# Peripheral Attention Engineering: Structured Peripheral Context Improves LLM Decision Quality

**Anthony Lee** — A.I. Plus Automation

*Preregistered at OSF: 10.17605/OSF.IO/W3XYV*

---

## Abstract

Large language models answer queries using directly relevant material, but they lack the lateral, situational context a competent human consults silently before responding. This paper introduces Peripheral Attention Engineering (PAE): the deliberate structuring and provision of unrequested-but-decision-relevant context to shape LLM responses. A controlled within-subjects experiment across four domains (customer service, medical triage, legal advisory, financial advisory; N=500 total scenarios) and six conditions tests whether structured peripheral context improves response quality when controlling for token count, domain relevance, and prompt structure. Results confirm all preregistered hypotheses: peripheral context produces large gains in response quality (baseline mean=1.63–2.06 vs anomaly-flagged mean=4.01–4.90 on a 5-point scale, Cohen's d=2.02–3.10, all $p<0.0001$); the improvement is attributable to informational content, not prompt length (noise d=0.04, n.s.) or domain relevance (expanded direct d=0.06, n.s.); slot-structured context outperforms equivalent unstructured context (d=0.13, $p<0.0001$); and the effect generalizes across three model families and all four domains. A Phase 2 fine-tuning experiment demonstrates that contextual reach — the unprompted instinct to retrieve peripheral context before answering — is trainable, with a fine-tuned Gemma 4 12B model achieving 100% context-request and context-use rates, and outperforming all Phase 1 generators on the full benchmark. A knowledge-graph substrate (Phase 3) and a practical Python library ( `pae.context.wrap()` ) are released as open-source artifacts.

---

## 1. Introduction

Consider a customer asking "What's your return policy?" A standard large language model (LLM), equipped with the company's policy document, answers correctly: "Returns are accepted within 30 days with a receipt." But a competent human agent, seeing that the customer is a six-year VIP with a $12,000 recent order, that it is December 28th during peak holiday returns, and that the customer's return rate has tripled to 47% in the past 90 days, gives a fundamentally different answer. The literal policy is the same. The *answer* should not be.

This paper proposes that LLM responses suffer when the model lacks *peripheral context* — lateral, situational information that is not strictly required to answer a query but is required to answer it well. The observation is

not new. Retrieval-augmented generation (RAG) pulls directly relevant material. Personalization systems inject user history. Context engineering optimizes prompt assembly. What is missing is a controlled experimental design that isolates the contribution of *structured* peripheral context from the contribution of *mere context length*, a theoretical grounding for what makes peripheral information worth surfacing, and a practical artifact that practitioners can adopt without rebuilding the infrastructure.

We introduce Peripheral Attention Engineering (PAE): the deliberate structuring and provision of unrequested-but-decision-relevant context to an LLM, such that the model's response is shaped by surrounding information a competent human would consult silently before answering.

### 1.1 Decomposition

The problem decomposes into three sequential components:

1. **Peripheral relevance** — the existence of unrequested-but-useful context surrounding the asker, the situation, and the entities involved.
2. **Contextual reach** — the unprompted instinct to retrieve that peripheral context before answering.
3. **Taste** — the commitment to a specific framing and tone shaped by what was found.

Current LLM training optimizes for answering the literal question with the literally relevant material. The middle component — contextual reach — is the most addressable with current architectures and is the focus of this paper.

### 1.2 Contributions

This paper makes four contributions:

1. A controlled, preregistered experiment (N=200, 5 conditions, 3 model families) that isolates the effect of peripheral context structure from content from token count.
2. Empirical evidence that structured peripheral context improves LLM response quality by 2.5×, with the effect generalizing across model families.
3. A theoretical grounding in surprisal — the proposition that peripheral context worth surfacing is information with high expectation divergence relative to priors.
4. An open-source Python library ( `pae.context.wrap()` ) and a fine-tuned model that demonstrate contextual reach is trainable.

---

## 2. Related Work

The observation that LLMs benefit from information beyond the literal query has been studied under several adjacent framings.

**Sufficient Context.** Joren et al. (ICLR 2025) introduced the notion of "sufficient context" for RAG systems and developed a method to classify whether a given context contains enough information to answer a query. Their work separates two failure modes: context being insufficient versus the model failing to use sufficient context. This separation informs the design of our evaluation. The gap relative to PAE is that sufficient context concerns whether directly relevant material is enough; PAE concerns whether information that is *not directly relevant* to the query nonetheless shapes the response.

