SupportBot: Continuous Case Mining for Grounded Technical Support Automation

Anonymous ACL submission

Abstract

We present **SupportBot**, a technical support automation system that continuously mines solved cases from community conversations to enable grounded response generation. Unlike traditional retrieval-augmented generation (RAG) systems that operate over static document collections, SupportBot dynamically extracts structured problem-solution pairs from ongoing chat streams, indexes them for semantic retrieval, and uses them alongside documentation to generate cited responses.

We evaluate SupportBot on 1,745 real support messages from a private technical community. Our ablation study reveals that retrieval source composition critically determines answer quality: documentation-only retrieval achieves 35.0% accuracy, while adding mined conversation history improves accuracy to 96.9%. The full system combining documentation, mined cases, and conversation context achieves 75.8% accuracy with an average judge score of 7.23/10. These results demonstrate the value of treating community conversations as a continuously growing knowledge base rather than ephemeral communication.

1 Introduction

Every day, technical communities solve hundreds of problems that never make it into official documentation. A user reports a cryptic error, an expert suggests checking a configuration file, the user confirms it worked—and this valuable exchange sits in a chat log, invisible to the next person facing the same issue. Meanwhile, automated support systems continue querying static documentation that cannot capture these real-world solutions.

Standard retrieval-augmented generation (RAG) pipelines (Lewis et al., 2020; Guu et al., 2020; Borgeaud et al., 2022) assume access to a well-structured, static document corpus. This assumption breaks down in technical support contexts where:

- Critical knowledge exists only in chat messages
- Solutions emerge through multi-turn conversations
- Confirmation signals indicate which solutions actually worked
- New issues and fixes arise faster than documentation updates

SupportBot addresses these challenges through a **case mining architecture** that treats community conversations as a continuously growing knowledge base. The system operates through two coupled processes: (1) an offline mining pipeline that extracts structured cases from solved conversations, and (2) an online response pipeline that retrieves from mined cases alongside static documentation.

Our contributions are:

1. **Continuous case mining**: An architecture that detects solved conversational arcs in real-time and extracts structured problem-solution pairs for indexing.
2. **Multi-source grounded generation**: A response pipeline that retrieves from mined cases, documentation, and conversation context, with citations to source material.
3. **Empirical validation**: Evaluation on 1,745 real support messages showing that conversational knowledge retrieval (97% accuracy) dramatically outperforms documentation-only retrieval (35% accuracy).

2 System Architecture

Figure 1 illustrates the SupportBot architecture, which consists of two asynchronous but coupled processes that share a common case index.

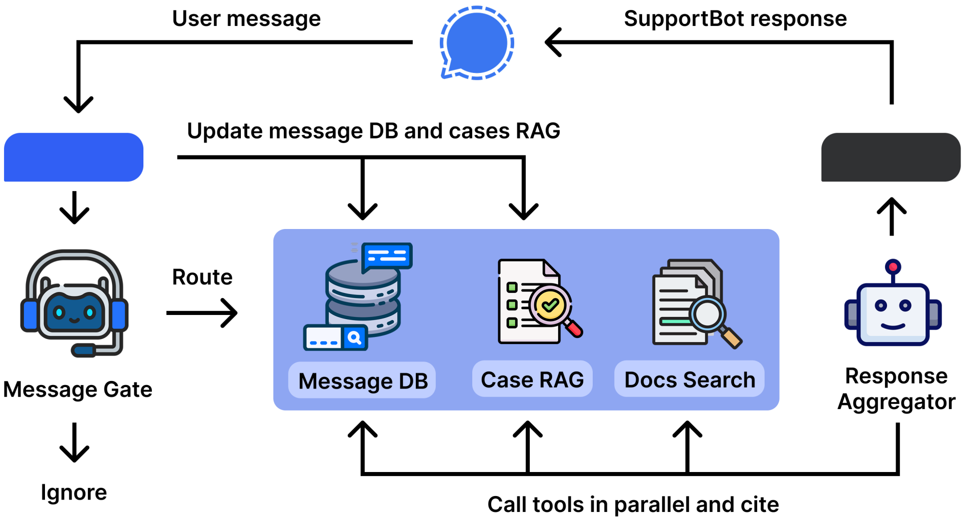

Figure 1: SupportBot architecture overview. The system consists of two coupled processes: continuous case mining from the message stream, and online response generation with multi-source retrieval.

