IFT 607 : Devoir 2
Travail individuel

Remise : 31 octobre 2014, 12h00 (**au plus tard**)

Ce devoir comporte 2 questions de programmation. Vous trouverez tous les fichiers nécessaires pour ce devoir ici : `http://info.usherbrooke.ca/hlarochelle/cours/ift607_A2014/devoir_2/devoir_2.zip`.

Veuillez soumettre vos solutions à l'aide de l'outil **turnin** :

```
turnin -c ift607 -p devoir_2 solution_ngramme2.py solution_etiqueteur.py
```

1. [**5 points**] Programmez des modèles de langue trigramme avec lissage par repli de Katz et lissage interpolé de Kneser Ney.

   Le programme doit être écrit dans le langage Python. Plus spécifiquement, vous devez compléter les fonctions et méthodes du fichier `solution_ngramme2.py` disponible sur le site web du cours. Vous avez à compléter les fonctions `extraire_vocabulaire` et `remplacement_unk`, la méthode `log_probabilite_phrase` de la classe parent `Trigramme`, ainsi que les méthodes `entrainement` et `log_probabilite_mot` des classes enfants `TrigrammeRepliKatz` et `TrigrammeKneserNey`.

   Tous les détails sur ces fonctions et méthodes sont contenus dans leur *docstring* (voir les méthodes de la classe parent pour les méthodes de `TrigrammeRepliKatz` et `TrigrammeKneserNey`).

   Le script Python `ngramme2.py` importera `solution_ngramme2.py` (qui doit être dans le même répertoire) et l'utilisera afin d'utiliser les deux modèles de langue sur le corpus Brown.

   Pour utiliser le corpus Brown, vous devez installer la librairie *nltk*, comme suit :

   ```
   pip install --user nltk
   ```

   Ensuite, vous devez télécharger le corpus Brown. Pour ce faire, exécuter les instructions Python suivantes (par exemple via l'interpréteur) :

   ```
   import nltk
   nltk.download('brown')
   ```

   Voici comment utiliser le script `ngramme2.py` :

   ```
   Usage: python ngramme2.py [mot1 mot2 ...]

   Si aucun argument n'est donné, une comparaison
   sera faite avec un cas pour lequel les résultats
   attendus sont connus.

   Optionnellement, une phrase, spécifiée mot à mot,
   peut être fournie. Le programme retournera alors
   la log-probabilité de cette phrase.
   ```

2. **[5 points]** Programmez un modèle HMM (ordre 1) pour l'étiquetage morpho-syntaxique. Utilisez le lissage *add delta* pour estimer les tables de probabilités du HMM.

   Le programme doit être écrit dans le langage Python. Plus spécifiquement, vous devez compléter les fonctions et méthodes du fichier `solution_etiqueteur.py` disponible sur le site web du cours. Vous avez à compléter les fonctions `extraire_vocabulaire` et `remplacement_unk`, ainsi que les méthodes `entrainement` et `etiqueter` de la classe `Etiqueteur`. Tous les détails sur ces fonctions et méthodes sont contenus dans leur *docstring*.

   À noter que, contrairement au cas des modèles de langue, les phrases n'ont pas à être encadrées des balises `<s>` et `</s>`. De plus, les phrases étiquetées correspondent à une **liste de paires**, où le premier élément de chaque paire est le mot et le deuxième l'étiquette grammaticale.

   Le script Python `etiqueteur.py` importera `solution_etiqueteur.py` (qui doit être dans le même répertoire) et l'utilisera afin d'appliquer le modèle HMM sur un sous-ensemble du corpus Penn Treebank.

   Pour utiliser ce corpus, vous devez installer la librairie *nltk*, comme suit :

   ```
   pip install --user nltk
   ```

   Ensuite, vous devez télécharger le corpus. Pour ce faire, exécuter les instructions Python suivantes (par exemple via l'interpréteur) :

   ```
   import nltk
   nltk.download('treebank')
   ```

   Voici comment utiliser le script `etiqueteur.py` :

   ```
   Usage: python etiqueteur.py [mot1 mot2 ...]

   Si aucun argument n'est donné, une comparaison
   sera faite avec un cas pour lequel les résultats
   attendus sont connus.

   Optionnellement, une phrase, spécifiée mot à mot,
   peut être fournie. Le programme retournera l'étiquetage
   morpho-syntaxique prédit par le modèle HMM.
   ```