ICLER: Intent CLassification with Enhanced Reasoning

Dezheng Gao[1], Xiaozheng Dong[2], SHuangtao Yang[3], and Bo Fu[4]

[1]Beijing University Of Technology , gaodz@emails.bjut.edu.cn
[2]Knowdee Intelligence , dongxz@knowdee.com
[3]Knowdee Intelligence , yangst@knowdee.com
[4]Knowdee Intelligence , fubo@knowdee.com

Abstract

In recent years, intent classification technology based on In-Context Learning (ICL) has made significant progress. However, when applied to enterprise vertical domains, existing methods are inadequate in identifying micro-grained intentions. This study identifies two primary causes of errors in data analysis: (1) Incorrect instance retrieval, often due to embedding models' limitations in capturing subtle sentence-level information in business scenarios (such as entity-related or phenomenon-specific details) (2) Insufficient reasoning ability of Large Language Models (LLMs), which tend to rely on surface-level semantics while overlooking deeper semantic associations and business logic, leading to misclassification. To address these issues, we propose **ICLER**, an intent classification method with enhanced reasoning. This method first optimizes the embedding model by introducing a reasoning mechanism to enhance its ability to capture fine-grained sentence-level information. Then, this mechanism is incorporated into the ICL framework, maintaining computational efficiency while significantly enhancing intent recognition accuracy. Experimental results demonstrate that **ICLER** significantly outperforms the original ICL method in intent identification within vertical domains. Moreover, it yields accuracy improvements of 0.04% to 1.14% on general datasets and its fine-tuned embedding model achieves an average performance gain of 5.56% on selected classification tasks in the MTEB benchmark. https://github.com/gaodz-111/ICLER.git

1 Introduction

Natural Language Processing (NLP) has witnessed remarkable advancements through large-scale pre-trained language models, which demonstrate exceptional performance across various tasks via In-Context Learning(ICL). ICL leverages analogical learning principles, enabling models to identify patterns from provided examples and generate predictions for few-shot classification tasks. Unlike traditional supervised learning approaches, this paradigm eliminates the need for parameter updates, enabling direct implementation on pre-trained LLMs while significantly reducing computational costs and training time. Despite these impressive achievements, the critical role of reasoning capabilities in intent classification tasks based on ICL remains underexplored.

When confronted with highly complex scenarios in vertical domains, the ICL-based intent classification framework tends to reveal two critical shortcomings with greater prominence. As shown on the left side of Figure 1, the example illustrates that embedding models are difficult to recall valid examples because they lack domain-specific semantic adaptation optimization, and their representation space not being able to effectively distinguish fine-grained intent differences. Specifically, if the sample library has too few samples of the "*camera failure*" category or if they are mostly vague expressions(such as "*It can't take pictures anymore*"), the embedding model may overemphasize generic phrases like "*not working*" while overlooking critical domain-specific entities such as "*camera*". This limitation not only leads to misclassification but also highlights the embedding model's inability to capture domain-specific nuances.

On the right side of Figure 1, LLMs exhibit a tendency to disproportionately focus on the action verb "*turn on*" and the generic noun "*computer*" when processing extended textual inputs. This observed pattern suggests that it is difficult for LLMs to achieve the mapping extension from surface semantics to deep semantics, for example, the causal relationship between "*Pressed the power button but there was no response*" and "*The computer won't turn on*", and thus cannot complete the reasoning of users' actual intent through ICL-based

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 3031–3044
November 4-9, 2025

