Projet Chatbot Juridique : Partie Similarité

Margaux Duhayon Milena Chaîne Ferial Yahiaoui Elvira Quesada

Institut national des langues et civilisations orientales

10 avril 2019

Plan de l'exposé

Introduction
Objectif

- L'objectif de notre travail de groupe était d'effectuer la recherche de similarité sur une question posée par un utilisateur.

Introduction
Récapitulatif

- Crawling => XML
- Prétraitement => conll
- Catégorisation => integration de la classe dans la recherche de similarité par Elvira => interface
  Pas d'integration de la classe dans notre recherche de similarité. => intérêt axé sur l'integration des prétraitements + trouver le meilleur calcul de similarité + génération de la réponse.
- Elvira => XML
- Margaux, Milena et Ferial => conll

Introduction
Récapitulatif

- Crawling => XML
- Prétraitement => conll
- Catégorisation => integration de la classe dans la recherche de similarité par Elvira => interface
  Pas d'integration de la classe dans notre recherche de similarité. => intérêt axé sur l'integration des prétraitements + trouver le meilleur calcul de similarité + génération de la réponse.
- Elvira => XML
- Margaux, Milena et Ferial => conll

Introduction
Récapitulatif

- Crawling => XML
- Prétraitement => conll
- Catégorisation => integration de la classe dans la recherche de similarité par Elvira => interface
  Pas d'integration de la classe dans notre recherche de similarité. => intérêt axé sur l'integration des prétraitements + trouver le meilleur calcul de similarité + génération de la réponse.
- Elvira => XML
- Margaux, Milena et Ferial => conll

Introduction
Récapitulatif

- Crawling => XML
- Prétraitement => conll
- Catégorisation => integration de la classe dans la recherche de similarité par Elvira => interface
  Pas d'integration de la classe dans notre recherche de similarité. => intérêt axé sur l'integration des prétraitements + trouver le meilleur calcul de similarité + génération de la réponse.
- Elvira => XML
- Margaux, Milena et Ferial => conll

Introduction
Récapitulatif

- Crawling => XML
- Prétraitement => conll
- Catégorisation => integration de la classe dans la recherche de similarité par Elvira => interface
  Pas d'integration de la classe dans notre recherche de similarité. => intérêt axé sur l'integration des prétraitements + trouver le meilleur calcul de similarité + génération de la réponse.
- Elvira => XML
- Margaux, Milena et Ferial => conll

Recherche de similarité sur le corpus en format XML : Elvira

- Présentation sur Jupyter Notebook. On te laisse la parole, Elvira :)

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- Parcours du corpus fichier par fichier : 4 couches.
- A chaque fichier parcouru, on récupère les identifiants comportant des q (questions) uniquement et on les ajoute dans une liste.
- Ensuite, on lit chaque fichier comportant des q, on split chaque ligne par une tabulation.
- On récupère le contenu de chaque colonne et on ajoute la colonne correspondant aux mots dans une liste, celle des lemmes dans une autre et la même chose pour le contenu des POS.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- Parcours du corpus fichier par fichier : 4 couches.
- A chaque fichier parcouru, on récupère les identifiants comportant des q (questions) uniquement et on les ajoute dans une liste.
- Ensuite, on lit chaque fichier comportant des q, on split chaque ligne par une tabulation.
- On récupère le contenu de chaque colonne et on ajoute la colonne correspondant aux mots dans une liste, celle des lemmes dans une autre et la même chose pour le contenu des POS.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- Parcours du corpus fichier par fichier : 4 couches.
- A chaque fichier parcouru, on récupère les identifiants comportant des q (questions) uniquement et on les ajoute dans une liste.
- Ensuite, on lit chaque fichier comportant des q, on split chaque ligne par une tabulation.
- On récupère le contenu de chaque colonne et on ajoute la colonne correspondant aux mots dans une liste, celle des lemmes dans une autre et la même chose pour le contenu des POS.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- Parcours du corpus fichier par fichier : 4 couches.
- A chaque fichier parcouru, on récupère les identifiants comportant des q (questions) uniquement et on les ajoute dans une liste.
- Ensuite, on lit chaque fichier comportant des q, on split chaque ligne par une tabulation.
- On récupère le contenu de chaque colonne et on ajoute la colonne correspondant aux mots dans une liste, celle des lemmes dans une autre et la même chose pour le contenu des POS.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- On obtient 4 listes, si on compte celles des idetntifiants. Pourquoi ?
- Ces 4 listes vont nous servir à créer des dictionnaires. Plus précisément, elles constitueront les valeurs de ces dictionnaires. Pourquoi ?
- Ils seront tous ajoutés à une liste globale. Pourquoi ?
- Cette liste golbale permettra de créer une nouvelle structure de données, DataFrames, de la bibilothèque Pandas.
  Ces DataFrames, permettent de stocker des données selon 2 dimensions : lignes et colonnes.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- On obtient 4 listes, si on compte celles des idetntifiants. Pourquoi ?
- Ces 4 listes vont nous servir à créer des dictionnaires. Plus précisément, elles constitueront les valeurs de ces dictionnaires. Pourquoi ?
- Ils seront tous ajoutés à une liste globale. Pourquoi ?
- Cette liste golbale permettra de créer une nouvelle structure de données, DataFrames, de la bibilothèque Pandas.
  Ces DataFrames, permettent de stocker des données selon 2 dimensions : lignes et colonnes.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- On obtient 4 listes, si on compte celles des idetntifiants. Pourquoi ?
- Ces 4 listes vont nous servir à créer des dictionnaires. Plus précisément, elles constitueront les valeurs de ces dictionnaires. Pourquoi ?
- Ils seront tous ajoutés à une liste globale. Pourquoi ?
- Cette liste golbale permettra de créer une nouvelle structure de données, DataFrames, de la bibilothèque Pandas.
  Ces DataFrames, permettent de stocker des données selon 2 dimensions : lignes et colonnes.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Parcours