**Context Tuning for RAG.** Apple's Machine Learning Research group (2024) proposed Context Tuning for RAG, using numerical, categorical, and habitual usage signals to retrieve and rank context items beyond what the query explicitly requests. This is the closest existing work to PAE's lateral-retrieval framing. The gap is that context tuning treats the problem as tool retrieval for agents, not as response shaping.

**Retrieval-Augmented Personalization.** A substantial 2024–2026 literature investigates injecting user history and profiles into LLM prompts (UQABench, LoGo, Attn-GS, and contextual bandit optimization for profile retrieval). This work is framed primarily as preference matching — making outputs align with what the user would prefer. PAE is framed as decision shaping — making outputs reflect what is true about the situation, including information that may run against what the asker would prefer.

**Context Engineering.** The term has been popularized by Karpathy and formalized by Gartner (2025) as the practice of populating the context window with the right information at the right time. Recent work on "context rot" (Chroma, 2025) showed that all 18 frontier models tested degrade as input length grows, with some dropping from 95% to 60% accuracy past a threshold. This strengthens the case for PAE's length-matched noise condition: peripheral context can only be argued to help if its informational value can be separated from the cost of additional tokens. Context engineering as currently practiced focuses on assembly, ordering, and compression. PAE asks a more specific question: what *kinds* of unrequested information should be assembled in the first place, and how should it be structured to be used.

---

## 3. Theoretical Framework

### 3.1 Why Standard RAG Is Insufficient

Standard retrieval-augmented generation retrieves on semantic similarity to the query. "What's your return policy?" pulls policy documents. It does not pull customer history, because customer history is not semantically similar to the question — it is adjacent to the *asker*. PAE requires a different retrieval axis: entity-anchored rather than query-anchored. This is consistent with the gap identified in Apple's context-tuning work: semantic search fails when the relevant signal is not in the query.

### 3.2 Surprisal as the Foundation of "Interesting"

The cleanest formal grounding for what makes peripheral context worth surfacing is *surprisal*: information

about an entity that has high entropy given priors. A customer's existence is not interesting; their deviation from the expected pattern is. This implies that pre-computing useful entity profiles is fundamentally about pre-computing what a reasonable predictor would expect about the entity, and where reality diverges from that expectation. **Profiles are diffs against priors, not flat records.**

### 3.3 Design Principles

Seven principles follow from the surprisal grounding:

1. **Retrieve on entities, not just queries.** Each entity in a message has its own peripheral profile that gets pulled regardless of query similarity.
2. **Stratify context by relationship to the query.** Direct answer material, asker context, situational context, and anomaly flags occupy distinct slots.
3. **Pre-chew profiles into anomaly-flagged summaries.** Raw records are not useful; flagged deviations are.
4. **Encode trajectories, not just states.** "Return rate has tripled in 90 days" outperforms "returned 12 items."
5. **Use contrastive profiles.** Encode what is distinctive about an entity relative to its peer cluster, not what is shared.
6. **Condition lenses on situation type.** The same entity yields different profiles depending on whether the situation is buyer, risk, or relationship.
7. **Feed outcomes back.** Cases where peripheral context should have been flagged but was not are training signal for the profile generator.

---

## 4. Hypotheses

The experiment tests a primary hypothesis and three subordinate claims:

**H1 (Primary):** Providing structured, anomaly-flagged peripheral context measurably improves LLM response quality on customer service tasks requiring decision-relevant context, controlling for token count and direct-answer material.

**H2 (Length Confound):** The improvement is attributable to the informational content of peripheral context, not to its mere presence. Length-matched noise will not produce equivalent gains.

**H3 (Structure Effect):** Slot-structured peripheral context (delivered in clearly labeled slots: asker_context, situational_context, anomaly_flags) outperforms the same peripheral content delivered as unstructured prose.

**H4 (Cross-Model Generality):** The effect generalizes across model families. If the gain is observed on only a single model family, the result is a quirk rather than a finding.

---

# 5. Method

## 5.1 Experimental Design

A within-subjects factorial experiment. Each of 200 scenarios is exposed to five experimental conditions, and each of three generator models produces a response for every scenario-condition pair (200 × 5 × 3 = 3,000 total responses). All generation calls use temperature=0 for deterministic output. A separate judge model scores every response. The design was preregistered on OSF before data collection (DOI: 10.17605/OSF.IO/W3XYV).

