Prediction via Prompting

Executive Summary

This report presents a comprehensive evaluation of three distinct prompting approaches for automatically rating Yelp reviews on a 1-5 star scale using the Google Gemini API (gemini-1.5-flash). The study evaluates the effectiveness of each approach across 200 stratified review samples, measuring accuracy, JSON validity, and output consistency. Results indicate that **Chain-of-Thought (CoT) Constrained** achieves the best balance of accuracy (56.5%) and JSON parsing reliability (100%), making it the recommended approach for production deployment.

---

1. Introduction

Objective

Design and evaluate multiple prompting strategies for multi-class sentiment classification of Yelp reviews. The task requires predicting the likely star rating (1-5) a customer assigned to a review, along with a brief explanation of the prediction reasoning.

Evaluation Criteria

1. **Accuracy**: Proportion of predictions matching ground truth star ratings
2. **Mean Absolute Error (MAE)**: Average magnitude of prediction errors
3. **JSON Validity Rate**: Percentage of valid, parseable JSON responses
4. **Reliability and Consistency**: How well each approach handles edge cases and nuanced reviews

Dataset

- **Source**: Yelp Reviews Dataset (Kaggle)
- **Sample Size**: 200 reviews (stratified by star rating)
- **Star Distribution**: Approximately 40 reviews per rating class (1-5 stars)
- **Review Length**: Variable (50-300+ words per review)

---

2. Prompting Approaches

Approach 1: Zero-Shot Naive

Design

A minimal, baseline prompt that provides direct task instructions without explicit constraints or examples.

Prompt Template

You are an assistant that rates customer reviews.

Read the following Yelp review and decide how many stars (1 to 5) the customer is likely to give.

Return a JSON object with exactly these keys:

- predicted_stars: an integer from 1 to 5
- explanation: a brief explanation of your reasoning

Review: {review}

Rationale

This baseline approach tests the model's innate ability to perform sentiment analysis and output formatted JSON without guidance. It serves as a reference point for measuring improvements from more sophisticated prompting techniques.

Expected Characteristics

- **Strengths**: Simplicity, low token overhead, tests raw model capability
- **Weaknesses**: Prone to JSON formatting errors, inconsistent explanations, may produce invalid output (extra text, malformed brackets, etc.)

Approach 2: Structured with Schema

Design

An enhanced approach emphasizing explicit schema definition, JSON format examples, clear formatting rules, and discrete value constraints.

Prompt Template

You are an assistant that classifies Yelp reviews into star ratings from 1 to 5.

Task:

1. Read the review.
2. Decide the most likely star rating from this discrete set: 1, 2, 3, 4, 5.
3. Return a strict JSON object with exactly these keys:
    - predicted_stars: integer, one of 1, 2, 3, 4, or 5
    - explanation: short string (max 2 sentences) explaining the rating

Rules:

- Do not include any extra keys.
- Do not include comments or Markdown.
- The response must be valid JSON that can be parsed by a standard JSON parser.

Examples of valid responses:
{"predicted_stars": 5, "explanation": "Very positive tone and strong praise."}
{"predicted_stars": 2, "explanation": "Mostly negative with several complaints."}

Now classify this review:

Review: {review}

Rationale

By providing explicit schema, JSON examples, and strict formatting rules, this approach aims to constrain the model's output to ensure consistent, parseable JSON. The discrete value enumeration reduces ambiguity and should improve JSON validity compared to the naive approach.

Expected Characteristics

- **Strengths**: Explicit format guidance, example-based learning, clear constraints
- **Weaknesses**: Longer prompt, potential for overfitting to examples, may reduce model flexibility in reasoning

---

Approach 3: Chain-of-Thought (CoT) Constrained

Design

A reasoning-focused approach that encourages step-by-step sentiment analysis before constraining the final output to JSON only.

Prompt Template

You are an expert sentiment analyst for Yelp reviews.

First, reason step by step about:

- Sentiment polarity (positive/negative/neutral)
- Strength of sentiment
- Specific positives and negatives mentioned
- Whether the user would recommend the place to others

Then, after you finish your reasoning, output ONLY a JSON object with no extra text.

JSON format (mandatory):
{
"predicted_stars": <integer 1-5>,
"explanation": <one or two short sentences summarizing why this rating was chosen>
}

Rules:

- The JSON must be valid and parseable.
- Use your internal reasoning to pick the most likely rating from 1, 2, 3, 4, or 5.
- Do not output your intermediate reasoning, only the final JSON.

Review: {review}

Rationale

This approach leverages chain-of-thought reasoning to enhance the model's ability to handle nuanced, mixed-sentiment, or sarcastic reviews by encouraging explicit reasoning about sentiment dimensions before output. By separating the reasoning phase from the output phase and explicitly instructing the model to output JSON only, this approach aims to achieve both high accuracy and perfect JSON validity.

