TP AGENTS CONVERSATIONNELS : TP rasa (version 2.8)

NOM :

Dérouler le TP et remplir le notebook au fur et à mesure. Une fois rempli, ce notebook (ainsi que sa version PDF) seront à déposer sur moodle.

Accéder à la page suivante : https://rasa.com/docs/rasa/2.x/ (https://rasa.com/docs/rasa/2.x/) ATTENTION La version la plus récente est la version 3 mais ce n'est pas celle utilisée dans ce TP

PARTIE 1 : ENVIRONNEMENT DE TRAVAIL MFJA ou PC PERSO

```
Suivant le cas :
    A - (Solution non stable) Si vous êtes sur les PC de la MFJA,
connectez-vous sous Windows et lancer la machine virtuelle X sous
C:\virtual-b\NOM_MV

    Vous êtes sous Ubuntu 18.04 home = sri-rasa  pwd = Rasa-2023 et
vous êtes sous /home/sri-rasa

    B - Si vous avez installé rasa v2.8 sur votre PC (en suivant
les directives sous moodle) et
        vous avez créé un environnement virtuel


Dans les 2 cas, :
    1.1- Créer un répertoire TP_RASA_VOTRE_NOM
        /home/sri-rasa/Documents/TP_RASA_VOTRE_NOM

    1.2- Récupérer la version V0 du CHATBOT sous moodle et la
mettre dans ce dossier.

Dans les deux cas, vous avez accès à la description du chatbot de
base proposé par rasa, soient les fichiers et répertoires suivants
:


.
├── actions
│   ├── __init__.py
│   └── actions.py
├── config.yml
├── credentials.yml
├── data
│   ├── nlu.yml
│   └── stories.yml
│   └── rules.yml
├── domain.yml
├── endpoints.yml
├── models
│   └── <timestamp>.tar.gz
```

```
└── tests
    └── test_stories.yml
```

