Natural Language Processing

for

a GARDENER robot

FRANCESCO

PERACCHIA

1906895

Introduction

The main goal of this project is find a design for a **Task-Oriented Dialog** between an hypothetical human user and a robotic agent, specially for a gardener robot, this agent must be able to perform many kind of actions, and mainly in direct interaction with his owner.
Here is necessary define the specification of this agent because the range of questions and answers make during the dialog depend on this, and also on the depth of the dialog that the agent and the used must have.
In this direction in order to define what kind of dialog they could have, is present a sequence of possible actions that the GARDENER is able to do.

TURN ON
TURN OFF
THANKS
GOODBYE
ACTION-LIST
CHECK
UPDATE

The information about the status of a plant can be used with the special actions CHECK and UPDATE, in these cases we are questioning the agent about one specific information that can be saved in a DB, or we are asking for a modification of an information in the same data base.
Obviously can be added many others action, specially command, like CROP, MOVE, MOVE SOMETHING ECC
Both CHECK and UPDATE actions are used with many possibilities, in particular information about, FLOWERING, HUMIDITY, TEMPERATURE, LAST-WATERING, NEXT-CROP can be useful, some of these are environment information while otherwise are strictly related to a single plant.

	A	B	C	D	E
1	KEY	NAME	FLOWERING	LAST-WATERING	NEXT-CROP
2	115254	Olmo	October	21/01/2021	null
3	121517	Salice	June	23/01/2021	null
4	225486	Apple tree	May	27/01/2021	June

This is an example of an instance saved in our DB, in CVS file, we can also use a JSON file to use directly this information in our project.
This separation of concept is reached in this way, before is used a classification with a model composed by only three layers for the search over the **intentions**, after this fist classification is performed a second classification over the **feature** of each possible intentions, hence we need to design structure that can follow this purpose of the problem, below is proposed a possible model.

```
class Model(nn.Module):
def __init__(self, input_size, hidden_size, num_classes):
super(Model, self).__init__()
self.l1 = nn.Linear(input_size, hidden_size)
self.l2 = nn.Linear(hidden_size, hidden_size)
self.l3 = nn.Linear(hidden_size, num_classes)
```

```
self.relu = nn.ReLU()
```

Model Training

We want to distinguish the intention of the user, to do this a model based on three layers is defined, hence we have input and output layers divided by a inner layer, parameters will be changed accordingly with the performance.
In this fist model we are trying to distinguish what intentions the user have, and what kind of action the agent should do between TURN ON, TURN OFF, THANKS, GOODBYE, ACTION-LIST, CHECK, UPDATE.
This is reached training a model over a fixed sample of possible patterns associated to a specific intents.

```
{
"tag": "goodbye",
"patterns": ["Bye",
"See you later",
"Goodbye"
],
"responses": [
"Thanks for visiting",
"Have a nice day",
"Come back again soon."
]
},
```

```
{
"tag": "thanks",
"patterns": ["Thanks", "Thank you", "Thank's a lot!"]
,
"responses": ["Happy to help!", "My pleasure"]
}
```

The model is trained to classify patterns in the associated tag or intent, in order to do this we need to compute:

1. a dictionary with all the encountered words where is used a tokenization and a stemming procedure

2. all possible association **(tokenized_pattern_sentence, tag)**

3. build a dataset **X_train** and **Y_train**, using the above relation, and the dictionary. Each elements in X_train is a tokenized_pattern_sentence where is applied a bag of words using the dictionary, and the same element in Y_train is the correspondent **tag** (that is the correct label)

Hence at the end we find something in this form :

