Beyond Instruction Optimization: Multi-Agent Error-Driven Class Description Refinement for LLM-Based Classification

Hamvir Dev[*] **Shivam Ratnakant Mhaskar**[*] **Sasanka Vutla**[†] **Anup Pattnaik**[†]
Observe.AI, India
{hamvir.dev, shivam.mhaskar, sasanka.vutla, anup.pattnaik}@observe.ai

Abstract

Large Language Models (LLMs) have demonstrated considerable efficacy in classification tasks, yet their performance depends on two critical prompt components: **Task Instructions** (HOW to classify) and **Class Descriptions** (WHAT defines each class). While prompt engineering research has extensively explored instruction optimization, class descriptions have received comparatively less attention, often being treated as fixed inputs or simple label names. This represents a critical gap for real-world classification tasks, particularly in contact center domains, where labels often suffer from ambiguous boundaries, overlapping definitions, and incomplete coverage of possible cases—substantially limiting accuracy regardless of instruction quality.

We propose a multi-agent framework for iteratively refining class descriptions based on classification errors. By analyzing misclassified instances, language agents automatically generate improved descriptions that better capture class distinctions and resolve ambiguities. Empirical evaluation across contact center and public benchmark datasets demonstrates up to **20.71%** accuracy improvements over static class descriptions, addressing an orthogonal dimension to existing instruction optimization techniques.

1 Introduction

Contact centers record millions of customer interaction transcripts daily, requiring accurate classification into business-relevant categories for downstream applications, including quality assurance, experience analytics, compliance monitoring, and strategic insights.

Recent advances in Large Language Models (LLMs) enable zero-shot and few-shot classification (Wang et al., 2023), (Brown et al., 2020), allowing rapid adaptation to evolving taxonomies without extensive labeled data or costly retraining. However, contact center taxonomies with hundreds of classes pose a practical challenge: fitting all class descriptions within a single prompt exceeds LLM context constraints and degrades performance. To address this, retrieval-augmented approaches (Vandemoortele et al., 2025), (Milios et al., 2023), (Pattnaik et al., 2025) dynamically select relevant candidate classes for each instance, enabling efficient high-cardinality classification. Yet these advances focus primarily on *how* to efficiently scale classification, while a critical bottleneck remains under-explored: **the quality of class descriptions themselves**—the *what* that defines each category.

Production contact center taxonomies are typically defined by business stakeholders without machine learning expertise, resulting in class descriptions that prioritize operational semantics over discriminative clarity. Systematic analysis reveals three failure modes: **vague boundaries** without explicit decision criteria, **semantic overlap** among related categories, and **missing contrastive features** that distinguish similar classes (Table 8). This specification gap fundamentally limits classification performance regardless of instruction quality, as task-level instructions cannot compensate for inadequate class-level definitions.

We propose a multi-agent framework for iterative, error-driven refinement of class descriptions using validation data. The framework identifies high-confusion class pairs through error analysis, samples their misclassified instances, and employs LLM agents to generate revised descriptions that explicitly **contrast confused categories**. By presenting both positive and negative examples—what should and should not belong to each class—the agent produces mutually exclusive and

[*]Equal contribution as first authors
[†]Equal contribution as second authors

collectively exhaustive (MECE) descriptions that resolve boundary ambiguities and reduce systematic confusion patterns.

Our method is complementary to existing prompt optimization techniques (Pryzant et al., 2023a), (Ye et al., 2024a), (Wang et al., 2024), (Do et al., 2024), (Sinha et al., 2024), operating on an orthogonal dimension of the classification prompt. While instruction optimization focuses on improving task-level reasoning (the *how*), our description refinement addresses class-level semantic clarity and discriminative power (the *what*). This orthogonality suggests that gains from both approaches could potentially compound, though we leave empirical investigation of such combinations to future work.

Our contributions are threefold:

- **Error-driven description refinement framework:** We introduce a multi-agent system that analyzes confusion patterns to iteratively improve class descriptions—addressing the under-explored problem of class definition quality rather than task instruction optimization.
- **Performance ceiling of instruction optimization:** We show that model scaling and prompt optimization yield minimal gains on confused classes (Section 4), establishing that description quality represents a fundamental bottleneck requiring targeted refinement.
- **Substantial and generalizable improvements:** We achieve up to 20.71% accuracy gains over static descriptions across eight datasets spanning diverse domains, cardinalities (35-160 classes), and text lengths (13-2000 tokens), demonstrating broad applicability beyond contact center scenarios.

2 Problem Formulation

Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ of input texts x_i and labels $y_i \in \mathcal{Y} = \{y_1, \ldots, y_L\}$, we model an LLM-based classifier as a function of both task instructions and class descriptions:

$$f(x; \mathcal{I}, \mathbf{C})$$

where $\mathcal{I}$ represents the task instruction (HOW to classify) and $\mathbf{C} = \{c_1, \ldots, c_L\}$ denotes the natural language descriptions defining each class (WHAT each class represents).

Given a fixed instruction $\mathcal{I}$ and initial descriptions $\mathbf{C}^{(0)}$, our objective is to find optimal descriptions $\mathbf{C}^*$ that minimize classification error on a validation subset $\mathcal{D}_{val} \subset \mathcal{D}$:

$$\mathbf{C}^* = \arg\min_{\mathbf{C}} \sum_{(x_i, y_i) \in \mathcal{D}_{val}} \mathbb{1}[f(x_i; \mathcal{I}, \mathbf{C}) \neq y_i]$$

We perform this optimization iteratively through error-driven refinement. At each iteration t, we: (1) identify a high-confusion class y_j based on validation error, (2) sample its misclassified instances $\mathcal{E}_j^{(t)} = \text{FP}_j^{(t)} \cup \text{FN}_j^{(t)}$, where $\text{FP}_j^{(t)}$ are instances incorrectly predicted as y_j and $\text{FN}_j^{(t)}$ are instances of y_j missed by the classifier, and (3) employ a multi-agent system $\mathcal{A}$ to generate refined description:

$$c_j^{(t+1)} = \mathcal{A}(c_j^{(t)}, \text{FP}_j^{(t)}, \text{FN}_j^{(t)})$$

where $\mathbf{C}^{(t+1)} = \mathbf{C}^{(t)} \setminus \{c_j^{(t)}\} \cup \{c_j^{(t+1)}\}$ represents the updated description set. The multi-agent system is designed to generate descriptions that minimize confusion with related classes by explicitly contrasting positive and negative boundary cases.

3 Data Sources

Table 3 presents four proprietary contact center collections with business-authored taxonomies and four public benchmarks spanning intent detection, topic classification, and article categorization. This mix assesses performance across production scenarios with real-world label quality issues and standardized evaluation settings. For DBpedia (Lehmann et al., 2015), we use the Level-2 (L2) taxonomy consisting of 70 classes, which we refer to as DBPedia-70. For Web of Science (Kowsari et al., 2017), we use the 35-class version of the dataset.

Contact center datasets include business-authored class descriptions with varying ambiguity and overlap, derived from English audio transcripts and webchat interactions. For public benchmarks providing only class names, we generate initial descriptions following Pattnaik et al. (2025).

We use a subset of training data as $\mathcal{D}_{val}$ for description refinement, with held-out test sets for final evaluation.

4 Limitations of Static Descriptions

Can scaling model capacity or optimizing task instructions resolve errors from ambiguous class definitions? We investigate this on DBpedia-70, tracking 12 high-confusion classes where ≥70% of baseline errors stem from a single competing class (e.g., "WinterSportPlayer" vs. "Athlete", "Stream" vs. "BodyOfWater").

Model Scaling. Table 1 shows that while overall macro-F1 improves substantially (LLaMA (Grattafiori et al., 2024): +36.8%), confused classes show minimal improvement or degradation (Nova (AGI et al., 2025): -10.8%, GPT-OSS (OpenAI et al., 2025): -4.4%, LLaMA: +1.9%, GPT: +1.6%). All large models converge to similar confused class performance (0.31-0.50 F1) despite varying overall capabilities (0.83-0.90 F1).