- On obtient 4 listes, si on compte celles des idetntifiants. Pourquoi ?
- Ces 4 listes vont nous servir à créer des dictionnaires. Plus précisément, elles constitueront les valeurs de ces dictionnaires. Pourquoi ?
- Ils seront tous ajoutés à une liste globale. Pourquoi ?
- Cette liste golbale permettra de créer une nouvelle structure de données, DataFrames, de la bibilothèque Pandas.
  Ces DataFrames, permettent de stocker des données selon 2 dimensions : lignes et colonnes.

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Exemple Dataframes

```
1  import pandas as pd
2  import pickle
3
4  # partie du code concernant la DataFrame +
5  #le fichier pickle
6
7          for filename in id_fi:
8              dico_q['id'] = filename
9                  dico_q['lemmes'] = listlem
10                 dico_q['mots'] = listw
11                 dico_q['pos'] = listpos
```

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Exemple Dataframes : Suite

```
            # puis, on met tous les dic dans une
            # liste globale

            liste_q.append(dico_q)


test_dataframe = pd.DataFrame(liste_q)
test_dataframe.to_pickle("./corpuspd.pkl")

# on crée ensuite un fichier pickle de tout le
# corpus pour que les calculs de similarité
# soient effectués
```

Recherche de similarité sur le corpus de prétaraitement en format conll : Margaux, Milena et Ferial

Manipulation des données conll : Exemple d'une DataFrame

```
MacBook-Air-de-Ferial:cours ferialyahiaoui$ python3 parcours_v2_dataframe_pickle_pos.py
Format de la dataframe :
                    id  ...                                                pos
0   iris7829_q1.conll  ...  [NOM, PUN, PRO:PER, VER:pres, VER:infi, ADV, P...
1   iris7829_q3.conll  ...  [PRP, PRO:DEM, PRO:REL, VER:pres, PRP, DET:ART...
2   iris7829_q2.conll  ...  [ADV, PRP, NOM, NOM, PRP, DET:POS, NOM, SENT, ...
3   iris7758_q1.conll  ...  [NOM, SENT, PRO:PER, VER:pres, ADJ, ADV, PUN, ...
4   iris6225_q1.conll  ...  [NOM, SENT, VER:pper, KON, PRO:PER, VER:pres, ...
5   iris6225_q3.conll  ...  [PRO:PER, PRO:PER, VER:pres, PUN, KON, KON, PR...
6   iris6225_q4.conll  ...  [PRP, DET:ART, NOM, PUN, PRO:IND, NOM, VER:pre...
7   iris6225_q2.conll  ...  [NOM, KON, VER:pper, PRP, PRO:DEM, NOM, SENT, ...
8   iris8035_q2.conll  ...      [NOM, PUN, NOM, ADV, PRP, DET:POS, NOM, SENT]
9   iris8035_q1.conll  ...  [NOM, PUN, PRO:PER, VER:pres, ADJ, PRP, PRO:IN...
10  iris7231_q1.conll  ...  [NOM, NAM, VER:pres, VER:pper, VER:pres, VER:i...
11  iris6146_q1.conll  ...  [NOM, PRP, PRO:IND, PUN, VER:pres, DET:ART, NO...

[12 rows x 4 columns]
```

Similarité : Margaux, Milena et Ferial

Comparaisons de différentes mesures de similarité avec scikit-learn

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- récupération d'une question de l'utilisateur et application des mêmes prétraitements
- test de trois types de similarités avec scikit : Cosinus, Euclidean, Manhattan

Similarité : Margaux, Milena et Ferial

Comparaisons de différentes mesures de similarité avec scikit-learn

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- récupération d'une question de l'utilisateur et application des mêmes prétraitements
- test de trois types de similarités avec scikit : Cosinus, Euclidean, Manhattan