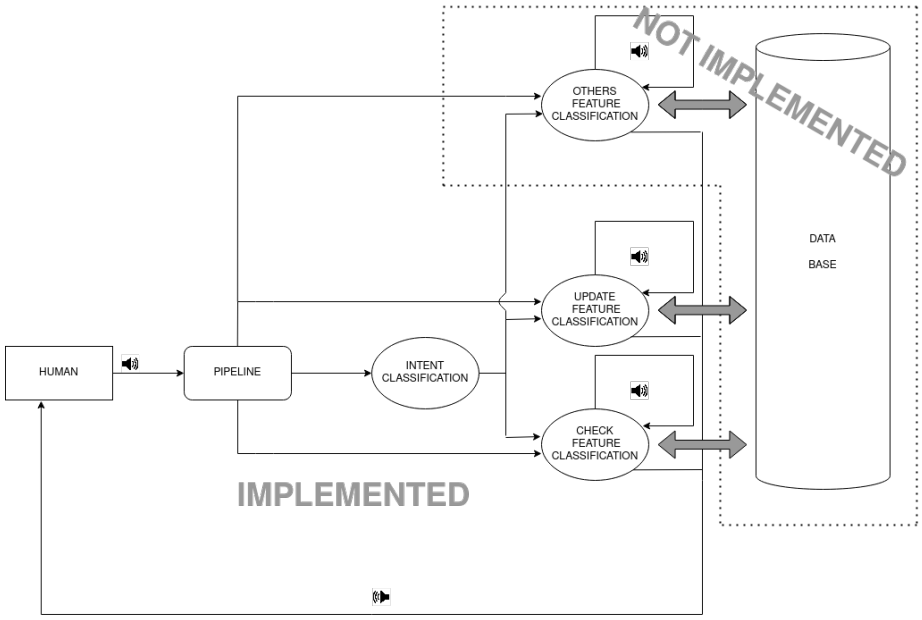
1. **X=train=[[bag of words of S],[bag of words of S]……]** where each elements is length with dimension of the dictionary.
   I.e for bag of words: [0 1 0 0 0 0 0 1 0 1 0 0 0 0 0… 0 0 1]

2. While with the associated tag **Y_train=[["goodbye"],["thanks"]….]**

3. than the model is trained with 1000 epochs and batch size 8

Pipeline

The manipulation of the input text speech is done using this pipeline, fist of all the audio is converted in text, using the function ‘recognize_google’, an offline approach to the speech to text problem, then the sentence is **tokenized** and is build a **bag of word** for each possible model, this is necessary because the model used for the intent classification is different from the successives models used for the feature extraction, this means different input size and different dictionary to build the relative bag of word**,** this array is converted in a form that can be used in the pre-trained models using torch features.
Below is presented the general structure of the chat-loop model.



chat loop example

Human user:
“when this plant will bloom ? ”

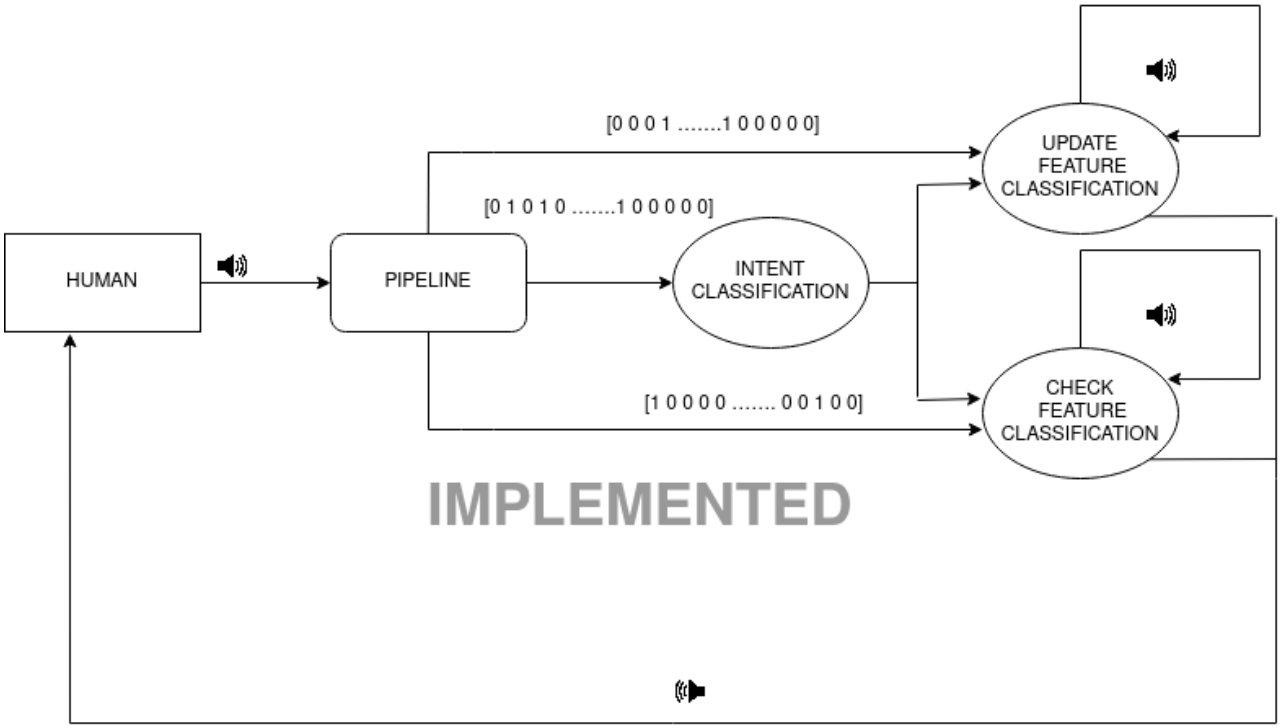
Pipeline
[when, this, plant, will, bloom,?]