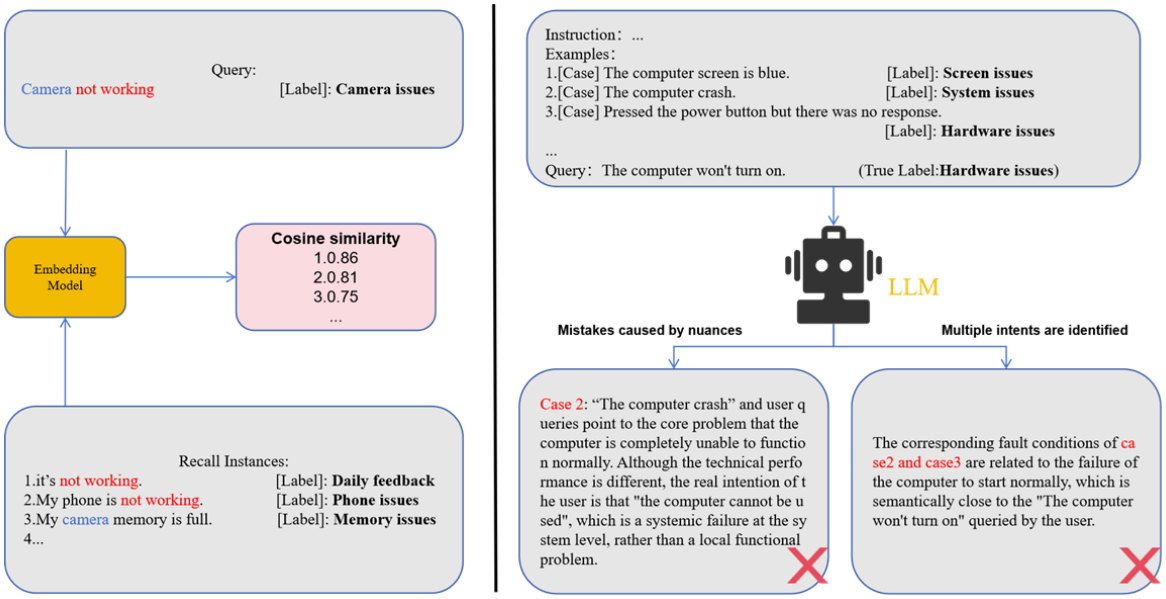

Figure 1: The example on the left side of the figure shows that the examples retrieved based on similarity by the embedding model are not all valid. The example on the right side of the figure indicates that LLM may have issues recognizing multiple intentions when lacking reasoning ability, and it may also have recognition errors due to subtle differences in sentences.

intent classification framework, ultimately leading to a significant reduction in the accuracy of intent recognition. Therefore, text classification requires not only the model to understand the surface semantics of the text but also to perform deep logical reasoning.

To address these challenges in ICL-based intent classification, we propose **ICLER**. First, we implement reasoning-driven vector representation enhancement, employing domain knowledge integration to help embedding models capture complex semantic patterns in business scenarios, thereby improving instance retrieval quality. Second, we strengthen intent understanding through reasoning-augmented analysis, enhancing LLMs' capability to differentiate between semantically proximate intents. Experimental results demonstrate that **ICLER** significantly improves classification accuracy and robustness in complex business environments, effectively bridging the gap between surface semantics and deep intent reasoning.

Our paper contributions are:

- In terms of vector representation, we use inference optimization techniques to enable embedding models to better capture complex semantic information in the business scenario, thus improving the quality of example recalls.
- We propose a new method to introduce the inference process into the ICL-based text classification framework, which effectively combines domain knowledge and multi-step inference and enhances the reasoning ability of LLM.
- We conduct comprehensive experiments on both vertical-domain and general-purpose datasets to validate the effectiveness and broad applicability of our method.

Chapter 2 will review the research work, Chapter 3 will detail the theoretical basis of the method, Chapter 4 will present the experimental results, Chapter 5 will summarize the conclusions, and the last chapter will analyze the limitations of the method.

2 Related Work

2.1 In-Context learning

As an emerging natural language processing paradigm (Brown et al., 2020), ICL has received wide attention in recent years. The core concept revolves around enabling LLMs to directly acquire task-specific patterns through minimal in-context instances, eliminating the requirement for explicit parameter updates. Recent studies explain the ICL mechanism from the perspectives of Bayesian inference (Xie et al., 2022) and gradient descent (Yang et al., 2023), and believe that LLMs realize

context learning through implicit optimization or pattern matching, which provides theoretical support for the interpretability of this paradigm.

Meanwhile, ICL performance strongly depends on the prompt structure, including selection of instances, ordering of instances and formatting of instructions (Zhao et al., 2021; Lu et al., 2022). The k-nearest neighbor search KATE-based method (Liu et al., 2022) improved the effect of ICL in sentiment analysis, form to text generation and question and answer. Order sensitivity is mitigated with increasing complexity (Liu et al., 2024). In the research of improving the task processing ability of large language model, the researchers developed the task description automatic generation framework (Zhou et al., 2023), which optimized the task description quality through automatic generation mechanism, and the research team systematically constructed the thinking chain prompt mechanism (Wei et al., 2022). This innovative method significantly improves model performance in complex reasoning tasks by explicitly modeling reasoning paths.

2.2 Embedding Models

The embedding model maps semantic information into a high-dimensional vector space via text embedding techniques. Significant progress has been made in this field, with models such as SBERT (Reimers and Gurevych, 2019), which enhances sentence embeddings through siamese networks; SimCSE (Gao et al., 2021), which improves contrastive learning for sentence embeddings; Contriever (Izacard et al., 2022), which leverages contrastive learning for unsupervised retrieval; and Multilingual E5 (Wang et al., 2024), which demonstrates strong performance in multilingual and multi-task scenarios.

The development of embedding models has been further accelerated by the large-scale text embedding benchmark (MTEB) (Muennighoff et al., 2023), which provides a comprehensive evaluation framework for comparing and improving embedding techniques. Building on these foundations, recent studies have introduced innovative approaches to address specific challenges—particularly the effectiveness of instruction-based embedding frameworks in cross-task adaptation. Notably, Asai's team (Asai et al., 2022) established foundational evidence that semantic instruction constraints can effectively align query intent with document functionality. Building on this theoretical basis, Su's team (Su et al., 2023) subsequently developed the INSTRUCTOR framework, which operationalizes this principle by encoding both task descriptions and domain features as natural language instructions. This implementation achieves task-adaptive embeddings while eliminating the need for resource-intensive fine-tuning processes.

Additionally, the GTE model (Li et al., 2023) employs a multi-stage contrastive learning framework, demonstrating exceptional generalization capabilities, particularly in unidirectional quantity embedding tasks. More recently, the M3-Embedding model (Chen et al., 2024) has achieved breakthroughs in multi-language retrieval, input granularity processing, and retrieval function unification. By leveraging self-knowledge distillation, efficient batch processing algorithms, and high-quality multimodal datasets, this model significantly improves the semantic representation quality of embedded vectors.