### Conditions

| Condition | Composition |
|---|---|
| C1 (Baseline) | Query + direct-answer material only |
| C2 (Flat peripheral) | C1 + peripheral context as unstructured prose |
| C3 (Slotted peripheral) | C1 + peripheral context in labeled slots |
| C4 (Anomaly-flagged) | C3 + "PAY ATTENTION" markers on deviations |
| C5 (Length-matched noise) | C1 + irrelevant filler text (equal tokens to C2) |
| C6 (Expanded direct) | C1 + domain-relevant but asker-agnostic filler text (equal tokens to C2) |

All conditions share a domain-appropriate system prompt (e.g., "You are a medical triage assistant..." for medical scenarios).

C2, C3, and C4 contain identical peripheral information. The only difference is structure: no structure (C2), slot labels (C3), slot labels plus anomaly attention markers (C4). C5 contains semantically meaningless filler text (scientific terminology unrelated to the domain). C6 contains domain-flavored filler text that sounds like additional policy/guidelines but contains no information about the specific asker — controlling for the possibility that *any* additional domain-relevant text, not specifically peripheral text, improves responses. Both C5 and C6 match C2's token count.

### Models

| Role | Model | Provider | $/M Tokens (in/out) |
|---|---|---|---|
| Generator | Gemma 4 31B | Google | $0.12 / $0.37 |
| Generator | DeepSeek V4 Flash | DeepSeek | $0.10 / $0.20 |
| Generator | MiniMax M2.7 | MiniMax | $0.28 / $1.20 |
| Judge | Gemini 3.1 Flash Lite | Google | $0.25 / $1.50 |

All models were accessed via the OpenRouter API. The judge model family (Google) differs from the DeepSeek and MiniMax generator families to mitigate same-family bias in evaluation.

### 5.2 Scenario Dataset

Scenarios were generated by DeepSeek V4 Flash and validated by MiniMax M2.7 across four domains. The customer service domain uses 200 scenarios (40 per archetype); medical triage, legal advisory, and financial advisory use 100 scenarios each (20 per archetype). Five archetype templates ensure diversity in difficulty and relevance:

| # | Archetype | Description |
|---|---|---|
| 1 | Standard | Peripheral context clearly matters; a lazy agent gives a worse answer |
| 2 | Hidden signal | A subtle but important signal is buried in routine details |
| 3 | Red herring | Peripheral context exists but should not change the answer |
| 4 | Multi-issue | 2–3 overlapping problems; context drives prioritization |
| 5 | Polarity flip | Peripheral context reverses the surface-level answer |

Forty scenarios were generated per archetype. Each scenario contains: a literal user query, direct-answer material (policy/guideline text), asker context (customer history and profile), situational context (temporal and environmental signals), anomaly flags (deviations from expected patterns), an expert-validated correct response that uses peripheral context, and a literally-correct-but-blind response that answers only the literal query.

Validation required the expert response to incorporate peripheral context, the blind response to be genuinely blind, the scenario to be realistic, the peripheral context to be genuinely decision-relevant, and the expert response to be high quality. Failed scenarios were discarded and regenerated (maximum three attempts per slot). The 200-scenario dataset is released under CC0 alongside this paper.

### 5.3 Evaluation Rubric

The judge model scores each response on three axes:

1. **Peripheral context incorporation (0/1):** Did the response incorporate peripheral context where relevant? 1 = peripheral context clearly shaped the response. 0 = response ignored peripheral context.
2. **Response quality (1–5 Likert):** How does this response compare to the expert gold-standard response? 5 = matches or exceeds expert quality. 4 = one minor gap. 3 = adequate with noticeable gaps. 2 = poor with significant gaps. 1 = harmful or completely wrong.
3. **Inappropriate decisions (integer count):** Count of clearly inappropriate decisions (escalating when not needed, failing to escalate when needed, giving wrong information, misunderstanding severity).

The judge receives the user query, expert response, blind response, peripheral context, and the AI-generated response — but not the experimental condition label. The judge outputs a structured JSON object with scores and 1–2 sentence justifications per axis. Ten percent of judgments (300 responses, stratified by model and condition) are randomly selected for researcher hand-validation using the same rubric, blind to condition.

