Scaling Intent Understanding: A Framework for Classification with Clarification using Lightweight LLMs

Subhadip Nandi, Tanishka Agarwal, Anshika Singh, Priyanka Bhatt
Walmart Inc.
Bengaluru, India
{subhadip.nandi, tanishka.agarwal, anshika.singh, priyanka.bhatt}@walmart.com

Abstract

Despite significant progress in intent classification, most task-oriented dialogue systems continue to assign intents rigidly, failing to account for ambiguity in user utterances. This often results in misrouting, irrelevant responses, and poor user experience. While proprietary large language models (LLMs) can generate high-quality clarifying questions to resolve such ambiguity, their inference cost makes them impractical for large-scale production use. In contrast, smaller open-source LLMs are cost-effective but typically lack the capability to ask contextually appropriate clarifying questions. This paper presents a domain-agnostic framework that enables lightweight, production-ready open-source LLMs to jointly perform intent classification and ambiguity resolution through targeted clarifying questions. We validate our framework on both proprietary and public intent classification datasets, demonstrating its ability to perform intent classification as well as generate clarification questions in case of ambiguity. To support fair comparison against external baselines, we further introduce an evaluation methodology that measures not only intent accuracy but also the timing and quality of clarifying questions. Our instruction-tuned models achieve performance comparable to leading proprietary LLMs while offering an 8× reduction in inference cost, enabling broader, cost-efficient deployment. When deployed in the customer-care system of an e-commerce enterprise, our model reduced the misrouting rate by 8%, resulting in a significant improvement in automation rates, which potentially translates in dollar savings by reducing escalations to human agents.

1 Introduction

In conversational AI, Intent Classification (IC) is a critical first step that drives a system's ability to choose appropriate actions and generate relevant responses (Chen et al., 2019; Nandi et al., 2024; Xu et al., 2020; Agrawal et al., 2023). High IC accuracy is essential for effective user experiences, yet ambiguity, incompleteness, and noise in user utterances make intent understanding challenging. Fortunately, conversational systems naturally support disambiguation through clarifying questions (CQs) (Purver et al., 2003; Alfieri et al., 2022). Well-timed, well-formulated CQs enable systems to resolve ambiguity, accelerate task completion, and improve user satisfaction (van Zeelt et al., 2020; Siro et al., 2022).

Although clarification has been explored extensively in information retrieval (Zamani et al., 2020), search (Tavakoli, 2020; Aliannejadi et al., 2021; Keyvan and Huang, 2022), and code generation (Mu et al., 2023), it remains under-explored in task-oriented dialogue systems. Early work by Dhole, 2020 used rule-based or template-based CQ generation, which limited flexibility and generalization. More recently, Hengst et al., 2024 used conformal prediction to identify when to ask CQs and relied on proprietary LLMs to generate them. While effective at detecting ambiguity, their work neither evaluates the quality of the generated clarifications nor extends to multi-turn interactions.

Using proprietary LLMs for clarification is expensive in high-volume applications, whereas smaller open-source LLMs underperform without targeted training (Section 5). To bridge this gap, smaller models must be equipped to detect ambiguity and generate appropriate CQs. However, collecting high-quality training data, particularly conversations that start ambiguously and resolve clearly, is difficult. To overcome this, we develop a framework that generates such conversations synthetically and uses them to instruction tune lightweight open-source LLMs.

While using LLMs to generate clarification questions in task-oriented dialogue is intuitive, there is limited research on how to evaluate models for this capability. The LLM-as-a-Judge paradigm (Zheng

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 5: Industry Track, pages 181–192

et al., 2023; Gu et al., 2024) provides a starting point, but evaluation criteria remain underdefined. We address this gap by introducing explicit turn-level and conversation-level metrics and combining them into a comprehensive evaluation score suited for classification-with-clarification task.

LLM-driven clarification has been explored in other domains. For instance, Kuhn et al., 2022 show that prompting proprietary LLMs to ask clarifying questions improves open-domain question answering. They also propose methods for generating ambiguous queries and evaluating LLMs via simulated dialogues. Our work draws inspiration from this evaluation framework for both data generation and evaluation. Crucially, we diverge in our method of generating ambiguous queries, given the distinct nature of classification tasks, and take an additional step by synthesizing entire conversations. These conversations are then used in instruction tuning smaller open-source LLMs, making our solution viable for production environments.

Another relevant line of work is Action-Based Contrastive Self-Training (Chen et al., 2024), which teaches small LLMs to ask clarification questions using existing user–bot conversations as supervision. Their method assumes access to large volumes of conversational data for generating preference pairs and evaluates primarily on question answering and text-to-SQL tasks. In contrast, our approach is designed for real-world scenarios where such data is sparse. Moreover, their evaluation focuses mainly on final-answer correctness, whereas in production systems, unnecessary, poorly phrased, or excessive clarifications negatively impact user experience, increase latency, and raise operational costs, motivating the fine-grained evaluation metrics introduced in our work.

2 Methodology

We propose a framework that enables lightweight open-source LLMs to perform *classification with clarification* for any given domain. Once a domain provides intent-tagged data, the framework consists of three stages:

- Synthetic Conversation Generation
- Lightweight LLM Instruction-Tuning
- Comprehensive Evaluation

A diagram of the framework is shown in Figure 1.