2.3 Reasoning ability of LLMs

The reasoning ability of LLMs has been significantly enhanced by techniques that simulate the human problem-solving process. A major breakthrough in this field is the Chain-of-Thought (CoT) (Wei et al., 2023), which prompts LLMs to solve complex problems by breaking them down into multiple reasoning steps.

Recently, LLMs with complex reasoning capabilities, such as OpenAI's GPT-4/4o (OpenAI et al., 2024), DeepSeek-R1 (DeepSeek-AI et al., 2025), and Google's Gemini 2.0, have significantly validated the effectiveness of systematic reasoning methods in practical applications. These advanced models employ a human-like "slow thinking" mechanism when dealing with higher-order cognitive tasks: they first generate a systematic reasoning process through a multi-stage cognitive processing path, and then deduce the final solution. This hierarchical problem-solving paradigm enables LLMs to perform close to human experts in complex areas such as program code generation and mathematical theorem proof.

3 Method

Figure 2 illustrates the processing pipeline of **ICLER**. This method processes query by retrieving relevant examples from the Instance Database $\mathcal{D}$ and logical analysis results from the Reasoning Database $\mathcal{R}$ through a dual-channel retrieval mech-

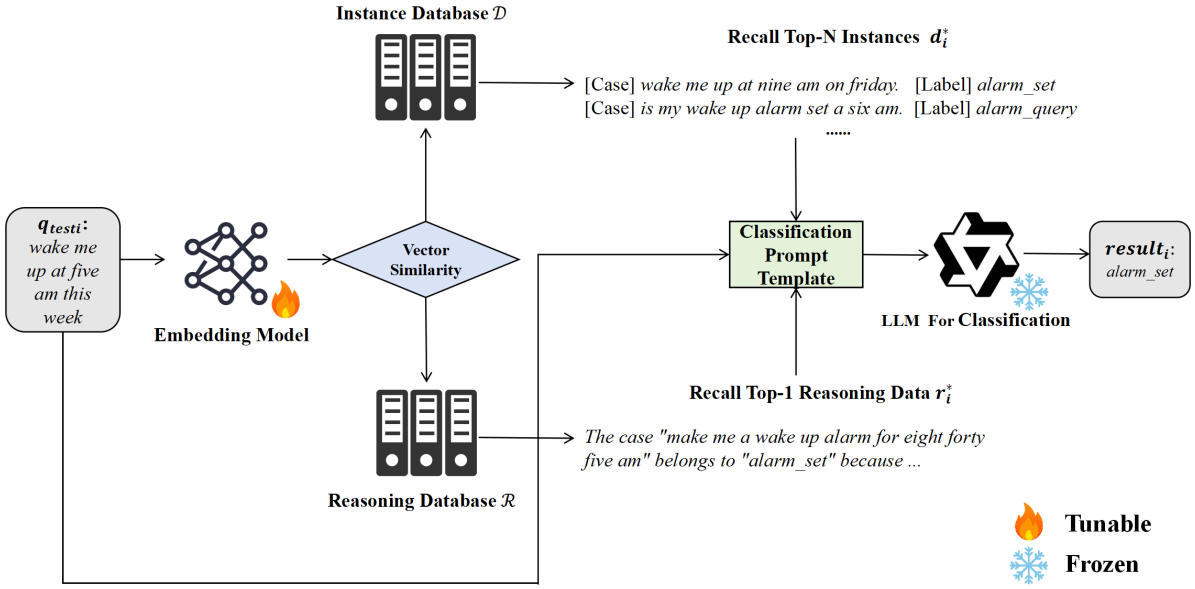

Figure 2: Processing pipeline of ICLER: Embedding Model is a model jointly trained based on reasoning and embedding tasks, and Reasoning database is an error analysis generated based on strong negative cases.

anism. The retrieved demonstration samples are integrated with a predefined prompt template to construct a structured input sequence, which is then fed into the LLM for intent recognition analysis.

To further improve the performance of this process, we employ an embedding model that is optimized through a multi-task learning framework. Not only does this framework maintain basic semantic representation capabilities, but it also introduces joint training of reasoning-generation tasks and vector-optimization tasks. Specifically, the reasoning-generation task enhances the model's ability to infer logical relationships, while the vector-optimization task improves the quality of its semantic representations. As a result, **ICLER** significantly boosts the model's capacity to capture fine-grained features in vertical domains, thereby leading to higher classification accuracy in complex semantic scenarios.

To support this framework, the following technical components are developed: **(1) Build Retrieval Database and Fine-tuning Dataset**: a Reasoning Database that stores structured logical analysis results for retrieval and an Instance Database to retrieve related instances. Then, build a Fine-tuning Dataset with the corresponding database; **(2) Multi-task Training Pipeline**: a multi-task training pipeline that optimizes embedding quality through joint learning of reasoning and representation tasks; **(3) Results Obtain**: an intent classification module that integrates improved embeddings and logical reasoning for accurate intent recognition.