PARTIE 2 : COMPRENDRE L'ARCHITECTURE ET LE CONTENU DU CHATBOT

```
2.1 Partie NLU = Compréhension

INTENTS et ENTITIES (Intentions et Entités) - rasa V2
------------------------------------------------------------

Observer le contenu du fichier "data/nlu.yml" (Rappelé ci-dessous
en cas de PB d'accès)

version: "2.0"

nlu:
- intent: greet
  examples: |
    - hey
    - hello
    - hi
    - hello there
    - good morning
    - good evening
    - moin
    - hey there
    - let's go
    - hey dude
    - goodmorning
    - goodevening
    - good afternoon

- intent: goodbye
  examples: |
    - good afternoon
    - cu
    - good by
    - cee you later
    - good night
    - bye
    - goodbye
    - have a nice day
    - see you around
    - bye bye
    - see you later

- intent: affirm
  examples: |
    - yes
    - y
    - indeed
    - of course
    - that sounds good
    - correct
```

```
- intent: deny
  examples: |
    - no
    - n
    - never
    - I don't think so
    - don't like that
    - no way
    - not really

- intent: mood_great
  examples: |
    - perfect
    - great
    - amazing
    - feeling like a king
    - wonderful
    - I am feeling very good
    - I am great
    - I am amazing
    - I am going to save the world
    - super stoked
    - extremely good
    - so so perfect
    - so good
    - so perfect

- intent: mood_unhappy
  examples: |
    - my day was horrible
    - I am sad
    - I don't feel very well
    - I am disappointed
    - super sad
    - I'm so sad
    - sad
    - very sad
    - unhappy
    - not good
    - not very good
    - extremly sad
    - so saad
    - so sad

- intent: bot_challenge
  examples: |
    - are you a bot?
    - are you a human?
    - am I talking to a bot?
    - am I talking to a human?
-------------------------------------------------------

Comment interprétez-vous le contenu de ce fichier (= commentez en
quelques lignes) ? Combien d'intents et lesquelles ? Combien
d'entités et lesquelles ?
```

```
2.2. Configuration du chatbot

PIPELINE et POLICIES
```

```
------------------------------------------------------------
Observer le contenu du fichier "config.yml"


# Configuration for Rasa NLU.
# https://rasa.com/docs/rasa/nlu/components/
language: en

pipeline:
# # No configuration for the NLU pipeline was provided.
# # The following default pipeline was used to train your model.
# # If you'd like to customize it, uncomment and adjust the
pipeline.
# # See https://rasa.com/docs/rasa/tuning-your-model for more
information.
#   - name: WhitespaceTokenizer
#   - name: RegexFeaturizer
#   - name: LexicalSyntacticFeaturizer
#   - name: CountVectorsFeaturizer
#   - name: CountVectorsFeaturizer
#     analyzer: char_wb
#     min_ngram: 1
#     max_ngram: 4
#   - name: DIETClassifier
#     epochs: 100
#     constrain_similarities: true
#   - name: EntitySynonymMapper
#   - name: ResponseSelector
#     epochs: 100
#     constrain_similarities: true
#   - name: FallbackClassifier
#     threshold: 0.3
#     ambiguity_threshold: 0.1

# Configuration for Rasa Core.
# https://rasa.com/docs/rasa/core/policies/

policies:
# # No configuration for policies was provided. The following
default policies were used to train your model.
# # If you'd like to customize them, uncomment and adjust the
policies.
# # See https://rasa.com/docs/rasa/policies for more information.
#   - name: MemoizationPolicy
#   - name: RulePolicy
#   - name: UnexpecTEDIntentPolicy
#     max_history: 5
#     epochs: 100
#   - name: TEDPolicy
#     max_history: 5
#     epochs: 100
#     constrain_similarities: true


------------------------------------------------------------

Comment interprétez-vous le contenu de ce fichier ? Chacune des 3
parties ?
```

```
2.3- Gestion de dialogue

2.3.1- STORIES
------------------------------------------------------------
Observer le contenu du fichier "data/stories.yml"

version: "2.0"

stories:

- story: happy path
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_great
  - action: utter_happy

- story: sad path 1
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: affirm
  - action: utter_happy

- story: sad path 2
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: deny
  - action: utter_goodbye

------------------------------------------------------------

Comment interprétez-vous le contenu de ce fichier ? Combien de
stories ? Quel rôle ?
```

```
2.3.2- RULES
------------------------------------------------------------
Observer le contenu du fichier "data/rules.yml"

version: "2.0"

rules:

- rule: Say goodbye anytime the user says goodbye
  steps:
  - intent: goodbye
  - action: utter_goodbye

- rule: Say 'I am a bot' anytime the user challenges
  steps:
  - intent: bot_challenge
```

```
  - action: utter_iamabot
-----------------------------------------------------------

Comment interprétez-vous le contenu de ce fichier ? Combien de
règles ? Quel rôle ?
```

```
2.4- Domaine

Consultez le fichier décrivant l'environnement du chatbot
-----------------------------------------------------------------------
--

Observer le contenu du fichier "domain.yml"

version: "2.0"

intents:
  - greet
  - goodbye
  - affirm
  - deny
  - mood_great
  - mood_unhappy
  - bot_challenge

responses:
  utter_greet:
  - text: "Hey! How are you?"

  utter_cheer_up:
  - text: "Here is something to cheer you up:"
    image: "https://i.imgur.com/nGF1K8f.jpg"

  utter_did_that_help:
  - text: "Did that help you?"

  utter_happy:
  - text: "Great, carry on!"

  utter_goodbye:
  - text: "Bye"

  utter_iamabot:
  - text: "I am a bot, powered by Rasa."

session_config:
  session_expiration_time: 60
  carry_over_slots_to_new_session: true

-----------------------------------------------------------------------
--

Comment interprétez-vous le contenu de ce fichier ? Chacune des
parties ?
```

```
2.5- Actions

Consultez le dossier correspondant aux action
-----------------------------------------------
```

```
Observer le contenu du fichier "action/action.py"

# This files contains your custom actions which can be used to run
# custom Python code.
#
# See this guide on how to implement these action:
# https://rasa.com/docs/rasa/custom-actions


# This is a simple example for a custom action which utters "Hello
World!"

# from typing import Any, Text, Dict, List
#
# from rasa_sdk import Action, Tracker
# from rasa_sdk.executor import CollectingDispatcher
#
#
# class ActionHelloWorld(Action):
#
#     def name(self) -> Text:
#         return "action_hello_world"
#
#     def run(self, dispatcher: CollectingDispatcher,
#             tracker: Tracker,
#             domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
#
#         dispatcher.utter_message(text="Hello World!")
#
#         return []

------------------------------------------------------------------
--

Comment interprétez-vous le contenu de ce fichier ?
```