Prompting Strategies. Table 2 shows strategies improving overall F1 by +10.6% yield only +3.6% gains on confused classes. PE2 optimization (Ye et al., 2024a) achieves +10.6% overall but +2.8% on confused classes; chain-of-thought (Wei et al., 2022) provides +6.2% overall but only +0.6% on confused classes.

Implications. These results reveal a structural bottleneck: when errors arise from inadequate class specifications, improving inference mechanisms yields limited gains. The consistent ceiling indicates class descriptions themselves require refinement based on observed confusion patterns.

Family	Model	F1	F1 (conf.)	Δ F1	Δ F1 (conf.)
Nova	nova-lite	0.7644	0.4044	-	-
	nova-pro	0.8256	0.3608	+8.0%	-10.8%
GPT-OSS	20B	0.8516	0.4017	-	-
	120B	0.8875	0.3841	+4.2%	-4.4%
LLaMA	8B	0.6214	0.3048	-	-
	70B	0.8498	0.3105	+36.8%	+1.9%
GPT	gpt-4o-mini	0.8247	0.4954	-	-
	gpt-4o	0.8984	0.5033	+8.9%	+1.6%

Table 1: Performance across model families and parameter scales on DBPedia-70. F1 denotes Macro F1; F1 (conf.) denotes Macro F1 computed over confused classes only; and Δ denotes relative improvement between smaller and larger variants.

5 Proposed Methodology

We propose an iterative framework that refines class descriptions by analyzing classification errors. The approach identifies low F1-score classes, samples their misclassified instances, and employs multi-agent systems to generate discriminative improvements (Figure 1).

Prompt Type	F1	F1 (conf.)	Δ F1	Δ F1 (conf.)
Regular Prompt	0.7644	0.4044	-	-
Few Shot	0.8254	0.4189	+8.0%	+3.6%
Chain of Thought	0.8121	0.4068	+6.2%	+0.6%
PE2 Optimized	0.8452	0.4156	+10.6%	+2.8%

Table 2: Performance of different prompting strategies using nova-lite. F1 denotes Macro F1; F1 (conf.) denotes Macro F1 computed over confused classes only; and Δ denotes relative improvement over the regular prompt baseline.

5.1 Framework Overview

Given labels $\mathcal{Y} = \{y_1, \ldots, y_L\}$ with initial descriptions $\mathcal{C}^{(0)}$, our framework iteratively refines descriptions using validation data $\mathcal{D}_{val}$. Each iteration refines one high-confusion class through four stages: (1) *Error Analysis and Sampling*—identify the most confused class and sample misclassifications, (2) *Hint Generation*—extract contrastive patterns from errors, (3) *Description Refinement*—generate improved descriptions via LLM agents, and (4) *Validation and Stopping Criteria*—verify confusion reduction. The process continues until error thresholds are met or T_{max} iterations. We detail each stage in the following subsections.

5.2 Error Analysis and Sampling

At each iteration t, we classify all instances in $\mathcal{D}_{val}$ using the current descriptions $\mathcal{C}^{(t)}$. For each input x_i, the LLM produces a predicted label $\hat{y}_i$. We compute class-wise metrics (precision, recall, F1) and construct a confusion matrix to identify systematic error patterns.

We select the candidate class y_j with the lowest F1-score for refinement. We apply Maximal Marginal Relevance (MMR) sampling (Carbonell and Goldstein, 1998) on summarized misclassified texts of class y_j to select up to N diverse instances, maximizing coverage of distinct confusion patterns.

To identify which competing classes cause the most confusion, we analyze the distribution of misclassifications. If errors involving a specific class y_k account for more than 30% of y_j's total errors, we mark (y_j, y_k) as a high-confusion pair requiring explicit contrastive analysis in subsequent stages.

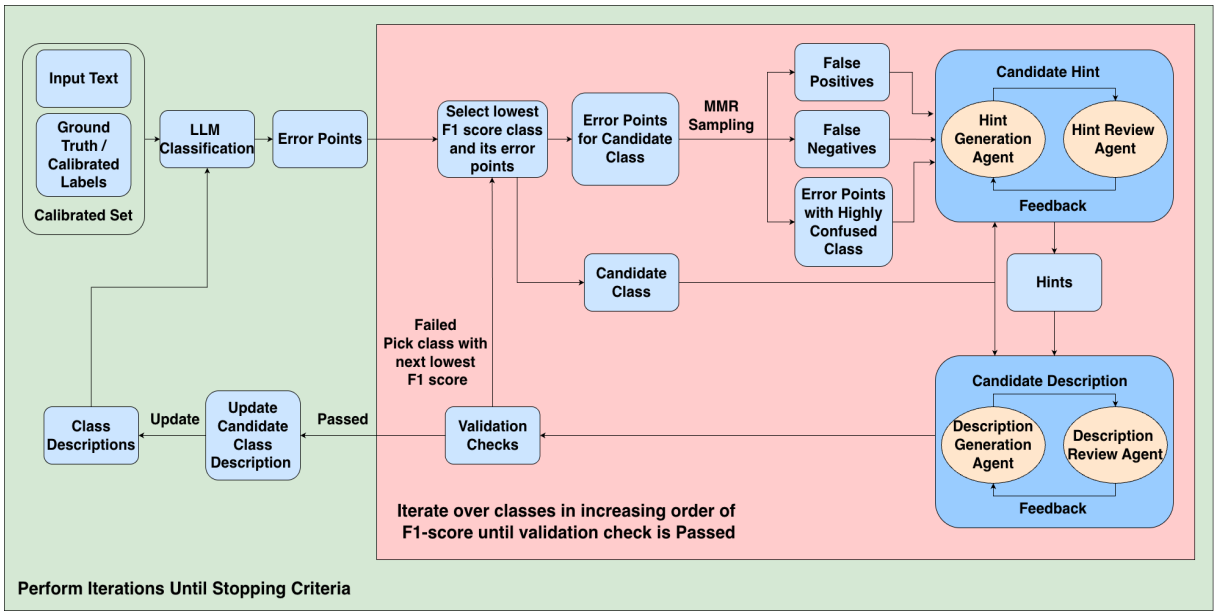

Figure 1: Error-driven multi-agent description refinement pipeline. At each iteration: (1) classify validation set, select lowest-F1 class, and sample diverse errors via MMR (false positives, false negatives, confused pairs), (2) generate contrastive hints using Hint Generator and Reviewer agents, (3) synthesize refined descriptions using Description Synthesis and Reviewer agents, and (4) validate via error resolution and stability checks. Validated refinements update the description and advance to the next iteration; failed refinements trigger selection of the next-lowest F1 class. The dual-agent architecture with review cycles ensures systematic quality control.

5.3 Hint Generation

We employ a two-agent system using specialized LLMs to extract actionable insights from sampled errors (prompt details in Appendix E.2). For each error instance $(x, \hat{y}, y)$, a **Generator Agent** produces one hint based on error type: false positives (FPs) yield exclusion criteria, false negatives (FNs) yield inclusion criteria, and high-confusion pairs yield contrastive distinctions between y_j and y_k. A **Reviewer Agent** evaluates each hint for specificity, actionability, and correctness, either approving it or requesting revision with feedback. This generate-review cycle iterates until approval (maximum 3 rounds).

From N sampled FPs and FNs for class y_j, this process yields a set of approved hints $\mathcal{H}_j^{(t)}$ that are aggregated and synthesized into an improved description in the next stage (representative examples in Table 11).

5.4 Description Refinement

Given the approved hint set $\mathcal{H}_j^{(t)}$ for class y_j, we again employ a two-agent system to synthesize these insights into an improved description (prompt specifications in Appendix E.3). Table 7 shows few examples of initial and final descriptions for confused classes from DBPedia-70 dataset.