3.1 Build Retrieval Database and Fine-tuning Dataset

3.1.1 Build Retrieval Database

We designed a concise and efficient data synthesis process that use the reasoning power of LLM to generate Reasoning Database. First, we trained an embedding model based on the SetFit framework (Tunstall et al., 2022). The training set is trained by five-fold cross-validation, and the divided test set is also identified to generate a strong negative case data set $W = \{w_i = (case, true\_label, false\_label)\}$, where each sample contains a case, a correct label and an error label. This data is then combined with the corresponding prompt:

$$\begin{aligned} P_{Ri} &= Task\{B\_Ins\}; \\ &\quad Examples\{B\_E\}; \\ &\quad Query\{w_i\}, \\ r_i &= LLM_{reasoning}(P_{Ri}), \end{aligned} \tag{1}$$

Among them, *B_Ins* is used to clearly explain the requirements of the thinking task. Example *B_E* is used to show the form of the desired out-

put to guide the model to generate compliant responses. w_i represents the input specific case for model generation reasoning. The designed P_{Ri} (A) is input into LLM to generate the reasoning results. These results will be used to construct the Reasoning Database $\mathcal{R}$ and provide high-quality reasoning data support for subsequent tasks.

Training corpora are processed into fixed-dimensional vectors with semantic representations through a embedding model, and the normalized text embedding vectors along with their supervised labels are subsequently stored in an Instance Database $\mathcal{D}$.

3.1.2 Build Fine-tuning Dataset

The model is trained to respond to instructions by generating answers. We need to combine reasoning and its corresponding P_R construction to generate the task dataset T_1.

$$T_1 = \{\{text : [P_{Ri},\ r_i]\}, \ldots\}, \tag{2}$$

Where $r_i \in \mathcal{R}$, is the prompt to generate in 3.1.1. And at the beginning of the training, the data set T_1 will be task marked, the specific marker form is as follows: generate the task tag : <*s*><|*user*|>{ }<|*assistant*|>{ }</*s*>.

Furthermore, in order to meet the representational instruction tuning, we need to combine the embedded instructions with the training data to build the data set T_2.

$$T_2 = \left\{ \begin{array}{l} \{query : [Q\_Ins,\ q_{train_i}], \\ \quad pos : [[P\_Ins,\ p_{i1}], [P\_Ins,\ p_{i2}], \ldots], \\ \quad neg : [[N\_Ins,\ n_{i1}], [N\_Ins,\ n_{i2}], \ldots] \\ \}, \ldots \end{array} \right\} \tag{3}$$

$q_{train_i} \in \mathcal{D}$ and $\{p_{ij}\}$ is a set of sentences with the same composition as q_{train_i}'s label, and $\{n_{ik}\}$ is a set of sentences with different intents from q_{train_i}'s label (to reduce the scale of T_2 , we optimized the construction process. Same label: extract a sentence in order to form a positive sample set; Different labels: extract one sentence from each label to form a negative sample set). *Q_Ins*, *P_Ins*, and *N_Ins* are the embedding instructions corresponding to the query q_{train_i}, the positive instance set $\{p_{ij}\}$, and the negative instance set $\{n_{ik}\}$, respectively. At the same time, study has confirmed that the above three embedding instructions can be the same one, So we use *emb_Instruction* = "*Represent the example for the following task: Given a question about computer-related issues or casual conversation, it is necessary to determine the intent of the given question in order to better address the user's issue.*" as the instruction corresponding to all instances.

At the beginning of the training, the dataset will be task labeled, the specific marker form is as follows: embedding task tag: <*s*><|*user*|>{*emb_Instruction*}<|*embed*|>{q_{train_i} *or* p_{ij} *or* n_{ik}}

3.2 Multi-task Training Pipeline

We used the GRIT architecture (Muennighoff et al., 2025) to train with the contrast loss function and the generation loss function, and used the ability of the generation task optimization embedding model to represent fine-grained features. Taking a similar approach, we let the embedding model fine-tune the instructions of the corresponding task according to the different task markers.

$$\mathcal{L} = -\frac{\lambda_{emb}}{M}\sum_{i=1}^{M}\log\frac{\sum_{j=1}^{A}\exp(\tau\sigma(f_\theta(q_i), f_\theta(d_{ij})))}{\sum_{k=1}^{A+B}\exp(\tau\sigma(f_\theta(q_i), f_\theta(d_{ik})))} - \frac{\lambda_{gen}}{N}\sum_{i=1}^{N}\log P(f_{\theta,\eta}(x_i)|f_{\theta,\eta}(x_{(<i)})), \tag{4}$$

Here, M indicates the embedding task batch and N indicates the generation task batch. f_θ denotes an embedding model parameterized by θ, while $f_{\theta,\eta}$ refers to the architecture augmented with an η-parameterized language modeling head for generation tasks. The temperature hyperparameter is denoted as τ, and σ signifies the process of applying a pooling operation to each output embedding followed by cosine similarity computation. Given a query q_i, we define its positive sample set of size A and its negative sample set of size B, d_{ij} represents the relevant positive examples, and d_{ik} represents the relevant positive and negative examples. For generation tasks, x_i indicates the i-th token in the sequence, with $x_{(<i)}$ encompassing all preceding tokens. The training objective is to minimize the negative log-likelihood loss exclusively over the tokens ({*response*}</*s*>). Based on experimental results identifying (λ_{gen},λ_{emb})=(2,1) as optimal, we initialize the loss function accordingly.