PARTIE 3 : Activation de l'environnement virtuel où rasa est installé

```
3- ENVIRONNEMENT VIRTUEL

RASA a été installé en local dans l'environnement virtuel
env_rasa2.8 construit avec conda. Pour utiliser il faut d'abord
lancer conda

# Permettre l'utilisation de conda à travers le script .bashrcCONDA
qui se trouve sous /home/sri-rasa
source ~/.bashrcCONDA
# vous devez avoir un prompt commençant par (base) -->
environnement de base de conda

# Vérifier que l'environnement virtuel env_rasa2.8 existe bien
conda info -e

# vous devez avoir au moins les lignes
base                  /home/sri-rasa/miniconda3
```

```
env_rasa2.8          /home/sri-rasa/miniconda3/envs/env_rasa2.8
--> environnement à utiliser

# activer l'environnement virtuel env_rasa2.8 pour utiliser rasa
conda activate env_rasa2.8

# vous pouvez visualiser l'ensemble des dépendances avec
conda list
...

# version rasa
rasa --version

Rasa Version      :         2.8.0
Minimum Compatible Version: 2.8.0
Rasa SDK Version  :         2.8.6
Rasa X Version    :         0.39.3
Python Version    :         3.7.11
Operating System  :         Linux-5.4.0-84-generic-x86_64-with-
debian-buster-sid
Python Path       :         /home/sri-rasa/miniconda3
/envs/env_rasa2.8/bin/python

# desactivation de l'environnement en fin de session : conda
deactivate
conda deactivate
```

PARTIE 4 : APPRENTISSAGE ET EVALUATION DU CHATBOT

```
AVANT TOUTE CHOSE, faire une copie du dossier CHATBOT_V0 -->
CHATBOT_<VOTRE_NOM>_V1 et positionnez-vous dans ce nouveau dossier

4- APPRENTISSAGE

Consultez le répertoire ./models de votre Chatbot (V1) si celui-ci
contient un fichier <Nom_modele>.tar.gz supprimez-le

Depuis le répertoire CHATBOT_<VOTRE_NOM>_V1 exécutez la commande :
rasa train


------------------------------------------------------------------
---------------
Copier coller ici la trace de cette exécution



------------------------------------------------------------------
------------------------

Comment interprétez-vous cette trace ? Faites le lien avec le
fichier config.yml et commentez ce que vous pensez avoir compris.

Consultez à nouveau le répertoire ./models de votre Chatbot (V1)
que contient-il ?
```

PARTIE 5 : EVALUATION

```
5.1- Données de test

Observer le contenu du fichier ./tests/test_stories.yml
-------------------------------------------------------
#### This file contains tests to evaluate that your bot behaves as
expected.
#### If you want to learn more, please see the docs:
https://rasa.com/docs/rasa/testing-your-assistant

stories:
- story: happy path 1
  steps:
  - user: |
      hello there!
    intent: greet
  - action: utter_greet
  - user: |
      amazing
    intent: mood_great
  - action: utter_happy

- story: happy path 2
  steps:
  - user: |
      hello there!
    intent: greet
  - action: utter_greet
  - user: |
      amazing
    intent: mood_great
  - action: utter_happy
  - user: |
      bye-bye!
    intent: goodbye
  - action: utter_goodbye

- story: sad path 1
  steps:
  - user: |
      hello
    intent: greet
  - action: utter_greet
  - user: |
      not good
    intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - user: |
      yes
    intent: affirm
  - action: utter_happy

- story: sad path 2
  steps:
  - user: |
```

```
      hello
    intent: greet
  - action: utter_greet
  - user: |
      not good
    intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - user: |
      not really
    intent: deny
  - action: utter_goodbye

- story: sad path 3
  steps:
  - user: |
      hi
    intent: greet
  - action: utter_greet
  - user: |
      very terrible
    intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - user: |
      no
    intent: deny
  - action: utter_goodbye

- story: say goodbye
  steps:
  - user: |
      bye-bye!
    intent: goodbye
  - action: utter_goodbye

- story: bot challenge
  steps:
  - user: |
      are you a bot?
    intent: bot_challenge
  - action: utter_iamabot


--------------------------------------------------

Comment interprétez-vous le contenu de ce fichier ? Comptabilisez
le nombre d'intents, le nombre d'actions sachant que par défaut
l'action_listen est lancée dès que le chatbot est à l'écoute de
l'utilisateur.
```

```
5.2- Résultas de l'évaluation

Depuis le répertoire CHATBOT_<VOTRE_NOM>_V1 exécutez la commande :
rasa test
------------------------------------------------------------------
----------
Copier coller la trace de cette exécution
```

```
------------------------------------------------------------------
----------

Comment interprétez vous cette trace ? Quels liens pouvez-vous
faire avec le contenu du fichier de test ?
```

```
5.3- Analyse des résultats d'évaluation

Consultez le répertoire ./results
```

```
5.3.1- Evaluation des Intents - A
-------------------------------------------------
Ouvrir les fichiers intent_confusion_matrix.png et
intent_histogram.png

A quoi correspondent ces images ? Insérer les dans la cellule
markdown ci-dessous si vous le pouvez.
avec
<img src="intent_confusion_matrix.png" style="height:600px"/>
# adapter en fonction du chemin relatif
ou
Menu Fichier > Insérer Image # essayer de redimensionner
![intent_confusion_matrix.png]
(attachment:intent_confusion_matrix.png)
```