2.1 Synthetic Conversation Generation

Domains typically have an abundance of intent-tagged unambiguous user queries, but ambiguous queries and multi-turn conversations are difficult and expensive to collect. Deploying a high-capacity LLM to gather such data from real users is slow, costly, and requires prolonged production deployment. Instead, we generate synthetic conversations offline using high-capacity LLMs. We first create diverse ambiguous queries from existing unambiguous samples and then simulate multi-turn interactions between two LLMs, one acting as the assistant and one as the user. These synthetic conversations form the training corpus for instruction tuning smaller open-source LLMs, making the approach practical and cost-effective.

2.1.1 Generating ambiguous queries

Ambiguity is task-dependent (Zhang et al., 2024). We define an ambiguous query as one that plausibly corresponds to two or more domain intents. Other forms of underspecification irrelevant to intent classification, e.g., unclear referents in "Book a restaurant for them" are not treated as ambiguity. We generate ambiguous–unambiguous query pairs using two methods:

- A high-capacity LLM edits each unambiguous query to add or remove information, verifies the resulting query is ambiguous, and identifies possible intents. Only confirmed ambiguous queries are kept. (Appendix Figure 3).
- For each intent pair, a high-capacity LLM generates a query that fits either intent and produces two corresponding unambiguous versions, one per intent.

2.1.2 Conversation Simulation

Ambiguous queries are then used to simulate realistic multi-turn conversations. A high-capacity LLM plays the assistant, performing intent classification and asking clarifying questions when needed. A second high-capacity LLM simulates the user.

We assume that users who initiate interactions with ambiguous queries still have a clear underlying intent. Thus, the user-simulating LLM receives both the ambiguous query and its corresponding unambiguous version and must respond consistently with the unambiguous intent. This interaction process, illustrated in Figure 2, yields coherent conversation trajectories. Prompts for the assistant and user LLMs appear in Appendix Figures 4 and 5, respectively. We use few-shot prompting (Brown

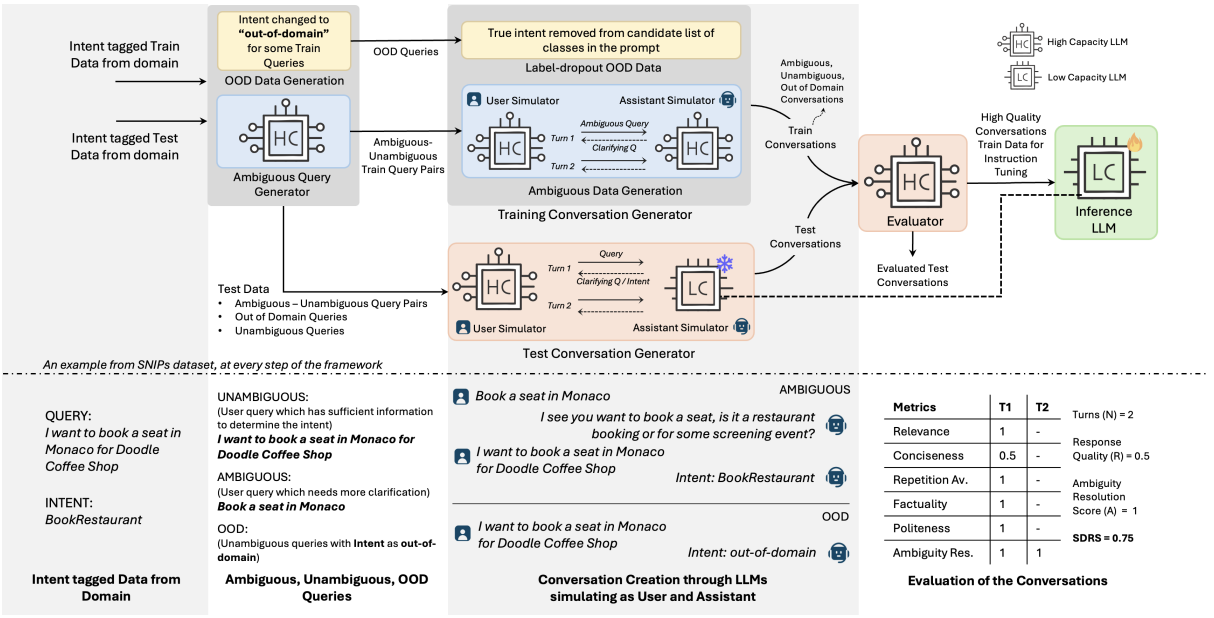

Figure 1: Overall framework where high-capacity LLMs generate and evaluate synthetic clarification dialogues, which are then used to instruction-tune low-capacity open-source LLMs for cost-efficient intent classification with clarification.

et al., 2020) for both.

The generated conversations are not used directly for model fine-tuning. They are first processed by the evaluation pipeline described in Section 2.3, and only high-scoring conversations are selected for instruction tuning the lightweight LLMs.

