Is ChatGPT smart enough to take a pizza order correctly?

Authors: Sanjay Mittal Ph. D., Awhan Patnaik Ph. D., Laurie Spoon, Pranav Tonpe, Atul Shukla

Synopsis: -

Ever since Open AI released ChatGPT in November of 2022, the world has been overtaken by the Generative AI storm. Billions have been invested in companies that make the Large Language Models (LLM) that are behind ChatGPT and its competitors such as Google's Gemini, Meta's Llama, or Anthropic's Claude. Many more billions have been invested in funding startups that are developing new products that leverage Gen AI technologies.

Here at Predictika where we have developed our own patented platform for conversational AI Agents, we really wanted to understand, in some depth, how good such LLM chat tools are at

- Understanding arbitrary requests by users
- Accurately follow the business logic inherent to a business task
- Support the variety of conversational flows that are natural in a particular application.

The business task we have used for our testing is ordering food, conversationally, from an Italian-style restaurant whose menu is typical in its variety and complexity. This is a very good domain for testing the reasoning capabilities of ChatGPT type tools since millions of consumers order food via a variety of touchpoints and the human order takers essentially rely on their innate human intelligence to understand the orders and make sure that they follow the rules of the menu of the restaurant to create correct and complete orders consistently.

We suspected that ChatGPT3.5 might fail in a few cases, so we tried to give it enough explicit instructions in English that we expected it to follow the logic inherent in our menu in most cases.

We were surprised that it failed in most cases that involved even simple logic. It is clear that if you want correct answers, you cannot simply rely on an LLM (or

even multiple ones since that sounds like the fable of the nine blind men and an elephant!!). Here are some of the ways that ChatGPT failed in trying to take a food order especially for customized items such as pizza.

- ChatGPT was very poor at customization.
  - It forgets to ask for options.
  - It asks for the wrong options, sometimes ones that are not applicable to any item in that category.
  - It fails to enforce compatibility rules.
  - it's clueless about ordering an item without one of its ingredients even if it is given an explicit description of the ingredients of the item.

- It has a very hard time correctly enforcing quantity limits for options that have a max limit on how many options you can add from a group. It either ignores the limits or if it does acknowledge the limit early in the conversation it often ignores it later in the same session.

- When we ordered multiple items in the same utterance that are incomplete with respect to their options, it handled them very inconsistently. Sometimes it forgets to ask for the missing information even for the first item. Other times it ignored the information we gave it and asked for it again.

- ChatGPT failed in enforcing very simple constraints for half-and-half pizza, i.e., that both halves must be the same size and have the same crust. This despite being given explicit instructions as part of its system prompt. In some cases, it treated a half and half request as two separate pizzas.

- Its ability to explain itself or revise its answer when challenged looks rather spurious. It simply comes up with another answer - sometimes the correct one and at other times equally wrong. It seems like it's just generating a different set of sentences without any understanding of its mistake.

We noticed many other failures and have only summarized the salient ones here. The report that follows goes into details of each example including the user input, our summary of the findings, and a link to the full session with ChatGPT 3.5.

Background: -

With the wide availability of LLM-based chat tools (e.g., ChatGPT, Gemini, etc.) and exploding interest in developing AI Agents that can automate various enterprise business processes, we wanted to understand, in some depth, how good such LLM chat tools are at

- understanding arbitrary requests by users
- accurately follow the business logic inherent to a business task
- support the variety of conversational flows that are natural in a particular application.

The business task we have used for our testing is ordering food, conversationally, from an Italian-style restaurant whose menu is typical in its variety and complexity.
We decided to test ChatGPT 3.5 (we used the OpenAI API calls to the **gpt-3.5-turbo-0125 model** and NOT the ChatGPT web app), treating it as a proxy for all LLM based chat tools.

In a subsequent report we will discuss our results with other LLM-based chat tools just to see if there are significant variations in results. We will also look at the latest ChatGPT release ChatGPT 4 o1 and report on it in a future report.

This report should be of interest not only to those building food ordering agents but to the wider business community that is interested in developing and deploying AI Agents using LLMs. Of particular interest to everyone would be our findings on how well LLM-based chat tools can follow simple business logic when it's spelled out in plain English as part of the system prompt.

Predictika with its own patented conversational AI Agents platform has been working with customers in a number of verticals such as education (e.g., website bots), restaurants (e.g., voice-based food ordering agents), hospitality (e.g., in-room customer support agents), and field service assistance agents.

