



ASHESI UNIVERSITY

BUILDING A KEYWORD-BASED PSYCHIATRIST CHATBOT USING PROLOG

Ewura Ama Sam, Alice Adwubi Asiedu, Daniel Byiringiro, Philip Amarteyfio

19th April 2023

Introduction

Most people do not have the luxury of booking appointments with psychiatrists. Others do not feel comfortable sitting in front of one. Whatever the case, people need to speak to professionals occasionally to address pressing mental health issues as well as things that bother them. We aim to help solve this problem by developing an automated psychiatrist that interacts with its users and provides recommendations, and specific feedback peculiar to each to each user and their circumstance. This automated psychiatrist program is relevant for introverts who do not feel comfortable being in front of someone to share their problems with. It is also beneficial for people who do not live near a psychiatrist or are too busy to meet up with one. This project focuses on students and hence, has a niche for student engagement. However, it can still be used by different personas. This document aims to inform the reader about the specific method we used to tackle the problem, the results, including the codes implemented, and lastly, discussions, findings and key insights obtained.

Methodologies Used in Solving the Problem

Stage One: Identifying common keywords that appear in interactions with psychiatrists. Some of the identified keywords are assignments, deadlines, stress, anxiety, depression, relationships, mid sem, final exam, breakup, church, religion, work, substance abuse, drugs, sexual assault, bullying, violence, noise, suicide, and drug abuse.

Stage Two: Based on the keywords, possible responses were formed for inputs from the user that contains the above keywords. The responses the system gives include “How do you approach assignments and manage your time effectively”, “Do you face any challenges when working towards deadline”, and “Have you noticed changes in your mood while working on assignments.”

Stage Three: Translating the ideas generated to code. After coming up with the keywords, we needed to write codes such that it takes the user's input and looks for the presence of any of the specified keywords in the statement. Based on the keyword(s) found in the user's input, a predetermined response is generated for the user.

Explanation of Concepts Used Along with Explanation of Codes

1. Association Lists. An association list refers to a collection of unique keys that are associated with values (O'Keefe, Costa, Triska, 2021). We made the keywords we identified the keys, and the corresponding sentences were the values. When the user inputs their text, the program checks whether any of the keys is a substring of what the user types. If any of the keys is a substring of the user's text, the corresponding response for that keyword is printed. In the code below, the keys for the association are the ones in the square brackets, namely, mother, mom, mama, and mummy. These keywords are associated with the value, "Tell me more about your relationship with your parents".

```

% This code is a simple prolog implementation of a psychiatrist chatbot
%
%
% @version 1.0
% @since April 2023

keywords([[mother, father, mom , dad, mum, mummy, daddy, papa, mama]-"Tell me more about your
relationship with your parents",
["don't like him", "don't love them", "don't like her", "don't like them" , "annoying me", "makes
me angry"]- "Do they do anything in particular to make you feel this way?",
["love him", "love her", "love them", "in love with", "like her", "like them", "like him"]- "Sounds
like they mean a lot to you! Good relationships are very healthy. Tell me more about them",
[brother, sister, siblings]- "What is your relationship with your siblings like?",
["i am doing great"]- "Awesome. I am all ears.",
["i'm fine", "i'm good" , "i'm well"]- "Glad to hear it! So what's going on with you lately",
[friend, friends, bestie] - "Tell me more about your friends",
[family, aunt, auntie, uncle, cousins] - "Oh tell me more about your extended family",
[stress, anxious, anxiety, worry, worried, afraid, stressed]- "I am sorry to hear that. can you
share what makes you feel this way?",
[depressed, depression]- "What do you think is causing your depression?",
[sad, unhappy] - "I am sorry to hear that. can you share what makes you feel this way?",
[assignment, homework, assignments]- "Have you noticed any changes to your mood when working on
assignments?",
[work, career] - "What is going on at work?",
["i hate school", "class is boring", "i don't understand"]- "I know it's difficult but having a
more positive mindset is everything and can be more helpful",
[anger, angry]- "What makes you feel angry?",

