TP N°6 MODELES DE LANGAGE STATISTIQUES

Sauvegardez le fichier en mettant votre NOM dans le nom du notebook. Ce document servira de compte rendu de TP et devra être déposé sur moodle à la fin de la séance.

Après avoir récupéré et lu le sujet, répondre à chacune des questions posées en :
1) Récupérant la commande exécutée sous ubuntu ainsi que la trace associée
2) Commentant le résultat et/ou en apportant des éléments de réponses dans la zone de texte dédiée

RAPPELS: les commandes à utiliser sont des lignes de commandes Unix (cf 1A). Les mécanismes de redirection sont donc utilisés pour (1) envoyer des données en entrée de la commande (<) (2) sauvegarder des données de sortie dans un fichier (>) (3) et envoyer le résultat d'une commande à la commande suivante (|). Les [] indiquent que le contenu est optionnel et que la valeur indiquée est la valeur par défaut. Ne pas inscrire explicitement les [] dans les commandes ...

APPRENTISSAGE DES MODELES DE LANGAGE N-GRAMMES

Génération de la liste de mots et calcul du nombre d'occurrences

QUESTION 1 : En utilisant l'ensemble des fichiers dédiés à l'apprentissage (CORPUS_APP), utiliser l'outil text2wfreq pour générer la liste des mots présents dans ces fichiers ainsi que leur nombre d'occurrences (fichier .wfreq).

```
# text2wfreq    [ -hash 1000000 ] [ -verbosity 2 ] < .text     > .wfreq
# exécuter la commande dans un terminal, copier coller la commande et la trace et
copier coller le début (10 premières # lignes commande head) et la fin du fichier
généré (10 dernières lignes commande tail). Trop long sinon
...
```

QUESTION 2 : A l'aide de la commande unix sort, trier la liste de façon à la classer les mots par nombres d'occurrences décroissants. Quels sont les mots les plus fréquents ? A quoi correspondent-ils et pourquoi ?

Réponse :

Entrée [ ]:

Génération du vocabulaire

QUESTION 3 : Utiliser l'outil wfreq2vocab pour obtenir la liste des mots distincts

```
# wfreq2vocab   [ -top 20000 | -gt 10] [ -records 1000000 ] [ -verbosity 2]  <
.wfreq     > .vocab
# exécuter la commande dans un terminal, copier coller la commande et la trace et
copier coller le début (10 premières # lignes commande head) et la fin du fichier
généré (10 dernières lignes commande tail). Trop long sinon
...
```

Liste des bigrammes et trigrammes et nombres d'occurrences associés

QUESTION 4 : Utiliser l'outil text2wngram pour générer l'ensemble des paires de mots présentes dans le corpus d'apprentissage (fichier .wngram) et compter leur nombre d'occurrences.

```
# text2wngram [ -n 3 ] [ -temp /usr/tmp/ ][ -chars n ] [ -words m ][ -gzip | -compress ] [ -verbosity 2 ] < .text     #  > .wngram
# exécuter la commande dans un terminal, copier coller la commande et la trace et copier coller le début (10 premières # lignes commande head) et la fin du fichier généré (10 dernières lignes commande tail). Trop long sinon
...
```

Quels sont les bigrammes les plus fréquents ? A quoi correspondent-ils ?

Réponse ...

QUESTION 5 : Utiliser l'outil text2idngram pour générer une autre version de cette liste (fichier .idngram) et comparer le résultat avec le précédent. Ne pas oublier l'option (-write_ascii) pour générer le résultat sous forme ascii et non binaire plus difficile à lire mais plus efficace en terme de traitement …

```
# text2idngram -vocab .vocab [ -buffer 100 ] [ -temp /usr/tmp/ ] [ -files 20 ] [ -gzip | -compress ] [ -n 3 ]
# [ -write_ascii ] [ -fof_size 10 ] [ -verbosity 2 ]< .text > .idngram

# exécuter la commande dans un terminal, copier coller la commande et la trace et copier coller le début (10 premières # lignes commande head) et la fin du fichier généré (10 dernières lignes commande tail). Trop long sinon
...
```

QUESTION 6 : Examinez la trace affichée pendant l'exécution de la commande et commentez les éléments qui vous semblent significatifs.