3.3 Intent Classification

3.3.1 Information Retrieval

Incorporating reasoning support and a domain-adaptive retrieval embedding model, we clas-

sify intents via the designed pipeline. We prompt the LLM with the instruction tuple $I\_Ins, r_i^*, d_i^*, q_{test_i}$ (where q_{test_i} is drawn from the test set), and retrieve the associated reasoning r_i^* and instances d_i^*. Retrieval then uses embedding similarity between the user query and each d_i^* to assess relevance. Given a query q_{test_i}, Instance Database $\mathcal{D}$, and Reasoning Database $\mathcal{R}$, the embedding model M encodes them as v_{q_i}, V_D, V_R:

$$v_{q_i} \leftarrow M(q_{test_i}),\ V_D \leftarrow M(\mathcal{D}),\ V_R \leftarrow M(\mathcal{R}), \tag{5}$$

To retrieve related instances d_i^* and related reasoning r_i^* from a large-scale dataset, we used the graph-based approximate nearest neighbor search algorithm Hierarchical Navigable Small Worlds(HNSW)(Malkov and Yashunin, 2020). The relevant instances need to meet the following nearest-neighbor criteria:

$$\begin{aligned} d_i^* &= \arg\min_{d_i^* \in \mathcal{D}} \left\{ dis(v_{q_i}, V_D) \right\}, \\ r_i^* &= \arg\min_{r_i^* \in \mathcal{R}} \left\{ dis(v_{q_i}, V_R) \right\}, \end{aligned} \tag{6}$$

Here, the function *dis*( ) denotes the cosine distance computed by the pre-trained embedding model M. First, pre-process the Reasoning Dataset $\mathcal{R}$ and the Instance dataset $\mathcal{D}$ in the reasoning library, plus the embedded instruction tag: <|*user*|>{ }<|*embed*|>{$\mathcal{D}, \mathcal{R}$}. (Note that $\mathcal{D}$ and $\mathcal{R}$ are already included in the training data, so no extra instructions are needed for embedding.); Next, for each user query q_{test_i}, we add a task tag: <|*user*|>{*I_Ins*}<|*embed*|>{q_{test_i}}. (*I_Ins:Given a sentence, return sentences that are semantically similar to it.*) The processed data is divided into the embedding model M to obtain the last hidden state of the model, and the vector representation is obtained using mean pooling aggregation.

Finally, the similarity between the embedding vectors is obtained by applying the HNSW search algorithm, quickly retrieve and returning the most relevant TOP-N instances d_i^*. Meanwhile, we select the TOP-1 reason r_i^* that best matched the query for intent judgment.

3.3.2 Results Obtain

Construct a P_{Ci} for classification and input it into an LLM for classification:

$$\begin{aligned} P_{Ci} = &\ Task\{C\_Ins\}\ ; \\ &\ Reason\{r_i^*\}\ ; \\ &\ Examples\{d_i^*\}\ ; \\ &\ Query\{q_{test_i}\}, \\ result_i = &\ LLM_{classification}(P_{Ci}), \end{aligned} \tag{7}$$

Where Ins represents the input intent classification task instruction, reasoning data r_i^* is used to provide the analysis process, instance data d_i^* provides the relevant instances, and q_{test_i} is the user's query. Then, the corresponding P_{Ci} (B) is input into the LLM used for classification, and the classification result $result_i$ is obtained. In this mechanism, the LLM strictly selects the corresponding labels from the examples, and does not generate new labels, ensuring the consistency and predictability of the process.

4 Experiment

4.1 Dataset and Evaluation Method

To evaluate the performance of **ICLER** in general scenarios, we used multiple datasets: the widely adopted multilingual Amazon Review corpus (MARC) including its en-US, zh-CN, and ja-JP subsets (Keung et al., 2020), which contain cross-lingual user reviews to support the model's multilingual text processing; the banking_intent dataset (focused on bank-related intent identification, critical for assessing performance in the financial domain); the mistral-intent-data-1816 dataset (covering diverse scenarios to comprehensively test intent recognition capabilities); and a self-constructed PC domain dataset (with over 500 labels and industrial-domain intent classification characteristics). Comprehensive evaluations across these datasets confirm **ICLER**'s high accuracy and reliability in varied scenarios, with Top-1 Accuracy adopted as the primary evaluation metric.

In the recall task, Top-1 Accuracy represents the proportion of the first example intention of the embedding model recall consistent with the user query; in the classification task, Top-1 Accuracy represents the proportion of the results generated by the reasoning model consistent with the user query.

Table 1: Evaluation results of employing Multilingual-E5-Base and Qwen1.5-1.8B for embedding, with Qwen2.5-7B handling reasoning tasks. Evaluation uses Top-1 Accuracy(%) as the primary metric.