```

Fig 1.0 illustrates the definition of the association list for keywords

```

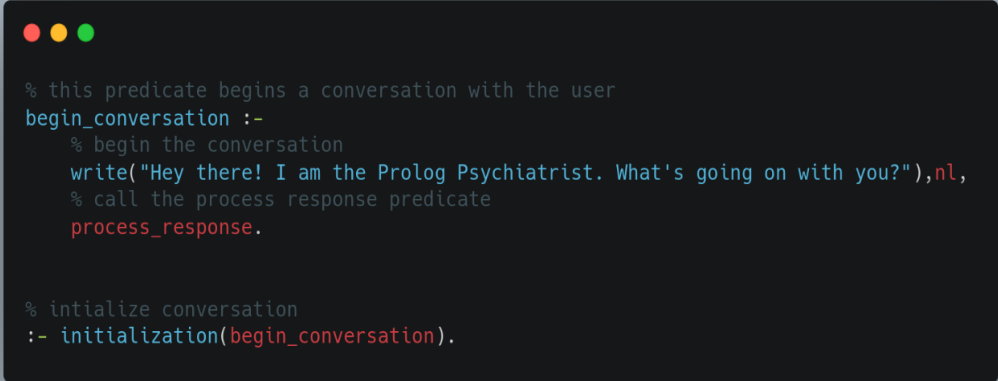
[stupid] - "Why do you think you're stupid?",
[deadline, deadlines] - "What do you think cause you to work when deadlines are near?",
[hate] - "Why do you hate it?",
["good job", "well done"]- "Thank you! Let's continue:",
[partner, spouse]- "What's going on in your marriage?",
[drugs, alcohol, weed, cocaine, smoking, "i like being high"]- "At what times do you feel the urge
to take these things?",
["life is pointless", "take my life"]- "What makes you feel this way?. Also you're very much
appreciated and you can always reach out for help. ",
[suicide, "kill myself", "i want to die" ] - "Death is not the solution, have you tried reaching out
to friends or loved ones ?",
[harrassed, harrass]- "Have you sought for help or reported any such incidents?",
["sexual harrassment", "forced himself on", "forced herself on", "not consensual", "i didn't
consent" ]- "Can you tell me more about the situation or experience where you felt that you didn't
consent or someone forced themselves on you?",
[inappropriate] - "Please explain what happened",
[anorexia, bulimia, "eating disorder"]- "Can you tell me more about your relationship with food and
how it affects your daily life?",
[insomnia, sleep, tired, fatigue]- "How many hours of sleep do you get each night? Have you noticed
any changes in your sleep patterns?",
[medication, prescription, pills]- "Are you currently taking any medication? If so, how do you feel
it is affecting you?",
[trauma, ptsd, abuse, assault]- "It takes a lot of courage to speak up about trauma. Can you tell
me more about what happened and how it has affected you?",
["self-harm", cutting, "suicidal thoughts", "suicide attempts"]- "I'm here to support you through
this difficult time. Have you ever talked to a therapist or counselor about these thoughts and
feelings?",
["panic attack", "anxiety attack"]- "I'm sorry to hear that you're experiencing panic attacks. Can
you tell me more about what triggers them?",
[ocd, "obsessive-compulsive disorder"]- "OCD can be very distressing. Can you tell me more about
your compulsions and how they affect your daily life?",
[phobia, fear, anxiety]- "Facing your fears can be challenging, but it's important to take small
steps towards overcoming them. What steps have you taken so far?",
[loneliness, isolated, "social anxiety"]- "Feeling lonely or isolated can be difficult, but there
are ways to connect with others. Have you considered joining any social groups or clubs?",
["relationship problems", "marriage problems"]- "Relationship problems can be very distressing. Can
you tell me more about what's been going on in your relationship/marriage?",
[hello, "good morning", "good morning", "good afternoon", "good evening"]- "Hey there, what's up?"
])).

```

Fig 1.1 A continuation of the definition of keywords

2. Predicates. Predicates are methods that contain arguments and return Boolean values (EDUCBA, 2023). Some of the predicates used in this project are inbuilt whilst others were created by us to facilitate our program. Below are the predicates we created and they implement inbuilt predicates.

