

NLP Assignment 1 - A Simple Chatbot
Virtual Shopping Assistant for "MARVEL Costume Store"

Mhd.Shadi Ghasan Hasan

April 2021

Contents

0.1	Introduction	2
0.2	Problem Background	2
0.3	Algorithm	3
0.4	Results	6
0.5	Conclusion	7

0.1 Introduction

Chatbots, as the name suggests, are machines that are able to communicate and respond to humans via natural language. This can be done through text or speech. In fact, many people in recent years have interacted with one on their mobile phones or when contacting some business through social media. Virtual assistants, such as Siri, Google Assistant, or Alexa, are just examples of these machines.

The development of chatbots has started with ELIZA that was created by Joseph Weizenbaum. ELIZA played the role of a psychotherapist. The program was able to respond to users' inputs by matching patterns and looking for keywords in the input, then creating a response accordingly. Today's chatbots and virtual assistants can do much more than that of course. With advancements in computing power, Natural Language Processing techniques, and Machine Learning, chatbots and virtual assistants have entered many industries and are already substituting humans in customer service and other conversation-based roles.

This report introduces JARVIS; a Virtual Shopping Assistant for an imagined "MARVEL Costume Store". The bot was named JARVIS after Iron Man's natural language virtual assistant in the MARVEL Cinematic Universe. In the movies, JARVIS is a rather very intelligent machine, capable of running businesses for his owner as well as managing security. Our JARVIS on the other hand, is much more limited to an ELIZA-style pattern matching chatbot that is given the responsibility of helping MARVEL fans choose a superhero outfit.

0.2 Problem Background

As explained in the previous section, JARVIS is intended to play the role of a shopping assistant. Therefore, the bot is expected to be able to detect keywords, phrases, and questions that are common in a shopping scenario. Once a key pattern has been found, the bot should randomly provide an appropriate response or encourage the user, in case no pattern was found, to say something else that contains a key pattern. In other words, the bot should be able to follow ELIZA's footsteps in matching and responding to key words or key phrases.

Besides pattern matching and random choice of response, the bot should react to some phrases by echoing the input line back to the user. However, in this case, pronouns should be reversed to create a proper conversation. An example of this could go as follows:

USER: I love my dog!
BOT: Why do you love your dog?

As seen in the previous example, the bot detected some key phrase (*e.g. I love*) and replied with the same input line by echoing it back after converting "I" to "you" and "my" to "your".

Finally, the bot should demonstrate some level of intelligence. There are many advanced features that market-ready chatbots and virtual assistants enjoy nowadays. JARVIS will be able to recognize requests for superhero outfits that are **not** available in the store, and propose something else instead. This allows JARVIS to practice some kind of world knowledge.

To summarize, JARVIS's objectives are: A detection of keywords and/or key phrases with appropriate responses, a random generation and selection of answers, an intelligent continuation of the dialogue in case no keyword or key phrase was detected, a reversal of pronouns in the dialogue (*e.g.* "my" - "your"), and at least one intelligent feature, such as a level of world knowledge.

Depending on the objectives mentioned above, a chat with JARVIS should look something like this:

```
USER: Hi!
JARVIS: Hello!
USER: What's your name?
JARVIS: My name is JARVIS. How can I help you?
USER: I love MARVEL movies!
JARVIS: It's interesting that you love MARVEL movies!
USER: Turkish coffee is the best kind of coffee.
JARVIS: How does that affect your choice of a superhero outfit?
USER: I'd like to buy an Ant Man outfit.
JARVIS: I'm sorry! Ant Man isn't available. How about Iron Man?
```

In the first two input-response pairs, JARVIS would be finding keywords and generating an appropriate reply. Such a reply shouldn't be fixed and repeated every time, but variable and somewhat random. When the talk shifted to coffee, something that JARVIS wouldn't be programmed to recognize, the reply brought back the superhero topic. It would also be able to echo back some lines (*e.g.* "I love MARVEL movies!") after swapping pronouns to a more suitable case. Finally, it would be able to recognize Ant Man as a superhero that is not available, and choose an available alternative. This means, that JARVIS would have enough world knowledge to comprehend the concept of a superhero.

0.3 Algorithm

It has been explained that JARVIS will be a pattern matching chatbot. The basic principle behind JARVIS's algorithm is the exploitation of regular expressions in order to discover the intent of the user behind every input line. Once the user's intent was found, JARVIS has some ready made answers for those intents that can randomly choose from. Algorithm 1 shows how this functionality, and more, were implemented.