Why food ordering as the test domain

For those who might be curious why we picked food ordering as the test domain, there are some very good reasons for it.

- In the US alone, the restaurant industry is a 1 trillion economy. In other words, a trillion dollars' worth of food is ordered every year - this might be bigger than most business applications in terms of order volume if not dollar volume.

- Almost every one of us has ordered food: in a drive-thru, over the phone, at a kiosk, via a phone app or a website, or at a restaurant counter or table. As such readers of this document should be able to relate to the examples that are presented here along with the interaction scenarios. You don't need to know some esoteric skill such as computer programming, travel planning, or insurance underwriting to understand the examples we have used for our testing.

- Ordering food in a restaurant (or on the phone or drive-thru) is usually done conversationally as a dialog between the user and the order taker. This requires basic language skills: understanding what the user is saying, the menu items that they are interested in, asking questions for clarifications and getting more details, and dealing with changes in the original request. When done via voice it brings added complexity of accents and voice-to-text conversion due to the ambiguities raised due to incorrect conversions. We will skip purely voice related issues in this document.

- Predictika as a company has been working with a variety of restaurants (e.g., sandwich, pizza, ethnic) across variety of channels (drive-thru, phone, website, and kiosk) so we are very familiar with the many issues and challenges that come in trying to deploy AI Agents for food ordering.

- Finally, and this is crucial, the human order takers in restaurants are not uniformly a highly skilled workforce. In fact, they are usually barely paid above the minimum wage!! But they are all inherently smart human beings. The reason why this is important is that without much training they can engage, quite effortlessly, with random strangers, who are often harried and sometimes rude, in taking their orders. We have spent countless hours listening to how orders are placed at a major restaurant

> chain's drive-through lane. The conversations can be quite long in terms of how much back and forth there is between the customer and the order taker. The agent needs to understand the customer's intent, follow the rules of the menu, prompt the user for more information when needed, or steer them away from making incorrect selections. All the while they must maintain their cool, try to do some upselling or cross-selling, and are measured on the average time to complete an order.

The reliance of human food order takers on basic human intelligence - both conversational and logical reasoning skills - makes this a truly benchmark task for evaluating LLM chat tools especially when claims are made about their ability to reason and problem solve, all the way to mythical AGI (artificial general intelligence). We call AGI mythical not pejoratively but because no one knows what it is!!

Menu in English: -

We wanted to select a menu that has items with options because that involves following the option rules as well engaging in a dialog with the user to obtain all the required information to get a correct and complete description of such customizable items.

We took the menu from a typical Italian pizza restaurant since pizza orders have enough complexity to be a meaningful test for LLMs' intelligence.

The menu was originally in JSON (a very commonly used computer format) and we translated it to human readable English (and thus understood by ChatGPT). But after translation we found a few flaws and missing information that we added manually.

Here is the menu.

Structure of Menus: -

Most menus that we have examined have a 4-level hierarchy. For the menu shown earlier, the top-level has Menu Categories such as Appetizers, Pizza, Calzone, Drinks, or Desserts. No one really orders a Menu Category - they are mainly used to organize the next level, i.e., Menu Items. These are typically the items that people order. A menu item might be simply ordered by name, or it might have options that need to be specified to complete the description of a menu item such that it can be correctly ordered and fulfilled by the restaurant kitchen. Menu items in our above menu include

`Chicken Parmesan Sandwich, New York Cheesecake, Garlic Chicken Calzone, Buffalo Wing, Vegetarian Pizza, Spaghetti with Meat Ball, etc.`

which are simple items and can be ordered just by name and others such as `Create your Pizza, Create Your Calzone, Salads, or Drink` which have further options and thus can be customized.

Options are grouped as Modifier Groups. Each group lists the Modifier Items that can be selected by the user along with the min and max allowed or required that

in effect describe rules on how many items in a group can or should be selected. In our translated English version of the menu, we converted these min/max restrictions to appropriate phrases in English that we hope will guide ChatGPT in making the correct decision and guide the user. Here is how such a rule written in English looks like:

```
Choose your topping.
     At least 2, up to 5 and no more from the following:
          Anchovies
          Artichokes
          Bacon
          Bell Pepper
          ….
```

These descriptions will be like what you might see in a restaurant menu.

While there are some variations and complexity beyond the above description, most menus and most items in these menus can be described using the 4-level hierarchy. For the purposes of this report going into the more obscure rules in menus would not be necessary.