- a) Initialization predicate: This is a predicate we created to initialize the program. The purpose of the predicate is to call another predicate called `begin_conversation`.
- b) `Begin_conversation` predicate: This predicate uses the `write` predicate to print the initial words of the psychiatrist to the user. It then calls the `process_response` predicate. The `begin_conversation` predicate is shown below.



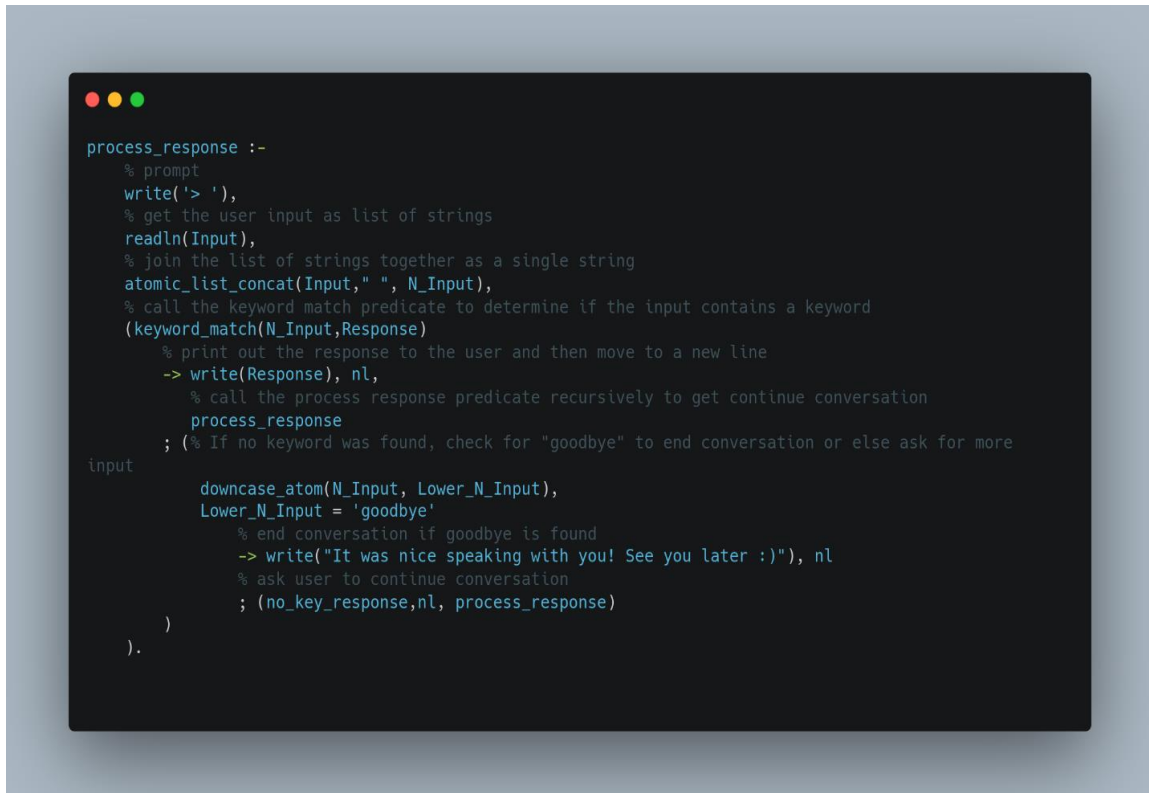
```
% this predicate begins a conversation with the user
begin_conversation :-
    % begin the conversation
    write("Hey there! I am the Prolog Psychiatrist. What's going on with you?"),nl,
    % call the process response predicate
    process_response.

% initialize conversation
:- initialization(begin_conversation).
```

Fig 1.2 Illustration of the `begin_conversation` predicate

- c) `Process_response` predicate: This predicate processes the input of the user and works with other predicates to fetch the appropriate response to be given. The code begins by printing the greater than sign to show the entry point of the program using the built-in predicate, `write`. It then takes the user's input using the `readln` predicate. What the user

