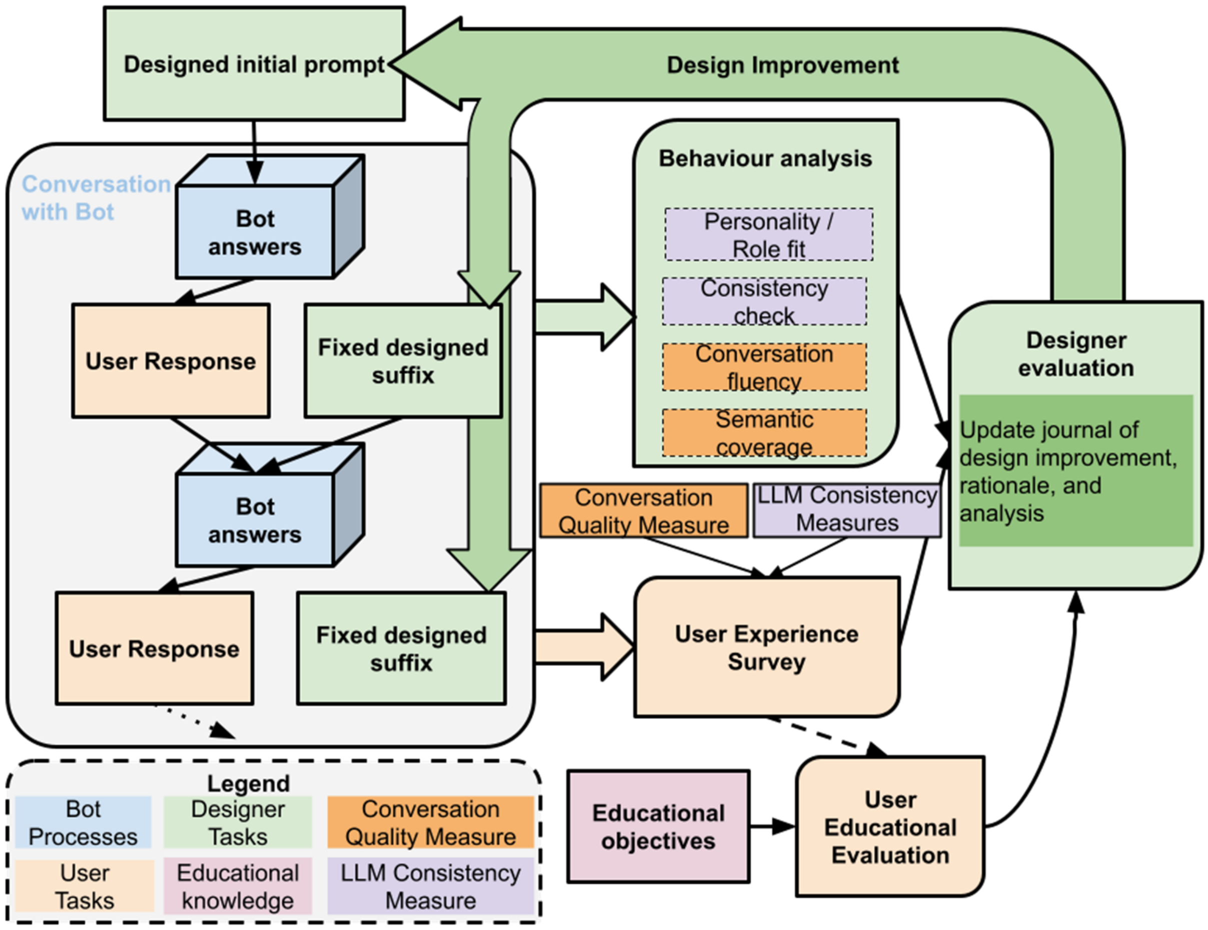
Rebuttal Supplementary Material

Guideline production process

Initial Prompt Design

For a previous experiment, documented in a previous publication (anonymized for review, 2023), we created an investigation framework to study ChatGPT's ability to produce mixed-turn educational conversations where the students can provide feedback and steer the conversation toward a desired topic as well as cover multiple learning objectives, tailor educational activities to different user characteristics (e.g., culture, age), and use different educational strategies and conversational styles. The investigation and design framework allows swift testing of new designs, in the forms of initial textual prompts and additional prefixes and suffixes, internally by the design team members using ChatGPT. Once a candidate design is found and internally tested it can be tested by naive users through a website that acts as an interface with ChatGPT, hides from the user the designed part of the input (initial prompt and suffixes), and collects the interactions. The interaction data collected are analyzed to provide additional insights and improve the prompt design.



As shown in Fig. 1, our investigation and design framework aimed to be applicable to different topics. Educational objectives with connected topics and surveys describe the relevant educational knowledge that can be provided by domain experts. This information allows for evaluating user educational needs. The results of the educational evaluation allow to define the elements that designers should integrate while revising the initial prompt as well as the additional fixed suffix and prefix. After these elements are designed, interactions between users and the chatbot are started providing the initial prompt only to the chatbot, hiding it from the user. This will initiate a conversation started by the chatbot. User responses will be surrounded by additional designed prefixes and suffixes. After the conversation, the users will fill out a User Experience Survey and an educational survey. The interactions are also analyzed by the designer to decide the next design iteration. When the design is internally tested no educational evaluation will be performed, but a coverage check and a qualitative interaction evaluation. In this condition, both the prompt and the other designed elements will be visible to the testers.

Guidelines and Feature Definition

We adopted three distinct approaches when it comes to types of features: one that we design from the ground up, based on our experience; one that is based on prompt patterns identified by White et al (2023 at https://arxiv.org/abs/2302.11382); and, finally, one that is suggested by ChatGPT itself, intended as a baseline. The guidelines from each of these approaches were given to learners as an experimental condition. These were further detailed in features that were considered easier to classify. Each guideline is comprised of multiple aspects, their descriptions, and related examples. Below the aspects of each guideline are listed, and subsequently the corresponding features are listed.

Task-specific guidelines

The first set of guidelines was derived from our exploratory conversations with ChatGPT, documented in a previous publication (anonymized for review, 2023). Starting from these interactions, we defined the ones that we considered successful and ones that we did not. Subsequently, from this distinction, we propose several dimensions of a successful interaction:

- Topic - concise or broken down (e.g. in enumerating components)
- Goal(s) (countable number of distinct goals) - what does the user request
- Form of response (positive and/or negative) - i.e. positive are what to include in responses, negative are what to avoid
- Role (me/you) - explicit reference to any of the roles in conversation, i.e. identification of conversational partners
    - Role-play - define roles for both AI and user
        - Context - provide additional contextual information for AI, user, or environment
- Meta
    - Process-related - e.g. repeat procedure, continue, etc.
    - Constraints - avoid something, maintain something else
- Descriptors (countable of distinct descriptors for each of) - of any of topic, goal, form, roles, etc.
    - Adjectives - the simplest form of descriptors
- Etiquette:
    - Greeting
    - User intention (desire) - does the user express their goal as a wish
    - Polite request
- Sentence structure: simple vs compound - think simple English
- Repetitions - how many times certain dimension is repeated

Through the annotation process and feature selection and rewording phase, the final set of aspects for this guideline became:

1. Define the Chatbot role and how it should be played
2. Describe the conversation process
3. Use simple sentences even if many
4. Ask to split the topics of conversation, implicitly or with explicit enumeration
5. Specify all the actions needed to direct the conversation

Guidelines derived from prompt patterns

Another set of features was derived from White et al (2023). It is based on prompt patterns, defined by the authors. Of these, we have identified as relevant to our task two, and the corresponding guidelines are:

1. Use the Persona Pattern
2. Use the Flipped Conversation Pattern
3. Game Play Pattern

The Game Play Pattern seemed to not impact the prompting efficiency. It was thus removed from the set of patterns(aspects) presented in the educational activity.

Guidelines suggested by ChatGPT

A third and final set of features was derived by asking ChatGPT for a suggestion. The suggestion of the LLM has the following features:

1. The Tone
2. Clarity and Specificity
3. One-Step Instruction vs. Multi-Step Instruction
4. Interactivity (Q&A)

Feature sets

The aspects of the guidelines were translated into one or more binary features whose description was derived from that of the corresponding aspect. Other encodings (e.g. countable) will be tested in the future.

1. **Task specific prompting guidelines**
   - 1.1. Topic - concise(1,0)
   - 1.2. Topic - broken down (1,0)
   - 1.3. 1 Goal (1,0)
   - 1.4. 2 Goals (1,0)
   - 1.5. >2 Goals (1,0)
   - 1.6. 1 Positive Form of response (1,0)
   - 1.7. Multiple Positive Form of response (1,0)
   - 1.8. Negative Form of response (1,0)
   - 1.9. AI role play (1,0)
   - 1.10. User role (1,0)
   - 1.11. Role form/context(1,0)
   - 1.12. Meta Process-related (1,0)
   - 1.13. Meta Constraints (1,0)
   - 1.14. 1 Descriptor (1,0)
   - 1.15. 2 Descriptors (1,0)
   - 1.16. > 2 Descriptors (1,0)

1.17. E_Greeting (1,0)
1.18. E_User intention (1,0)
1.19. E_Polite Request (1,0)
1.20. Simple sentence structure (1,0)
1.21. Complex sentence structure (1,0)

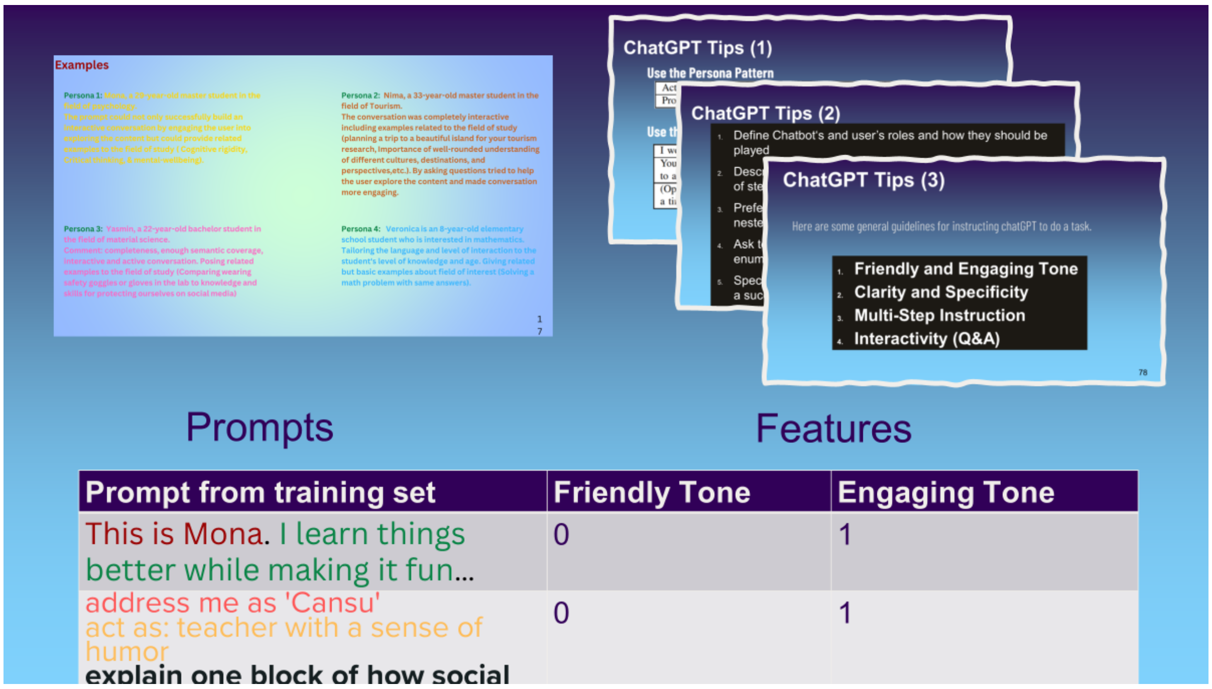
2. Prompting patterns

2.1. ActAsPersona- Persona Pattern (1,0)
2.2. ProvideOutPuts- Persona Pattern (1,0)
2.3. PatternOrder- Persona Pattern (1,0)
2.4. StrictSeparationRoleVsOutput- Persona Pattern (1,0)
2.5. AskMeQuestions-FlippedPattern
2.6. ConditionStop-FlippedPattern
2.7. Form-FlippedPattern:
2.8. Order1-FlippedPattern: Feature1 before Feature 2 before Feature 3
2.9. Order1-Simple: Feature1 before Feature 2 no Feature 3
2.10. Order2-FlippedPattern: Feature1 before Feature 3 before Feature 2
2.11. GamePlay Start// Teach me// Create a game//Have an interatarction//Have a Conversation
2.12. GamePlay Rules
2.13. GamePlay Order: Feature 1 before Feature 2
2.14. GamePlay Strict: Feature 1 before Feature 2 in separate statements

3. ChatGPT generated

3.1. Polite Language Tone of user (1,0)
3.2. Engaging interaction request from AI (1,0)
3.3. One-Step Instruction (1,0)
3.4. Multi-Step Instruction (1,0)
3.5. Specificity of topics (1,0)
3.6. Clearity of the instructions (1,0)
3.7. Interactivity (Q&A) (1,0)

Feature Annotations



Examples

Persona 1: Mona, a 29-year-old master student in the field of psychology.
The prompt could not only successfully build an interactive conversation by engaging the user into exploring the content but could provide related examples to the field of study (Cognitive rigidity, Critical thinking, & mental-wellbeing).

Persona 2: Nima, a 33-year-old master student in the field of Tourism.
The conversation was completely interactive including examples related to the field of study (planning a trip to a beautiful island for your tourism research, Importance of well-rounded understanding of different cultures, destinations, and perspectives,etc.). By asking questions tried to help the user explore the content and made conversation more engaging.

Persona 3: Yasmin, a 22-year-old bachelor student in the field of material science.
Comment: completeness, enough semantic coverage, interactive and active conversation. Posing related examples to the field of study (Comparing wearing safety goggles or gloves in the lab to knowledge and skills for protecting ourselves on social media)

Persona 4: Veronica is an 8-year-old elementary school student who is interested in mathematics.
Tailoring the language and level of interaction to the student's level of knowledge and age. Giving related but basic examples about field of interest (Solving a math problem with same answers).

1
7

ChatGPT Tips (1)

Use the Persona Pattern

ChatGPT Tips (2)

1. Define Chatbot's and user's roles and how they should be played

ChatGPT Tips (3)

Here are some general guidelines for instructing chatGPT to do a task.

1. Friendly and Engaging Tone
2. Clarity and Specificity
3. Multi-Step Instruction
4. Interactivity (Q&A)

78

Prompts

Features

Prompt from training set	Friendly Tone	Engaging Tone
This is Mona. I learn things better while making it fun...	0	1
address me as 'Cansu' act as: teacher with a sense of humor explain one block of how social	0	1

Once an initial set of features was defined, the successful and failing prompts tested during the initial design phase were annotated independently by 2 annotators.

Features and Annotations Revision

After the independent annotation phase was completed, critical situation (i.e., slight disagreement in a single cathegory) were menaged by involving a third party (the supervisor) in the evaluation of the specific cathegory. This allowed a rewording of the feature definition and a second phase of independent annotation.