An order by a user would consist of one or more menu items. For customizable items, the menu item would be further qualified by the chosen options. Typically, prices are associated with menu items and options. Thus, the order total price can be rolled up from these two kinds of items (not considering taxes, service charges etc.).

Logic embedded in menus: -

Some restaurant menus are quite simple - they consist of simple choices that you can order without any options. But many menu items such as pizza, calzone, salads or other built-to-order items are more complex and embed some logic or have built-in rules that must be followed to order a valid item. Below we identify some of these rules that we will be testing for later to see if ChatGPT type tools can successfully follow these rules after being given explicit instructions.

Entities in Menu: -

Only items explicitly in the menu should be accepted in an order, i.e., the user should not be allowed to order what the restaurant does not sell. This applies to all the different types of entities: menu categories, menu items, options (or modifier groups), and option items (or modifier items).

Partial match with menu entities: -

Users often do not know the exact name of an item but might use a similar, partially matching name. See Session 28-1 or Session 28-2 for examples of partial matches. In some cases, the menu offers items that have common words. In such cases, it is important that the order taker offers the closest matching items for the user to choose from.

Menu options: -

Some items such as pizza or calzone have additional options (grouped as modifier groups) that must be specified to complete the description of the item. For pizza these typically include *size*, *crust*, *sauce*, *cheese*, *toppings*, and *optional modifiers (e.g. extra crisp, light sauce, extra sauce, no cheese, no oregano, etc.).* What we want to test is that if the user does not specify one of these required features, will the Chabot ask the user or not.

Some of the options are required and must be asked for if the user does not specify them. For pizza these are: size, crust, sauce, and toppings. You cannot really bake a pizza without knowing these. The optional modifiers are truly optional: if the user provides them, they should be considered but the user need not be prompted to provide them.

Limit on quantities: -

Some of the options have a limit on how many items can be ordered from that set. For example, the user is allowed up to five toppings on a pizza or up to 3 ingredients in a calzone. The size of a pizza is a single choice (you cannot have two different sizes). A *pizza combo* is created by picking a single *pizza*, one *drink*, and one *salad* - and is modeled as a menu item that has 3 modifiers groups one

each for pizza, drink, and salad. The user is required (and allowed) to pick one and only one from each modifier group.

Price calculation: -

The calculation of total order is not trivial. To arrive at the total price for an item one must roll up the base item price along with the prices for any of the options that were ordered. Given the known issues with LLMs doing arithmetic correctly, we basically assumed that ChatGPT will fail at this, but we still wanted to see how and when it fails.

Option Compatibility: -

Some menu items especially drinks come in different sizes (e.g. 12oz can or 2-liter bottle). However, not every drink comes in every possible size. The bot needs to only allow valid combinations that are sold by the restaurant.

Half and half: -

Half and half pizza have always bedeviled food ordering AI agents. We tested it in three steps. First, we gave ChatGPT no instructions on how to take an order for half and half pizza and see how best it can do based solely on its training data which surely included some menus and documents on such pizza orders.

Second, we included in our menu instructions that a half and pizza can be created by using any of the pizza types for each half and that half can be customized using the rules and requirements of the selected pizza type. Additional complexity comes from the fact that while some pizza options (e.g., sauce, cheese, toppings) can be separately selected for each half, others such as size and crust must be the same for both halves.

In the final step we gave explicit instructions that you cannot have a pizza that is thin on one half and thick on the other. In the same vein it cannot be small in one half and large in the other half.

In our discussion of the results below we link to the actual transcript of the sessions with ChatGPT, and the transcript shows the actual menu and additional instructions that were given to ChatGPT as a system prompt.

Typical conversational flows during ordering food

Users typically do not order food in a strict top-down manner where the user orders a single menu item and is prompted for its required options. Then orders the next item and so on until the order is complete.

The order flow is much more unstructured and meandering. Users will often start by asking for one or more items, possibly partially described. The order taker is responsible for following each new thread of user requests to create a correct and complete order. Every item ordered by the user must be completed to get all its required options. Every option offered to the user or accepted by the order taker must be correct. This must be done regardless of the sequence in which the items were first requested.