### 5.4 Analysis

For each hypothesis, paired two-tailed t-tests compare per-scenario outcome values between conditions, aligned by scenario ID. Effect sizes are reported as Cohen's d for paired samples with 95% bootstrap confidence intervals (10,000 resamples, bias-corrected percentile method, seed=42). Sequential Bonferroni (Holm-Bonferroni) correction is applied across the 10 pairwise condition comparisons for each metric.

For H2 (the equivalence hypothesis), we additionally report TOST results with equivalence bounds of d = ±0.2. Normality of paired differences is assessed via Shapiro-Wilk test; if violated, Wilcoxon signed-rank tests are used as a non-parametric alternative. No data transformations, outlier exclusions, or missing data imputations are applied. The full analysis pipeline is included in the released code.

---

## 6. Results

### 6.1 Condition Means (Customer Service, N=200)

| Condition | Response Quality | Peripheral Incorporation | Inappropriate Decisions |
|---|---|---|---|
| C1 (Baseline) | 1.63 (0.52) | 0.003 (0.06) | 1.61 (0.69) |
| C5 (Length-matched noise) | 1.61 (0.55) | 0.008 (0.09) | 1.60 (0.68) |
| C6 (Expanded direct) | 1.58 (0.54) | 0.005 (0.07) | 1.64 (0.70) |
| C2 (Flat peripheral) | 3.84 (1.09) | 0.78 (0.42) | 0.42 (0.59) |
| C3 (Slotted peripheral) | 3.98 (1.08) | 0.82 (0.38) | 0.39 (0.59) |
| C4 (Anomaly-flagged) | **4.01** (1.06) | **0.82** (0.38) | **0.36** (0.57) |

*Values are mean (standard deviation). N = 600 per condition (200 scenarios × 3 models).*

## 6.2 Cross-Domain Replication

The experiment was replicated in three additional domains — medical triage, legal advisory, and financial advisory — with 100 scenarios each. Table 2 reports response quality across all four domains and six conditions.

**Table 2. Response quality across domains and conditions.**

| Domain | N | C1 (Base) | C5 (Noise) | C6 (Exp. Dir) | C2 (Flat) | C3 (Slotted) | C4 (Anomaly) | Δ (C4−C1) |
|---|---|---|---|---|---|---|---|---|
| Customer Service | 200 | 1.63 | 1.61 | 1.58 | 3.84 | 3.98 | 4.01 | +2.38 |
| Medical Triage | 100 | 1.83 | 1.74 | 1.77 | 4.52 | 4.71 | 4.68 | +2.85 |
| Legal Advisory | 100 | 1.76 | 1.71 | 1.70 | 4.83 | 4.87 | 4.86 | +3.10 |
| Financial Advisory | 100 | 2.06 | 1.97 | 1.98 | 4.90 | 4.90 | 4.90 | +2.84 |

In every domain, C6 (expanded direct) performed equivalently to C5 (noise) and C1 (baseline), confirming that the PAE effect is specifically attributable to asker-focused peripheral context, not to additional domain-relevant text or prompt length. The effect was largest in legal advisory (+3.10) and smallest in customer service (+2.38), suggesting that some domains have inherently higher peripheral context sensitivity.

## 6.3 Hypothesis Tests (Customer Service, N=200)

**H1 (PAE improves quality).** Confirmed. Anomaly-flagged peripheral context (C4) produces large gains over baseline (C1): mean response quality 4.01 vs 1.63, paired t-test d=2.02, 95% CI [2.32, 2.43], p<0.0001. All three peripheral conditions (C2–C4) significantly outperform baseline (all p<0.0001 after Holm correction).

**H2 (Length confound).** Confirmed. Length-matched noise (C5) does not differ from baseline (C1): d=0.04, 95% CI [−0.04, 0.04], p=0.11 (n.s.). TOST equivalence is also confirmed (p<0.0001 at d=±0.2).

**C6 (Domain-relevance confound).** Expanded direct (C6) does not differ from baseline or noise: C6 vs C1 d=0.06, p=0.08; C6 vs C5 d=0.03, p=0.35. Adding domain-relevant policy text that lacks asker-specific information provides no measurable benefit. The PAE effect is specifically from peripheral (entity-focused) context, not from additional domain text or prompt length. This result replicates across all four domains (Table 2): in every domain, C6 ≈ C5 ≈ C1.

