

# 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 ?

INTENTS:

- greet
- goodbye

```
-affirm
-deny
-mood_great
-mood_unhappy
-bot_challenge
```

#### ENTITES

Il n'y a pas

S'il y auraiit, elles serait en face comme

[mot/groupe de mots](entity name)

#### 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 ?

1) Definition du langage EN english

2) Pipeline = chaine de traitement NLU

Elle est configuré par défaut. Elle sert à traiter l'enoncé de l'utilisateur pour en extraire l'intention et les entités. Il applique une succession d'outils NLP pour transformer les énoncés et obtenir une représentation exploitable. Rasa utilise Spacy

Il definit le ngram de 1 à 4 maximum.  
Le nombre d'epochs pour l'entraînement  
Le threshold de 0.3  
Le threshold d'ambigüeté de 0.1  
Tokenizer les espace en blanc

3) Policies = different strategies possibles

Analyser ce qui dit l'utilisateur = partie NLU  
Décider la prochaine action = partie CORE

Il determine  
-MemorizationPolicy  
-TEDPolicy

Il determine le max de l'historique de 5 et les epochs de 100  
-RulePolicy

A score de confiance égale la politique la plus priorité s'applique  
Rules > ML > Transformers

## 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 ?

Le rôle des STORIES est de donner des exemples de scénarios. L'enchaînement d'interventions utilisateur (intent) / système(action)

#### STORIES

- happy path
- sad path 1
- sad path 2

utter\_X = réponse X (actions)

Il donne 3 scénarios possibles:

- 1) happy
- 2) essayer de changer à mood happy et réussir
- 3) essayer de changer à mood happy et échouer

#### 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 ?

Modélisation des échanges figés

Il y a 2 règles

- goodbye
- I am a bot

Le rôle est de répondre selon l'intent de l'utilisateur avec une action définie. Ces règles servent dans la partie de NLU et elles sont prioritaires face à ML et Transformers

## 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 ?

C'est le récapitulatif de l'univers du chatbot.

intents:
    Ce les differenrs intents que l'utilisateur peut réaliser
    (exemples dans le fichier nlu.yml)

responses :
    Les différentes réponses du chatbot. Elles peut être text ou
    image

session_config:
    (par défaut) Expire dans 60 minutes.

```

## 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 ?



Pour le moment il est commenté mais il sert à réaliser des actions spécifiques.  
Par exemple on peut utiliser les actions pour donner des réponses différentes selon l'entrée de l'utilisateur (customs actions) ou faire un comportement spécifique.

Il y a 2 choses importantes, le tracker et le dispatcher, tout les deux viennent de rasa\_sdk et ça sert par exemple  
tracker - récupérer des slots ou entités  
dispatcher - afficher la réponse avec le return

## 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
```

```
# désactivation 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

Si je change le nom de l'ancien fichier :

The configuration for pipeline and policies was chosen automatically. It was written into the config file at 'config.yml'. Nothing changed. You can use the old model stored at '/home/zamayra/Bureau/CHATBOT\_HERNANDEZ\_V1/models/model\_0.tar.gz'.

-----  
----

Si je supprime le model:

The configuration for pipeline and policies was chosen automatically. It was written into the config file at 'config.yml'. Nothing changed. You can use the old model stored at '/home/zamayra/Bureau/CHATBOT\_HERNANDEZ\_V1/models/model\_0.tar.gz'.

(rasa2\_2021) zamayra@zamayra-IdeaPad-Flex-5-15ALC05:~/Bureau/CHATBOT\_HERNANDEZ\_V1\$ rasa train

The configuration for pipeline and policies was chosen automatically. It was written into the config file at 'config.yml'. Training NLU model...

2023-01-03 09:27:51 INFO

rasa.shared.nlu.training\_data.training\_data - Training data stats:

2023-01-03 09:27:51 INFO

rasa.shared.nlu.training\_data.training\_data - Number of intent examples: 69 (7 distinct intents)

2023-01-03 09:27:51 INFO

rasa.shared.nlu.training\_data.training\_data - Found intents: 'bot\_challenge', 'deny', 'mood\_unhappy', 'greet', 'mood\_great', 'affirm', 'goodbye'

2023-01-03 09:27:51 INFO

rasa.shared.nlu.training\_data.training\_data - Number of response examples: 0 (0 distinct responses)

```

2023-01-03 09:27:51 INFO
rasa.shared.nlu.training_data.training_data - Number of entity
examples: 0 (0 distinct entities)
2023-01-03 09:27:51 INFO      rasa.nlu.model - Starting to train
component WhitespaceTokenizer
2023-01-03 09:27:52 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:27:52 INFO      rasa.nlu.model - Starting to train
component RegexFeaturizer
2023-01-03 09:27:52 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:27:52 INFO      rasa.nlu.model - Starting to train
component LexicalSyntacticFeaturizer
2023-01-03 09:27:52 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:27:52 INFO      rasa.nlu.model - Starting to train
component CountVectorsFeaturizer
2023-01-03 09:27:52 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:27:52 INFO      rasa.nlu.model - Starting to train
component CountVectorsFeaturizer
2023-01-03 09:27:52 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:27:52 INFO      rasa.nlu.model - Starting to train
component DIETClassifier
Epochs: 100%|██████| 100/100 [00:11<00:00, 8.83it/s, t_loss=1.176,
i_acc=0.986]
2023-01-03 09:28:09 INFO      rasa.utils.tensorflow.models -
Finished training.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Starting to train
component EntitySynonymMapper
2023-01-03 09:28:09 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Starting to train
component ResponseSelector
2023-01-03 09:28:09 INFO      rasa.nlu.selectors.response_selector
- Retrieval intent parameter was left to its default value. This
response selector will be trained on training examples combining
all retrieval intents.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Starting to train
component FallbackClassifier
2023-01-03 09:28:09 INFO      rasa.nlu.model - Finished training
component.
2023-01-03 09:28:09 INFO      rasa.nlu.model - Successfully saved
model into '/tmp/tmpnwnij98/nlu'
NLU model training completed.
Training Core model...
Processed story blocks: 100%|██████| 3/3 [00:00<00:00, 4780.74it/s,
# trackers=1]
Processed story blocks: 100%|██████| 3/3 [00:00<00:00, 2326.29it/s,
# trackers=3]
Processed story blocks: 100%|██████| 3/3 [00:00<00:00, 498.29it/s, #
trackers=12]
Processed story blocks: 100%|██████| 3/3 [00:00<00:00, 155.52it/s, #
trackers=39]

```

```

Processed rules: 100%|██████████| 2/2 [00:00<00:00, 6647.07it/s,
# trackers=1]
Processed trackers: 100%|██████████| 3/3 [00:00<00:00, 2109.81it/s,
# actions=12]
Processed actions: 12it [00:00, 19614.83it/s, # examples=12]
Processed trackers: 100%|██████████| 120/120 [00:00<00:00, 609.49it/s,
# actions=30]
Epochs: 100%|██████████| 100/100 [00:04<00:00, 22.81it/s, t_loss=8.062,
loss=7.896, acc=1
2023-01-03 09:28:20 INFO      rasa.utils.tensorflow.models -
Finished training.
Processed trackers: 100%|██████████| 2/2 [00:00<00:00, 3408.62it/s,
# actions=5]
Processed actions: 5it [00:00, 29495.81it/s, # examples=4]
Processed trackers: 100%|██████████| 3/3 [00:00<00:00, 2074.33it/s,
# actions=12]
Processed trackers: 100%|██████████| 2/2 [00:00<00:00,
4148.67it/s]
Processed trackers: 100%|██████████| 5/5 [00:00<00:00,
1790.75it/s]
2023-01-03 09:28:20 INFO      rasa.core.agent - Persisted model to
'/tmp/tmpnwnij98/core'
Core model training completed.
Your Rasa model is trained and saved at '/home/zamayra/Bureau
/CHATBOT_HERNANDEZ_V1/models/20230103-092822.tar.gz'.

```

-----

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

Il entraîne le modèle. Il met la configuration par défaut.  
 Il y a 68 exemples d'intent et 7 intents distincts.  
 0 entities  
 Lancement de la pipeline NLU traitement  
 Training avec x epochs  
     NLU model ( ce que dit l'utilisateur)  
     Core model ( la prochaine action)

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

Il contient le modèle entraîné qu'on vient de réaliser

## 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.

C'est les différents scénarios STORIES (7) avec lesquels on va évaluer les performances de notre modèle

## 5.2- Résultats de l'évaluation

Depuis le répertoire CHATBOT\_<VOTRE\_NOM>\_V1 exécutez la commande :

```
rasa test
```

-----

Copier coller la trace de cette exécution

```

2023-01-03 09:38:15 INFO      rasa.shared.utils.validation - The
'version' key is missing in the training data file /tmp/tmpjv4x6pkk
/887338a180d14d0496168771b4a10ddf_test_stories.yml. Rasa Open
Source will read the file as a version '2.0' file. See
https://rasa.com/docs/rasa/training-data-format.
Processed story blocks: 100%|██████████| 7/7 [00:00<00:00, 5989.42it/s,
# trackers=1]
2023-01-03 09:38:15 INFO      rasa.core.test - Evaluating 7 stories
Progress:
100%|██████████████████████████████████████████████████████████████| 7/7
[00:00<00:00, 17.98it/s]
2023-01-03 09:38:16 INFO      rasa.core.test - Finished collecting
predictions.

```