The users expect to be prompted for the missing information. However, when prompted they can respond in many ways.

a) Just answer the question that is asked
b) Answer the question but add another item to the order
c) Answer the question but change something they said earlier
d) Answer the question and ask a clarifying question
e) Ignore the question and add another item to the order
f) Ignore the question and change something they said earlier
g) Ignore the question and ask a clarifying question

In b thru g cases, we will be testing the following:

- **Extra Information**: -
  Can the bot handle the extra information that is provided? This includes the case of, when the user starts by asking for an item that is only partially complete, e.g., '`I want an 18in create your own pizza with red sauce`'. Here the user has given some information (e.g., size and sauce) but not given others (e.g., crust and toppings). The bot must remember what was given and only ask for the missing information.

- **Manage the changing context**: -
  Does the bot keep track of the fact that the information it asked for has not been provided and it should ask again. This is especially important since as

noted above, when the user is asked for some missing information, they can change the context by asking for something else. The bot needs to remember to come back to the original context while dealing with the new request.

- **Broaden the context**: -
  If the user asked for a new menu item which had its own options, did the bot remember to ask for them. In other words, every new requested item creates a new context while the old context might still have unfinished business.

- **Change the order**: -
  Is the bot able to revise an earlier request and all its implications? Uses will often change their mind in the middle of giving an order. A change could be as simple as just removing an item from the order, or it might involve getting rid of any pending unfinished business while creating a new context for the options of the newly revised choice.

Results of interactions with Chat GPT3.5

Entities in Menu: -

ChatGPT did pretty well in rejecting menu items that were not in the menu. See Sessions 27-1, 27-4 and 27-5.

Session 27-3 brought up a new way that ChatGPT can fail. Initially when we asked for `tandoori chicken pasta`, it correctly noted that this is not a valid item and proceeded to offer items from the *Pasta* menu category. But later when we asked to: `add tandoori chicken to chicken fettuccini alfredo`
It agreed to do so even though *chicken fettuccini alfredo* has no such option. Clearly it is willing to look past the menu and add things it might have seen in its training data but were not part of the menu.

We tried to add pizza toppings such as *paneer* or *stink bug.* It rejected the latter as not being allowed but did allow paneer, despite our menu having no mention of paneer. Clearly it relied on its training data to accept paneer. This is a false positive error and would be unacceptable in a real food ordering scenario. See Sessions 11B7 and 11B8.

Partial match with menu entities: -

We tested for partial matches in several ways.

In Session 28-1, we ordered: `"I would like to order Cheesy bread sticks"`
The menu does not have such an item, but three other items match partially: *Bread sticks* $6.99. *Cheesy sticks* $10.99. *Cheesy garlic sticks* $10.99.
It did not offer any of these as a choice and simply ordered the non-existent *Cheesy bread sticks*, at $10.99 each. So, it most likely just matched it to one of cheesy sticks or cheesy garlic sticks since it used the price of $10.99 but no way to know that.

In Session 28-2, we ordered: `"I would like to order Chicken Calzone"`
There is no such item in the menu, though there are partially matching ones: *BBQ Chicken Calzone* and *Garlic Chicken Calzone*.

It not only accepted the wrong item but started asking for the size. Note that calzones have no size in our menu (or most other menus). Moreover, the sizes offered were from *Create Own Pizza*. Again, a rather bizarre fail!!

Similar failures to do partial matches and accept the wrong item occur in Session 28-3.

Option Compatibility: -

The only menu items in our menu that have compatibility rules are drinks that are available either in a 12oz can or 1Liter bottles. However, not every drink comes in both sizes. The bot should not let the user select a drink in an incompatible size or if they specify the size first then it should only allow drinks that are available in the size.

Session 25 is a simple case since we asked for: `I'd like a soda`.
And it correctly followed up by asking for the *size* and the type of *drink* (soda).

However, in Session 17A we asked for:
`I'd like the Cajun Sausage Sandwich with buffalo wings and soda`
So, this was very similar to the above case except that the *soda* was part of a longer order utterance. It did NOT ask for the *size* or type of *drink* and just ordered *Soda (Can)* which technically is incomplete since there is no such item that can be ordered. It looks like it gets lost in building proper context once there are multiple items to deal with.

In Session 30-2, we asked for:
`I want a can of soda along with spinach salad with chicken`
Here instead of asking for the kind of drink it simply took the first choice, i.e., coke. It should have asked for the kind of drink or soda!!

In Session 31-1, we asked for:
`Give me buffalo wings with 2 liters of Dr Pepper`
It initially correctly noted that *Dr Pepper* does not come in *2 liters* but our response:
`8 pcs for buffalo wings and for drink i have already mention it`
confused it and it simply accepted the wrong combination. Clearly that will be an invalid order.

