|  |  |  |
| --- | --- | --- |
| PMO-TALN | TP : Désambiguisation lexicale | |
| Bois Cédric – M1 WIC – 2015 2016 | |  |

Sommaire

[Exercice 1 2](#_Toc438813650)

[Exercice 2 2](#_Toc438813658)

[Exercice 3 3](#_Toc438813660)

[Exercice 4 3](#_Toc438813662)

[Exercice 5 3](#_Toc438813663)

[Exercce 6 3](#_Toc438813664)

[Exercice 7 4](#_Toc438813666)

[Exercice 8 à 10 4](#_Toc438813667)

[Exercice 11 5](#_Toc438813668)

[Exercice 12 5](#_Toc438813669)

[Exercice 13 5](#_Toc438813670)

[Exercice 14 5](#_Toc438813671)

[Exercice 16 5](#_Toc438813672)

[Exercice 17 6](#_Toc438813673)

[Exercice 18 6](#_Toc438813674)

[Exercice 19 6](#_Toc438813675)

[Exercice 20 6](#_Toc438813676)

[Exercice 21 7](#_Toc438813677)

[Exercice 22 8](#_Toc438813678)

[Exercice 23 8](#_Toc438813679)

[Exercice 24 8](#_Toc438813680)

[Exercice 25 9](#_Toc438813681)

[Exercice 27 10](#_Toc438813682)

## Exercice 1

<instance id="d005.s011.t001" lemma="drawing" pos="n">drawings</instance>  
 <instance id="d005.s011.t002" lemma="improve" pos="v">improved</instance>  
 <instance id="d005.s011.t003" lemma="careless" pos="a">careless</instance>  
 <instance id="d005.s011.t004" lemma="habit" pos="n">habits</instance>  
 <instance id="d005.s011.t005" lemma="continue" pos="v">continued</instance>

### <instance id="d005.s011.t001" lemma="drawing" pos="n">drawings</instance>

* Position : document 5, 11ème phrase, 1er terme
* lemme : drawing
* terme : drawings

### <instance id="d005.s011.t002" lemma="improve" pos="v">improved</instance>

* Position : document 5, 11ème phrase, 2eme terme
* lemme : improve
* terme : improved

### <instance id="d005.s011.t003" lemma="careless" pos="a">careless</instance>

* Position : document 5, 11ème phrase, 3eme terme
* lemme : careless
* terme : careless

### <instance id="d005.s011.t004" lemma="habit" pos="n">habits</instance>

* Position : document 5, 11ème phrase, 4eme terme
* lemme : habit
* terme : habits

### <instance id="d005.s011.t005" lemma="continue" pos="v">continued</instance>

* Position : document 5, 11ème phrase, 5eme terme
* lemme : continue
* terme : continued

### Quels sont les quatre parties du discours des mots annotés ?

* N = nom
* A = adjectif
* R = adverbe
* V = verbe

### Pourquoi certains mots ne sont pas annotés ?

Parce qu’ils ne présentent pas d’ambiguïté.

## Exercice 2

### Pourquoi y-a-t-il plusieurs synsets possible dans certain cas ?

Il y a plusieurs synsets car, le dictionnaire peut renvoyer plusieurs sens à chaque mot dans cette phrase.

## Exercice 3

### Pour chacun des exemples, donnez la précision, le rappel et le F-score.

* Fichier 1

Total: A = 1.000 (2269 / 2269), P = 0.74394 (1688 / 2269), R = 0.74394 (1688 / 2269), F1 = 0.74394

* Fichier 2

Total: A = 1.000 (2269 / 2269), P = 0.73601 (1670 / 2269), R = 0.73601 (1670 / 2269), F1 = 0.73601

* Fichier 3

Total: A = 1.000 (2269 / 2269), P = 0.75187 (1706 / 2269), R = 0.75187 (1706 / 2269), F1 = 0.75187

* Fichier 4

Total: A = 1.000 (2269 / 2269), P = 0.62054 (1408 / 2269), R = 0.62054 (1408 / 2269), F1 = 0.62054

* Fichier 5

Total: A = 0.859 (1948 / 2269), P = 0.76027 (1481 / 1948), R = 0.65271 (1481 / 2269), F1 = 0.70240

A = attemped : Nombre de mots non annotés / nombre de mots.  
P = précision : Nombre de mots bien annotés / nombre de mots annotés   
R = rappel : Nombre de mots bien annotés / nombre de mots.  
F1 = F. Mesure