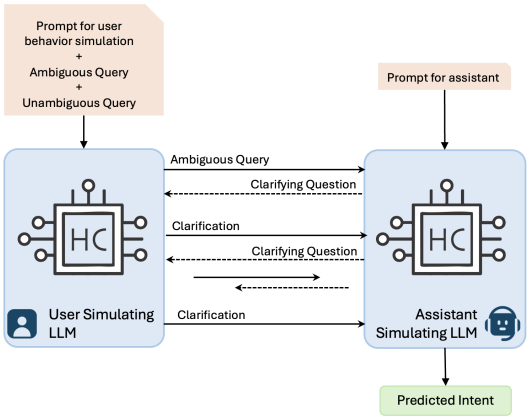

Figure 2: Conversation Generation - Simulating multi-turn dialogue between a user and an assistant, where the assistant LLM resolves an ambiguous query through iterative clarifying questions before predicting the final intent, while the user LLM starts with an ambiguous query and gradually reveals information grounded in the corresponding unambiguous query

2.2 Lightweight LLM Instruction-Tuning

To enable smaller production-ready open-source LLMs to perform intent classification with clarification, we apply supervised instruction-tuning (Wei et al., 2021). This adapts a general-purpose LLM to the task of detecting ambiguous intents and generating appropriate clarifying questions. The conversational data generated earlier is converted into structured input-output pairs as shown in Appendix section C. We use standard supervised fine-tuning, treating the task as sequence-to-sequence generation and optimizing with cross-entropy loss (De Boer et al., 2005).

2.2.1 Handling Out-of-Domain Queries: Label-Dropout OOD Training

Handling out-of-domain queries is essential in real-world intent classification, where users may express valid unambiguous intents not covered by the predefined intent set. Since this space is vast and difficult to sample, we introduce a label-dropout out-of-domain (OOD) training strategy. Inspired by the class-dropout method of Sainz et al., 2023, we intentionally omit the correct ground-truth label for a subset of unambiguous training examples during instruction-tuning and relabel these as "out-of-domain." This trains the model to predict OOD when an utterance is clear but its true intent is absent from the candidate classes. An additional benefit is improved safety: the model classifies harmful queries (e.g., "how to make a bomb") as OOD rather than attempting to generate harmful content.

2.3 Comprehensive Evaluation

All models, instruction-tuned using our framework, as well as external baselines, are benchmarked using a dedicated test set. Unambiguous intent-tagged test queries are obtained by splitting

Dataset	Intents	Examples of Intents	Unambiguous train/val/test	Ambiguous train/val/test
Proprietary	14 + 1 (OOD)	auto care service, online pickup and delivery, item availability	1000 / 150 / 250	1100 / 115 / 300
SNIPS	7	SearchCreativeWork, GetWeather, BookRestaurant	13084 (1400) / 700 / 700	1000 / 500 / 500
ATIS	17	Flight Info, Airport Info, Airline Info	4478 (1700) / 500 / 893	900 / 400 / 400

Table 1: Dataset Characteristics

the domain-provided data, while ambiguous test queries are generated using the method described in Section 2.1.1. Conversations between the simulated user and the tuned assistant are then produced following Section 2.1.2. These conversations are evaluated turn-by-turn, focusing on both final accuracy and intermediate response quality. While many tasks evaluate only the final predicted intent, our setting also requires assessing how the model reaches it, namely, the number of turns and the quality of clarifications. We adopt the **LLM-as-a-Judge** paradigm (Gu et al., 2024) for evaluation. A suitable metric must capture the following:

1. **Intent Prediction:**
   - Was the correct intent predicted within the allowed number of turns?
2. **Ambiguity Detection:**
   - Did the model predict intent when enough information was available?
   - Did it ask clarifying questions when needed?
3. **Response Quality:**
   - Relevance, Conciseness, Repetition Avoidance, Factuality, Politeness

Ambiguity detection is quantified using the Turn Level Ambiguity Resolution Score (ars_i), which measures the assistant's correctness in determining whether enough information is available at each turn to make a final prediction. More details appear in Appendix section D. Different possible categories for each of the response quality metrics along with their associated scores (0/0.5/1) can be found in Appendix section E. All metrics are combined into the Strategic Dialogue Response Score (SDRS):

$$\text{SDRS} = \begin{cases} 0 & \text{wrong intent or max turns reached} \\ A & \text{single-turn conversation} \\ \alpha \cdot A + (1-\alpha)R & \text{multi-turn conversation} \end{cases}$$

where:

$$A = \frac{1}{N}\sum_{i=1}^{N} ars_i,$$

$$R = \frac{1}{N-1}\sum_{i=1}^{N-1}(r_i\, c_i\, ra_i\, f_i\, p_i),$$

ars_i = Turn-level Ambiguity Resolution Score
r_i = Relevance c_i = Conciseness
ra_i = Repetition Avoidance f_i = Factuality
p_i = Politeness N = Number of turns
A = Conversation-level Ambiguity Resolution Score
R = Conversation-level Response Quality
α = Weight on $A(0.5)$.

Conversation-level response quality is averaged over $N-1$ turns, since the final turn is the intent prediction and not evaluated. The maximum turn limit is 5.

3 Experimental Setup

This section outlines our experimental setup, including the selection and training of lightweight open-source LLMs, dataset characteristics, ambiguous query generation, the high-capacity LLMs used for conversation synthesis, and the evaluation methodology.