Before diving into the algorithm, the variables *available_items*, *patterns*, and *responses*, that the algorithm relies heavily on, shall be defined. *available_items* is a simple list of the MARVEL superhero costumes that customers can choose

from. *patterns* is a dictionary with key-value pairs that map a number of patterns (*i.e. values*), as regular expressions, to possible user intents (*i.e. keys*). The last important variable, *responses*, is also a dictionary, but with its values being the ready made responses that the bot can use; and each group of responses mapped to a key that represents the user's intent. The *patterns* and *responses* dictionaries allow the bot to check for a match between its patterns, defined in the *patterns* dictionary, and the input line. Consequently, discovering the user's intent, and using that intent to access the right group of responses in the *responses* dictionary and randomly outputting one of them.

It is important to have a closer look at some of the key-value pairs in *patterns*, and *responses* as they are crucial for fulfilling some of JARVIS's core objectives, in particular echoing back user lines after swapping pronouns, continuing the dialogue if no pattern resulted in a match, and recommending an available outfit when an unavailable one was selected. To achieve the latter, all MARVEL heroes were included as values for the key "*hero_choice*" in *patterns*. This allows JARVIS to realise that the user has chosen a hero outfit; whether available or not. The response will be created after consulting the *available_items* list, then indexing the *responses* dictionary using the keys "*hero_choice_available*" or "*hero_choice_unavailable*" accordingly. Note that, the selected hero will be mentioned in the response, along with a random recommendation in the case of an unavailable choice. Moreover, if no match was found between the input line and the defined patterns, JARVIS has a special key in *responses* named "*default*" for this case. It results in outputs that encourage the user to talk about their favourite hero; thus, a higher probability of a match in the next input. Finally, if the discovered intent turned out to be "*reg_chat*", JARVIS includes a great proportion of the input line in its reply after changing pronouns (*e.g. "my" - "your"*).

Algorithm 1 JARVIS's Algorithm

```

1: available_items  $\leftarrow$  list of available items
2: patterns  $\leftarrow$  dictionary of regular expressions mapped to intent labels
3: responses  $\leftarrow$  dictionary of bot responses mapped to intent labels

4: function REPLACE-PRONOUNS(sentence)
5:   if "me" was found in sentence then change "me" to "you"
6:   else if "my" was found in sentence then change "my" to "your"
7:   else if "you" was found in sentence then change "you" to "me"
8:   else if "your" was found in sentence then change "your" to "my"
9:   end if
10:  return sentence
11: end function

12: function MATCH-INTENT(message)
13:  matched_intent  $\leftarrow$  None
14:  echo_line  $\leftarrow$  None
15:  for all intent, pattern in patterns do

```

```

16:     if pattern was found in message then
17:         matched_intent  $\leftarrow$  intent
18:         if intent was "reg_chat" then
19:             echo_line  $\leftarrow$  REPLACE-PRONOUNS(part of message)
20:         else if intent was "hero_choice" then
21:             echo_line  $\leftarrow$  name of hero in message
22:         end if
23:     end if
24: end for
25:     return matched_intent, echo_line
26: end function

27: function RESPOND(user_line)
28:     user_intent, echo_line  $\leftarrow$  MATCH-INTENT(user_line)
29:     if user_intent was "hero_choice" then
30:         if echo_line was in available_items then
31:             bot_line  $\leftarrow$  RANDOM(responses["hero_choice_available"])
+echo_line
32:         else
33:             bot_line  $\leftarrow$  RANDOM(responses["hero_choice_unavailable"])
+echo_line + RANDOM(available_items)
34:         end if
35:     else if user_intent was "reg_chat" then
36:         bot_line  $\leftarrow$  RANDOM(responses[user_intent]) + echo_line
37:     else if user_intent was None then
38:         bot_line  $\leftarrow$  RANDOM(responses["default"])
39:     else
40:         bot_line  $\leftarrow$  RANDOM(responses[user_intent])
41:     end if
42:     return bot_line
43: end function

44: function SEND(input_line)
45:     PRINT(input_line)
46:     reply  $\leftarrow$  RESPOND(input_line)
47:     PRINT(reply)
48: end function

```

Algorithm 1 has four functions. **REPLACE-PRONOUNS** takes a string as an argument, looks for pronouns, and swaps them to something more suitable as an output. **MATCH-INTENT** also takes in a string, iterates through the patterns in *patterns*, and returns the key of the pattern that results in a match as *matched_intent*, or None otherwise. A part of the string could also be returned as *echo_line*; that is, the selected hero if the intent was found to be "hero_choice", or the pronouns-swapped input line if the intent was "reg_chat". **RESPOND** creates a response line for JARVIS based on the intent returned

form MATCH-INTENT. Through nested if-else statements, it returns a default line when no intent was matched, responds to regular chatting lines with the pronoun-swapped input line along with some additions to create a conversation, and informs the user if their selected outfit is available and suggests something else if not. The function also responds to greetings and other possible input lines in a shopping scenario. **SEND** prints the user's line on the screen as well as the response returned from RESPOND.

