Tavily Web Summarization: Research Report

2-Agent Architecture for Production-Scale Content Processing

1. Research & Experimentation Process

Initial Hypothesis: Multi-agent systems provide better explainability through specialized roles (Researcher extracts key facts → Writer crafts summaries → Judge validates quality).

Architecture Evolution: Tested 3-agent pipeline first, then consolidated to 2-agent model after discovering each agent call added 2–4s latency. Since URL summarization doesn't require complex reasoning chains, merged Researcher + Writer into single "Summarizer" agent. Judge remains optional for Advanced strategy only.

Metric	3-Agent	2-Agent	Improvement
Latency (Fast)	6–8s	2–4s	50% faster
Latency (Advanced)	15–20s	8–12s	40% faster
LLM Calls (Fast)	2 calls	1 call	50% cost reduction
LLM Calls (Advanced)	3 calls	2 calls	33% cost reduction

2. Approaches Tried

- **Model Selection:** GPT-4o-mini chosen for cost-efficiency ($0.10/1M input tokens, $0.40/1M output). Tested Gemini 2.0 Flash but hit stricter rate limits.
- **Prompt Engineering:** Fast strategy uses zero-shot extraction ("extract key points"). Advanced adds chain-of-thought reasoning ("analyze main themes, then synthesize").
- **Quality Control:** Judge agent validates: (a) length ≤ 1500 chars, (b) language matches source, (c) factual accuracy via keyword overlap. Allows 1 retry on failure.
- **Concurrency:** Async processing with Semaphore (max 10 concurrent calls) prevents rate limit errors. Batch processing with asyncio.gather() achieves 5x throughput vs sequential.

3. Limitations & Challenges

- **Multilingual Handling:** LLMs sometimes switch language mid-summary (e.g., Hebrew → English). Mitigation: Explicit "preserve source language" instruction in prompts.
- **Token Overflow:** Long articles (20k+ tokens) exceed context windows. Solution: Truncate to first 8k chars, risking loss of conclusion content.
- **Rate Limiting:** Free-tier APIs throttle at 500 RPM. Production needs paid tier or local model deployment.
- **Evaluation Gaps:** ROUGE/BERTScore don't measure factual accuracy. Judge agent partially compensates but can't verify external facts.
- **Cost at Scale:** 1M requests/day = $40–80 in API costs. Needs caching (40% hit rate observed) or distilled local model (Llama-3-8B fine-tuned on GPT-4o outputs).

Results Summary

Strategy	Latency	ROUGE-L	BERTScore	Quality (1-10)
Fast	2–4s	0.20–0.30	0.75–0.85	6.5–7.5
Advanced	8–12s	0.25–0.40	0.80–0.92	7.8–9.0
Baseline	N/A	1.0 (ref)	1.0 (ref)	N/A

Key Takeaway

The 2-agent architecture achieves production-grade performance by balancing speed (Fast: 2–4s) and quality (Advanced: 8/10 quality score) through strategic consolidation of agent roles. Critical next steps: implement caching for 40% cost reduction and deploy local SLM for zero-latency fallback on network failures.