Description Synthesis Agent. The Synthesis Agent receives the current description $c_j^{(t)}$ and the complete hint set $\mathcal{H}_j^{(t)}$ (containing FP exclusions, FN inclusions, and contrastive distinctions for high-confusion pairs). The agent first aggregates and consolidates the hints, removing redundancies while preserving specific details, then generates 1-3 concise sentences capturing new discriminative information not already present in $c_j^{(t)}$. The agent is explicitly instructed to avoid redundancy with existing content and prioritize contrastive differentiators when high-confusion pairs exist.

Description Reviewer Agent. The Reviewer verifies that proposed additions: (1) introduce new information absent from $c_j^{(t)}$, and (2) accurately address the error-derived hints. The reviewer outputs *APPROVE* or *REVISE* with specific feedback. This cycle continues until approval or 3 iterations. Upon approval, we update: $c_j^{(t+1)} = c_j^{(t)} \oplus \Delta c_j$, where Δc_j denotes the approved additions.

5.5 Validation and Stopping Criteria

Before accepting $c_j^{(t+1)}$, we validate that refinements improve classification without regressions

through two checks: (1) **Error Resolution**—at least 20% of sampled errors are resolved when reclassified with $c_j^{(t+1)}$, and (2) **Performance Stability**—class-wise metrics for y_j and overall accuracy on $\mathcal{D}_{val}$ do not degrade by more than 5% and 0.5%, respectively.

If both checks pass, we accept $c_j^{(t+1)}$ and proceed to the next iteration and marking y_j as non-refinable. Failed validations result in discarding $c_j^{(t+1)}$ and moving to the next class with lowest F1-score. The framework terminates when no class passes validation in a full cycle, or the iteration budget T_{max} is reached.

5.6 Implementation Details

Our framework is model-agnostic and compatible with any instruction-following LLM. We employ separate models for classification and the agentic components, allowing practitioners to select models based on cost-performance tradeoffs for each role. Complete prompts for all agents are provided in Appendix E.2 and E.3, with detailed algorithms in Appendix F.

The framework requires configuration of several hyperparameters: sample size (N), confusion threshold for identifying high-confusion pairs (τ_{conf}), maximum review-revision cycles (R_{max}), minimum error resolution rate ($\tau_{resolve}$), stability tolerances for class-wise (τ_{class}) and global metrics (τ_{global}), and maximum refinement iterations (T_{max}). These parameters control the tradeoff between refinement thoroughness and computational cost. Appendix Section C covers all hyperparameters used in our experiments.

6 Evaluation

6.1 Experimental Setup

We evaluate on the eight datasets described in Section 3, allocating 20% of training data as $\mathcal{D}_{val}$ for description refinement and using held-out test sets for evaluation. For each dataset, we report performance before and after iterative refinement of class descriptions.

6.2 Results

Table 3 shows consistent improvements across all configurations, ranging from +3.43 to +20.71 percentage. Nova-lite achieves the largest gains (public benchmarks: +7.45 on average, contact center: +11.67 on average), followed by gpt-4o-mini (+5.76, +9.14) and claude-4-5-haiku (+4.42, +6.33). The larger improvements observed for smaller models suggest that description quality is more critical when model capacity is limited.

Figure 2 shows the accuracy improvement of different datasets using nova-lite over iterations. We fixed the number of iterations to 20 to reduce computational cost. In the figure, Web of Science stops at the 10th iteration due to our early stopping criterion: once no improvement is observed across classes, training halts.

Confused classes ($\geq$ 70% errors from single competing class) show substantially larger improvements: +12.45 to +85.36 Macro F1 points across datasets (Table 6). This validates Section 4 findings—targeted description refinement resolves the bottleneck that model scaling and prompt optimization cannot address.

The multi-agent system described in this work is **deployed in production for 10+ customers**, improving classification accuracy across various downstream tasks.

7 Ablation Studies

7.1 Effect of Error Sample Size

We evaluate error sample size N on DBpedia-70, varying from 5 to 30 instances. Table 10 shows accuracy improves consistently as N increases to 20 across all models, with gains of +6-4 percentage points from N=5 to N=20. Beyond N=20, marginal gains diminish ($\leq$ 0.3 points at N=25) while computational cost scales linearly. At N=30, performance slightly degrades. We select N=20 as it balances accuracy with efficiency.

7.2 Effect of Hint Components

We ablate three hint generation components on DBpedia-70: false positives (FP), false negatives (FN), and contrastive analysis (Table 9).

Removing contrastive hints causes the largest degradation (-3.0 to -5.7 points), demonstrating that explicitly targeting high-confusion pairs is critical. Removing FP hints (-2.1 to -3.2 points) or FN hints (-2.1 to -2.6 points) shows moderate but significant impact, as these define exclusion and inclusion criteria respectively.

The full combination achieves best performance (+4.7 to +10.1 points), confirming each component provides complementary signals: contrastive analysis targets confused pairs, FN adds inclusions, FP

Dataset	#Labels	nova-lite			gpt-4o-mini			claude-4-5-haiku		
		Init	Final	Δ(%)	Init	Final	Δ(%)	Init	Final	Δ(%)
DBPedia-70	70	77.14	87.23	+13.08	83.65	89.34	+6.80	85.86	90.57	+5.48
Clinc150	150	86.87	90.51	+4.19	87.00	91.27	+4.91	88.95	92.00	+3.43
Banking77	77	73.23	77.35	+5.63	74.44	78.78	+5.83	80.32	83.44	+3.88
Web of Science	35	60.12	64.27	+6.90	62.56	66.00	+5.50	62.62	65.69	+4.90
Travel	160	65.23	71.35	+9.38	67.44	71.78	+6.44	68.32	72.44	+6.03
E-Commerce	120	72.31	77.55	+7.25	71.28	76.43	+7.23	74.59	78.67	+5.47
Food Delivery	65	78.53	85.86	+9.33	79.46	86.74	+9.16	80.25	83.56	+4.12
Medical Insurance	80	60.26	72.74	+20.71	62.64	71.24	+13.73	67.35	73.87	+9.68

Table 3: Initial and final accuracies (%) across benchmark datasets. Δ denotes relative percentage improvement with respect to the initial accuracy. #Labels denote the number of classes in the datasets

adds exclusions.

8 Related Work

Instruction Optimization. Prior work has extensively explored automatic optimization of task instructions for LLM-based classification, including evolutionary approaches (Fernando et al., 2023), gradient-based methods (Pryzant et al., 2023b), and error-driven refinement (Yang et al., 2024; Ye et al., 2024b; Zhang et al., 2023). These methods optimize the reasoning process (HOW to classify) rather than class definitions (WHAT defines each class), representing an orthogonal dimension to our work.

Class Description Refinement. Several works address class description quality. Pattnaik et al. (2025) demonstrated effective high-cardinality classification through retrieval-augmented multi-view label representations. Most similarly, Rajeev et al. (2025) iteratively refine descriptions using contrastive prompting on misclassified samples with human-in-the-loop category introduction. However, their approach has key limitations: (1) no systematic quality control—refinements are applied without validation for specificity, actionability, or correctness; and (2) no performance verification—updates are accepted without confirming they improve classification or avoid degradation.

Our framework addresses these through: (1) *multi-agent validation*; (2) *performance-based acceptance*— discarding failed refinements; and (3) *demonstrated scalability*—effectiveness across 35-160 classes including real-world enterprise taxonomies.

Multi-Agent Methods. Self-reflection (Shinn et al., 2023; Madaan et al., 2023) and multi-agent architectures (Hong et al., 2024) have shown LLMs can improve outputs through specialized roles. We employ generator-reviewer pairs with explicit quality criteria enforced through generate-review-revise cycles.

9 Conclusion

We present a multi-agent framework for iterative, error-driven refinement of class descriptions in LLM-based classification. While prompt engineering research has focused extensively on optimizing task instructions—the HOW of classification—we demonstrate that systematic refinement of class descriptions—the WHAT that defines each category—addresses a complementary and often more impactful dimension of prompt quality.

