Clutch or Cry at SemEval-2026 Task 12: Offline Retrieval-Augmented Generation with Frozen DeBERTa for Abductive Event Reasoning

Arshad Khatib[†], **Aayush Prasad**[†], **Rudra Trivedi**[†], **Naveen Kumar**[‡], **Shrikant Malviya**[‡]
[†]Department of Artificial Intelligence, SVNIT Surat
[‡]Department of Computer Science and Engineering, SVNIT Surat
{u24ai112, u24ai091, u24ai068}@aid.svnit.ac.in
{naveenkumar,shrikant}@coed.svnit.ac.in

Abstract

This paper presents our system for SemEval-2026 Task 12: Abductive Event Reasoning. We initially approached this task using direct fine-tuning of standard language models; however, the limited volume of training data caused severe overfitting, yielding up to 95% accuracy on training splits but failing to generalize on unseen test data. Attempts to mitigate this by deploying smaller models resulted in token-limit truncations and random guessing. Given the strict all-or-nothing evaluation metric, these guesses were heavily penalized. To overcome these compounding issues, we pivoted to a two-stage offline Retrieval-Augmented Generation (RAG) pipeline. By decoupling semantic evidence retrieval from classification and fine-tuning a partially frozen `deberta-v3-large` model, we constrained the model's reasoning space to explicitly retrieved clues. This pipeline successfully curbed overfitting and random guessing, achieving a robust Exact Match accuracy of 0.72.

1 Introduction

SemEval-2026 Task 12 focuses on Abductive Event Reasoning, a complex challenge requiring systems to deduce the implicit causes or outcomes of target events from a provided text corpus. The task is formulated as a multi-label multiple-choice problem where any combination of options may be correct. Crucially, the evaluation platform (Codabench) employs a strict Exact Match metric: selecting even one incorrect option, or missing one correct option, results in a score of zero for the entire instance.

Our system's architecture evolved significantly throughout the competition. Initially, we treated the task as a standard sequence classification problem, feeding full documents and questions directly into large language models. This direct approach failed due to the limited size of the training dataset. High-capacity models rapidly memorized the training data—achieving near 95% accuracy on training and development splits—but suffered catastrophic performance drops on the test dataset due to severe overfitting.

To counteract this, we experimented with smaller, less parameter-heavy models, hypothesizing they would be less prone to memorization. Unfortunately, these smaller models possessed restricted context windows. When forced to truncate long input documents to satisfy token limits, the models lost critical causal information. Bereft of context, the models resorted to random guessing. Under the unforgiving Codabench evaluation metric, this guessing behavior decimated our overall score.

These systemic failures motivated a major architectural pivot. We realized that rather than forcing a model to read an entire document and blindly guess the causal link, we could utilize a system that explicitly queries the corpus for the answer first. This led to our final proposed system: an offline Retrieval-Augmented Generation (RAG) framework (Lewis et al., 2020). By employing a dense retriever to isolate the most relevant sentence before classification, and pairing it with a heavily regularized, partially frozen language encoder (He et al., 2021), we successfully eliminated the need for long-context guessing and prevented small-dataset overfitting.

Our source code is available at github.com/aayush-decoder/semeval-2026-abductive-event-reasoning

2 Background and Related Work

Abductive reasoning in natural language processing requires inferring the most plausible explanation for an observed event. Because reasoning in this domain often relies on implicit causes, our approach is informed by recent advancements

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 2078–2081

in hypothesis generation and retrieval. For instance, Qin et al. (2023) demonstrated that generating causal hypotheses improves downstream event prediction. Similarly, our RAG-based system treats the explicit retrieval of causal evidence from the provided corpus as a strict prerequisite for the classification phase, preventing the model from relying on hallucinated priors.

3 System Overview

Our system is designed to maximize reasoning capacity and avoid the token truncation issues encountered in earlier iterations. It consists of two primary phases, illustrated in Figure 1.

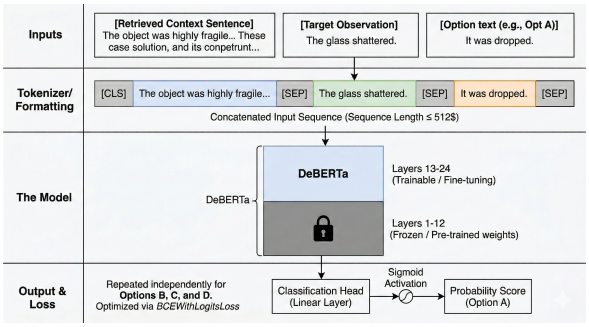

Figure 1: Architecture of the proposed offline RAG pipeline. The retriever extracts context, which is concatenated with the observation and options for multi-label classification.

3.1 Phase 1: Semantic Retrieval (Offline)

We utilize `all-MiniLM-L6-v2` (Reimers and Gurevych, 2019) as a dense retriever to encode all available paragraphs in the document corpus into a vector space. For each instance, we encode the target observation and compute cosine similarity across the document embeddings to retrieve the single most relevant contextual sentence ($top\_k = 1$).

Crucially, we empirically determined that restricting retrieval to $top\_k = 1$ yielded optimal downstream performance.

We empirically evaluated retrieval with varying numbers of retrieved sentences ($k \in \{1, 3, 5\}$). Retrieving more than one sentence consistently degraded downstream performance due to the introduction of contradictory or irrelevant causal information. Consequently, we restrict retrieval to $top\_k = 1$ for all experiments.