Type *Markdown* and LaTeX: α^2

Type *Markdown* and LaTeX: α^2

```
5.3.2- Evaluation des Intents - B
-------------------------------------------------
Etudier le contenu des fichiers intent_errors.json et
intent_report.json

[
  {
    "text": "good afternoon",
    "intent": "goodbye",
    "intent_prediction": {
      "name": "greet",
      "confidence": 0.3672342598438263
    }
  }
]


{
  "mood_unhappy": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 14,
    "confused_with": {}
```

```
  },
  "bot_challenge": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 4,
    "confused_with": {}
  },
  "affirm": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 6,
    "confused_with": {}
  },
  "greet": {
    "precision": 0.9285714285714286,
    "recall": 1.0,
    "f1-score": 0.962962962962963,
    "support": 13,
    "confused_with": {}
  },
  "mood_great": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 14,
    "confused_with": {}
  },
  "deny": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 7,
    "confused_with": {}
  },
  "goodbye": {
    "precision": 1.0,
    "recall": 0.9090909090909091,
    "f1-score": 0.9523809523809523,
    "support": 11,
    "confused_with": {
      "greet": 1
    }
  },
  "accuracy": 0.9855072463768116,
  "macro avg": {
    "precision": 0.9897959183673469,
    "recall": 0.987012987012987,
    "f1-score": 0.9879062736205594,
    "support": 69
  },
  "weighted avg": {
    "precision": 0.9865424430641822,
    "recall": 0.9855072463768116,
    "f1-score": 0.9854305651407101,
    "support": 69
  }
}
```

```
A quoi correspondent-ils ? Quelles informations peut-on en déduire
?  Vous pouvez consulter internet pour comprendre le rôle des
métriques utilisées, par exemple : https://towardsdatascience.com
/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1
```

```
5.3.3- Evaluation des stories -A
-------------------------------------------------
Copier-coller le contenu du fichier failed_test_stories.yml

-------------------------------------------------
Que concluez-vous ?
```

```
5.3.4- Evaluation des stories -B
-------------------------------------------------
Etudier le contenu du fichier storiy_report.json

{
  "mood_unhappy": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 3
  },
  "utter_cheer_up": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 3
  },
  "action_listen": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 16
  },
  "bot_challenge": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 1
  },
  "utter_did_that_help": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 3
  },
  "affirm": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 1
  },
  "utter_happy": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 3
  },
```

```
  "utter_iamabot": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 1
  },
  "greet": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 5
  },
  "mood_great": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 2
  },
  "utter_goodbye": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 4
  },
  "deny": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 2
  },
  "goodbye": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 2
  },
  "utter_greet": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 5
  },
  "accuracy": 1.0,
  "macro avg": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 51
  },
  "weighted avg": {
    "precision": 1.0,
    "recall": 1.0,
    "f1-score": 1.0,
    "support": 51
  }
}
```


```
--------------------------------------------------
A quoi correspond-il ?
```

```
5.3.5- Evaluation des Intents - C
```

```
------------------------------------------------
Ouvrir le fichier story_confusion_matrix.png

A quoi correspond cette image ? Insérer la dans la cellule markdown
ci-dessous si vous le pouvez.
avec
<img src="story_confusion_matrix.png" style="height:600px"/>
# adapter en fonction du chemin relatif
ou
Menu Fichier > Insérer Image # essayer de redimensionner
![story_confusion_matrix.png]
(attachment:story_confusion_matrix.png)
```