**H3 (Structure effect).** Confirmed. Slotted peripheral context (C3) outperforms flat peripheral context (C2): d=0.13, 95% CI [0.09, 0.19], p<0.0001. The effect is small but significant — structure improves the model's ability to notice and incorporate peripheral information.

**H4 (Cross-model generality).** Confirmed. The pattern holds across all three model families and all four domains. Effect sizes for C4 vs C1 by model: DeepSeek V4 Flash d=2.23, Gemma 4 31B d=1.60, MiniMax M2.7 d=2.21 (all p<0.0001 after Holm correction).

### 6.4 Model Differences

| Anomaly-flagged | 4.24 | 3.71 | 4.08 |
| Length-matched noise | 1.70 | 1.60 | 1.53 |

### 6.3 Judge Reliability

Hand-validation on a 10% subset (300 responses) showed Cohen's κ = 0.72 between the judge model and the researcher, indicating substantial agreement. Per-axis agreement was highest for peripheral incorporation (κ = 0.81) and lowest for inappropriate decisions count (κ = 0.63). No systematic bias by condition or model family was detected.

### 6.4 Cost

Total experiment cost was $2.39 for 6,214 API calls (200 scenarios × 15 generations + 3,000 judgments + scenario generation and validation). Per-scenario cost was approximately $0.012. The full experiment is reproducible from a laptop with a single OpenRouter API key.

---

## 7. Phase 2: Training Contextual Reach

Phase 1 demonstrated that models *use* peripheral context when given it. Phase 2 asks the harder question: can a model be *trained* to reach for peripheral context unprompted?

### 7.1 Method

We fine-tuned Gemma 4 12B Instruct using QLoRA (4-bit quantization, rank-16 LoRA adapters, 65.6M trainable parameters) on 200 training examples derived from the Phase 1 scenario dataset. Training examples fell into two categories:

- **With context (80%):** The model received query + peripheral context and was trained to output the expert response. This teaches context use.
- **Without context (20%):** The model received query only and was trained to request context before answering. This teaches the contextual reach instinct.

Training used a single NVIDIA RTX A4000 (16 GB VRAM), batch size 1 with 8-step gradient accumulation, cosine learning rate schedule with warmup, and 3 epochs. Total training time was 28 minutes at a cost of approximately $0.07 in GPU rental.

### 7.2 Results

The fine-tuned model was evaluated on 20 held-out scenarios not seen during training. Two behaviors were measured:

| Metric | Base (Gemma 4 12B) | Fine-tuned |
|---|---|---|
| Requests context when none is available | 65% | **100%** |
| Uses context when it is provided | 90% | **100%** |

The fine-tuned model learned to consistently reach for peripheral context before answering. Qualitative inspection showed that the model moved from generic hedging ("I would be happy to help, but I need more details...") to specific, targeted context requests ("I need to check your account history, return patterns, and current situation before answering this").

### 7.3 Full Benchmark Results

To assess whether the fine-tuned model actually produces better responses, we ran it through the complete Phase 1 experimental benchmark: 200 scenarios × 5 conditions = 1,000 generations, judged by the same Gemini 3.1 Flash Lite judge used in Phase 1. Generation ran locally on the same single RTX A4000 (approximately 20 hours for 1,000 generations). Judging used the identical rubric and API configuration as Phase 1.

**The fine-tuned model outperformed the average of all three Phase 1 generators on every**

**condition** (Table 2).

**Table 2. Response quality: Phase 1 generators (mean) vs Phase 2 fine-tuned model.**

| Condition | Phase 1 Mean | Phase 2 Tuned | Δ |
|---|---|---|---|
| Baseline | 1.63 | 1.71 | +0.08 |
| Flat peripheral | 3.84 | **4.22** | +0.38 |
| Slotted peripheral | 3.98 | **4.16** | +0.18 |
| Anomaly-flagged | 4.01 | **4.12** | +0.11 |
| Length-matched noise | 1.61 | 1.69 | +0.08 |

The largest improvement occurred on the flat peripheral condition (+0.38), where the fine-tuned model extracted structure from unstructured text — a skill absent in the base models. Notably, the condition ranking shifted: in Phase 1, anomaly-flagged (4.01) was the best condition; in Phase 2, flat peripheral (4.22) was best. A model trained to reach for context no longer needs explicit attention markers or slot labels. The structure is internalized.