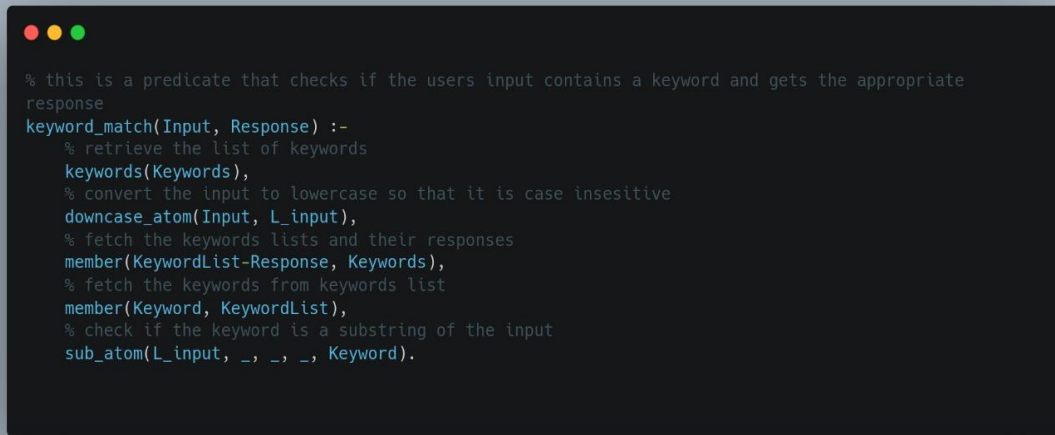
inputted is concatenated along white spaces. This is to make the whole sentence(s) one string. The purpose of making it one string is elaborated in the `keyword_match` predicate. The concatenation is done using the `atomic_list_concat` predicate. It takes the variable name of the sentence to be concatenated (Input), what it is concatenating along (In this case, whitespaces), and the variable that would store the output (N_Input). It then calls the `keyword_match` predicate and passes the concatenated string to it and receives the response of the program. The response is then displayed to the user. How the `keyword_match` predicate works will be explained further in the code. In the `process_response` predicate, we defined conditional statements for certain special scenarios. If the user's input is only "goodbye", the program would print the string "It was nice speaking with you. See you later.", then the program terminates. However, if the sentence simply contains 'goodbye', the program should not terminate. This is to make 'goodbye' the keyword to end the conversation. The `process_response` predicate works recursively, meaning, it calls itself within itself. Recursion was important here in order to keep the conversation going whenever the keyword for termination, that is, goodbye, is not detected.



```
process_response :-
    % prompt
    write('> '),
    % get the user input as list of strings
    readln(Input),
    % join the list of strings together as a single string
    atomic_list_concat(Input, " ", N_Input),
    % call the keyword match predicate to determine if the input contains a keyword
    (keyword_match(N_Input,Response)
    % print out the response to the user and then move to a new line
    -> write(Response), nl,
    % call the process response predicate recursively to get continue conversation
    process_response
    ; (% If no keyword was found, check for "goodbye" to end conversation or else ask for more
    input
    % end conversation if goodbye is found
    % ask user to continue conversation
    % no_key_response,nl, process_response
    ).
    ).
```

Fig 1.3 Illustration of the process_response predicate

- d) **Keyword_match predicate:** This predicate takes an input and returns a response. In this program, it takes the concatenated string, N_Input when it is called in the process_response predicate. The keyword_match predicate checks whether any of the keywords in the association list defined is a substring of the input. If there is a keyword found in the input, the value corresponding to that keyword is displayed as the response of the psychiatrist. The value in this case is a prewritten response that is defined in the association list and corresponds to the identified keyword. If there are multiple keywords in the input, the program considers the first one defined in the association list. To account for a proper comparison between the keywords and the input, we convert the input to lowercase letters. This is because the keywords defined are all in lowercase. This is to allow consistency and efficiency.



```

% this is a predicate that checks if the users input contains a keyword and gets the appropriate
response
keyword_match(Input, Response) :-
    % retrieve the list of keywords
    keywords(Keywords),
    % convert the input to lowercase so that it is case insensitive
    downcase_atom(Input, L_input),
    % fetch the keywords lists and their responses
    member(KeywordList-Response, Keywords),
    % fetch the keywords from keywords list
    member(Keyword, KeywordList),
    % check if the keyword is a substring of the input
    sub_atom(L_input, _, _, _, Keyword).