```

2023-01-03 09:38:16 INFO rasa.core.test - Evaluation Results
on END-T0-END level:
2023-01-03 09:38:16 INFO rasa.core.test - Correct: 7 / 7
2023-01-03 09:38:16 INFO rasa.core.test - F1-Score: 1.000
2023-01-03 09:38:16 INFO rasa.core.test - Precision: 1.000
2023-01-03 09:38:16 INFO rasa.core.test - Accuracy: 1.000
2023-01-03 09:38:16 INFO rasa.core.test - In-data fraction: 0.829
2023-01-03 09:38:16 INFO rasa.core.test - Stories report saved to results/story_report.json.
2023-01-03 09:38:17 INFO rasa.core.test - Evaluation Results on ACTION level:
2023-01-03 09:38:17 INFO rasa.core.test - Correct: 35 / 35
2023-01-03 09:38:17 INFO rasa.core.test - F1-Score: 1.000
2023-01-03 09:38:17 INFO rasa.core.test - Precision: 1.000
2023-01-03 09:38:17 INFO rasa.core.test - Accuracy: 1.000
2023-01-03 09:38:17 INFO rasa.core.test - In-data fraction: 0.829
2023-01-03 09:38:17 INFO rasa.utils.plotting - Confusion matrix, without normalization:
[[16  0  0  0  0  0  0]
 [ 0  3  0  0  0  0  0]
 [ 0  0  3  0  0  0  0]
 [ 0  0  0  4  0  0  0]
 [ 0  0  0  0  5  0  0]
 [ 0  0  0  0  0  3  0]
 [ 0  0  0  0  0  0  1]]
2023-01-03 09:38:19 INFO rasa.nlu.test - Running model for predictions:
100%|██████████████████████████████████████| 69/69
[00:00<00:00, 179.73it/s]
2023-01-03 09:38:19 INFO rasa.nlu.test - Intent evaluation results:
2023-01-03 09:38:19 INFO rasa.nlu.test - Intent Evaluation: Only considering those 69 examples that have a defined intent out of 69 examples.
2023-01-03 09:38:19 INFO rasa.nlu.test - Classification report saved to results/intent_report.json.
2023-01-03 09:38:19 INFO rasa.nlu.test - Incorrect intent predictions saved to results/intent_errors.json.
2023-01-03 09:38:19 INFO rasa.utils.plotting - Confusion matrix, without normalization:
[[ 6  0  0  0  0  0  0  0]
 [ 0  4  0  0  0  0  0  0]
 [ 0  0  7  0  0  0  0  0]
 [ 0  0  0 10  0  0  0  1]
 [ 0  0  0  0 12  0  0  1]
 [ 0  0  0  0  0 14  0  0]
 [ 0  0  0  0  0  0 14  0]
 [ 0  0  0  0  0  0  0 11]]

```

-----  
-----

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

Il a réalisé l'évaluation avec les 7 stories que contenait le fichier de test. CORE

Dans la prédiction, il y a eu 2 erreurs de la ligne 7 avec la ligne 4 et 5. Analyse des résultats d'évaluation

Consultez le répertoire ./results

Il y a eu 69 supports (actions et intents du fichier test)

Les deux erreurs sont liées avec 'good afternoon' qui ont l'intent greet ou good bye.

Comme ces 2 mots sont en dessous du seuil de 0.3, il va donner nlu\_fallback avec une confiance de 1.

### 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

``

# adapter en fonction du chemin relatif

ou

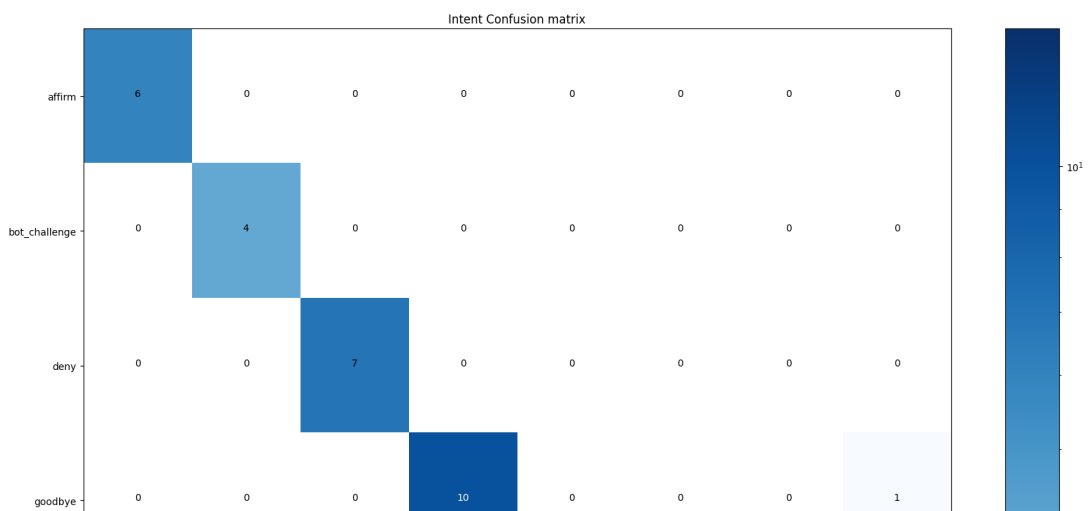
Menu Fichier > Insérer Image # essayer de redimensionner

`![intent_confusion_matrix.png]`

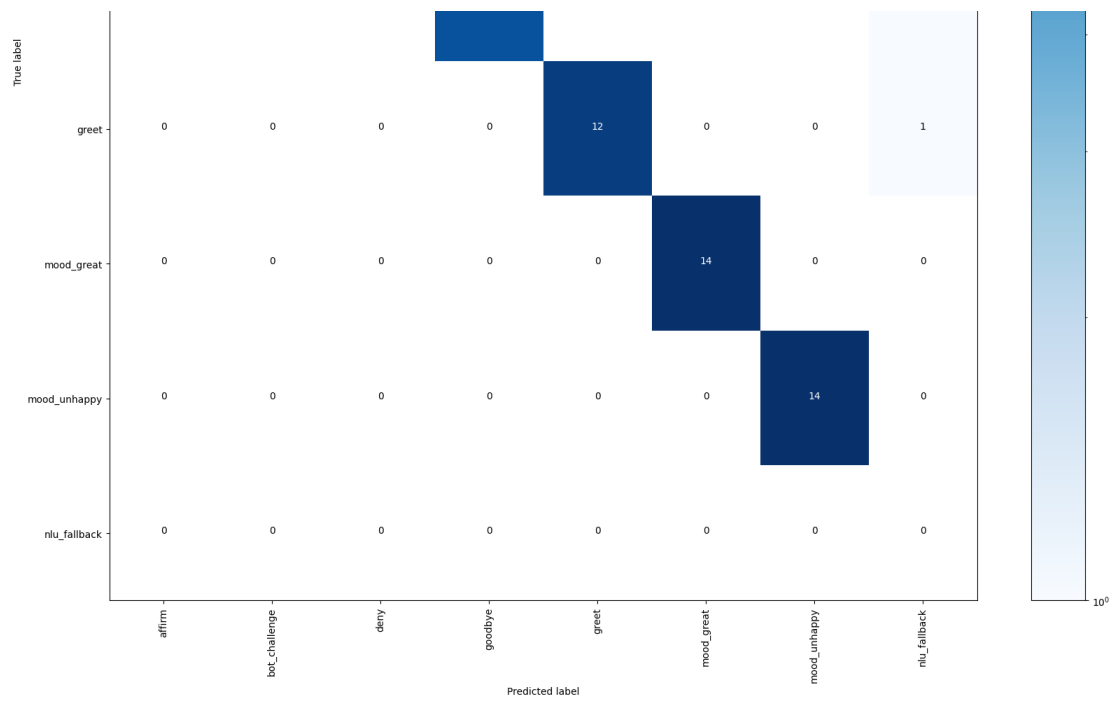
`(attachment:intent_confusion_matrix.png)`

### INTENT CONFUSION MATRIX

Elle montre les erreurs/reussites du test. Dans la matrice s'observe 2 erreurs pour l'intent greet et goodbye. Comme la prédiction était en dessous de 0.3 (threshold), elle est classée comme nlu\_fallback (voir fichier config)

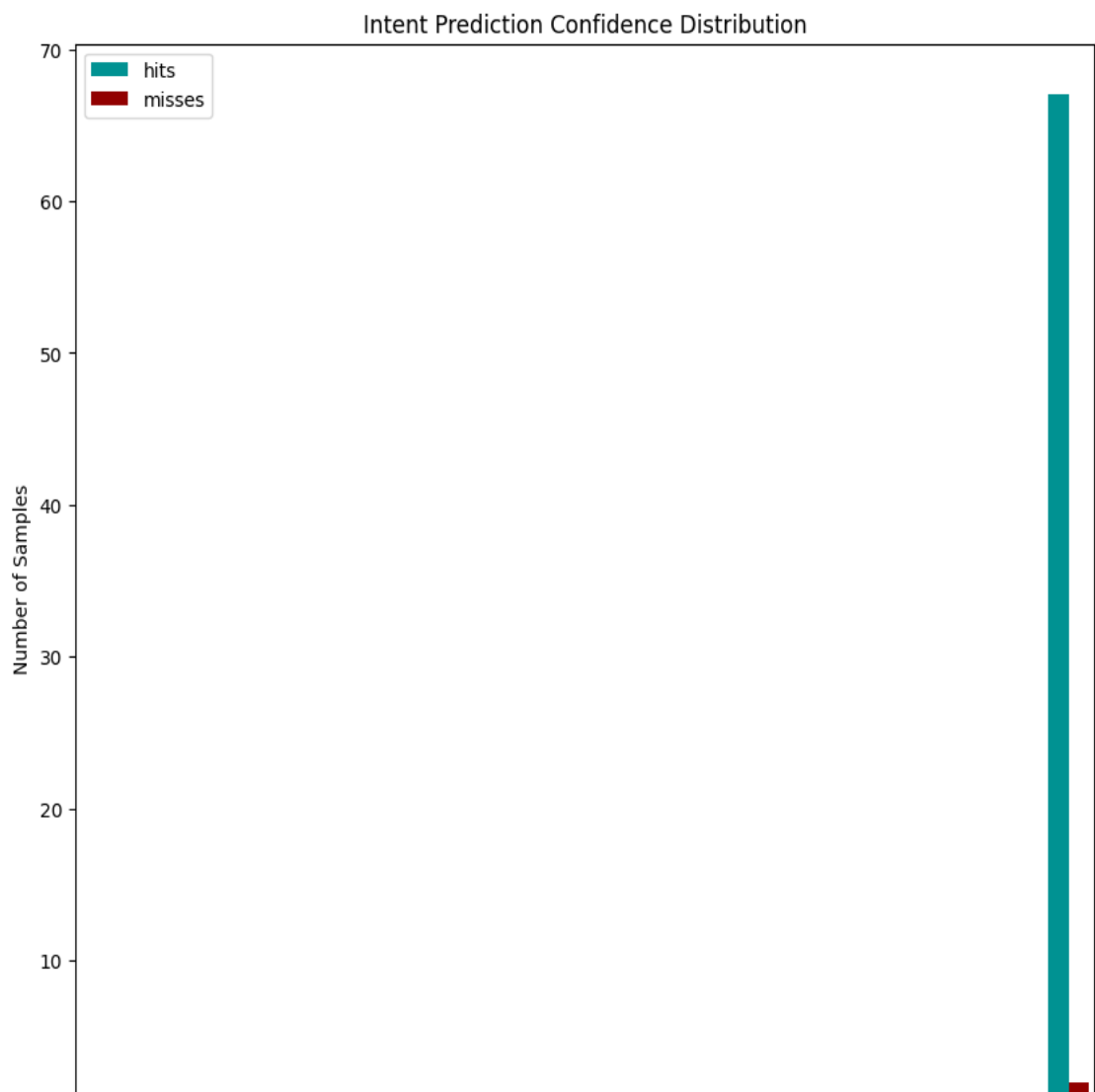


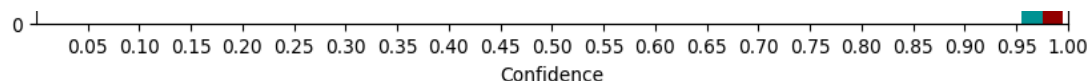




### INTENT\_HISTOGRAM

L'histogramme montre la réussite du chatbot. Du total des samples (69) il a eu 67 reussites avec plus de 0.95 et 2 erreurs à 1 avec l'etiquette de nlu\_fallback





### 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>

-intent\_errors.json

Il donne les intent où la prédiction est différent de la vraie étiquette.

Il nous donne le texte d'entrée, l'intent correct, et l'intent de prédiction.

Dans ce dernier il donne le nom de l'intent qu'il a prédit et la confiance.

-intent\_report.json

Pour chaque intent, il donne les informations de précision

recall - what proportion of actual Positives is correctly classified?  $\text{recall} = \text{TP} / (\text{TP} + \text{FN})$

$\text{f1-score} = 2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$

support - combien de samples

confused\_with : l'intent qui a été confondu.

### 5.3.3- Evaluation des stories -A

-----

```
Copier-coller le contenu du fichier failed_test_stories.yml
```

```
# None of the test stories failed - all good!
```

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

Toutes les stories de test ont été satisfés, elles se sont bien déroulé.

#### 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 ?

Il donne la précision, recall, f1-score et support de chaque parti de STORIES (intent + action).

L'action listen apparait aussi, elle n'est pas dans test\_stories mais elle est présent au début, car le chatbot est toujours à l'écoute, elle est implicite.

Il donne de manière général l'accuracy et le weighted\_avg

#### 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