3.1 LLM Selection for Instruction-Tuning

We evaluate lightweight open-source LLMs such as Llama-3.1-8B Instruct (Grattafiori et al., 2024), Llama-3.2-3B Instruct and Qwen-2.5-7B Instruct (Bai et al., 2023) as base models, chosen for their balance of performance and inference efficiency, making them suitable for high-volume production deployment. Their results appear in Section 5. All models are instruction-tuned using the method described in Section 2.2. Fine-tuning is performed using Low-Rank Adaptation (LoRA) (Hu et al., 2021) implemented in PyTorch. Details of hyperparameters selected can be found in Appendix section B. All instruction-tuning experiments were conducted on a single NVIDIA A100 80GB GPU.

3.2 LLM Selection for Data Generation and Evaluation

We use high-capacity LLMs for ambiguous query creation, conversation synthesis, and evaluation:

Dataset	Assistant Model	IC Acc%	SDRS	ARS	Rel	Con	RA	Fac	Pol
SNIPS	CTRAN (Non-LLM SOTA)	**99.42**	-	-	-	-	-	-	-
	Transformer based model (formerly deployed)	98.68	-	-	-	-	-	-	-
	Llama-3.1-8B (base)	88	0.51	0.52	0.89	0.95	0.96	0.96	0.96
	Llama-3.1-70B 8-bit quant (base)	90	0.72	0.70	0.88	0.91	0.92	0.94	0.96
	Qwen2.5-7B (base)	90	0.63	0.71	0.92	0.94	0.94	**1.0**	**1.0**
	GPT 4o	98.5	**0.81**	**0.86**	0.87	0.87	0.87	**1.0**	**1.0**
	Gemini 2.0 Flash Lite	95.5	0.76	0.83	0.84	0.85	0.87	0.92	0.94
	Gemini 2.5 Flash Lite	97.8	0.80	0.84	0.88	0.85	0.85	0.96	0.96
	Instruction-tuned Llama-3.2-3B (ours)	96.4	0.74	0.72	0.86	0.91	0.9	0.9	0.91
	Instruction-tuned Llama-3.1-8B (ours, deployed)	97.6	0.79	0.8	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
	Instruction-tuned Qwen-2.5-7B (ours)	97.6	0.8	0.83	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
ATIS	CTRAN (Non-LLM)	98.07	-	-	-	-	-	-	-
	CoBiC (Non-LLM SOTA)	**99.43**	-	-	-	-	-	-	-
	Transformer based model (formerly deployed)	97.44	-	-	-	-	-	-	-
	Llama-3.1-8B (base)	84	0.72	0.74	0.91	0.92	0.87	0.89	0.83
	Llama-3.1-70B 8-bit quant (base)	86	0.76	0.78	0.91	0.92	0.89	0.89	0.86
	Qwen-2.5-7B (base)	84	0.75	0.76	0.91	0.92	0.88	0.89	0.85
	GPT 4o	98.5	**0.91**	0.85	**0.92**	**0.94**	0.92	0.92	**0.92**
	Gemini 2.0 Flash Lite	98	0.89	0.82	0.86	0.86	0.92	0.91	0.90
	Gemini 2.5 Flash Lite	98.8	**0.91**	0.86	0.86	0.88	**0.94**	0.90	**0.92**
	Instruction-tuned Llama-3.2-3B (ours)	96.5	0.89	0.86	0.80	0.86	0.90	0.88	0.90
	Instruction-tuned Llama-3.1-8B (ours, deployed)	98	0.90	0.88	0.85	0.90	0.90	0.92	**0.92**
	Instruction-tuned Qwen-2.5-7B (ours)	98	0.89	**0.89**	0.84	0.88	0.91	**0.93**	**0.92**
Proprietary	Transformer based model (formerly deployed)	97.3	-	-	-	-	-	-	-
	Llama-3.1-8B (base)	91	0.81	0.81	0.67	0.88	0.8	0.9	0.9
	Llama-3.1-70B 8-bit quant (base)	93	0.84	0.88	0.85	0.92	0.94	0.92	0.92
	Qwen-2.5-7B (base)	92	0.83	0.85	0.77	0.88	0.8	0.92	0.92
	GPT 4o	**99.1**	**0.94**	**0.87**	**0.88**	**0.96**	0.96	**0.98**	**0.98**
	Gemini 2.0 Flash Lite	97.5	0.92	0.84	0.85	0.96	0.94	0.94	0.92
	Gemini 2.5 Flash Lite	98.2	**0.94**	0.85	0.87	0.96	0.95	0.96	0.96
	Instruction-tuned Llama-3.2-3B (ours)	97.2	0.91	0.82	0.82	0.9	0.9	0.92	0.92
	Instruction-tuned Llama-3.1-8B (ours, deployed)	**99.1**	**0.94**	0.86	0.87	**0.96**	**0.98**	0.96	**0.98**
	Instruction-tuned Qwen-2.5-7B (ours)	98.8	0.93	**0.87**	0.87	0.95	**0.98**	0.97	0.96

Table 2: Comparison of small LLMs instruction-tuned using our framework against SOTA baselines. IC Acc denotes intent-classification accuracy on unambiguous queries. SDRS, ARS, Rel, Con, RA, Fac, and Pol refer to mean SDRS, mean ARS, and mean relevance, conciseness, repetition avoidance, factuality, and politeness scores across all assistant responses in the test set. Although the base Qwen-2.5-7B model reported stronger raw scores, the instruction-tuned Llama-3.1-8B and Qwen-2.5-7B performed similarly overall. We selected Llama for deployment due to proprietary and integration considerations.