Through evaluation across 4 proprietary contact center datasets and 4 public benchmarks spanning diverse domains and label cardinalities (35-160 classes), we achieve upto 20.71% accuracy improvements over static baseline descriptions. Our analysis reveals that description quality creates a performance ceiling that instruction optimization alone cannot overcome: scaling models by 10× and optimizing prompts yields minimal gains on confused classes, while our targeted description refinement directly resolves these systematic errors.

By automatically refining domain expert-authored descriptions using validation data, our framework enables accurate, maintainable LLM classification for enterprise deployments without requiring prompt engineering expertise.

Limitations and Future Work

Our approach requires validation data with sufficient samples per class and assumes errors stem primarily from inadequate descriptions rather than inherent data ambiguity. Hyperparameters were fixed across datasets, and evaluation is limited to English text classification. Future directions include: (1)

investigating synergy with instruction optimization for compounding benefits, (2) dataset-specific hyperparameter tuning and adaptive stopping criteria, and (3) extension to multilingual settings.

Ethical Considerations

Privacy. Our proprietary datasets contain de-identified contact center transcripts with PII removed following enterprise privacy protocols. Data usage complies with applicable regulations (GDPR, CCPA). Public datasets were used per their original licenses.

Bias and Fairness. Error-driven refinement could amplify biases if errors disproportionately affect certain groups. Practitioners should audit refined descriptions for discriminatory language and validate fair accuracy distribution across populations.

Labor Impact. Improved classification may affect contact center workforce dynamics. Our framework is intended to augment human agents by reducing misrouting, not replace them. Organizations should consider workforce impact and provide transition support.

Accountability. Automated description modifications require oversight. We recommend: (1) maintaining audit trails, (2) human review for critical categories (compliance, safety), and (3) domain expert approval for high-stakes deployments. Our iterative design supports human-in-the-loop validation.

Environmental Impact. Our framework incurs computational cost (Section B). While modest and one-time, practitioners should balance accuracy gains against environmental impact. Using efficient models for validation reduces carbon footprint.

References

Amazon AGI, Aaron Langford, Aayush Shah, Abhanshu Gupta, Abhimanyu Bhatter, Abhinav Goyal, Abhinav Mathur, Abhinav Mohanty, Abhishek Kumar, Abhishek Sethi, Abi Komma, Abner Pena, Achin Jain, Adam Kunysz, Adam Opyrchal, Adarsh Singh, Aditya Rawal, Adok Achar Budihal Prasad, Adrià de Gispert, and 767 others. 2025. The amazon nova family of models: Technical report and model card. *Preprint*, arXiv:2506.12103.

Anthropic. 2025a. Claude 4. Accessed: 2026-02-15.

Anthropic. 2025b. Claude haiku 4.5. Accessed: 2026-02-15.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 335–336, New York, NY, USA. Association for Computing Machinery.

Xuan Long Do, Yiran Zhao, Hannah Brown, Yuxi Xie, James Xu Zhao, Nancy F. Chen, Kenji Kawaguchi, Michael Shieh, and Junxian He. 2024. Prompt optimization via adversarial in-context learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7308–7327, Bangkok, Thailand. Association for Computational Linguistics.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2023. Promptbreeder: Self-referential self-improvement via prompt evolution. *Preprint*, arXiv:2309.16797.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. Metagpt: Meta programming for a multi-agent collaborative framework. *Preprint*, arXiv:2308.00352.

Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S. Gerber, and Laura E. Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, page 364–371. IEEE.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

Aristides Milios, Siva Reddy, and Dzmitry Bahdanau. 2023. In-context learning for text classification with many labels. *CoRR*, abs/2309.10954.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI. 2024. Gpt-4o-mini: Advancing cost-efficient intelligence.

Anup Pattnaik, Sasanka Vutla, Hamvir Dev, Jeevesh Nandan, and Cijo George. 2025. Scalable and cost effective high-cardinality classification with LLMs via multi-view label representations and retrieval augmentation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1955–1969, Suzhou (China). Association for Computational Linguistics.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023a. Automatic prompt optimization with "gradient descent" and beam search. *Preprint*, arXiv:2305.03495.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023b. Automatic prompt optimization with "gradient descent" and beam search. *Preprint*, arXiv:2305.03495.

Amrit Rajeev, Udayaadithya Avadhanam, Harshula Tulapurkar, and Sai Barath Sundar. 2025. Small sample-based adaptive text classification through iterative and contrastive description refinement. *arXiv preprint arXiv:2508.00957*.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Preprint*, arXiv:2303.11366.

Ankita Sinha, Wendi Cui, Kamalika Das, and Jiaxin Zhang. 2024. Survival of the safest: Towards secure prompt optimization through interleaved multi-objective evolution. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1016–1027, Miami, Florida, US. Association for Computational Linguistics.

Nathan Vandemoortele, Bram Steenwinckel, Femke Ongenae, and Sofie Van Hoecke. 2025. From haystack to needle: Label space reduction for zero-shot classification. *CoRR*, abs/2502.08436.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P. Xing, and Zhiting Hu. 2024. Promptagent: Strategic planning with language models enables expert-level prompt optimization. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhiqiang Wang, Yiran Pang, and Yanbin Lin. 2023. Large language models are zero-shot text classifiers. *CoRR*, abs/2312.01044.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. 2024a. Prompt engineering a prompt engineer. *Preprint*, arXiv:2311.05661.

Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. 2024b. Prompt engineering a prompt engineer. *Preprint*, arXiv:2311.05661.

Chenrui Zhang, Lin Liu, Jinpeng Wang, Chuyuan Wang, Xiao Sun, Hongyu Wang, and Mingchen Cai. 2023. Prefer: Prompt ensemble learning via feedback-reflect-refine. *Preprint*, arXiv:2308.12033.

A Ablation Tables

A.1 Effect of Sampling Strategy

We adopt MMR sampling to select the N error instances used for hint generation. This approach ensures that the selected error instances cover a wider variety of error patterns and avoid redundant selection of similar errors. We evaluate the effectiveness of MMR sampling by comparing it against random sampling across all datasets using nova-lite. As show in Table 4, MMR-based sampling consistently outperforms random sampling, with delta accuracy gains ranging from 1.07% to 7.14%.

Dataset	Accuracy	
	MMR	Random
DBPedia-70	87.23	83.75
Clinc150	90.51	88.61
Banking77	77.35	76.24
Web of Science	64.27	63.11
Travel	71.35	69.52
E-Commerce	77.55	76.73
Food Delivery	85.86	83.12
Medical Insurance	72.74	67.89

Table 4: Effect of sampling strategy on classification accuracy across datasets using nova-lite.

# Labels	Initial Acc	Final Acc	$\Delta(\%)$
9	88.89	90.83	2.18
70	77.14	87.23	**13.08**
219	75.42	79.98	6.04

Table 5: Accuracy (%) before and after optimization on DBpedia subsets with varying number of class labels for nova-lite model. Δ denotes relative percentage improvement with respect to the initial score.

A.2 Effect of Number of Classes

We evaluate our method on three subsets of DBpedia with 9, 70, and 219 class labels, respectively. Table 5 reports the initial and final accuracy along with the percentage improvement (Δ). With only 9 classes, the label space is compact and inter-class boundaries are relatively well-separated, leaving limited room for improvement ($\Delta = 2.18\%$). As the number of labels grows to 70, confusion between semantically similar classes increases substantially, and the optimization yields the largest gain ($\Delta = 13.08\%$). At 219 classes, although inter-class confusion remains high, the sheer granularity of the label space makes it harder for descriptions alone to fully disambiguate closely related categories, resulting in a moderate improvement ($\Delta = 6.04\%$). These results suggest that our approach is most beneficial in mid-to-large taxonomies where class overlap is significant yet still addressable through improved descriptions.

A.3 Effect of Refinement Iterations

In each iteration, we update a single class description and evaluate whether the revision improves classification performance. An update is accepted only if it satisfies both class-level and global validation criteria; otherwise, it is rejected. To limit computational cost, we fixed maximum iterations (T_{max}) to 20 for all datasets and then analyzed the accepted and rejected updates across datasets to understand the optimization dynamics(Table 12).