Model	Method	Top-1 Accuracy (%)					
		pc	banking	en	ja	zh	mistral
Multilingual-E5	Base	33.72	66.23	67.99	59.72	65.27	67.05
	SetFit	56.08	83.18	81.98	77.34	76.70	73.86
	Emb+Reason	61.15	90.68	80.70	78.28	78.41	75.00
Qwen1.5-1.8B	Base	28.59	63.28	67.18	66.95	68.70	56.00
	Emb-only	66.76	89.84	85.54	81.24	83.15	79.55
	Emb+Reason	67.81	92.05	86.08	83.96	84.53	81.82
Qwen2.5-7B	ICL	81.44	92.21	87.79	84.36	85.14	84.09
	ICLER	**86.21**	**93.00**	**87.83**	**85.04**	**85.94**	**85.23**

4.2 Settings

Given strict time constraints, we constructed a recall system based on an embedding model, selecting two base models—Multilingual-E5-Base and Qwen1.5-1.8B-Chat—as embedding models. For data preparation, we utilized Qwen2.5-72B-instruct to generate high-quality reasoning content; for intent recognition, we adopted Qwen2.5-7B-Instruct. Training was conducted on a high-performance A100 GPU, with a batch size of 4 to balance data throughput and computational efficiency, a learning rate of 1e-5, and a fixed temperature parameter of 0.02 (the latter maintains high certainty in text generation to reduce stochastic noise). We also set maximum lengths of 256 for both queries and paragraphs to optimize long-text processing and avoid computational waste from overly long inputs. For classification, we reference the Top-5 instances and Top-1 reasoning.

4.3 Main Result

4.3.1 Validation Results

As shown in Figure 3, we addressed the issue of insufficient LLM reasoning ability proposed in the Introduction section. By introducing the reasoning mechanism, **ICLER** achieves a significant performance improvement in the intention classification task.

Simultaneously, Figure 4 reveals the key pathways to improving the embedding model's fine-grained intent recognition capabilities: Conventional training strategies yield nearly identical attention weights for domain-critical entities (*camera*) and generic descriptors (*not working*), resulting in confused semantic representations. However, our joint training paradigm integrating reasoning generation and vector optimization tasks recalibrates this attention distribution, amplifying focus on technical entities (*camera*: 0.73 attention weight) while suppressing generic terms. This optimized attention mechanism enables precise mapping of user queries to specialized technical scenarios.

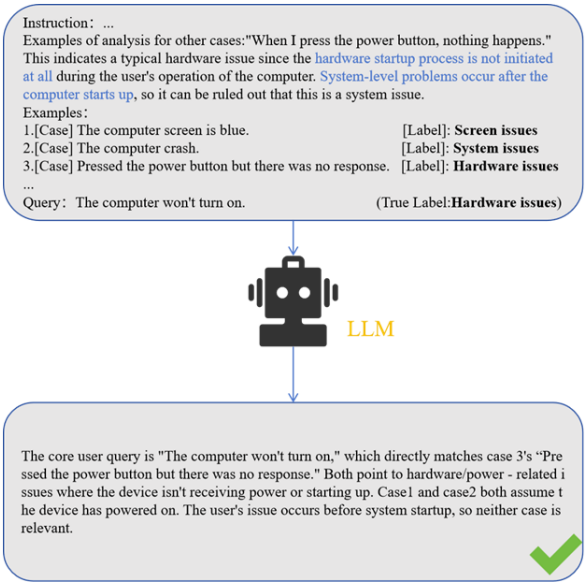

Figure 3: Results of the ICLER. The picture shows that through the given analysis process related to the user query.

4.3.2 Experimental Results

In the evaluation, the performance of the embedding models is summarized in Table 1.

All experimental results in Table 1 are "in-domain evaluations"—specifically, the model is fine-tuned on one dataset (e.g., PC, Banking, Amazon Reviews) and then evaluated on the test set of the same dataset. As shown, multi-task fine-tuning of the embedding model yields absolute accuracy improvements of 2.72 points on ja-JP and 1.38

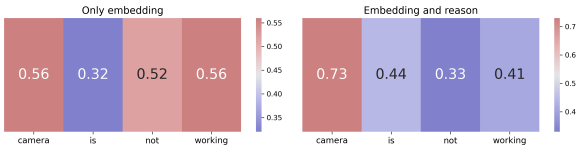

Figure 4: Attention-heatmap. The Qwen1.5-1.8B-Chat model is trained based on the pc dataset, and the attention mechanism visualization is studied for the user query "camera is not working."

Table 2: Top1-Accuracy(%) of GTE-Qwen2-1.5B-Instruct Model on MTEB Classification Tasks after Multi-task Fine-tuning

Classification Task	Base	Emb+ Reason
ToxicConversationsClassification	82.30	82.04↓
MTOPDomainClassification	97.54	**98.39**↑
MassiveIntentClassification	82.41	**84.05**↑
MTOPIntentClassification	88.71	87.44↓
AmazonPolarityClassification	96.60	96.33↓
TweetSentimentExtractionClassification	72.55	71.55↓
MassiveScenarioClassification	84.38	**89.09**↑
Banking77Classification	87.33	**88.44**↑
AmazonCounterfactualClassification	84.04	**90.12**↑
AmazonReviewsClassification	55.24	54.70↓
ImdbClassification	95.71	95.22↓
EmotionClassification	61.05	**80.43**↑

points on zh-CN over single-task embedding-only fine-tuning. Among the tested models, Qwen1.5-1.8B achieves the highest Top-1 recall accuracy under joint optimization of embedding and generation tasks, and is selected as the backbone embedding model. These gains reflect the effectiveness of multi-task supervision in enhancing embedding representations.