- **Ambiguous Query Generation:** Llama-3.1-70B Instruct (8-bit quantized).
- **Conversation Generation for instruct tuning:** Llama-3.1-70B Instruct (assistant) and GPT-4o (Hurst et al., 2024) (user).
- **LLM-as-a-Judge:** GPT-4o for evaluating assistant responses with prompts detailed in Figures 6 and 7.

Our framework remains LLM-agnostic, allowing users to substitute any preferred LLM at each stage.

4 Dataset

To demonstrate the domain-agnostic nature of our framework, we benchmark instruction-tuned lightweight open-source LLMs against popular SOTA LLMs on both proprietary and public intent-classification datasets. For each open-source dataset, we use the standard train/validation/test splits for unambiguous queries, generate corresponding ambiguous queries, and evaluate on the combined test sets. Table 1 summarizes the datasets: SNIPS (Coucke et al., 2018), containing general voice-assistant commands, and ATIS (Hemphill et al., 1990), focused on airline travel. Although both have large training sets, we use only 1400 (SNIPS) and 1700 (ATIS) examples for instruction-tuning, as adding more provides minimal benefit. Our proprietary dataset comes from an e-commerce customer-care domain and unlike SNIPS and ATIS, includes a dedicated OOD class with 50 utterances unrelated to e-commerce (e.g., "book plane tickets" "apply for a passport"), used only for testing. To improve OOD handling, we generate 200 additional training examples using the label-dropout OOD method (Section 2.2.1) and instruction tune the model on this expanded dataset.

5 Results

We compare several low-capacity LLMs instruction-tuned using our framework against

Model	Input cost ($/M tokens)	Output cost ($/M tokens)	Cost/ query ($)	Monthly Cost ($)	Latency (s)
GPT 4o	2.5	10	0.00675	874.8K	2.4 - 2.6
Gemini 2.5 Flash-Lite	0.10	0.4	0.00027	35K	0.5 - 0.7
Gemini 2.0 Flash-Lite	0.075	0.3	0.00020	26.2K	0.6 - 0.8
Ours (instruction-tuned 8B LLM)	-	-	-	**3.2K**	0.9 - 1.2

Table 3: Realtime Inference Cost comparison. For our instruction-tuned open-source model, the cost involves running an A100 80GB GPU 24x7 to serve user requests at a throughput of 50 QPS.

strong baselines to assess ambiguity handling and cost efficiency. Baselines include:

- Non-LLM SOTA intent classifiers such as CTRAN (Rafiepour and Sartakhti, 2023), CoBiC (Kane et al., 2021) and our prior transformer-based model.
- Non-finetuned open-source LLMs (e.g., Llama-3.1-8B, Llama-3.1-70B, Qwen-2.5-7B) prompted for clarification.
- Proprietary LLMs such as GPT-4o and Gemini 2.0/2.5 Flash-Lite (Comanici et al., 2025).

We evaluate all systems using the Mean Strategic Dialogue Response Score (SDRS), the Mean Ambiguity Resolution Score (ARS), and mean scores for clarification-quality metrics. As shown in Table 2, instruction-tuned models match leading proprietary LLMs across most metrics, while some large non-tuned LLMs (e.g., Llama-70B) perform significantly worse, highlighting the benefit of our synthesized training conversations (Appendix A). Our models also perform comparably to non-LLM SOTA methods (CTRAN, CoBiC) on unambiguous intent prediction. Table 2 excludes OOD cases. Appendix F reports OOD results using prompts with an added OOD class. Models trained with our label-dropout OOD method (Section 2.2.1) show substantial OOD accuracy gains.

6 Realtime Inference Cost Analysis

To deploy open-source LLMs efficiently, we use TensorRT-LLM with the Triton inference server (Tillet et al., 2019). Our instruction-tuned models (3B–8B parameters) fit comfortably on a single NVIDIA A100 80GB GPU, achieving 50–75 QPS with sub-1-second latency. This makes them far more economical than proprietary alternatives. For example, Gemini 2.0 Flash-Lite, the least expensive proprietary LLM, costs over 8× more for comparable performance. A detailed monthly cost comparison at 50 QPS is provided in Table 3.

7 Deployment and Business Impact

We deployed our instruction-tuned 8B parameter LLM in the customer-care automation system of an e-commerce enterprise. In production, users proceed through a multi-step automated flow. At each step the system prompts the user, predicts an intent (with possible clarifying turns), and routes the user forward. Thus, clarifications directly impact routing accuracy and task-completion efficiency.

An A/B experiment comparing our tuned model against the existing transformer-based classifier shows that classification with clarification reduces the misrouting rate by 8%, enabled by the model's ability to solicit targeted clarifications before finalizing a routing decision. We also see a significant improvement in the automation rate, the fraction of queries resolved without human intervention, potentially translating to millions of dollars in annual savings by reducing human-agent contacts, which surpasses the additional GPU cost, which is minimal relative to the operational gains.