Algorithm 1 Continuous Case Mining

Require: Message buffer B, case index $\mathcal{K}$

for each message m arriving in stream **do**
  $B \leftarrow B \cup \{m\}$
  if DetectSolvedArc(B) **then**
    $c \leftarrow$ ExtractCase(B)
    $\mathcal{K} \leftarrow \mathcal{K} \cup \{c\}$
    $B \leftarrow B \setminus \text{messages}(c)$ ▷ Optional cleanup
  end if
end for

2.1 Case Mining Pipeline

The case mining pipeline monitors incoming messages and identifies solved conversational arcs that can be converted into structured cases.

Algorithm 1 shows the mining procedure. For each incoming message, the system checks whether recent conversation forms a completed problem-solving arc. Solved status is determined through:

- Explicit confirmation signals (“fixed”, “works now”, “thanks, that solved it”)
- Platform-native indicators (accepted answers, positive reactions)
- Temporal patterns (question followed by detailed response followed by acknowledgment)

When a solved arc is detected, the system extracts a structured case containing:

- **Problem**: The original question or issue description
- **Solution**: The accepted or confirmed resolution
- **Context**: Supporting details (error messages, configurations, screenshots)
- **Metadata**: Timestamp, participants, confidence score

Extracted cases are embedded and indexed for semantic retrieval. The original messages may optionally be removed from the active buffer to reduce noise in subsequent processing.

2.2 Response Generation Pipeline

Algorithm 2 describes the response generation process. When a new message arrives:

Gate Classification. A classifier first determines whether the message requires a response. Messages are classified as SUPPORT (questions requiring answers), NOISE (greetings, acknowledgments), or AMBIGUOUS (unclear intent). Only SUPPORT messages proceed to retrieval.

Multi-Source Retrieval. For answerable questions, the system retrieves from three sources:

1. **Mined cases**: Semantic search over previously extracted problem-solution pairs
2. **Documentation**: Search over official technical documentation

Algorithm 2 Multi-Source Response Generation

Require: Query q, case index $\mathcal{K}$, docs index $\mathcal{D}$, history H

$g \leftarrow \textsc{GateClassifier}(q)$
if $g =$ NOISE **then**
  return $\varnothing$ ▷ Skip non-questions
end if
$R_k \leftarrow \textsc{RetrieveCases}(q, \mathcal{K}, k)$
$R_d \leftarrow \textsc{RetrieveDocs}(q, \mathcal{D}, k)$
$R_h \leftarrow \textsc{GetRecentHistory}(H)$
$R \leftarrow R_k \cup R_d \cup R_h$
if $|R| = 0$ **or** $\textsc{LowConfidence}(q, R)$ **then**
  return ABSTAIN ▷ Escalate to human
end if
return $\textsc{GenerateResponse}(q, R)$

Statistic	Value
Total messages	1,745
Unique participants	312
Extracted cases	847
Time span	6 months

Table 1: Dataset statistics for the proprietary evaluation corpus.

3. **Conversation history**: Recent messages providing conversational context

Retrieved content is ranked by relevance and deduplicated before generation.

Abstention Policy. If no relevant content is found or retrieval confidence is low, the system abstains from generating a response and flags the question for human review. This policy prevents hallucination on questions outside the system's knowledge boundary.

Response Generation. The final response is generated by a language model conditioned on the query and retrieved context. Generated responses include citations to source cases and documentation.

3 Experimental Setup

3.1 Dataset

We evaluate SupportBot on a proprietary dataset of technical support conversations from a private community focused on hardware and embedded systems. The dataset contains 1,745 messages spanning six months of community activity.

