AI With Python Workshop

This notebook complements the powerpoint slides during the workshop and will be used to do the coding exercises

AI Project using Python

In this project we will create a virtual assistant that understands natural language. We will be able to interact with it in English to perform the following tasks:

- **Chit-chat**
  - Tell jokes
- **Weather**
  - Get current weather for any city
- **Movies**
  - Get rating for a movie
  - Find the director(s) of a movie

Learning outcomes

In this tutorial you will learn:

- The basic terminologies required in a virtual assistant
  - Intents
  - Slots
  - Entities
  - Utterances
- How to use SNIPS-NLU to understand natural language and detect intents, slots, and entities from utterances.
- How to use the detected intents, slots and entities to get information from APIs

Part 2.1: Preparation

We will use the following libraries in this tutorial

- **snips-nlu**
  - This library deals with the Natural Language Understanding to detect intents, slots, and entities.
- **pyjokes**
  - This library provides jokes based on Python.
- **pyowm**
  - This library provides the weather API to get weather information.
- **imdbpy**
  - This library provides the IMDB movie APU to get movie information.

2.1.1 Install and Import libraries

Run the cell below to install and import the required libraries and functions.

Note: *We have pre-written some code to simplify the weather and movie rating APIs in the file* `utils.py` *. You can view the file later to understand the inner working in more detail.*

```
# Run this cell
!pip install --upgrade pip
!pip install pyjokes
!pip install snips-nlu
!pip install pyowm
!pip install imdbpy
!python -m snips_nlu download en
!git clone https://github.com/xxcuhk/workshop_utils
from workshop_utils.utils import *
import pyjokes
import json
from snips_nlu import SnipsNLUEngine
from snips_nlu.default_configs import CONFIG_EN
```

2.1.2 Create training dataset

Prepare the training dataset to train your Natural Language Understanding (NLU) Engine.

In our training dataset, we will have the following **intents**:

1. **tell_joke** : To detect that the user is asking the virtual assistant for a joke. There are no slots required for this intent.
    - **Example utterances:** "Hi, tell me a joke.", "I'm bored. Entertain me with a funny joke."
2. **get_weather** : To detect that the user is asking for current weather of a city. For this intent we need to define an entity for `city` .
    - **Example utterances:** "How is the weather in New York?", "I wonder how the weather conditions are like in Hong Kong right now?"
3. **get_rating** : To detect that the user is asking the rating for a movie. For this intent we need to define an entity for `movie_name` .
    - **Example utterances:** "How good is the movie Batman?", "I want to know the movie ratings for Fast and Furious"
4. **get_director** : To detect that the user is asking for who is the director of a movie. For this intent we need to define an entity for `movie_name` .
    - **Example utterances:** "Who directed Tenet?", "I want to know the director of the movie Ip Man"

We will have the following **entities**:

1. **city**
    - **Examples**: Hong Kong, New York, Dublin, London
2. **movie_name**
    - **Examples**: Star Wars, Ip Man, The Dark Knight, La la land

We have created a starter dataset for you with some example intents and some example entities in the file `dataset.yaml`

2.1.3 Convert the dataset to json format

Run the next cell to convert the dataset to json format to train the NLU Engine.

```
# Run this cell

!snips-nlu generate-dataset en workshop_utils/1234.yaml > dataset.json
```

2.1.4 Open the dataset

To open the dataset, we will follow the following steps:

1. Use `open` function to load the file into Python in a variable called `dataset_file`.

   To do this you need to run: `dataset_file = open("dataset.json", "r")`

2. Use `load` function from `json` as `json.load(dataset_file)` into a variable called `training_dataset`.

   To do this you need to run: `training_dataset = json.load(dataset_file)`

```
# Write the code below
dataset_file = open("dataset.json", "r")
training_dataset = json.load(dataset_file)
```

Part 2.2: Train the NLU engine

2.2.1 Initialize the Snips-NLU Engine with English Configuration

We will start our Snips-NLU engine using the `SnipsNLUEngine()` function. `config=CONFIG_EN` will load the English language configuration in our NLU engine.

The Snips-NLU engine will be saved in a variable called `NLUengine`.

To do this you need to run: `NLUengine = SnipsNLUEngine(config=CONFIG_EN)`

```
# Write the code below
NLUengine = SnipsNLUEngine(config=CONFIG_EN)
```

2.2.2 Train the NLU Engine

We will now train the NLU engine using our training dataset. `fit()` function is called to train the model.

