LLM Workshop: Using LLMs for Research and Policy Analysis[1]

Minchul Shin and Vitaly Meursault[2]

December 19, 2025

[1] This slide deck is based on an internal LLM workshop held for the Research Department of the Federal Reserve Bank of Philadelphia in the Fall of 2025 and covers Minchul's portion. The LLM space is moving quickly, and some of this material may already be outdated (as of December 2025). Read at your own risk.

Plan for Today

Minchul

- Structured output and function calling
- Simple examples: Classification, Metadata extraction, RAG Agent, etc.

Vitaly

- More serious applications to research workflows

Motivation: Economist's Briefing Helper

Think about a common part of an economist's job: preparing a recurring briefing note

- Prepare a short report on recent economic developments over a pre-defined date range
- Fairly standardized structure

Question: Can we automate the routine parts so that we can focus on more creative work?

Economist's Briefing Helper (hypothetical) is an example workflow we will use to illustrate these ideas.

Economist's Briefing

Here is a typical way to do it:

1. Check the economic release calendar for the reporting period
2. For each day, review the relevant indicator releases
3. For each indicator:
   3.1 Read the official releases and related commentary
   3.2 Summarize and keep notes for writing
4. Once enough material is collected, start writing the briefing note

Economist's Briefing: Writing Process with Collected Data

CPI (Aug)
+0.3% m/m, core **+0.2%**

Unemployment rate
3.8% (−0.1pp)

Retail Sales
+0.4% m/m, control **+0.2%**

Import Prices
-0.3% m/m, ex-petroleum **-0.1%**

Labor

Unemployment edged **down** to 3.8%. Wage gains are easing, consistent with gradual cooling but resilient labor demand.

Prices

Core CPI slowed to **0.2%** m/m. Softer **import prices** signal less pipeline pressure; services disinflation is proceeding gradually.

Demand

Retail sales rose again. Consumption is cooling only gradually as real incomes firm with easing inflation.

Key to Automation

What LLMs can offer in this process:

- **Summarization**
  This is an official report on CPI. Summarize it in three sentences.
- **Classification**
  Classify the given document into one of the following categories:
  {*Labor, Prices, Demand*}

Issue with LLMs

LLMs are designed and trained to serve a general public. They are talkative by default.

Categorize the given document based on the following categories: {Labor, Prices, Demand}

Document:
Retail sales rose again. Consumption is cooling only gradually as real incomes firm with easing inflation.

The document should be categorized under **Demand**.

Here's why:

- It discusses **retail sales** and **consumption**, which are directly related to aggregate demand in the economy.
- While it mentions **real incomes** and **inflation**, those references serve as context for why demand (consumption) is evolving, not as the main focus.
- There's no direct discussion of **Labor** (employment, wages, participation) or **Prices** (inflation measures, cost indices) as standalone topics.

✅ **Final Category: Demand**

Structured Output

Structured output constrains an LLM to respond in a predefined structure (often JSON), which makes downstream parsing and validation reliable.

Our classification example:

Pre-specified format (Schema[3]):
{ "Category": ["Labor", "Prices", "Demand"],
"Evidence": "... short supporting text ..." }

Example output:
{ "Category": "Demand",
"Evidence": "Retail sales rose again; consumption is cooling gradually." }

[3] This is an informal schema to give you an idea.

In Practice (OpenAI API Example)

```
schema = {
    "name": "categorize_document",
    "schema": {
        "type": "object",
        "properties": {
            "Category": {
                "type": "string",
                "enum": ["Labor", "Prices", "Demand"]
            },
            "Evidence": {
                "type": "string"
            }
        }
    }
}


doc = "Retail sales rose again. Consumption is cooling only gradually..."


response = client.chat.completions.create(
    model="gpt-4.1",
    messages=[
        {"role": "user", "content": f"Categorize: {doc}"}
    ],
    response_format={"type": "json_schema", "json_schema": schema}
)
```

Structured output

Structured output is useful when we want LLMs to respond in a pre-specified format

There are several ways to achieve structured output:

- Inject an additional prompt describing the schema, followed by post-validation
- Provider-enforced schema adherence (e.g., “strict” JSON Schema)
- Constrained decoding (grammar/automata-guided decoding)
- … it remains an active area of research

Economist's Briefing Helper

[Procedure for a Briefing Note]

1. Check the economic release calendar for the reporting period **(Programmable)**
2. For each day, review the relevant indicator releases **(Programmable)**
3. For each indicator release:
   3.1 Read official releases and related commentary **(Use LLMs)**
   3.2 Summarize and keep notes for writing **(Use LLMs)**
4. Once enough material is collected, start writing the briefing note:
   4.1 Categorize each document into sections **(Use LLMs)**
   4.2 For each section, summarize the categorized documents **(Use LLMs)**

[Final Remarks]

- The same approach applies to many recurring research and policy-adjacent workflows (e.g., weekly monitoring notes, meeting prep, internal memos).
- If you run models locally, you can keep sensitive text on your own infrastructure.

Additional Examples

Two additional examples of structured output:

1. Metadata extraction from earnings reports
2. Function calling and RAG agents

Example 1: Earnings Report Analysis

Jonas and I analyzed the effect of tariff shocks on prices (Arias and Shin, 2026).