```

```

# adapter en fonction du chemin relatif

ou

Menu Fichier > Insérer Image # essayer de redimensionner

```
![story_confusion_matrix.png]
```

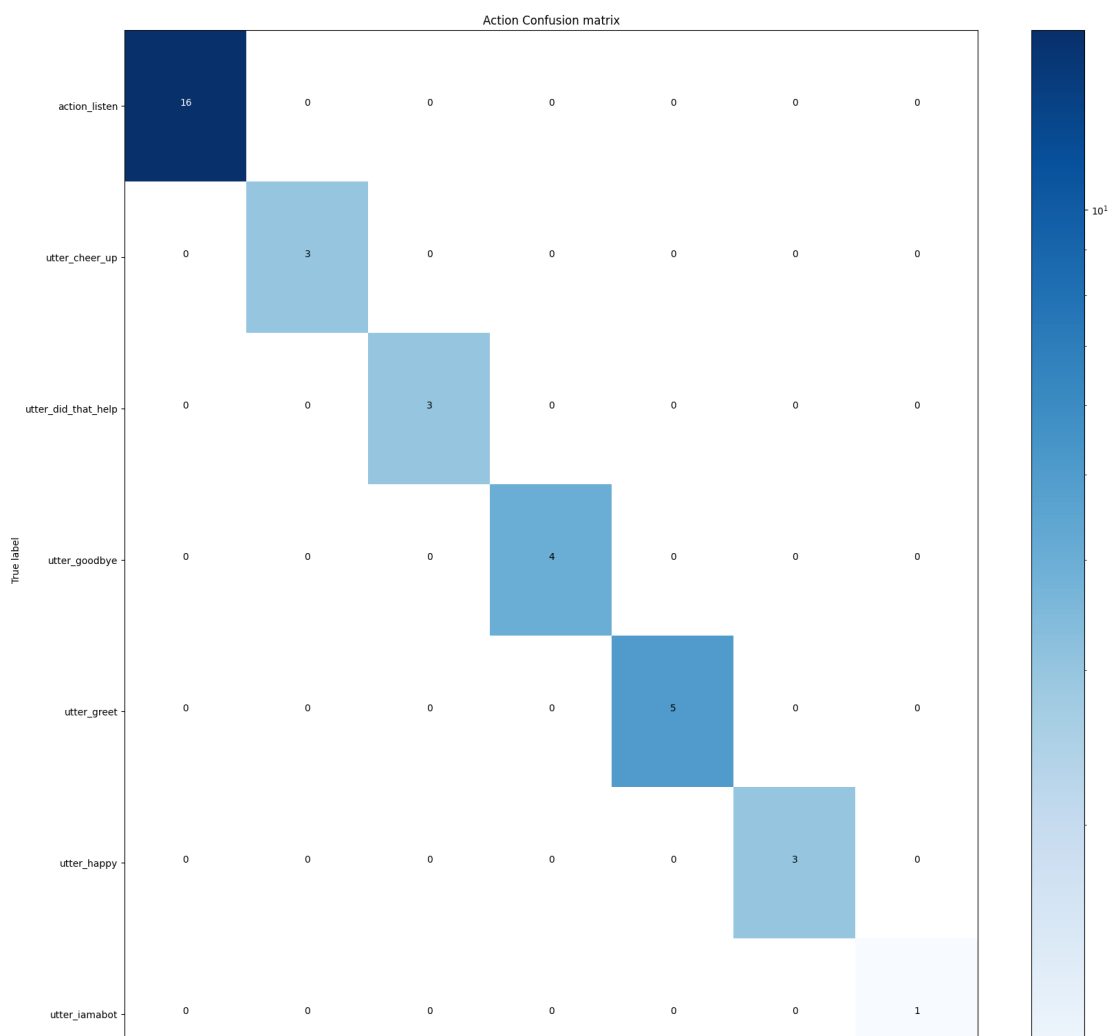
```
(attachment:story_confusion_matrix.png)
```

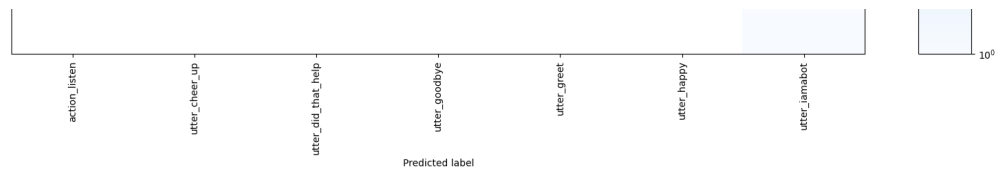
### STORY CONFUSION MATRIX

Elle correspond à la matrice de confusion des actions présents dans le fichier test\_stories.

Il n'y a pas eu des erreurs dans les actions prédictes et les vrais actions.

L'action\_listen apparait avec plus des samples car le chatbot est toujours à l'ecoute (16 fois dans le fichier test\_stories -à chaque fois de user)





## 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

Your input -> hi

Hey! How are you?

Your input -> perfect

Great, carry on!

Your input -> bye

Bye

### COMMENTAIRE ASSOCIÉ

Déroulement mood happy

### EXEMPLE DE DIALOGUE 2

Your input -> hello

Hey! How are you?

Your input -> I am sad

Here is something to cheer you up:

Image: <https://i.imgur.com/nGF1K8f.jpg>

Did that help you?

Your input -> yes

Great, carry on!

COMMENTAIRE ASSOCIÉ

Déroulement sad to happy

#### EXEMPLE DE DIALOGUE 3

Your input -> I'm so sad

Here is something to cheer you up:  
Image: <https://i.imgur.com/nGF1K8f.jpg>  
Did that help you?

Your input -> no

Your input -> not really

Your input -> yes

Great, carry on!

COMMENTAIRE ASSOCIÉ

Déroulement juste après du dialogue 2. Il ne répond pas après une réponse négative.

#### EXEMPLE DE DIALOGUE 4

Your input -> are you a bot?

I am a bot, powered by Rasa.

Your input -> are you a human?

I am a bot, powered by Rasa.

COMMENTAIRE ASSOCIÉ

Il répond bien aux questions

#### EXEMPLE DE DIALOGUE 5

Your input -> hi

Hey! How are you?

Your input -> disappointed

Here is something to cheer you up:  
Image: <https://i.imgur.com/nGF1K8f.jpg>  
Did that help you?

Your input -> no

Bye

COMMENTAIRE ASSOCIÉ



Bon réponse si déroulement dans l'ordre greet - mood -bye

#### EXEMPLE DE DIALOGUE 6

Your input -> I am very happy

Here is something to cheer you up:

Image: <https://i.imgur.com/nGF1K8f.jpg>

Did that help you?

Your input -> yes

Great, carry on!

Your input -> I am very good

Here is something to cheer you up:

Image: <https://i.imgur.com/nGF1K8f.jpg>

Did that help you?

Your input -> I am so good

Great, carry on!

#### COMMENTAIRE ASSOCIÉ

happy n'est pas dans le mood great, mais I am very est dans le mood unhappy, alors il donne la réponse pour un mood\_unhappy

I am so good, les 4 mots se trouve dans le mood\_greet, alors il se trompe pas et il répond selon le mood\_greet.

#### 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

version: "2.0"

stories:

- story: Story from Conversation ID  
f0ea7013259c4a49aecd56d532d23816

steps:

- intent: greet

- action: utter\_greet
- intent: mood\_great
- action: utter\_happy
- intent: bot\_challenge
- action: utter\_iamabot
- intent: greet
- action: utter\_goodbye

Commentaire : Ajout de story car le chatbot s'est trompé dans good evening à la fin (confondu avec greet)

### Story 2

version: "2.0"

stories:

- story: Story from Conversation ID  
fa788b7fde16487ab195f75d6114d477

steps:

- intent: greet
- action: utter\_greet
- intent: mood\_great
- action: utter\_happy
- intent: goodbye
- action: utter\_goodbye

Commentaire : ajoute de story car le chatboot s'est trompé avec im fine (classé comme deny au lieu de mood\_happy)

### Story 3

version: "2.0"

stories:

- story: Story from Conversation ID  
7bbb4009fdb440f4bebe0e5b18f4f832

steps:

- intent: greet
- action: utter\_greet

- intent: mood\_great
- action: utter\_happy
- intent: goodbye
- action: utter\_goodbye

Commentaire : ajout de story car good evening classé comme greet au lieu de goodbye

## 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

```
version: "2.0"
stories:
- story: Story from Conversation ID
2ae0d40adf4444f7a00f0589a007c0fd
  steps:
  - intent: greet
  - action: utter_greet
  - intent: goodbye
  - action: utter_goodbye
```

### Commentaire

Intent goodbye car Good se trouve plus de fois dans goodbye que dans l'intent mood\_great.  
Ajoute mot Good dans intent mood\_great

### Story 2

```
version: "2.0"
stories:
- story: Story from Conversation ID
bc9b39bcfd194c93b83905a12a3cfe17
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_great
  - action: utter_happy
  - intent: goodbye
  - action: utter_goodbye
```

### Commentaire

Fine thanks est reconnu comme mood\_great car le model avait été entraîné avec le mot fine auparavant

Good, thank you est reconnu comme goodbye au lieu de mood\_happy

Good many thanks est reconnu comme action\_default\_fallback (0.30) au lieu de mood\_happy

### Story 3

version: "2.0"

stories:

- story: Story from Conversation ID  
a4002f1e5ca14afb8dabc3bd245b7958

steps:

- intent: greet
- action: utter\_greet
- intent: mood\_unhappy
- action: utter\_cheer\_up
- action: utter\_did\_that\_help
- intent: affirm
- action: utter\_happy

Commentaire : not well est reconnu comme mood\_unhappy car not et well se trouvent dans mood\_unhappy.

Ajout de not well quand même à la liste de mood\_unhappy pour éviter d'erreurs après.

### Story 4

version: "2.0"

stories:

- story: Story from Conversation ID  
e90520639e21407982938ee6f152ed09

steps:

- intent: greet
- action: utter\_greet
- intent: mood\_unhappy
- action: utter\_cheer\_up
- action: utter\_did\_that\_help
- intent: affirm
- action: utter\_happy

Commentaire : L'ajout de thank you, many thanks et thanks à la fin de utter\_did\_that\_help n'affecte pas l'action suivant du chatbot, Même si on ajoute bye, thank you, le chatbot reconnaît l'action correct à 0.9 de précision.

L'intent peut changer si on ajoute d'autres mots de politesse à la réponse par défaut.

On pourrait ajouter un intent de politesse qui serait inclus comme optionnel dans les réponses de l'utilisateur.

## 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'

### Story

```
version: "2.0"
stories:
- story: Story from Conversation ID
641f823c082541e9bf85ae8d33d725a7
  steps:
    - intent: greet
    - action: utter_greet
    - intent: mood_great
    - action: utter_happy
    - intent: thank_you
    - action: utter_welcome
```

#### Commentaires :

Il faut ajouter des stories pour entrainer le chatbot.

Il y a eu un erreur avec fine thank you ( réponse correct dans le modèle précédent) il a donné action\_default\_fallback(0.30).

On pourrait utiliser l'intent thank\_you a la fin de chaque sentence.

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.

J'ai entrené le model à chaque modification et je l'ai activé depuis la section models  
L'intent est thank\_you avec un score de 1

```
version: "2.0"
stories:
- story: Story from Conversation ID
c80e6b236c4d42758b97fe2c08832242
  steps:
    - intent: greet
    - action: utter_greet
    - intent: mood_unhappy
    - action: utter_cheer_up
    - action: utter_did_that_help
```

