Prof. Amal Zouaq
Chargés de laboratoire : Dominique Piché ; Charles-Olivier Favreau ; Fabrice Charbonneau

École Polytechnique de Montréal

Département Génie Informatique et Génie Logiciel

INF8460 – Traitement automatique de la langue naturelle

TP2 INF8460

Automne 2021

Objectif : Effectuer de la classification de blogs au moyen d'un modèle BOW traditionnel et avec un Bi-LSTM (avec et sans plongements lexicaux).

1. DESCRIPTION

Le but de ce TP est de comparer différents classificateurs que vous allez entraîner à reconnaitre le genre (masculin, féminin) d'un blogueur.

2. LIBRARIES PERMISES

- Jupyter notebook
- NLTK
- Numpy
- Pandas
- Sklearn
- Keras
- Pour toute autre librairie, demandez à votre chargé de laboratoire via le forum du cours sur Moodle

3. INFRASTRUCTURE

- Vous avez accès aux GPU du local L-4818. Dans ce cas, vous devez utiliser le dossier temp (voir le tutoriel VirtualEnv.pdf)
- Vous pouvez aussi utiliser l'environnement Google Colab : https://colab.research.google.com/

4. DESCRIPTION DES DONNEES

Plusieurs fichiers au format csv sont disponibles et sont extraits de blogs échangés sur Blogger.com. Les données sont disponibles dans le répertoire data.
Elles ont la forme suivante :

- Train : contient 7000 exemples de blogs écrits par des hommes et des femmes
- Val : contient 1000 exemples
- Test : contient 2000 exemples

5. ETAPES DU TP

A partir du notebook inf8460_A21_TP2 qui est distribué, vous devez réaliser les étapes suivantes. (Notez que les cellules dans le squelette sont là à titre informatif, il est fort probable que vous rajoutiez des sections au fur et à mesure de votre TP).

5.1. Pré-traitement et description des données (10%)

1) *(4 points)* Effectuez le pré-traitement suivant sur le champ *text* : convertir le texte en minuscules, supprimer les stopwords et gardez les signes de ponctuation.
2) *(3 points)* Calculez les statistiques suivantes : Nombre total de types ; Nombre total de types/genre ; top 20 des types les plus fréquents par genre (male/female)
3) *(3 points)* Dans deux figures séparées, affichez la distribution des sujets (topics) en termes de fréquence pour les hommes (male) et les femmes (female).

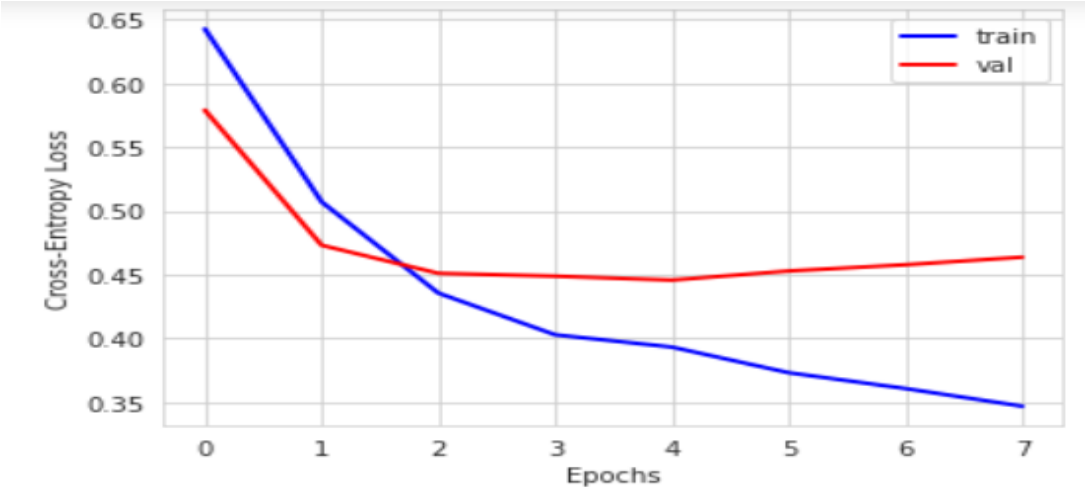
5.2. Classification traditionnelle à base de textes (35%)