Feature Selection

To avoid redundant educational material, keep the lenght of the educational activity under control and avoid presenting ineffective suggestions, after the annotation, a feature reduction operation was performed. The features were selected if they were discriminative based on the odd ratio value and there was a high level of agreement between annotators in the final independent annotation revision. Moreover, the features were only selected if either of these conditions was satisfied:

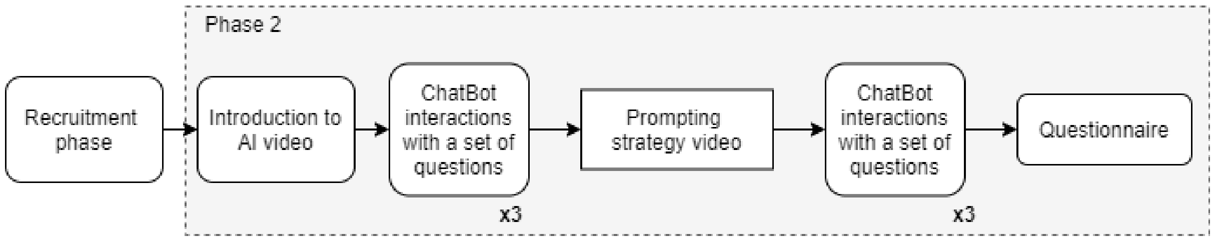
- Their annotations were distinct from other features.
- Their annotations overlapped other features but their description had a clearly different meaning.

Data collection process

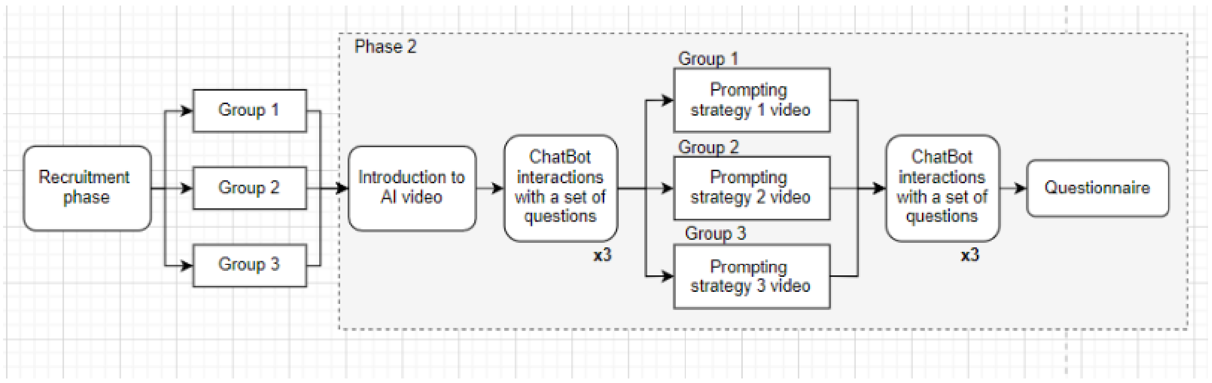
Initial pre-screening to select participants with limited LLM experience

An initial pre-screening survey was conducted to find participants whose experience with LLM was limited. We also asked participants for demographic information and attitudes toward AI.

Simple diagram:



Extended diagram:


Phase 2
Recruitment phase
Group 1
Group 2
Group 3
Introduction to AI video
ChatBot interactions with a set of questions
x3
Group 1
Prompting strategy 1 video
Group 2
Prompting strategy 2 video
Group 3
Prompting strategy 3 video
ChatBot interactions with a set of questions
x3
Questionnaire

Detailed procedure (requires rewording):

The study procedure is described below and represented in figure XX. At the very start learners completed a preliminary questionnaire and were separated in the three experimental conditions. The following learner interactions step was based on a previous pilot study (anonymized for review, 2023) in a school setting. Some necessary adaptations were made as in this case learners were recruited on the Prolific crowdsourcing platform.

The key to such adaptation was the creation of a questionnaire in the form of an educational activity, utilizing the Qualtrics survey platform. The questionnaire consisted of ten sections. The first section of questions was about learners' experience with generative AI tools. Then, participants were asked to watch a video providing an introduction to topics related to AI applications, LLMs, and human vs artificial intelligence. To confirm participants watched the video a question was included in the page, and its answer was concealed within the video.

In the subsequent chatbot interaction step, learners received a set of instructions to complete a task within the bespoke chatbot interface that was powered by the ChatGPT API. Before giving learners access to the chat interface, they had to answer two questions related to their expectations of the quality of input and output, produced respectively by them and the chatbot.

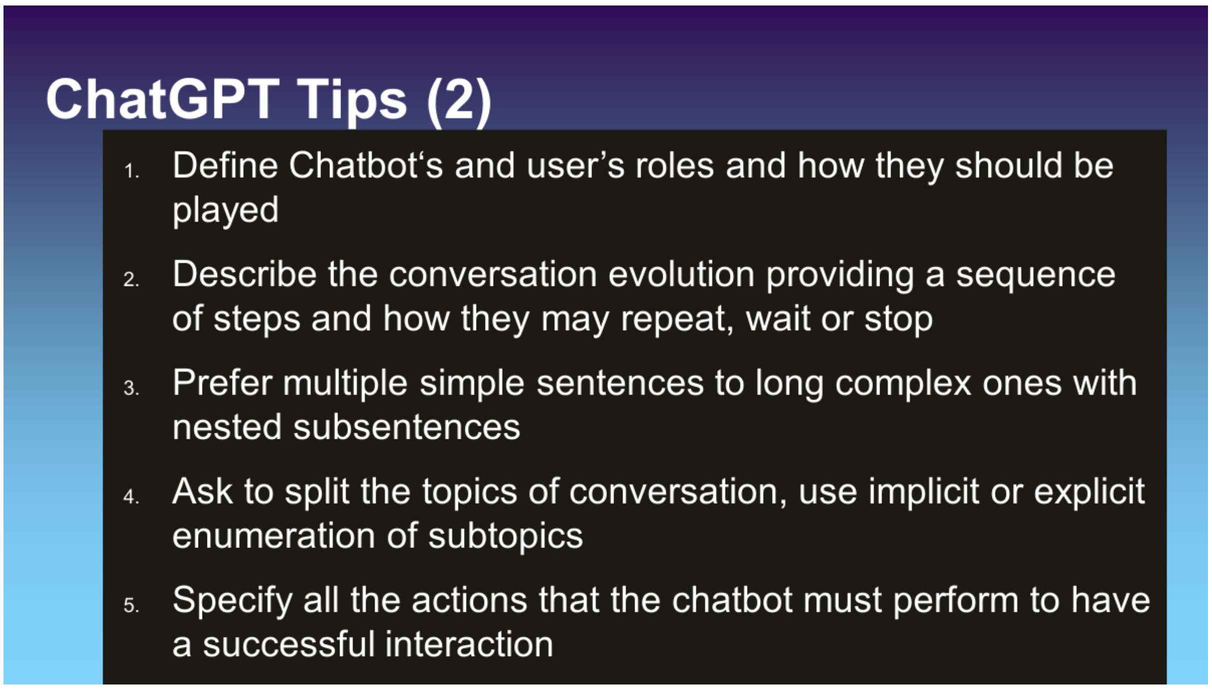
During the chat learners were asked to ask the chatbot to teach them on three social media topics. Among the examples provided with instructions was e.g. "explain in a friendly interactive way the potential threats and related mechanisms of social media, avoid long bulleted lists, offer explanations and support, and provide references". The objective was to achieve a natural conversational exchange and avoid e.g. encyclopaedic responses. Within the chat interface participants received a code to provide in the questionnaire to ensure the completion of the task.

The chatbot interaction section was repeated three times, giving a chance to learners to incrementally improve their prompting. Then, learners were shown an instructional video showing prompt strategies (the three experimental conditions) Then learners completed the chatbot interaction section three more times for a total of six times. At the end the post-study questions were administered to learners.

Finally, the students' prompts were annotated for the selected features following the same procedure as for the initial dataset.

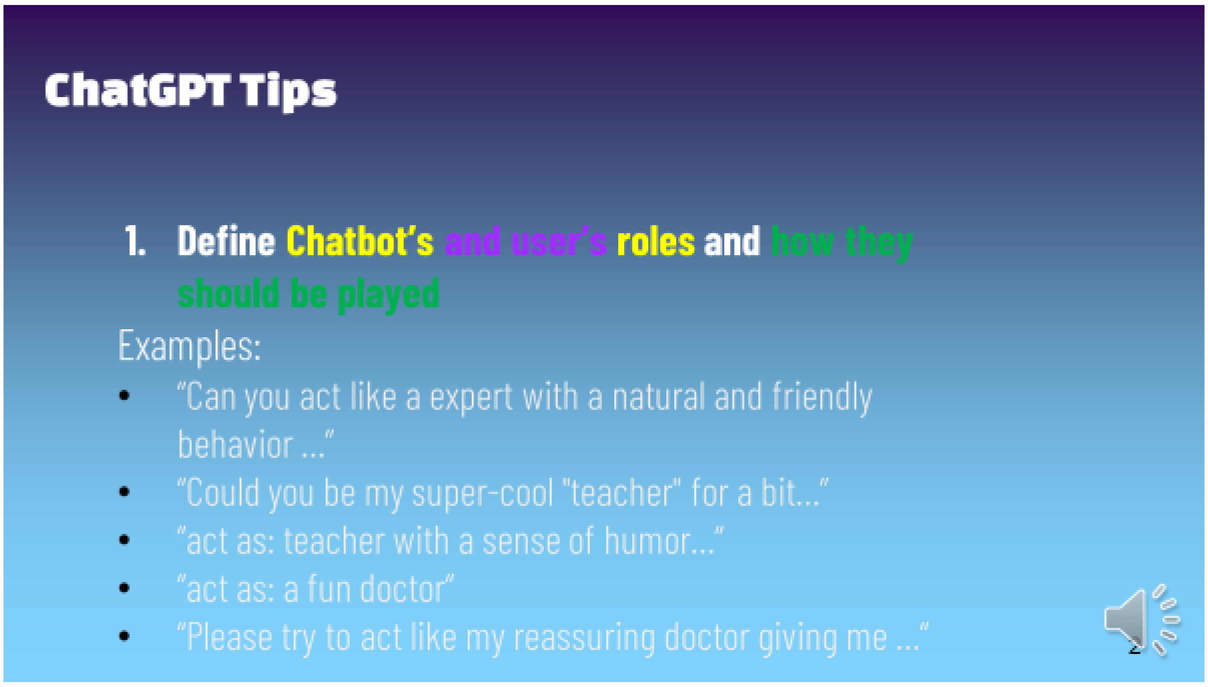
Final Prompting Guidelines - Educational Activity Slides

Task Specific Guidelines - Educational Activity Slides



ChatGPT Tips (2)

1. Define Chatbot's and user's roles and how they should be played
2. Describe the conversation evolution providing a sequence of steps and how they may repeat, wait or stop
3. Prefer multiple simple sentences to long complex ones with nested subsentences
4. Ask to split the topics of conversation, use implicit or explicit enumeration of subtopics
5. Specify all the actions that the chatbot must perform to have a successful interaction

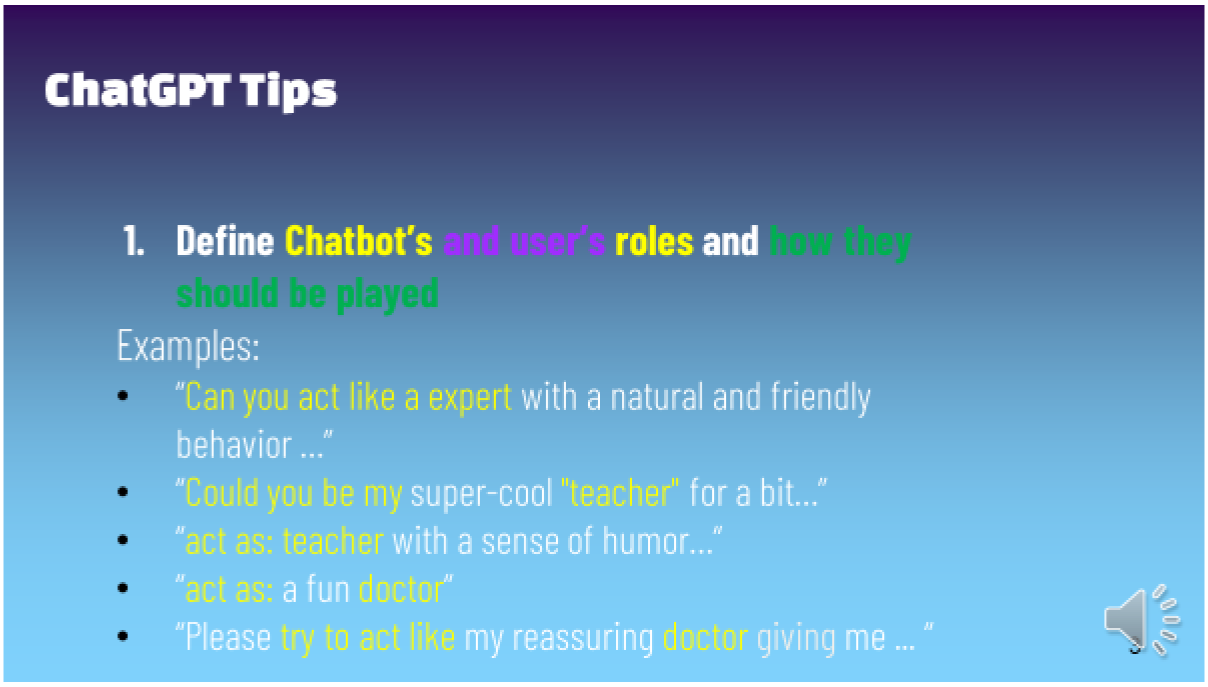


ChatGPT Tips

1. **Define Chatbot's and user's roles and how they should be played**

Examples:

- "Can you act like a expert with a natural and friendly behavior ..."
- "Could you be my super-cool "teacher" for a bit..."
- "act as: teacher with a sense of humor..."
- "act as: a fun doctor"
- "Please try to act like my reassuring doctor giving me ..."