punctuation mark are removed

[when, this, plant, will, bloom]

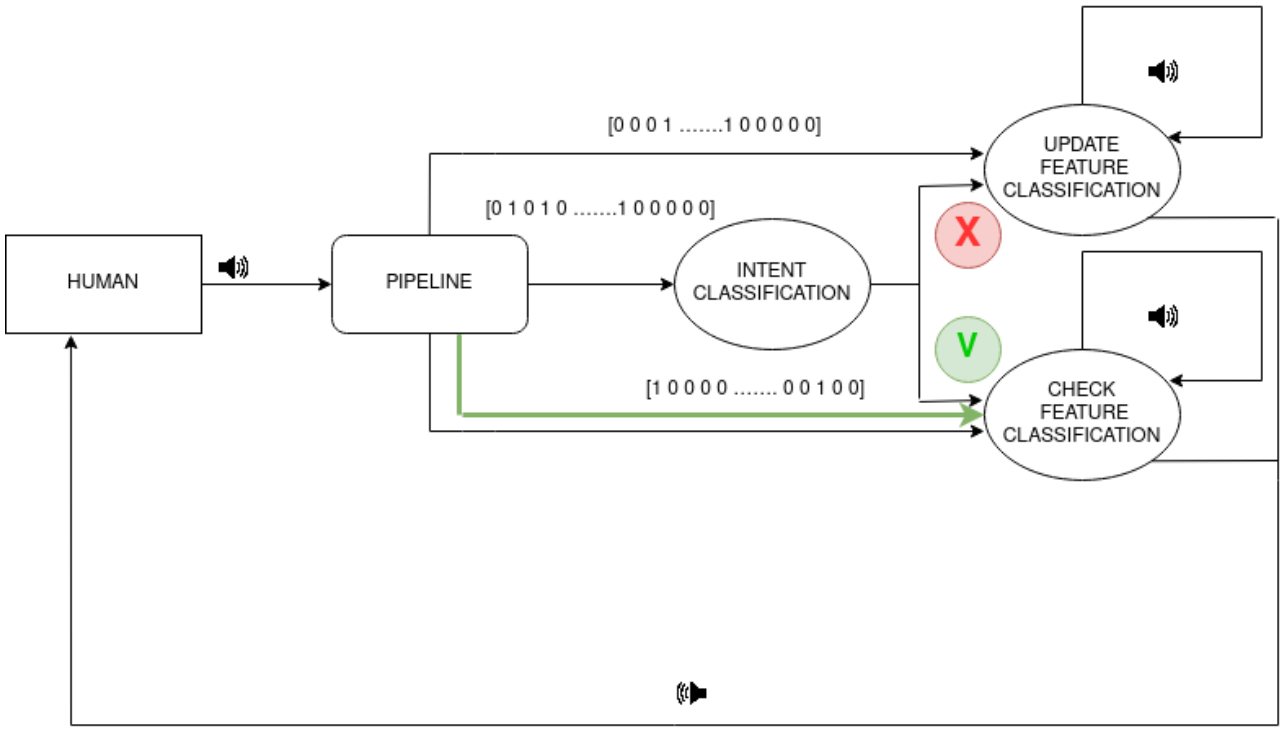
bag of word is build for the three different models

