## Feb 14th

Next Steps

* Experiment 1 use full nestful schema and user queries
* Perform Experiment 2
* Metric updates
  + Calculate the precision and recall for the required parameters
  + Calculate the precision and recall for the optional parameters
  + Calculate how many parameters were hallucinated
* Once we find out how the model does w/ optional vs. required parameters:
  + Can we prove that the LLM is putting more emphasis on the use of required parameters?
* Look into smaller parameter models
  + Deepseek
    - deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B
    - deepseek-ai/DeepSeek-R1-Distill-Qwen-7B
    - deepseek-ai/DeepSeek-R1-Distill-Qwen-32B
  + IBM Granite 8b, 2b, 1b
  + Mixtral-8x7B-Instruct-v0.1
* Log the context length and # of parameters for each model
  + E.g. 123 billion parameters for mistral large

## Feb 4th:

Write up of our project proposal:

KinetiQ is an innovative evaluation framework that addresses a critical gap in Tool-using LLM research—understanding how these models handle optional parameters in API calls. As AI-driven automation increasingly relies on LLMs to interact with APIs, ensuring accurate parameter inference is crucial for reliable tool execution. Our research will

(1) assess how LLMs infer optional parameters,

(2) contribute synthesized samples to the NestFul dataset for broader evaluation and

(3) introduce a robust metric to quantify LLM behavior in tool invocation.

By systematically testing LLMs across real-world OpenAPI specifications, we aim to uncover biases in how they prioritize required vs. optional parameters, which has direct implications for improving API-driven AI systems.

Our experiments will rigorously evaluate LLM decision-making across multiple APIs by varying prompt clarity and API schema constraints. We hypothesize that LLMs prioritize required parameters while struggling with the nuanced inference of optional ones. By modifying API schemas and prompt structures, we will measure whether LLMs adapt their behavior based on parameter necessity. The outcomes of this study will provide actionable insights for developers and researchers, leading to better-designed APIs and LLM models that handle tool invocation more intelligently. With the increasing reliance on AI for automation, this research has strong industry relevance, making it a valuable investment for advancing trustworthy and efficient AI systems.

## 

## 

## 

## Dec 19, 2024 | [KinetiQ - Research project discussion](https://www.google.com/calendar/event?eid=N2I0dnZwMm01ZDIyNG5oNnNoazlsZ2VnMmsga3AyOTczQGNvbHVtYmlhLmVkdQ)

Attendees: [Krithika Prakash](mailto:kp2973@columbia.edu) [Elizabeth Kelly](mailto:ek3222@columbia.edu) [Parijat Dube](mailto:pd2637@columbia.edu)

**What we are proposing for the Research Paper (3-fold):**

* Evaluation of the inferencing capabilities of the Tool LLMs (wrt optional parameters)
* Synthesizing new samples from NestFul - Dataset contribution
* Metric for evaluation of the behavior of the tool calling LLM in predicting parameters

**Hypothesis**: Tool calling LLMs mostly pay more attention for required parameters, some LLMs can infer the prompts better for populating optional parameters.

* **Baseline**: Take any OpenAPI 2.0 (or 3.0) spec from NestFul as-is and check what required and optional parameters are being populated by LLM. Example: Take a Flight API that originally has “Adults” as an optional parameter. Run the prompts (give me two tickets) and check which ones are being filled.
* **Experiment 1**: Keep the API as the same, but change the prompts slightly:
  + Prompt 1 (explicit) : Give me tickets for two adults -> → all LLMs can infer this correctly and will populate the adults parameter
  + Prompt 2 (slighlty vague): Give me tickets for two people → Does the LLM infer people to be the ‘adults’ optional parameter?
  + Prompt 3 (more ambiguous): Give me tickets for me and my husband → Does the LLM infer “for me and my husband” to be the ‘adults’ optional parameter ?
* **Experiment 2**: Change the API/Schema - make “adults” as a required parameter:
  + Repeat experiment 1 -> Does the LLM behave differently now that Adults is a required parameter.
* **~~Experiment 3:~~** ~~Pass entire APISpec/Schema as input to the LLM along with the combinations of prompts from Experiment 1~~

Above experiments (on many APIs from NESTFul) should prove our hypotheis that LLMs pay more attention to parameters if they are “required” more so than if “optional”.