Attempting to retrieve a wider window of documents introduced severe context noise and contradictory statements, which actively degraded classification accuracy. Performing this extraction offline ensures that the downstream classifier is only fed highly relevant, token-efficient evidence.

3.2 Phase 2: Multi-Label Classification

Our core reasoning engine is `microsoft/deberta-v3-large`. To adapt it to the multi-label requirement without triggering the strict Codabench penalty, we replace the standard Cross-Entropy loss with `BCEWithLogitsLoss`, allowing the model to treat each option as an independent binary classification task rather than a forced multiple-choice guess.

Anti-Overfitting Measures: To mitigate overfitting, we freeze the bottom 12 layers of the 24-layer DeBERTa-v3-Large encoder. Preliminary experiments with varying freezing depths indicated that freezing 12 layers provided the best balance between training stability and task adaptability. This drastically reduces the trainable parameter count, forcing the model to rely on its pre-trained semantic understanding rather than memorizing the specific training dataset.

3.3 Pipeline Example

To illustrate the data flow, consider a sample target event: *"The glass shattered."* During Phase 1, the retriever scans the corpus and extracts the highest-scoring context: *"The object was highly fragile and dropped from a height."*

During Phase 2, the system constructs a flat sequence using special tokens: `[CLS] The object was highly fragile... [SEP] The glass shattered. [SEP] Option A: It was dropped. [SEP]`

This format is generated iteratively for all four options, allowing the model to assess the validity of each statement independently against the retrieved evidence.

4 Experimental Setup

Our system was developed using PyTorch and the Hugging Face `transformers` library (Wolf et al., 2020). Experiments were executed on an NVIDIA H100 GPU within an HPC SLURM environment.

To fit the 435M-parameter DeBERTa model into memory without sharding, we employed gradient checkpointing and mixed-precision training (`bfloat16`). We trained with a strict batch size

of 1 and accumulated gradients over 16 steps (effective batch size of 16).

Token Length Optimization: The maximum sequence length for the tokenizer was strictly bounded to 512 tokens to conserve GPU memory. While traditional full-document inputs would heavily exceed this limit and trigger automatic truncation, our offline RAG preprocessing guarantees that input sequences rarely reach this boundary. By distilling entire documents down to a single relevant context sentence prior to classification, we ensure that critical causal evidence is preserved and not lost to the truncation process.

Optimization was performed using AdamW with a learning rate of $5e^{-6}$ and a weight decay of 0.01. Training was capped at 8 epochs with early stopping based on Exact Match accuracy on the development split.

5 Results

5.1 Main Quantitative Findings

Our final system, utilizing the `deberta-v3-large` architecture with Offline RAG and layer freezing, achieved an **Exact Match Accuracy of 0.72** on the official test set.

5.2 System Evolution and Ablation Study

To validate our architectural decisions, we documented the performance failures that led to our final system design. Table 1 outlines this progression.

Model Config	Context	Train Acc.	Test Acc.
Large (No Freeze)	Full Doc	∼95.0%	Poor
Small Model	Truncated	49.5%	<35%
Large (Frozen)	**RAG ($k = 1$)**	**Balanced**	**72.0%**

Table 1: Evolution of the system, demonstrating the initial overfitting and the subsequent stabilization via RAG and regularization.

Overfitting in Direct Approaches: Our initial baseline utilized a large language model fed with full documents. While this achieved near 95% accuracy during training, the lack of regularization on the small dataset led to severe overfitting, rendering the model useless on the unseen test set.

Token Limits and Random Guessing: We subsequently evaluated `microsoft/deberta-v3-small` using naive context truncation (limiting documents to the first 500 characters) to fit the smaller context window. This model peaked at 49.57% validation accuracy but quickly degraded to 34.19%. Error analysis revealed that truncating the text frequently discarded the actual causal clues. Without evidence, the model resorted to random guessing. Because the Codabench platform assigns a score of zero for any partially incorrect answer, these guesses completely derailed the system's performance.

Stabilization via RAG: The final system resolved these compounding errors. By using RAG, we bypassed the token limits, feeding the large model only the most critical sentence. By freezing the bottom 12 layers, we prevented the 95% overfitting scenario, allowing the model to generalize and secure the final 0.72 test accuracy.

6 Conclusion and Future Works

We presented a two-stage offline RAG approach for SemEval-2026 Task 12: Abductive Event Reasoning. Our system's evolution highlights the dangers of direct fine-tuning on limited datasets, where large models overfit and small models resort to penalized random guessing due to token constraints. By decoupling retrieval from training, restricting context to $top\_k = 1$ to reduce noise, and applying aggressive layer freezing to a DeBERTa-v3-Large classifier, we successfully engineered a robust reasoning pipeline. This approach effectively mitigates overfitting and hallucination, yielding a highly competitive final accuracy of 0.72 under strict Exact Match evaluation.

A limitation of this method is its reliance on single-sentence retrieval, which may fail when causal evidence is distributed across multiple sentences. Future work will explore contradiction-aware multi-sentence retrieval strategies while preserving robustness under Exact Match evaluation.

References

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

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

Yujia Qin and 1 others. 2023. Language models can improve event prediction by few-shot abductive reasoning. *arXiv preprint arXiv:2305.16646*.

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*. Association for Computational Linguistics.

Thomas Wolf and 1 others. 2020. Transformers: State-of-the-art natural language processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.