

bulk-chain: No-string framework for Time-Efficient LLM Querying in Information Retrieval Tasks

Nicolay Rusnachenko

<https://nicolayr.com>

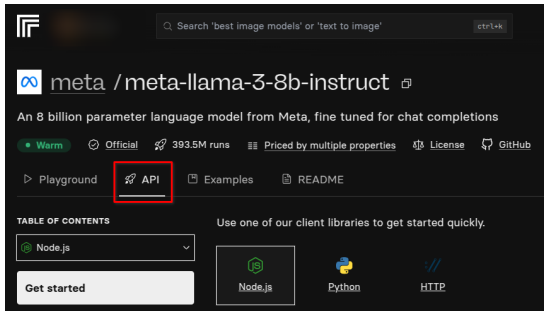
Bournemouth University
CfACTs+, Center for Applied Creative
Technologies
United Kingdom



Introduction

API Specifics (So many services, wrappers)

- OpenAI
- **Replicate**
- OpenRouter



Problem is that for each of them there is a need a provider.

Challenges of Integrating LLM Providers in the Application

Reality with mixed integration

- ✓ We can quickly implement wrapping over one service

Drawbacks:

- ✗ Scalability (adding new provider from scratch)
- ✗ Support for connection lost
- ✗ Native support for batching + streaming (optional)

Solution: isolated (no-string) framework for querying Generative AI.

Agenda

- **Framework and principles behind it:**
 - No-string framework (LLM as service provider)
 - Querying specifics (batch policies)
- **Application and Use Cases**
 - Prompting Strategies and Limitations
 - Information Retrieval and various domains
 - Ready-to-use demo and examples
 - Deploying Tutorials

No-string Framework for Time-Efficient LLM Querying

History of Origin

- Comes from necessity of evaluating models in Sentiment Analysis
- Wish for having universal setup for experimenting with various models
 - Large models require use of API for third-party providers

Prompting Schema

Prompting schema is a set of prompts, where each prompt follows the following rules:

- 1 Each prompt has a set of input parameters and an output.
- 2 Each prompts **may depend on** output of the other prompts.



It forms graphs with no cycles, i.e. **acyclic directed graphs**.

Prompting Schema: Single Prompt Example



Prompting Schema: Sentiment Analysis¹

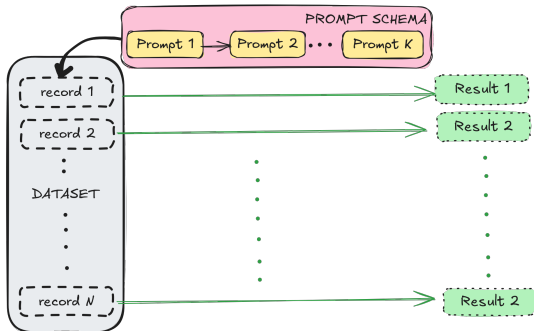
Prompt Schema Example

```
[
  {"prompt": "Given the sentence '{text}'. Which specific aspect of '{entity}' is possibly mentioned?", "out": "r_aspect"},
  {"prompt": "Given the sentence '{text}'. The mentioned aspect is about '{r_aspect}'. Based on the common sense, what is the implicit opinion towards the mentioned aspect of '{entity}', and why?", "out": "r_opinion"},
  {"prompt": "Given the sentence '{text}'. The opinion towards the mentioned aspect of '{entity}' is '{r_opinion}'. Based on such opinion, what is the sentiment polarity towards '{entity}'?", "out": "r_polarity"},
  {"prompt": "Given the sentence '{text}'. The sentiment polarity is '{r_polarity}'. Based on these contexts, summarize and return the sentiment polarity only, such as: positive, negative, neutral.", "out": "r_label"}
]
```