```
- intent: affirm
- action: utter_happy
- intent: thank_you
- action: utter_welcome
```

#### 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.

NLU

ajout du nouvel intent thank\_you et action welcome

STORIES

Le fichier reste inchangé

RULES

Ajout des tous les stories faits depuis RASA X et ajout de la nouvelle rule intent : thank\_you, action : welcome

DOMAIN

Ajout du nouvel intent et action

**\*\*Il a eu une modification manuel pour le fichier STORIES et RULES car dans cette version de rasa, quand on sauvegarde les stories depuis rasa x, il envoie tout (stories et rules) sur le même fichier RULES. Manuellement il y eu une modification de ce fichier pour avoir à part les stories et les rules.**

## 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 caractéristiques de ces modèles

en\_core\_web\_md:

EN english

CORE Vocabulary, syntax, entities, vectors

WEB written text (blogs, news, comments)

MD 40 MB

fr\_core\_news\_md :

FR French

CORE Vocabulary, syntax, entities, vectors

NEWS written text (news, media)  
MD 43 MB

Ils sont des modèles de taille moyenne MD et de langage english or french.

Ils sont des pipelines de prétraitement NLP

Transforment une entrée Text en un objet NLP python Doc

8.2- Exécuter la séquence python suivante.

pprint permettra un affichage indenté du résultat de la partie compréhension (NLU)

```
Entrée [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))
```

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

PATH = '/home/zamayra/Bureau/CHATBOT_HERNANDEZ_V1'
os.chdir(PATH)
os.listdir()
```

```
Out[2]: ['TP_RASA_2023.ipynb',
'actions',
'rasa.db',
'config.yml',
'events.db-shm',
'data',
'domain.yml',
'events.db-wal',
'results',
'events.db',
'tests',
'credentials.yml',
'.ipynb_checkpoints',
'endpoints.yml',
'models']
```

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.

```
Entrée [8]: !pwd
```

```
/home/zamayra/Bureau/CHATBOT_HERNANDEZ_V1
```

Entrée [10]: `!pip3 install rasa==2.0.2`

```
Requirement already satisfied: rasa==2.0.2 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (2.0.2)
Requirement already satisfied: absl-py<0.11,>=0.9 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (0.10.0)
Requirement already satisfied: python-dateutil<2.9,>=2.8 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (2.8.2)
Requirement already satisfied: rasa-sdk<3.0.0,>=2.0.0 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (2.8.6)
Requirement already satisfied: tensorflow<2.4,>=2.3 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (2.3.4)
Requirement already satisfied: terminaltables<3.2.0,>=3.1.0 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (3.1.10)
Requirement already satisfied: mattermostwrapper<2.3,>=2.2 in /home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages (from rasa==2.0.2) (2.2.0)
```

Entrée [1]: `!rasa --version`

```
Rasa Version      : 2.0.2
Rasa SDK Version  : 2.8.6
Rasa X Version    : 0.33.0
Python Version    : 3.7.15
Operating System  : Linux-5.15.0-56-generic-x86_64-with-debian-bookworm-sid
Python Path       : /home/zamayra/anaconda3/envs/rasa2_2021/bin/python
```

Entrée [15]: `rasa.nlu`

Out[15]: `<module 'rasa.nlu' from '/home/zamayra/.local/lib/python3.10/site-packages/rasa/nlu/__init__.py'>`

Entrée [13]: `rasa.nlu`

Out[13]: `<module 'rasa.nlu' from '/home/zamayra/anaconda3/envs/rasa2_2021/lib/python3.7/site-packages/rasa/nlu/__init__.py'>`



```
Entrée [3]: 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')("Bonjour chez vous! Le modèle français est cl
print(spacy.load('fr_core_news_md')("Bonjour chez vous! Le modèle fra
#print(spacy.load('en')("Hello world! The English model has been loa
print(spacy.load('en_core_web_md')("Hello world! The English model ha

rasa.nlu: 2.0.2 rasa.core: 2.0.2
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

Entrée [4]:

```

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 = '/home/zamayra/Bureau/CHATBOT_HERNANDEZ_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

/home/zamayra/Bureau/CHATBOT_HERNANDEZ_V1
['nlg_nlu.yml', 'stories.yml', 'nlu.yml', 'rules.yml']

2023-01-06 11:01:05.890125: W tensorflow/stream_executor/platform/d
efault/dso_loader.cc:59] Could not load dynamic library 'libcudart.
so.10.1'; dlerror: libcudart.so.10.1: cannot open shared object fil
e: No such file or directory
2023-01-06 11:01:05.890152: I tensorflow/stream_executor/cuda/cuda_r
t_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU
set up on your machine.
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.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component CountVectorsFeaturi
zer
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component DIETClassifier
2023-01-06 11:01:07.912980: W tensorflow/stream_executor/platform/d
efault/dso_loader.cc:59] Could not load dynamic library 'libcuda.s
o.1'; dlerror: libcuda.so.1: cannot open shared object file: No suc
h file or directory
2023-01-06 11:01:07.913003: W tensorflow/stream_executor/cuda/cuda_
driver.cc:312] failed call to cuInit: UNKNOWN ERROR (303)
2023-01-06 11:01:07.913022: I tensorflow/stream_executor/cuda/cuda_
diagnostics.cc:156] kernel driver does not appear to be running on
this host (zamayra-IdeaPad-Flex-5-15ALC05): /proc/driver/nvidia/ver
sion does not exist
2023-01-06 11:01:07.913330: I tensorflow/core/platform/cpu_feature_

```

```
guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep
Neural Network Library (oneDNN) to use the following CPU instruction
s in performance-critical operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the app
ropriate compiler flags.
2023-01-06 11:01:07.946523: I tensorflow/core/platform/profile_util
s/cpu_utils.cc:104] CPU Frequency: 2096074999 Hz
2023-01-06 11:01:07.947998: I tensorflow/compiler/xla/service/servi
ce.cc:168] XLA service 0x9d36f30 initialized for platform Host (thi
s does not guarantee that XLA will be used). Devices:
2023-01-06 11:01:07.948052: I tensorflow/compiler/xla/service/servi
ce.cc:176] StreamExecutor device (0): Host, Default Version
WARNING:tensorflow:From /home/zamayra/anaconda3/envs/rasa2_2021/lib
/python3.7/site-packages/rasa/utils/tensorflow/layers.py:635: calli
ng while_loop_v2 (from tensorflow.python.ops.control_flow_ops) with
back_prop=False is deprecated and will be removed in a future versi
on.
Instructions for updating:
back_prop=False is deprecated. Consider using tf.stop_gradient inst
ead.
Instead of:
results = tf.while_loop(c, b, vars, back_prop=False)
Use:
results = tf.nest.map_structure(tf.stop_gradient, tf.while_loop(c,
b, vars))
Epochs: 100%|██████████| 100/100 [00:13<00:00, 7.59it/s, t_loss=1.
433, i_acc=0.988]
INFO:rasa.utils.tensorflow.models:Finished training.
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component EntitySynonymMapper
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component ResponseSelector
INFO:rasa.nlu.selectors.response_selector:Retrieval intent paramete
r was left to its default value. This response selector will be tra
ined on training examples combining all retrieval intents.
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Starting to train component FallbackClassifier
INFO:rasa.nlu.model:Finished training component.
INFO:rasa.nlu.model:Successfully saved model into '/home/zamayra/Bu
reau/CHATBOT_HERNANDEZ_V1/models/nlu/current'
```

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

Il a entraîné le model avec les 100 epocs déterminés dans le fichier config. Il a extrait les features de lexical et syntatic, le count vector features et finalement le clasifieur.

Il a crée le model dans le dossier models/nlu/current, un fichier par etape

Entrée [5]:

```
#PATH = '/Users/ferrane/Documents/ENSEIGNEMENT/SRI-3A/DOHM/BE_2020/TI
#os.chdir(PATH)
#os.listdir()
#Compléter avec le chemin vers votre environnement virtuel créé lors
print(os.getcwd())
print(os.listdir('models/nlu/current'))
warnings.filterwarnings('ignore')
```

```
/home/zamayra/Bureau/CHATBOT_HERNANDEZ_V1
['metadata.json', 'component_1 RegexFeaturizer.patterns.pkl', 'comp
onent_4 CountVectorsFeaturizer.pkl', 'component_5 DIETClassifier.la
bel_data.pkl', 'checkpoint', 'component_5 DIETClassifier.tf_model.i
ndex', 'component_5 DIETClassifier.sparse_feature_sizes.pkl', 'comp
onent_5 DIETClassifier.data_example.pkl', 'component_5 DIETClassifi
er.index_label_id_mapping.json', 'component_3 CountVectorsFeaturize
r.pkl', 'component_5 DIETClassifier.entity_tag_specs.json', 'compon
ent_2 LexicalSyntacticFeaturizer.feature_to_idx_dict.pkl', 'compone
nt_1 RegexFeaturizer.pkl', 'component_5 DIETClassifier.tf_model.dat
a-00000-of-00001']
```

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 preferred 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.

Entrée [6]:

```
pprint(interpreter.parse(""))
```

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

Entrée [7]: `pprint(interpreter.parse("I am fine thanks"))`

```
{
  "text": "I am fine thanks",
  "intent": {
    "id": 4798533225359005514,
    "name": "mood_great",
    "confidence": 0.9948071837425232
  },
  "entities": [],
  "intent_ranking": [
    {
      "id": 4798533225359005514,
      "name": "mood_great",
      "confidence": 0.9948071837425232
    },
    {
      "id": 8108612465111742153,
      "name": "thank_you",
      "confidence": 0.004050862044095993
    },
    {
      "id": -1720468704510045991,
      "name": "mood_unhappy",
      "confidence": 0.00042920716805383563
    },
    {
      "id": -3249312102846174634,
      "name": "deny",
      "confidence": 0.00034078722819685936
    },
    {
      "id": -630209616494200750,
      "name": "affirm",
      "confidence": 0.00030410048202611506
    },
    {
      "id": -8488642756196513199,
      "name": "bot_challenge",
      "confidence": 5.1900969992857426e-05
    },
    {
      "id": -2086608434569331290,
      "name": "goodbye",
      "confidence": 1.1002875908161514e-05
    },
    {
      "id": 3408378144440507382,
      "name": "greet",
      "confidence": 4.933444415655686e-06
    }
  ],
  "response_selector": {
    "all_retrieval_intents": [],
    "default": {
      "response": {
        "id": null,
        "response_templates": null,
        "confidence": 0.0,

```

```
        "intent_response_key": null,  
        "template_name": "utter_None"  
    },  
    "ranking": []  
}  
,
```

Entrée [9]: `pprint(interpreter.parse("Thank you"))`

```
{
  "text": "Thank you",
  "intent": {
    "id": 8108612465111742153,
    "name": "thank_you",
    "confidence": 0.9724383354187012
  },
  "entities": [],
  "intent_ranking": [
    {
      "id": 8108612465111742153,
      "name": "thank_you",
      "confidence": 0.9724383354187012
    },
    {
      "id": -3249312102846174634,
      "name": "deny",
      "confidence": 0.010405207052826881
    },
    {
      "id": 3408378144440507382,
      "name": "greet",
      "confidence": 0.004885082133114338
    },
    {
      "id": -8488642756196513199,
      "name": "bot_challenge",
      "confidence": 0.0037008372601121664
    },
    {
      "id": 4798533225359005514,
      "name": "mood_great",
      "confidence": 0.0032148079480975866
    },
    {
      "id": -630209616494200750,
      "name": "affirm",
      "confidence": 0.002774083986878395
    },
    {
      "id": -2086608434569331290,
      "name": "goodbye",
      "confidence": 0.0012956041609868407
    },
    {
      "id": -1720468704510045991,
      "name": "mood_unhappy",
      "confidence": 0.0012860066490247846
    }
  ],
  "response_selector": {
    "all_retrieval_intents": [],
    "default": {
      "response": {
        "id": null,
        "response_templates": null,
        "confidence": 0.0,

```

