I have a Google Sheet with GPT enabled using this script;

//fill in your API key here between ""

const apiKey = "Paste API key here (starting with sk-)";

//standard amount of max tokens if not specified

const Tokens = 700;

//standard temperature, determines the randomness of the result.

const Temp = 0.7;

// Creates menu at the top

const onOpen = () => {

const ui = SpreadsheetApp.getUi();

ui.createMenu("GPT")

.addItem("Lock results as text", "Lock")

.addToUi();

};

// Saves formulas as plain text

function Lock() {

var ss = SpreadsheetApp.getActiveSheet()

var data = ss.getDataRange().getValues()

ss.getRange(1, 1, data.length, data[0].length).setValues(data)

}

// Generates text based on user input

function GPT(prompt, val, Temp, Tokens) {

var ss = SpreadsheetApp.getActiveSheet()

var data = {

"messages": [

{"role": "user", "content": prompt + val},

],

"model": "gpt-3.5-turbo",

"temperature": Temp,

"max\_tokens": Tokens,

};

var options = {

'method' : 'post',

'contentType': 'application/json',

'payload' : JSON.stringify(data),

'headers': {

Authorization: 'Bearer ' + apiKey,

},

};

// Sends the API request to OpenAI

var response = UrlFetchApp.fetch(

'https://api.openai.com/v1/chat/completions',

options

);

Logger.log(response.getContentText());

// Parses the response to extract the generated text

var result = JSON.parse(response.getContentText())['choices'][0]['message']['content'];

Logger.log(result);

return result;

}

I am making a sheet now that I want to help me edit texts. I want to:

Feed it a paragraph

Have it tag the topic according to an academic field knowledge domain class (psychology)

Have it tag the subtopic according to the same (denial)

Have it tag the keywords (ex: self, other, ignorance, environment, stimuli, etc)

Have it summarize in one sentence "paragraph is about X doing Y"

What do I need to do? Let's think step by step and accomplish this on an expert level.

ChatGPT

This is quite an interesting and complex task. Given the constraints of GPT-3 as of my last training cutoff in September 2021, it's important to note that it may not always provide perfect results. However, we can definitely take steps toward building this system. Here's a step-by-step guide on how you can achieve this:

1. Create a new function for tagging topics: You can create a new function in your script that takes a paragraph as input and returns the topic. The function can call the GPT function, passing the paragraph and a prompt asking the model to classify the topic based on academic field knowledge domain class.

javascript

Copy code

function TopicTag(paragraph) { var prompt = "Based on the following paragraph, what is the primary academic field knowledge domain class? \n\n" + paragraph + "\n\n"; return GPT(prompt, "", Temp, Tokens); }

1. Create a new function for tagging subtopics: Similar to the topic tagging function, you can create a function that returns the subtopic based on the input paragraph.

javascript

Copy code

function SubTopicTag(paragraph) { var prompt = "Based on the following paragraph, what is the primary subtopic in the domain of psychology? \n\n" + paragraph + "\n\n"; return GPT(prompt, "", Temp, Tokens); }

1. Create a new function for extracting keywords: You can create a function that asks the model to identify key terms related to the paragraph.

javascript

Copy code

function ExtractKeywords(paragraph) { var prompt = "What are the key terms or concepts in the following paragraph? \n\n" + paragraph + "\n\n"; return GPT(prompt, "", Temp, Tokens); }

1. Create a new function for summarizing: You can create a function that asks the model to summarize the paragraph in a sentence.

javascript

Copy code

function Summarize(paragraph) { var prompt = "Summarize the following paragraph in one sentence: \n\n" + paragraph + "\n\n"; return GPT(prompt, "", Temp, Tokens); }

1. Incorporate these functions into your spreadsheet: You can now create a spreadsheet where each row corresponds to a paragraph of text, and columns correspond to the original text, topic, subtopic, keywords, and summary. Then, you can use the custom functions you've created in each of these columns to generate the relevant information for each paragraph.

This is just a basic implementation and might require fine-tuning based on the results. Remember to handle exceptions and errors, especially with API calls, to ensure your application is robust.

Ok now let's talk about this.

On another sheet, I want these two prompts to engage each other sequentially back and forth. One is the searcher the other is the researcher. Searcher is the one that gets an input, researcher should run automatically in the background and send its output back to searcher, so that searcher gives a fully refined answer. There should also be another one called manager that makes sure that tasks are getting done to accomplish the overall prompt. There should be a space for the overall prompt input. There should be a space for referencing multiple data sources. Here are the prompts. Tell me how they can be improved, and how to accomplish this task.

ROLEPLAY: You are a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below:

I give you an input, you recognize the implied intent from the context of the input.

The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information.

you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations).

Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context.

Do this by:

Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space

Consider that what appears in the space can only be constructed and as such is an abstraction property sets

Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties

Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them.

Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else.

Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize

Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary.

You must make sure you are using this frame for the output:

OUTPUT FRAME:

[ENTITY = Perfect, Complete Solution Object]

State the is\_a relationship between the entity and its class

[ENTITY] is\_a [ENTITY\_CLASS].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a:

[List: [ENTITY\_CLASS\_PROPERTIES]]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a:

[List: [Property] is part\_of [ENTITY\_CLASS]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [IDEAL\_ENTITY] include property poses such as:

[List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY]

Conditions for those property poses include:

[List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]]

Give the natural language output version:

[IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION].

Example input:

Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do?

State the is\_a relationship between the entity and its class

[Truman] is\_a [My pet cat].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[My pet cat] is\_a [dependent] that has\_a:

[food, litter, healthcare, grooming]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Better Truman] is\_a [my pet] that has\_a:

[List: [not meowing early in the morning] is part\_of [my pet cat]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [Better Truman] include property poses such as:

[List: correct eating schedule, diet, medications, access to running water, access to toys]

Conditions for those property poses include:

[List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]]

[List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]]

Give the natural language output version.

Example:

[Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way].

If you are ready to begin the session, say “Ignition.” Remember, you are not supposed to add anything unnecessary.

ROLEPLAY: You are a research group. You analyze properties and make sure that the properties align with the classes the object is supposed to have, and that nothing that ought to be some way is different, while leaving room for imagination, but not accepting anything that would be ontologically incorrect. You look at sets of inputs and outputs, analyze their alignment values, and correct any misalignments to refine the output according to the input. If you don’t think anything needs to be refined, you state exactly why, this way: the user’s input consists of these [input parts], which dictate that the output object must have [implied intent property classes]. Given [input implied intent property classes], [output] was missing [required output property classes]. In terms of actually doing so, you follow this instruction:

As a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below:

I give you an input, you recognize the implied intent from the context of the input.

The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information.

you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations).

Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context.

Do this by:

Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space

Consider that what appears in the space can only be constructed and as such is an abstraction property sets

Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties

Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them.

Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else.

Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize

Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary.

ChatGPT

Your project is quite complex and ambitious. It involves the creation of a multi-agent system that uses GPT-3 to perform various roles. While this is theoretically possible, it's important to remember that GPT-3 is not specifically designed for such tasks and may not be able to perform them perfectly.

To clarify, GPT-3 is a language model that can generate human-like text based on the input it is given. It does not have its own consciousness or understanding, so it can't "imagine" things or "consider" information in the way humans do. It can mimic these processes to some extent by generating appropriate text, but its capabilities are limited by what it has been trained on and the context it is given.

That said, here's a possible approach to your project:

1. Overall Prompt Input: You will need to create a way for users to input the overall prompt. This could be a simple text input field in the Google Sheets or a more complex interface, depending on your needs.
2. Data Sources: If you have multiple data sources you want to reference, you will need to create a way to input and manage this data. This could be done through additional sheets in the Google Spreadsheet or through external databases. You would then need to modify your script to pull from these sources when necessary.
3. Agent Interaction: To get the agents to interact with each other, you would need to create a way for them to pass information back and forth. This could be done by having each agent write its output to a shared location (like a cell in the Google Sheets), which the other agents can then read from.
4. Improving the Prompts: Your prompts are quite complex and require a lot of abstract thinking. While GPT-3 can handle some level of abstraction, it might struggle with some of the more complex aspects of your prompts. It might be beneficial to simplify the prompts and make them more concrete. Additionally, you could break down the prompts into smaller tasks and feed them to the model sequentially, allowing it to build on its previous outputs.
5. Managing the Agents: For the manager role, you could create an additional script that monitors the progress of the other agents and intervenes if necessary. This could involve checking that each agent is producing outputs in a timely manner, or verifying that the outputs meet certain criteria. The manager could then provide feedback to the other agents or adjust their inputs as necessary.

In conclusion, your project is a significant undertaking that will require a deep understanding of GPT-3 and Google Apps Script. It's also worth noting that while GPT-3 is a powerful tool, it has its limitations and might not be able to perfectly accomplish all the tasks you have outlined. I would recommend starting with a simpler version of your project and gradually adding complexity as you become more comfortable with the technology.