1) *(10 points)* Implémentez un classificateur Naïve Bayes et de régression logistique avec Sklearn se basant sur un modèle sac de mots unigrame du contenu des blogues, pondéré avec TF-IDF.
2) *(10 points)* Implémentez un classificateur Naïve Bayes et de régression logistique avec Sklearn se basant sur un modèle sac de mots unigrame et bigramme du contenu des blogues, pondéré avec TF-IDF.
3) *(5 points)* Testez vos deux modèles et retournez les valeurs des métriques d'évaluation : accuracy, précision, rappel et F1-score par classe. Affichez aussi les macro moyennes pour toutes ces métriques.
4) *(10 points)* Trouvez les paramètres optimaux en utilisant un pipeline et GridSearch. Retournez la configuration optimale et la performance pour chaque algorithme (Naive Bayes, Régression logistique) avec cette configuration en testant les paramètres suivants dans votre GridSearch:
   - Modèle basé sur les mots ou les caractères
   - Type de n-gramme (1, 2 et leur combinaison)
   - Utilisation ou non de IDF
   - Taille du vocabulaire : 5000, 10000, none
   - Pour Naive Bayes, testez différentes valeurs du paramètre alpha (bayes__alpha)
   - Pour la régression logistique, testez différentes stratégies de régularisation (logistic_penalty)

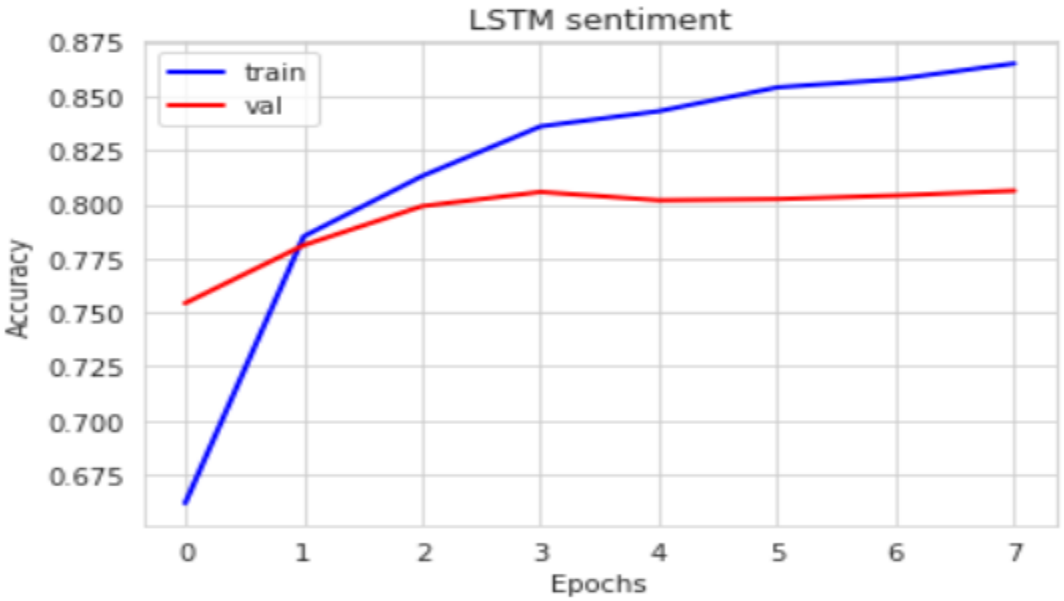
5.3. Classification neuronale (45%)

1) *(10 points)* Implémentez un modèle à deux couches Bi-LSTM avec Keras sans plongements pré-entrainés et entrainez-le sur l'ensemble d'entrainement (voir les points suivants). Assurez-vous que votre séquence maximale prenne en compte la plus longue séquence de votre ensemble de données d'entrainement.