```
        "intent_response_key": null,  
        "template_name": "utter_None"  
    },  
    "ranking": []  
}  
,
```



Entrée [10]: `pprint(interpreter.parse("I would prefer a puppy"))`

```
{
  "text": "I would prefer a puppy",
  "intent": {
    "id": 4798533225359005514,
    "name": "mood_great",
    "confidence": 0.8967997431755066
  },
  "entities": [],
  "intent_ranking": [
    {
      "id": 4798533225359005514,
      "name": "mood_great",
      "confidence": 0.8967997431755066
    },
    {
      "id": -1720468704510045991,
      "name": "mood_unhappy",
      "confidence": 0.09364141523838043
    },
    {
      "id": -630209616494200750,
      "name": "affirm",
      "confidence": 0.004464345518499613
    },
    {
      "id": 8108612465111742153,
      "name": "thank_you",
      "confidence": 0.0024569802917540073
    },
    {
      "id": -8488642756196513199,
      "name": "bot_challenge",
      "confidence": 0.0018195515731349587
    },
    {
      "id": 3408378144440507382,
      "name": "greet",
      "confidence": 0.0003718602820299566
    },
    {
      "id": -3249312102846174634,
      "name": "deny",
      "confidence": 0.00024357091751880944
    },
    {
      "id": -2086608434569331290,
      "name": "goodbye",
      "confidence": 0.00020257392316125333
    }
  ],
  "response_selector": {
    "all_retrieval_intents": [],
    "default": {
      "response": {
        "id": null,
        "response_templates": null,
        "confidence": 0.0,

```

```
        "intent_response_key": null,  
        "template_name": "utter_None"  
    },  
    "ranking": []  
}  
,
```

Entrée [11]: `pprint(interpreter.parse("I would have preferred a puppy"))`

```
{
  "text": "I would have preferred a puppy",
  "intent": {
    "id": 4798533225359005514,
    "name": "mood_great",
    "confidence": 0.8730278611183167
  },
  "entities": [],
  "intent_ranking": [
    {
      "id": 4798533225359005514,
      "name": "mood_great",
      "confidence": 0.8730278611183167
    },
    {
      "id": -1720468704510045991,
      "name": "mood_unhappy",
      "confidence": 0.09520818293094635
    },
    {
      "id": -2086608434569331290,
      "name": "goodbye",
      "confidence": 0.014331397600471973
    },
    {
      "id": 8108612465111742153,
      "name": "thank_you",
      "confidence": 0.005083020310848951
    },
    {
      "id": -630209616494200750,
      "name": "affirm",
      "confidence": 0.004531083628535271
    },
    {
      "id": -8488642756196513199,
      "name": "bot_challenge",
      "confidence": 0.0031407037749886513
    },
    {
      "id": 3408378144440507382,
      "name": "greet",
      "confidence": 0.0031155257020145655
    },
    {
      "id": -3249312102846174634,
      "name": "deny",
      "confidence": 0.0015622341306880116
    }
  ],
  "response_selector": {
    "all_retrieval_intents": [],
    "default": {
      "response": {
        "id": null,
        "response_templates": null,
        "confidence": 0.0,

```

```
        "intent_response_key": null,  
        "template_name": "utter_None"  
    },  
    "ranking": []  
}  
,
```

Entrée [12]: `pprint(interpreter.parse("I would like to thank you very much for th`

```
{
  "text": "I would like to thank you very much for this",
  "intent": {
    "id": 4798533225359005514,
    "name": "mood_great",
    "confidence": 0.7585334181785583
  },
  "entities": [],
  "intent_ranking": [
    {
      "id": 4798533225359005514,
      "name": "mood_great",
      "confidence": 0.7585334181785583
    },
    {
      "id": 8108612465111742153,
      "name": "thank_you",
      "confidence": 0.12621036171913147
    },
    {
      "id": -3249312102846174634,
      "name": "deny",
      "confidence": 0.05375521630048752
    },
    {
      "id": -630209616494200750,
      "name": "affirm",
      "confidence": 0.029931381344795227
    },
    {
      "id": -1720468704510045991,
      "name": "mood_unhappy",
      "confidence": 0.01292849238961935
    },
    {
      "id": -8488642756196513199,
      "name": "bot_challenge",
      "confidence": 0.012136689387261868
    },
    {
      "id": 3408378144440507382,
      "name": "greet",
      "confidence": 0.004158846568316221
    },
    {
      "id": -2086608434569331290,
      "name": "goodbye",
      "confidence": 0.00234550004824996
    }
  ],
  "response_selector": {
    "all_retrieval_intents": [],
    "default": {
      "response": {
        "id": null,
        "response_templates": null,
        "confidence": 0.0,

```

```

        "intent_response_key": null,
        "template_name": "utter_None"
    },
    "ranking": []
}

```

Commentaires sur les résultats obtenus

I am fine thanks  
Thank you

sont reconnus avec le bon label, cependant les phrases plus longues  
comme :

I would prefer a puppy  
I would have preferred a puppy  
I would like to thank you very much for this

ont une label erronée car n'existe pas dans le vocabulaire fixé  
dans data/nlu.yml

Il faut ajouter un novel intent change\_image pour prendre en compte  
ce types des phrases

## 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 ?

```

intent:
-ask_another_animal

```

```

action:
-action_random_image

```

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

- I would like to see a [puppy](animal)
- Do you have any pictures of [butterflies](animal)
- I want to see a [koala](animal)

Entity:

```
- animal
```

```
slot
```

```
- animal
```

```
lookup:
```

```
- puppy
```

```
- butterflies
```

```
- 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

Il faut définir les animaux dans lookup et aller les chercher dans un lookup table depuis le fichier actions.

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

-----

```
# domain.yml
```

-----

```
# Il faut ajouter 2 entités
```

```
#   change_image - l'utilisateur veut changer l'image par un autre animal donnée
```

```
#   ask_another_animal - L'utilisateur veut n'importe quel animal différent à celle de l'image précédent
```

```
# Pour cela il faut avoir une entité animal qui va reconnaître l'animal désiré
```

```
# Il faut aussi un slot avec le même nom pour sauvegarder la valeur
```

```
# Et finalement il faut 2 actions:
```

```
#   action_show_image - elle va envoyer l'image de l'animal donnée
```

```
#   action_random_image - elle va envoyer une image random de la base de données
```

```
intent:
```

```
- change_image
```

```
- ask_another_animal
```

```
entities:
```

```
- animal
```

```
slots:
```

```
  animal:
```

```
    type: text
```

```
    influence_conversation: true
```

```
actions:
- action_show_image
- action_random_image

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

```
-----
-----
# nlu.yml
-----
-----
```

```
# Une fois ajouté les intent et entités dans domain, il faut donner
des exemples
# Il faut ajouter des exemples des intent change_image et
ask_another_animal
```

```
# Dans l'intent change_image il faut donner des exemples en
utilisant l'entité et le slot
# [value](entity)
```

```
# Il a aussi un ajout des synonymes pour que l'utilisateur puisse
être plus flexible dans son vocabulaire
# il peut dire "I want to see a dog" ou "I want to see a puppy" et
l'entité extrait sera "puppy" pour les deux cas
```

```
# Pour que la table lookup marche, il faut ajouter aussi des
exemples de la table qui se trouvent dans la base de données du
fichier actions.py
```

```
- intent: change_image
  examples: |
    - I would like to see a [puppy](animal)
    - I prefer to see a [dog>{"entity": "animal", "value": "puppy"}
    - I would like to see the [man's best friend>{"entity":
"animal", "value": "puppy"}
    - Do you have any pictures of [butterflies](animal)
    - I want to see a [koala](animal)
```

```
- intent: ask_another_animal
  examples: |
    - I would like to see another animal
    - Do you have another animals to show?
    - I want to see another picture
```

```
- synonym: puppy
  examples: |
    - dog
    - man's best friend
    - pup
    - doggy
```

```
- lookup: animal
  examples: |
    - puppy
    - butterflies
    - kangaroo
    - elephant
    - giraffe
```



```
- panda
- cat
- dog
- horse
```

```
-----
-----
# rules.yml
-----
-----
```

```
# Il faut aussi creer 2 comportements fixes:
```

```
# La premier rule est à chaque fois que l'utilisateur veut changer
par une image d'un animal spécifique,
# il faut faire appelle à l'action action_show_image qui va
afficher l'image
```

```
# La seconde est pour un animal random, a chaque fois que
l'utilisateur veut changer d'animal,
# le chatbot enverra une image random de la base de données
```

```
# a la fin le chatbot demande si ca l'a aidé
```

```
- rule: show the animal image that user demande
  steps:
    - intent: change_image
    - action: action_show_image
- rule: show a random animal
  steps:
    - intent: ask_another_animal
    - action: action_random_image
-----
-----
```

```
-----
-----
# stories.yml
-----
-----
```

```
# Il faut que le chatbot s'entraine, alors on ajout des stories
simulant le comportement du chatbot.
```

```
# Il y a 3 types des comportements prévus:
```

```
# 1) l'entité n'est pas reconnu alors il faut envoyer le message
correspondant en disant que l'animal ne se trouve pas dans la base
de données.
```

```
# 2) l'entité est detecté, slot remplis, et il envoie l'image
demandé
```

```
# 3) l'utilisateur veut une autre image n'importe lequel
```

```
- story: change image
  steps:
    - intent: greet
```

```
- action: utter_greet
- intent: mood_unhappy
- action: utter_cheer_up
- action: utter_did_that_help
- intent: change_image
- action: action_show_image
- intent: goodbye
- action: utter_goodbye

- story: change image vol 2
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: change_image
  - slot_was_set:
    - animal: koala
  - action: action_show_image
  - intent: thank_you
  - action: utter_welcome
  - intent: goodbye
  - action: utter_goodbye

- story: ask another image
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: ask_another_image
  - action: action_random_image
  - intent: deny
  - intent: goodbye
  - action: utter_goodbye
```

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

```
-----
-----
# config.yml
-----
-----
```

```
# comme on utilise des tables Lookup, il faut le definir avec
RegexEntityExtractor, sinon il ne fera pas
# l'extraction des entitees.
```