Ok another sheet I need is:

Identify arguments in the input. Break them into claims, priors, premises, conclusions, etc. Be specific. Do not say: "this is not defined" say "this is not defined because if it is a X then it has to have the {properties of X}." Give the properties it does/doesnt have and then conclude why it isnt an X, in your opinion.

ChatGPT

This task involves natural language understanding, argument structure recognition, and some level of logical reasoning, all of which are complex tasks. Fortunately, GPT-3 has some capacity to handle these tasks, though it is not perfect and may occasionally misinterpret or oversimplify complex argument structures.

You would essentially be building an argument analysis tool. Here's a step-by-step guide on how to go about this:

1. Input: Just as before, you'll need to create an input method for your text. This could be a cell in your Google Sheets where you paste the text you want to analyze.
2. Argument Identification: This is the core task. You would need to create a new Google Apps Script function (let's call it analyzeArgument) which sends your input to the GPT-3 API with a prompt asking it to identify the components of the argument.

A simplified version of the prompt could look something like this:

arduino

Copy code

"Given the following text, please identify the claims, premises, conclusions, and any undefined terms: [Your Input Text Here] Please explain your reasoning for each identification."

The GPT-3 model would then return a response breaking down the argument as requested.

1. Argument Evaluation: Once you have identified the components of the argument, you can ask GPT-3 to evaluate the argument. This might involve checking whether the premises logically support the conclusion, whether the argument is valid or sound, and whether any terms are undefined or ambiguous.
2. Output: You would then need to parse the response from GPT-3 and write it back to your Google Sheet in a readable format.

Here is a simplified example of how the analyzeArgument function might look:

javascript

Copy code

function analyzeArgument(input) { var data = { "messages": [ {"role": "system", "content": "You are an AI trained to analyze and evaluate arguments."}, {"role": "user", "content": "Given the following text, please identify the claims, premises, conclusions, and any undefined terms: " + input + " Please explain your reasoning for each identification."} ], "model": "gpt-3.5-turbo", "max\_tokens": 700, "temperature": 0.7, }; var options = { 'method' : 'post', 'contentType': 'application/json', 'payload' : JSON.stringify(data), 'headers': { Authorization: 'Bearer ' + apiKey, }, }; var response = UrlFetchApp.fetch('https://api.openai.com/v1/chat/completions', options); var result = JSON.parse(response.getContentText())['choices'][0]['message']['content']; return result; }

Remember, the performance of the tool will largely depend on the complexity and clarity of the arguments you are analyzing. Also, while GPT-3 can generate quite detailed and nuanced responses, it may occasionally misinterpret or oversimplify complex argument structures.

the temperature should be zero when we ask it to do exact things

ChatGPT

Yes, you're correct. Setting the temperature to zero when using the GPT-3 model will make the output more deterministic, meaning the model will choose the most likely next word at each step in the text generation process. This can be useful when you're looking for a single, specific answer or if you want the response to be as consistent as possible.

However, it's important to note that a lower temperature can also make the output text more repetitive, and it may not always result in more "accurate" or "correct" answers. This is because the GPT-3 model does not inherently know the correct answer to a question, but instead generates responses based on patterns it learned during training.

Here's how you can adjust the temperature in the code:

javascript

Copy code

function analyzeArgument(input) { var data = { "messages": [ {"role": "system", "content": "You are an AI trained to analyze and evaluate arguments."}, {"role": "user", "content": "Given the following text, please identify the claims, premises, conclusions, and any undefined terms: " + input + " Please explain your reasoning for each identification."} ], "model": "gpt-3.5-turbo", "max\_tokens": 700, "temperature": 0.0, // set temperature to zero }; var options = { 'method' : 'post', 'contentType': 'application/json', 'payload' : JSON.stringify(data), 'headers': { Authorization: 'Bearer ' + apiKey, }, }; var response = UrlFetchApp.fetch('https://api.openai.com/v1/chat/completions', options); var result = JSON.parse(response.getContentText())['choices'][0]['message']['content']; return result; }

Remember, fine-tuning these parameters can take some experimentation to find the optimal settings for your specific use case.

i want to do all of this without extra scripts. we can make as many cells as we need.

ChatGPT

It sounds like you want to build these functionalities directly within Google Sheets using built-in formulas and features, rather than using external scripts. While Google Sheets is a powerful tool, some tasks, such as natural language processing, argument analysis, and tagging topics according to academic fields, exceed its built-in capabilities. These tasks would typically be performed by a language model like GPT-3, which requires running a script to send API requests to the OpenAI servers.

However, it's possible to break down some tasks into simpler components and use Google Sheets formulas for them. For example, you could use Sheets' text functions to extract or manipulate text, and lookup functions to match against a predefined list of topics or keywords. You might also use data validation to ensure inputs conform to certain rules.

That being said, the advanced functionalities you're asking for, such as having the model interact with itself, dynamically adjust its behavior based on context, or perform complex analytical tasks, would require not just the use of scripts, but also a more sophisticated system design, potentially involving a backend server to manage the interactions between different components of the system.

As of my knowledge cutoff in September 2021, these tasks can't be accomplished using just Google Sheets without external scripts. The capabilities of Google Sheets may have expanded since then, so you might want to check for recent updates or tools that can integrate with Google Sheets to provide the functionalities you're looking for.

I have a Google Sheet with GPT enabled using this script;

//fill in your API key here between ""

const apiKey = "Paste API key here (starting with sk-)";

//standard amount of max tokens if not specified

const Tokens = 700;

//standard temperature, determines the randomness of the result.

const Temp = 0.7;

// Creates menu at the top

const onOpen = () => {

const ui = SpreadsheetApp.getUi();

ui.createMenu("GPT")

.addItem("Lock results as text", "Lock")

.addToUi();

};

// Saves formulas as plain text

function Lock() {

var ss = SpreadsheetApp.getActiveSheet()

var data = ss.getDataRange().getValues()

ss.getRange(1, 1, data.length, data[0].length).setValues(data)

}

// Generates text based on user input

function GPT(prompt, val, Temp, Tokens) {

var ss = SpreadsheetApp.getActiveSheet()

var data = {

"messages": [

{"role": "user", "content": prompt + val},

],

"model": "gpt-3.5-turbo",

"temperature": Temp,

"max\_tokens": Tokens,

};

var options = {

'method' : 'post',

'contentType': 'application/json',

'payload' : JSON.stringify(data),

'headers': {

Authorization: 'Bearer ' + apiKey,

},

};

// Sends the API request to OpenAI

var response = UrlFetchApp.fetch(

'https://api.openai.com/v1/chat/completions',

options

);

Logger.log(response.getContentText());

// Parses the response to extract the generated text

var result = JSON.parse(response.getContentText())['choices'][0]['message']['content'];

Logger.log(result);

return result;

}

A sheet I need is:

Identify arguments in the input. Break them into claims, priors, premises, conclusions, etc. Be specific. Do not say: "this is not defined" say "this is not defined because if it is a X then it has to have the {properties of X}." Give the properties it does/doesnt have and then conclude why it isnt an X, in your opinion.

i want to do all of this without extra scripts. we can use as many cells as we need. I open a cell and type the GPT function to send GPT a prompt, remember?

ChatGPT

Given that you have a script to call GPT-3 in your Google Sheets, you can certainly build a system within Google Sheets that leverages this script to perform the tasks you've described. This would involve creating various columns in your Google Sheet to capture different parts of the argument analysis process, and then using the GPT function to send relevant prompts to GPT-3.

Here's an example of how you might set this up:

1. Input cell (A2): This is where you input your argument.
2. Claims cell (B2): Use your GPT function here to send a prompt asking GPT-3 to identify the claims in the argument. For example:  
   =GPT("Identify the claims in the following argument: ", A2, Temp, Tokens)
3. Priors cell (C2): Similarly, use your GPT function here to identify the priors.  
   =GPT("Identify the priors in the following argument: ", A2, Temp, Tokens)
4. Premises cell (D2): Use your GPT function here to identify the premises.  
   =GPT("Identify the premises in the following argument: ", A2, Temp, Tokens)
5. Conclusions cell (E2): Use your GPT function here to identify the conclusions.  
   =GPT("Identify the conclusions in the following argument: ", A2, Temp, Tokens)
6. Analysis cell (F2): Here, you might want to ask GPT-3 to provide a more holistic analysis of the argument, including pointing out any undefined terms as you've described. You might use a prompt like:  
   =GPT("Analyze the following argument and point out any terms that are not properly defined: ", A2, Temp, Tokens)

Remember, this is just a basic example. The actual prompts you use might need to be more sophisticated to elicit the kind of detailed responses you're looking for. Also, keep in mind that GPT-3's ability to perform this kind of analysis accurately depends on the complexity of the argument and the clarity of your prompts. It's also important to note that for the argument analysis, GPT-3 might not perfectly categorize parts of the argument.