Intent Classification: [0 1 0 1 0 …….1 0 0 0 0 0]
Check Feature Classification: [1 0 0 0 0 ……. 0 0 1 0 0]
Update Feature Classification: [0 0 0 1 …….1 0 0 0 0 0]



Intent Classification: [0 1 0 1 0 …….1 0 0 0 0 0] is fist classified using the fist model, is a CHECK intention, hence we have to extract the feature from this sentences, in order to do this can be recognized one of the following fields: FLOWERING, HUMIDITY, TEMPERATURE, LAST-WATERING, NEXT-CROP.

Check Feature Classification: [1 0 0 0 0 ……. 0 0 1 0 0] is send in input to the CHECK FEATURE CLASSIFICATION MODEL, where is find that the sentence “when this plant will bloom ? ” is a FLOWERING related question. Below is presented this part of the schema, where the remaining Update Feature Classification: [0 0 0 1 …….1 0 0 0 0 0] is discarded.



The following part is not completely implemented, for now we are able to classify the intention of the user in a finite number of possible features.
Taking the same example “when this plant will bloom ? ” and change “this plant” part with a specific plant that is present in the knowledge base of the agent, in example “ Salice”, we obtain a realistic question “when Salice will bloom ? ” .

	A	B	C	D	E
1	KEY	NAME	FLOWERING	LAST-WATERING	NEXT-CROP
2	115254	Olmo	October	21/01/2021	null
3	121517	Salice	June	23/01/2021	null
4	225486	Apple tree	May	27/01/2021	June

note:
we are supposing that is present only a “Salice” instance in the DB, if this is not true the agent must reply with a question in order to clarify what is the target of the question.

This sentence “when Salice will bloom ? ” for the above explanation is correctly classified with a CHECK intention over a FLOWERING feature a plant, this plant now **is not generic.**

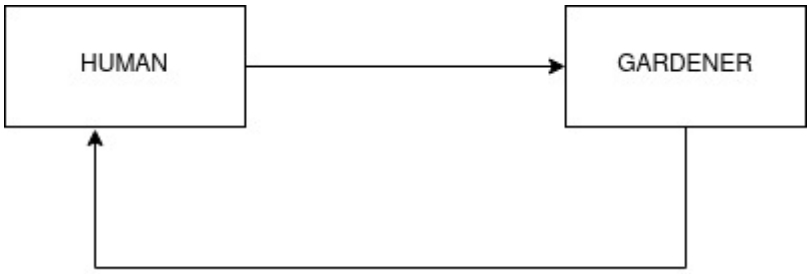
We can use **POS** tagging to extract this information from the sentence, discarding all the verbs and preposition, and checking if “Salice” is our data base, in a positive case we can return the correspondent answer composed by a predefined part “The Salice will bloom in” + an information extracted from the DB “June” = “The Salice will bloom June”
Finally we have to convert this text sentence an audio file in mp3 format and then play this.

```
def text_speech(text):
language='en'
myobj = gTTS(text=text, lang=language, slow=False)
myobj.save("file.mp3")
os.system("mpg321 file.mp3")
os.remove("file.mp3")
return text
```

The main drawback of this project is that this system is not able to see if an answer of the user to a question of the agent is coherent, for now is only able to see if is related one of the possible intentions specified.

Manage the Initiative

In a fist implementation the dialog follow the rule **stimulus-reaction** when the user is always providing a stimulus and for this the initiative of the dialog is always taken by him, this is not realistic and the dialog soon become boring.



To overcome this problem the Gardener, our agent, must be able to take the initiative, it react to the user stimulus but with a probability p instead waiting for a new stimulus, hence it aims to ask him questions or to stimulate the user to new interactions with its abilities.