## Exercice 4

-w affiche les mots mal annotés  
-m affiche la référence des mots manquants  
-d affiche l'analyse de tous les doccuments du dossier  
-x exclude one-cluster words  
-a affiche toutes les réponses de chaque terme

## Exercce 6

### Quelle sont les deux premières définitions ?

[S:](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=cat&i=0&h=0000000000#c) (n) cat, [true cat](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=true+cat) (feline mammal usually having thick soft fur and no ability to roar: domestic cats; wildcats)

[S:](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=cat&i=1&h=0000000000#c) (n) [guy](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=guy), cat, [hombre](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=hombre), [bozo](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=bozo), [sod](http://wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o7=&o5=&o9=&o6=&o3=&o4=&s=sod) (an informal term for a youth or man) "a nice guy"; "the guy's only doing it for some doll"; "the poor sod couldn't even buy a drink"

On affecte un entier à chacun des mots

Feline = 1  
mammal = 2  
usually = 3  
having = 4  
thick = 5  
soft = 6  
fur = 7  
and = 8  
no = 9  
ability = 10  
to = 11  
roar =12  
domestic = 13  
cats =14  
wildcats = 15  
an =16  
informal =17  
term = 18  
for =19  
a =20  
youth 21   
or =22  
man =23

Ensuite, on regroupe les ensembles d’entiers qui codent les mots, par définitions

D1->{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}  
D2 ->{16, 17, 18, 19, 20, 21, 22, 23}

Ces ensembles sont triés par ordre croissant

## Exercice 7

<ids>cat%1:05:00:: </ids>  
<def>8 22 112 428 528 705 1216 1228 1538 2450 2518 7518 9842 31822 45730 </def>

<ids>cat%1:18:00:: </ids>  
<def>0 0 72 134 409 482 10357 13768 16164 </def>

On peut voir que la seconde définition présente deux fois le même mot (0). Ce mot doit correspondre à « a » et « an » que l’on a considéré comme différents précédemment.

## Exercice 8 à 10

Voir fichier java.

## Exercice 11

Function getSimilarity1

Similarité entre pine et cone sur le dictionnaire Dict-Lesk.xml : 0.8  
Similarité entre pine et cone sur le dictionnaire Dict-Lesk-etendu.xml : 11.533334  
Similarité entre pine et cone sur le dictionnaire dict\_all\_stopwords\_stemming\_semcor\_dso\_wordnetglosstag\_150.xml : 4.4666667

## Exercice 12

media,radio,7.42

tiger,tiger,10.00

stock,jaguar,0.92

football,soccer,9.03

brother,monk,6.27

Le fichier contient une liste de couples de mots, associée à un nombre compris entre 0 et 10. Ce nombre représente la similarité entre ces deux mots : on peut voir que, entre deux fois le même mot, la similarité est de 10 (tiger, tiger), alors que entre deux mots qui n’ont rien à voir (stock, jaguar), la similarité est très basse.

## Exercice 13

La corrélation entre les séries {1,4,8,12,2,5,4} et {4,2,1,6,6,10,4} est de 0.004243272981903036.

## Exercice 14

Utilisation de OpenCSV  
Corrélation entre humain et lesk :0.20313210443967666

Corrélation entre humain et lesk étendu :0.21253348768870164

Corrélation entre humain et lesk corpus :0.21388841987844234

## Exercice 16

36 combinaisons pour la chaine "go" dans le 1e dictionnaire

96 combinaisons pour la chaine "mouse pilot computer" dans le 1e dictionnaire

13440 combinaisons pour la chaine "dog eat bone every day" dans le 1e dictionnaire

470400 combinaisons pour la chaine "doctor be hospital last day night" dans le 1e dictionnaire

7832809874592301056 combinaisons pour la chaine "pictures paint be flat round figure be very often foot do look be stand ground all point downward be hanging air" dans le 1e dictionnaire

36 combinaisons pour la chaine "go" dans le 2e dictionnaire

96 combinaisons pour la chaine "mouse pilot computer" dans le 2e dictionnaire

13440 combinaisons pour la chaine "dog eat bone every day" dans le 2e dictionnaire

470400 combinaisons pour la chaine "doctor be hospital last day night" dans le 2e dictionnaire

7832809874592301056 combinaisons pour la chaine "pictures paint be flat round figure be very often foot do look be stand ground all point downward be hanging air" dans le 2e dictionnaire

36 combinaisons pour la chaine "go" dans le 3e dictionnaire

96 combinaisons pour la chaine "mouse pilot computer" dans le 3e dictionnaire

13440 combinaisons pour la chaine "dog eat bone every day" dans le 3e dictionnaire

470400 combinaisons pour la chaine "doctor be hospital last day night" dans le 3e dictionnaire

7832809874592301056 combinaisons pour la chaine "pictures paint be flat round figure be very often foot do look be stand ground all point downward be hanging air" dans le 3e dictionnaire

## Exercice 17

La première et la dernière phrase ne semblent pas désambiguisable. En effet, la première ne présente pas de contexte, et la dernière présente beaucoup trop de combinaisons pour pouvoir toutes les tester.