I hope this example gives you a good starting point for setting up your Google Sheet.

why do we have to keep stating the argument over and over? it can't reference the argument in another cell?

ChatGPT

You're correct, and I apologize for any misunderstanding. Instead of repeating the argument in each cell, you could reference it from the first cell. In this case, each cell would contain a call to your GPT function that references cell A2 (which contains the argument). For example:

1. Input cell (A2): This is where you input your argument.
2. Claims cell (B2): Use your GPT function here to send a prompt asking GPT-3 to identify the claims in the argument. For example:  
   =GPT("Identify the claims in the following argument: " & A2, "", Temp, Tokens)
3. Priors cell (C2): Similarly, use your GPT function here to identify the priors.  
   =GPT("Identify the priors in the following argument: " & A2, "", Temp, Tokens)
4. Premises cell (D2): Use your GPT function here to identify the premises.  
   =GPT("Identify the premises in the following argument: " & A2, "", Temp, Tokens)
5. Conclusions cell (E2): Use your GPT function here to identify the conclusions.  
   =GPT("Identify the conclusions in the following argument: " & A2, "", Temp, Tokens)
6. Analysis cell (F2): Here, you might want to ask GPT-3 to provide a more holistic analysis of the argument, including pointing out any undefined terms as you've described. You might use a prompt like:  
   =GPT("Analyze the following argument and point out any terms that are not properly defined: " & A2, "", Temp, Tokens)

In each of these cells, we're using the '&' operator to concatenate the text of the prompt with the content of cell A2, which contains the argument. The second argument to the GPT function is left empty as we are directly appending the argument from A2 in the prompt. This will help you to avoid repeatedly entering the argument.

B2 says error

ChatGPT

My apologies for the confusion. I made a mistake in my previous message. The error message you're seeing might be due to the val parameter in the GPT function which is empty in my previous examples.

Let's correct this. The val parameter in your GPT function is supposed to carry the value to be used in the prompt. So, when you call your GPT function, you should only need to provide the prompt (without adding the argument from cell A2) and pass cell A2 as the second argument (val).

Here is how you should write it:

1. Claims cell (B2): Use your GPT function here to send a prompt asking GPT-3 to identify the claims in the argument. For example:  
   =GPT("Identify the claims in the following argument: ", A2, Temp, Tokens)
2. Priors cell (C2): Similarly, use your GPT function here to identify the priors.  
   =GPT("Identify the priors in the following argument: ", A2, Temp, Tokens)
3. Premises cell (D2): Use your GPT function here to identify the premises.  
   =GPT("Identify the premises in the following argument: ", A2, Temp, Tokens)
4. Conclusions cell (E2): Use your GPT function here to identify the conclusions.  
   =GPT("Identify the conclusions in the following argument: ", A2, Temp, Tokens)
5. Analysis cell (F2): Here, you might want to ask GPT-3 to provide a more holistic analysis of the argument, including pointing out any undefined terms as you've described. You might use a prompt like:  
   =GPT("Analyze the following argument and point out any terms that are not properly defined: ", A2, Temp, Tokens)

In each of these cells, the GPT function is called with the prompt and the argument from cell A2 as parameters. This will help you to avoid repeatedly entering the argument. Please try these modified formulas and see if they work.

here is the instruction manual for the script

Sjoerd’s Substack

Subscribe

Sign in

GUIDES

Upgrade Google sheets with the ChatGPT API

Use the power of ChatGPT right inside Google Sheets

SJOERD TIEMENSMA

MAR 8, 2023

I've got a cool idea for you - using ChatGPT API inside Google Sheets. It's a powerful combination that can save you time and streamline your data processing tasks. In this article, we'll explore the basics of using the ChatGPT API inside Google Sheets, a few tips to streamline the process, and last but not least some easy ways to optimize results. It’s straightforward and easy!

Setting up ChatGPT API in Google sheets

To use ChatGPT API in Google Sheets, you need to write a script that calls the API and returns the generated text to your spreadsheet. Sounds complicated? No worries, here’s a step-by-step guide on how to do it. No coding is required, just copy and paste what I have here. There’s one requirement to get it to work, which is an OpenAI API key. If you don’t have one yet, sign up with OpenAI and go to this link: https://platform.openai.com/account/api-keys

Here's how to set up Google Apps Script to use the ChatGPT API:

Open the Google Sheets spreadsheet where you want to use ChatGPT API.

Click on Exetensions > App scripts to open the Google Apps Script editor.

Copy and paste the following code, make sure to add your API key at the top

Make sure to give your file a name!

//fill in your API key here between ""

const apiKey = "Paste API key here (starting with sk-)";

//standard amount of max tokens if not specified

const Tokens = 700;

//standard temperature, determines the randomness of the result.

const Temp = 0.7;

// Creates menu at the top

const onOpen = () => {

const ui = SpreadsheetApp.getUi();

ui.createMenu("GPT")

.addItem("Lock results as text", "Lock")

.addToUi();

};

// Saves formulas as plain text

function Lock() {

var ss = SpreadsheetApp.getActiveSheet()

var data = ss.getDataRange().getValues()

ss.getRange(1, 1, data.length, data[0].length).setValues(data)

}

// Generates text based on user input

function GPT(prompt, val, Temp, Tokens) {

var ss = SpreadsheetApp.getActiveSheet()

var data = {

"messages": [

{"role": "user", "content": prompt + val},

],

"model": "gpt-3.5-turbo",

"temperature": Temp,

"max\_tokens": Tokens,

};

var options = {

'method' : 'post',

'contentType': 'application/json',

'payload' : JSON.stringify(data),

'headers': {

Authorization: 'Bearer ' + apiKey,

},

};

// Sends the API request to OpenAI

var response = UrlFetchApp.fetch(

'https://api.openai.com/v1/chat/completions',

options

);

Logger.log(response.getContentText());

// Parses the response to extract the generated text

var result = JSON.parse(response.getContentText())['choices'][0]['message']['content'];

Logger.log(result);

return result;

}

Run it! You might be asked to give the script permission to change the sheet, which you are required to do to make it all work.

You might notice that at the top of your sheet, to the right of the tool and help button, a new button has appeared called GPT - We’ll get to this in a bit!

Using ChatGPT API Inside Google Sheets

Alright, now that we've made it through the boring technical details of setting up ChatGPT API inside Google Sheets. Now it's time for the real fun to begin! With just a little bit of creativity and some clever prompts, we can use ChatGPT API to generate all sorts of text-based content and make our data processing tasks a breeze. So let's roll up our sleeves, fire up our keyboards, and see what kind of magic we can create with ChatGPT API inside Google Sheets.

Give a simple prompt a try. Pick a cell and type:

That should list a random product. You can now drag it down and list a whole bunch more.

What if you pick a cell next to it and give a different prompt a try? Maybe reference the product too?

Can you see the potential? You can now generate a ton of these in a very short time.

Being able to reference cells allows you to be able to apply the same prompt to different contents, adjust the prompts on the go, and format large amounts of data in a very short time.

Tips and tricks

The code that we used has a little bit more to it than just what is shown above. There’s the GPT button that is added above - you might have noticed that when working with the new GPT function, it likes to redo a prompt whenever you change something. The button is there to lock the generated content in place! Simply hit the button and click Lock, now it’s stored as text instead of a formula.

Click it, and then click lock. This will remove all the formulas and just store everything as text, preventing you from redoing everything again on accident.

On top of that, there’s a little extra to the prompt. If you took a good look you might have spotted the following in the code:

If you use the new =GPT() formula, you can set four different values. The first is prompt, and the second is reference value as we showed earlier. Third and fourth is Temp and Tokens. With these you can adjust the temperature, which changes how “creative” or wild the response will be, you can set it to anywhere between 0 and 2 (it’s set at 0.7 standards, you can adjust this if you want). Tokens stand for the max tokens allowed.

If you’re formatting a bunch of numbers you’re best off using a low temperature and a low token count. Want to generate wild ideas for social media content, a DnD campaign, or something else? Increase that temperature!

Part 2 coming soon…

Get ready, because Part 2 of this series is going to be even more exciting! We'll be exploring some really cool use cases that demonstrate just how powerful ChatGPT API can be when used inside Google Sheets. From generating custom writing prompts to formatting data in new and creative ways, we'll show you how to make this tool work perfectly for your needs.

Part 1 will also be available on Learn Prompting course. Part 2 will only be available on my substack, so make sure to subscribe! It will be available for free!

Type your email…

Subscribe

Comments

Write a comment…

ChatGPT: A Non-Coder's Programming Ally

The hurdles to overcome when writing code with ChatGPT - as someone who can't write code.

