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

TP1 INF8460

Automne 2021

1. DESCRIPTION

Dans ce TP, l'idée est d'effectuer de la recherche de passages de texte dans un corpus à partir d'une question en langue naturelle. Les questions et passages sont en anglais.

Voici un exemple :

Entrée : Question : What causes precipitation to fall?

Solution - Trouver un passage qui contient la réponse à la question : In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called "showers".

Ici la réponse est en gras dans le texte.

2. LIBRARIES PERMISES

- Jupyter notebook
- NLTK
- Numpy
- Pandas
- Sklearn
- Pour toute autre librairie, demandez à votre chargé de laboratoire

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)

4. DESCRIPTION DES DONNEES

Dans ce projet, vous utiliserez le jeu de données dans le répertoire *data*. Il est décomposé en données d'entrainement (train), de validation (dev) et de test (test).
Nous ne mettrons à votre disposition que les données d'entrainement et de validation. Les données de test ne contiennent pas le paragraphe de réponse et doivent être complétées avec les résultats de votre système.

Nous vous fournissons un ensemble de données qui comprend un corpus (*corpus.csv*) qui contient tous les passages et leurs identificateurs (ID) et un jeu de données qui associe une question, un passage, et une réponse qui est directement extraite du passage. Notez que certains passages contiennent des balises HTML et qu'il vous faudra procéder à un prétraitement de ces passages pour les enlever.
Ce jeu de données est composé de trois sous-ensembles :

- *Train* : ensemble d'entraînement de la forme <QuestionID, QuestionText, PassageID, Réponse>. Le but est donc d'entrainer votre modèle à retrouver le passage qui contient la réponse à la question.
- *Validation* : De la même forme que le Train, il vous permet de valider votre entraînement et de tester les performances de certains modules.
- *Test* : Un ensemble secret qui est utilisé pour évaluer votre système complet. Il est de la forme <QuestionID, Question>. Votre système doit trouver dans le corpus **corpus.csv** le ou les passages les plus pertinents.

Notez qu'il est possible de répondre aux requis du TP sans utiliser la réponse à la question. C'est à vous de choisir si vous utilisez la réponse ou non.

5. ETAPES DU TP

A partir du notebook *inf8460_A21_TP1* qui est distribué, vous devez réaliser les étapes suivantes. (Noter 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 (12 points)

Les passages et questions de votre ensemble de données doivent d'abord être représentés et indexés pour ensuite pouvoir effectuer une recherche de passage pour répondre à une question. On vous demande donc d'implémenter une étape de pré-traitement des données.

1) (*6 points*) Complétez les fonctions retournant les informations suivantes (une fonction par information, chaque fonction prenant en argument le corpus (passages, questions) composé d'une liste de phrases segmentées en jetons/tokens) :
   a. Le nombre total de jetons (mots non distincts)
   b. Le nombre total de mots distincts (les types qui constituent le vocabulaire)
   c. Les N mots les plus fréquents du vocabulaire (N est un paramètre avec une valeur par défaut de 10) ainsi que leur fréquence
   d. Le ratio jeton/type
   e. Le nombre total de lemmes distincts
   f. Le nombre total de racines (stems) distinctes

2) (*1 point*). Ecrivez une fonction `explore_corpus()` qui fait appel à toutes les fonctions en 1) et imprime leur résultat.

3) (*5 points*) Pour la suite du TP, vous devez effectuer le pré-traitement du corpus (questions, passages) en convertissant le texte en minuscules, en segmentant le texte, en supprimant les mots outils et en lemmatisant le texte. Chaque opération doit avoir sa fonction python si elle n'est pas déjà implantée dans la question 1) précédente.

5.2. Représentation de questions et de passages (14 points)

1) (*10 points*) En utilisant sklearn et à partir de votre corpus pré-traité, vous devez implanter un modèle M1 qui permet de représenter chaque passage et question avec votre vocabulaire, en utilisant un modèle sac de mots des n-grammes (n=1) qu'ils contiennent et en pondérant ces éléments avec TF-IDF. Notez que les questions doivent aussi être inclues dans la construction du vocabulaire.