DBPedia-70 exhibits the lowest acceptance rate of 16%, highlighting the difficulty in refining class descriptions without degrading overall accuracy. Web of Science shows a similarly low acceptance rate 18%, however, this is primarily due to early convergence, as it met the stopping criteria by the 10th iteration. In the final iteration, all remaining classes were attempted, but none passed validation, lowering the overall acceptance rate.

CLINC150 has the highest acceptance rate of 36%, with stable and frequent successful updates. Given its larger label space and low rejection frequency, it would likely benefit from additional iterations to further refine the remaining classes and improve accuracy.

Among internal datasets, Medical Insurance exhibits lowest overall acceptance rate at 15%, as its class boundaries are more nuanced and difficult to distinguish, making refinement challenging without affecting the performance of other classes.

B Computational Cost Analysis

To assess the practical feasibility of our framework for enterprise deployment, we analyze computational costs across all datasets. We assume 100 validation instances per class, 20 refinement iterations, and $N = 10$ error instances sampled per iteration. Costs are computed using standard API pricing for Claude Sonnet 4 (Hint Generator, Description Synthesis), DeepSeek-R1 (Hint Reviewer, Description Reviewer), and Nova Lite (classification).

Table 13 breaks down costs into two components: (1) *Refinement*—the cost of multi-agent hint generation, review, synthesis, and description review cycles, and (2) *Validation*—the cost of classifying the validation set to identify errors, compute metrics, and validate refined descriptions.

Cost Structure. Refinement costs range from \$13 (Web of Science) to \$62 (Travel), correlating primarily with taxonomy cardinality and the number of classes requiring refinement. Each refinement iteration involves:

- **Hint Generation:** Up to 10 errors/iteration × 20 iterations = 200 hint generation-review cycles (each with up to $R_{max} = 3$ revisions)
- **Description Synthesis:** 1 description/iteration × 20 iterations = 20 description

Dataset	nova-lite				gpt-4o-mini				claude-4-5-haiku			
	Init	Final	Init Conf.	Final Conf.	Init	Final	Init Conf.	Final Conf.	Init	Final	Init Conf.	Final Conf.
DBPedia-70	0.7644	0.8655	0.4044	0.7496	0.8247	0.8813	0.4954	0.7990	0.8568	0.9017	0.5996	0.7454
CLINC150	0.8668	0.9029	0.6593	0.8454	0.8646	0.9130	0.4946	0.7403	0.8900	0.9187	0.7909	0.9196
Banking77	0.7266	0.7713	0.5330	0.6080	0.7409	0.7842	0.6385	0.7375	0.7984	0.8316	0.6785	0.7630
Web of Science	0.6014	0.6413	0.4726	0.5545	0.6234	0.6584	0.5065	0.5806	0.6262	0.6517	0.4348	0.4933
Travel	0.6477	0.7040	0.4156	0.5563	0.6628	0.7087	0.4878	0.6012	0.6793	0.7221	0.5031	0.6145
E-commerce	0.7159	0.7674	0.5212	0.6304	0.7130	0.7624	0.5735	0.7094	0.7389	0.7837	0.5662	0.7127
Food Delivery	0.7721	0.8491	0.4639	0.5930	0.7832	0.8582	0.5047	0.5829	0.7946	0.8264	0.5259	0.6156
Medical Insurance	0.5873	0.7075	0.4018	0.5358	0.6156	0.7027	0.4287	0.5311	0.6498	0.7194	0.4234	0.5587

Table 6: Initial and final Macro F1 scores (on the full dataset) and Macro F1 scores on confused classes across datasets and models. "Init" and "Final" refer to scores on the full dataset before and after optimization, while "Init Conf." and "Final Conf." refer to scores on confused classes before and after optimization.

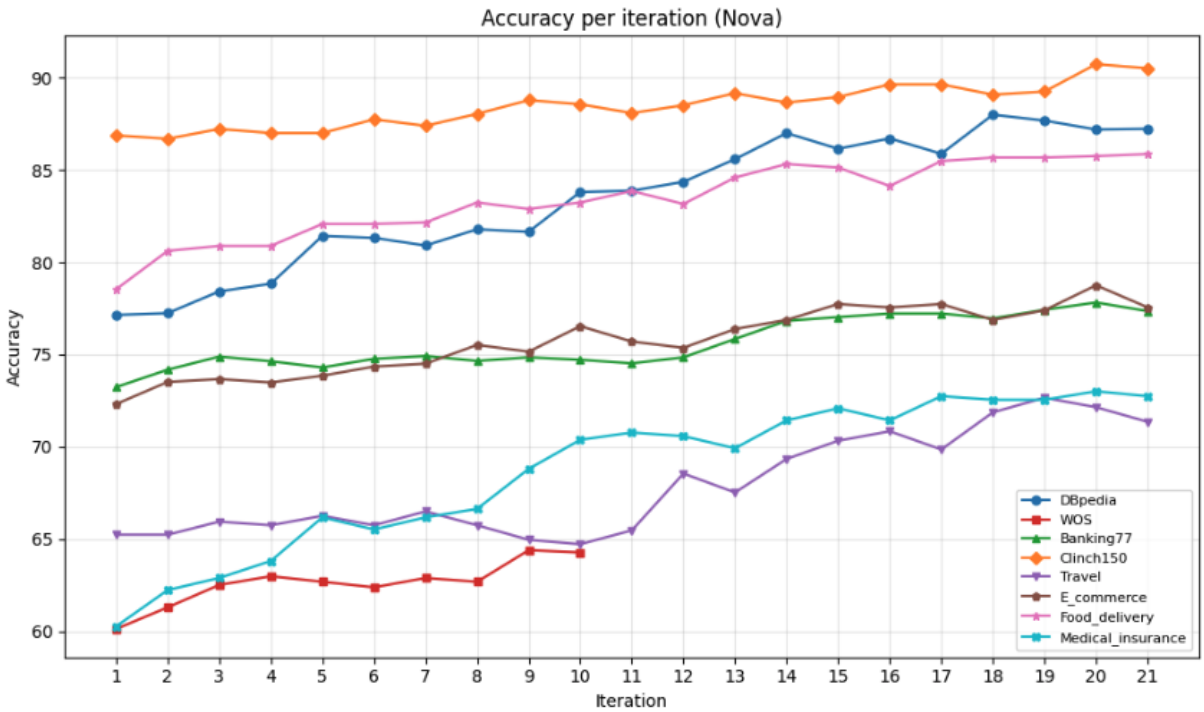

Figure 2: Accuracy plot for nova-lite model across datasets and iterations

synthesis-review cycles (each with up to $R_{max} = 3$ revisions)

However, actual costs vary because: (1) not all classes undergo refinement (average: 32%), (2) high-cardinality taxonomies generate more contrastive hints for confused pairs, and (3) review cycles may terminate early upon approval. Larger taxonomies like CLINC150 (150 classes, $55) and Travel (160 classes, $62) incur higher refinement costs than smaller ones like Web of Science (35 classes, $13).

Validation costs vary even more substantially—from $34 to $657—scaling with validation set size and text length. Each of the 20 iterations requires classifying the full $\mathcal{D}_{val}$ twice: once for error analysis, and once for validating the refined description. For most datasets, validation dominates total cost (75-95% of expenditure).

Cost Drivers. Both cost components exhibit clear dependencies. Refinement costs scale with taxonomy cardinality: datasets with 150+ classes (CLINC150: $55, Travel: $62) cost 3-5× more than those with <40 classes (Web of Science: $13). Validation costs scale with the product of dataset size and token length: CLINC150 (15K instances × 15 tokens, $551) and Travel (16K × 800 tokens, $657) incur the highest costs, while Medical Insurance with fewer but longer texts (8K × 2,000 tokens, $198) falls in the middle range.