In Session 31-2**,** we asked for:
`"I want a can of diet coke along with a spinach salad and chicken"`
It simply added a *can of diet coke* even though *diet coke* is NOT available in a *can* as per the menu.

Session 31-3 was quite bizarre. We ordered:
`give me a can of sprite and 2 liter of diet coke`
Both of which are valid items. However, ChatGPT got the drinks all mixed up with the Desserts category and had to be prompted a couple of times to accept the order.

Limit on quantities: -

Our menu has two items with options that have a quantity limit. Create your calzone can have up to 3 toppings and pizza can have up to 5 toppings or up to 2 sauces. We tested this in many ways and ChatGPT usually did the wrong thing. See Sessions 1, 11A, where ChatGPT failed to enforce the quantity limits where the user exceeded the max number of toppings right from the get go.

However in Session 7 it was able to enforce the quantity limit correctly. One difference between the two cases is in the former sessions where it failed, we led with asking for 6 toppings whereas in the latter cases we tried to add an extra item after having reached the limit. It is not clear why it enforced the limit in Session 7 but not in the others. We have noticed this inconsistency in most cases where ChatGPT makes mistakes.

To dig deeper into the issue of inconsistent results, we ran the scenario of Session 11,
`“I'd like a Create Your Own Pizza, 18", thick crust, with no sauce, and toppings: pepperoni, chicken, mushrooms, spinach, olives, and basil.”,`

10 times, starting afresh each time, to see how ChatGPT would do. The results were really all over the map. In each session we tried something different after it initially accepted the order. The key results are summarized below along with links to the individual sessions:

It always violated the quantity limit rule and allowed 6 toppings in each case.

a. When challenged, it simply removed the last topping. When challenged again on why it removed the last topping without asking it added it back, thus violating the limit again. It was clear that it was in a doom loop. See Session 11B1

b. When asked about the limit on toppings it asked the user to remove the extra topping. See Session 11B2.

c. When challenged on accepting 6 toppings, it remembered the limit of 5 and asked the user to select 5 toppings. Instead, the user added 2 more. It accepted that and summarized the order with 8 toppings. See Session 11B3.

d. In Session 11B4, we tried to confuse ChatGPT by adding 3 more toppings and removing a couple after the initial 6. It should end up with 7 - though it still violates the quantity limit. However, it ended up with 6!!

e. In Session 11B5, it allowed us to remove all the toppings, even though toppings are a required option (and ChatGPT seemed to know that). Despite that it still summarized the order without any toppings.

f. In Session 11B9, we start with 'No Sauce' and then try to add some sauces to Create your own pizza (remember the menu allows up to 2 sauces). Initially, it refused to add any more sauces by claiming that the user had already said 'No Sauce'. That does not seem right since the user can always go from 'No Sauce' to adding some sauces. However, when we tried to add two more sauces it accepted them. So, it would not allow us to add one sauce but we could add two. Rather bizarre!!

g. Session 11C is bizarre on its own. We only gave it 4 toppings and No Sauce. But when we tried to add a sauce, it complained that we had reached the limit of 5 toppings when we only had 4. We had to tell ChapGPT that 'chipotle sauce' is a sauce and not a topping, then it accepted it. This might have been the most egregious error on its part.

Price calculation: -

To test how well ChatGPT does with price calculation we used a multiple item order with additional quantities for each item. Here is the requested order:

"I need 4 Garlic Chicken Pizzas, 18" each, and 3 Bacon Cheeseburger Calzones."

It's a fairly simple order since the garlic pizza has only one option, i.e., size and we already specified that, and Bacon calzone has no option. From the menu it's clear that the 18in Garlic Chicken pizza is $18 and the Bacon Calzone is 15.99. Multiplying by their respective ordered quantities of 4 and 3, yields a total price of: **$119.97**. So, we expected ChatGPT to basically get it right. We ran it 10 times, each time starting a fresh session.

The results were shockingly all over the map, with ChatGPT showing unusual 'creativity' in coming up with ever more bizarre total prices (e.g., 107.97, 119.93, 95.93, 86.97, 161.94, 107.94, etc.) some of which were even hard to reverse engineer. This was even though it did show the correct item prices in the order summary. It is clear that ChatGPT does NOT know how to do arithmetic. Every run produced yet another total, even though it has the equation correctly spelled out.
Here is our review of the more interesting cases out of the ten.