0.4 Results

JARVIS's algorithm was tested against a group of inputs to create a typical shopping chat between a MARVEL fan and JARVIS. Figure 1 shows a snapshot of this test chat.

By carefully inspecting the chat, it can be seen that the responses for the first, second, and third inputs (*i.e.* "Hi!", "Who is this?", and "I'm interested in buying a marvel outfit") were created by the pattern matching mechanism explained earlier; that is, discovering the intent and outputting a random response accordingly. The same mechanism was used to respond to the lines where the user asked for the price, thanked, and farewelled the bot.

When the user inquired about Spider Man (a MARVEL hero whose costume is not available), JARVIS discovered that the intent was "hero_choice", consulted *available_items*, and chose a response from *responses* with the key "hero_choice_unavailable". However, in the case of Iron Man, the key used for indexing *responses* was "hero_choice_available", as Iron Man's costume is included in *available_items*. Note that in both cases the hero's name is extracted from the input string and included in the output. Furthermore, in the case of Spider Man, JARVIS randomly suggested Hulk from *available_items* as a substitute.

In the case of the input line "You know what! I like all marvel heros. They're my favourite!", JARVIS responded with "I understand that you like all marvel heros. they're your favourite!". JARVIS concluded that this input falls under the intent "reg_chat" and replied with a part of the input line but changed the pronouns from the 1st person case to the 2nd.

The "default" case in *responses* was used to generate a reply to the input "Dogs are fun! Cats are not.". This line has nothing that is relevant to superheroes, shopping, or casual conversational lines. As a result, JARVIS did not recognize the string as falling under any of its predefined user intents. In other words, the *user_intent* value is None. The resulting reply asked about the relevance of this input to the user's favourite hero. While there is clearly no relevance in this particular input, the reply shifts the discussion back to the superhero topic in hopes for a new input that would result in a match.

```

You: Hi!
JARVIS: Good day! How can I help you today?
You: Who is this?
JARVIS: My name is JARVIS. You might know me from Iron Man. I'm now here to help YOU be a superhero. How can I help?
You: I'm interested in buying a marvel outfit
JARVIS: 'Captain America', 'Hulk', 'Thor', 'Black Widow', and 'Iron Man' are all available in our store.
You: Well! I like Spider Man!! Do you have that?
JARVIS: I'm afraid I can't help you with that as spider man is currently not available. How about hulk?
You: No, thanks! How about Iron Man? is that available?
JARVIS: My mind just BLEW when you said iron man because we have that available for you!
You: SUPER!!!! How much is it?
JARVIS: We charge 20 000 for any costume.
You: You know what! I like all marvel heros. They're my favourite!
JARVIS: I understand that you like all marvel heros. they're your favourite!
You: Dogs are fun! Cats are not.
JARVIS: You do know that you're keeping me from SAVING THE WORLD right?! Is that related to your favourite hero?
You: Thanks for your help!
JARVIS: I'm always here if you need me.
You: Goodbye!
JARVIS: Goodbye!

```

Figure 1: Test Chat with JARVIS

0.5 Conclusion

In conclusion, JARVIS has been able to meet all of the predefined objectives. 'He' detected keywords and key phrases, generated responses randomly based on inputs, proceeded with the conversation when no match was found, converted pronouns to create outputs from the user's words, and demonstrated some world knowledge by distinguishing superheroes and linking them to available items.

JARVIS is a rather simple chatbot. More advanced features can be further developed. For example, **Machine Learning** can be used to train JARVIS on large sets of users' lines along with their intents as labels. This can open the horizon for a wider spectrum of intent recognition than what can be achieved through regular expressions. **Statefulness** is another area with great potential for improvement. It can allow JARVIS to have different replies depending not only on the user's line, but on the respective state of the conversation. Converting user lines to **Database queries** can also be helpful when the simple *available_items* list is replaced by a database of costumes with lots of entries, different prices, sizes, and colors.