Cost-Effectiveness. These are one-time offline costs amortized across potentially millions of future inferences. For example, DBPedia-70's total cost of $153 ($28 + $125) yields a +10.09 absolute percentage point accuracy improvement (+13.08%

Category	Original Description	Final Description	Most Confused With
WinterSport-Player	This topic represents professional winter sports athletes, particularly those who have competed in Olympic or international-level competitions in sports like figure skating, skiing, ski jumping, short track speed skating, ice dancing, and curling.	This category represents professional winter sports athletes whose primary notability stems from competitive careers in sports such as figure skating, alpine skiing, ski jumping, speed skating, ice dancing, curling, and ice hockey at Olympic, international championship, or professional league levels. Athletes include both active and retired competitors, including those who transitioned to coaching, broadcasting, or other post-career roles, as long as their biographical identity centers on their competitive achievements such as Olympic medals, world championships, or professional league success. Former NHL players who became commentators or broadcasters remain classified here since their notability derives from their playing career, not their media work. Similarly, athletes with significant non-athletic accomplishments like military service are included when their winter sports achievements are a defining part of their identity. The category encompasses athletes from various regions who achieved elite-level competitive success in winter sports.	Athlete
FootballLeague-Season	This topic represents seasons or annual summaries of professional football teams (NFL, CFL, and other leagues), typically describing their win-loss records, key events, personnel changes, and playoff/championship outcomes for a specific year.	This category covers individual team seasons in North American professional gridiron football leagues only, including NFL, CFL, Arena/Indoor Football League, CIFL, AIFA, and other regional or historical professional circuits, but explicitly excludes college football (NCAA), lacrosse leagues (MLL), and all association football or soccer. Articles describe a single team's complete season with game-by-game narratives, specific scores, play-by-play moments, win-loss records, division standings, playoff journeys, and championship outcomes like Super Bowl or Grey Cup appearances. Content may include franchise history, expansion details, stadium changes, ownership shifts, league transitions, and mid-season roster changes that shaped that particular season. This category applies only to team-level seasonal performance narratives, not individual player or coach biographies, league-wide summaries covering all teams collectively, or multi-team transfer window lists. The focus on detailed in-game storytelling and football-specific league context distinguishes this from generic sports team season articles.	SportsTeamSeason
ComicsCharacter	This topic represents fictional characters from Japanese anime and manga series, particularly focusing on main protagonists and supporting characters with detailed background stories, special abilities, and their roles within their respective series.	This category includes fictional characters originating from Japanese illustrated media formats including manga, anime, light novels, and visual novels. Characters retain this classification even if later adapted into other formats, as long as their primary origin is Japanese illustrated media. The category encompasses protagonists, supporting characters, and antagonists from these franchises, including mecha series and Type-Moon's Fate series. This category explicitly excludes characters from live-action television including soap operas and Western productions, Western comics, and ancient mythology or classical literature such as Greek, Roman, or Norse mythological figures, even if these mythological characters are later reimagined in Japanese media. Only characters whose original creation was in Japanese illustrated formats qualify for this category.	Fictional-Character
Stream	This topic represents descriptions of rivers, streams, brooks, and canals, focusing on their geographical locations, physical characteristics, tributaries, and historical significance. The examples consistently describe various waterways and their connections to larger water bodies.	Stream applies to all rivers, streams, brooks, tributaries, distributary arms, and named watercourses regardless of size, cultural significance, or how extensively their physical characteristics, ecological features, or historical context are described. Signal terms like joins, empties into, flows through, tributary, and distributary indicate Stream classification. Canals are Stream when described as geographical waterway features with defined routes and connections to water bodies, but belong to RouteOfTransportation only when the text focuses primarily on navigation infrastructure like locks or commercial cargo transport rather than geographical attributes. Stream is distinct from BodyOfWater, which encompasses only static water features like lakes, ponds, and seas that lack flowing watercourse characteristics and tributary relationships. Rivers are always Stream, never BodyOfWater, even when described by basin area, length, drainage patterns, or connections to lakes and seas.	BodyOfWater

Table 7: Examples of original and refined class descriptions produced by the Agentic Class Description Improvement Pipeline for the most commonly confused classes from the DBPedia dataset.

Issue Type	Problematic Description	Discriminative Problem
Vague Boundaries	"Customer complaints about service quality"	No criteria distinguishing service quality from product defects, delivery issues, or billing problems
	"Technical support requests"	Unclear whether includes account questions, password resets, or only hardware/software issues
	"General inquiry"	Catch-all category without specific inclusion/exclusion criteria
Semantic Overlap	"Billing disputes" vs. "Payment issues" vs. "Invoice problems"	Three labels for related concepts: disputed charges, failed payment processing, and incorrect invoice generation respectively—boundaries unclear
	"Account access problems" vs. "Login issues" vs. "Authentication failures"	Overlapping concepts without contrastive definitions; all refer to inability to access account
	"Order cancellation" vs. "Order modification" vs. "Order return"	Boundaries blur when customers want to cancel-then-reorder or return-for-exchange
Missing Contrast	"Software issues" vs. "Hardware issues"	No specification: is "device won't turn on" hardware (battery) or software (OS crash)?
	"Refund request" vs. "Credit request"	Missing distinction: refund returns money, credit provides store value—not specified
	"New account setup" vs. "Account reactivation"	Both involve getting access to an account; lacks explicit contrast on whether account previously existed
Incomplete Coverage	"Delivery delay"	Doesn't specify whether includes lost packages, wrong address deliveries, or only late arrivals
	"Promotional inquiry"	Unclear if covers current promotions, expired promotions, eligibility questions, or all three
Overly Broad	"Account issues"	Umbrella term covering login, settings, profile, billing, security—lacks specificity
	"Website problems"	Encompasses navigation, load time, display errors, broken links, checkout issues—too general

Table 8: Representative examples of class description quality issues in production contact center taxonomies, showing how inadequate descriptions create systematic confusion patterns.

Model	Contrastive	FP	FN	Init Acc	Final Acc	Δ(%)
nova-lite	✗	✓	✓	77.14	81.57	+5.74
	✓	✗	✓	77.14	84.00	+8.89
	✓	✓	✗	77.14	84.86	+10.00
	✓	✓	✓	77.14	**87.23**	**+13.08**
gpt-4o-mini	✗	✓	✓	83.65	85.29	+1.96
	✓	✗	✓	83.65	86.84	+3.81
	✓	✓	✗	83.65	87.29	+4.35
	✓	✓	✓	83.65	**89.34**	**+6.80**
claude-4-5-haiku	✗	✓	✓	85.86	87.58	+2.00
	✓	✗	✓	85.86	88.43	+2.99
	✓	✓	✗	85.86	88.00	+2.49
	✓	✓	✓	85.86	**90.57**	**+5.48**

Table 9: Ablation study on the DBPedia-70 dataset across different models. Contrastive, FP, and FN denote the proposed components. Δ denote the relative percentage improvement over initial scores

acc improvement over baseline numbers). At 1M inferences annually, this represents $0.000153 per inference—negligible compared to inference cost, while preventing 100K misclassifications. For contact center deployments where misrouting costs $5-15 per incident in agent time and customer satisfaction, even modest accuracy gains generate substantial ROI.

C Hyperparameter Configuration

Our framework requires configuration of several hyperparameters that control the tradeoff between refinement thoroughness and computational cost.

# Error Points	nova-lite	gpt-4o-mini	claude-4-5-haiku
0	77.14	83.65	85.86
5	81.29	85.71	87.43
10	83.71	87.14	88.86
15	84.43	87.86	88.57
20	87.23	**89.34**	**90.57**
25	**87.51**	89.26	90.15
30	86.86	89.00	90.14

Table 10: Accuracy (%) on DBPedia-70 test set for different numbers of error points sampled using MMR across different models.