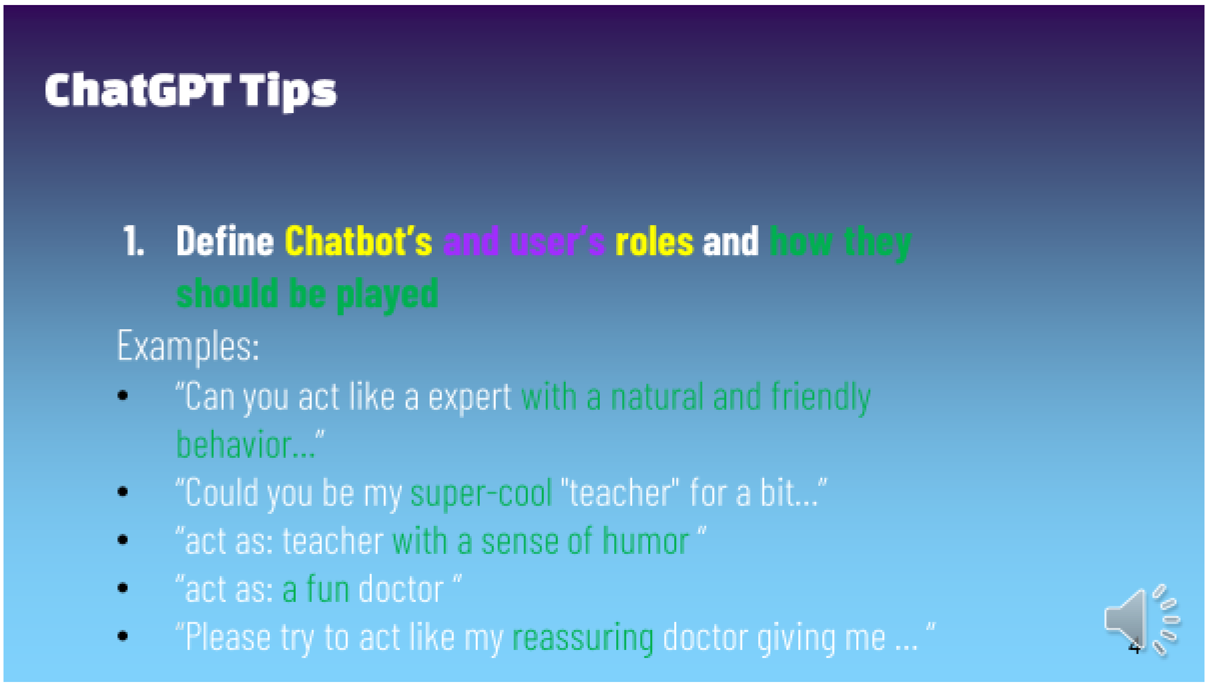


ChatGPT Tips

1. **Define Chatbot's and user's roles and how they should be played**

Examples:

- "Can you act like a expert with a natural and friendly behavior ..."
- "Could you be my super-cool "teacher" for a bit..."
- "act as: teacher with a sense of humor..."
- "act as: a fun doctor"
- "Please try to act like my reassuring doctor giving me ... "

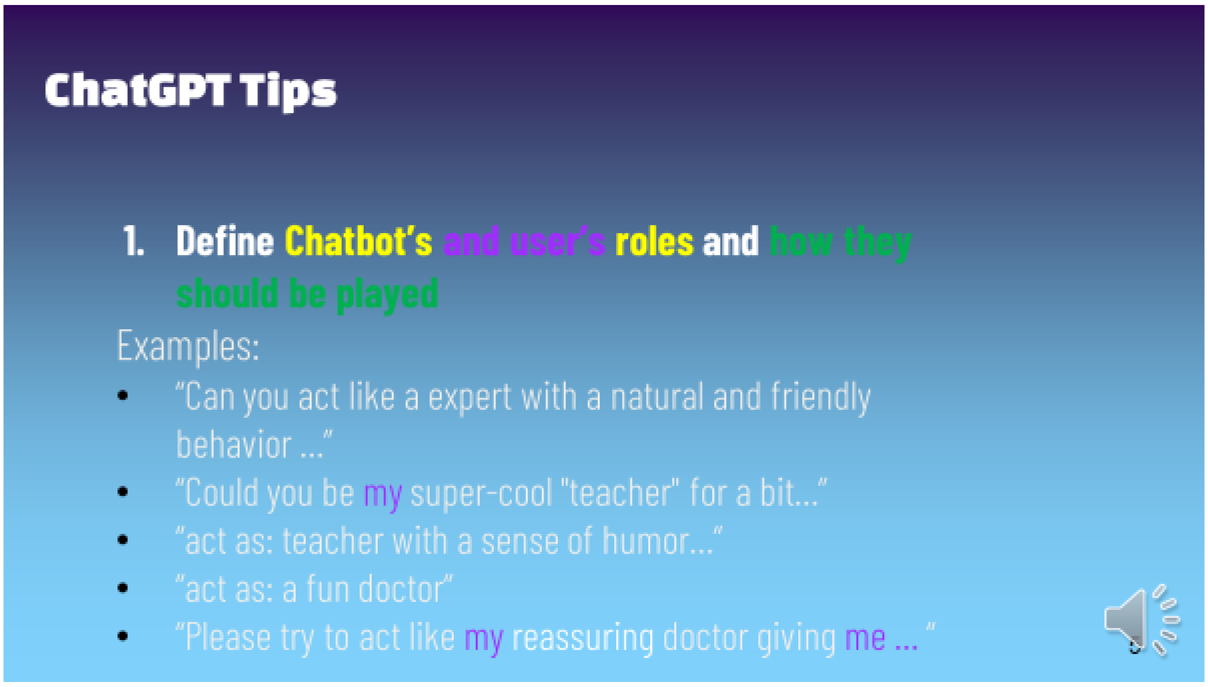


ChatGPT Tips

1. **Define Chatbot's and user's roles and how they should be played**

Examples:

- "Can you act like a expert with a natural and friendly behavior…"
- "Could you be my super-cool "teacher" for a bit…"
- "act as: teacher with a sense of humor "
- "act as: a fun doctor "
- "Please try to act like my reassuring doctor giving me … "

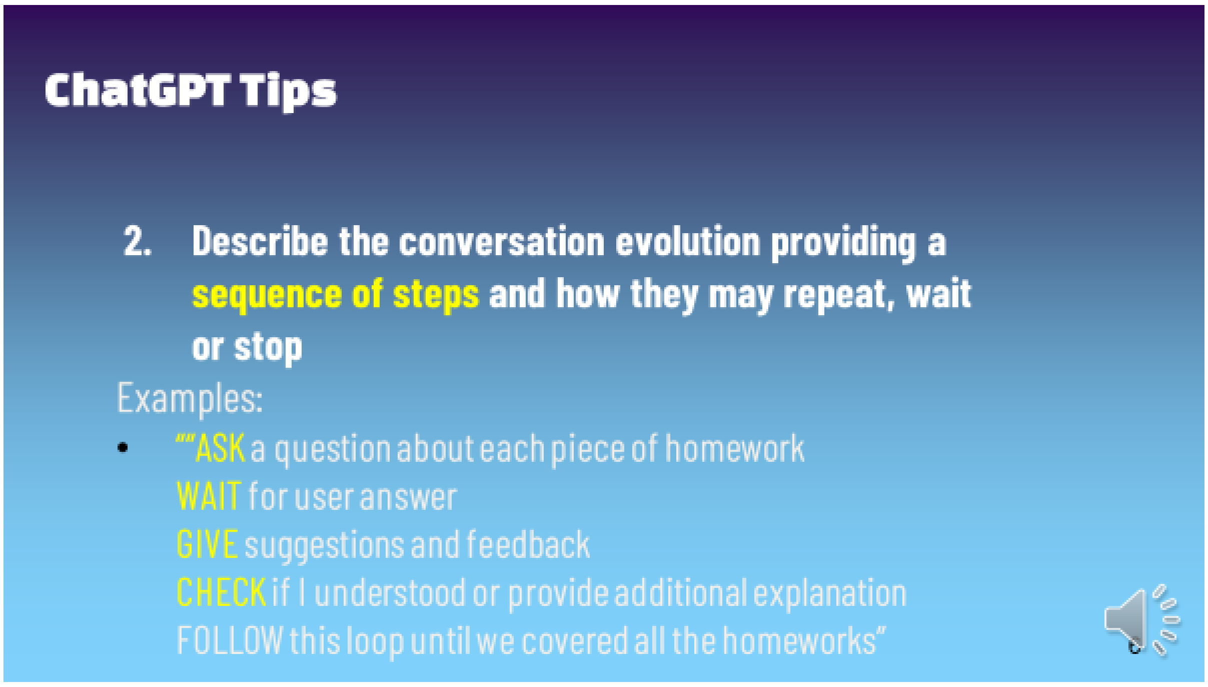


ChatGPT Tips

1. **Define Chatbot's and user's roles and how they should be played**

Examples:

- "Can you act like a expert with a natural and friendly behavior …"
- "Could you be my super-cool "teacher" for a bit…"
- "act as: teacher with a sense of humor…"
- "act as: a fun doctor"
- "Please try to act like my reassuring doctor giving me … "

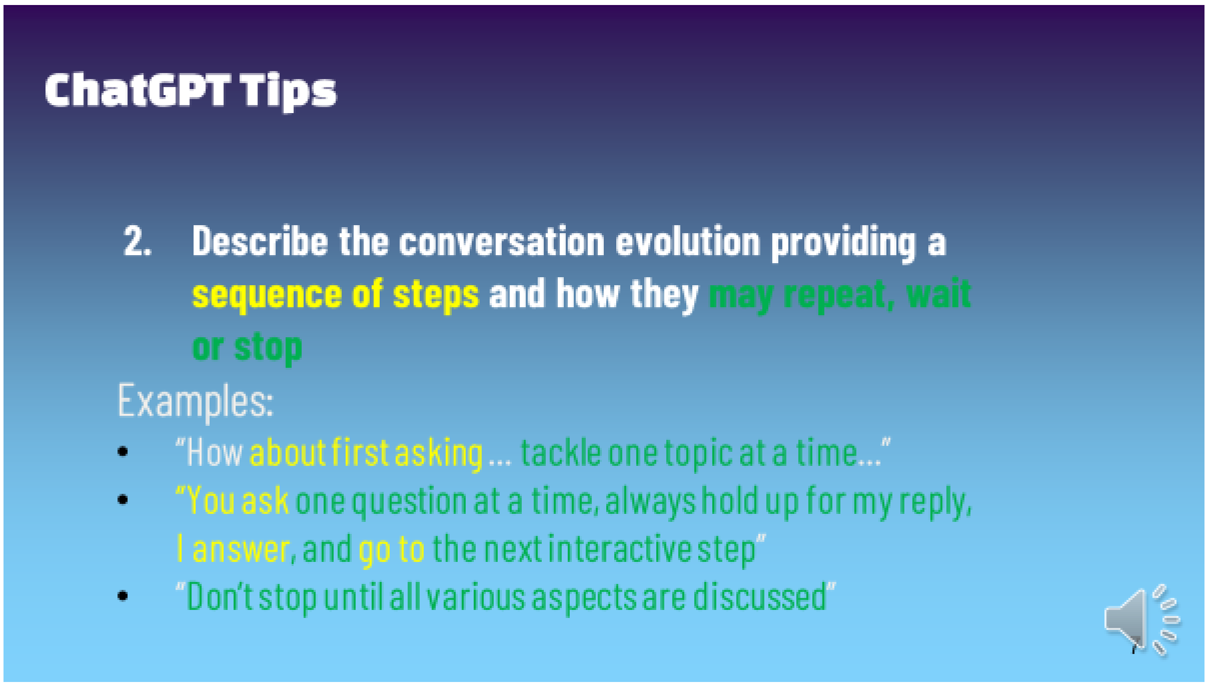


ChatGPT Tips

2. **Describe the conversation evolution providing a sequence of steps and how they may repeat, wait or stop**

Examples:

- ""ASK a question about each piece of homework
  WAIT for user answer
  GIVE suggestions and feedback
  CHECK if I understood or provide additional explanation
  FOLLOW this loop until we covered all the homeworks"

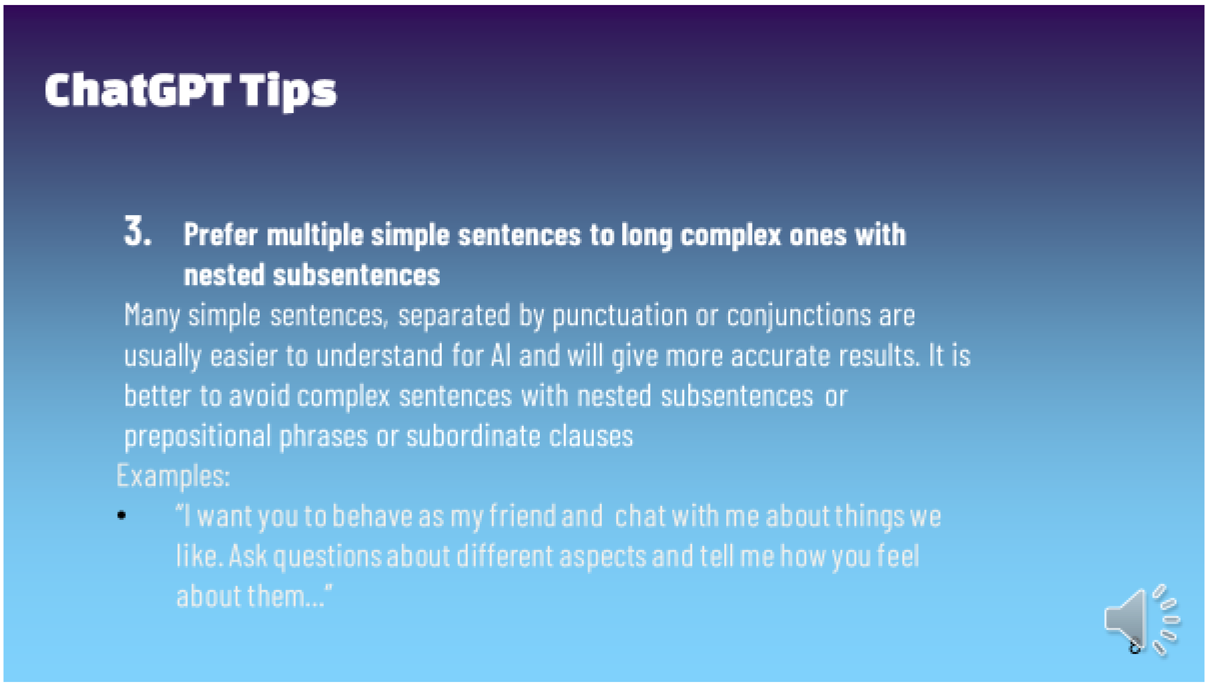


ChatGPT Tips

2. **Describe the conversation evolution providing a sequence of steps and how they may repeat, wait or stop**

Examples:

- "How about first asking … tackle one topic at a time…"
- "You ask one question at a time, always hold up for my reply, I answer, and go to the next interactive step"
- "Don't stop until all various aspects are discussed"



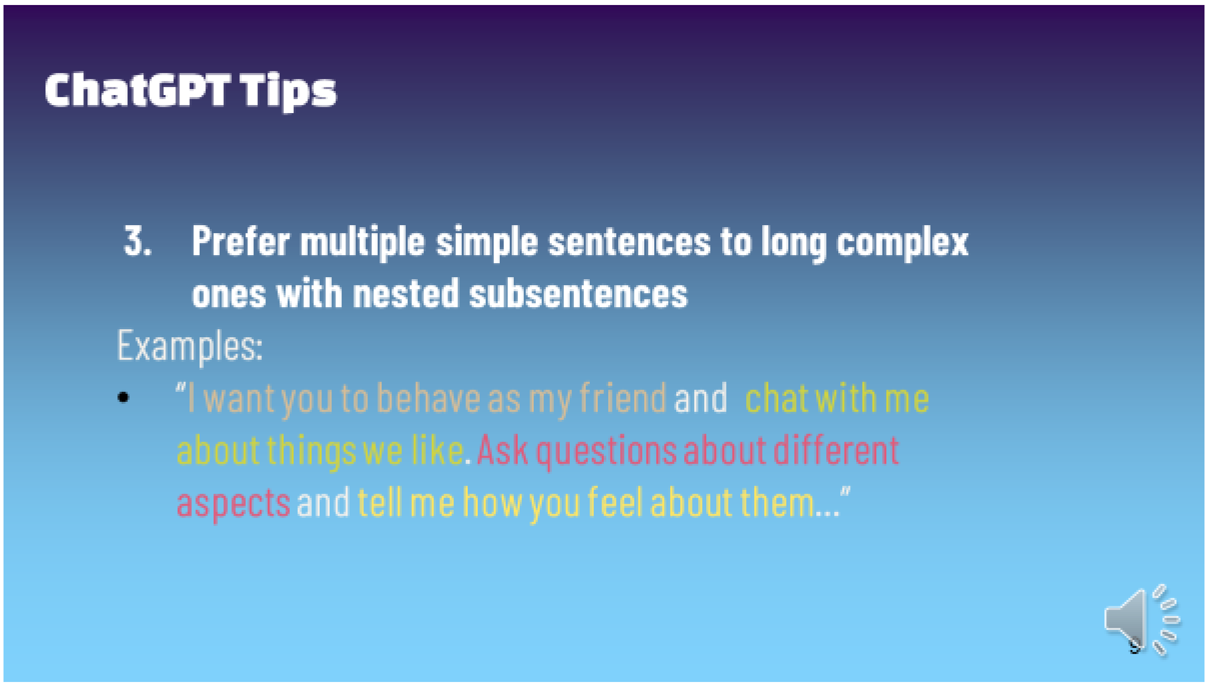
ChatGPT Tips

3. **Prefer multiple simple sentences to long complex ones with nested subsentences**

Many simple sentences, separated by punctuation or conjunctions are usually easier to understand for AI and will give more accurate results. It is better to avoid complex sentences with nested subsentences or prepositional phrases or subordinate clauses