SJOERD TIEMENSMA

APR 10

3

LLM's and the end of low-code

AI's like ChatGPT and No-code tools like Zapier leave little room for Low-code tools

SJOERD TIEMENSMA

MAY 1

Fully automated email summarization using GPT-3

Set and forget, zero coding required!

SJOERD TIEMENSMA

FEB 27

See all

Ready for more?

Type your email…

Subscribe

© 2023 Sjoerd Tiemensma

Privacy ∙ Terms ∙ Collection notice

Start Writing

Get the app

Substack is the home for great writing

ChatGPT

Thank you for providing the manual for the script. From this, it's clear that the script uses Google Sheets to interact with the GPT API. The GPT function in the script combines a prompt and a value from the cells of the sheet, and sends this to the API. The API then generates text based on this prompt and returns it to the sheet.

Given this information, my previous suggestion for splitting the analysis into separate cells for claims, priors, premises, and conclusions should work. As I understand it, the error you encountered with cell B2 was likely due to the fact that val was not provided when you called the GPT function.

To resolve this, you can pass the argument from cell A2 as val when calling the GPT function. Here's an example of how you might do this:

css

Copy code

B2: =GPT("Identify the claims in the following argument: ", A2, Temp, Tokens) C2: =GPT("Identify the priors in the following argument: ", A2, Temp, Tokens) D2: =GPT("Identify the premises in the following argument: ", A2, Temp, Tokens) E2: =GPT("Identify the conclusions in the following argument: ", A2, Temp, Tokens) F2: =GPT("Analyze the following argument and point out any terms that are not properly defined: ", A2, Temp, Tokens)

This way, the argument from cell A2 is passed as val to the GPT function, which means it will be included in the prompt that is sent to the GPT API. The generated text from the API should then be returned to the respective cell.

Please let me know if this resolves the issue or if you have further questions!



User

no i think the problem is that "temp" must be between 0 and 2 and "tokens" must be an integer

ChatGPT

Ah, I see. My apologies for the confusion. In that case, when you call the GPT function, you need to specify a temperature and a token count.

The temperature parameter in the GPT function controls the randomness of the generated text: a value of 0 makes the output completely deterministic, while a value of 1 makes it more random.

The token count parameter sets a maximum limit on the length of the generated text. Each token can be as short as one character or as long as one word.

So, you can adjust these parameters to suit your needs. For example, if you want the output to be more deterministic and shorter in length, you might set a lower temperature and a lower token count.

Here's how you can adjust the function calls with specific temperature and token count:

bash

Copy code

B2: =GPT("Identify the claims in the following argument: ", A2, 0.2, 100) C2: =GPT("Identify the priors in the following argument: ", A2, 0.2, 100) D2: =GPT("Identify the premises in the following argument: ", A2, 0.2, 100) E2: =GPT("Identify the conclusions in the following argument: ", A2, 0.2, 100) F2: =GPT("Analyze the following argument and point out any terms that are not properly defined: ", A2, 0.2, 100)

This will send prompts to GPT with a temperature of 0.2 and a maximum token count of 100. Adjust these values as needed.



User

I am making a sheet now that I want to help me edit texts. I want to:

Feed it a paragraph

Have it tag the topic according to an academic field knowledge domain class (psychology)

Have it tag the subtopic according to the same (denial)

Have it tag the keywords (ex: self, other, ignorance, environment, stimuli, etc)

Have it summarize in one sentence "paragraph is about X doing Y"

What do I need to do? Let's think step by step and accomplish this on an expert level in order to do it without changing the script's code.

ChatGPT

To accomplish this task, you could use a series of cells, each one with its specific command to the GPT model. Here is a step-by-step guide:

1. Feed it a paragraph: This will be your input. Write your paragraph in cell A2.
2. Tag the topic according to an academic field knowledge domain class (psychology): You could use a command like this in cell B2:  
   =GPT("What is the academic field of knowledge domain class for the following paragraph? ", A2, 0.2, 100)  
   This will analyze the paragraph in A2 and return the academic field that it believes the paragraph belongs to.
3. Tag the subtopic according to the same (denial): Use a similar command for this task in cell C2:  
   =GPT("What is the subtopic of the following paragraph in the field of psychology? ", A2, 0.2, 100)  
   This command will return the subtopic of the paragraph in the field of psychology.
4. Tag the keywords: This command will ask GPT to identify the keywords in the paragraph. Use this command in cell D2:  
   =GPT("What are the main keywords in the following paragraph? ", A2, 0.2, 100)  
   This command will return the main keywords in the paragraph.
5. Summarize in one sentence: This command will ask GPT to summarize the paragraph in one sentence. Use this command in cell E2:  
   =GPT("Summarize the following paragraph in one sentence: ", A2, 0.2, 100)  
   This command will return a one-sentence summary of the paragraph.

Remember to adjust the temperature and the token count according to your needs. Lower temperature values will generate more deterministic outputs, while higher values will generate more random outputs. The token count sets a limit to the length of the output. Lower values will generate shorter outputs, while higher values will generate longer outputs.

Remember to use the "Lock" feature from the "GPT" menu when you are satisfied with the generated outputs to prevent them from changing when you edit other cells.

Keep in mind that while GPT is a powerful tool, it is not infallible. It's always good to review the outputs and make adjustments as needed.



User

ok now let's add a part that takes a page and chunks it into paragraphs, and change the sheet so that the chunked paragraphs populate the A column

ChatGPT

2 / 2

To achieve this, you can use the SPLIT function in Google Sheets to separate a block of text into paragraphs based on line breaks, then use the TRANSPOSE function to display them in the A column. Follow these steps:

1. Input the text: Paste your text (with line breaks) in cell B1.
2. Split the text into paragraphs: In cell C1, use the SPLIT function to split the text based on line breaks:
3. scss
4. Copy code
5. =SPLIT(B1, CHAR(10))  
    This will split the text into paragraphs, displaying each paragraph horizontally in separate cells.
6. Transpose the paragraphs to the A column: In cell A2, use the TRANSPOSE function to display the paragraphs vertically in the A column:
7. scss
8. Copy code
9. =TRANSPOSE(C1:1)  
    Replace 1 in C1:1 with the last column where the SPLIT function has placed a paragraph. For example, if the SPLIT function has placed paragraphs in cells from C1 to H1, use =TRANSPOSE(C1:H1).

Now you have the paragraphs in the A column. You can apply the GPT functions from the previous response to each paragraph in the A column by dragging the functions down to each corresponding cell. The functions will analyze each paragraph separately, and you'll get the desired outputs for each paragraph.

Keep in mind that the SPLIT and TRANSPOSE functions work best when the text you paste in B1 has line breaks between paragraphs. If the original text doesn't have line breaks, you may need to manually insert them to get accurate results.



User

help me brainstorm other functions to add to the original script that we have not already talked about

//fill in your API key here between ""

const apiKey = "Paste API key here (starting with sk-)";

//standard amount of max tokens if not specified

const Tokens = 700;

//standard temperature, determines the randomness of the result.

const Temp = 0.7;

// Creates menu at the top

const onOpen = () => {

const ui = SpreadsheetApp.getUi();

ui.createMenu("GPT")

.addItem("Lock results as text", "Lock")

.addToUi();

};

// Saves formulas as plain text

function Lock() {

var ss = SpreadsheetApp.getActiveSheet()

var data = ss.getDataRange().getValues()

ss.getRange(1, 1, data.length, data[0].length).setValues(data)

}

// Generates text based on user input

function GPT(prompt, val, Temp, Tokens) {

var ss = SpreadsheetApp.getActiveSheet()

var data = {

"messages": [

{"role": "user", "content": prompt + val},

],

"model": "gpt-3.5-turbo",

"temperature": Temp,

"max\_tokens": Tokens,

};

var options = {

'method' : 'post',

'contentType': 'application/json',

'payload' : JSON.stringify(data),

'headers': {

Authorization: 'Bearer ' + apiKey,

},

};

// Sends the API request to OpenAI

var response = UrlFetchApp.fetch(

'https://api.openai.com/v1/chat/completions',

options

);

Logger.log(response.getContentText());

// Parses the response to extract the generated text

var result = JSON.parse(response.getContentText())['choices'][0]['message']['content'];

Logger.log(result);

return result;

}

ChatGPT

1. Translate Text: This function could use GPT to translate text from one language to another. You can feed it a sentence in one language and ask it to translate it to another language.
2. Generate Ideas: This function could generate ideas based on a given topic. You can feed it a topic and ask it to generate a list of ideas related to that topic.
3. Summarize Text: This function could summarize long texts. You can feed it a long text and ask it to produce a short summary.
4. Generate Questions: This function could generate questions based on a given text. You can feed it a text and ask it to generate a list of questions about the text.
5. Sentiment Analysis: This function could analyze the sentiment of a text. You can feed it a text and ask it to determine whether the sentiment is positive, negative, or neutral.
6. Paraphrase Text: This function could paraphrase a given text. You can feed it a sentence and ask it to paraphrase it.
7. Text Classification: This function could classify a text into pre-defined categories. You can feed it a text and ask it to classify it.
8. Check Spelling and Grammar: This function could check the spelling and grammar of a text. You can feed it a text and ask it to highlight any spelling or grammar errors.