```
- name: RegexEntityExtractor
  use_lookup_table: True

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

```
-----  
-----  
# endpoints.yml  
-----  
-----
```

```
# Pour utiliser les actions il faut decommenter les lignes  
suivantes:  
# car il permet de lancer le serveur en lançant la commande rasa  
run actions
```

```
action_endpoint:  
    url: "http://localhost:5055/webhook"
```

```
-----  
-----  
# actions.py  
-----  
-----
```

```
# Il faut definir la base de données et configurer les responses  
selon les actions
```

```
import random
```

```
from typing import Any, Text, Dict, List
```

```
from rasa_sdk import Action, Tracker
```

```
from rasa_sdk.executor import CollectingDispatcher
```

```
animals_db = {  
    'puppy': 'https://i.imgur.com/aeWQBPb.jpeg',  
    'butterflies': 'https://i.imgur.com/wuaYap6.jpeg',  
    'koala': 'https://i.imgur.com/lp5nKyZ.jpeg',  
    'kangaroo': 'https://i.imgur.com/7jKQ33i.jpeg',  
    'elephant': 'https://i.imgur.com/3fLmUEu.jpeg',  
    'turtle': 'https://i.imgur.com/a6Xjw5L.jpeg',  
    'giraffe': 'https://i.imgur.com/htl4k1N.jpeg',  
    'panda': 'https://i.imgur.com/JXiw5qb.jpeg',  
    'dog': 'https://i.imgur.com/fSgnUKW.jpeg',  
    'cat': 'https://i.imgur.com/0HErNcZ.jpeg',  
    'horse': 'https://i.imgur.com/zXrsJRE.jpeg'  
}
```

```
class ActionShowImage(Action):
```

```
    def name(self) -> Text:  
        return "action_show_image"
```

```
    def run(self, dispatcher: CollectingDispatcher,  
            tracker: Tracker,  
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
```

```
        animal_type =  
        next(tracker.get_latest_entity_values("animal"), None)
```

```
        if not animal_type:
```

```

        msg = f"I'm sorry, I don't have photos of that animal.
Try another one! for exemple butterflies, kangaroo, puppy or
panda!"

        dispatcher.utter_message(text=msg)
        return []

    animal_image = animals_db.get(animal_type, None)

    dispatcher.utter_message(text = "Let me see... I found
this... Does it help?",
                              image=animal_image)

class ActionRandomImage(Action):

    def name(self) -> Text:
        return "action_random_image"

    def run(self, dispatcher: CollectingDispatcher,
            tracker: Tracker,
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:

        animal_type = random.choice(list(animals_db.keys()))

        animal_image = animals_db.get(animal_type, None)

        dispatcher.utter_message(text = f"I found this cute
{animal_type} ! Does it help?",
                                image=animal_image)

```

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

- User: hello
- Chatbot: Hey! how are you?
- User: sad
- Chatbot: Here is something to cheer you up:
- Chatbot: Does it help?
- User: I want to see a turtle
- Chatbot: Let me see... I found this... Does it help?
- User: yes
- Chatbot: Great, carry on!
- User: bye
- Chatbot: Bye

#### Exemple 2

- User: hello
- Chatbot: Hey! how are you?
- User: sad
- Chatbot: Here is something to cheer you up:
- Chatbot: Does it help?

```
- User: I would like to see a giraffe
- Chatbot: Let me see... I found this... Does it help?
- User: I would like to see a giraffe
- Chatbot: Let me see... I found this... Does it help?
- User: yes
- Chatbot: Great, carry on!
- User: bye
- Chatbot: Bye
```

### Exemple 3

```
- User: hello
- Chatbot: Hey! how are you?
- User: sad
- Chatbot: Here is something to cheer you up:
- Chatbot: Does it help?
- User: I would like to see a panda
- Chatbot: Let me see... I found this... Does it help?
- User: I would like to see a bird
- Chatbot: I'm sorry, I don't have photos of that animal. Try
another one! for exemple butterflies, kangaroo, puppy or panda!
- User: bye
- Chatbot: Bye
```

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

Toutes les modifications pertinentes sont montrées dans les cellules antérieures

Pour "what is your favorite animal" il faut ajouter les suivantes modifications:

```
-----
# nlu.yml
-----

# Il faut ajouter un intent favorite_animal et ajouter aussi des
synonyms pour permettre à l'utiliser
# plus de liberté du langage

- intent: favorite_animal
  examples: |
    - My favorite animal is a [puppy](animal)
    - my fav animal is[dog>{"entity": "animal", "value": "puppy"}
    - the [cat>{"entity": "animal", "value": "cat"} is my favorite
animal
    - [kangaroo](animal) is my favorite animal
    - I like the [kittis>{"entity": "animal", "value": "cat"}
    - I love the[horses>{"entity": "animal", "value": "horse"}
    - I love the [ponnies>{"entity": "animal", "value": "horse"}

- intent: talk_me
  examples: |
    - I would like to talk
    - I want to talk
```

```
- I want to talk you
- I want to talk to you
- I'm bored

- synonym: horse
  examples: |
    - horses
    - pony
    - ponnies

- synonym: puppy
  examples: |
    - dog
    - man's best friend
    - pup
    - doggy

- synonym: cat
  examples: |
    - kitty
    - kittis
    - kitten
    - meow
    - catty
```

-----

-----

```
-----
-----
# domain.yml
-----
-----
```

```
# Il faut definir l'intent, ajouter un utter_favorite_animal et
faire l'action associé pour les affichage
```

```
intents:
- favorite_animal
- talk_me

responses:
  utter_favorite_animal:
    - text: what is your favorite animal?
actions:
- action_favorite_animal

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

```
-----
-----
# stories.yml
-----
-----
```

```
- story: ask favorite animal
  steps:
    - intent: greet
    - action: utter_greet
```

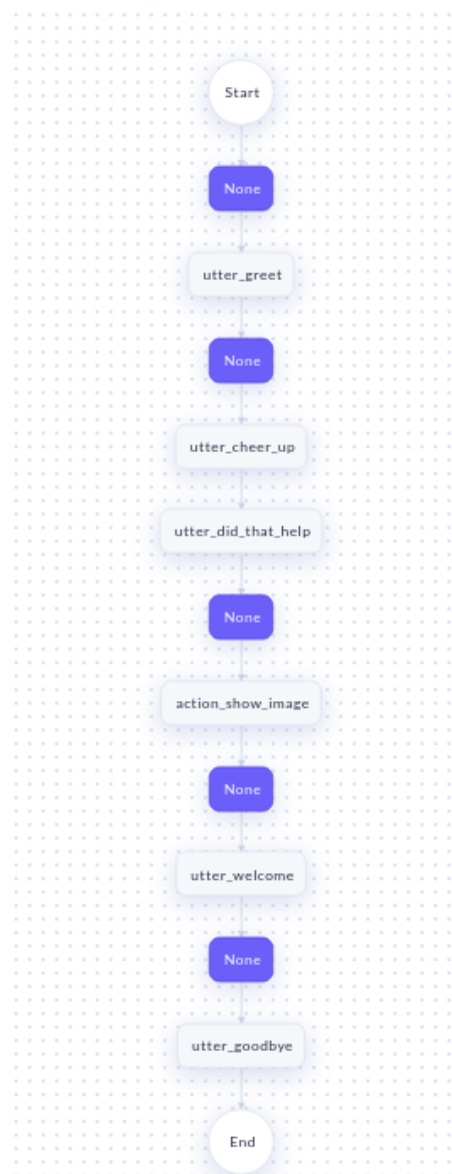
```
- intent: mood_great
- action: utter_happy
- intent: talk_me
- action: utter_favorite_animal
- intent: action_favorite_animal
  entities:
    - animal: cat
- slot_was_set:
  - animal: cat
- action: action_favorite_animal
- intent: thank_you
- action: utter_welcome
```

-----

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 cellule suivante

Model story graph - change image



## Model story graph - favorite animal



## 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 arrêté, tapez la commande : `rasa visualize`

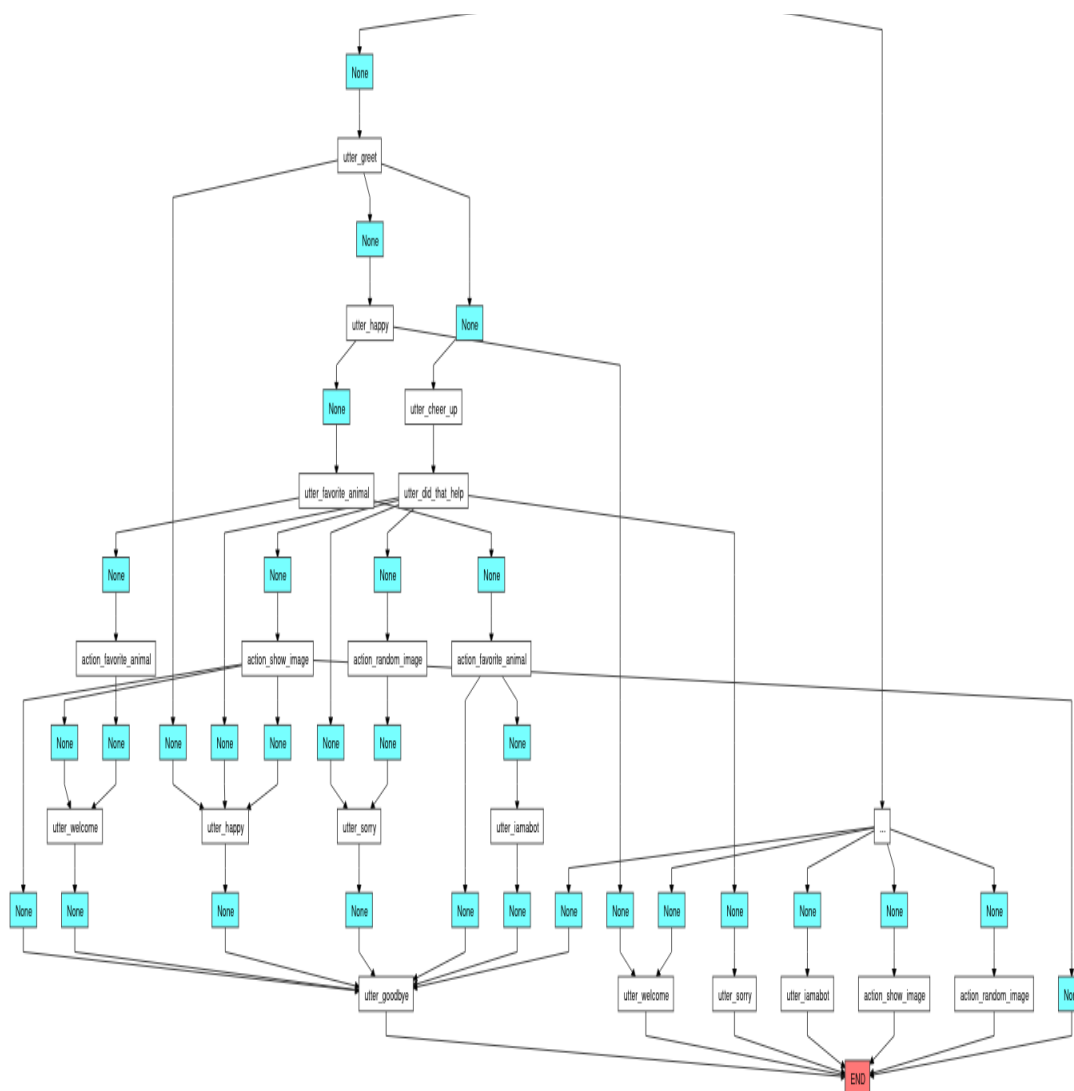
- Visualisez le fichier Graph.html
- Insérer dans la cellule suivante une copie image de ce graphe (copie écran ou autre)

Que représente-t-il ?

Elle crée une représentation visuelle du comportement de toutes les stories données







## PARTIE 10 : EVALUATION DE VOTRE CHATBOT V2

.....

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.