## Exercice 18

Voir algoExhaustif et calculCombinaisons.

## Exercice 19

mouse pilot computer :

* mouse : 4e sens
* pilot : 8e sens
* computer : 1er sens

dog eat bone every day :

* dog : 1er sens
* eat : 3e sens
* bone : 1er sens
* every : 1er sens
* day : 1er sens

doctor be hospital last day night :

* doctor : 1er sens
* be : 2e sens
* hospital : 2e sens
* last : 2e sens
* day : 1er sens
* night : 1er sens

## Exercice 20

* computer%1:06:00::
* pilot%2:38:01::
* mouse%1:06:00::
* dog%1:05:00::
* eat%2:30:00::
* bone%5:00:01:bony:00
* every%5:00:00:all:00
* day%1:28:05::
* night%1:28:02::
* day%1:28:05::
* last%3:00:00::
* hospital%1:14:00::
* be%2:42:02::
* doctor%1:04:00::

## Exercice 21

//Permet de charger le corpus situé à l’adresse path  
public void Corpus.loadCorpus(Dictionary d, String path)

//renvoie les texts d’un corpus  
public ArrayList<Text> Corpus .getTexts()

//Création d’une configuration en choisissant, pour chaque mot du texte, un sens de manière aléatoire si randomGeneration = true, sinon, le premier sens est choisi.  
public ProblemConfiguration(int n, boolean randomGeneration, Text s)

//calcul la somme des similarités des sens choisis pour chaque combinaison de deux mots du texte.  
public void ProblemConfiguration.computeScore(Text text, Dictionary dictionary)

//renvoie le score d’une configuration  
public double ProblemConfiguration.getScore()

//Fonction qui choisie un mot du texte de manière aléatoire et qui va changer la sélection de son sens de manière aléatoire également.  
public void ProblemConfiguration.makeChange(Text t)

//Fonction qui écrit dans un fichier, la liste du sens choisi pour chacun des mots  
public void writeResult(Text text, PrintStream ps)

## Exercice 22

1ere exécution :

Texte d001

Score initial= 262652.0

Score aprÃ¨s changement=262652.0

Texte d002

Score initial= 292632.0

Score aprÃ¨s changement=292632.0

Texte d003

Score initial= 429040.0

Score aprÃ¨s changement=429040.0

Texte d004

Score initial= 893316.0

Score aprÃ¨s changement=893316.0

Texte d005

Score initial= 240449.0

Score aprÃ¨s changement=240136.0

2de exécution :

Texte d001

Score initial= 255156.0

Score aprÃ¨s changement=255156.0

Texte d002

Score initial= 306129.0

Score aprÃ¨s changement=306129.0

Texte d003

Score initial= 415784.0

Score aprÃ¨s changement=415784.0

Texte d004

Score initial= 930457.0

Score aprÃ¨s changement=930457.0

Texte d005

Score initial= 269827.0

Score aprÃ¨s changement=269951.0

On peut voir que après changement, le résultat n’est pas forcément meilleurs et qu’il peut même être pire que le score initial. Il en est de même sur différentes exécutions, certains résultats sont meilleurs et d’autres pire, d’une exécution sur l’autre. On peut donc en déduire que ces résultats ne sont pas fiables.

## Exercice 23

Pour évaluer le score, la fonction à besoin d’un dictionnaire qui est alors rajouté parmi les paramètres de la fonction.