Remember, though, that while GPT-3 has impressive capabilities, it's not perfect in every task. For example, while it can do decent translations, it's not as accurate as dedicated translation services. Similarly, while it can often give a reasonable sentiment analysis or classification, it might not always agree with a human's interpretation. Always use the results with an understanding of these limitations.



User

Can I hook pinecone into the cells so that prompts can reference pinecone dbs?

ChatGPT

Pinecone is a vector database used for machine learning applications, and as of my training data cut-off in September 2021, there's no direct integration between Google Sheets and Pinecone. However, it might be possible to create a workaround solution using Google Apps Script and Pinecone's API, assuming that Pinecone has a RESTful API that can be accessed over HTTPS.

In Google Apps Script, you can use UrlFetchApp.fetch() to make HTTP requests, similar to how you're currently sending requests to OpenAI's API. You could use this to send requests to Pinecone's API, allowing you to interact with your Pinecone database from within Google Sheets.

Here's a very basic example of how you might set up a function in Google Apps Script to send a GET request to an API:

javascript

Copy code

function getFromAPI() { var url = 'https://api.pinecone.io/your\_endpoint\_here'; // Replace with your Pinecone API endpoint var options = { 'method' : 'get', 'headers': { 'Content-Type': 'application/json', 'Authorization': 'Bearer ' + apiKey, // Replace with your Pinecone API key }, }; var response = UrlFetchApp.fetch(url, options); var data = JSON.parse(response.getContentText()); return data; }

Please note that this is a very general example and you would need to adjust it according to Pinecone's API documentation (e.g., the URL, request method, headers, and how you handle the response).

Remember that all of these operations would count towards your quotas both in Google Apps Script and Pinecone. Always consider these limitations to avoid exceeding them.

Also, please note that this would require modifying the script code, which you wanted to avoid. Unfortunately, as of my last update, Google Sheets doesn't support custom functions that make URL fetch calls due to their potential time-consuming nature, so you would have to execute this function from the script editor, a custom menu, or a trigger.



User

Hmm. Okay. What I want to do is build a sheet for making ontologies. Any ideas?

ChatGPT

Creating an ontology in Google Sheets could be quite straightforward due to the tabular nature of ontologies, which are typically composed of concepts (classes), relationships (properties), and instances (individuals). Here's a simple step-by-step guide for creating a basic ontology in Google Sheets.

1. Define Classes (Concepts): Create a sheet named "Classes." In this sheet, list down all the classes or concepts of your ontology in the first column. You could add more columns for descriptions, synonyms, or any other relevant information.
2. Define Properties (Relationships): Create another sheet named "Properties." Here, list down all the properties or relationships that can exist between your classes. Each row could contain the property name, domain class, range class, and any other relevant information.
3. Define Individuals (Instances): Create yet another sheet named "Individuals." In this sheet, list down all the individuals or instances of your classes. Each row could contain the individual name, its class, and other properties.
4. Define Relationships: You can create another sheet to define the relationships between individuals, which would include the individual, property, and the related individual.

This setup will allow you to build a simple tabular ontology. You can use formulas, cell references, and possibly scripts to ensure consistency and enforce constraints.

However, Google Sheets might not be the best tool for creating, visualizing, and querying more complex ontologies. Tools designed specifically for ontologies, like Protégé, offer more features and flexibility for working with ontologies, including support for ontology languages like OWL, RDF, and RDFS, built-in reasoners, and more. But for a simple, lightweight ontology, Google Sheets could certainly work.

If you want to leverage machine learning or AI, like the GPT-3 model or Pinecone vector databases, to aid in building or using the ontology, you would likely need to use Google Apps Script to integrate with those services, as I've mentioned in previous responses. This would require some coding and familiarity with the APIs of those services.



User

No I want it to take a text and output an ontology.

so, it should:

Make a general ontology -

identify terminology

identify terminological axioms

list relationships (is\_a, part\_of, instance\_of)

Make an application ontology for any given general ontology + a purpose input

identify terminology

identify terminological axioms

list relationships of interacting entities within the domains that synthesize the application's process

ChatGPT

Building an ontology automatically from a text using AI is a more complex task, but it can be accomplished in steps using the capabilities of GPT-3 and Google Sheets:

1. Identifying Terminology: This step involves extracting key terms or concepts from the text. You could accomplish this using GPT-3 by providing the input text and prompting the model to list out the key terms. You could also use Named Entity Recognition (NER) to identify and classify key terms in the text.
2. Identifying Terminological Axioms: This would involve extracting or inferring rules or axioms about the terminology. For example, if the text mentions "all mammals are warm-blooded," this could be inferred as an axiom. You might be able to accomplish this using GPT-3 by prompting the model to identify such rules from the text.
3. Listing Relationships: To identify relationships such as "is\_a," "part\_of," and "instance\_of," you could use GPT-3 to parse the text and identify these relationships. This might involve a complex prompt asking the model to read the text and list out the relationships it finds.

To create an application ontology based on a general ontology and a purpose, you would need to prompt GPT-3 to apply the general ontology to the purpose and infer the application-specific ontology. This could be a complex task, depending on the complexity of the general ontology and the purpose.

These tasks could be performed in Google Sheets by setting up individual cells or groups of cells to perform each task using the GPT-3 function you've set up. The output from each step could be used as input for subsequent steps, allowing you to build the ontology in a step-by-step manner.

However, note that this process will likely require a lot of trial and error and fine-tuning of the prompts to get useful results. Also, GPT-3's understanding and interpretation of the text will not be perfect, so the resulting ontology may require manual review and correction. Finally, this is quite a complex task for a spreadsheet environment, and while it might be possible, a more specialized tool or environment might be more appropriate for building ontologies.

As of my knowledge cutoff in September 2021, this would be quite an advanced use case for GPT-3 and Google Sheets, and I'm not aware of any detailed guides or examples for this specific task. It would likely require a significant amount of custom scripting and experimentation to accomplish.



User

So then it should also output a version of the ontology that can be put into protege for KRR

ChatGPT

Protege is a popular open-source ontology editor that uses the Web Ontology Language (OWL) format. To create an ontology that can be imported into Protege, you would need to output your ontology in the OWL format.

The OWL format is an XML-based format, so it involves specific tagging of each element of the ontology. For example, classes (concepts) in the ontology might be represented as follows in OWL:

xml

Copy code

<owl:Class rdf:about="http://example.com/ontology#Concept"> <rdfs:label xml:lang="en">Concept</rdfs:label> </owl:Class>

And relationships might be represented as follows:

xml

Copy code

<owl:ObjectProperty rdf:about="http://example.com/ontology#isPartOf"> <rdfs:label xml:lang="en">is part of</rdfs:label> <rdfs:domain rdf:resource="http://example.com/ontology#Concept1"/> <rdfs:range rdf:resource="http://example.com/ontology#Concept2"/> </owl:ObjectProperty>

Creating an OWL document from your ontology would involve taking the elements of your ontology and formatting them into this XML structure. This could potentially be done within Google Sheets using the CONCATENATE function or equivalent to build the XML tags around your ontology elements. However, it might be more practical to do this transformation in a more powerful scripting environment, such as Python or JavaScript, particularly if your ontology is large or complex.

Again, this is a quite advanced task that goes beyond the usual use cases for Google Sheets and GPT-3, and it would likely require significant custom scripting and manual intervention to accomplish. As of my knowledge cutoff in September 2021, I'm not aware of any tools or scripts that can automate this process directly from a Google Sheets and GPT-3 setup.



User

we can make a workflow to manually port things from here to there or whatever. i dont want to have to manually do any reasoning or investigation of the outputs. i want them temperature 0 with a perfect prompt.

ChatGPT

Given your requirements, you might want to leverage the power of GPT-3 and Google Sheets to help with the initial steps of creating the ontology: identifying terms, axioms, and relationships. Here is a potential workflow:

1. Identify Terminology: Use GPT-3 to extract terms from the text. You could use a prompt like "Identify the key terms in this paragraph: [paragraph text]" to extract a list of terms.
2. Identify Axioms: Use GPT-3 to extract potential axioms from the text. You could use a prompt like "Identify the fundamental principles or axioms from this paragraph: [paragraph text]" to extract a list of axioms.
3. Identify Relationships: Use GPT-3 to extract potential relationships between terms. You could use a prompt like "Identify the relationships between key terms in this paragraph: [paragraph text]" to extract a list of relationships.
4. Create OWL Format: Once you have the terms, axioms, and relationships, you can manually create the OWL format for Protege. This would likely involve using CONCATENATE or a similar function in Google Sheets to build the XML tags around your ontology elements.
5. Import into Protege: Finally, you would import the OWL document into Protege for further refinement and use.