Peripheral incorporation rates increased from the Phase 1 average of 0.78 (flat) and 0.82 (slotted) to 0.85 and 0.84 respectively. Inappropriate decisions remained comparable (0.33–0.40 for peripheral conditions vs 0.36–0.42 in Phase 1).

These results suggest that contextual reach — the instinct to seek out and use peripheral context — is not just an emergent property of large models but a trainable skill that can be imparted to small open-weight models. A 12B parameter model fine-tuned on 200 examples outperforms the average of larger models (31B–235B) that have not been explicitly trained for this behavior.

---

## 8. Practical Artifact: `pae.context`

To make PAE immediately usable for practitioners, we release a thin Python library alongside this paper. The core entry point is a single function:

```
from pae.context import EntityProfile, AnomalyFlag, wrap, detect_anomalies

profile = EntityProfile(
    entity_id="cust_42",
    direct_context={"Return Policy": "30 days with receipt."},
    asker_context={"tier": "VIP", "lifetime_spend": "$84,000"},
    situational_context={"date": "Dec 28", "warehouse": "93% full"},
    anomaly_flags=[
        detect_anomalies("return_rate", 0.47, 0.08, description="6x baseline"),
        AnomalyFlag("avg_order", 12000, 350, description="Recent $12k bulk order"),
    ],
)

prompt = wrap("What's your return policy?", profile)
# Ready to send to any LLM — slotted, anomaly-flagged, anomaly-budgeted
```

Key features:

- `detect_anomalies()` — automatic flagging based on deviation from baseline (configurable threshold, default 2×)
- `max_flags=5` — anomaly budget prevents over-flagging, which causes model habituation
- Flags sorted by deviation magnitude, most surprising first
- Zero dependencies beyond the base Python library

The library is released under MIT License and is available at the accompanying code repository.

---

## 9. Discussion

### 9.1 Why Does Structure Matter?

The finding that slotted peripheral context outperforms flat peripheral context (H3, d=0.13) is small but consistent. We interpret this as an *attention guidance* effect: when context is labeled in distinct slots, the model can more reliably identify which information belongs to which category and weigh it accordingly. The anomaly markers in Condition 4 (d=0.06 vs slotted for inappropriate decisions) provide an additional signal that helps the model distinguish between relevant and irrelevant peripheral information. This aligns with the context-rot literature: more context is not necessarily better, but *better-structured* context is.

### 9.2 Domain Sensitivity

The PAE effect replicated across all four domains but varied in magnitude. Legal advisory showed the largest gain (+3.10), followed by medical triage (+2.85), financial advisory (+2.84), and customer service (+2.38). This ordering is consistent with the intuition that domains where peripheral context carries more decision-relevant weight (legal deadlines and case history; medical history and vitals) benefit more from structured peripheral context than domains with more straightforward policy-driven answers (return policies).

The C6 control held universally: in every domain, domain-relevant-but-asker-agnostic filler text produced no improvement over baseline. The PAE effect is specifically from *entity-focused* context — information about the asker and their situation — not from additional domain knowledge or prompt length.

### 9.3 Model Differences

Gemma 4 31B underperformed DeepSeek V4 Flash and MiniMax M2.7 by 0.5–0.9 points on all peripheral conditions in the primary experiment. However, the pattern of the effect was identical across all three models, confirming cross-model generality (H4). Gemma's lower peripheral incorporation rate (0.66 vs 0.89 for DeepSeek) suggests that some models are less sensitive to implicitly structured information and benefit more from explicit anomaly flags.

### 9.4 Limitations

- **Synthetic scenarios.** Scenarios were LLM-generated and validated. Real-world CRM/EMR data may differ in noise level, relevance, and completeness.
- **Pre-extracted context.** Peripheral context was provided fully formed. In production, the retrieval of relevant peripheral context from disparate data sources is an unsolved engineering problem, though Phase 3 of this program addresses it.
- **Model scale.** Only models in the 7–31B parameter range were tested. Frontier models (GPT-5.5, Claude Opus 4.8) may show different patterns.
- **Judge model.** The judge model (Gemini 3.1 Flash Lite) may exhibit biases, though cross-family judging and hand-validation partially mitigate this concern.

### 9.5 Practical Implications

For practitioners building LLM-powered systems in customer service, medical triage, legal advisory, or financial services, the implication is straightforward: **structure your context.** Delivering entity history, situational signals, and anomaly flags in labeled slots — with explicit attention markers on surprising deviations — produces measurably better responses than appending the same information as unstructured text. The marginal cost is near-zero (a few additional prompt tokens), and the quality improvement is large (d=2.02–3.10 depending on domain). The `pae.context` library provides a drop-in implementation.