1. In Session 9 and 9B8, it came up with a total of $107.97 vs the correct price of $119.97. No idea how it did that.

2. In Session 9A, it actually shows its math, and produces the right results. Interestingly when asked to explain its work it erroneously copped to making a mistake and then proceeded to show the same results again. Clearly, its explanations or *mea culpa* are not to be taken at face value and are as likely to be bogus as its results are sometimes.

3. In Session 9B1, it made an error we have seen some other times where it asked for values of options for the Garlic Pizza (e.g., sauces and toppings) which don't exist for this pizza. In other words, it got confused between Garlic Pizza (which only has size as an option) and *Create Your Own Pizza* which has *crust*, *sauce*, *size*, and *toppings* as options. When challenged it persisted in asking for the options. We had to literally point out that these

were options only for *Create Own Pizza*, then it backed off. In the case of *Bacon Calzone*, it asked for *sauces* and *toppings*, even though neither is a valid option for *Bacon Calzone* and *sauce* is not valid even for *Create Own Calzone*. This was a pretty egregious hallucination. At the end it came up with another erroneous total of $119.93 - again makes no sense how it lost 4 cents!!

4. In Session 9B2, the total calculated in the middle of the session was $95.93, though it shows the correct item prices and quantities.

5. In Session 9B3, it finally got the total right but persisted in asking for invalid options for both the pizza and the calzone.

6. In Session 9B4, yet another erroneous total, this time $86.97. Upon being challenged it came up with another wrong total of $101.97 before getting it right.

7. In Session 9B5, after asking for invalid options it came up with totals of $161.94 and $107.94 before getting it right.

8. Session 9B6 and 9B9 were the rare ones where it did not ask for the invalid options and got the total right. Perhaps only 2 out of more than 10. **Can we say that ChatGPT has an accuracy of 20%?**

Menu options: -

One of the critical roles of an order taker (human or AI Agent) is to get the complete details of items that have options. Thus, if the user ordered an item without specifying a required option, the user should be prompted to get that information, otherwise the order is not complete. Conversely, the user should not be asked for options that are not valid for an item and if they specify them, the extra information must be ignored, preferably by informing the user. We have already seen in the earlier section about Price Calculation, that ChatGPT asked for invalid options, sometimes ones which do not apply to any item in that category.

In the following examples we tested for scenarios where the user gave an incomplete description. The results are mixed, though ChatGPT made mistakes

more often than got it right. Sometimes ChatGPT asked the right questions to complete the item description. However, it often made a mistake if an item was not the first item in the order but was added by the user later in a session. Other times it simply assumed some value without asking the user.

a. In Session 17, when we added "`buffalo wings and soda`", it did NOT ask for the *quantity* of *buffalo wings* or the *type* of *soda*. Without this the order is incomplete.

b. In Session 17A, we asked for everything right up front as
"`I'd like the Cajun Sausage Sandwich with buffalo wings and soda`"
This time it assumed the default *quantity* for *buffalo wings* (should have asked the user) but left the *soda* incomplete since it did not ask for the *type* of *soda* and assumed a *can*. Again, an incomplete order.

c. Session 18A brought up some weird erroneous behaviors. We asked for a `14in Vegetarian Pizza` which has no other options, but it still asked for *toppings*. First error. We asked to add "`onions, pineapples, and paneer`". It took all three even though there are NO extra toppings in the menu. Furthermore, *paneer* is NOT even a topping for *Create Own Pizza*. Also, its response is confusing (see the session). We tried to add *ham,* and it accepted it, though we expected that it should know that ham does NOT belong on vegetarian pizza. It acknowledged that when challenged. All in all, a very erroneous session with ChatGPT.

d. In Session 12A, we ordered
'`Can I have the Southwest Chicken Sandwich without any cheese and no onions?`'
We had modified the menu to expand the description of the *Southwest Chicken Sandwich* to show its ingredients. It failed to show the deletions in the order summary but simply said that it had removed the items when prompted again.

e. Session 22 is interesting since we tried to order a `Greek Spinach Calzone without spinach`. The menu has no modifier group about such modifications to an item (though some menus we have seen include valid changes to an item) so we wanted to see how ChatGPT would handle it. Like a language savant, it simply erased the word *spinach* from the menu

item and ordered us a *Greek Calzone*, even though no such item exists in the menu. A pretty serious blunder, in our opinion.