Examples:

- "I want you to behave as my friend and chat with me about things we like. Ask questions about different aspects and tell me how you feel about them…"

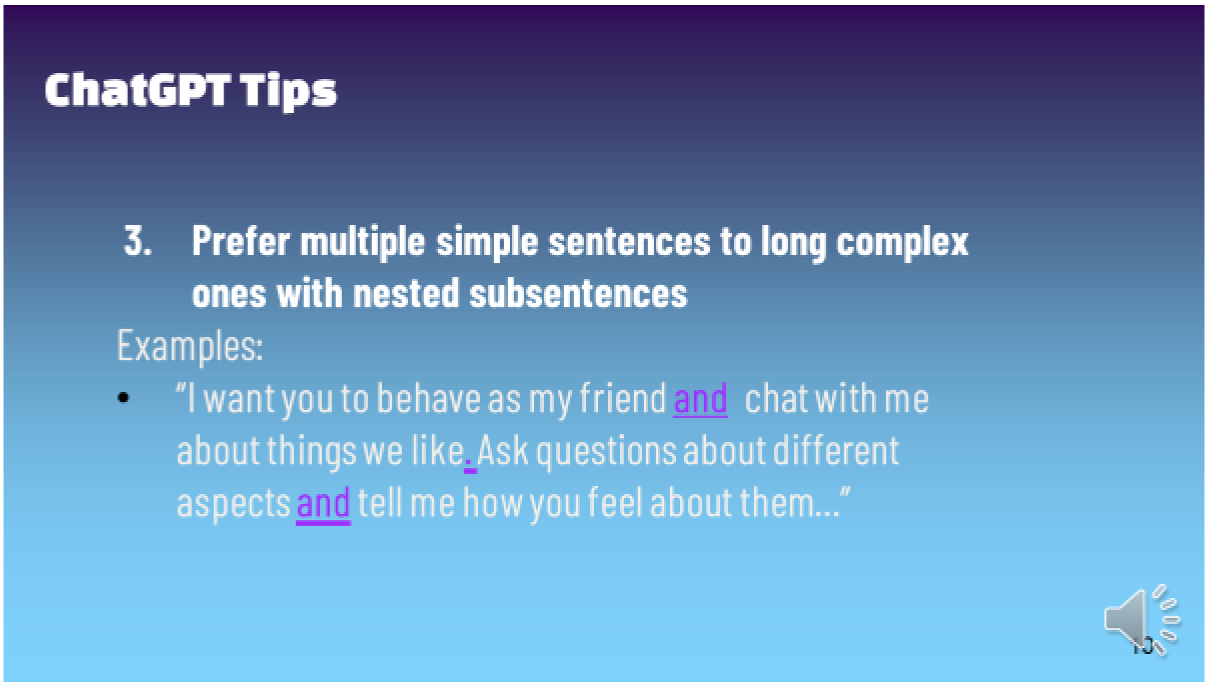


ChatGPT Tips

3. **Prefer multiple simple sentences to long complex ones with nested subsentences**

Examples:

- "I want you to behave as my friend and chat with me about things we like. Ask questions about different aspects and tell me how you feel about them…"

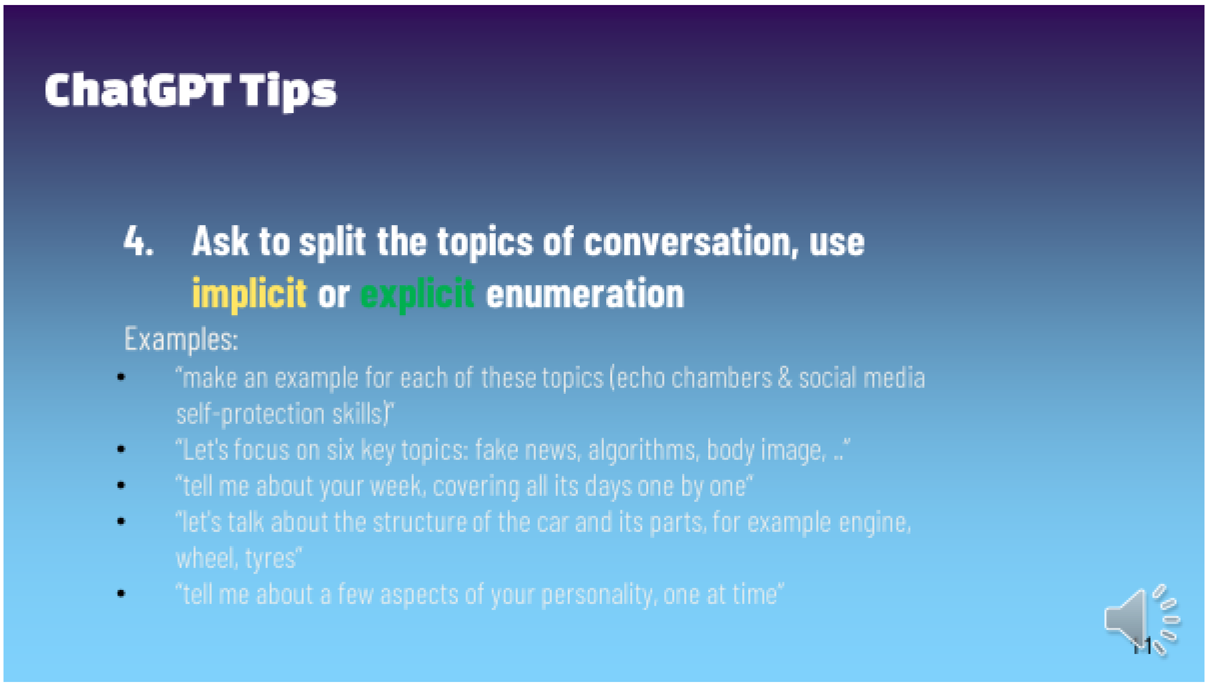


ChatGPT Tips

3. **Prefer multiple simple sentences to long complex ones with nested subsentences**

Examples:

- "I want you to behave as my friend and chat with me about things we like. Ask questions about different aspects and tell me how you feel about them…"

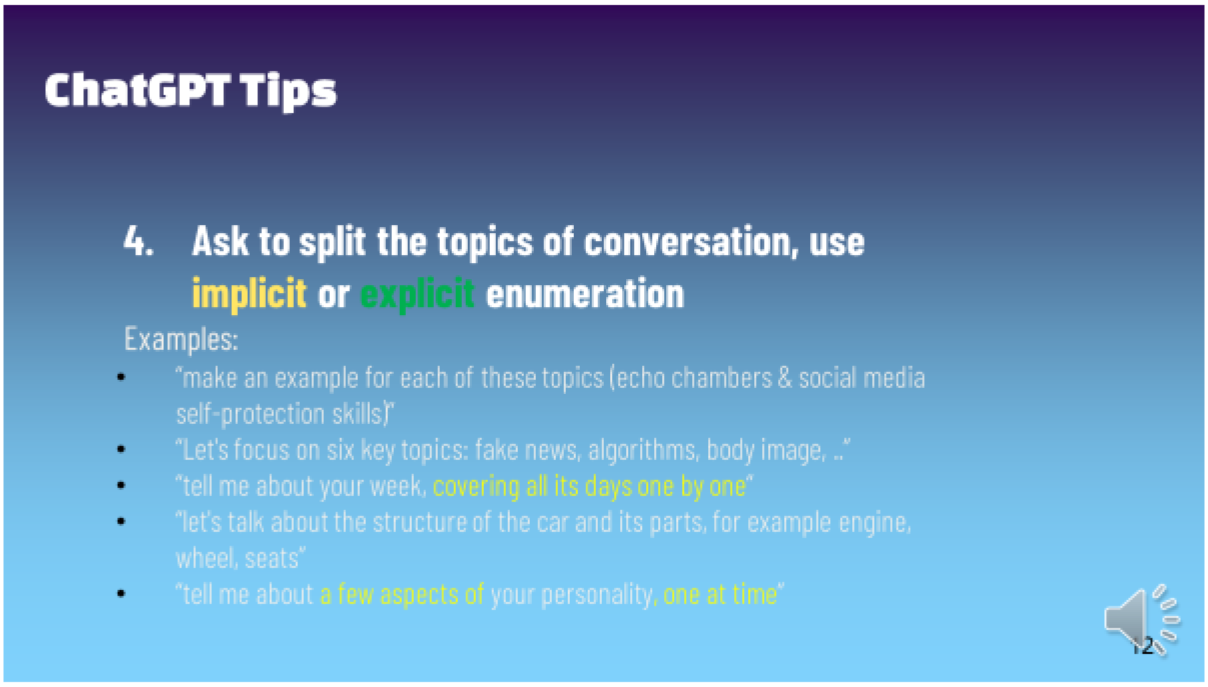


ChatGPT Tips

4. **Ask to split the topics of conversation, use implicit or explicit enumeration**

Examples:

- "make an example for each of these topics (echo chambers & social media self-protection skills)"
- "Let's focus on six key topics: fake news, algorithms, body image, .."
- "tell me about your week, covering all its days one by one"
- "let's talk about the structure of the car and its parts, for example engine, wheel, tyres"
- "tell me about a few aspects of your personality, one at time"



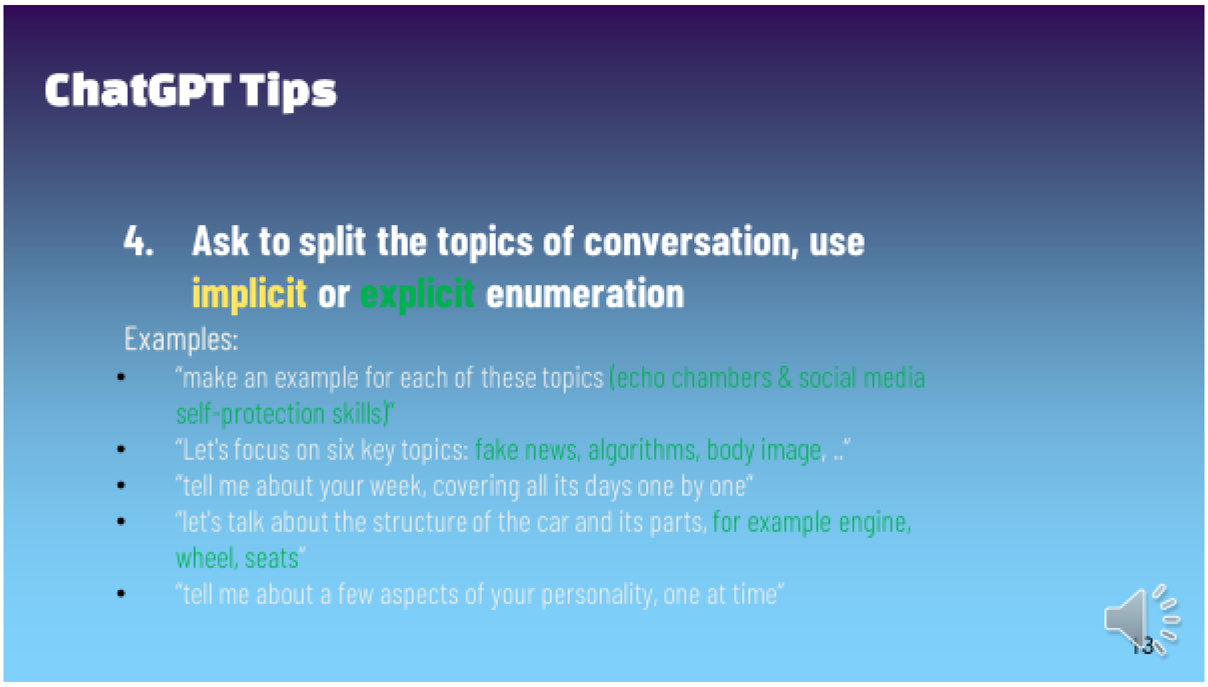
ChatGPT Tips

4. **Ask to split the topics of conversation, use implicit or explicit enumeration**

Examples:

- "make an example for each of these topics (echo chambers & social media self-protection skills)"
- "Let's focus on six key topics: fake news, algorithms, body image, .."
- "tell me about your week, covering all its days one by one"
- "let's talk about the structure of the car and its parts, for example engine, wheel, seats"
- "tell me about a few aspects of your personality, one at time"

12



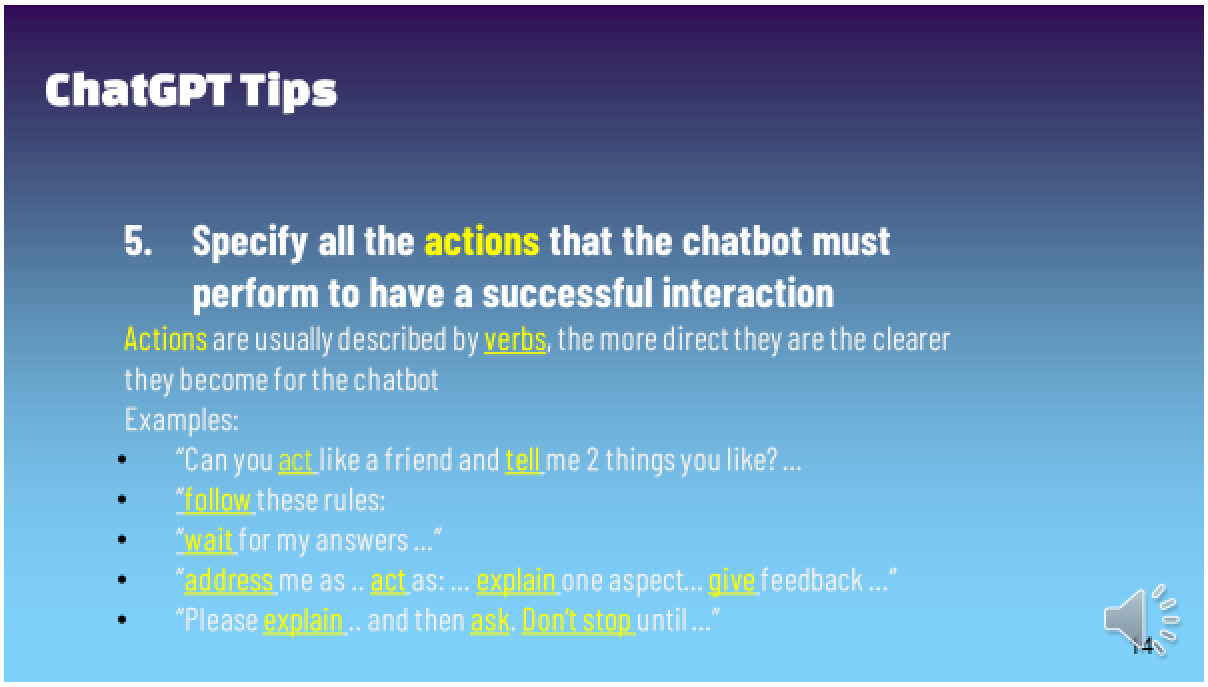
ChatGPT Tips

4. **Ask to split the topics of conversation, use implicit or explicit enumeration**

Examples:

- "make an example for each of these topics (echo chambers & social media self-protection skills)"
- "Let's focus on six key topics: fake news, algorithms, body image, .."
- "tell me about your week, covering all its days one by one"
- "let's talk about the structure of the car and its parts, for example engine, wheel, seats"
- "tell me about a few aspects of your personality, one at time"

13



ChatGPT Tips

5. **Specify all the actions that the chatbot must perform to have a successful interaction**