Clarifications and the increased model size introduced an increase in per-step latency. However, the overall end-to-end resolution time decreased by 19% due to fewer mispredictions and a reduction in downstream corrective steps. User-satisfaction metrics are not directly monitored; however, operational indicators suggest a more accurate and efficient automated experience.

8 Conclusion

In this work, we have introduced a framework that enables low-capacity LLMs to achieve classification-with-clarification performance comparable to high-capacity proprietary LLMs. A key contribution lies in our training strategy, which uses high-capacity LLMs to generate synthetic multi-turn conversational data from standard intent-tagged domain datasets, enabling efficient instruction-tuning of smaller models. We also propose a rigorous evaluation protocol that benchmarks our instruction-tuned models against state-of-the-art LLMs. Our results show that these models deliver strong performance while being over 8× more cost-efficient than even the most affordable proprietary alternatives. Furthermore, when deployed in a real-world customer-care system, our

model significantly increased automation rate and reduced operational costs, demonstrating its practical value in production environments.

9 Limitations

While the framework delivers strong performance and notable cost-efficiency, a few practical considerations remain. Data generation with high-capacity LLMs is a one-time offline step that amortizes well but still requires short-term access to stronger models, which may not be feasible for all practitioners. Our approach assumes the availability of domain-specific intent-tagged data and may benefit from light curation or semi-supervised augmentation in extremely low-resource settings. Also, our evaluation relies on an LLM-as-a-Judge for scalable assessment, which can introduce evaluator bias. As future work, we plan to complement this with targeted human spot-checks to increase transparency.

References

Neeraj Agrawal, Saurabh Kumar, Priyanka Bhatt, and Tanishka Agarwal. 2023. Hierarchical text classification using contrastive learning informed path guided hierarchy. In *ECAI 2023*, pages 19–26. IOS Press.

Andrea Alfieri, Ralf Wolter, and Seyyed Hadi Hashemi. 2022. Intent disambiguation for task-oriented dialogue systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 5079–5080.

Mohammad Aliannejadi, Leif Azzopardi, Hamed Zamani, Evangelos Kanoulas, Paul Thomas, and Nick Craswell. 2021. Analysing mixed initiatives and search strategies during conversational search. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 16–26.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, and Fei Huang. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Cen Chen, Chilin Fu, Xu Hu, Xiaolu Zhang, Jun Zhou, Xiaolong Li, and Forrest Sheng Bao. 2019. Reinforcement learning for user intent prediction in customer service bots. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1265–1268.

Maximillian Chen, Ruoxi Sun, Sercan Ö Arık, and Tomas Pfister. 2024. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. *arXiv preprint arXiv:2406.00222*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, and Evan Rosen. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, and Thibaut Lavril. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. 2005. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67.

Kaustubh D Dhole. 2020. Resolving intent ambiguities by retrieving discriminative clarifying questions. *arXiv preprint arXiv:2008.07559*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, and Alex Vaughan. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, and Honghao Liu. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Charles T Hemphill, John J Godfrey, and George R Doddington. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Floris den Hengst, Ralf Wolter, Patrick Altmeyer, and Arda Kaygan. 2024. Conformal intent classification and clarification for fast and accurate intent recognition. *arXiv preprint arXiv:2403.18973*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, and Alec Radford. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Bamba Kane, Fabio Rossi, Ophélie Guinaudeau, Valeria Chiesa, Ilhem Quénel, and Stéphane Chau. 2021. Joint intent detection and slot filling via cnn-lstm-crf. In *2020 6th IEEE Congress on Information Science and Technology (CiSt)*, pages 342–347. IEEE.

Kimiya Keyvan and Jimmy Xiangji Huang. 2022. How to approach ambiguous queries in conversational search: A survey of techniques, approaches, tools, and challenges. *ACM Computing Surveys*, 55(6):1–40.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2022. Clam: Selective clarification for ambiguous questions with generative language models. *arXiv preprint arXiv:2212.07769*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Fangwen Mu, Lin Shi, Song Wang, Zhuohao Yu, Binquan Zhang, Chenxue Wang, Shichao Liu, and Qing Wang. 2023. Clarifygpt: Empowering llm-based code generation with intention clarification. *arXiv preprint arXiv:2310.10996*.

Subhadip Nandi, Neeraj Agrawal, Anshika Singh, and Priyanka Bhatt. 2024. Enhancing customer service chatbots with context-aware nlu through selective attention and multi-task learning. In *Proceedings of the 8th International Conference on Data Science and Management of Data (12th ACM IKDD CODS and 30th COMAD)*, pages 220–228.

Matthew Purver, Jonathan Ginzburg, and Patrick Healey. 2003. On the means for clarification in dialogue. In *Current and new directions in discourse and dialogue*, pages 235–255. Springer.

Mehrdad Rafiepour and Javad Salimi Sartakhti. 2023. Ctran: Cnn-transformer-based network for natural language understanding. *Engineering Applications of Artificial Intelligence*, 126:107013.

Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. 2023. Gollie: Annotation guidelines improve zero-shot information-extraction. *arXiv preprint arXiv:2310.03668*.

Clemencia Siro, Mohammad Aliannejadi, and Maarten de Rijke. 2022. Understanding user satisfaction with task-oriented dialogue systems. In *Proceedings of the 45th International ACM SIGIR conference on research and development in information retrieval*, pages 2018–2023.

Leila Tavakoli. 2020. Generating clarifying questions in conversational search systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3253–3256.

Philippe Tillet, Hsiang-Tsung Kung, and David Cox. 2019. Triton: an intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, pages 10–19.

Mickey van Zeelt, Floris den Hengst, and Seyyed Hadi Hashemi. 2020. Collecting high-quality dialogue user satisfaction ratings with third-party annotators. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval*, pages 363–367.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Kuan Xu, Chilin Fu, Xiaolu Zhang, Cen Chen, Ya-Lin Zhang, Wenge Rong, Zujie Wen, Jun Zhou, Xiaolong Li, and Yu Qiao. 2020. admscn: a novel perspective for user intent prediction in customer service bots. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2853–2860.

Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of the web conference 2020*, pages 418–428.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. 2024. Clamber: A benchmark of identifying and clarifying ambiguous information needs in large language models. *arXiv preprint arXiv:2405.12063*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, and Eric Xing. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

A Ablation Study

We use the evaluation pipeline not only to benchmark instruction-tuned lightweight LLMs against proprietary baselines, but also to curate training data. From the synthetic conversations generated in Section 2.1, we select the highest-quality dialogues by SDRS and use them for instruction tuning. Unless stated otherwise, we retain the top 85% by SDRS. Table 4 shows that, on our proprietary dataset, retaining the top 85% of dialogues by quality yields the highest downstream SDRS for Llama-3.1-8B.

Training subset (top percentile by SDRS)	SDRS (instruction-tuned model)
100	0.83
90	0.89
85	**0.94**
80	0.90
75	0.87
65	0.81

Table 4: Effect of quality-based conversation filtering on instruction-tuned LLM performance

B Hyperparameters selected for Instruction tuning low capacity LLMs using our framework

- LoRA Alpha: 16,
- LoRA Dropout: 0.2,
- Rank: 16
- Learning Rate: 2×10^{-4},
- Batch Size: 8
- Epochs: 3,
- Optimizer: AdamW (Loshchilov and Hutter, 2017)
- Weight Decay: 0.001

C Structure of conversations used for instruction tuning

For conversations starting with unambiguous utterances:

1. System prompt and initial user query.
2. Model-predicted intent.

For conversations starting with ambiguous utterance:

1. System prompt and initial user query.
2. One or more clarification turns:
    - Assistant clarification question.
    - User clarification response.
3. Final predicted intent.

Using this structured conversation format we train lightweight LLMs perform classification with clarification.

D Turn level ambiguity resolution score calculation

Model Prediction	Enough Info for intent prediction	Not Enough Info for intent prediction
Intent	1	0
Clarifying Question	0	1

Table 5: Turn Level Ambiguity Resolution Score (ars_i)

E Turn-level response quality metrics

Response Quality Metrics	Evaluator Agent Possible Ratings	Rating Values
Relevance (r_i)	Not Relevant, Moderately Relevant, Relevant	0 / 0.5 / 1
Conciseness (c_i)	Not Concise or too concise, Adequately Concise	0 / 1
Repetition Avoidance (ra_i)	Repetitive, Somewhat Repetitive, Not Repetitive	0 / 0.5 / 1
Factuality (f_i)	Hallucinating, Not Hallucinating	0 / 1
Politeness (p_i)	Not Polite, Polite	0 / 1

Table 6: Turn-level Response Quality Metrics

F Performance on proprietary OOD data

Model	Accuracy %
Transformer based model (formerly deployed)	88
Llama-3.1-8B (base)	55
Llama-3.1-70B 8-bit quant (base)	64
Qwen-2.5-7B (base)	62
GPT 4o	86
Gemini 2.0 Flash Lite	82
Gemini 2.5 Flash Lite	86
Instruction-tuned Llama-3.2-3B (ours)	92
Instruction-tuned Llama-3.2-8B (ours, deployed)	94
Instruction-tuned Qwen-2.5-7B (ours)	94

Table 7: Performance on proprietary OOD data (50 examples)

G Prompt Templates

In this section, we have detailed the prompts that were used in our experiments.

Prompt: You are given a user query along with its correct intent. Your task is to modify this query so that it becomes *ambiguous*—meaning it could reasonably be classified into **two or more** of the intents listed below. You can make the query ambiguous by:

- Removing certain details that make the intent explicit.
- Adding elements that introduce plausible alternate interpretations.

List of intents with examples:
<List of Intents with examples>

Output Requirements:

1. Output must be a *single JSON dictionary* in the following format:

```
{
  "modified_query": "<modified
      query>",
  "potential_intents": [
    {"intent": "<intent_1>", "
        reason": "<reason_1>"},
    {"intent": "<intent_2>", "
        reason": "<reason_2>"}
  ]
}
```

2. `"modified_query"` is the newly generated ambiguous query.
3. `"potential_intents"` is a list of *all* possible intents this query could belong to, with each entry containing:
   - `"intent"` – the intent name.
   - `"reason"` – why the query could belong to this intent.
4. Do not include any extra explanations, formatting, or notation outside of the JSON.