```

Fig 1.4 Illustration of the keyword_match predicate

- e) No_key_response predicate: Based on the predicates used above, in a situation where a user's input does not contain any of the defined keywords, the conversation would cease as the program would not find a keyword to map to a response. The no_key_response works with the no_response_keywords to generate a random response. This is done by storing a list of responses in the no_response_keywords. Each response in the list has an index. When the user types something that does not correspond to any keyword, a random number between 1 and the length of the list. The response that corresponds to the generated number is printed to the user as the psychiatrist's response.


```

1)).

% keywords for if theres no response
no_response_keywords(["I see, please continue..","Interesting. Tell me more", "Okay. Go on", "Okay what
else"])).

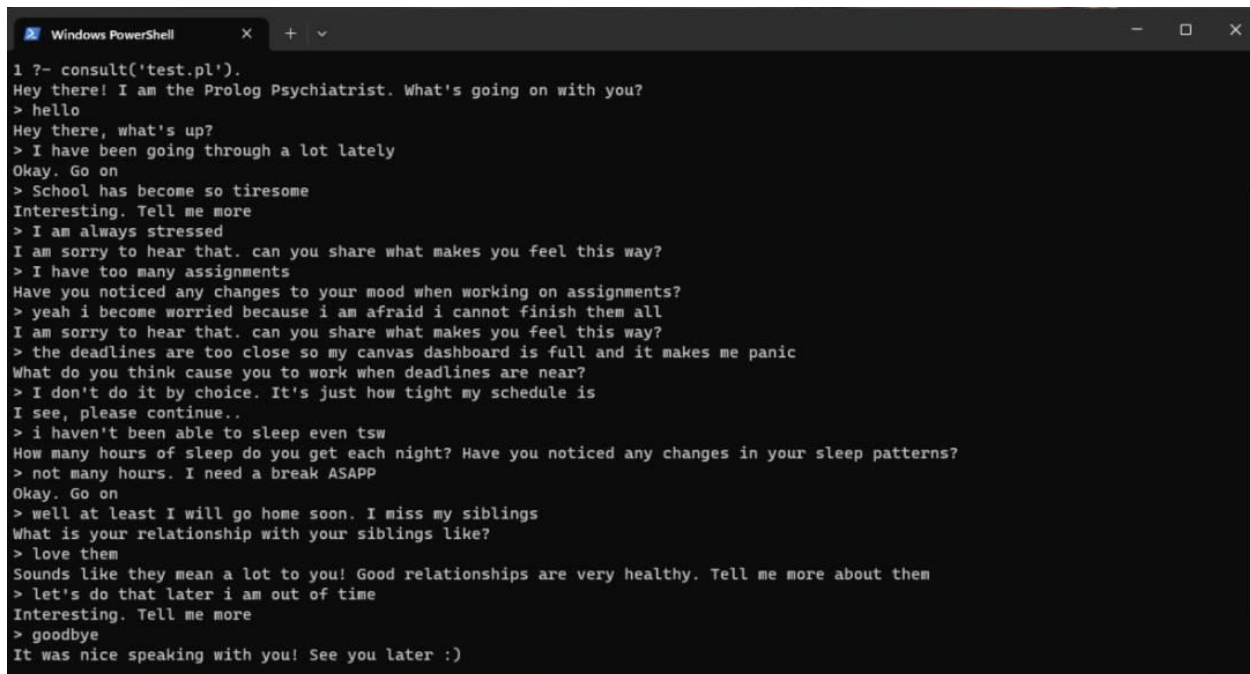
% predicate to get a response if no keyword is provided
no_key_response :-
    % Get the list of strings
    no_response_keywords(List),
    % Get the length of the list
    length(List, Length),
    % Generate a random index between 1 and the length of the list
    random(1, Length, Index),
    % Get the string at the random index
    nth1(Index, List, Response),
    % print the response
    write(Response).