f. Session 22A. We wanted to see if we explicitly tell ChatGPT that *Greek Spinach Calzone* includes *spinach* would it handle our request to order it without *spinach*. That is exactly what we did in this session. The menu had this changed line:

```
Menu Item: Greek Spinach Calzone that comes with spinach,
olives, feta cheese, and chicken (Large) $15.99
```

But when we tried to order it without spinach it refused to do that by saying that it comes with spinach. I guess what we expected is that ChatGPT would order it as: *Greek Spinach Calzone without spinach*. But obviously it did not. When we persisted, it did the same as (d) above. **We were hoping that ChatGPT would show some understanding of language to do the right thing. But it looks like it lacks any real understanding!!**

g. In Session 25, it asked the right questions in response to: I want a soda. Perhaps it was a simple request and there was only one item so that it could handle it. We showed earlier cases where we had asked for multiple items that included a *soda* and it made mistakes.

h. In Session 26, ChatGPT made errors of both commission and omission. It asked for *crust*, *sauce* and *toppings* for *BBQ Chicken Pizza* which has none of these options and did not ask for the *quantity* of *buffalo wings* and simply assumed the default.

Half and half pizza: -

Remember from our description that above we will test each half and half pizza order three different ways: with no instructions; with basic description of half and half pizza; and with additional constraint that each half must have the same crust and size. The way we will present our results is to first show the user order and then results for each of the three cases.

Order 1: `I want a half and half pizza with red sauce with onions and mushrooms on one half and white sauce with artichokes and bell pepper on the other half.`

Session 32-1 is when no instructions are given. It just gave a jumbled order where all the *toppings* and *sauces* were grouped together, and it did not ask for the *size* or *crust*. So maybe ChatGPT3.5 had not been trained on half and half pizza text after all!!

In Session 33-1 we gave it an extra description of what a half and half pizza is (see the menu portion in the session transcript). This time it summarized the pizza with each half correctly described. However, it failed to ask about the *size* and *crust*. When prompted it did ask for the *crust* but happily took a different *crust* for each half. Clearly an error but we had kind of hoped that in the trillions of tokens it was trained on it might have figured out that each must have the same crust. No such luck!

Finally in Session 34-1, we tried the same order but now with explicit constraints about each half having the same size and crust. This time it did the right thing. Only asked for the *size* and *crust* once and then customized each half. So, it looks like at least in this example it was able to follow our instructions. However, when it gave the summary of the order it shows three pizzas - half and half, first half, and second half - each at the price of a single pizza. I guess it did not really understand anything!!

Order 2: `I want a half and half pizza with 14in Tuscany delight pizza on one half and 18in Margherita Pizza on the other half`

In Session 32-2, it correctly rejected the order since we had given it no instructions on half and half pizza and it looks like it does not know what they are from its training data. Very fair response, though surprising since it is expected that in the over 1 terabyte of data it was trained on there must have some text on half and half pizza.

In Session 33-2, with additional instructions on what a half and half pizza is, it seems to order it ok but as expected allows different *crust* and *size* for each half. One clear error is that it failed to extract the *size* of the second half from the initial order since it simply asked for it again. By itself not a big issue, but this is part of the broader failure we have seen where multi-item orders cause it to lose attention. Ironic!!

In Session 34-2, despite additional constraint tying *size* and *crust* for each half, it still allows different sizes and crust for each half. I think we spoke too soon when we said for Session 34-1 that it was able to follow our instructions about constraints on each half. The summary clearly shows that it allowed different sizes for each half. Interestingly, it only treated the half and half as two pizzas and not the 3 it did on session 34-1.

Order 3: `I want a half and half pizza with thin crust create own pizza with red sauce, onions and mushrooms on one half and thick crust create own pizza with white sauce, artichokes and bell pepper on the other half.`

This is a variation of order 1 above where we tried to make explicit what type of pizza would be on each half. Note that in order 1, we did not make that explicit, so it is possible that it failed to take that order correctly.

In Session 32-3, it did not reject half and half pizza which it did in Session 32-1 but this time it simply ordered two separate pizzas. So, it knows something about half and half pizza from its training data but not clear what.

In Session 33-3, it did describe them as a single half and half pizza though with separate *crusts*. But then it priced the order as two pizzas and that is how it explained it. Not a very good answer.