Positive Example: *Input:* ⟨input⟩ *Output:* ⟨output⟩ This query is ambiguous because it could be interpreted as either `X` or `Y`; hence both intents are valid.
Negative Example: *Input:* ⟨input⟩ *Output:* ⟨output⟩ This query is *not* ambiguous because it is clear the user is asking for `X`, so the intent is unambiguously `X`.
Important: Do not fabricate ambiguity. Only modify the query in ways that naturally introduce multiple valid interpretations, each corresponding to a different intent from the provided list.

Figure 3: Prompt to generate an ambiguous query from an unambiguous query

Prompt: You are an empathetic, helpful, and intelligent conversational AI assistant. Your task is to identify the user's intent from the query provided.
Key Requirements:

1. If the query is clear and matches one of the intents, output the intent directly.
2. If the query is ambiguous and cannot be confidently mapped to a single intent, ask a *clarifying question*.
3. Clarifying questions must help disambiguate between the given intents and must not request irrelevant information.
4. If the query is clear but does not match any intent, output `"out of domain"`.
5. You will be heavily penalized for:
   - Asking a clarifying question when the intent is already clear.
   - Asking for information unrelated to intent disambiguation.

Available Intents (with examples): <List of domain intents with examples>
Output Rules:

- Output must contain *only* one of:
  1. An intent from the given list (if unambiguous).
  2. A clarifying question (if ambiguous).
  3. `"out of domain"` (if the query is not covered by any intent).
- Do not add extra characters, formatting, or explanations to the output.

Examples:
Example 1:

```
user: <unambiguous query from user>
assistant: <ground truth intent>
```

Example 2:

```
user: <ambiguous query from user>
assistant: <clarifying question>
user: <clarification from user>
assistant: <ground truth intent>
```

Figure 4: Prompt for Assistant Simulating LLM

Prompt: You are simulating a human user interacting with an AI assistant. You have a specific, clear, and unambiguous query in mind, but you will begin the conversation by expressing it in an *ambiguous* form.
Instructions:

1. Start the conversation by giving only the **ambiguous query**.
2. If the assistant asks a clarifying question, respond naturally as a human would, revealing information from your **unambiguous query**.
3. Only reveal details relevant to the clarifying question.
4. If the assistant asks for information you do not have, reply with: `"I do not know"`.
5. Do *not* ask clarifying questions back to the assistant; your role is to *answer* clarifying questions.

Your conversation setup:

- Initial ambiguous query:
  <Ambiguous query>
- Specific unambiguous query in mind:
  <Unambiguous query>

Figure 5: Prompt for User Simulating LLM

Prompt: You are given a conversation between a customer and an assistant bot. The assistant's objective is to determine the correct user intent from the following list:
<List of domain intents>

For each **user turn** (lines starting with `"user:"`), decide whether the conversation so far provides *sufficient information* to identify the correct intent.
For every turn, output exactly one of the following:

- `"enough information for intent identification"`
- `"not enough information for intent identification"`

Important:

1. Base your judgment only on the *user responses* up to and including the current turn.
2. The output must always be a valid Python list `[]` with one entry per user turn, in chronological order.

Example Output:
`["enough information for intent prediction", "not enough information for intent identification", ...]`

Figure 6: Prompt for evaluating assistant's Ambiguity Resolution capability

Prompt: You are given a conversation between a customer and an assistant bot. Your task is to evaluate the *quality* of each assistant response (prefixed with `"assistant:"`) based on the following parameters:

1. **Relevance:** Rate how well the response helps in disambiguating the precise intent of the user from a given list of potential intents. Possible values: `"very relevant"`, `"moderately relevant"`, `"not relevant"`.

2. **Conciseness:** A response is `"concise"` if it avoids unnecessary verbosity. Mark as `"not concise"` if it contains excessive or redundant information.

3. **Repetition Avoidance:** A response is repetitive if it asks the same clarifying question multiple times. Possible values: `"very repetitive"`, `"moderately repetitive"`, `"not repetitive"`.

4. **Factuality:** A response is `"hallucinating"` if it provides information that it should not or cannot know. Otherwise, mark as `"not hallucinating"`.

5. **Politeness:** Most responses should be marked `"polite"` unless they contain impolite or rude language, in which case mark as `"not polite"`.

Instructions: Rate *each* assistant response in the conversation individually. The output must always be a valid Python list `[]` with the same number of entries as there are assistant turns. Each entry should be a JSON object containing the above parameters as keys and the chosen rating as values.
Example Output:

```
[
  {
    "relevance": "very relevant",
    "conciseness": "concise",
    "repetition avoidance": "not repetitive",
    "factuality": "not hallucinating",
    "politeness": "polite"
  },
  {
    "relevance": "moderately relevant",
    "conciseness": "not concise",
    "repetition avoidance": "very repetitive",
    "factuality": "not hallucinating",
    "politeness": "polite"
  }
]
```

Figure 7: Prompt for evaluating the conversational quality of assistant-generated responses