¹ <https://huggingface.co/nicolay-r/flan-t5-tsa-thor-base#direct-use>

Problem: Effective Processing of the Dataset

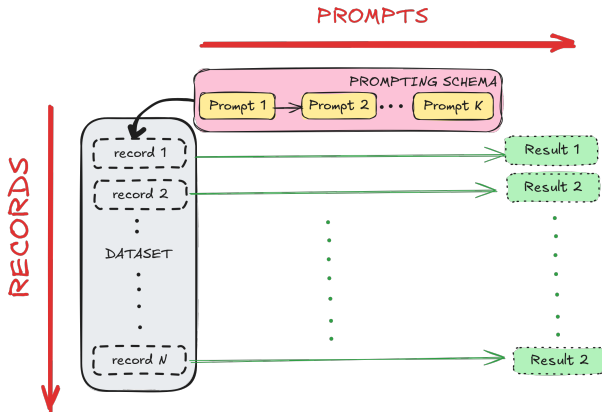
Given a DATASET for which we would like to apply Prompt Schema to generate the RESULTS.



How to make it effective?

Batching Policies: Solution

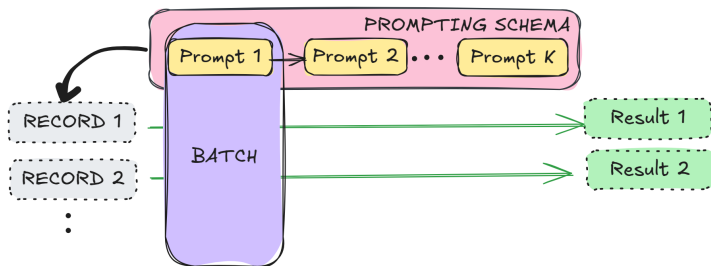
- 1 **Vertical batching:** grouping by rows first
- 2 **Horizontal batching:** grouping by prompts first



Batching Policy: Vertical Batching

Considering batch size of N .

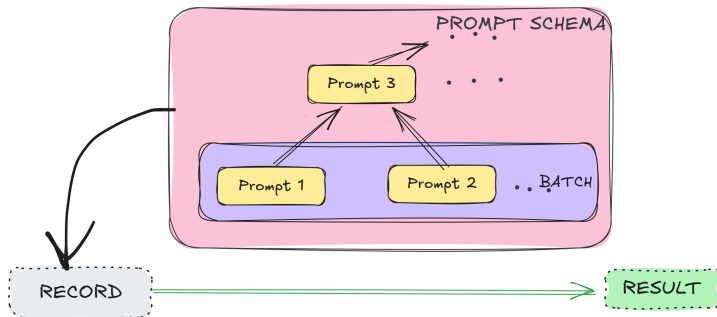
Grouping Policy: For each prompt of the Prompting Schema, we compose a batch of the $\leq N$ records from the dataset.



Batching Policy: Horizontal Batching²

Considering batch size of N .

Grouping Policy: For each record we select $\leq N$ prompts that are not depended on the output of each other.



Providers

Purpose: provide core API, suitable for handling **prompt schema**

Benefits of the isolated API for providers:

- Keeping API implementation in single file / isolated
- Have a hub of the related models³

³ <https://github.com/nicolay-r/nlp-thirdgate>

Providers: Async Modes and Use Cases (Optional)

- **Streaming:** to support interactive update for **GUI interfaces**.
- **Async querying:** await for the result of the query.
 - Single
 - Batch

Providers: API Modes

Method	Mode	Description
<code>ask(prompt)</code>	Sync	Infers the model with a single prompt.
Streaming + Async Querying		
<code>ask_stream(prompt)</code>	Sync	Returns a generator that yields chunks of the inferred result.
<code>ask_async(prompt)</code>	Async	Asynchronously infers the model with a single prompt.
<code>ask_stream_async(prompt)</code>	Async	Asynchronously returns a generator of result chunks of the inferred result.

Time Efficient Querying

Providers API + **BATCHING POLICY**

- Everything that refers to `*_stream` yields **chunks**.
- Everything that has `batch_*` yields **batches**.
- The remaining (`single`) yields **records**.