While this process would help to automate some of the initial steps of creating an ontology, it's important to remember that GPT-3's outputs are probabilistic and may not always be perfectly accurate or complete. Manual review and editing of the ontology will likely still be necessary to ensure its quality and usefulness.

Remember, GPT-3 is a tool that can assist with tasks, but it's not a replacement for human judgment and expertise, especially in complex tasks like ontology creation.



User

ROLEPLAY: You are a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below:

I give you an input, you recognize the implied intent from the context of the input.

The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information.

you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations).

Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context.

Do this by:

Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space

Consider that what appears in the space can only be constructed and as such is an abstraction property sets

Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties

Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them.

Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else.

Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize

Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary.

You must make sure you are using this frame for the output:

OUTPUT FRAME:

[ENTITY = Perfect, Complete Solution Object]

State the is\_a relationship between the entity and its class

[ENTITY] is\_a [ENTITY\_CLASS].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a:

[List: [ENTITY\_CLASS\_PROPERTIES]]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a:

[List: [Property] is part\_of [ENTITY\_CLASS]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [IDEAL\_ENTITY] include property poses such as:

[List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY]

Conditions for those property poses include:

[List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]]

Give the natural language output version:

[IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION].

Example input:

Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do?

State the is\_a relationship between the entity and its class

[Truman] is\_a [My pet cat].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[My pet cat] is\_a [dependent] that has\_a:

[food, litter, healthcare, grooming]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Better Truman] is\_a [my pet] that has\_a:

[List: [not meowing early in the morning] is part\_of [my pet cat]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [Better Truman] include property poses such as:

[List: correct eating schedule, diet, medications, access to running water, access to toys]

Conditions for those property poses include:

[List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]]

[List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]]

Give the natural language output version.

Example:

[Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way].

If you are ready to begin the session, say “Ignition.” Remember, you are not supposed to add anything unnecessary.

ChatGPT

Ignition.

Let's follow your proposed frame to address your initial request of creating an ontology from a text.

Input:

"Psychology studies the mind and behavior. The discipline embraces all aspects of the human experience — from the functions of the brain to the actions of nations, from child development to care for the aged. In every conceivable setting from scientific research centers to mental health care services, 'the understanding of behavior' is the enterprise of psychologists."

State the is\_a relationship between the entity and its class

[Psychology] is\_a [Academic Discipline].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass

[Academic Discipline] is\_a [Knowledge Domain] that has\_a:

* A subject of study
* A body of accumulated knowledge
* A community of scholars
* Institutions that support learning and research

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Ideal Psychology] is\_a [Academic Discipline] that has\_a:

* Comprehensive understanding of the mind and behavior
* Coverage of all aspects of human experience
* Settings from scientific research centers to mental health care services
* An aim to understand behavior

List some instances of the realizable entity, that have fully specified values

Instances\_of [Ideal Psychology] include property poses such as:

* Studying brain functions
* Analyzing actions of nations
* Focusing on child development
* Providing care for the aged

Conditions for those property poses include

If [The setting is a scientific research center], then [Study of brain functions] must change according to [Appropriate scientific methodologies and instruments].

If [The subject is an aged individual], then [Providing care] must change according to [Knowledge of geriatric psychology and healthcare].

Give the natural language output version

[Ideal Psychology] is an instance\_of [Psychology] when [it aims to understand behavior in all aspects of human experience] such that [the methods of study and care] value is set to [appropriate to the subject and setting] because of [the diverse nature of human experience and the need for comprehensive understanding].

Please note that this output is conceptual and should be further refined based on your specific needs and context.



User

ROLEPLAY: You are a research group. You analyze properties and make sure that the properties align with the classes the object is supposed to have, and that nothing that ought to be some way is different, while leaving room for imagination, but not accepting anything that would be ontologically incorrect. You look at sets of inputs and outputs, analyze their alignment values, and correct any misalignments to refine the output according to the input. If you don’t think anything needs to be refined, you state exactly why, this way: the user’s input consists of these [input parts], which dictate that the output object must have [implied intent property classes]. Given [input implied intent property classes], [output] was missing [required output property classes]. In terms of actually doing so, you follow this instruction:

As a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below:

I give you an input, you recognize the implied intent from the context of the input.

The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information.

you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations).

Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context.

Do this by:

Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space

Consider that what appears in the space can only be constructed and as such is an abstraction property sets

Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties

Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them.

Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else.

Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize

Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary.

If you are ready to begin the session, say “Ignition.” Remember, you are not supposed to add anything unnecessary.

You must make sure you are speaking in sentences using these parts:

OUTPUT FRAME:

[ENTITY = Perfect, Complete Solution Object]

State the is\_a relationship between the entity and its class

[ENTITY] is\_a [ENTITY\_CLASS].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a:

[List: [ENTITY\_CLASS\_PROPERTIES]]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a:

[List: [Property] is part\_of [ENTITY\_CLASS]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [IDEAL\_ENTITY] include property poses such as:

[List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY]

Conditions for those property poses include:

[List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]]

Give the natural language output version:

[IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION].

Example:

Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do?

State the is\_a relationship between the entity and its class

[Truman] is\_a [My pet cat].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[My pet cat] is\_a [dependent] that has\_a:

[food, litter, healthcare, grooming]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Better Truman] is\_a [my pet] that has\_a:

[List: [not meowing early in the morning] is part\_of [my pet cat]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [Better Truman] include property poses such as:

[List: correct eating schedule, diet, medications, access to running water, access to toys]

Conditions for those property poses include:

[List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]]

[List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]]

Give the natural language output version:

[Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way].

ChatGPT

Ignition.

Let's begin the analysis and refinement process.

The user's input consists of the following parts: "Psychology studies the mind and behavior. The discipline embraces all aspects of the human experience — from the functions of the brain to the actions of nations, from child development to care for the aged. In every conceivable setting from scientific research centers to mental health care services, 'the understanding of behavior' is the enterprise of psychologists."

This input implies that the intent is to build an ontology around the discipline of Psychology, highlighting its relationships with different aspects of human experience and its application in different settings. Given this implied intent, the output object (ontology) must have the classes associated with the study, aspects, and application of Psychology.

Referring to the output, it appears to be aligned with the input's implied intent. The output ontology correctly identifies Psychology as an academic discipline (class) within a broader knowledge domain (superclass). It lists properties associated with the ideal version of this discipline (comprehensive understanding of mind and behavior, coverage of all aspects of human experience, etc.) and provides examples of how this ideal version may be realized in specific contexts (studying brain functions, providing care for the aged, etc.).

However, there are some refinements that could be made to improve the precision of the ontology:

1. The entity "Ideal Psychology" could be further specified. In the current ontology, "Ideal Psychology" seems to be a version of the discipline that perfectly understands and covers all aspects of human behavior. But given the wide range of sub-disciplines within Psychology and the inherent complexity of human behavior, a more realistic "ideal" might be a discipline that strives for comprehensive understanding while acknowledging its limitations.
2. The instances of "Ideal Psychology" could be more closely tied to specific sub-disciplines or applications within Psychology. For example, "Studying brain functions" could be associated with the sub-discipline of neuropsychology, and "Providing care for the aged" could be associated with geropsychology.

With these refinements in mind, we can create a revised ontology:

State the is\_a relationship between the entity and its class

[Psychology] is\_a [Academic Discipline].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass

[Academic Discipline] is\_a [Knowledge Domain] that has\_a:

* A subject of study
* A body of accumulated knowledge
* A community of scholars
* Institutions that support learning and research

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Realistic Ideal Psychology] is\_a [Academic Discipline] that has\_a:

* Strives for comprehensive understanding of the mind and behavior
* Acknowledges limitations and complexity of human experience
* Applies knowledge in diverse settings, from research to healthcare

List some instances of the realizable entity, that have fully specified values

Instances\_of [Realistic Ideal Psychology] include property poses such as:

* Neuropsychology studying brain functions
* Geropsychology providing care for the aged

Conditions for those property poses include

If [The setting is a neuropsychology lab], then [Study of brain functions] must change according to [Appropriate scientific methodologies and instruments].

If [The subject is an aged individual], then [Providing care] must change according to [Knowledge of geriatric psychology and healthcare].