## Exercice 24

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Texte d001 | | Texte d002 | | Texte d003 | | Texte d004 | | Texte d005 | |
| Itérations | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final |
| 5 | 266287 | 266335 | 314682 | 315281 | 432830 | 436909 | 883292 | 883292 | 242267 | 245561 |
| 10 | 258250 | 259383 | 302162 | 302965 | 420446 | 422070 | 907607 | 912454 | 247927 | 252170 |
| 20 | 253196 | 256993 | 291954 | 298473 | 414025 | 419891 | 915602 | 920253 | 247378 | 248898 |
| 50 | 275811 | 283841 | 300463 | 310184 | 407176 | 422172 | 901567 | 924015 | 253093 | 261255 |
| 100 | 262611 | 275830 | 300260 | 315389 | 403973 | 418933 | 906706 | 943418 | 240188 | 265417 |
| 1000 | 280991 | 385859 | 301675 | 436187 | 424298 | 581587 | 955022 | 1242833 | 255268 | 407345 |
| 10000 | 261630 | 454481 | 293848 | 553144 | 416702 | 770024 | 939134 | 1736401 | 243117 | 571580 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Différence  entre le   score  initiale  et le  score  finale  Itérations | Texte d001 | Texte d002 | Texte d003 | Texte d004 | Texte d005 |
| 5 | 48 | 599 | 4079 | 0 | 3294 |
| 10 | 1133 | 803 | 1624 | 4847 | 4243 |
| 20 | 3797 | 6519 | 5866 | 4651 | 1520 |
| 50 | 8030 | 9721 | 14996 | 22448 | 8162 |
| 100 | 13219 | 15129 | 14960 | 36712 | 25229 |
| 1000 | 104868 | 134512 | 157289 | 287811 | 152077 |
| 10000 | 192851 | 259296 | 353322 | 797267 | 328463 |

On peut voir que le score s’améliore dans la majeure partie des cas. En effet, le principe de cet algorithme est qu’on ne retient qu’un score qui est meilleur que le précédent. On peut voir que plus le nombre d’itérations est important, plus le score s’améliore. Néanmoins, cet algorithme reste tout de même couteux, puisqu’il a fallut plusieurs heures pour mener à terme les 10000 itérations sur les cinq textes. De plus il donne une configuration avec un score, mais nous n’avons aucune idée sur la relation entre ce score et le score optimal.

## Exercice 25

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Texte d001 | | Texte d002 | | Texte d003 | | Texte d004 | | Texte d005 | |
|  | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final | Score Initial | Score final |
| 1ere exécution | 257627 | 276614 | 298533 | 317580 | 394924 | 413478 | 906664 | 935916 | 252259 | 274723 |
| 2eme exécution | 265726 | 287522 | 287090 | 310324 | 416927 | 437476 | 937977 | 965096 | 256350 | 273545 |
| 3eme exécution | 247966 | 267243 | 289434 | 313241 | 409815 | 430584 | 932797 | 966432 | 224603 | 251199 |

A chaque exécution, des résultats différents sont obtenus, ce qui est logique puisque l’algorithme utilise de l’aléatoire, mais les scores restent tout de même relativement proches.

## Exercice 27

Les essaies sont réalisés sur le texte d001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | M | S | Score final | Temps d’exécution (s) |
| 1000 | 100 | 4 | 451840 | 2778 |
| 500 | 100 | 4 | 449058 | 1632 |
| 300 | 100 | 4 | 444721 | 1014 |
| 1000 | 50 | 4 | 445438 | 1921 |
| 100 | 50 | 4 | 422731 | 428 |
| 300 | 299 | 4 | 311334 | 12998 |
| 300 | 100 | 20 | 447505 | 1413 |
| 300 | 100 | 10 | 449359 | 1132 |
| 300 | 100 | 2 | 447110 | 954 |

On peut s’apercevoir que le paramètre s n’a que très peu d’influence, en effet, qu’il soit égale à 2, 4, 10 ou 20, avec n = 300 et m = 100, on obtient un résultat toujours autour de 447000.

De plus, on peut voir que si n et m sont trop proches, on un obtient un score faible, ce qui est logique, puisque cet algorithme sélectionne les meilleurs résultats pour faire évoluer le tableau de probabilités. Ici, pratiquement tous les résultats sont gardés, le tableau ne peut donc pas évoluer.

On peut aussi voir que N et M doivent être assez grands pour obtenir des résultats convenables. On observe que lorsque N=100 et M=50, on obtient un résultat inférieur qu’avec des valeurs supérieures. On s’aperçoit également que plus N est grand, plus le temps d’exécution de l’algorithme est important.

Il est également nécessaire que N ne soit pas trop petit, car, dans ce cas, le programme va établir des probabilités qui ne seront pas représentatives de la population.

Il faut donc choisir N assez grand pour obtenir un résultat convenable, mais pas trop grand non plus pour garder un temps d’exécution acceptable. D’après les résultats, il faudrait choisir M dans l’intervalle [1/3N ;1/2N]. S doit rester petit pour ne pas rajouter d’itérations inutiles.