Table 1 summarizes the dataset characteristics. Messages include text, images, and file attachments in a mix of languages (primarily Ukrainian and English).

3.2 Evaluation Protocol

We use an LLM-as-judge evaluation protocol following recent work on automated assessment (Zheng et al., 2023). A separate language model evaluates each system response on a scale of 0–10 based on:

- Accuracy: Is the information correct?
- Relevance: Does it address the user's question?
- Completeness: Are important details included?
- Citation quality: Are sources properly attributed?

We report two metrics:

- **Average Score**: Mean judge rating across all evaluated responses
- **Accuracy@7**: Percentage of responses receiving a score of 7 or higher, indicating acceptable quality for deployment

3.3 System Configurations

To isolate the contribution of each retrieval source, we evaluate four system configurations:

1. **Docs-only**: Retrieval from documentation only
2. **Chat-only**: Retrieval from conversation history only
3. **Docs+Chat**: Combined documentation and conversation retrieval
4. **Full System**: Documentation, mined cases, and conversation history

All configurations use the same language model (Gemini 2.0 Flash) and identical prompting strategies, differing only in retrieval source composition.

4 Results

Table 2 presents our main results. Several patterns emerge:

Configuration	N	Avg	Acc@7
Docs-only	1000	3.55	35.0%
Chat-only	163	9.55	96.9%
Docs+Chat	50	7.66	78.0%
Full System	198	7.23	75.8%

Table 2: Evaluation results across system configurations. N is the number of evaluated queries. Avg is the mean judge score (0–10). Acc@7 is the percentage of responses scoring $\geq$7.

Configuration	Avg	vs Docs
Docs-only (baseline)	3.55	—
Chat-only	9.55	+6.00
Docs + Chat	7.66	+4.11
Full (Docs + Chat + Cases)	7.23	+3.68

Table 3: Ablation results showing improvement over documentation-only baseline.

Documentation alone is insufficient. The docs-only configuration achieves only 35.0% accuracy, with an average score of 3.55/10. This result reflects a fundamental limitation: technical documentation cannot anticipate all user issues or capture community-specific solutions.

Conversational knowledge is highly effective. The chat-only configuration achieves 96.9% accuracy with an average score of 9.55/10. This striking result demonstrates that real support conversations contain precisely the knowledge needed to answer similar questions—they represent actual problems users encountered and solutions that actually worked.

Combining sources enables broader coverage. The full system achieves 75.8% accuracy with an average score of 7.23/10. While accuracy is lower than chat-only in isolation, this configuration answers a broader range of questions, including those about historical issues no longer in the active conversation buffer and novel problems requiring documentation-based reasoning.

4.1 Ablation Analysis

Table 3 presents ablation results relative to the documentation-only baseline. Chat-only retrieval provides the largest improvement (+6.00), demonstrating that conversational knowledge contains precisely the information needed for technical support. The Docs+Chat combination achieves strong results (+4.11), while the full system including mined cases shows slightly lower average scores but substantially broader coverage—it can answer questions about historical issues no longer in the active conversation window.

4.2 Error Analysis

We manually analyzed 50 low-scoring responses (judge score < 5) to identify failure modes:

- **Out-of-domain questions** (42%): Questions about topics not covered in any retrieval source
- **Retrieval failures** (28%): Relevant information exists but was not retrieved
- **Generation errors** (18%): Correct context retrieved but response poorly synthesized
- **Ambiguous queries** (12%): Question intent unclear, leading to misaligned responses

Out-of-domain questions represent the largest failure category, suggesting that the system's abstention mechanism should be more aggressive for questions outside its knowledge boundary.

5 Discussion

5.1 Case Mining Quality

The effectiveness of mined cases depends on extraction quality. Our current system uses an LLM-based extractor that achieves approximately 85% precision on manual inspection of 100 sampled cases. Common extraction errors include:

- Incomplete problem descriptions when context spans multiple messages
- Premature closure detection on tentative "this might work" responses
- Missing nuance in solutions that depend on specific configurations

Future work should explore structured extraction models trained specifically for support conversation parsing.