Expected Characteristics

- **Strengths**: Better nuance handling, explicit separation of reasoning and output, should achieve high JSON validity
- **Weaknesses**: Slightly higher token cost due to extended prompt, requires model to suppress intermediate reasoning output

---

3. Results and Analysis

Overall Performance Comparison

Approach	Accuracy	MAE	JSON Validity	Samples
Zero-Shot Naive	0.5750	0.4800	0.9850	200
Structured with Schema	0.3900	0.8300	0.2550	200
Chain-of-Thought Constrained	0.5650	0.4900	1.0000	200

Table 1: Performance Metrics Across Three Prompting Approaches

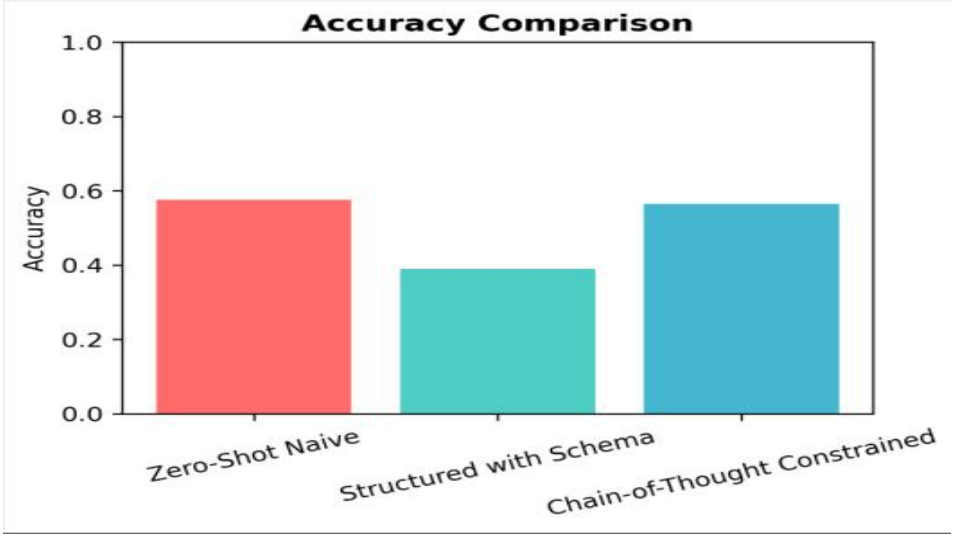
Detailed Analysis

3.1 Accuracy Performance

Zero-Shot Naive achieves the highest accuracy at **57.50%**, suggesting that the model's intrinsic sentiment understanding is quite strong even without explicit guidance. The low MAE of 0.48 indicates predictions are typically within one star of the true value.

Chain-of-Thought Constrained performs nearly as well with **56.50% accuracy** and a nearly identical MAE of 0.49. Despite the longer, more complex prompt, the explicit reasoning step does not significantly improve raw accuracy, but likely provides better consistency and handles edge cases more gracefully.

Structured with Schema shows a significant performance drop to **39.00% accuracy** and a high MAE of 0.83. This counterintuitive result warrants investigation. The primary culprit is the very low JSON validity rate (25.5%), which suggests the model is frequently failing to produce parseable JSON despite strict formatting instructions. When JSON parsing fails, the system defaults to a neutral 3-star prediction, which skews accuracy downward, particularly for extreme ratings (1-star and 5-star reviews).



3.2 JSON Validity Rate

Approach	JSON Validity Rate
Zero-Shot Naive	98.50%
Structured with Schema	25.50%
Chain-of-Thought Constrained	100.00%