To do this you need to run: `NLUengine.fit(training_dataset)`

```
# Write the code below

NLUengine.fit(training_dataset)
```

2.2.3 Use the NLU Engine to parse the intention

Let's try to use our engine on the utterance `"How's the weather in Hong Kong"`

Use the function `prediction = NLUengine.parse(your utterance)`

```
# Write the code below

prediction = NLUengine.parse("How is the weather in Hong Kong?")
```

2.2.4 Print the prediction

To print the prediction in a more readable format we will use `json.dumps()` function as:

`print(json.dumps(prediction, indent=2))`

```
# Write the code below

print(json.dumps(prediction, indent=2))
```

2.2.5 Get the intent

To get the intent we access the intent name element from the resulted `prediction` dictionary.

We have made a function for you to get the intent easily. You can use `get_intent(prediction)` to get the intent.

```
# Write the code below

print(get_intent(prediction))
```

2.2.6 Get the entity

You can use our function `get_entity(prediction)` to get the slot's entity.

```
# Write the code below

print(get_entity(prediction))
```

2.2.7 Get the slots's value

You can use our function `get_slot_value(prediction)` to get the slot's value.

```
# Write the code below

print(get_slot_value(prediction))
```

Part 2.3: Integrate the NLU engine with APIs

We have provided you with the following pre-defined functions:

1. `pyjokes.get_joke()` : This function from pyjokes library returns a joke (a nerdy programming based joke).
2. `get_city_weather(city)` : Given a city, it will print its current temperature and weather condition.
3. `get_movie_rating(movie_name)` : Given a movie name, it will print its IMDB rating.
4. `get_movie_directors(movie_name)` : Given a movie name, it will print the name(s) of its director(s).

We will now use these functions to integrate our NLU with the APIs to get a working virtual assistant.

2.3.1 Define a function

First we define a function called `assistant` that given an utterance, returns an appropriate response based on the user's intent. The parameter should be `utterance` .

The function should work in the following manner:

1. Like 2.2.3, use the NLU Engine to parse the intention.

   To do this you need to run: `prediction = NLUengine.parse(utterance)`

2. Like 2.2.5, get the intent.

   To do this you need to run: `intent = get_intent(prediction)`

3. Like 2.2.6, get the entity.

   To do this you need to run: `entity = get_entity(prediction)`

4. If the intent is `tell_joke` , print the output of `pyjokes.get_joke()` function.

5. Else if the intent is `get_weather`, get the value of slot `city` and use the `get_city_weather(city)` function.
6. Else if the intent is `get_rating`, get the value of slot `movie_name` and use the `get_movie_rating(movie_name)` function.
7. Else if the intent is `get_director`, get the value of slot `movie_name` and use the `get_movie_directors(movie_name)` function.
8. Else `print("Unknown intent")`.

Note: Inside the if-statements for `get_weather`, `get_rating`, and `get_director`, you need to add another if-statement to check if the entity is correct. If it is not correct, you need to print "Sorry, please try again."

```
# Write the code below

def assistant(utterance):
    prediction = NLUengine.parse(utterance)
    intent = get_intent(prediction)
    entity = get_entity(prediction)

    if intent == "tell_joke":
        print(pyjokes.get_joke())

    elif intent == "get_weather":

        if entity == "city":
            city_name = get_slot_value(prediction)
            get_city_weather(city_name)
        else:
            print("Sorry, can you try again?")

    elif intent == "get_rating":

        if entity == "movie_name":
            movie_name = get_slot_value(prediction)
            get_movie_rating(movie_name)
        else:
            print("Sorry, can you try again?")

    elif intent == "get_director":

        if entity == "movie_name":
            movie_name = get_slot_value(prediction)
            get_movie_directors(movie_name)
        else:
            print("Sorry, can you try again?")

    else:
        print("Unknown intent")
```

2.3.2 Create a conversation loop

We will create a loop that keeps on going until the user enters **Bye**. `while-loop` will never stop unless it sees a `break`.

The loop has to accomplish the following things:

1. Keep asking for user input until the user enters **Bye**
2. Call the `assistant` function on the user's input

   You can call it like this: `assistant(user_input)`

To break the loop, we will use a new keyword called `break`.

To get input from the user we will use a Python function called `input`.

We have partially written the code below to help you. Please fill the remaining code.

```
# Complete the code belo

print("Welcome to the virtual assistant. How can I help you?")
while True:
    print("------------------------------------")
    user_input=str(input("Enter your input: "))

    # This if statement should break the loop if the user_input is "Bye"
    if user_input == "Bye" or user_input == "bye":
        print("Have a good day!")
        break

    else:
        print("Assistant: ")
        assistant(user_input)
```

End of Part 2

Thank you for attending the workshop!

Don't forget to stop the computation for your notebook. Go to `Run` and then click on `Stop Machine`