

Why and How to Evaluate LLM Outputs

From Vibes to Systematic Assessment

Franck Albinet

franckalbinet@gmail.com

Independent Data Science & AI Consultant

October 10, 2025

Welcome to Day 3

Yesterday's Achievement

You Built Prompts!

On Day 2, you:

- Learned the 7 building blocks of effective prompts
- Built a paper extraction prompt iteratively
- Saw improvements with each cycle
- Got outputs that seemed pretty good

But here's the question: How do you *really* know if it's working?

Today's Challenge

Moving Beyond "Vibes"

Today we'll learn:

- Why LLM evaluation is not optional
- How to systematically identify failure modes
- Techniques for scalable evaluation
- Hands-on practice with real examples

Goal: Replace gut feelings with systematic assessment

The Evaluation Problem

Why Traditional Testing Doesn't Work

The LLM Challenge

Unlike traditional software:





- LLMs don't produce deterministic outputs
- Same input can yield different responses
- Responses are subjective and context-dependent
- Quality is multifaceted

Example: A summary might be factually accurate but miss the key insight.
Or sound persuasive while being wrong.

The “Vibes Are Off” Problem

What Does Quality Even Mean?

Consider a paper summary that:

-  Contains no factual errors
-  Follows the specified format
-  Misses the main research contribution
-  Emphasizes minor details

Is this good or bad? It depends on your needs!

The core challenge: How do we assess whether an LLM pipeline is performing adequately? And how do we diagnose where it's failing?

Quality Is Multi-Faceted

Multiple Dimensions to Consider

A “good” LLM output must satisfy multiple criteria:

- **Accuracy:** Factually correct information
- **Completeness:** Covers all important points
- **Relevance:** Focuses on what matters for your purpose
- **Clarity:** Easy to understand and well-structured
- **Consistency:** Follows specified format and style

The challenge: An output can excel in some dimensions but fail in others

Example: A perfectly formatted, grammatically flawless summary that misses the main point

Example: Missing the Key Insight

When Accuracy Isn't Enough

Original paper's main contribution:

"Method X performs 10× better than Method Y in low-temperature conditions ($<5^{\circ}\text{C}$)."

LLM summary says:

"The paper compares Method X and Method Y across various conditions and presents statistical results."

Is this accurate? Yes, technically. **Is it useful?** No!

What Went Wrong?

The Missing Pieces

The summary missed:

- **The magnitude:** 10× performance difference (dramatic!)
- **The key condition:** <5°C (when does it matter?)
- **The significance:** Why this finding changes things

Result: Technically correct but practically useless for literature review

This is the “vibes are off” problem - you can’t point to a factual error, but something is clearly wrong

Scale Makes It Worse

The Production Reality

When you process 50 papers:

- Can't manually review every output thoroughly
- Can't rely on "it looks okay" as a quality bar
- Need to catch systematic failures
- Must understand performance patterns

Without systematic evaluation: You're flying blind

What You've Already Been Doing

Open Coding: Your First Evaluation Method

You've Been Evaluating All Along!

Days 1-2, you captured observations:

- “The output format keeps changing”
- “It missed the methodology details”
- “Figures aren’t described consistently”
- “Sometimes it’s too brief, sometimes too detailed”

This is open coding - capturing observations without filtering or organizing

The Value of Open Coding

Unbiased Observation

Why open coding works:

- No preconceived categories
- Captures unexpected issues
- Lets patterns emerge naturally
- Avoids confirmation bias

The challenge: It doesn't scale, and patterns stay buried in observations

Your Observations So Far

What We've Collected

From your shared doc, you've noted:

- [TO BE FILLED BASED ON DAY 1-2 OBSERVATIONS]

Next step: How do we make sense of all these observations?

From Observations to Patterns

Axial Coding: Finding the Structure

Organizing Your Observations

Axial coding: Grouping related observations into meaningful categories

The process:

1. Collect all open codes
2. Look for similar themes
3. Group related observations
4. Name the patterns
5. Identify core failure modes

Key insight: This transforms scattered observations into actionable insights

Live Demo: Axial Coding with Solvelt

Let's Analyze Your Observations Together

I'll now use an LLM to:

- Load your Day 1-2 observations from the shared doc
- Identify common themes and patterns
- Group similar failure modes
- Show you what emerges

Watch how AI can help with qualitative analysis at scale

[Live Demo Section]

Axial Coding in Action

[Loading observations, analyzing patterns, and generating categories ...]

Expected output:

- 3-5 main failure mode categories
- Examples of each
- Frequency/severity indicators

What We Found

Common Failure Modes

From your observations, the main patterns are:

1. **[Category 1 from axial coding]**

- [Examples]

2. **[Category 2 from axial coding]**

- [Examples]

3. **[Category 3 from axial coding]**

- [Examples]

Now we have specific, named failure modes to work with!