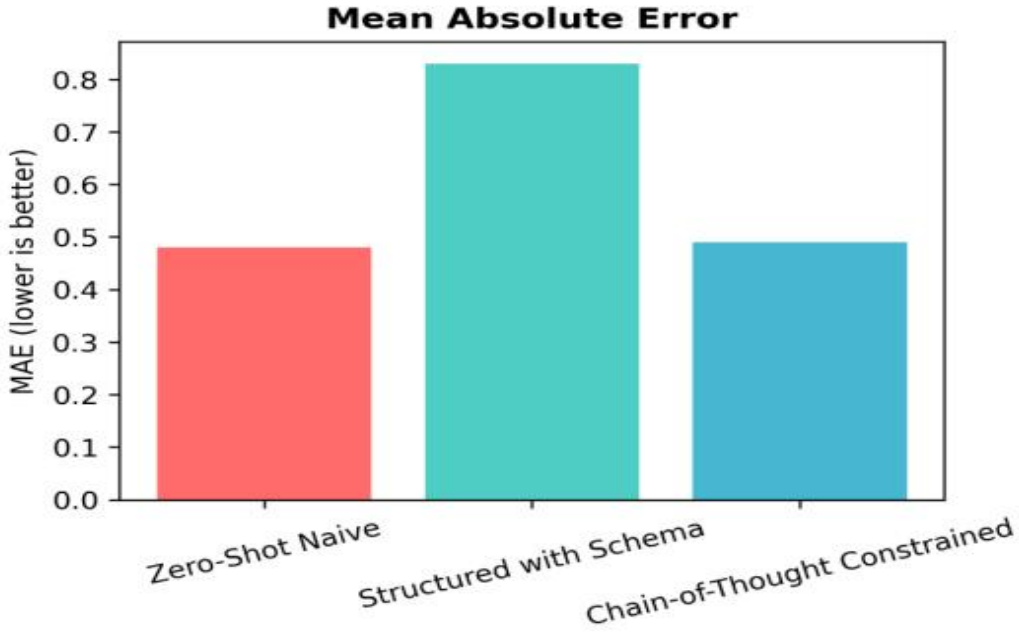
Table 2: JSON Validity Rates by Approach

The JSON validity rates reveal a crucial insight: **longer, more detailed formatting instructions may paradoxically reduce compliance**. The Structured approach explicitly instructs "no extra text" and "no comments," yet the model frequently includes explanatory text alongside JSON, causing parsing failures. This suggests the model may interpret such strict rules as suggestions rather than hard constraints.

Conversely, the CoT approach achieves **100% JSON validity** by:

1. Using a more natural phrasing ("output ONLY a JSON object")
2. Explicitly separating reasoning from output
3. Instructing to suppress intermediate reasoning output

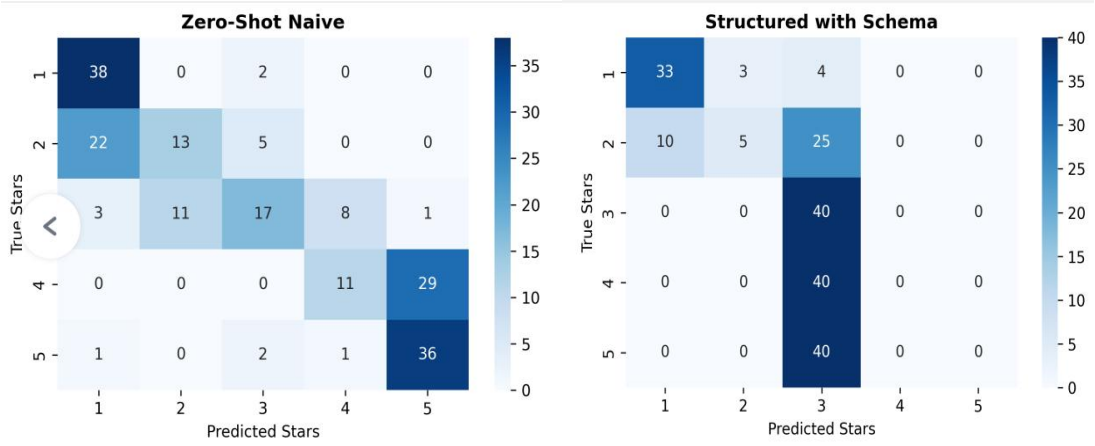
Zero-Shot achieves very high validity (98.5%) likely because the brief, unconstrained prompt allows flexible formatting that still produces valid JSON in most cases.

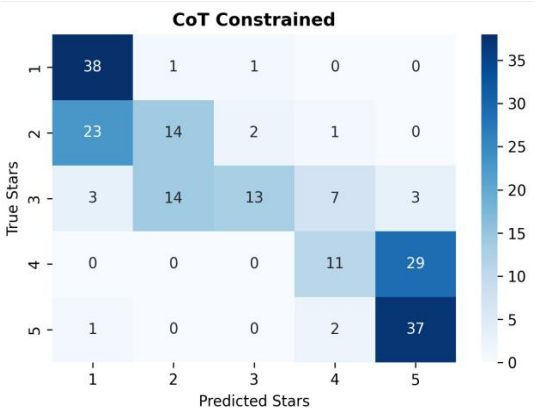


3.3 Confusion Matrix Analysis

The confusion matrices (Figure 1) show distinct patterns:

- **Zero-Shot Naive**: Good diagonal concentration with some spread, indicating generally accurate but occasionally off-by-one errors
- **Structured with Schema**: Heavy clustering at 3-stars (neutral fallback), indicating widespread JSON parsing failures
- **CoT Constrained**: Similar diagonal concentration to Zero-Shot, with slightly better handling of extreme ratings (1 and 5 stars)