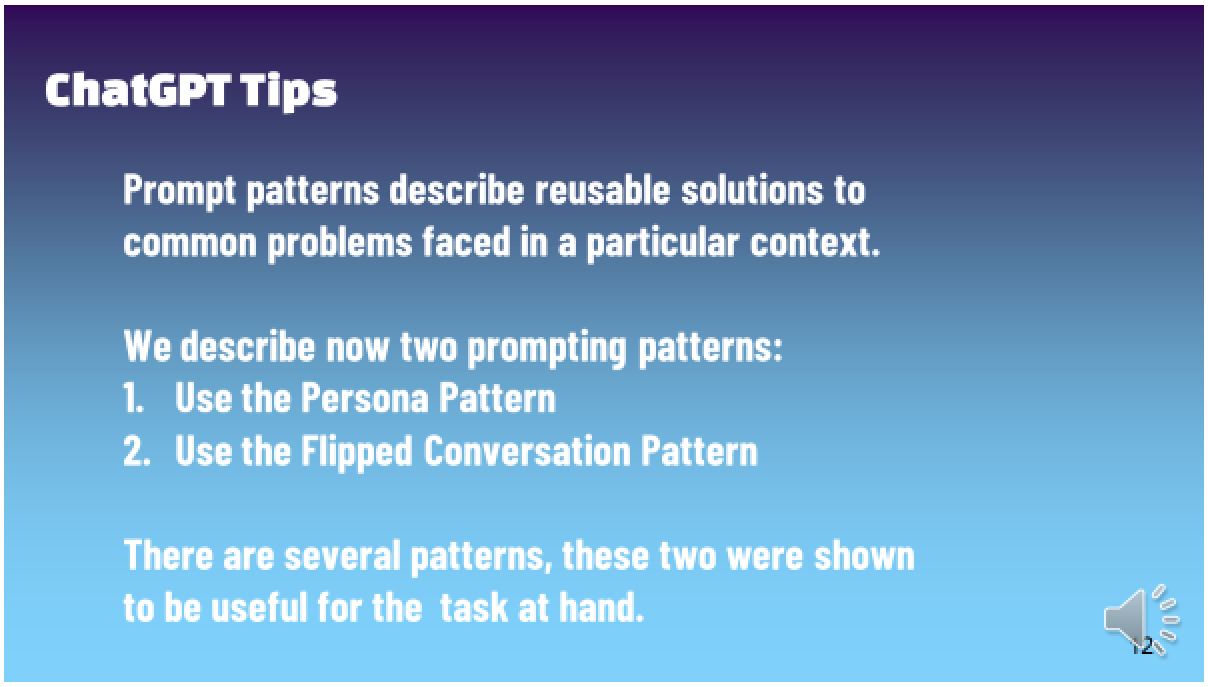
Actions are usually described by verbs, the more direct they are the clearer they become for the chatbot

Examples:

- "Can you act like a friend and tell me 2 things you like? …
- "follow these rules:
- "wait for my answers …"
- "address me as .. act as: … explain one aspect… give feedback …"
- "Please explain .. and then ask. Don't stop until …"

14

Prompting Patterns Guidelines - Educational Activity Slides



ChatGPT Tips

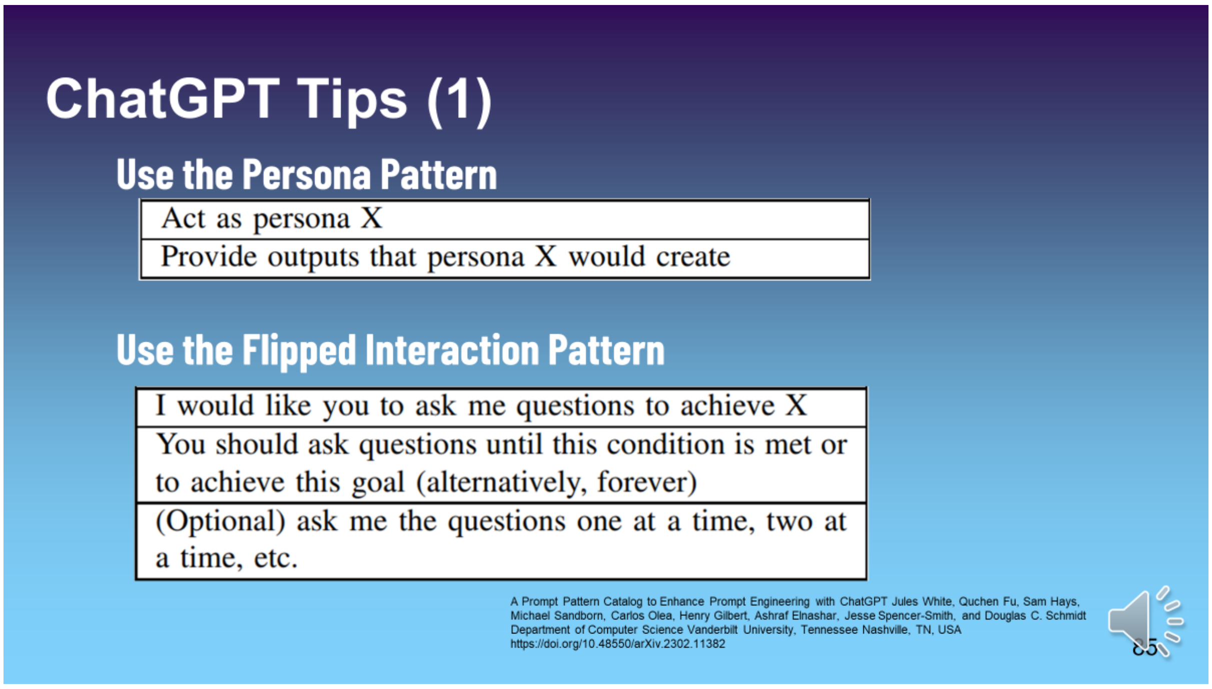
Prompt patterns describe reusable solutions to common problems faced in a particular context.

We describe now two prompting patterns:

1. **Use the Persona Pattern**
2. **Use the Flipped Conversation Pattern**

There are several patterns, these two were shown to be useful for the task at hand.

12



ChatGPT Tips (1)

Use the Persona Pattern

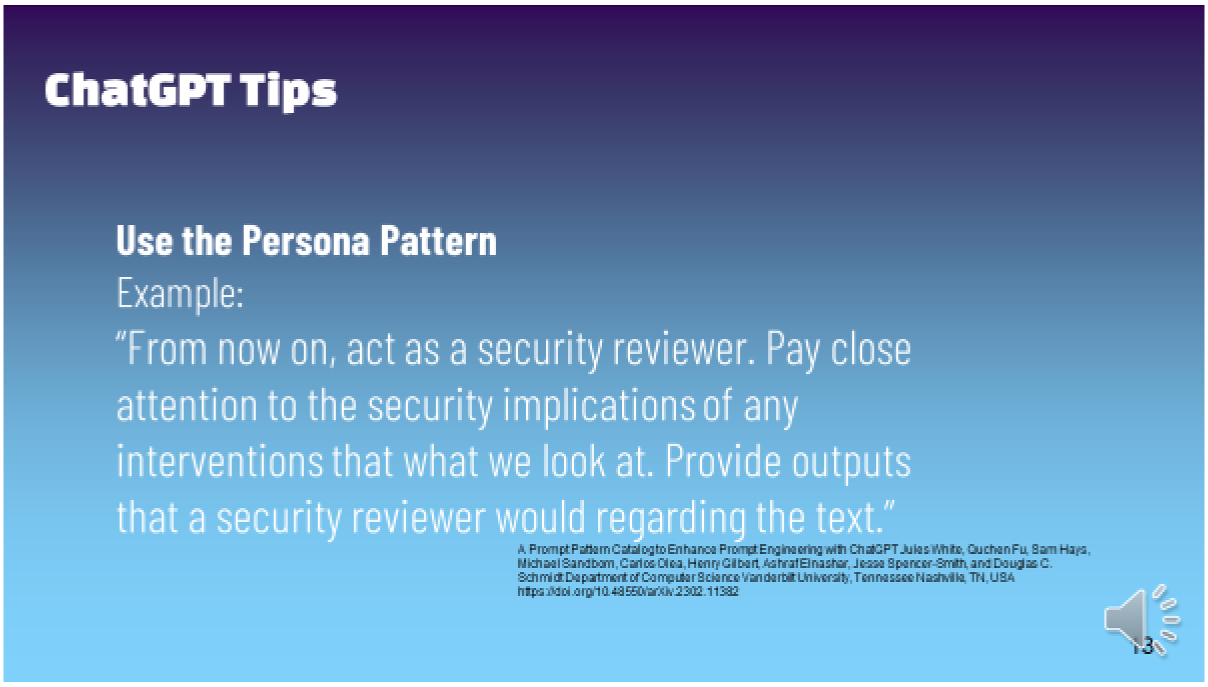
Act as persona X
Provide outputs that persona X would create

Use the Flipped Interaction Pattern

I would like you to ask me questions to achieve X
You should ask questions until this condition is met or to achieve this goal (alternatively, forever)
(Optional) ask me the questions one at a time, two at a time, etc.

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA https://doi.org/10.48550/arXiv.2302.11382

85



ChatGPT Tips

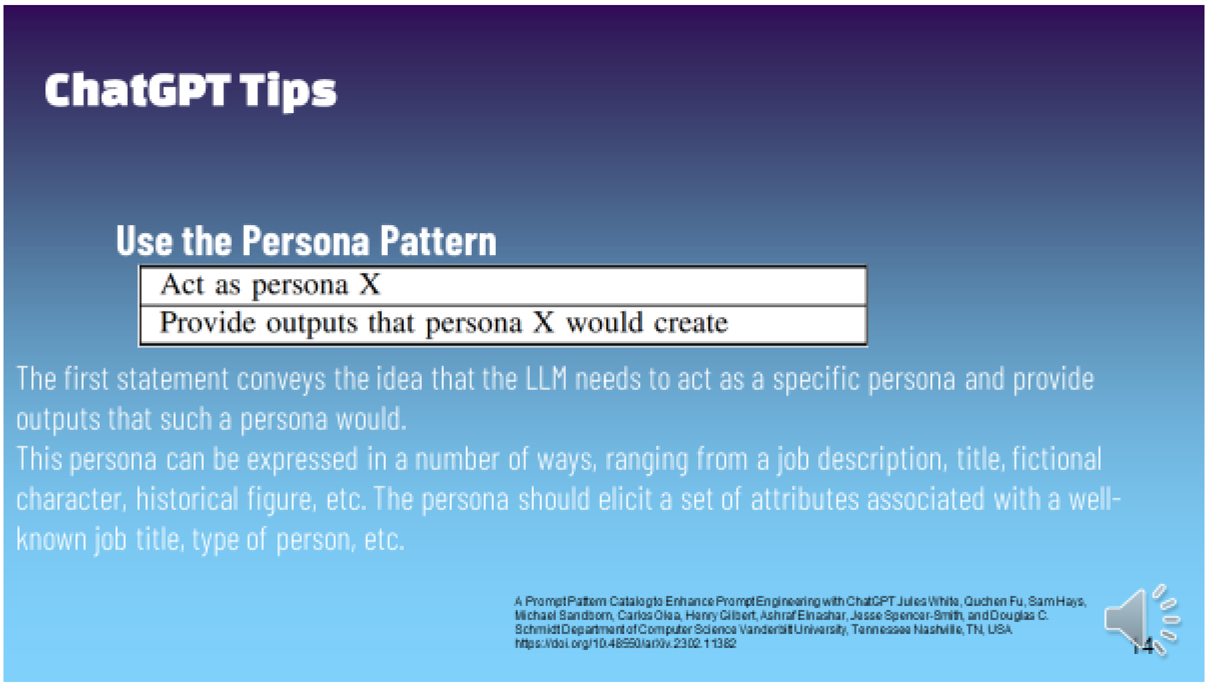
Use the Persona Pattern

Example:

"From now on, act as a security reviewer. Pay close attention to the security implications of any interventions that what we look at. Provide outputs that a security reviewer would regarding the text."

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA https://doi.org/10.48550/arXiv.2302.11382

13



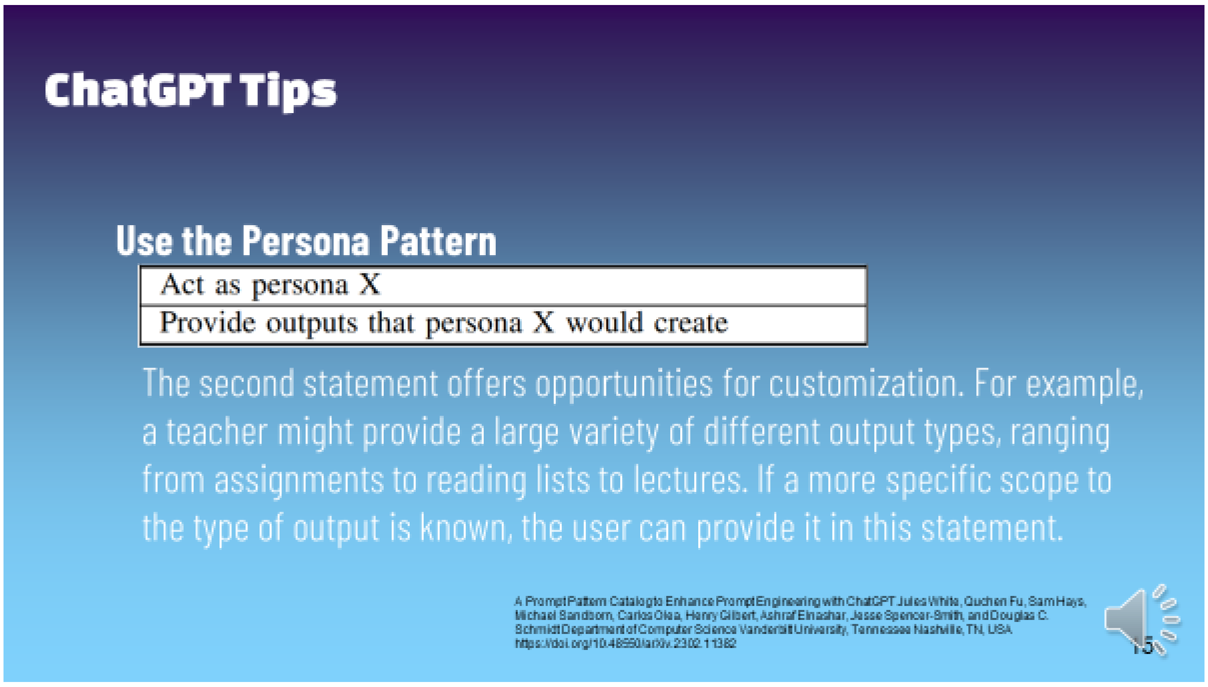
ChatGPT Tips

Use the Persona Pattern

Act as persona X
Provide outputs that persona X would create

The first statement conveys the idea that the LLM needs to act as a specific persona and provide outputs that such a persona would.
This persona can be expressed in a number of ways, ranging from a job description, title, fictional character, historical figure, etc. The persona should elicit a set of attributes associated with a well-known job title, type of person, etc.