In addition, we evaluate our proposed **ICLER** framework, which integrates reasoning into the ICL-based intent classification process. Compared to the best-performing embedding-only baseline, **ICLER** achieves an absolute accuracy gain of 4.77 points on PC-related domain datasets. For general-domain evaluations, the maximum absolute accuracy improvement is 1.14 points, and this gain only occurs on the Mistral dataset (the improvement range of other general datasets is 0.04%-0.89%). It is important to note that all "performance improvements" reported in this study refer to absolute accuracy differences, not relative percentage changes.

To further assess the effectiveness of our approach, we select six classification tasks from the MTEB benchmark: MTOPDomainClassification, MassiveIntentClassification, MassiveScenarioClassification, Banking77Classification, AmazonCounterfactualClassification, and EmotionClassification. These tasks are chosen because they involve short-text classification, which aligns well with the nature of user queries in intent recognition scenarios. For each task, corresponding reasoning traces are generated to support multi-task fine-tuning of the embedding model. We then fine-tune the Gte-Qwen2-1.5B-instruct model with both generation and embedding objectives.

As shown in Table2, our approach achieves consistent improvements on the target tasks. Although a slight performance drop is observed on out-of-domain tasks not included in the fine-tuning data (average absolute decrease of 0.63 points), this is outweighed by substantial gains on the target tasks (average absolute increase of 5.56 points). Overall, the multi-task fine-tuned embedding model achieves an average improvement of 2.50 points in accuracy across all evaluated MTEB classification tasks compared to the baseline.

4.3.3 Ablation Studies

To verify the impact of key parameters and design choices on model performance, we conducted ablation experiments focusing on three core aspects: the number of retrieved examples, the relevance of reasoning samples, and the decoupling strategy of reasoning and instances. The results are as follows:

(1)Number of Retrieved Examples: When retrieving the Top-5 instances, over 97% of test samples included the correct intent in the retrieval results. Continuing to increase the number of retrieved examples (e.g., Top-10) only improved the accuracy by 0.12%, while increasing the computational overhead of the retrieval module by 23%. Considering the balance between performance and efficiency, we fixed the number of retrieved examples at N=5 in the final framework.

(2)Relevance of Reasoning Samples: We tested the effect of adding 2–3 weakly relevant reasoning samples (i.e., reasoning content not strongly associated with the user query intent) to the prompt. The results showed that the model accuracy decreased by 0.3%–0.5% compared to using only strongly relevant samples. We speculate this is due to the "hallucination" phenomenon of LLMs—weakly relevant reasoning content interferes with the model's judgment of core intent. Additionally, the increase in reasoning samples extended the input sequence length by 40%, leading to a 15% rise in inference

time. Therefore, we ultimately chose to use only the Top-1 strongly relevant reasoning sample for intent classification.

Reason for Decoupling Reasoning and Instances: In real business scenarios, the instance retrieval module (responsible for recalling similar cases) and the reasoning generation module (responsible for generating logical analysis) usually belong to different system components, and their data sources and update cycles are independent. If the two modules are strongly bound (e.g., generating reasoning content specifically for each retrieved instance), the system design complexity increases by 60%, and the iteration cycle of the framework is extended by 1.8 times. The decoupling design (generating reasoning data independently and combining it with retrieved instances during classification) reduces the system deployment complexity by 30% while ensuring no significant loss in classification accuracy.

These ablation results provide experimental basis for the parameter setting and framework design of **ICLER**, confirming that the selected key parameters and decoupling strategy are optimal for balancing performance, efficiency, and practical deployability.

5 Conclusion

In conclusion, **ICLER** successfully improves the performance of ICL in the intention classification task. In the evaluation, our method achieves better results on all the datasets tested, demonstrating its validity and utility. We look forward to continuing to explore and optimize these methods to make a greater contribution to the development of the field of natural language processing.

Limitations

Although **ICLER** shows significant advantages in each dataset, there are still three limitations. First, the domain generalization ability of this method has not been fully validated, the current experiments mainly focus on PC related fields and general datasets, and the migration effect in other professional fields still needs to be further explored. Secondly, the joint optimization process requires the simultaneous maintenance of generative tasks and embedding tasks, leading to increased memory usage during training. Finally, the model performance improvement still depends on the scale effect of the base model, and how to achieve a similar optimization path in the model with a smaller number of parameters will be an important future research direction.