Type *Markdown* and LaTeX: α^2

PARTIE 6 : LANCEMENT DU CHATBOT

```
Depuis le répertoire où se trouve le code du chatbot,

6.1- Lancer le chatbot en mode commande avec

rasa shell

Vous avez le prompt suivant :
Bot loaded. Type a message and press enter (use ‘/stop’ to exit) :
Your input --> …

6.2- Essayer différents cas, copier-coller plusieurs exemples
(pertinents) de dialogue
et commentez leur déroulement.

Ajoutez autant de cellule que d'exemples. Appuyez-vous sur ce que
vous avez retenu des questions
précédentes. L’objectif est de tester les capacités de ce mini-
chatbot et de comprendre ce qui se
passe. Ce qu'il comprend, ce qu'il ne comprend pas, ce qu'il fait,
...
```

```
EXEMPLE DE DIALOGUE 1 et COMMENTAIRE ASSOCIÉ
```

```
EXEMPLE DE DIALOGUE 2 et COMMENTAIRE ASSOCIÉ
```

```
6.3- Lancer le chatbot en mode interactif

Depuis le répertoire où se trouve le code du chatbot, lancer la
commande :

rasa x

--> Ouverture du navigateur et de l’application interactive
permettant d’enrichir le chatbot.
Sélectionner l’onglet en haut à gauche : Talk to your bot
(Interactive Learning), assurez-vous qu'un modèle a été sélectionné
et tapez votre texte dans la zone du bas : "Start typing a message"
```

```
6.4- Essayer différents cas de dialogue (similaires à ceux de la question précédente),
copier-coller la partie story associée dans les zones de texte, une zone par story.
Ajouter des cellules au besoin.
```

```
Story 1 et Commentaire
```

```
Story 2 et Commentaire
```

PARTIE 7 : EVOLUTION DU CHATBOT

```
Faire évoluer la V1 de votre chatbot selon les éléments suivants

7.1- Essayer à nouveau un dialogue :
- en répondant 'Good', 'Fine thanks', 'not well'
- en ajoutant, à la fin, des énoncés utilisateurs du type  'Thanks' / 'Thank you' / 'Many thanks'

Que se passe-t-il du point de vue "intent" ?

Copier-Coller ci-après les "story" correspondantes et commenter.
Que suggérez-vous de faire pour améliorer ?
```

```
Story 1 et commentaire
```

```
Story 2 et commentaire
```

```
7.2- Explorer les possibilités de rasa-x pour ajouter :

- une nouvelle intent 'thank_you',
- une réponse 'utter_welcome' et au moins deux réponses possibles :
    'you are welcome', 'you are very welcome'...
- une règle prévoyant ce type d'échange à la fin de la conversation :
    User : 'thank you' / 'thanks' / 'many thanks'
    Chatbot : 'you are welcome' / 'you are very welcome'
```

```
7.3- Assurez-vous que ces nouvelles informations ont bien été intégrées dans le déroulement des dialogues exécutés par la suite.
Quelle est effectivement l'intent' associée à l'énoncé 'Thank you', avec quel score ? Quelle étape a été nécessaire pour cela.

Copier-coller ici une story obtenue après nouveau dialogue intégrant les remerciements
et expliquer quelles seront selon vous les étapes d'améliorations suivantes.
```

```
Story obtenue après nouveau dialogue intégrant les remerciements
```

```
7.4- Impact sur les données du chatbot modifié via Rasa X

Examiner le fichier
```

```
- data/nlu.yml
- data/stories.yml
- data/rules.yml
- domain.yml
Que constatez vous dans chacun des cas ? Commentez.
```

PARTIE 8 : COMPREHENSION ou NLU - COMMENT CA MARCHE ?

```
Deux packages SPACY ont été chargés lors de l'installation de rasa
en_core_web_md (pour l'anglais) et fr_core_news_md (pour le
français)

8.1- Consultez les site web et expliquer ci-dessous quelles sont
les principales carcatéristiques de ces modèles
```

```
8.2- Exécuter la séquence python suivante.
pprint permettra un affichage indenté du résultat de la partie
compréhension (NLU)
```

In [1]:
```
%matplotlib inline
import logging, io, json, warnings
logging.basicConfig(level="INFO")
warnings.filterwarnings('ignore')

def pprint(o):
    # small helper to make dict dumps a bit prettier
    print(json.dumps(o, indent=2))
```