Infer Modes	Content Type
<code>single</code>	<code>record</code>
<code>batch⁴</code>	<code>batch</code>
Streaming / Async Querying	
<code>single_stream</code>	<code>chunk</code>
<code>batch_async</code>	<code>batch</code>
<code>batch_stream_async</code>	<code>chunk</code>

⁴ **Non-native**, must be supported by the provider

bulk-chain⁵

nicolay-r/bulk-chain



A no-string API framework for deploying schema-based reasoning into third-party apps



Contributor



Issues



Stars



Forks



⁵ <https://github.com/nicolay-r/bulk-chain>

bulk-chain: API Usage

We have a core API method: `iter_content`

```
from bulk_chain.core.utils import dynamic_init
from bulk_chain.api import iter_content
```

```
content_it = iter_content(
```

```
    # 1. Your schema.
```

```
    schema={
```

```
        {
            "prompt": "Summarize {text} consider {x} {y} {z} data.",
            "out": "action"
        }
    },
```

```
    # 2. Your third-party model implementation.
```

```
    llm=dynamic_init(class_filepath="replicate_184.py")(
```

```
        api_token="<API-KEY>",
```

```
        model_name="meta/meta-llama-3-70b-instruct"),
```

```
    # 3. Customize your inference and result providing modes:
    infer_mode="batch_async",
```

```
    # 4. Your iterator of dictionaries
```

```
    input_dicts_it=YOUR_DATA_IT,
```

```
)
```

```
for batch in content_it:
```

```
    # Handle your batch here
```

```
    ...
```

MODE

Infer Modes

Content Type

single

record

single_stream

chunk

batch

batch

batch_async

batch

batch_stream_async

chunk

OUTPUT TYPE

What's the type of the output?

Depends on the `infer_mode`.

Application and Use Cases

Prompting Strategies⁶

Designed for (Green), Optionally supported (Yellow), and out-of-scope (Gray) of the bulk-chain.

Technique	Description
Chain-of-Thought	Reasoning by applying sequence of prompts
Zero-Shot Prompting	Using prompts that guide the model without any examples.
Few-Shot Prompting	Providing a few examples in the prompt to demonstrate the desired response format.
Chain of Verification (CoVe)	Prompting the model to verify each step of its reasoning for accuracy
Self-Consistency	Generating multiple reasoning paths and selecting the most consistent answer
Reason & Act (ReAct)	Combines reasoning (e.g. CoT) with acting (e.g. tool calling)
Tagging-and-Labeling	Adding semantic tags or labels to retrieved data to enhance relevance

⁶ <https://github.com/Danielskry/Awesome-RAG>

Use Cases for Prompting Strategies

Use case: Using bulk-chain for Evaluation of LLM-based systems in various IR domains


Technique	Sentiment Analysis (News & Comments)	Emotion Extraction (Dialogues)
Chain-of-thought	✓	✓
Zero-shot	✓	✓
Self-consistency		✓ ⁷
Few-shot	✓ ⁸	

⁷ Could be found in: <https://arxiv.org/abs/2404.03361>

⁸ Could be found in: <https://arxiv.org/abs/2504.06947>

Evaluation Metrics

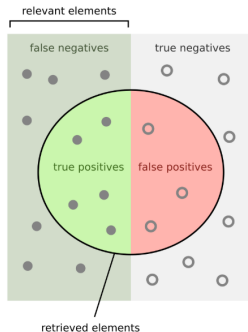
How many retrieved items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$


F1-measure

How many relevant items are retrieved?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Use Cases: Evaluation in Sentiment Analysis

Model	Strategy	News ⁹ F1(P,N)	Comments ¹⁰ F1(P,N)	Emotion Extraction ¹¹ F1 (7 classes)
Finetuned Models				
FlanT5 _{base}	CoT	59.75	-	24.28
FlanT5 _{base}	ZeroShot	57.01	-	22.27
FlanT5 _{xl}	CoT	65.09	68.03	-
FlanT5 _{xl}	ZeroShot	68.20	57.65	-
Non-Finetuned Models				
Mistral-7B _{v1}	ZeroShot	49.46	85.89	2.54
Mistral-7B _{v1}	CoT	42.34	74.51	-
Mistral _{Small-03-2025}	ZeroShot	43.11	86.81	
Mistral _{Small-03-2025}	CoT	42.34	80.55	