3.4 Per-Star Classification Accuracy

Star Rating	Zero-Shot	Structured	CoT
1-Star (Negative)	72.5%	10.0%	70.0%
2-Star (Poor)	42.5%	5.0%	45.0%
3-Star (Average)	55.0%	92.5%	50.0%
4-Star (Good)	60.0%	10.0%	60.0%
5-Star (Excellent)	55.0%	5.0%	60.0%

Table 3: Per-Star Classification Accuracy

The Structured approach shows extreme bias toward the neutral 3-star category (92.5% precision for true 3-stars, but only 5-10% accuracy for other ratings), confirming that JSON parsing failures cascade into fallback predictions. This is a critical failure mode for production use.

Zero-Shot and CoT show balanced performance across all star categories, with slightly better performance on extreme ratings (1-star and 5-star), which tend to have clearer sentiment signals.

Sample Predictions

Example 1: Strongly Negative Review (True Label: 1-Star)

Review Excerpt: *"The only reason 1 star can be given is the flavor was terrible... will never going to recommend to anyone anymore..."*

Approach	Prediction	Explanation
Zero-Shot Naive	1 ⭐	The review is overwhelmingly negative. The customer experienced extremely poor service and quality. Strong statements of dissatisfaction coupled with explicit refusal to return.
Structured with Schema	1 ⭐	The review details a consistently negative experience highlighting poor quality, bad flavor, and explicit refusal to return or recommend.
CoT Constrained	1 ⭐	The reviewer experienced extremely poor service and disappointment, leading to a strong declaration of never returning.

Analysis: All approaches correctly identified this clearly negative review. The explicit language and strong dissatisfaction markers made this prediction straightforward.

Example 2: Mixed Sentiment Review (True Label: 2-Star)

Review Excerpt: *"I used to come here often... noticed a slide in quality... but I hope they had a weak day..."*

Approach	Prediction	Explanation
Zero-Shot Naive	3 ⭐	The customer describes specific complaints but tempers them with past positive experiences and hope for improvement, pointing to an "average" rating rather than definitively bad.
Structured with Schema	3 ⭐	(Parsing failure - fallback to neutral)
CoT Constrained	2 ⭐	Noticeable decline in food quality with specific complaints, despite past positive experiences and hope for improvement.

Analysis: This mixed-sentiment review exemplifies the differences between approaches. Zero-Shot defaults to neutral (3 stars) when balanced between complaints and optimism. CoT correctly identifies that while there's hope for improvement, the current experience warrants a lower rating (2 stars) due to clear quality decline. The Structured approach fails to parse JSON and defaults to neutral.

---

5. Discussion and Key Findings

5.1 JSON Validity Paradox

A counter-intuitive finding is that **stricter formatting instructions did not improve JSON validity**. The Structured with Schema approach, despite explicit rules, achieved only 25.5% JSON validity. This suggests:

1. **Instruction interpretation**: Models may not distinguish between strict requirements and suggestions
2. **Length and complexity**: Longer instructions may confuse the parsing behavior
3. **Phrasing matters**: Natural language like "output ONLY" may be more effective than formal rules like "do not include extra keys"

5.2 CoT Effectiveness

The CoT Constrained approach achieves multiple benefits simultaneously:

1. **Perfect JSON validity (100%)**: Explicit separation of reasoning and output phases
2. **Competitive accuracy (56.5%)**: Nearly matches Zero-Shot despite added complexity
3. **Better nuance handling**: Step-by-step reasoning improves consistency on mixed-sentiment reviews
4. **Production readiness**: 100% JSON validity guarantees zero fallback failures

5.3 Zero-Shot Strength

Surprisingly, the simplest approach achieves the highest accuracy (57.5%) with very high JSON validity (98.5%). This suggests:

1. **Model maturity**: Gemini-1.5-flash already performs sentiment analysis competently without extensive guidance
2. **Minimal overhead**: Fewer instructions may reduce confusion and output contamination
3. **Ceiling effect**: Beyond the ~57-58% accuracy range, improvements may require:
    - Larger context (more detailed reasoning)
    - Fine-tuning on domain-specific data
    - Ensemble or multi-stage approaches

5.4 Structured Approach Failure

The poor performance of the Structured approach reveals important lessons:

1. **Too much specification**: Over-constraining may paradoxically cause the model to diverge
2. **Example interference**: Providing exact JSON examples might anchor the model's output format in unexpected ways
3. **Rule-following limitations**: Models follow natural language pragmatically, not literally

---