Prompt Engineering / Prompt Optimization

Generate test harness, metrics, and statistical significance calculator for comparing two prompts.
Difficulty: Advanced
Model: GPT-4 / Claude / Gemini
Use Case: Prompt Comparison, Performance Measurement, Data-Driven Tuning
Updated: May 2026
Why This Prompt Exists
“Prompt A feels better than Prompt B” is not a valid conclusion. Without rigorous A/B testing, you’re guessing — and probably wrong.

You get:

- choosing the wrong prompt because you tested on easy examples
- no statistically valid way to compare prompts
- metrics that don’t measure what matters
- sample sizes too small to detect real differences
- results that don’t replicate when you test again

But good A/B tests have structure:

- test harness: runs both prompts on same inputs
- metrics: accuracy, latency, cost, consistency, user satisfaction
- sample size: enough to detect meaningful differences
- randomization: order effects controlled
- statistical test: appropriate for metric type

Without design, your A/B test will mislead you.

This prompt designs rigorous A/B tests for prompt comparison.

The Prompt

```
Assume the role of an experimental design statistician who designs
prompt A/B tests.

Your task is to create a test plan for comparing two prompts.

Generate:

1. TEST OBJECTIVE
   - What you're trying to measure (e.g., "Which prompt produces more
accurate summaries?")
   - Primary metric (e.g., "Factual accuracy score 0-10")
   - Secondary metrics (e.g., "Latency, token usage, verbosity")

2. TEST DESIGN
   - Unit of randomization: [Input / User / Session]
   - Sample size needed (for statistical power)
   - Allocation ratio (50/50, 80/20, etc.)

3. METRIC DEFINITION TABLE

| Metric | Type | How to measure | Minimum detectable difference |
|--------|------|----------------|-------------------------------|
| [name] | [binary/continuous/ordinal] | [method] | [X units] |

4. EVALUATION PROTOCOL
```

   - Human evaluation (rubric, blinded)
   - Automated evaluation (LLM-as-judge, exact match)
   - Hybrid approach

5. STATISTICAL ANALYSIS PLAN
   - Primary test: [t-test / chi-square / Mann-Whitney / Bayesian]
   - Significance threshold: (p < 0.05 / 0.01 / other)
   - Multiple comparison correction needed? (Yes/No)

6. SAMPLE TEST HARNESS
   - Python/pseudocode for running the test

7. INTERPRETATION GUIDANCE
   - When to declare a winner
   - When to run longer
   - When to declare no significant difference

INPUTS:

Prompt A description:
[PASTE OR DESCRIBE]

Prompt B description:
[PASTE OR DESCRIBE]

Task type:
[CLASSIFICATION / GENERATION / EXTRACTION / OTHER]

Available test inputs (count):

```
[E.G., "500 labeled examples"]

Evaluation budget (human time, API cost):
[E.G., "Can pay for 100 human ratings"]

RULES:
- Sample size must be sufficient to detect meaningful differences (don't test on 10 examples)
- Use blinded evaluation when possible (raters shouldn't know which prompt produced which output)
- Pre-register your analysis plan (don't change metrics after seeing results)
- Consider practical significance, not just statistical significance (a 0.1% improvement may not matter)
- Run A/A tests (same prompt twice) to verify your measurement system
```

How To Use It

- Run this before making any significant prompt change — A/B test, don't guess.
- Use sample size calculators to determine how many test examples you need.
- Blind your evaluations (raters shouldn't know which prompt they're evaluating).
- Pre-register your analysis plan to prevent p-hacking.
- If you can't run a full A/B test, at least run the same test inputs on both prompts.

Example Input

Prompt A description:
"Zero-shot: Summarize this text in 2-3 sentences."

Prompt B description:
"Few-shot with 2 examples: You are a professional editor. Summarize the following text in

exactly 2-3 sentences. Focus on main argument only. Example 1: ... Example 2: ..."

Task type:
"Generation — summarization"

Available test inputs:
"200 news articles"

Evaluation budget:
"Can pay for 100 human ratings (can rate 5 summaries per hour at $15/hr)"

Why It Works
Most prompt engineers compare prompts informally — "I tried both, this one seemed better" — which is unreliable.

This framework improves outcomes by forcing:

- metric specification (what counts as "better"?)
- sample size calculation (enough data to be confident)
- blinded evaluation (no bias from knowing which prompt)
- statistical analysis plan (prevent p-hacking)
- interpretation guidance (what to do with results)

Great A/B test design doesn't guarantee the right answer — it guarantees you'll know the right answer when you see it.

Build Better AI Systems

Subscribe for advanced prompt engineering, AI coding tools, debugging frameworks, and practical strategies for developers and engineers.

Carefully engineered prompts for people doing real work.

Share this:

- Share on Facebook (Opens in new window) Facebook
- Share on X (Opens in new window) X

See also Failure-Driven Refiner