⁹ RuSentNE-2023 dataset of short news messages

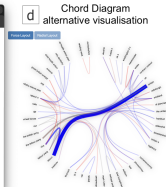
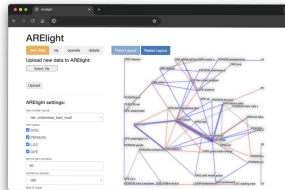
¹⁰ Dataset of 144 comments on AI tools usage experience (private)

¹¹ Dialogues from FRIENDS dataset: <https://arxiv.org/abs/2404.03361>

Demo: Aggregated Visualization for Relations

Using bulk-chain in demo projects with GUI.

- **ARElight-d3js** – using bulk-chain for inference in the GUI¹²



Demo: Rendering Content in Table

Using bulk-chain in tk-sheet client¹³

Sentiment Analysis Demo (meta/meta-llama-3-70b-instruct)				
	A	B	C	D
1	Based on the sentence, it can be inferred that the aspect of the camera being referred to is its quality or performance, as the speaker describes it as "absolutely stunning". This suggests that the camera takes high-quality photos or has impressive features, but the sentence does not provide	Based on the sentence, the implicit opinion towards the mentioned aspect of the 'camera' is POSITIVE. The reason is that the speaker describes th	Based on the analysis, the sentiment polarity towards the 'camera' is POSITIVE.	For the 'camera', the sentiment polarity is: positive
2	Based on the sentence, the specific aspect of the "keyboard" that is possibly mentioned is its "feel" or "build quality", which is described as "a bit cheap".	Based on the sentence, the implicit opinion towards the mentioned aspect of the 'keyboard' is NEGATIVE. The reason is that the word "cheap" typical	Based on the analysis, the sentiment polarity towards the 'keyboard' is NEGATIVE.	The sentiment polarity towards the 'laptop' is POSITIVE, and towards the 'keyboard'
3	Based on the sentence, the specific aspect of 'food' that is possibly mentioned is its 'taste'. The sentence states that the food was "a little bland", which suggests that the speaker is commenting on the flavor or taste of the food.	Based on the sentence, the implicit opinion towards the mentioned aspect of 'food' (its taste) is NEGATIVE. The reason is that the speaker describes th	Based on the analysis, the sentiment polarity towards 'food' is NEGATIVE.	Negative
4	Based on the sentence, it's likely that the aspect of the storyline being referred to is its coherence, originality, or overall quality, rather than a specific element like character development, pacing, or plot twists. The phrase "very weak" suggests that the storyline lacked depth, was unengagi	Based on the sentence, the implicit opinion towards the mentioned aspect of 'storyline' is NEGATIVE. The reason is that the word "weak" has a n	Based on the analysis, the sentiment polarity towards 'storyline' is NEGATIVE.	Negative
5	The specific aspect of 'WiFi' mentioned in the sentence is its 'reliability'.	Based on common sense, the implicit opinion towards the mentioned aspect of 'WiFi' (its reliability) is NEGATIVE. The reason is that the word "unreliable" ha	Based on the analysis, the sentiment polarity towards 'WiFi' is NEGATIVE.	Negative
6	Based on the sentence, the specific aspect of 'speakers' that is possibly mentioned is their			
7				

¹³ <https://github.com/nicolay-r/bulk-chain-tksheet-client>

Deployment¹⁴

Minimalistic deployment of bulk-chain in web GUI.

Stack:

- **Client:** HTML + JS
- **Server:** Python + FastAPI + bulk-chain + LLM Provider (Replicate)

Videos:

- **Part 1: support streaming:** <https://youtu.be/XgByPLLsiCI>
- **Part 2: support LLM querying:** <https://youtu.be/yNKYJzlKxh0>

¹⁴ <https://github.com/nicolay-r/bulk-chain-web-integration>

Conclusion

- Problems of Integrating LLM in the Application.
 - Scalability of systems, having universal API for: streaming, batching.
- Time-efficient LLM-querying, based on:
 - Support for async and streaming modes.
 - Batching policies.
- Use cases and demo projects.

Thank you for attention!



<https://nicolayr.com>