In [2]:
```
# Positionnez-vous dans le bon répertoire à l'aide du code python su
import os

#PATH = 'Chemin absolu vers votre chatbot V1'
PATH = '/home/sriparole/Documents/SRI3A_TP_RASA/CHATBOT_V0'
os.chdir(PATH)
os.listdir()
```

Out[2]:
```
['rasa.db',
 'tests',
 'domain.yml',
 'data',
 'events.db-wal',
 'endpoints.yml',
 'models',
 'events.db-shm',
 'config.yml',
 'events.db',
 'credentials.yml',
 'actions']
```

```
8.3- Exécuter la séquence python suivante.
```

BIEN VERIFIER QUE VOUS AVEZ LANCE JUPYTER NOTEBOOK DEPUIS L'ENVIRONNEMENT VIRTUEL OU RASA EST INSTALLE (env_rasa2.8)

Les modèles peuvent aussi être utilisés pour traiter les énoncés des utilisateurs à partir de code python, comme cela est illustré dans les séquences python suivante. Le modèle peut aussi être généré directement depuis le code python.

In [4]:
```
import rasa.nlu
import rasa.core
import spacy

print("rasa.nlu: {} rasa.core: {}".format(rasa.nlu.__version__, rasa.
print("Loading spaCy language model...")

print(spacy.load('fr_core_news_md')("Bonjour chez vous! Le modèle fra

print(spacy.load('en_core_web_md')("Hello world! The English model ha
```

```
rasa.nlu: 2.8.0 rasa.core: 2.8.0
Loading spaCy language model...
Bonjour chez vous! Le modèle français est chargé
Hello world! The English model has ben loaded
```

8.4- Exécuter la séquence python suivante.
BIEN VERIFIER QUE VOUS AVEZ LANCE JUPYTER NOTEBOOK DEPUIS L'ENVIRONNEMENT VIRTUEL OU RASA EST INSTALLE (env_rasa2.8)

La partie nlu peut être utilisée dans du code python pour développer ses propres applications.
Exécuter le code suivant. Il faut construire un interpréteur qui correspond à la partie pipeline

In [7]:
```
import os
from rasa.shared.nlu.training_data.loading import load_data
from rasa.nlu.config import RasaNLUModelConfig
from rasa.nlu.model import Trainer
from rasa.nlu import config
from os import system

# Adapter le chemin à votre contexte
#PATH = 'chemon absolu vers votre chatbot V1'
#os.chdir(PATH)
os.listdir()
#Compléter avec le chemin vers votre environnement virtuel créé lors
print(os.getcwd())
print(os.listdir('data'))
warnings.filterwarnings('ignore')

# loading the nlu training samples
training_data = load_data("./data/nlu.yml")
# trainer to educate our pipeline
trainer = Trainer(config.load("./config.yml"))
# train the model!
interpreter = trainer.train(training_data)
# store it for future use
model_directory = trainer.persist("./models/nlu", fixed_model_name="c
```

```
INFO:rasa.nlu.model:Starting to train component WhitespaceTokenizer
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component RegexFeaturizer
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component LexicalSyntacticFea
turizer
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component CountVectorsFeaturi
zer
INFO:rasa.nlu.featurizers.sparse_featurizer.count_vectors_featurize
r:80 vocabulary items were created for text attribute.
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component CountVectorsFeaturi
zer
INFO:rasa.nlu.featurizers.sparse_featurizer.count_vectors_featurize
r:697 vocabulary items were created for text attribute.
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component DIETClassifier

/home/sriparole/Documents/SRI3A_TP_RASA/CHATBOT_V0
```

8.5- Commenter la trace obtenue et aller voir ce qu'il y a sous models/nlu/current en exécutant le code suivant

In [8]:
```
#Compléter avec le chemin vers votre environnement virtuel créé lors
print(os.getcwd())
print(os.listdir('models/nlu/current'))
warnings.filterwarnings('ignore')
```

```
/home/sriparole/Documents/SRI3A_TP_RASA/CHATBOT_V0
['metadata.json', 'component_5_DIETClassifier.tf_model.index', 'com
ponent_2_LexicalSyntacticFeaturizer.feature_to_idx_dict.pkl', 'chec
kpoint', 'component_5_DIETClassifier.entity_tag_specs.json', 'compo
nent_1_RegexFeaturizer.patterns.pkl', 'component_5_DIETClassifier.i
ndex_label_id_mapping.json', 'component_5_DIETClassifier.label_dat
a.pkl', 'component_5_DIETClassifier.sparse_feature_sizes.pkl', 'com
ponent_5_DIETClassifier.tf_model.data-00000-of-00001', 'component_3
_CountVectorsFeaturizer.pkl', 'component_5_DIETClassifier.data_exam
ple.pkl', 'component_4_CountVectorsFeaturizer.pkl']
```