5.2 Abstention vs. Coverage Tradeoffs

The full system achieves lower accuracy than chat-only because it attempts to answer a broader range of questions. This reflects a fundamental tradeoff: aggressive abstention maximizes precision but reduces system utility, while permissive answering increases coverage but risks incorrect responses.

Our current threshold (abstain when retrieval confidence < 0.3) was tuned for reasonable coverage while maintaining $>75\%$ accuracy. Production deployments may adjust this threshold based on the relative costs of false answers vs. missed questions.

5.3 Multimodal Support

Support conversations frequently include screenshots, log files, and configuration snippets. The current system preserves attachment metadata and can reference them in responses, but does not perform visual understanding of image content. Integrating vision-language models for screenshot analysis represents a promising extension.

6 Related Work

Retrieval-Augmented Generation. RAG systems (Lewis et al., 2020; Guu et al., 2020; Borgeaud et al., 2022) augment language models with retrieved documents to improve factual grounding. Our work extends RAG to conversational knowledge sources that require extraction before retrieval.

Conversational AI for Support. Prior work on support automation includes intent classification (Qu et al., 2019), response selection (Wu et al., 2017), and dialogue state tracking (Budzianowski et al., 2018). SupportBot differs by focusing on knowledge extraction from historical conversations rather than scripted dialogue flows.

Knowledge Base Construction. Automatic knowledge base construction from text has been studied extensively (Carlson et al., 2010; Mitchell et al., 2018). Our case mining approach can be viewed as domain-specific knowledge extraction tailored for support conversations.

LLM-as-Judge. Using language models for evaluation has gained traction due to scalability (Zheng et al., 2023; Dubois et al., 2024). We adopt this approach while acknowledging its limitations and the need for periodic human validation.

7 Conclusion

We presented SupportBot, a support automation system built on a key insight: community conversations are not merely communication—they are a continuously growing knowledge base of solved problems. By mining structured cases from conversational arcs and retrieving from them alongside documentation, SupportBot achieves substantially better performance than documentation-only systems.

Our experiments on 1,745 real support messages reveal a striking pattern: documentation-only retrieval achieves just 35% accuracy, while conversational knowledge retrieval achieves 97%. The full system combining all sources achieves 76% accuracy while covering a broader range of questions. These results suggest that industrial support automation should prioritize conversational knowledge extraction over documentation curation.

Future directions include improving extraction precision through specialized parsing models, integrating vision-language understanding for screenshot analysis, and studying how system performance evolves as the case index grows over deployment timescales.

Limitations

- **Single domain**: We evaluate on one technical community; generalization to other domains requires further study.
- **Language coverage**: The evaluation corpus contains primarily Ukrainian and English; multilingual performance may vary.
- **LLM-as-judge bias**: Automated evaluation may not fully capture human preferences; we mitigate this with fixed prompts and manual validation of sampled cases.
- **Temporal dynamics**: We do not study how system performance changes as the case index grows over time.

Ethics Statement

Our evaluation uses data from a private community with appropriate access permissions. All user identifiers are anonymized before analysis. The system includes an abstention mechanism to prevent answering questions beyond its knowledge, though automation bias remains a concern in production deployments. We do not release the proprietary evaluation data due to privacy considerations.

Data Availability

The proprietary evaluation dataset cannot be released due to the sensitive nature of the technical discussions and privacy considerations for community members. We provide detailed statistics and methodology to enable reproducibility with similar data sources.

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 1306–1313.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2024. Alpacafarm: A simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems*, volume 36.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3929–3938.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, and 1 others. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Chen Qu, Liu Yang, W. Bruce Croft, Yongfeng Zhang, Johanne R. Trippas, and Minghui Qiu. 2019. User intent prediction in information-seeking conversations. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*, pages 25–33.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 496–505.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36.