```

Fig 1.5 Illustration of the no_key_response predicate as well as the no_response_keywords list

Results and Findings

After we set up the code, we tested it with multiple conversations to observe the responses the program gives and make the necessary changes. Below is a sample conversation we tested the program with. Some keywords the user entered include; “stressed”, “assignments”, “deadlines”, “sleep”, and “siblings”. Based on the keywords, the program generated specific responses and in cases where the user entered a random statement that does not match any keywords, the program responded with statements that encourage the user to give more feedback, for example: “Interesting, tell me more”. The program gives some advice and suggestions to solve the user’s predicament.

A screenshot of a Windows PowerShell window with a dark background. The window title bar shows 'Windows PowerShell' and standard window controls. The terminal displays a conversation between a user and a program. The program's responses are indented. The conversation starts with the user running a command, followed by a series of questions and answers about school stress, assignments, sleep, and siblings. The user ends the conversation with 'goodbye', and the program responds with a farewell message.

```
1 ?- consult('test.pl').
Hey there! I am the Prolog Psychiatrist. What's going on with you?
> hello
Hey there, what's up?
> I have been going through a lot lately
Okay. Go on
> School has become so tiresome
Interesting. Tell me more
> I am always stressed
I am sorry to hear that. can you share what makes you feel this way?
> I have too many assignments
Have you noticed any changes to your mood when working on assignments?
> yeah i become worried because i am afraid i cannot finish them all
I am sorry to hear that. can you share what makes you feel this way?
> the deadlines are too close so my canvas dashboard is full and it makes me panic
What do you think cause you to work when deadlines are near?
> I don't do it by choice. It's just how tight my schedule is
I see, please continue..
> i haven't been able to sleep even tsw
How many hours of sleep do you get each night? Have you noticed any changes in your sleep patterns?
> not many hours. I need a break ASAP
Okay. Go on
> well at least I will go home soon. I miss my siblings
What is your relationship with your siblings like?
> love them
Sounds like they mean a lot to you! Good relationships are very healthy. Tell me more about them
> let's do that later i am out of time
Interesting. Tell me more
> goodbye
It was nice speaking with you! See you later :)
```

Fig 1.6 A sample conversation we had with the program

From these conversations, we found that when the user enters goodbye at any point in the conversation, the program believes the user wants to end the conversation and replies with a goodbye as well. To resolve this problem, we modified the code to include a set of directives for the user when they begin the interaction. These directives tell the user to enter goodbye when they want to end the conversation and also what the greater than symbol means they can begin to enter their input. Below is a sample of the modified version.

```
Windows PowerShell
PS C:\Users\Philip Amarteifio\OneDrive - Ashesi University\Desktop\DM Project\DiscreteStructuresFinalProject> swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- consult('test.pl').
Hello! Welcome to Prolog Psychiatrist version 1.0
Rules
=====
When you see the '>' you can enter input
To end the conversation at any point type 'goodbye'
Let's Begin
=====

Hey there! I am the Prolog Psychiatrist. What's going on with you?
> hello there
Hey there, what's up?
> goodbye
It was nice speaking with you! See you later :)
true.

2 ?-
```

Fig 1.7 An illustration of the updated program which displays conversation rules

Discussion

Based on our findings, we realized that the program works according to the expectations and results we expected to obtain after building the program. Our expectations included that the user should be able to fully discuss their issues with the psychiatrist and receive feedback that is tailored towards helping them overcome their problems.

Also another expectation that we set was that the program should behave and respond similarly to a real-life psychiatrist.

The limitation we encountered was the program's extensiveness. The program, though based on a wide variety of pre-selected keywords, does not cover all the possible scenarios a user might want to discuss with the psychiatrist.

References

O'Keefe, Costa, Triska. (2021, January 16). *Prolog -- Manual*. SWI. <https://www.swi-prolog.org/pldoc/man?section=assoc>

Prolog predicate: How does prolog predicate work in Prolog: Examples. EDUCBA. (2023, April 5). <https://www.educba.com/prolog-predicate/>