)==1 && M{i,1}(1)== 'o' %se existir uma palavra em M e a primeira letra for o

po=po+cell2mat(M(i,2)); %somamos o numero de palavras começadas por o

end

if ismember(M(i,1), h)==1 && M{i,1}(1)== 'r' %se existir uma palavra em M e a primeira letra for r

pr=pr+cell2mat(M(i,2)); %somamos o numero de palavras começadas por r

end

end

pt= pa+pm+po+pr; % obter o numero de palavras totais

pa= pa/pt;%probabilidades de começar em a e com as limitações

pm= pm/pt;

po= po/pt;

pr= pr/pt;

Usando estas probabilidades para criar o estado inicial da matriz de transição obtivemos um aumento de aproximadamente 10% da probabilidade de uma palavra gerada ser uma palavra em português sem qualquer limite, por sua vez os outros resultados para diferentes n foram os seguintes:

n=∞ :0.6149

n=8 :0.6168

n=6 :0.6380

n=4 :0,7310

O código usado para alterar a letra inicial foi o seguinte:

n=4;

m=[0 0.3 0 0.3 0; %r

0.3 0 0.3 0.1 0; %o

0 0.2 0 0.2 0; %m

0.7 0 0.7 0 0; %a

0 1/2 0 0.4 0]; %.

%cria uma matriz de transição

basedados= ['r','o','m','a',' '];%cria caracteres na mesma posição que na matriz

pa= 0.3546;

pm= 0.2598;

po=0.3128 ;

pr=0.0728;

a=cell(10e5, 1);%aloca espaço para 10e5 palavras em cell

for i=1: 10e5 %ciclo cria e aloca 10e5 palavras no cell criado anterior

prand=rand();

if(pa>prand)

t=4;

elseif (pm+pa)>prand

t=3;

elseif (pm+pa+po)>prand

t=2;

else

t=1;

end

a{i}=basedados(crawl2(m,t,5,n)) ;

end

4-Alterando a matriz da pergunta anterior para a seguinte:

m=[0 0.3 0 0.3 0; %r

0.3 0 0.3 0.1 0; %o

0.1 0.2 0 0.2 0; %m

0.6 0 0.7 0 0; %a

0 0.5 0 0.4 0]; %.

Obtivemos os seguintes resultados da probabilidade de criar palavras em português:

n=∞ :0.6039

n=8 :0.6051

n=6 :0.6230

n=4 :0.7159

Em relação aos exercícios anteriores, podemos observar uma redução de aproximadamente 2% de eficiência em todos os tamanhos limitantes das palavras. Estes resultados podem ser atribuídos à grande diferença entre as palavras que contêm “rm” em relação à combinação do qual o retiramos, mas também pode ser devido a ser uma transição entre dois estados não terminais, o que faz com que a palavra, “assumindo que começa em r”, tenha no mínimo 3 letras. “rm” não é normalmente um início típico de uma palavra portuguesa, o que dá que uma palavra começada com “rm” tenha sempre no mínimo quatro letras o que pode ser atribuído à diminuição da eficiência.

5-