Why This Matters

From Vague to Specific

Before axial coding:

“Sometimes the output isn’t quite right”

After axial coding:

“In 40% of cases, figure descriptions are missing or incomplete”

This specificity enables: Targeted prompt improvements AND scalable evaluation

Scaling Evaluation

The Manual Evaluation Bottleneck

The Reality Check

You've identified specific failure modes. Now:

- Can you manually check 100 paper extractions?
- Can you consistently apply the same criteria?
- Can you track improvements over time?
- Can you evaluate in production?

Manual evaluation doesn't scale - but the principles can!

LLM-as-Judge: The Key Insight

Using AI to Evaluate AI

The idea: Use an LLM to detect specific failure modes at scale

Wait, isn't that circular reasoning?

No! Here's why:

- We're not asking it to improve itself
- We're asking it to do binary classification
- We validate against human labels
- Mathematics handles the imperfection

How LLM-as-Judge Works

The Simple Version

1. **Define a specific failure mode** (from axial coding)
2. **Create a judge prompt** to detect that failure
3. **Validate on test set** (human-labeled examples)
4. **Run at scale** on new outputs
5. **Correct for judge errors** using statistics

Key requirement: The judge just needs to be better than random chance

The Statistical Foundation

Why Imperfect Judges Work

From medical diagnostics:

- Imperfect tests can still estimate disease prevalence
- We measure the test's error rates
- Mathematics corrects for systematic errors
- Same principle applies to LLM evaluation

The Rogan-Gladen correction (1978) handles this mathematically

For details: see my blog post at franck-albi-net.pla.sh/post/llm-as-a-judge

What You Need

The Requirements

For LLM-as-judge to work:

1. **Clear, binary failure mode** - “missing figures” not “poor quality”
2. **Human-labeled test set** - ground truth for validation
3. **Judge better than random** - $\text{TPR} + \text{TNR} > 1$
4. **Statistical correction** - accounts for judge errors

Critical: Garbage in, garbage out - the failure mode must be well-defined

Hands-On Preview

Today's Practice: Session 3B

Toy Example - Plagiarism Detection

What you'll do:

- Write a judge prompt to detect plagiarism
- Test it on sample texts
- See LLM-as-judge in action

Why plagiarism? Clear binary task + prepares for Day 4 IP constraints

Today's Practice: Session 3C

Apply to Your Work

Choose one:

Option A: Reproduce axial coding on your Days 1-2 observations

Option B: Design how you'd integrate LLM eval in your research domain

Goal: Make evaluation concrete for YOUR context

The Plagiarism Example

Why This Toy Problem?

Perfect for learning because:

- Universally understood concept
- Clear binary classification
- Easy to create test examples
- Directly relevant to Day 4 (IP constraints)

You'll experience: Manual labeling → judge prompt → evaluation → interpretation

What You'll Learn

Skills for Today

By end of Day 3, you'll be able to:

- Identify specific failure modes systematically
- Create judge prompts for binary classification
- Validate judges against human labels
- Understand when evaluation is reliable
- Apply this to your own use cases

Most importantly: You'll move from “vibes” to systematic assessment

The Evaluation Mindset

Evaluation Is Not Optional

Why This Matters

In production LLM systems:

- Outputs vary unpredictably
- Failures can be subtle
- Scale makes manual review impossible
- Users depend on reliability

Without evaluation: You can't improve, can't trust, can't deploy confidently

The Iterative Cycle

How Evaluation Drives Improvement

1. **Look at your data** (inputs and outputs)
2. **Prompt engineering** (build/refine)
3. **Open coding** (capture observations)
4. **Axial coding** (identify failure patterns)
5. **Systematic evaluation** (LLM-as-judge, code-based & deterministic)
6. **Prioritize** prominent failure modes
7. **Refine prompt** → back to step 2

Exercise: Which steps help address which of the Three Gulfs?

Good Enough vs. Perfect

Setting Your Bar

Questions to ask:






- What's the cost of each failure type?
- How much manual review can you afford?
- What's your acceptable error rate?
- Can humans catch what the LLM misses?

There's no universal threshold - it depends on your use case and constraints

Questions & Transition

Before We Start Hands-On

Recap

1.  LLM evaluation is not optional
2.  Open coding captures observations
3.  Axial coding identifies patterns
4.  LLM-as-judge scales detection
5.  Statistics handle imperfection

Any questions before we dive into the plagiarism example?

Next: Hands-On Practice

Session 3B Coming Up

You'll receive:

- A dataset of 15-20 text samples
- Some plagiarized, some original
- Instructions for the evaluation exercise

You'll create:

- Manual labels for a test set
- A judge prompt to detect plagiarism
- Validation against your labels