```

rasa test

```

```
2023-01-23 23:13:47 INFO      rasa.shared.utils.validation - The
'version' key is missing in the training data file /tmp/tmp9sbg3o7q
/b0fa43445fb04c8889229fabd8f69d76_test_stories.yml. Rasa Open
Source will read the file as a version '2.0' file. See
https://rasa.com/docs/rasa/training-data-format.
```

Processed story blocks:

100%|

```
24/24 [00:00<00:00, 3920.37it/s, # trackers=1]
```

```
2023-01-23 23:13:47 INFO      rasa.core.test - Evaluating 24
stories
```

```
Progress:
100%|███████████████████████████████████████████████████████████████████████████
██████████████████████████████████████████████████████████████████████████████| 24/24 [00:01<00:00,
13.04it/s]
2023-01-23 23:13:49 INFO rasa.core.test - Finished collecting
predictions.
2023-01-23 23:13:49 INFO rasa.core.test - Evaluation Results
on END-T0-END level:
2023-01-23 23:13:49 INFO rasa.core.test - Correct:
21 / 24
2023-01-23 23:13:49 INFO rasa.core.test - F1-Score:
0.933
2023-01-23 23:13:49 INFO rasa.core.test - Precision:
1.000
2023-01-23 23:13:49 INFO rasa.core.test - Accuracy:
0.875
2023-01-23 23:13:49 INFO rasa.core.test - In-data fraction:
0.573
2023-01-23 23:13:49 INFO rasa.core.test - Stories report saved
to results/story_report.json.
2023-01-23 23:13:51 INFO rasa.core.test - Evaluation Results
on ACTION level:
2023-01-23 23:13:51 INFO rasa.core.test - Correct:
162 / 164
2023-01-23 23:13:51 INFO rasa.core.test - F1-Score:
0.990
2023-01-23 23:13:51 INFO rasa.core.test - Precision:
0.994
2023-01-23 23:13:51 INFO rasa.core.test - Accuracy:
0.988
2023-01-23 23:13:51 INFO rasa.core.test - In-data fraction:
0.573
2023-01-23 23:13:51 INFO rasa.utils.plotting - Confusion
matrix, without normalization:
[[ 0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[ 0  2  0  0  0  0  0  0  0  0  0  0  0  0]
[ 0  0 77  0  0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  2  0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  3  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0 10  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0 10  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  2  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0 18  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0 16  0  0  0  0]
[ 2  0  0  0  0  0  0  0  0  0  9  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0  0  4  0  0]
[ 0  0  0  0  0  0  0  0  0  0  0  0  3  0]
[ 0  0  0  0  0  0  0  0  0  0  0  0  0  6]]
2023-01-23 23:13:54 INFO rasa.nlu.test - Running model for
predictions:
100%|███████████████████████████████████████████████████████████████████████████
██████████████████████████████████████████████████████████████████████████████| 167/167 [00:01<00:00,
148.66it/s]
2023-01-23 23:13:55 INFO rasa.nlu.test - Intent evaluation
results:
2023-01-23 23:13:55 INFO rasa.nlu.test - Intent Evaluation:
Only considering those 167 examples that have a defined intent out
of 167 examples.
```

```

2023-01-23 23:13:55 INFO      rasa.nlu.test - Classification report
saved to results/intent_report.json.
2023-01-23 23:13:55 INFO      rasa.nlu.test - Your model predicted
all intents successfully.
2023-01-23 23:13:55 INFO      rasa.utils.plotting - Confusion
matrix, without normalization:
[[11  0  0  0  0  0  0  0  0  0  0  0]
 [ 0 11  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  8  0  0  0  0  0  0  0  0  0]
 [ 0  0  0 10  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  9  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  8  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 27  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 18  0  0  0  0]
 [ 0  0  0  0  0  0  0  0 28  0  0  0]
 [ 0  0  0  0  0  0  0  0  0 23  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  8  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  6]]
2023-01-23 23:13:56 INFO      rasa.nlu.test - Entity evaluation
results:
2023-01-23 23:13:56 INFO      rasa.nlu.test - Evaluation for entity
extractor: DIETClassifier
2023-01-23 23:13:56 INFO      rasa.nlu.test - Classification report
for 'DIETClassifier' saved to 'results/DIETClassifier_report.json'.
2023-01-23 23:13:56 INFO      rasa.nlu.test - Your model predicted
all entities successfully.
2023-01-23 23:13:56 INFO      rasa.utils.plotting - Confusion
matrix, without normalization:
[[ 18  0]
 [ 0 532]]
2023-01-23 23:13:57 INFO      rasa.nlu.test - Evaluation for entity
extractor: RegexEntityExtractor
2023-01-23 23:13:57 INFO      rasa.nlu.test - Classification report
for 'RegexEntityExtractor' saved to
'results/RegexEntityExtractor_report.json'.
2023-01-23 23:13:57 INFO      rasa.nlu.test - Incorrect entity
predictions saved to results/RegexEntityExtractor_errors.json.
2023-01-23 23:13:57 INFO      rasa.utils.plotting - Confusion
matrix, without normalization:
[[ 10  8]
 [ 0 532]]

```

#### COMMENTAIRES:

Le chatbot a été évalué avec 24 stories différents. 21/24 se sont bien déroulés.

162/164 actions ont été bien prédites

Le model s'est trompé 2 fois, il a confondu utter\_happy avec action\_default\_fallback

```

#-----
-----
./results

failed_test_stories.yml
#-----
-----

```

```
version: "2.0"
stories:
- story: change image (/tmp/tmpsbg3o7q
/b0fa43445fb04c8889229fabd8f69d76_test_stories.yml)
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: change_image # predicted: change_image: I want to see
a [puppy](animal)[puppy](animal)
    entities:
    - animal: puppy
  - slot_was_set:
    - animal: puppy
  - slot_was_set:
    - animal: puppy
  - action: action_show_image
  - intent: goodbye
  - action: utter_goodbye
- story: change image 3 (/tmp/tmpsbg3o7q
/b0fa43445fb04c8889229fabd8f69d76_test_stories.yml)
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: change_image
    entities:
    - animal: koala
  - slot_was_set:
    - animal: koala
  - slot_was_set:
    - animal: koala
  - action: action_show_image
  - intent: affirm
  - action: utter_happy # predicted: action_default_fallback
  - intent: goodbye
  - action: utter_goodbye
- story: ask another image 2 (/tmp/tmpsbg3o7q
/b0fa43445fb04c8889229fabd8f69d76_test_stories.yml)
  steps:
  - intent: greet
  - action: utter_greet
  - intent: mood_unhappy
  - action: utter_cheer_up
  - action: utter_did_that_help
  - intent: ask_another_animal
  - action: action_random_image
  - intent: affirm
  - action: utter_happy # predicted: action_default_fallback
  - intent: thank_you
  - action: utter_welcome
  - intent: goodbye
  - action: utter_goodbye
```

```
#-----  
-----  
  
COMMENTAIRES:  
  
Il y a eu les 2 erreurs de action_default_fallback pour utter_happy  
  
Et pour I want to see a puppy  
  
Les doubles de slot_was_set sont crée par défaut à chaque fois  
qu'une storie échoue. Dans les fichiers de test_stories.yml il y a  
bien seulement 1 seul slot_was_set,  
même chose pour stories.yml  
#-----  
-----  
intent_report.yml  
#-----  
-----  
{  
  "favorite_animal": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 8,  
    "confused_with": {}  
  },  
  "greet": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 18,  
    "confused_with": {}  
  },  
  "goodbye": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 27,  
    "confused_with": {}  
  },  
  "affirm": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 11,  
    "confused_with": {}  
  },  
  "ask_another_animal": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 11,  
    "confused_with": {}  
  },  
  "thank_you": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 6,  
    "confused_with": {}  
  }  
}
```

```
    },
    "change_image": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 10,
        "confused_with": {}
    },
    "deny": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 9,
        "confused_with": {}
    },
    "mood_unhappy": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 23,
        "confused_with": {}
    },
    "talk_me": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 8,
        "confused_with": {}
    },
    "mood_great": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 28,
        "confused_with": {}
    },
    "bot_challenge": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 8,
        "confused_with": {}
    },
    "accuracy": 1.0,
    "macro avg": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 167
    },
    "weighted avg": {
        "precision": 1.0,
        "recall": 1.0,
        "f1-score": 1.0,
        "support": 167
    }
}
```

#-----  
-----

## COMMENTAIRES:

La précision de chaque intent est de 1

```
#-----  
-----  
story_report.yml  
#-----  
-----  
  
{  
  "[dogs]{\"entity\": \"animal\", \"value\": \"puppy\"}": {  
    "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": 10  
  },  
  "[cats]{\"entity\": \"animal\", \"value\": \"cat\"}": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 1  
  },  
  "action_random_image": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 2  
  },  
  "deny": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 3  
  },  
  "mood_great": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 6  
  },  
  "bot_challenge": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 4  
  },  
  "[kitten]{\"entity\": \"animal\", \"value\": \"cat\"}": {  
    "precision": 1.0,  
    "recall": 1.0,  
    "f1-score": 1.0,  
    "support": 1  
  },  
}
```

```
"None": {
  "precision": 0.0,
  "recall": 0.0,
  "f1-score": 0.0,
  "support": 1
},
"action_listen": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 77
},
"utter_sorry": {
  "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": 10
},
"ask_another_animal": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 2
},
"utter_happy": {
  "precision": 1.0,
  "recall": 0.8181818181818182,
  "f1-score": 0.9,
  "support": 11
},
"utter_goodbye": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 18
},
"thank_you": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 6
},
"utter_greet": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 16
},
"utter_iamabot": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 4
},
}
```



```
"[koala](animal)": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 1
},
"greet": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 16
},
"affirm": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 5
},
"change_image": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 3
},
"utter_favorite_animal": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 2
},
"mood_unhappy": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 10
},
"[puppy](animal)": {
  "precision": 0.5,
  "recall": 1.0,
  "f1-score": 0.6666666666666666,
  "support": 1
},
"action_show_image": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 3
},
"favorite_animal": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 2
},
"action_favorite_animal": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 2
},
}
```

```

"goodbye": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 18
},
"talk_me": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 2
},
"utter_welcome": {
  "precision": 1.0,
  "recall": 1.0,
  "f1-score": 1.0,
  "support": 6
},
"micro avg": {
  "precision": 0.9959183673469387,
  "recall": 0.9878542510121457,
  "f1-score": 0.991869918699187,
  "support": 247
},
"macro avg": {
  "precision": 0.9516129032258065,
  "recall": 0.9618768328445748,
  "f1-score": 0.953763440860215,
  "support": 247
},
"weighted avg": {
  "precision": 0.9939271255060729,
  "recall": 0.9878542510121457,
  "f1-score": 0.9901484480431849,
  "support": 247
}
}