Relancer sans l'option -write_ascii pour obtenir une version binaire à utiliser par la suite (question 8).
 Commande + trace

QUESTION 7 : Reprendre les mêmes étapes (Q4, Q5 et Q6) mais générer les fichiers correspondants aux triplets de mots.
Ajouter les cellules (text brut) nécessaires.

QUESTION 8 : A partir du fichier .idngram binaire généré précédemment, construire un modèle de langage bigramme pour chaque méthode de prélèvement proposée. Utiliser l'outil idngram2lm et l'option –arpa pour générer un fichier au format ARPA (.arpa).

```
# idngram2lm -idngram .idngram
#           -vocab .vocab
#           -arpa .arpa | -binary .binlm
#         [ -context .ccs ]
#         [ -calc_mem | -buffer 100 | -spec_num y ... z ]
#         [ -vocab_type 1 ][ -oov_fraction 0.5 ]
#         [ -linear | -absolute | -good_turing | -witten_bell ]
#         [ -disc_ranges 1 7 7 ] [ -cutoffs 0 ... 0 ]
#         [ -min_unicount 0 ][ -zeroton_fraction 1.0 ]
#         [ -ascii_input | -bin_input ][ -n 3 ]
#         [ -verbosity 2 ] [ -four_byte_counts ]
```

```
#         [ -two_byte_bo_weights
#         [ -min_bo_weight -3.2 ] [ -max_bo_weight 2.5 ]
#            [ -out_of_range_bo_weights 10000 ] ]

Remarque : vous pouvez consulter la documentation section « Discounting strategies
» pour voir les détails concernant les méthodes de prélèvement et calcul du
discount ratio (delta ou d(r)).

# exécuter la commande dans un terminal, copier coller la commande et la trace
```

```
Quelles informations sont données dans la trace. Quelles sont celles qui vous
paraissent significatives. Comparer les valeurs obtenues lors de la génération des
différents modèles.
Commentez ici
```

```
QUESTION 9 : Reprendre l'étape précédente (Q8) pour générer un modèle de langage
trigramme avec chacune des méthodes de prélèvement disponibles.

Ajouter les cellules nécessaires pour garder la trace de votre travail
```

```
Vous disposez normalement de 4 modèles de langage bigrammes et 4 modèles de langage
trigrammes appris sur le même corpus. Vous allez les tester sur le même fichier de
test.
```

EVALUTATION DES MODELES DE LANGAGE

```
Calcul de la perplexité et du taux de mots hors vocabulaire. Utiliser le fichier
qui se trouve dans CORPUS_TEST1 pour effectuer cette évaluation.

QUESTION 10 : Evaluer chacun des 8 modèles générés précédemment en utilisant la
combinaison de commandes suivante :
```

```
# echo "perplexity –text corpus_test -probs fichier.probs -oovs fichier.oovs -
annotate fichier.annote " | evallm –arpa # modele_a_tester

#avec
#   evallm [ -binary .binlm | -arpa .arpa [ -context .ccs ]

# exécuter les commandes d'évaluation dans un terminal, copier coller la commande
et la trace
```

```
Commenter une des traces
...
```

ANALYSE DES RESULTATS

```
QUESTION 11 : Présenter sous forme de tableau ou équivalent (cf. exemple) les
différents résultats obtenus (OOV, Entropie, Perplexité) et déterminer quel est le
modèle a priori le plus performant.
```

Méthode de prélèvement Evaluation Modèle Bigramme PERPLEXITE et OOV Evaluation Modèle Trigramme PERPLEXITE et OOV

- Linear ... ...

- Absolute ... ...

- Good Turing ... ...

- Witten Bell ... ...

QUESTION 12 : Consulter les différents fichiers générés (.probs, .oovs, .annote). Copier-coller les 20 premières lignes (commande head) de chaque dans une cellule texte brut et commenter.

QUESTION 13 : Utiliser votre meilleur modèle bigramme et votre meilleur modèle trigramme et évaluer les sur le fichier contenu dans CORPUS_TEST2. Copier-coller les résultats et commenter.

Faire un pdf de votre notebook (Ctrl P + imprimer dans un fichier) Déposer le notebook et sa version pdf sur moodle.