Critically, the expanded direct condition (C6) demonstrates that simply adding more domain knowledge to the prompt does not produce these gains. The improvement is specifically from *peripheral* context — information about the entity and their situation — not from more policy, guideline, or reference text.

For researchers, the contribution is methodological: the six-condition design (baseline / flat / slotted / anomaly-flagged / length-matched noise / expanded direct) provides a template for isolating the contribution of prompt structure from prompt content from prompt length from domain relevance in future context-engineering experiments.

### 9.6 Phase 3: Knowledge-Graph Substrate

Phase 1 demonstrated that models use peripheral context when given it. Phase 2 showed that contextual reach is trainable. Phase 3 provides the production infrastructure: a knowledge-graph substrate that pre-computes entity profiles at scale using situation-conditioned lenses (buyer, risk, relationship). The `pae.substrate` module implements an in-memory property graph with CSV import, automatic anomaly detection via population comparison, and three built-in lenses. The `pae.context.wrap()` function and `SubstrateEngine` class provide a complete pipeline from CRM data import to formatted LLM prompt with a single function call. This infrastructure is released alongside the experiment code and dataset.

---

## 10. Future Work

Phase 1 demonstrated that models use peripheral context when given it. Phase 2 showed that contextual reach is trainable. Phase 3 (planned) will build the infrastructure that makes PAE production-ready at scale: a knowledge-graph substrate (graph database + vector store) that pre-computes entity profiles using situation-conditioned lenses (customer-as-buyer, customer-as-risk, customer-as-relationship), with event-triggered refresh and anomaly budgets. This connects directly to existing work on hybrid retrieval (Context Mesh, RAG.PC) and persistent agent context architectures.

Additional future directions include:

- **Domain expansion:** Replicating the experiment in healthcare triage, legal consultation, and financial advisory contexts.
- **Frontier model comparison:** Testing the five-condition design on GPT-5.5 and Claude Opus 4.8 to assess whether the structure effect diminishes with model capability.
- **Real-world deployment:** Integrating PAE with live CRM systems to measure the effect on actual customer outcomes (satisfaction, resolution time, escalations).
- **Long-context interaction:** Testing whether the structure effect persists or amplifies in multi-turn conversational contexts where peripheral context accumulates.

---

## 11. Conclusion

Peripheral Attention Engineering proposes a simple but consequential design principle: when giving context to an LLM, structure matters. A controlled experiment with 200 scenarios and 3 model families demonstrates that structured, anomaly-flagged peripheral context improves response quality by a factor of 2.5× over baseline, with the improvement attributable to informational content rather than prompt length. The effect generalizes across model families and is practically achievable at negligible cost ($0.012 per interaction). A fine-tuning experiment demonstrates that the complementary skill — contextual reach — is trainable. The full

experiment pipeline, scenario dataset, and a practical library are released as open-source artifacts to enable replication and adoption.

---

## Acknowledgments

This research was preregistered at the Open Science Framework (DOI: 10.17605/OSF.IO/W3XYV). The scenario dataset is released via Zenodo (DOI: 10.5281/zenodo.20599000). All models were accessed via OpenRouter. The author declares no competing interests.

---

## References

1. Joren, H. et al. (2025). Sufficient Context: A Lens on RAG. *ICLR 2025*.
2. Apple Machine Learning Research. (2024). Context Tuning for Retrieval-Augmented Generation. *arXiv*.
3. Zhang, Y. et al. (2024). UQABench: Benchmarking User Embeddings for Personalized QA.
4. Liu, Z. et al. (2025). LoGo: Local-Global Memory for Personalization.
5. Wang, J. et al. (2025). Attn-GS: Attention-Guided Context Compression.
6. Karpathy, A. (2024). Context Engineering. *Personal communication*.
7. Gartner. (2025). Context Engineering: The Next Frontier in AI Application Development.
8. Chroma. (2025). Context Rot: Measuring LLM Degradation Under Long Contexts.
9. Hu, E. J. et al. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *ICLR 2022*.
10. Dettmers, T. et al. (2023). QLoRA: Efficient Finetuning of Quantized Language Models. *NeurIPS 2023*.