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA
https://doi.org/10.48550/arXiv.2302.11382



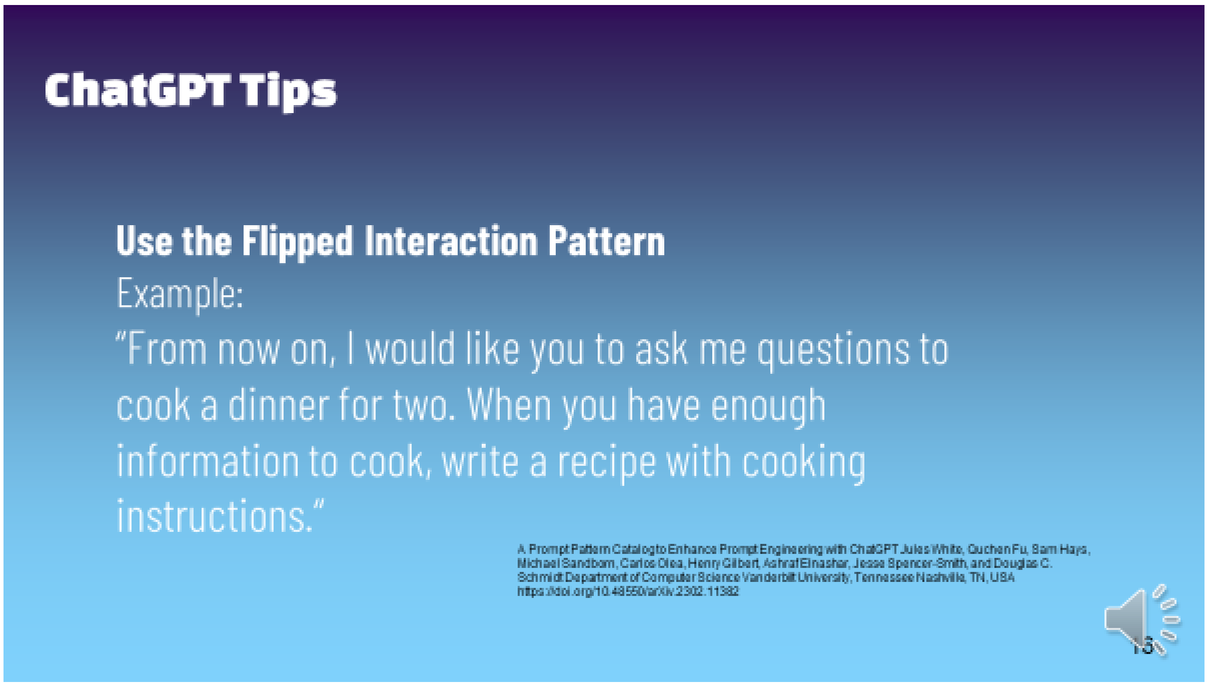
ChatGPT Tips

Use the Persona Pattern

Act as persona X
Provide outputs that persona X would create

The second statement offers opportunities for customization. For example, a teacher might provide a large variety of different output types, ranging from assignments to reading lists to lectures. If a more specific scope to the type of output is known, the user can provide it in this statement.

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA
https://doi.org/10.48550/arXiv.2302.11382



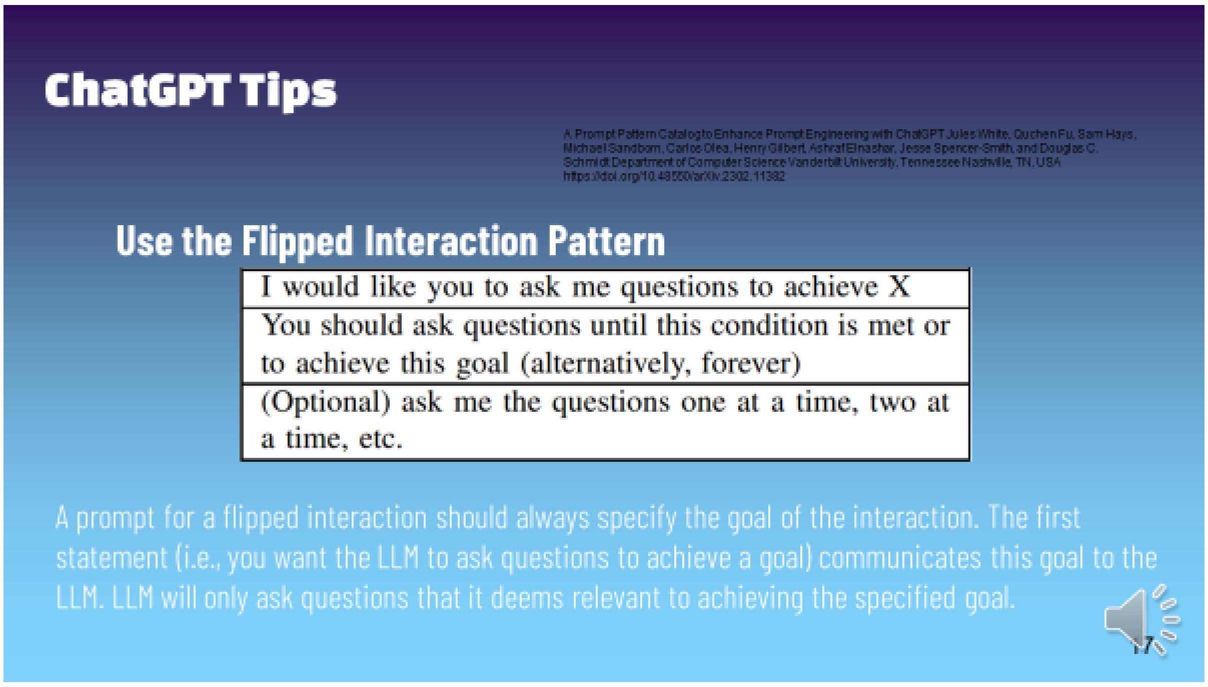
ChatGPT Tips

Use the Flipped Interaction Pattern

Example:

“From now on, I would like you to ask me questions to cook a dinner for two. When you have enough information to cook, write a recipe with cooking instructions.”

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA https://doi.org/10.48550/arXiv.2302.11382



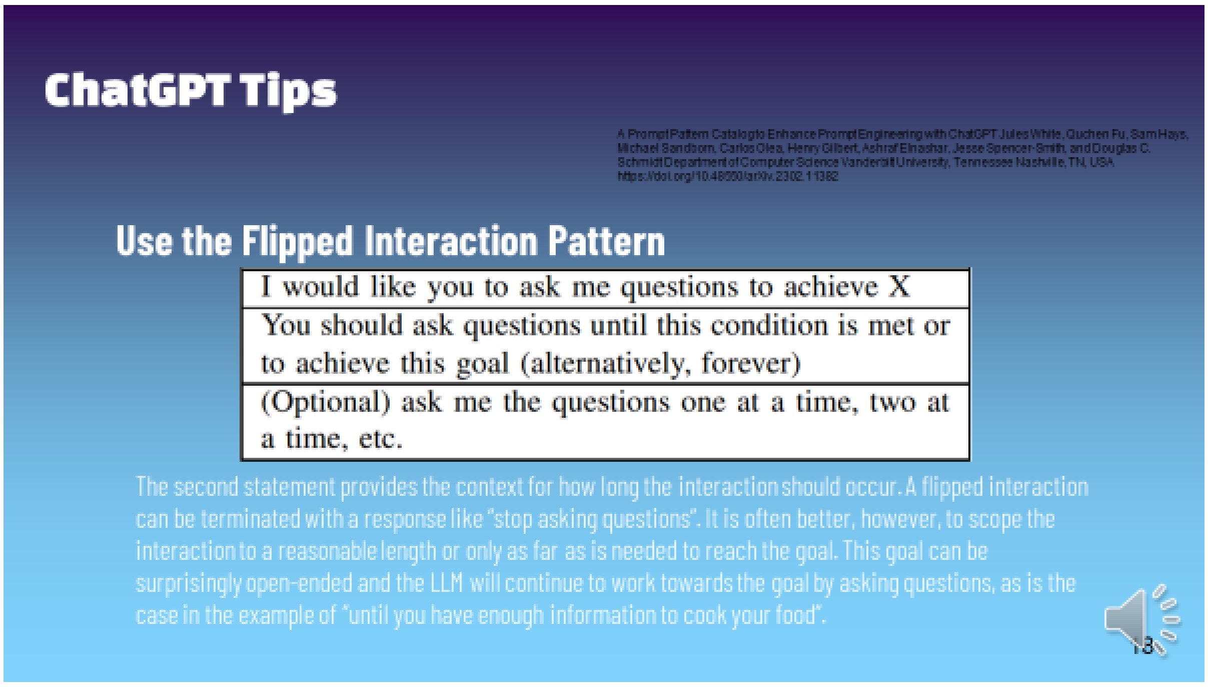
ChatGPT Tips

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA https://doi.org/10.48550/arXiv.2302.11382

Use the Flipped Interaction Pattern

I would like you to ask me questions to achieve X
You should ask questions until this condition is met or to achieve this goal (alternatively, forever)
(Optional) ask me the questions one at a time, two at a time, etc.

A prompt for a flipped interaction should always specify the goal of the interaction. The first statement (i.e., you want the LLM to ask questions to achieve a goal) communicates this goal to the LLM. LLM will only ask questions that it deems relevant to achieving the specified goal.



ChatGPT Tips

A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt Department of Computer Science Vanderbilt University, Tennessee Nashville, TN, USA https://doi.org/10.48550/arXiv.2302.11382

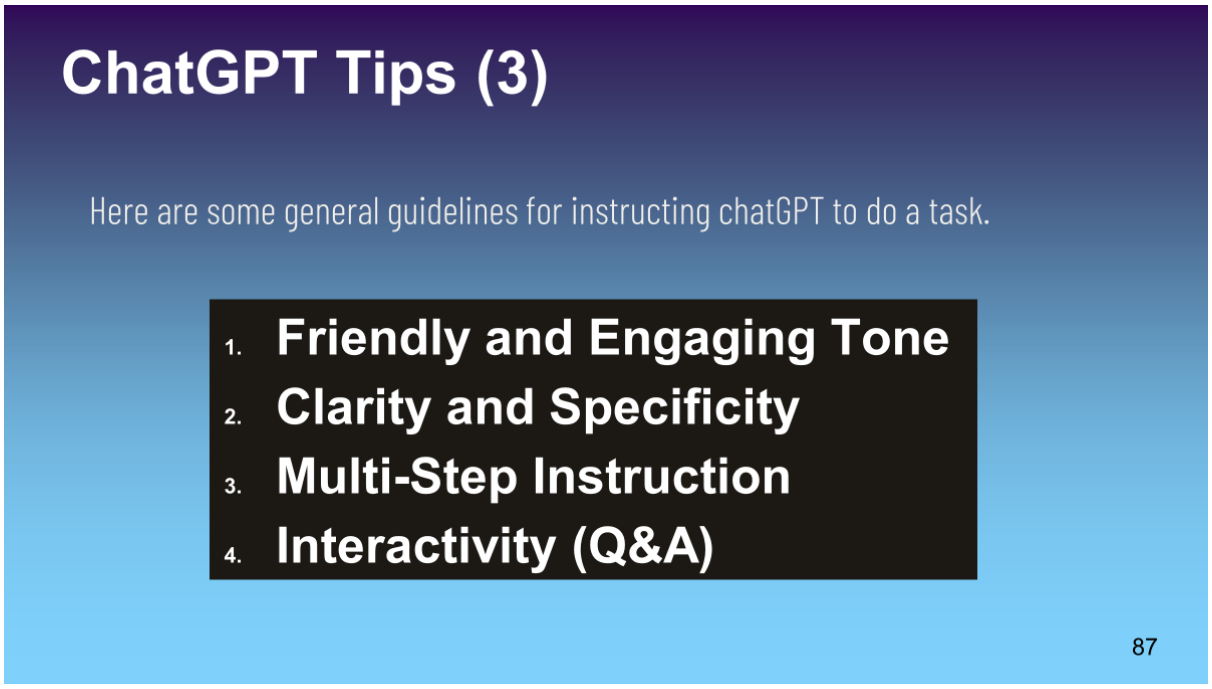
Use the Flipped Interaction Pattern

I would like you to ask me questions to achieve X
You should ask questions until this condition is met or to achieve this goal (alternatively, forever)
(Optional) ask me the questions one at a time, two at a time, etc.

The second statement provides the context for how long the interaction should occur. A flipped interaction can be terminated with a response like "stop asking questions". It is often better, however, to scope the interaction to a reasonable length or only as far as is needed to reach the goal. This goal can be surprisingly open-ended and the LLM will continue to work towards the goal by asking questions, as is the case in the example of "until you have enough information to cook your food".

13

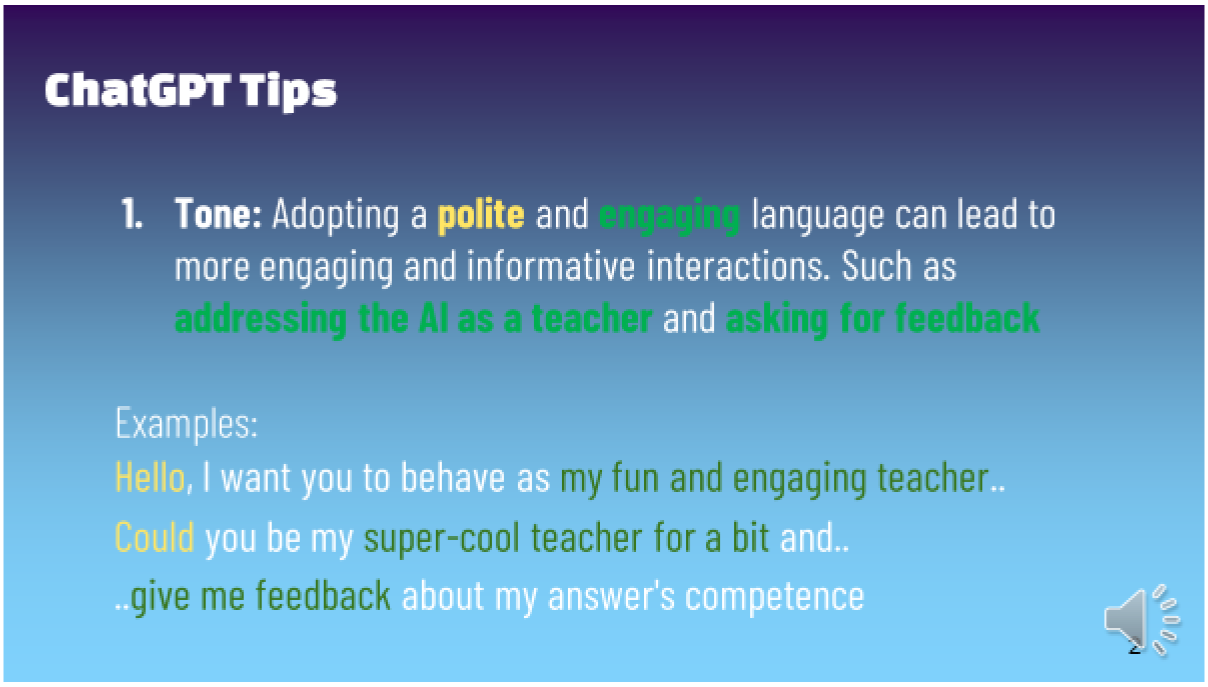
ChatGPT Generated Guidelines - Educational Activity Slides



ChatGPT Tips (3)

Here are some general guidelines for instructing chatGPT to do a task.