Similarité : Margaux, Milena et Ferial

Comparaisons de différentes mesures de similarité avec scikit-learn

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- récupération d'une question de l'utilisateur et application des mêmes prétraitements
- test de trois types de similarités avec scikit : Cosinus, Euclidean, Manhattan

Similarité : Margaux, Milena et Ferial

Comment gérer un corpus prétraité avec scikit

```
1  def faux_tokeniseur(qqch):
2      """
3      permet d'éviter les prétraitements de scikit
4      """
5      return qqch
6
7  tfidf = TfidfVectorizer(analyzer='word',
8  tokenizer=faux_tokeniseur,
9  preprocessor=faux_tokeniseur,
10 token_pattern=None)
```

Similarité : Margaux, Milena et Ferial

$02_tfidf.py$

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- apprentissage du vocabulaire et des fréquences inversées
- application de ces poids sur le corpus

Similarité : Margaux, Milena et Ferial

$02_t fidf.py$

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- apprentissage du vocabulaire et des fréquences inversées
- application de ces poids sur le corpus

Similarité : Margaux, Milena et Ferial

$02_t fidf.py$

- utilisation de scikit-learn et du TfIdfVectorizer sur le corpus
- apprentissage du vocabulaire et des fréquences inversées
- application de ces poids sur le corpus

Similarité : Margaux, Milena et Ferial

$03_s im.py$

- script d'interface avec l'utilisateur : il doit s'éxécuter relativement rapidement
- utilisation de phrase2conll et du pickle tfidf sur la question de l'utilisateur
- trois similarités : Cosinus, Euclidean, Manhattan

Similarité : Margaux, Milena et Ferial

$03_s im.py$

- script d'interface avec l'utilisateur : il doit s'éxécuter relativement rapidement
- utilisation de phrase2conll et du pickle tfidf sur la question de l'utilisateur
- trois similarités : Cosinus, Euclidean, Manhattan

Similarité : Margaux, Milena et Ferial

$03_s im.py$

- script d'interface avec l'utilisateur : il doit s'éxécuter relativement rapidement
- utilisation de phrase2conll et du pickle tfidf sur la question de l'utilisateur
- trois similarités : Cosinus, Euclidean, Manhattan

Génération de texte : Margaux, Milena et Ferial

Module Markovify & chaîne de Markov

- Markovify : générateur de chaîne de Markov

- Corpus de réponse et réponse de la question la plus similaire
- Construction du modèle à partir du texte brut

```
model_a = markovify.NewlineText(text_a)
```

- Création de la phrase

```
answer1 = model_b.make_sentence(tries=100)
```

Génération de texte : Margaux, Milena et Ferial

Module Markovify & chaîne de Markov

- Markovify : générateur de chaîne de Markov

- Corpus de réponse et réponse de la question la plus similaire
- Construction du modèle à partir du texte brut

```
model_a = markovify.NewlineText(text_a)
```

- Création de la phrase

```
answer1 = model_b.make_sentence(tries=100)
```

Génération de texte : Margaux, Milena et Ferial

Module Markovify & chaîne de Markov

- Markovify : générateur de chaîne de Markov

markovify 0.7.1

```
pip install markovify
```

- Corpus de réponse et réponse de la question la plus similaire
- Construction du modèle à partir du texte brut

```
model_a = markovify.NewlineText(text_a)
```

- Création de la phrase

```
answer1 = model_b.make_sentence(tries=100)
```

Génération de texte : Margaux, Milena et Ferial

Module Markovify & chaîne de Markov

- Markovify : générateur de chaîne de Markov

- Corpus de réponse et réponse de la question la plus similaire
- Construction du modèle à partir du texte brut

```
model_a = markovify.NewlineText(text_a)
```

- Création de la phrase

```
answer1 = model_b.make_sentence(tries=100)
```

Génération de texte : Margaux, Milena et Ferial

Fonction generation

- Problème de corpus
- Solution utilisée

```
model_combo = markovify.combine([ model_a, model_b ], [1, 3])
```

```
answer2 = model_combo.make_sentence()
```

Génération de texte : Margaux, Milena et Ferial

Fonction generation

- Problème de corpus
- Solution utilisée

```
model_combo = markovify.combine([ model_a, model_b ], [1, 3])
```

```
answer2 = model_combo.make_sentence()
```

Génération de texte : Margaux, Milena et Ferial

Exemple de réponse

Question : Je veux changer de nom ?

Réponse générée à partir de la réponse

```
RÉPONSE : bonjour , le principe est l'immutabilité de son nom de famille. vous pouvez vous renseig
ner auprès du service d'état-civil de votre père comme nom de votre commune. salutations.
```

Réponse générée sur le grand corpus

```
RÉPONSE : Ces ressortissants sont admis sur le droit public et le droit privé.
```

Conclusion
Remarques

- Données difficiles à exploiter notamment à cause de la structure des interactions sur les forums. => absence d'une vraie correpondance question - réponse
- Conséqunece : difficulté pour calculer une vraie similarité question - réponse.
- To be continued...