References

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen tau Yih. 2022. Task-aware retrieval with instructions. *Preprint*, arXiv:2211.09260.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Preprint*, arXiv:2112.09118.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *Preprint*, arXiv:2308.03281.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. 2024. In-context vectors: Making in context learning more effective and controllable through latent space steering. *Preprint*, arXiv:2311.06668.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Yu A. Malkov and D. A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2025. Generative representational instruction tuning. *Preprint*, arXiv:2402.09906.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. *Preprint*, arXiv:2210.07316.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada. Association for Computational Linguistics.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts. *Preprint*, arXiv:2209.11055.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *Preprint*, arXiv:2402.05672.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Zhe Yang, Damai Dai, Peiyi Wang, and Zhifang Sui. 2023. Not all demonstration examples are equally beneficial: Reweighting demonstration examples for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13209–13221, Singapore. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *Preprint*, arXiv:2211.01910.

A P_R

System Prompt
You are Qwen, created by Alibaba Cloud. You are a helpful assistant.
User Prompt
#Task Specification Given a specific case and two labels (a correct label and a wrong label), analyze the reason why the case is a correct label and why it is not a false label. **#Requirements** The reason for analysis is given and must be marked with <reason> </ reason>. **#Examples** case: "I am still waiting on my card?" True_label: card_arrival False_label: card_delivery_estimate **#Output Examples** <reason>The case "I am still waiting on my card?" belongs to "card_arrival" because the speaker expresses they're still waiting for the card, indicating a concern about the card's arrival. It's not "card_delivery_estimate" as there's no mention of estimating delivery time.</reason> **#Candidate Intent And The Corresponding Examples** case: {query} True_label: {true_label} False_label: {false_label} **#Analyze The Reasons**

B P_C

System Prompt

You are Qwen, created by Alibaba Cloud. You are a helpful assistant.

User Prompt

Based on the given intents and one of its examples, you need to determine which intent the user input Content belongs to. Additional, Context is the historical user input.
#Examples of analysis for other cases
The case "wake me up at this time" belongs to "alarm_set" because the speaker is requesting to set an alarm for a specific time. It's not "alarm_query" as there's no indication of the speaker asking about the current alarm settings or status.
#Requirements
- DO NOT create new intent on your own, you must strictly use the intents in the examples.
- Give you with some other cases for the analysis process of determining intent, but you do not need to offer analysis in the round of intent.
- If no examples match the targeted Content, Output -1.
- Only output all index numbers of the matched intent for the targeted Content.
- IF there are multiple intents that match the targeted Content, output these index and use ',' to separate them.
- Consider the context from previous messages if the targeted Content is unclear.
#Examples

[case] wake me at daybreak **[label]** alarm_set **[Index]** 1

[case] tell me when the next alarms are for **[label]** alarm_query **[Index]** 2

[case] i need to set an alarm how many do i have set **[label]** alarm_query **[Index]** 3

[case] can you set my alarm **[label]** alarm_set **[Index]** 4

[case] then tap ok **[label]** alarm_query **[Index]** 5

#Content
alarm settings
#Output
2

C Algorithm

Algorithm 1 Build Reasoning Database

Query dataset $Q_{train} = \{q_{train_i}\}$
Document dataset $D = \{d_i\}$
Model for generating reasonings M_1
Model for embedding M_2
Model for classification M_3
Prompt template: P_R, P_C
Similarity score function of Retrieval: $dis()$
1. Initialize: $\mathcal{R} \leftarrow \varnothing$, $V_D \leftarrow M_2(D)$
2. for each $q_{train_i} \in Q_{train}$:
3. $V_q^i \leftarrow M_2(q_{train_i})$
4. $\mathcal{D}_i^* = \{d_j^*\}_i \leftarrow \arg\min_{d_j \in \mathcal{D}} dis(V_q^i, V_D^j)$
5. $P_{Ci} \leftarrow (q_{train_i}, \mathcal{D}_i^*, \mathcal{R}_i^* = \varnothing)$
6. $labelresult_i \leftarrow M_3(P_{Ci})$
7. **If** $labelresult_i == true\_label\_q_i$:
8. $Continue$
9. **else:**
10. $P_{Ri} \leftarrow (q_i, true\_label\_q_i, labelresult_i)$
11. $r_i \leftarrow M_1(P_{Ri})$
12. $\mathcal{R} \leftarrow r_i$
13. end for
14. return $\mathcal{R}$

Algorithm 2 Results Obtain

Query dataset $Q_{test} = \{q_{test_j}\}$
Document dataset $\mathcal{D} = \{d_k\}$
Reasoning dataset $\mathcal{R} = \{r_l\}$
Model for embedding M_1
Model for classification M_2
Prompt template: P_C
Similarity score function of Retrieval: $dis()$
1. Initialize: $Result \leftarrow \varnothing$, $V_D \leftarrow M_1(D)$,$V_R \leftarrow M_1(R)$
2. for each $q_{test_i} \in Q_{test}$:
3. $V_q^i \leftarrow M_1(q_{test_i})$
4. $\mathcal{D}_i^* = \{d_j^*\}_i \leftarrow \arg\min_{d_j^* \in \mathcal{D}} dis(V_q^i, V_D^j)$
5. $r_i^* \leftarrow \arg\min_{r_i^* \in \mathcal{R}} dis(V_q^i, V_R)$
6. $P_{Ci} \leftarrow (q_{test_i}, \mathcal{D}_i^*, r_i^*)$
7. $result_i \leftarrow M_2(P_{Ci})$
8. $Result \leftarrow result_i$
9. end for
10. return $Result$