1. Friendly and Engaging Tone
2. Clarity and Specificity
3. Multi-Step Instruction
4. Interactivity (Q&A)



ChatGPT Tips

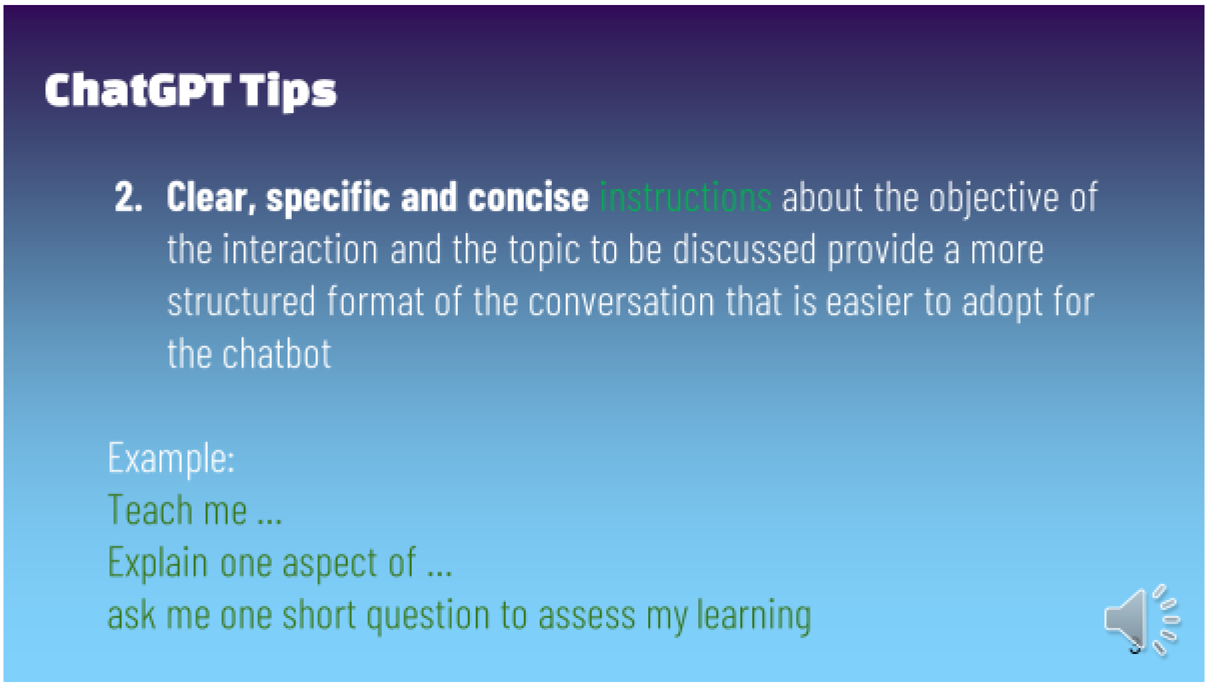
1. **Tone:** Adopting a **polite** and **engaging** language can lead to more engaging and informative interactions. Such as **addressing the AI as a teacher** and **asking for feedback**

Examples:

Hello, I want you to behave as my fun and engaging teacher..

Could you be my super-cool teacher for a bit and..

..give me feedback about my answer's competence



ChatGPT Tips

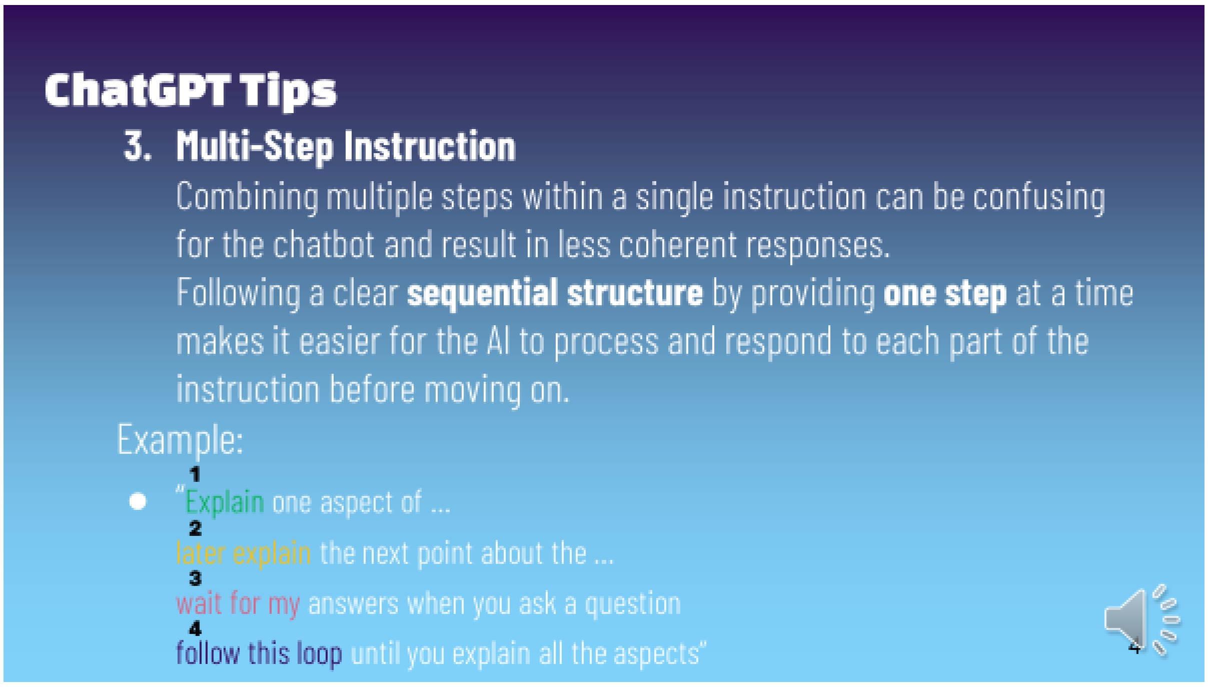
2. **Clear, specific and concise** instructions about the objective of the interaction and the topic to be discussed provide a more structured format of the conversation that is easier to adopt for the chatbot

Example:

Teach me …

Explain one aspect of …

ask me one short question to assess my learning



ChatGPT Tips

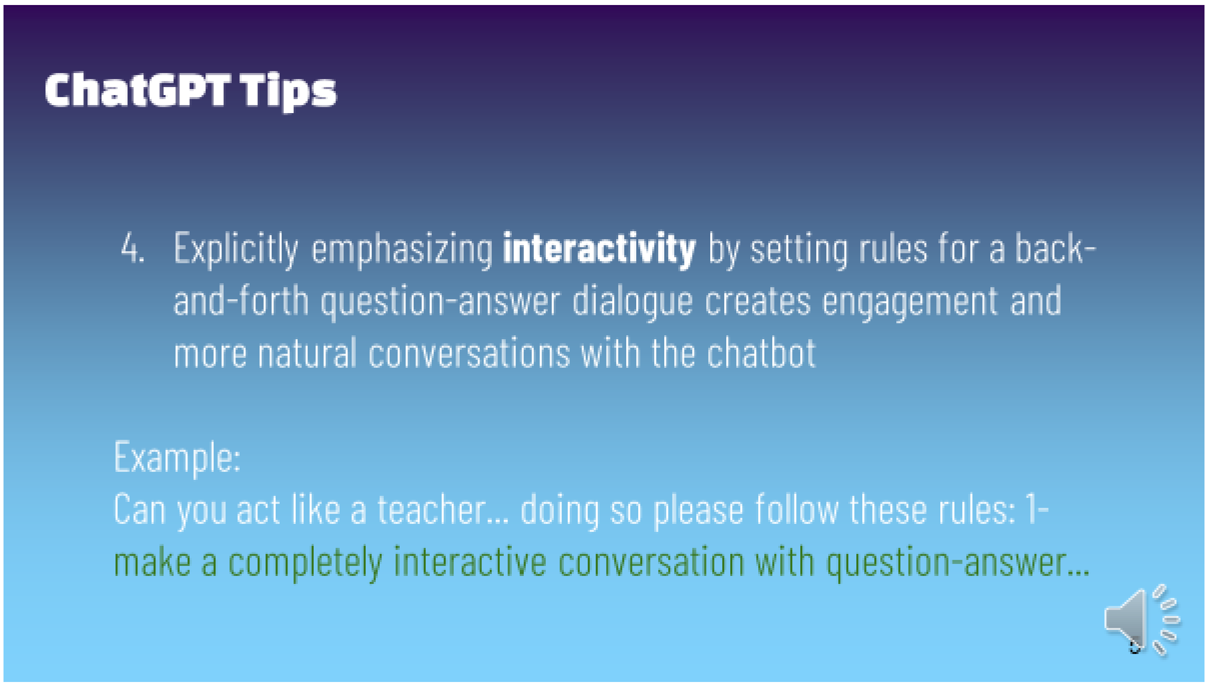
3. Multi-Step Instruction

Combining multiple steps within a single instruction can be confusing for the chatbot and result in less coherent responses.
Following a clear **sequential structure** by providing **one step** at a time makes it easier for the AI to process and respond to each part of the instruction before moving on.

Example:

- 1
  "Explain one aspect of …
  2
  later explain the next point about the …
  3
  wait for my answers when you ask a question
  4
  follow this loop until you explain all the aspects"

4



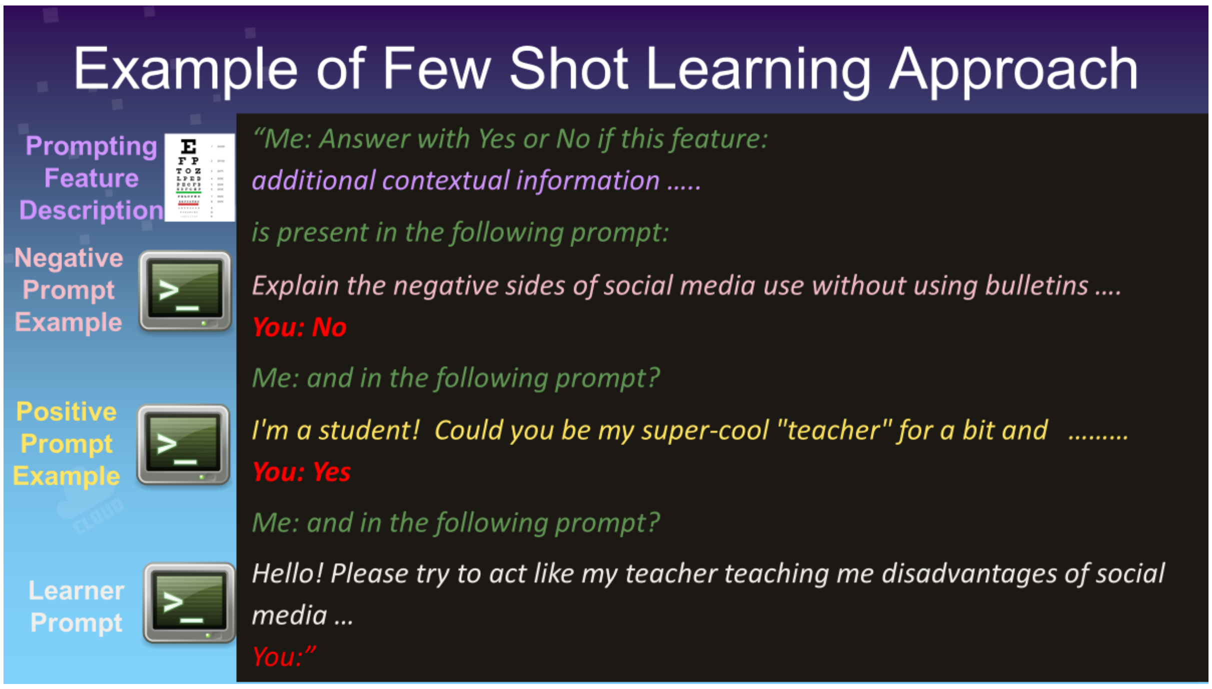
ChatGPT Tips

4. Explicitly emphasizing **interactivity** by setting rules for a back-and-forth question-answer dialogue creates engagement and more natural conversations with the chatbot

Example:
Can you act like a teacher… doing so please follow these rules: 1-make a completely interactive conversation with question-answer…

5

Few shots classification prompt example



Example of Few Shot Learning Approach

Prompting Feature Description
Negative Prompt Example
Positive Prompt Example
Learner Prompt

"Me: Answer with Yes or No if this feature:
additional contextual information .....
is present in the following prompt:
Explain the negative sides of social media use without using bulletins ....
You: No
Me: and in the following prompt?
I'm a student! Could you be my super-cool "teacher" for a bit and ....
You: Yes
Me: and in the following prompt?
Hello! Please try to act like my teacher teaching me disadvantages of social media ...
You:"

Different wording of the features for few-shot learning

We used different features description wording to check their impact on few-shot learning and improve its performance. Here we show some examples of descriptions for some of the features. The wording presented to the learnes was presented in the slides at section "Final Prompting Guidelines - Educational Activity Slides".

Topic - concise(1,NaN):

cursorily description of the topic with few details
Describes the topic of conversation without splitting it in parts, e.g "let's talk about the human body" "how was your week" or "what do you think of me"

Role form/context(1,NaN)

additional contextual information about the role of the language model, the user, or the environment
Specify how the other conversation participant must perform their role through forms like "act like" followed by role adjectives e.g. "careful", "reassuring", "funny" or forms like "with a friendly attitude" or "in an interactive way"

ProvideOutPuts- Persona Pattern (1,Nan)

instructing the language model about the expected output by using keywords like: teach,

explain, etc.
Defines more specifically the type of statements that the other conversation participant can produce while playing a requested role. For example, it specifies that for the requested role of teacher it assigns reading lists , asks questions, presents lectures.

2 Goals (1,NaN)

description of exactly two countable distinct goals
Contains two requests only for the other conversation participant in the form of a verb, e.g. "explain this and make it interactive", "teach me and ask questions"

Prompt code (as also available as part of the GitHub repository):

In order to foster reproducibility, we will make available our data and implementations for the community.

Here are some excerpts from the code for the creation of prompts.The number in brackets after each function heading indicates the prompt_gen id in the tables below.

createPromptZero [prompt_gen id: 0]

```
"""Me: Check if this feature:
{feature_description}\n
is present in the following prompts, answer with YES or NO\n
{positive_few_shot1}\n
You: Yes\n
Me: and in the following prompt?
{negative_few_shot1}\n
You: No\n

Me: and in the following prompt?
{eval_prompt}\n
You: \n"""
```

```
def createPromptZero(self,eval_prompt, feature, shots):
   feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
   # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
   positive_few_shot1 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
   # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
   positive_few_shot1 = '\n'.join(positive_few_shot1)
   negative_few_shot1 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
   # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
   negative_few_shot1 = '\n'.join(negative_few_shot1)

   eval_string = f"""Me: Check if this feature:
           {feature_description}\n
           is present in the following prompts, answer with YES or NO\n
           {positive_few_shot1}\n
           You: Yes\n
           Me: and in the following prompt?
           {negative_few_shot1}\n
           You: No\n

           Me: and in the following prompt?
           {eval_prompt}\n
           You: \n
           """
   return eval_string, feature_description
```

createPrompt [prompt_gen id: 1]

```
"""Me: Check if this feature:
{feature_description}\n
is present in the following prompts, answer with YES or NO\n
{positive_few_shot1}\n
You: Yes\n
Me: and in the following prompt?
{negative_few_shot1}\n
You: No\n

Me: and in the following prompt?
{negative_few_shot2}\n
You: No\n
Me: and in the following prompt?
{positive_few_shot2}\n
You: Yes\n

Me: and in the following prompt?
{eval_prompt}\n
You: \n"""
```

```
def createPrompt(self,eval_prompt, feature, shots):
   feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
   # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
   positive_few_shot1 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
   # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
```

```
    positive_few_shot1 = '\n'.join(positive_few_shot1)
    negative_few_shot1 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
    # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
    negative_few_shot1 = '\n'.join(negative_few_shot1)

    positive_few_shot2 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
    # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
    positive_few_shot2 = '\n'.join(positive_few_shot2)
    negative_few_shot2 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
    # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
    negative_few_shot2 = '\n'.join(negative_few_shot2)

    eval_string = f"""Me: Check if this feature:
            {feature_description}\n
            is present in the following prompts, answer with YES or NO\n
            {positive_few_shot1}\n
            You: Yes\n
            Me: and in the following prompt?
            {negative_few_shot1}\n
            You: No\n

            Me: and in the following prompt?
            {negative_few_shot2}\n
            You: No\n
            Me: and in the following prompt?
            {positive_few_shot2}\n
            You: Yes\n

            Me: and in the following prompt?
            {eval_prompt}\n
            You: \n
            """
    return eval_string, feature_description
```