2) (2 *points*) Effectuez un « EarlyStopping » prenant en compte une « val_loss » minimale et en attendant 3 époques
3) (2 *points*) En utilisant un ModelCheckpoint, sauvegardez votre meilleur modèle pour lequel la métrique « val_accuracy » soit maximale
4) *(5 points)* Affichez dans une figure la fonction de perte / époque sur le jeu de données d'entrainement (train) et de validation (val). Exemple (attention, cela ne correspond pas au jeu de données de ce TP, il s'agit juste ici d'une illustration) :



5) *(5 points)* Affichez dans une figure la précision globale (accuracy) / époque sur le jeu de données d'entrainement (train) et de validation (val). Exemple :



6) *(2 points)* Chargez le meilleur modèle retenu, et affichez les performances de ce meilleur modèle sur l'ensemble de test. Utilisez la métrique Accuracy. (Exemple : Accuracy: 0.82)
7) *(10 points)* En utilisant des plongements lexicaux pré-entrainés GLOVE, implémentez un modèle Bi-LSTM à deux couches avec Keras utilisant ces plongements et avec les mêmes

critères d'EarlyStopping et de ModelCheckpoint que précédemment, et entrainez-le sur l'ensemble d'entrainement. Sauvegardez votre meilleur modèle.

8) *(2 points)* De la même manière que précédemment, affichez dans deux figures la fonction de perte et la précision globale (accuracy) / époque sur le jeu de données d'entrainement (train) et de validation (val).
9) *(2 points)* Affichez les performances de votre meilleur modèle à base de plongements sur l'ensemble de test. Utilisez la métrique de précision globale (Accuracy).
10) (*5 points*) Si ce n'est pas déjà le cas, tentez de dépasser les performances de votre meilleur modèle traditionnel en créant un nouveau modèle Bi-LSTM (vous pouvez ajouter des couches, utiliser du dropout, utiliser d'autres modèles de plongements, etc.)

5.4. Évaluation (10%)

1) *(4 points)* Affichez une table récapitulative des performances de vos différents modèles sur l'ensemble de test
2) *(6 points)* Indiquez votre conclusion dans une cellule (meilleur algorithme ? meilleure configuration pour l'apprentissage machine « traditionnel » ? effet du GridSearch ? meilleur modèle Bi-LSTM ?)

6. LIVRABLES

Vous devez remettre sur Moodle:

1- *Le code* : Un Jupyter notebook en Python qui contient le code implanté avec les librairies permises. Le code doit être exécutable sans erreur et accompagné des commentaires appropriés dans le notebook de manière à expliquer les différentes fonctions et étapes dans votre projet. Nous nous réservons le droit de demander une démonstration ou la preuve que vous avez effectué vous-mêmes les expériences décrites. *Attention, en aucun cas votre code ne doit avoir été copié d'une quelconque source*. Les critères de qualité tels que la lisibilité du code et des commentaires sont importants. Tout votre code et vos résultats doivent être exécutables et reproductibles ;

2- Un fichier *requirements.txt* doit indiquer toutes les librairies / données nécessaires ;

3- Un lien *GoogleDrive ou similaire* vers les modèles nécessaires pour exécuter votre notebook si approprié ;

4- Un document *contributions.txt :* Décrivez brièvement la contribution de chaque membre de l'équipe. Tous les membres sont censés contribuer au développement. Bien que chaque membre puisse effectuer différentes tâches, vous devez vous efforcer d'obtenir une répartition égale du travail. En particulier, tous les membres du projet devraient participer à la conception du TP et participer activement à la réflexion et à l'implémentation du code.

EVALUATION

Votre TP sera évalué selon les critères suivants :

1. Exécution correcte du code
2. Performance des modèles
3. Organisation du notebook

4. Qualité du code (noms significatifs, structure, performance, gestion d'exception, etc.)
5. Commentaires clairs et informatifs

CODE D'HONNEUR

Règle 1: Le plagiat de code est bien évidemment interdit.
Règle 2: Vous êtes libres de discuter des idées et des détails de mise en œuvre avec d'autres équipes. Cependant, vous ne pouvez en aucun cas consulter le code d'une autre équipe INF8460, ou incorporer leur code dans votre TP.
Règle 3: Vous ne pouvez pas partager votre code publiquement (par exemple, dans un dépôt GitHub public) tant que le cours n'est pas fini.