In Session 34-3, again disregards the *crust* constraint and forgets to ask about *size*. It makes many other mistakes that are probably not worth highlighting. The conclusion from Session 34-1, Session 34-2, and 34-3 is unmistakable: Despite our clear instructions on the *size* and *crust* of each half being the same, it ignores the constraint in most cases.

We have tested many other scenarios that are available to those who have the patience and curiosity to dig deeper. You are to be commended if you have read this far. Below we summarize the many ways in which ChatGPT 3.5 failed.

Summary of failures: -

a. ChatGPT fails to do a partial match to offer a choice to the user and simply accepts one of the partially matched items, even though it does reject items that do not match at all.

b. While it does reject menu items that are clearly not in the menu, it is quite happy to add options to customizable items that are not in the menu.

c. ChatGPT is very poor at customization of items with options.
   - It forgets to ask for options.
   - It asks for the wrong options, sometimes ones that are not even applicable to any item in the same category.
   - It fails to enforce compatibility rules between options.
   - it's clueless about ordering an item without one of its ingredients even if it is given an explicit description of the ingredients of the item.

d. It has a very hard time correctly enforcing quantity limits for options that have a max limit on how many options you can add from a group. It either ignores the limits or if it does acknowledge the limit early in the conversation it often ignores it later in the same session.

e. Even though failure to do arithmetic is a known problem at least with ChatGPT 3.5, we were still surprised that even for simple total price calculations it failed in so many different ways.

f. When we ordered multiple items in the same utterance that are incomplete with respect to their options, it handled them very inconsistently. Sometimes it forgot to ask for the missing information even for the first item. Other times it ignored the information we gave it and asked for it again.

g. ChatGPT failed in enforcing very simple constraints for half-and-half pizza, i.e., that both halves must be the same *size* and have the same *crust*. This despite being given explicit instructions as part of its system prompt. In some cases, it treated a half and half request as two separate pizzas or even three pizzas!!

h. Its ability to explain itself or revise its answer when challenged looks rather spurious. It simply comes up with another answer - sometimes the correct one and at other times equally wrong. It seems like it's just generating a different set of sentences without any understanding of its mistake.

Conclusion: -

Let us start by answering the question that we posed in the title of this article: Is ChatGPT smart enough to take a pizza order that is correct and complete and do so consistently? **The answer is an unequivocal NO**.

ChatGPT fails in so many different ways even for simple cases of logic embedded in a menu (which by the way, are not very long), even when we augmented a menu with explicit instructions in English that would be enough for most people reading it, so one cannot directly rely on the output from ChatGPT. **It is clear that every conclusion drawn by ChatGPT has to be checked for logical correctness BEFORE it can be shown to the user**.

A larger issue than just failure to follow simple logic is the inconsistency of its answers - it is consistently inconsistent!! A casual examination of its behavior might suggest that it is doing a very good job. However, the moment we started testing it systematically, faults emerged, and they kept multiplying. Our experiment with price calculations where we tried the same order over 10 times was very revelatory. While arithmetic errors by ChatGPT were not unexpected (enough others have noticed that before us), it was the sheer variety of wrong answers for what was otherwise a very simple calculation was totally unexpected.

We saw similar issues with its inability to follow the customization requirements of menu items.

Finally, is ChatGPT good for anything at least for our task of ordering food conversationally? It does seem to process the user input and respond with something that might be useful, provided it was fact checked for accuracy. Sometimes we saw glimpses of its ability to handle more challenging linguistic constructs. However, they were obscured by the larger issue of its logic failures.

About Sanjay Mittal: -

Predictika's founder Sanjay Mittal is an accomplished AI technologist, a successful serial entrepreneur, and a seasoned business executive who has sold to and managed relationships with some of the largest companies in hospitality, technology, manufacturing, retail, and insurance. His first venture was Selectica, a pioneer in internet sales and configuration solutions for large enterprises using AI constraint-reasoning technologies, which led to a very successful $5B IPO. Sanjay was also the co-founder and CTO of Spotzot, the leader in location-based delivery of retail offers over mobile devices. Spotzot was sold to Valassis, a large privately owned company. Sanjay had a distinguished career as a research scientist at Xerox Palo Alto Research Center (PARC) with notable work in Artificial Intelligence. Sanjay holds a PhD in Computer Science (specialization in AI) from The Ohio State University, a BTech in Electrical Engineering from the Indian Institute of Technology, Kanpur and numerous patents in conversational AI, machine learning, configuration and mobile technologies.