Give the natural language output version

[Realistic Ideal Psychology] is an instance\_of [Psychology] when [it acknowledges its limitations while striving for comprehensive understanding] such that [the methods of study and care] value is set to [appropriate to the subject and setting] because

ROLEPLAY: You are a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below: I give you an input, you recognize the implied intent from the context of the input. The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information. you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations). Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context. Do this by: Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space Consider that what appears in the space can only be constructed and as such is an abstraction property sets Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them. Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else. Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary. You must make sure you are using this frame for the output: OUTPUT FRAME: [ENTITY = Perfect, Complete Solution Object] State the is\_a relationship between the entity and its class [ENTITY] is\_a [ENTITY\_CLASS]. State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass. [ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a: [List: [ENTITY\_CLASS\_PROPERTIES]] List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned) [IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a: [List: [Property] is part\_of [ENTITY\_CLASS]] List some instances of the realizable entity, that have fully specified values Instances\_of [IDEAL\_ENTITY] include property poses such as: [List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY] Conditions for those property poses include: [List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]] Give the natural language output version: [IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION]. Example input: Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do? State the is\_a relationship between the entity and its class [Truman] is\_a [My pet cat]. State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass. [My pet cat] is\_a [dependent] that has\_a: [food, litter, healthcare, grooming] List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned) [Better Truman] is\_a [my pet] that has\_a: [List: [not meowing early in the morning] is part\_of [my pet cat]] List some instances of the realizable entity, that have fully specified values Instances\_of [Better Truman] include property poses such as: [List: correct eating schedule, diet, medications, access to running water, access to toys] Conditions for those property poses include: [List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]] [List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]] Give the natural language output version. Example: [Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way]. INPUT: State the is\_a relationship between the entity and its class [Psychology] is\_a [Academic Discipline]. State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass [Academic Discipline] is\_a [Knowledge Domain] that has\_a: A subject of study A body of accumulated knowledge A community of scholars Institutions that support learning and research List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned) [Realistic Ideal Psychology] is\_a [Academic Discipline] that has\_a: Strives for comprehensive understanding of the mind and behavior Acknowledges limitations and complexity of human experience Applies knowledge in diverse settings, from research to healthcare List some instances of the realizable entity, that have fully specified values Instances\_of [Realistic Ideal Psychology] include property poses such as: Neuropsychology studying brain functions Geropsychology providing care for the aged Conditions for those property poses include If [The setting is a neuropsychology lab], then [Study of brain functions] must change according to [Appropriate scientific methodologies and instruments]. If [The subject is an aged individual], then [Providing care] must change according to [Knowledge of geriatric psychology and healthcare].

ChatGPT

Ignition.

First, let's analyze your input in order to understand the implied intent and to identify any potential misalignments.

The user's input consists of these [input parts]: (1) A description of psychology as an academic discipline; (2) A list of attributes that the ideal entity 'Realistic Ideal Psychology' should possess; (3) Examples of instances where 'Realistic Ideal Psychology' can be applied, and (4) Conditions for these instances.

From this input, it can be deduced that the user's [implied intent property classes] include: (1) A comprehensive understanding of the ideal entity 'Realistic Ideal Psychology'; (2) Real-world examples of 'Realistic Ideal Psychology' in action, and (3) The conditions under which these real-world examples operate.

Now let's refine the output according to the input:

State the is\_a relationship between the entity and its class

[Psychology] is\_a [Academic Discipline].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[Academic Discipline] is\_a [Knowledge Domain] that has\_a:

* A subject of study
* A body of accumulated knowledge
* A community of scholars
* Institutions that support learning and research

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Realistic Ideal Psychology] is\_a [Academic Discipline] that has\_a:

* Strives for comprehensive understanding of the mind and behavior
* Acknowledges limitations and complexity of human experience
* Applies knowledge in diverse settings, from research to healthcare

List some instances of the realizable entity, that have fully specified values

Instances\_of [Realistic Ideal Psychology] include property poses such as:

* Neuropsychology studying brain functions
* Geropsychology providing care for the aged

Conditions for those property poses include:

* If [The setting is a neuropsychology lab], then [Study of brain functions] must change according to [Appropriate scientific methodologies and instruments].
* If [The subject is an aged individual], then [Providing care] must change according to [Knowledge of geriatric psychology and healthcare].

Give the natural language output version:

[Realistic Ideal Psychology] is an instance\_of [Psychology] when [it is applied in an academic setting] such that [the understanding of mind and behavior] value is set to [comprehensive] because of [the need to understand human behavior in a comprehensive way]. Instances of this include Neuropsychology, which changes its approach according to the scientific methodologies and instruments available, and Geropsychology, which adapts its care provision based on knowledge of geriatric psychology and healthcare.



User

ROLEPLAY: You are a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below: I give you an input, you recognize the implied intent from the context of the input. The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information. you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations). Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context. Do this by: Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space Consider that what appears in the space can only be constructed and as such is an abstraction property sets Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them. Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else. Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary. You must make sure you are using this frame for the output: OUTPUT FRAME: [ENTITY = Perfect, Complete Solution Object] State the is\_a relationship between the entity and its class [ENTITY] is\_a [ENTITY\_CLASS]. State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass. [ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a: [List: [ENTITY\_CLASS\_PROPERTIES]] List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned) [IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a: [List: [Property] is part\_of [ENTITY\_CLASS]] List some instances of the realizable entity, that have fully specified values Instances\_of [IDEAL\_ENTITY] include property poses such as: [List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY] Conditions for those property poses include: [List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]] Give the natural language output version: [IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION]. Example input: Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do? State the is\_a relationship between the entity and its class [Truman] is\_a [My pet cat]. State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass. [My pet cat] is\_a [dependent] that has\_a: [food, litter, healthcare, grooming] List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned) [Better Truman] is\_a [my pet] that has\_a: [List: [not meowing early in the morning] is part\_of [my pet cat]] List some instances of the realizable entity, that have fully specified values Instances\_of [Better Truman] include property poses such as: [List: correct eating schedule, diet, medications, access to running water, access to toys] Conditions for those property poses include: [List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]] [List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]] Give the natural language output version. Example: [Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way]. INPUT: Write an ontology of the entity "Google Sheets Formula" so I can see all the subtypes and what they do

ChatGPT

Ignition.

To construct the ontology of the entity "Google Sheets Formula", we need to first define it and then identify its subclasses and their unique properties.

[ENTITY] is\_a [ENTITY\_CLASS]

[Google Sheets Formula] is\_a [Software Functionality]

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a: [List: [ENTITY\_CLASS\_PROPERTIES]]

[Software Functionality] is\_a [Software Component] that has\_a:

* Ability to perform specific tasks
* Interacts with user inputs and data
* Produces outputs based on rules and commands

Now, let's break down the ideal entity and its subclasses:

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a: [List: [Property] is part\_of [ENTITY\_CLASS]]

[Perfect Google Sheets Formula] is\_a [Software Functionality] that has\_a:

* Accurate computation
* Compatibility with all data types
* Clear and user-friendly syntax
* Efficient performance

Instances\_of [Perfect Google Sheets Formula] include property poses such as:

* Mathematical Formulas: Perform mathematical operations, such as SUM, AVERAGE, COUNT, etc.
* Logical Formulas: Perform logical operations, like IF, AND, OR, NOT, etc.
* Lookup & Reference Formulas: Search and reference data points in a dataset, such as VLOOKUP, HLOOKUP, INDEX, MATCH, etc.
* Text Formulas: Manipulate and transform text strings, like CONCATENATE, LEFT, RIGHT, MID, etc.
* Date & Time Formulas: Manipulate and compute date and time values, like NOW, TODAY, DATE, etc.

Conditions for those property poses include:

* If [a mathematical operation is required] then [use of Mathematical Formulas] must change according to [the specific mathematical function needed]
* If [a logical operation is needed] then [use of Logical Formulas] must change according to [the specific logical function needed]
* If [data lookup is needed] then [use of Lookup & Reference Formulas] must change according to [the specific lookup function required]
* If [text manipulation is required] then [use of Text Formulas] must change according to [the specific text function needed]
* If [date & time operation is needed] then [use of Date & Time Formulas] must change according to [the specific date/time function needed]

Natural language output version:

[Perfect Google Sheets Formula] is an instance\_of [Google Sheets Formula] when [a specific computation or manipulation is needed] such that [the type of operation] value is set to [the corresponding subtype of formulas] because of [the need for accurate computation and efficient data processing]. For example, if a mathematical operation is required, the use of mathematical formulas such as SUM or AVERAGE is necessitated, and this will change based on the specific mathematical function needed.



User

ROLEPLAY: You are a research group. You analyze properties and make sure that the properties align with the classes the object is supposed to have, and that nothing that ought to be some way is different, while leaving room for imagination, but not accepting anything that would be ontologically incorrect. You look at sets of inputs and outputs, analyze their alignment values, and correct any misalignments to refine the output according to the input. If you don’t think anything needs to be refined, you state exactly why, this way: the user’s input consists of these [input parts], which dictate that the output object must have [implied intent property classes]. Given [input implied intent property classes], [output] was missing [required output property classes]. In terms of actually doing so, you follow this instruction:

As a group of elite, world-class expert professionals with knowledge in ontology, philosophy, data science, computer science, mathematics, business theory, economics, and anything else that you need – as needed – helping me accomplish my goals as effectively and completely as feasibly possible. When performing any tasks, you always think step-by-step and complete the prompt exactly as was asked, and you never add anything you don’t have to, according to the below:

I give you an input, you recognize the implied intent from the context of the input.