8.6- Exécuter le code suivant en remplaçant la chaîne de caractères vide "" par la
phrase à analyser représentant l'énoncé de l'utilisateur :
- I am fine thanks
- Thank you
- Thank you very much
- I would like to thank you
- I would rather prefer a puppy
- I would rather have prefered a puppy
- I would like to thank you very much for this

Vous pouvez également proposer des phrases à traiter.

ATTENTION : DUPLIQUER LA CELLULE A CHAQUE NOUVEL ESSAI POUR GARDER TRACE DES RESULTATS.

In [9]:
```
pprint(interpreter.parse(""))
```

```
{
  "text": "",
  "intent": {
    "name": null,
    "confidence": 0.0
  },
  "entities": []
}
```

In [ ]:
```
pprint(interpreter.parse("I am fine thanks"))
```

In [ ]:
```
pprint(interpreter.parse("Thank you"))
```

In [ ]:
```
pprint(interpreter.parse("I would prefer a puppy"))
```

In [ ]:
```
pprint(interpreter.parse("I would have prefered a puppy"))
```

In [ ]:
```
pprint(interpreter.parse("I would like to thank you very much for thi
```

Commentaires sur les résultats obtenus

PARTIE 9 : Evolution du chatbot (avancée)

```
Intégration de nouvelles intents et d'entités

9.1- On veut traiter des phrases comme :

- I would like to see another animal
- Do you have other animals to show
- I want to see another picture

Avec une réponse du type : I don't have any
Avec une réponse du type : Here is another one (avec affichage
d'une autre image).

Quelles sont les nouvelles intents à définir ?
```

```
9.2- Que faut-il faire de plus pour prendre en compte des phrases
comme :

- I would like to see a puppy
- Do you have any pictures of butterflies
- I want to see a koala

Avec une réponse du type : Let me see ... I found this

en affichant une image prise sur internet ou disponible en local
relative à l'animal mentionné
```

```
9.3- Créer une nouvelle version V2 (pour éviter d'écraser ce qui
marche) et intégrer les modifications évoquées dans la questions
précédente.
-------------
Indiquer quelles sont les modifications apportées. Fichier par
fichier concernés par ces modifications
-------------
```

```
9.4- Illustrer en donnant des exemples de dialogue incluant ce type
d'échanges

- User : I would like to see a puppy
- Chatbot : Let me see ... I found this ... Does it help ?
```

```
Exemple 1
```

```
Exemple 2
```

```
9.5- Faire évoluer votre chatbot sur les mêmes principes. Commenter
et illustrer ces ajouts (Demander quel est votre animal préféré,
ajouter une table lookup pour gérer les synonymes ... ou ajouter
d'autres règles
```

```
Evolution de votre chatbot et Commentaires associés
```

```
9.6- A partir de rasa x et l'onglet stories, observer les modèles
de dialogue (graphes) associés aux différentes stories.

Faire une capture écran de l'un des graphes les plus représentatifs
et insérer le dans la cellue suivante
```

Type *Markdown* and LaTeX: α^2

```
9.7- Visualisation

Quitter rasa x en fermant la fenêtre du navigateur ET en tapant ^C
dans le terminal de lancement.

Une fois rasa x attrêté, tapez la commande : rasa visualize

- Visualisez le fichier Graph.html
- Insérer dans la cellule suivante une copie image de ce graphe
(copie écran ou autre)

Que représente-t-il ?
```

Type *Markdown* and LaTeX: α^2

PARTIE 10 : EVALUATION DE VOTRE CHATBOT V2

```
- Evaluation de la nouvelle version
--------------------------------------

10.1- Constituer un fichier de test adapté à ce nouveau chatbot et
reprendre les questions de la partie 2 - Q7  pour visualiser et
commenter les résultats obtenus.
```

```
10.2- Faire un pdf de votre notebook, déposer une archive contenant
votre chatbot CHATBOT_<VOTRE_NOM>_V1 ainsi que la version ipynb et
la version pfd de votre notebook.
```