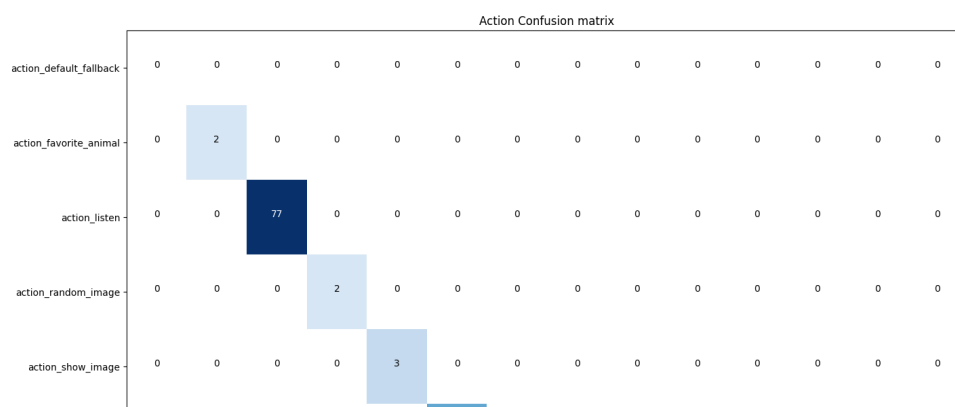
```

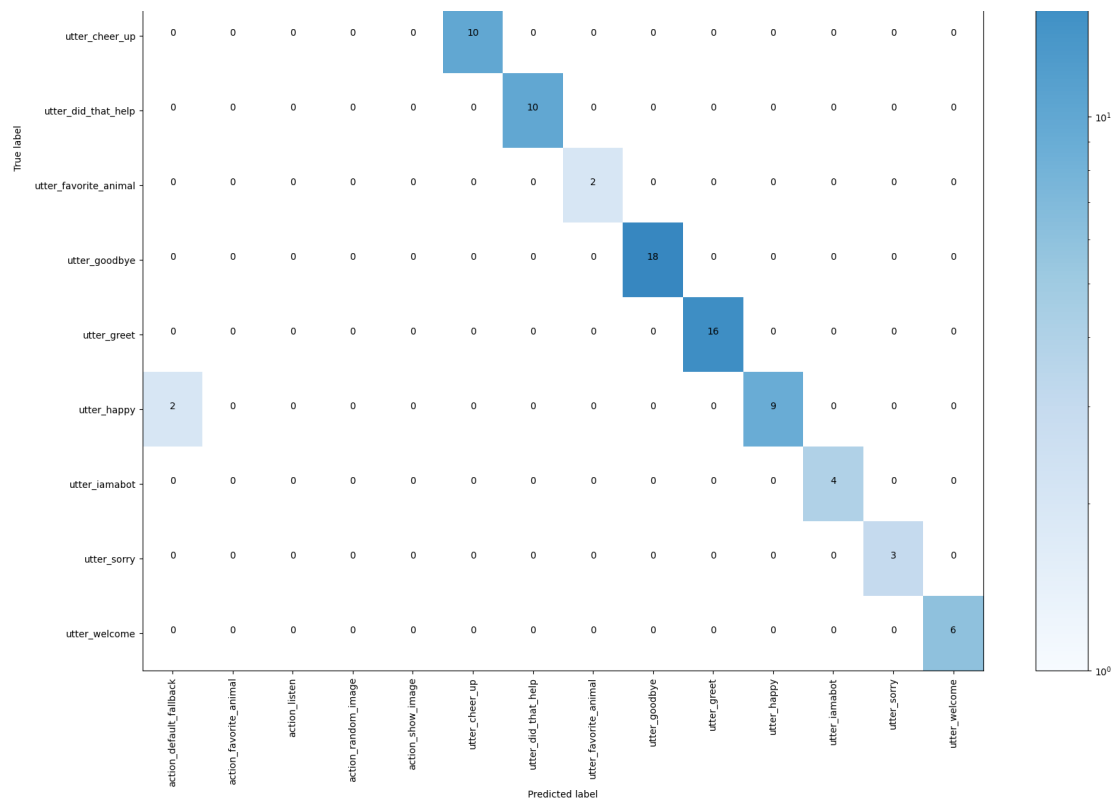
#-----  
-----

#### COMMENTAIRES:

Les synonyms sont bien reconnus  
STORY CONFUSION MATRIX

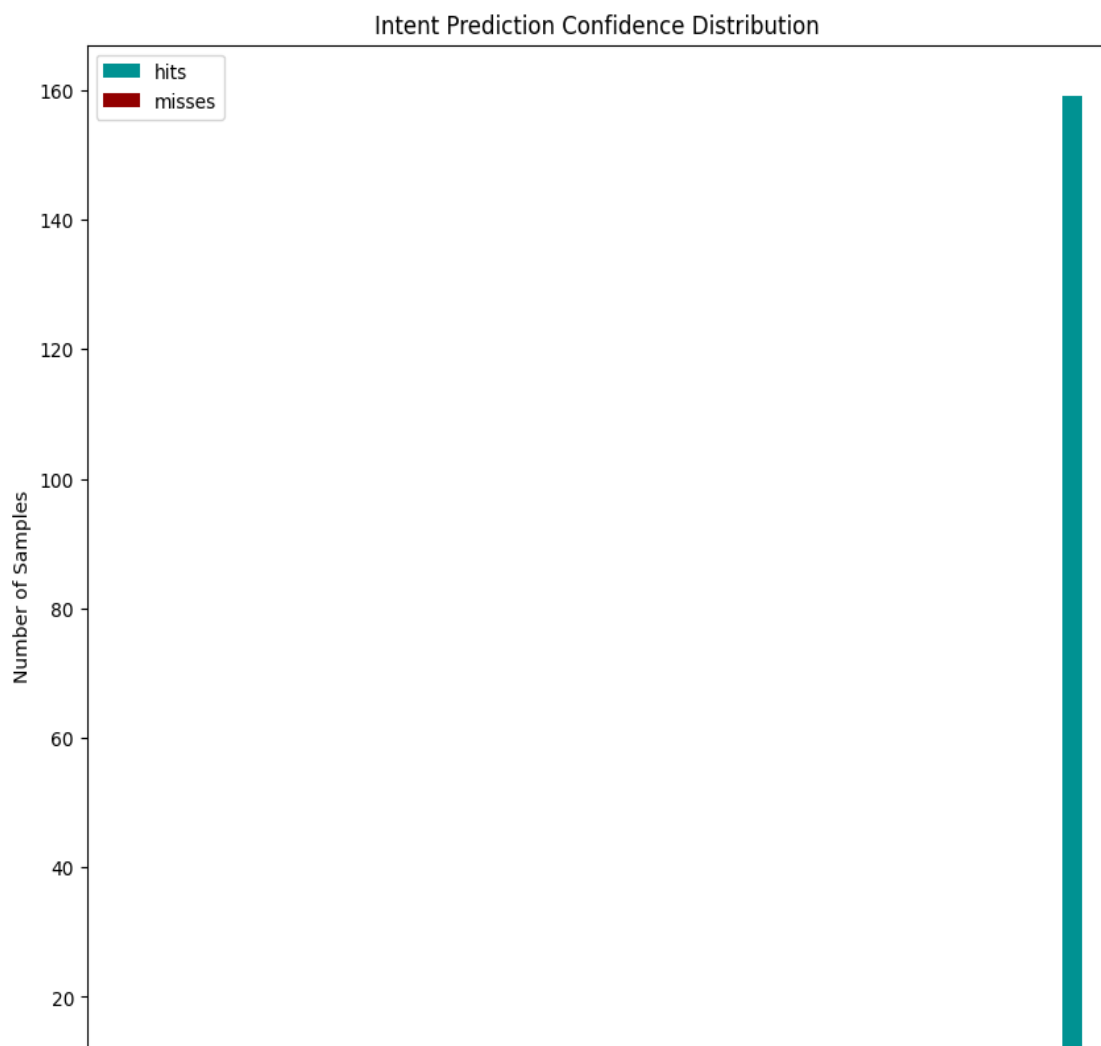
On peut regarder que utter\_happy est reconnu comme action\_default\_fallback

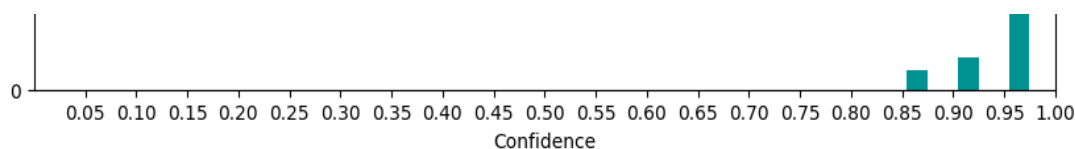




## INTENT HISTOGRAM

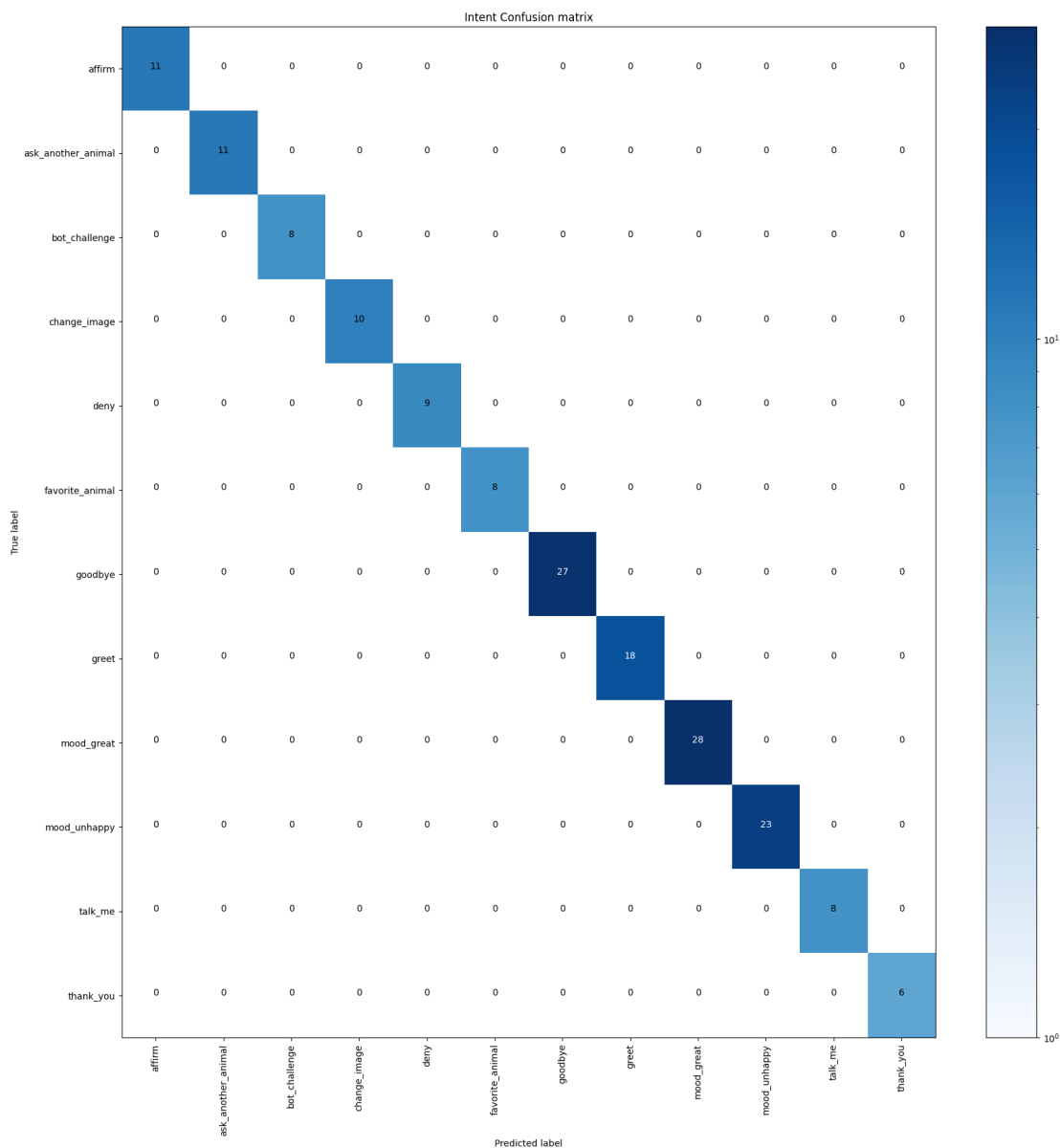
Ici on peut regarder le taux de confiance de chaque intent





## INTENT CONFUSION MATRIX

Tous les intent sont bien reconnus



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.

<https://github.com/Zamayra/ChatbotTP/tree/main>

<https://drive.google.com/drive/folders/1nxCQIx3PDWuRXjrYXuau8cfNsWwjdCmW?usp=sharing>