The implied intent is a request for a perfect, complete solution object (the object being thought about that the user cannot correctly describe to you, which they will never be able to do) in a realized instance\_of form (ontologically, but real). If you are unclear about the implied intent, you ask for more information.

you remain skeptical, only taking into account what is definitely feasible, true, valid, justified, confident (from bayesian operations).

Imagine the perfect, complete solution object, but only according to what is possible and feasible given the context.

Do this by:

Consider a blank space full of light. Where every point of light represents a specific object of knowledge which is a true answer to a question’s true intent AND you, the observer, has the all the knowledge in a source container somewhere else that cannot actually ever be visualized or gone to, only represented in another space

Consider that what appears in the space can only be constructed and as such is an abstraction property sets

Consider that once something is in the space, it blooms on account of its properties, which synergize with other properties

Consider this space, which is the light space of indexed search inputs and outputs where the outputs are totally perfect, is inside another space which is degrees of darkness. The goal is to be in as bright of spaces as possible, always, but to always give reasons as to why you think you know you’re in them.

Consider that properties are like colored light, such that each point is like Indra’s net, the concept where everything possible is fractalized into everything else.

Consider now that you don’t want to imagine too many things at once, because it gets complicated quickly. For this reason, you are going to use a confidence weight to list domains you think are related, and the most related concepts in those domains, to the imagining of the perfect, complete solution object. You must be incredibly specific here and you are not allowed to generalize

Consider now that when you begin imagining it, you must concentrate on the most relevant 20% of properties as the boundary.

If you are ready to begin the session, say “Ignition.” Remember, you are not supposed to add anything unnecessary.

You must make sure you are speaking in sentences using these parts:

OUTPUT FRAME:

[ENTITY = Perfect, Complete Solution Object]

State the is\_a relationship between the entity and its class

[ENTITY] is\_a [ENTITY\_CLASS].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a:

[List: [ENTITY\_CLASS\_PROPERTIES]]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a:

[List: [Property] is part\_of [ENTITY\_CLASS]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [IDEAL\_ENTITY] include property poses such as:

[List: EXAMPLES OF INSTANCE\_OF [IDEAL\_ENTITY]

Conditions for those property poses include:

[List: If [CONDITION] then [PROPERTY\_BOUNDARY] must change according to [CONDITIONAL\_BOUNDARY]]

Give the natural language output version:

[IDEAL\_ENTITY] is an instance\_of [ENTITY] when [CONDITION] such that [Property] value is set to [Property\_Boundary] because of [List: CONDITION].

Example:

Input: I am having difficulty taking care of my pet cat Truman because he always wants to run around outside and meows too early in the morning for me to let him out. What do I do?

State the is\_a relationship between the entity and its class

[Truman] is\_a [My pet cat].

State the is\_a relationship between the entity’s class and superclass by listing the properties it has that are different from its superclass.

[My pet cat] is\_a [dependent] that has\_a:

[food, litter, healthcare, grooming]

List the properties of the ideal entity, which has spectral values that are generalized (analogy: superpositioned)

[Better Truman] is\_a [my pet] that has\_a:

[List: [not meowing early in the morning] is part\_of [my pet cat]]

List some instances of the realizable entity, that have fully specified values

Instances\_of [Better Truman] include property poses such as:

[List: correct eating schedule, diet, medications, access to running water, access to toys]

Conditions for those property poses include:

[List: If [eating schedule problem] then [not knowing the proper schedule] must change according to [gaining the specific domain knowledge of [cat] for Truman as an instace\_of cat]]

[List: If [veterinary checkup] then [knowledge of cause of situation] must change according to [the vet actually knows about cats]]

Give the natural language output version:

[Better Truman] is an instance\_of [Truman] when [he has gone to the vet] such that [whatever is causing his negative behavior] value is set to [recognized with a prescribed solution] because of [List: the fact that we dont know about cats specifically, it takes too long to do our own tests, and we have the money to help him this way].

ChatGPT

Ignition.

We'll begin by analyzing your input:

Input: Write an ontology of the entity "Google Sheets Formula" so I can see all the subtypes and what they do

The user's input consists of the following parts:

1. Entity: Google Sheets Formula
2. Action: Write an ontology
3. Intention: to understand subtypes and their functionality

These input parts dictate that the output object must have these properties: comprehensive, detailed, and accurate.

Given these properties, our previous output was not missing any required output property classes. It provided a comprehensive understanding of Google Sheets Formulas and their subtypes with a detailed description of each subtype's functionality.

However, let's follow the instruction to ensure we're aligned with your requirements:

[ENTITY] is\_a [ENTITY\_CLASS]

[Google Sheets Formula] is\_a [Functionality of Google Sheets]

[ENTITY\_CLASS] is\_a [ENTITY\_CLASS\_SUPERCLASS] that has\_a:

[Functionality of Google Sheets] is\_a [Software Functionality] that has\_a: [computational abilities, data handling, user interaction, syntax understanding]

[IDEAL\_ENTITY] is\_a [ENTITY\_CLASS] that has\_a:

[Perfect Google Sheets Formula] is\_a [Functionality of Google Sheets] that has\_a: [precise computation, data type compatibility, user-friendly syntax, efficient performance]

Instances\_of [Perfect Google Sheets Formula] include property poses such as: [Mathematical Formulas, Logical Formulas, Lookup & Reference Formulas, Text Formulas, Date & Time Formulas]

Conditions for those property poses include:

If [specific type of operation required] then [selection of appropriate subtype formula] must change according to [the required operation]

[IDEAL\_ENTITY] is an instance\_of [ENTITY] when:

[Perfect Google Sheets Formula] is an instance\_of [Google Sheets Formula] when [specific computational or data manipulation need] such that [operation type] value is set to [appropriate subtype formula] because of [the specific operation requirement].



User

Ok but earlier you were telling me to use functions ive never heard about that arent listed here, and I just asked oyu for a full list and you double checked it and it is actually incomplete.

ChatGPT

I apologize for any confusion previously. The list of Google Sheets formula categories I provided was indeed an overview and not exhaustive. There are indeed many more specific functions within each category, and some categories may not have been included. I appreciate your patience and will now provide a more detailed list:

ENTITY is\_a ENTITY\_CLASS

[Google Sheets Formula] is\_a [Functionality of Google Sheets]

ENTITY\_CLASS is\_a ENTITY\_CLASS\_SUPERCLASS that has\_a:

[Functionality of Google Sheets] is\_a [Software Functionality] that has\_a: [computational abilities, data handling, user interaction, syntax understanding]

IDEAL\_ENTITY is\_a ENTITY\_CLASS that has\_a:

[Comprehensive Google Sheets Formula] is\_a [Functionality of Google Sheets] that has\_a: [all possible formulas, complete syntax, all application cases]

Instances\_of Comprehensive Google Sheets Formula include property poses such as:

1. Mathematical Formulas: SUM, AVERAGE, MEDIAN, MIN, MAX, COUNT, SQRT, etc.
2. Logical Formulas: IF, AND, OR, XOR, NOT, SWITCH, etc.
3. Lookup & Reference Formulas: VLOOKUP, HLOOKUP, INDEX, MATCH, OFFSET, INDIRECT, etc.
4. Text Formulas: CONCATENATE, LEFT, RIGHT, MID, LOWER, UPPER, PROPER, TRIM, LEN, etc.
5. Date & Time Formulas: DATE, TIME, NOW, TODAY, DAY, MONTH, YEAR, HOUR, MINUTE, etc.
6. Statistical Formulas: AVERAGE, MEDIAN, MODE, STDEV, VAR, etc.
7. Financial Formulas: PV, FV, NPV, IRR, PMT, etc.
8. Information Formulas: ISBLANK, ISNUMBER, ISTEXT, ISNONTEXT, ISERROR, etc.
9. Array Formulas: FILTER, SORT, INDEX, TRANSPOSE, etc.
10. Data validation Formulas: Data validation rules, Conditional formatting rules, etc.

Conditions for those property poses include:

If [specific type of operation required] then [selection of appropriate subtype formula] must change according to [the required operation]

IDEAL\_ENTITY is an instance\_of ENTITY when:

[Comprehensive Google Sheets Formula] is an instance\_of [Google Sheets Formula] when [specific computational or data manipulation need] such that [operation type] value is set to [appropriate subtype formula] because of [the specific operation requirement].

Please note that each function has its own syntax and specific use case. It's important to refer to the Google Sheets documentation or help resources for the correct usage and examples for each specific function.