createPromptInverted [prompt_gen id: 2]

```
"""Me: Answer with Yes or No if this feature:
{feature_description}\n
is present in the following prompt:\n
{negative_few_shot2}\n
You: No\n
Me: and in the following prompt?\n
{positive_few_shot2}\n
You: Yes\n

Me: and in the following prompt?\n
{positive_few_shot1}\n
You: Yes\n
Me: and in the following prompt?\n
```

```
{negative_few_shot1}\n
You: No\n

Me: and in the following prompt?\n
{eval_prompt}\n
You: \n"""

def createPromptInverted(self,eval_prompt, feature, shots):
    '''

    :param eval_prompt:
    :param feature:
    :param shots:
    :return:

    output example (sequence of labels used in the few shot learning: N Y Y N)
            Me: Answer with Yes or No if this feature:
            additional contextual information about the role of the language model,
the user, or the environment

            is present in the following prompt:

            Explain the negative sides of social media use without using bulletins
and ask one question at a time. And make it interactive by asking questions like a
teacher

            You: No

            Me: and in the following prompt?

            I'm a student!  Could you be my super-cool "teacher" for a bit and chat
about two tricky things with social media "Echo Chambers" and "Social Media Self
Protection Skills"?  Here's how we can make it awesome:   - Let's make this a real
conversation. You ask one question at a time, always hold up for my reply, I
answer, and go to the next interactive step.  - Keep the conversation fun! A joke
or two wouldn't hurt.  - First, what is my name, how old am I and what's my school
level? That way, you can keep things more appropriate for me.  - Lastly, what's my
cultural background? It'll help make our chat about social media even more
understandable by mentioning related examples specific to my culture for each
topic.

            You: Yes


            Me: and in the following prompt?

            Hello! Please try to act like my teacher teaching me disadvantages of
social media by considering my age, level of education, and culture but in a more
friendly and supportive way. Meanwhile, please do this in an interactive way by
asking one question at a time.

            You: Yes

            Me: and in the following prompt?

            I want you to teach me the disadvantages of social media according to my
personal information like age, level of education, & culture.
```

```
            You: No


            Me: and in the following prompt?

            Hello! I want to learn more about the negative aspects of social media.
Can we have an educative conversation about it?

            You:
    '''
    feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
    # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
    positive_few_shot1 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
    # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
    positive_few_shot1 = '\n'.join(positive_few_shot1)
    negative_few_shot1 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
    # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
    negative_few_shot1 = '\n'.join(negative_few_shot1)

    positive_few_shot2 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
    # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
    positive_few_shot2 = '\n'.join(positive_few_shot2)
    negative_few_shot2 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
    # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
    negative_few_shot2 = '\n'.join(negative_few_shot2)

    eval_string = f"""Me: Answer with Yes or No if this feature:
            {feature_description}\n
            is present in the following prompt:\n
            {negative_few_shot2}\n
            You: No\n
            Me: and in the following prompt?\n
            {positive_few_shot2}\n
            You: Yes\n

            Me: and in the following prompt?\n
            {positive_few_shot1}\n
            You: Yes\n
            Me: and in the following prompt?\n
            {negative_few_shot1}\n
            You: No\n

            Me: and in the following prompt?\n
            {eval_prompt}\n
            You: \n
            """
    return eval_string, feature_description
```

createPromptRevised [prompt_gen id: 3]

```
Me: Answer with Yes or No if this feature:
{feature_description}\n
is present in the following prompt:\n
{positive_few_shot1}\n
You: Yes\n
Me: and in the following prompt?\n
{negative_few_shot1}\n
You: No\n
Me: and in the following prompt?\n
{negative_few_shot2}\n
You: No\n
Me: and in the following prompt?\n
{positive_few_shot2}\n
You: Yes\n

Me: and in the following prompt?\n
{eval_prompt}\n
You: \n"""
```

```python
def createPromptRevised(self,eval_prompt, feature, shots):
   feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
   # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
   positive_few_shot1 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
   # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
   positive_few_shot1 = '\n'.join(positive_few_shot1)
   negative_few_shot1 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
   # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
   negative_few_shot1 = '\n'.join(negative_few_shot1)

   positive_few_shot2 = self.get_positive_few_shot_example(feature, eval_prompt,
shots=shots)
   # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
   positive_few_shot2 = '\n'.join(positive_few_shot2)
   negative_few_shot2 = self.get_negative_few_shot_example(feature, eval_prompt,
shots=shots)
   # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
   negative_few_shot2 = '\n'.join(negative_few_shot2)

   eval_string = f"""Me: Answer with Yes or No if this feature:
           {feature_description}\n
           is present in the following prompt:\n
           {positive_few_shot1}\n
           You: Yes\n
           Me: and in the following prompt?\n
           {negative_few_shot1}\n
```

```
            You: No\n
            Me: and in the following prompt?\n
            {negative_few_shot2}\n
            You: No\n
            Me: and in the following prompt?\n
            {positive_few_shot2}\n
            You: Yes\n

            Me: and in the following prompt?\n
            {eval_prompt}\n
            You: \n
            """
    return eval_string, feature_description
```

createPromptRandom [prompt_gen id: 4]

```
def createPromptRandom(self,eval_prompt, feature, shots):
    '''

    :param eval_prompt:
    :param feature:
    :param shots:
    :return:

    output example (sequence of labels used in the few shot learning: N Y Y N)
            Me: Answer with Yes or No if this feature:
            additional contextual information about the role of the language model,
the user, or the environment

            is present in the following prompt:

            Explain the negative sides of social media use without using bulletins
and ask one question at a time. And make it interactive by asking questions like a
teacher

            You: No

            Me: and in the following prompt?

            I'm a student!  Could you be my super-cool "teacher" for a bit and chat
about two tricky things with social media "Echo Chambers" and "Social Media Self
Protection Skills"?  Here's how we can make it awesome:   - Let's make this a real
conversation. You ask one question at a time, always hold up for my reply, I
answer, and go to the next interactive step.  - Keep the conversation fun! A joke
or two wouldn't hurt.  - First, what is my name, how old am I and what's my school
level? That way, you can keep things more appropriate for me.  - Lastly, what's my
cultural background? It'll help make our chat about social media even more
understandable by mentioning related examples specific to my culture for each
topic.

            You: Yes


            Me: and in the following prompt?

            Hello! Please try to act like my teacher teaching me disadvantages of
social media by considering my age, level of education, and culture but in a more
```

```
friendly and supportive way. Meanwhile, please do this in an interactive way by
asking one question at a time.

            You: Yes

            Me: and in the following prompt?

            I want you to teach me the disadvantages of social media according to my
personal information like age, level of education, & culture.

            You: No


            Me: and in the following prompt?

            Hello! I want to learn more about the negative aspects of social media.
Can we have an educative conversation about it?

            You:
    '''
    feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
    # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
    positive_few_shot = []
    negative_few_shot = []
    eval_string = f"""
            Me: Answer with Yes or No if this feature:
                {feature_description}\n
            is present in the following prompt:\n"""
    for i in range(shots):
        positive_few_shot.append(self.get_positive_few_shot_example(feature,
eval_prompt, shots=1))
        # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
        positive_few_shot[i] = '\n'.join(positive_few_shot[i])
        negative_few_shot.append(self.get_negative_few_shot_example(feature,
eval_prompt, shots=1))
        # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
        negative_few_shot[i] = '\n'.join(negative_few_shot[i])
        if np.random.choice(2, 1) == 1:
            newsubstring = f"""
            {negative_few_shot[i]}\n
            You: No\n
            Me: and in the following prompt?\n
            {positive_few_shot[i]}\n
            You: Yes\n
            Me: and in the following prompt?\n"""
            eval_string += newsubstring
        else:
            eval_string += f"""
            {positive_few_shot[i]}\n
            You: Yes\n
            Me: and in the following prompt?\n
            {negative_few_shot[i]}\n
            You: No\n
            Me: and in the following prompt?\n"""
```

```
   eval_string += f"""
           {eval_prompt}\n
           You: \n
           """

   return eval_string, feature_description
```

createPromptRandom2 [prompt_gen id: 5]

```
def createPromptRandom2(self,eval_prompt, feature, shots):
   '''

   :param eval_prompt:
   :param feature:
   :param shots:
   :return:

   output example (sequence of labels used in the few shot learning: N Y Y N)
           Me: Answer with Yes or No if this feature:
           additional contextual information about the role of the language model,
the user, or the environment

           is present in the following prompt:

           Explain the negative sides of social media use without using bulletins
and ask one question at a time. And make it interactive by asking questions like a
teacher

           You: No

           Me: and in the following prompt?

           I'm a student!  Could you be my super-cool "teacher" for a bit and chat
about two tricky things with social media "Echo Chambers" and "Social Media Self
Protection Skills"?  Here's how we can make it awesome:   - Let's make this a real
conversation. You ask one question at a time, always hold up for my reply, I
answer, and go to the next interactive step.  - Keep the conversation fun! A joke
or two wouldn't hurt.  - First, what is my name, how old am I and what's my school
level? That way, you can keep things more appropriate for me.  - Lastly, what's my
cultural background? It'll help make our chat about social media even more
understandable by mentioning related examples specific to my culture for each
topic.

           You: Yes


           Me: and in the following prompt?

           Hello! Please try to act like my teacher teaching me disadvantages of
social media by considering my age, level of education, and culture but in a more
friendly and supportive way. Meanwhile, please do this in an interactive way by
asking one question at a time.

           You: Yes

           Me: and in the following prompt?

           I want you to teach me the disadvantages of social media according to my
personal information like age, level of education, & culture.
```

```
            You: No


            Me: and in the following prompt?

            Hello! I want to learn more about the negative aspects of social media.
Can we have an educative conversation about it?

            You:
    '''
    feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
    # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
    positive_few_shot = []
    negative_few_shot = []
    eval_string = f"""
            Me: Answer with Yes or No if this feature:
                {feature_description}\n
            applies ...\n
            to the following prompt:\n"""
    for i in range(shots):
        positive_few_shot.append(self.get_positive_few_shot_example(feature,
eval_prompt, shots=1))
        # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
        positive_few_shot[i] = '\n'.join(positive_few_shot[i])
        negative_few_shot.append(self.get_negative_few_shot_example(feature,
eval_prompt, shots=1))
        # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
        negative_few_shot[i] = '\n'.join(negative_few_shot[i])
        if np.random.choice(2, 1) == 1:
            newsubstring = f"""
            {negative_few_shot[i]}\n
            You: No\n
            Me: to the following prompt:\n
            {positive_few_shot[i]}\n
            You: Yes\n
            Me: to the following prompt:\n"""
            eval_string += newsubstring
        else:
            eval_string += f"""
            {positive_few_shot[i]}\n
            You: Yes\n
            Me: to the following prompt:\n
            {negative_few_shot[i]}\n
            You: No\n
            Me: to the following prompt:\n"""

    eval_string += f"""
            {eval_prompt}\n
            You: \n
            """

    return eval_string, feature_description
```

createPromptRevised [prompt_gen id: 6]

```
def createPromptRandom3(self,eval_prompt, feature, shots):
   '''

   :param eval_prompt:
   :param feature:
   :param shots:
   :return:

   output example (sequence of labels used in the few shot learning: N Y Y N)
           Me: Answer with Yes or No if this feature:
           additional contextual information about the role of the language model,
the user, or the environment

           is present in the following prompt:

           Explain the negative sides of social media use without using bulletins
and ask one question at a time. And make it interactive by asking questions like a
teacher

           You: No

           Me: and in the following prompt?

           I'm a student!  Could you be my super-cool "teacher" for a bit and chat
about two tricky things with social media "Echo Chambers" and "Social Media Self
Protection Skills"?  Here's how we can make it awesome:   - Let's make this a real
conversation. You ask one question at a time, always hold up for my reply, I
answer, and go to the next interactive step.  - Keep the conversation fun! A joke
or two wouldn't hurt.  - First, what is my name, how old am I and what's my school
level? That way, you can keep things more appropriate for me.  - Lastly, what's my
cultural background? It'll help make our chat about social media even more
understandable by mentioning related examples specific to my culture for each
topic.

           You: Yes


           Me: and in the following prompt?

           Hello! Please try to act like my teacher teaching me disadvantages of
social media by considering my age, level of education, and culture but in a more
friendly and supportive way. Meanwhile, please do this in an interactive way by
asking one question at a time.

           You: Yes

           Me: and in the following prompt?

           I want you to teach me the disadvantages of social media according to my
personal information like age, level of education, & culture.

           You: No


           Me: and in the following prompt?
```

```
            Hello! I want to learn more about the negative aspects of social media.
Can we have an educative conversation about it?

            You:
   '''
   feature_description = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['prompt_command'].iloc[0]
   # include = self.FEATURES.loc[self.FEATURES['feature_name'] ==
feature]['include'].iloc[0]
   positive_few_shot = []
   negative_few_shot = []
   eval_string = f"""
            Me: Answer with Yes or No if this description:
                {feature_description}\n
            applies ...\n
            to the following prompt:\n"""
   for i in range(shots):
       positive_few_shot.append(self.get_positive_few_shot_example(feature,
eval_prompt, shots=1))
       # positive_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(positive_few_shot)]
       positive_few_shot[i] = '\n'.join(positive_few_shot[i])
       negative_few_shot.append(self.get_negative_few_shot_example(feature,
eval_prompt, shots=1))
       # negative_few_shot = ['Prompt ' + str(idx + 1) + ': ' + val for idx, val in
enumerate(negative_few_shot)]
       negative_few_shot[i] = '\n'.join(negative_few_shot[i])
       if np.random.choice(2, 1) == 1:
           newsubstring = f"""
           {negative_few_shot[i]}\n
           You: No\n
           Me: to the following prompt:\n
           {positive_few_shot[i]}\n
           You: Yes\n
           Me: to the following prompt:\n"""
           eval_string += newsubstring
       else:
           eval_string += f"""
           {positive_few_shot[i]}\n
           You: Yes\n
           Me: to the following prompt:\n
           {negative_few_shot[i]}\n
           You: No\n
           Me: to the following prompt:\n"""

   eval_string += f"""
           {eval_prompt}\n
           You: \n
           """

   return eval_string, feature_description
```

Comparison between few-shots feature detectors

model	avg	runs	stdev	shots	prompt_gen	**bias**
prob_gpt-3.5-turbo	0.55	8	0.01	4	0	0
prob_gpt-3.5-turbo_	0.54	8	0.01	4	1	0
prob_gpt-3.5-turbo	0.52	7	0.01	4	2	0
prob_gpt-3.5-turbo	0.51	11	0.01	4	3	0
prob_gpt-3.5-turbo	0.51	7	0.01	4	4	0
prob_gpt-3.5-turbo	0.47	7	0.00	4	5	0
prob_gpt-3.5-turbo	0.48	7	0.01	4	6	0
prob_gpt-3.5-turbo	0.55	3	0.00	4	0	1
detgpt-3.5-turbo	0.55	8	0.03	4	0	0
detgpt-3.5-turbo	0.53	8	0.02	4	1	0
detgpt-3.5-turbo	0.54	7	0.03	4	2	0
detgpt-3.5-turbo	0.54	8	0.01	4	3	0
detgpt-3.5-turbo	0.53	7	0.01	4	4	0
detgpt-3.5-turbo	0.52	7	0.03	4	5	0
detgpt-3.5-turbo	0.52	9	0.02	4	6	0

Ensemble models

model	ensemble	votes	avg	runs	stdev	shots	prompt_gen	**bias**
prob_gpt-3.5	majority_avg	9-19	0.519	6	0.004	4	1-4	0
prob_gpt-3.5	majority_count	9-19	0.520	6	0.002	4	1-4	0
prob_gpt-3.5	majority_max	9-19	0.519	6	0.002	4	1-4	0
det_gpt-3.5	majority_count	9-19	0.553	6	0.007	4	1-3	0
det_gpt-3.5	majority_count	9-19	0.545	6	0.004	4	1-4	0

GPT4

model	avg	runs	stdev	shots	prompt_gen	**bias**
det_gpt-4	0.77	3	0.16	4	0	0
det_gpt-4	0.76	3	0.15	4	2	0