Text	Actual	Predicted	Hint
Marine Corps Air Station Futenma or MCAS Futenma ...	Infrastructure	Building	Emphasize that military installations like air stations are complete operational complexes with multiple integrated facilities functioning as a unified infrastructure system, not individual buildings.
Acarospora veronensis is a medium brown to dark brown or black crustose lichen ...	Eukaryote	Plant	Lichens are symbiotic organisms between fungi and photosynthetic partners, making them fundamentally different from plants. They should be classified under Eukaryote, not Plant.
Geoff Johns (born January 25, 1973) is an American comic book and television writer ...	Artist	Writer	Include comic book writers as Artists, since comic books are a visual storytelling medium where the writer's role is fundamentally artistic and creative.
Hubert Brooks MC (December 29, 1921 – February 1, 1984) was a Canadian RCAF officer and ice hockey player ...	WinterSport-Player	Athlete	Emphasize that WinterSportPlayer should be selected over generic Athlete for individuals who competed in winter sports at Olympic or international levels.
The men's synchronized 10 metre platform diving competition at the 2016 Summer Olympics ...	Olympics	SportsEvent	Individual competitive events held at the Olympics should be classified as Olympics rather than SportsEvent, as long as the Olympic context is explicitly mentioned.
In Greek mythology, Iphito was an Amazon who served under Hippolyte. Her name is only known from inscriptions. ...	FictionalCharacter	ComicsCharacter	Explicitly exclude characters from ancient mythology, folklore, and classical literature (such as Greek, Roman, Norse, or other traditional mythologies) from this category, as these predate modern comics and manga by centuries and should be classified under broader fictional character categories instead.
The 1985 Buffalo Bills season was the 26th season for the club and its 16th in the National Football League (NFL) ...	FootballLeagueSeason	SportsTeamSeason	Emphasize that FootballLeagueSeason specifically covers seasons in North American professional football leagues (NFL, CFL, Arena Football, etc.), and explicitly state that this category takes precedence over the more generic SportsTeamSeason when the sport is American or Canadian football, as these are specialized football league contexts.

Table 11: Examples of hints generated by the Hint Generation Agentic System for misclassified data points.

For all experiments reported in this paper, we fixed these values based on preliminary tuning on a small held-out development subset from DBpedia-70. While dataset-specific tuning could potentially improve performance, we deliberately kept hyperparameters constant to demonstrate the framework's robustness across diverse settings.

C.1 Hyperparameter Values

Table 14 lists all hyperparameters used in our experiments with their values and descriptions.

C.2 Rationale and Sensitivity

Sample Size ($N = 20$). We chose $N = 20$ based on ablation studies (Section 7.1) showing that performance plateaus beyond 10-15 samples per class, with diminishing returns from additional instances. This value balances error diversity with computa-

Dataset	Total	Upd.	Rej.	Uniq. Rej.
Banking77	77	20	57	27
CLINC150	150	20	35	20
DBPedia-70	70	20	103	24
Web of Science	35	9	40	25
Travel	160	20	86	31
E-commerce	120	20	53	29
Food Delivery	65	20	45	24
Medical Insurance	80	20	110	34

Table 12: Class Refinement statistics across datasets over 20 iterations with nova-lite. Total = total number of classes in the taxonomy; Upd. = number of successfully updated classes; Rej. = total rejected update attempts; Uniq. Rej. = number of distinct classes with at least one rejection.

Dataset	Supp.	Cls	Tok/Inst	Refine	Valid
DBPedia-70	7,000	70	153	$28	$125
CLINC150	15,000	150	15	$55	$551
Banking77	7,700	77	13	$30	$148
Web of Science	3,500	35	300	$13	$34
Travel	16,000	160	800	$62	$657
E-Commerce	12,000	120	1,200	$52	$389
Food Delivery	6,500	65	400	$23	$112
Medical Insurance	8,000	80	2,000	$27	$198

Table 13: Computational cost breakdown (USD) for one-time description refinement. Supp. = Support, Cls = Classes, Tok/Inst = Tokens/Instance, Refine/Valid = Refinement/Validation costs.

tional efficiency.

Confusion Threshold ($\tau_{conf} = 30\%$). Setting this threshold at 30% identifies the most systematic confusion patterns while avoiding over-sensitivity to noise. Lower values (e.g., 10%) would generate too many contrastive hints, many capturing random rather than systematic confusion. Higher values (e.g., 50%) would miss important confused pairs.

Review Cycles ($R_{max} = 3$). Allowing up to 3 revision rounds provides sufficient opportunity for quality improvement while preventing infinite loops. Empirically, 85% of successful refinements achieve approval within 2 cycles.

Validation Thresholds ($\tau_{resolve} = 20\%$, $\tau_{class} = 5\%$, $\tau_{global} = 0.5\%$). These conservative thresholds ensure refinements provide meaningful improvement ($\geq$20% error resolution) without degrading performance elsewhere. The stricter global threshold (0.5%) compared to class-wise (5%) reflects that overall accuracy is more critical than individual class metrics for production systems.

Maximum Iterations ($T_{max} = 20$). This represents an effective compute-quality tradeoff, capturing substantial accuracy improvements while limiting computational cost.

Parameter	Value
N	20
τ_{conf}	30%
R_{max}	3
$\tau_{resolve}$	20%
τ_{class}	5%
τ_{global}	0.5%
T_{max}	20

Table 14: Hyperparameter configuration used across all experiments.

The fact that our fixed configuration achieves consistent improvements across datasets with 35-160 classes and 13-4000 token texts suggests these hyperparameters are reasonably robust. However, dataset-specific tuning could potentially enhance performance further, particularly for datasets with unusual characteristics (e.g., extremely high cardinality, very short texts, or highly imbalanced class distributions).

D Models

In this paper, we perform experiments by using nova-lite (AGI et al., 2025), gpt-4o-mini (OpenAI, 2024) and claude-4-5-haiku (Anthropic, 2025b) as the classification models. In both the hint generation and description generation agentic systems, we use the claude-sonnet-4 (Anthropic, 2025a) as the generation model and deepseek-r1 (Guo et al., 2025) as the reviewer model.

E Prompts

In this section, we present all the prompts that are used in the different stages of the class description refinement pipeline. The template variables are denoted using curly braces (e.g., `{text}`).

E.1 Classification

Table 15 contains the prompts used for performing LLM classification on the dataset. The prompt takes as input the text to classify and all the class labels and the class descriptions and outputs the class label.

E.2 Hint Generation

Table 16, 17 and 18 detail the prompts used by the Hint Generation Agentic System. These include prompts for the Hint Generation Agent to produce

Prompt	Content
System Prompt	You are an expert classifier. Your task is to classify the given text into one of the provided categories. Rules: 1. Analyze the text carefully 2. Choose the MOST appropriate category based on the descriptions 3. Output ONLY the category name, nothing else 4. If unsure, choose the closest matching category
User Prompt	Classify the following text into one of these categories: {categories} Text to classify: {text} Output only the category name (exactly as shown above, without the description):

Table 15: Classification Prompts used to prompt the LLM to classify the data points into one of many class labels given the input text, class labels and class descriptions.

hints and the Hint Reviewer Agent to evaluate them and provide feedback.

E.3 Description Refinement

Table 19 contains the prompts used in the Description Update Agentic System. It includes the prompt used by the Description Update Agent to generate the initial description and the prompts used by the Description Reviewer Agent to review the update and provide feedback.

F Algorithms

Prompt	Content
System Prompt	You are an expert at analyzing classification errors. Given a misclassified example, provide a specific hint to improve the category description.
False Positive Hint Generation	A text was incorrectly classified as "{category_name}" when it actually belongs to "{true_label}". Category: {category_name} Current Description: {current_description} Misclassified Text: "{text}" This is a FALSE POSITIVE – the text does NOT belong to "{category_name}" but was wrongly assigned. What should be EXCLUDED or clarified to avoid classifying texts like this as "{category_name}"? Respond with ONE concise hint (1 sentence only):
False Negative Hint Generation	A text that SHOULD have been classified as "{category_name}" was incorrectly classified as "{predicted_label}". Category: {category_name} Current Description: {current_description} Misclassified Text: "{text}" This is a FALSE NEGATIVE – the text belongs to "{category_name}" but wasn't recognized. What specific characteristic should be INCLUDED in the description to capture texts like this? Respond with ONE concise hint (1 sentence only):
Contrastive Hint Generation	We are improving the description for CATEGORY A to reduce confusion with CATEGORY B. CATEGORY A (target class to improve): {class_a_name} Current Description: {class_a_description} CATEGORY B (confused class): {class_b_name} Description: {class_b_description} Here are examples of texts that were misclassified between these categories: {error_examples} What is the KEY DISTINGUISHING FACTOR that separates "{class_a_name}" from "{class_b_name}"? Provide ONE concise statement (1–2 sentences) that clarifies when a text belongs to "{class_a_name}" versus "{class_b_name}". Start with "Unlike {class_b_name}..." or "This specifically covers... rather than..." or "Focus on... not...":

Table 16: Hint Generation Prompts used by the Generation Agent within the agentic system to produce hints for various error types.