We externally validate our empirical results by analyzing firm-level filings.

1. We downloaded 10-K earnings reports ($\approx$200 relevant firms).
2. For each document ($\approx$10,000 words), we extracted the following information:
    - Whether the firm reported higher costs due to tariffs (Yes or No).
    - Whether the firm reported increasing prices for customers due to tariffs (Yes or No).
    - Exact passages that discuss how tariffs impacted the business (Text).

Example 1: Earnings Report Analysis

Defining the schema in code:

```
schema = {
  "faced_higher_cost_due_to_tariff": {
    "type": "boolean",
    "description": "true = firm reports higher costs due
       to tariffs; false = does not."
  },
  "increased_prices_due_to_tariff": {
    "type": "boolean",
    "description": "true = firm reports increasing prices
        for customers due to tariffs; false = does not."
  },
  "exact_passages": {
    "type": "string",
    "description": "Exact passages that discuss how
       tariffs impact the business."
  },
}
```

Example 1: Earnings Report Analysis

Finally, turn them into a table for analysis
(e.g., the proportion of firms that reacted to tariffs).

NAICS	Number of Companies	Tariff Mention (%)	Cost Increase (%)	Price Pass-Through (%)
332 (Fabricated Metal)	45	82.2%	80.0%	64.4%
333 (Machinery)	100	66.0%	54.0%	47.0%
3352 (Appliance)	7	71.4%	71.4%	85.7%
33611 (Vehicle)	5	60.0%	60.0%	40.0%

Table 5: Tariff Mentions, Cost Increases, and Price Pass-Through in 10-K Filings. NAICS 332 refers to Fabricated Metal Product Manufacturing; NAICS 333 refers to Machinery Manufacturing; NAICS 3352 refers to Household Appliance Manufacturing; NAICS 33611 refers to Automobile and Light Duty Motor Vehicle Manufacturing. Tariff Mention indicates firms that explicitly discuss tariffs in their 10-K filings. Cost Increase indicates firms that report higher input costs. Price Pass-Through indicates firms that report adjusting product prices in response to cost increases.

Example 2: Function Calling

Function calling can be viewed as structured output

- It gives LLMs the ability to use a tool
- Recall: LLM + Tool = Agent

For LLMs to use an existing function, they need to determine three things:

1. Whether to call a function (Yes or No)
2. Which function to use (choose from the list of available functions)
3. The function arguments

Example 2: Function Calling

Suppose we have a function:

News Article Search Function

- Name: Retrieve_news
- Input: Search query and date range
- Output: A list of relevant news articles (titles and summaries)

For example:

$$output = Retrieve\_news(\text{"Consumer sentiment"}, \text{"(2025-09-08, 2025-09-22)"})$$

Here, `output` contains a list of news articles relevant to consumer sentiment.

Example 2: Function Calling

User query: Can you search and summarize news articles on consumer sentiment for the last two weeks?

LLM's response should include:

- Use function $\rightarrow$ Yes
- Which function $\rightarrow$ Retrieve_news
- Function arguments
  $\rightarrow$ {query = "consumer sentiment", date_range = "(2025-09-08, 2025-09-22)"}

Example 2: Function Calling with LLM+Tool

Conversation pattern with LLM+Tool

User query: Can you search and summarize news articles on consumer sentiment for the last two weeks?

LLM's response: `Function_call = (Yes, Retrieve_news, {query = ''consumer sentiment'', date_range = ''(2025-09-08, 2025-09-22)''})`

User's computer: Executes the function when the LLM returns a function-call message, then sends the function output back to the LLM.

LLM's response: Based on my news search, consumer sentiment over the last two weeks is ...

Example 2: Function Calling and RAG Agent

Often, the previous type of agent, LLM+Database, is referred to as a RAG agent.

Retrieval-Augmented Generation (RAG)

- Retrieve relevant passages from a database/index.
- Generate an answer using the retrieved passages as additional context.

RAG Agent (LLM + tools)

- The LLM decides *whether* to retrieve, *what* to query, and possibly *when* to retrieve again.
- This is commonly implemented via function calling.

Economic applications

- A basic RAG system might index recent economic news and public releases.
- You can expand it to include additional sources (including proprietary or restricted data) with appropriate access controls.

Caveats of structured output

If you are familiar with economic forecasting, structured output is analogous to conditional forecasting (imposing restrictions on forecasts).

One can imagine that performance depends on whether the restrictions are aligned with the true data-generating process.

A similar point applies to structured output:

- Constraints can increase *format validity* while potentially reducing *semantic accuracy* (e.g., incorrect field values, inconsistent entries).
- Constraints often require additional computation, increasing latency and cost.

For example, see Beurer-Kellner et al. (2024) for further discussion and recent attempts to improve structured output generation.

Thank you!! This is the end of my part.

References

Arias, Jonas E. and Shin, Minchul (2026). *An Empirical Characterization of the Dynamic Effects of U.S. Trade Policy: The Case of Steel*.

Beurer-Kellner, Luca, Marc Fischer, and Martin Vechev (2024). "Guiding LLMs the Right Way: Fast, Non-Invasive Constrained Generation." *arXiv* preprint arXiv:2403.06988.