2) (*4 points*) vous devez implanter un modèle M2 maintenant avec un modèle n-gramme (n=1,2) mélangeant les unigrammes et les bigrammes et pondéré avec TF-IDF.

Pour M1 et M2, assurez-vous de réutiliser la même fonction avec comme paramètre les n-grammes à considérer.

5.3. Ordonnancement des passages (10 points)

Maintenant que vous avez une représentation de vos passages et questions, il faut être capable de déterminer quel passage sera le plus pertinent pour la question posée. Il vous faut donc retrouver un top-N (N=1,5,10 …) de passages utiles pour répondre à la question. Ces passages devront être ordonnés du plus pertinent au moins pertinent. Idéalement le passage à la position 1 sera celui qui contient la réponse à la question.

Vous devez écrire des fonctions pour évaluer la similarité entre la représentation de la question et celle de chaque passage et retourner les N passage les plus similaires où N est un paramètre.

1) (*5 points*) En utilisant la distance euclidienne
2) (*5 points*) En utilisant la distance cosinus

5.4. Évaluation (15 points)

En utilisant votre *ensemble de validation* :

1) (*5 points*) Vous devez calculer la précision top-N (N=1,5,10, 50) de votre modèle M1 et M2 avec la distance euclidienne et cosinus et les afficher.
2) (*5 points*) Pour chacun de ces modèles, générez une courbe de performance faisant varier le N (N=1, 5, 10, 50)
3) (*5 points*) A cette étape, vous devez produire un fichier `passage_submission_M1.csv` et `passage_submission_M2.csv` qui contient pour toutes les questions de *l'ensemble de test* le top-N des passages retournés par votre modèle M1 et M2 pour y répondre. C'est à vous de déterminer si vous utiliserez la distance euclidienne ou cosinus basé sur vos résultats d'évaluation sur l'ensemble de validation en 1) et 2). Le fichier doit respecter le format suivant pour chaque top_N(N=1,5,10,50) : <QuestionID, PassageID1 ;… ;PassageIDN>. Le format est démontré dans `sample_passage_submission.csv`.

5.5. Le plus (24 points)

1) (*21 points*) Vous devez proposer un modèle M3 différent (basé sur l'apprentissage machine par exemple) afin de déterminer un score de pertinence d'un passage pour une question donnée et ordonner les passages.
   - Faites une petite recherche sur l'état de l'art en consultant https://nlp.stanford.edu/IR-book/information-retrieval-book.html
   - Vous êtes libres de proposer une autre métrique de poids, ou une autre façon d'ordonner les passages (exemple : méthodes de type *learning to rank*) et de partir de votre corpus initial ou de votre ordonnancement en M1/M2 (choisissez le meilleur) et de réordonnancer les passages obtenus par votre premier modèle.
   - Expliquez votre modèle et son intérêt dans votre notebook. Le nombre de points obtenus dépendra de l'effort mis dans cette partie.

2) (*2 point*) Vous devez ensuite afficher l'évaluation de votre modèle M3 tel que décrit dans la section 5.4 *Evaluation* en utilisant les mêmes fonctions. Notamment, vous devez comparer les performances de vos modèles M1, M2 et M3 sur l'ensemble de validation avec une courbe de performance faisant varier le N (N=1, 5, 10, …)
3) (*1 point*) En utilisant votre modèle M3, vous devez produire un fichier `passage_submission_M3.csv` qui contient pour toutes les questions de *l'ensemble de test* le top-N des passages retournés par votre système pour y répondre. Le fichier doit respecter le

format suivant pour chaque top_N (N=1,5,10,50) : <QuestionID, PassageID1…PassageIDN>. Le format est démontré dans `sample_passage_submission.csv`

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- Les fichiers de soumission de données de test *passage_submission_M1.csv, passage_submission_M2.csv et passage_submission_M3.csv ;*

5- 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 correcte 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.