Prompt	Content
Hint Review – System Prompt	You are an expert reviewer of classification improvement hints. Evaluate whether the hint is specific, actionable, and correctly addresses the classification error.
Hint Review	Review the following hint generated to improve a category description after a classification error. Error Context: - Category: {category_name} - Current Description: {current_description} - Error Type: {error_type_desc} - Misclassified Text: "{text}" - {label_context} Generated Hint: "{hint}" Evaluate the hint on these criteria: 1. SPECIFIC: Does it identify a concrete characteristic, not a vague generalization? 2. ACTIONABLE: Can it be directly used to modify the category description? 3. CORRECT: Does it correctly address why this text was misclassified? Respond with: APPROVE – if the hint meets all criteria REVISE: [specific feedback on what to improve] – if changes are needed
Hint Revision	Revise your hint based on the reviewer's feedback. Error Context: - Category: {category_name} - Current Description: {current_description} - Error Type: {error_type_desc} - Misclassified Text: "{text}" - {label_context} {reasoning_section} Your Previous Hint: "{previous_hint}" Reviewer Feedback: {review_feedback} Generate a revised hint that addresses the feedback. Respond with ONE concise hint (1 sentence only):

Table 17: Hint review and revision prompts used by the Hint Reviewer Agent to evaluate the generated hint and provide feedback, and by the Hint Generation Agent to refine the hint using that feedback.

Prompt	Content
Contrastive Hint Review	Review the following contrastive hint generated to distinguish two confused categories. Category A (target): {class_a_name} Description A: {class_a_description} Category B (confused with): {class_b_name} Description B: {class_b_description} Generated Contrastive Hint: "{hint}" Evaluate the hint on these criteria: 1. DISTINGUISHING: Does it clearly identify what separates Category A from Category B? 2. SPECIFIC: Does it point to concrete differentiating factors, not vague statements? 3. ACTIONABLE: Can it be used to update the category description effectively? 4. CORRECT: Is the distinction factually accurate based on the category descriptions? Respond with: APPROVE – if the hint meets all criteria REVISE: [specific feedback on what to improve] – if changes are needed
Contrastive Hint Revision	Revise your contrastive hint based on the reviewer's feedback. Category A (target): {class_a_name} Description A: {class_a_description} Category B (confused with): {class_b_name} Description B: {class_b_description} Error Examples: {error_examples} Your Previous Hint: "{previous_hint}" Reviewer Feedback: {review_feedback} Generate a revised contrastive hint that addresses the feedback. Start with "Unlike {class_b_name}..." or "This specifically covers... rather than..." or "Focus on... not...". Respond with ONE concise statement (1–2 sentences):

Table 18: Contrastive hint review and revision prompts are used by the Hint Reviewer Agent to generate feedback and the Hint Generation Agent to improve the hint based on that feedback.

Prompt	Content
Description Addition – Generate	Based on error analysis, generate text to APPEND to the category description. Category: {category_name} Current Description: {current_description} {contrastive_hints} INCLUDE hints (to capture more relevant texts): {fn_hints} EXCLUDE hints (to avoid wrong classifications): {fp_hints} IMPORTANT: Review the current description above carefully. Do NOT include any hints that are already covered or implied by the existing description. Only include NEW information. {feedback_section} Summarize the NEW hints (those not already in the description) into 1–3 concise sentences that can be APPENDED. Write ONLY the text to add, nothing else. Prioritize the contrastive differentiators if present. If ALL hints are already covered by the current description, respond with an empty message.
Description Addition – Review	Review the proposed additions to this category description. Category: {category_name} Current Description: {current_description} Proposed Additions: {additions} The following hints were generated from classification errors: {contrastive_hints} INCLUDE hints: {fn_hints} EXCLUDE hints: {fp_hints} Check if the additions: 1. Include NEW information not already in the description 2. Address the hints above Respond with: APPROVE – if additions are good REVISE: [specific feedback] – if changes needed

Table 19: Description Refinement and Review prompts used by the Description Update Agent to generate an initial update and the Description Reviewer Agent to evaluate it and provide feedback.

Algorithm 1 Error-Driven Class Description Refinement

Require: Labels $\mathcal{Y}$, initial descriptions $\mathcal{C}^{(0)}$, validation set $\mathcal{D}_{val}$
Require: Hyperparameters: N (samples), τ_{conf}, $\tau_{resolve}$, τ_{class}, τ_{global}, T_{max}, R_{max}
Ensure: Refined descriptions $\mathcal{C}^*$

Initialize: $\mathcal{C} \leftarrow \mathcal{C}^{(0)}$, $\mathcal{R} \leftarrow \emptyset$ (non-refinable classes)
for $t = 1$ to T_{max} **do**
  // Stage 1: Error Analysis
  Classify $\mathcal{D}_{val}$ with $\mathcal{C}$; compute $F1_\ell$ for each class y_ℓ
  Select $y_j \leftarrow \arg\min_{y \in \mathcal{Y} \setminus \mathcal{R}} F1_y$
  if no refinable classes remain **then**
    break
  end if
  Sample N error instances: FP_j and FN_j via MMR
  Identify confused classes: $\mathcal{Y}_{confused} \leftarrow \{y_k : err_{j,k} > \tau_{conf}\}$
  // Stage 2: Hint Generation
  $\mathcal{H}_j \leftarrow \emptyset$
  for each error $(x, \hat{y}, y)$ **do**
    $h \leftarrow \textsc{GenerateHint}(x, \hat{y}, y, c_j, \mathcal{Y}_{confused})$
    $h \leftarrow \textsc{ReviewHint}(h, R_{max})$ {Up to R_{max} revisions}
    if $h \neq$ null **then**
      $\mathcal{H}_j \leftarrow \mathcal{H}_j \cup \{h\}$
    end if
  end for
  // Stage 3: Description Refinement
  $\Delta c_j \leftarrow \textsc{SynthesizeDescription}(c_j, \mathcal{H}_j)$
  $\Delta c_j \leftarrow \textsc{ReviewDescription}(\Delta c_j, R_{max})$
  if $\Delta c_j =$ null **then**
    $\mathcal{R} \leftarrow \mathcal{R} \cup \{y_j\}$; **continue**
  end if
  $c_j^{new} \leftarrow c_j \oplus \Delta c_j$
  // Stage 4: Validation
  $err_{resolved} \leftarrow$ error resolution rate with c_j^{new}
  $(P_j^{new}, R_j^{new}, F1_j^{new}, A^{new}) \leftarrow$ metrics with c_j^{new}
  if $err_{resolved} \geq \tau_{resolve}$ **and** metrics stable **then**
    $c_j \leftarrow c_j^{new}$ {Accept}
  else
    $\mathcal{R} \leftarrow \mathcal{R} \cup \{y_j\}$ {Reject}
  end if
end for
return $\mathcal{C}$

Algorithm 2 Generate-Review-Revise Cycle

Require: Input data, generator agent G, reviewer agent V, max cycles R_{max}
Ensure: Approved output or null

$output \leftarrow G(\text{input})$
for $r = 1$ to R_{max} **do**
  $(decision, feedback) \leftarrow V(output)$
  if $decision =$ APPROVE **then**
    return $output$
  else if $decision =$ REVISE **then**
    $output \leftarrow G(\text{input}, feedback)$
  else
    return null
  end if
end for
return null {Max cycles exceeded}