**Next step 1**: If we have proved our hypothesis, the next step would be to check if there are any existing ways to fix this:

1. Fix by prompt engineering:

In the prompt, specify something like “LLM: please try to infer all optional parameters and try to fill values as much as possible based on the query”. With this new prompt, rerun the Experiment 1.

Here there are two business use cases. Some APIs would want all optional parameters to be filled with default values. Some other APIs would want only those that are relevant to be populated. Can the LLM behave accordingly, just by tuning the prompt. If not, how do we tell the LLM what is expected out of it?

1. Fix by frameworks:
   1. Langchain LLMs (experiment with multiple prompting methods - TAO loop - for sequence of APIs)
   2. Chain of thought, Reasoning, ReAct, etc..

If prompt engineering is also not able to solve this problem, then we have proved that there is a gap in the inference capabilities of the LLM

**Next Step 2:**

* Does this behavior of LLMs change with smaller or larger llms? Is the smaller LLM lack the inference capabilities on optional parameters ? There is a push to using smaller LLms, but if we say they lack these capabilities, then any user of smaller llms should consider this limitation.
* Does the LLM behave differently if there are a lot of optional parameters? We can plot these types of statistics. As the number of optional parameters increase, then this problem is more prevalent. We can come up with a metric “**Recall of Optional parameters**” and “**Recall of Required parameters**”

**Metric for validating if the LLM populates the parameters correctly.**

Assumption: Ground truth has the required parameters, and the needed optional parameters relevant to the prompt

* Only checking if the parameter is present, not the value
* **Highest penalty** - If LLM predicts parameters not in the spec, it is hallucinations and it should be penalized the most. —> no hallucinations should be allowed
* **Soft penalty** → If it predicts additional optional parameters not relevant to the prompt, not in the ground truth, but they are in the spec.
* **Hard penalty** → LLM doesn’t infer the prompt correctly and does not populate the optional parameters that are present in the ground truth

**Static evaluation** - What we have discussed so far above

**Dynamic evaluation** (another metric) - This is needed to identify the sensitivity of different optional parameters on the API request/response. In static evaluation we are penalizing the LLM based on matching with ground truth parameters. However some parameters could be benign and it will not have any impact on the actual output of the API. So, even though the LLM generated additional optional parameters that are not in the ground truth, if the dynamic execution response matches with the output we have in the ground truth, then we should consider the LLM to still be good. This way, we can be fair to the LLM. For this, we need API keys for RapidAPI.

**Deadlines we can target towards:**

* January 31st, 2025 - Conference(?)
* Feb 15th, 2025 - ARR review cycle
* Feb 15, 2025 - NAACL workshop <https://knowledge-nlp.github.io/naacl2025/organization.html>

We can still target the Feb 15 deadline if you have time to work on it.

Apart from the ARR feb review cycle, there is one workshop at NAACL 2025 that we can target. [https://knowledge-nlp.github.io/naacl2025/organization.html](https://urldefense.proofpoint.com/v2/url?u=https-3A__knowledge-2Dnlp.github.io_naacl2025_organization.html&d=DwMFaQ&c=009klHSCxuh5AI1vNQzSO0KGjl4nbi2Q0M1QLJX9BeE&r=NJ710DAKm6vUHMej_RcaYMprpeUSjO42AMLKun-kesU&m=NQ8J4WqHJ31-uYY3MTd0DCF_p0lFs1lyoEUuww5ibEnOcQ1rBw9glJcm7cUUL3Mj&s=L-MAMG8jjnlnI_U1mf1Q2AgpubnW-1xW865rfH4AKR4&e=)

Best,