

# FASTTRACK: Fast and Accurate Fact Tracing for LLMs

**Si Chen**  
Virginia Tech  
chensi@vt.edu

**Feiyang Kang**  
Virginia Tech  
fyk@vt.edu

**Ning Yu**  
Netflix Eyeline Studios  
ningyu.hust@gmail.com

**Ruoxi Jia**  
Virginia Tech  
ruoxijia@vt.edu

## Abstract

Fact tracing seeks to identify specific training examples that serve as the knowledge source for a given query. Existing approaches to fact tracing rely on assessing the similarity between each training sample and the query along a certain dimension, such as lexical similarity, gradient, or embedding space. However, these methods fall short of effectively distinguishing between samples that are merely relevant and those that actually provide supportive evidence for the information sought by the query. This limitation often results in suboptimal effectiveness. Moreover, these approaches necessitate the examination of the similarity of individual training points for each query, imposing significant computational demands and creating a substantial barrier for practical applications. This paper introduces FASTTRACK, a novel approach that harnesses the capabilities of Large Language Models (LLMs) to validate supportive evidence for queries and at the same time clusters the training database towards a reduced extent for LLMs to trace facts. Our experiments show that FASTTRACK substantially outperforms existing methods in both accuracy and efficiency, achieving more than 100% improvement in F1 score over the state-of-the-art methods while being  $\times 33$  faster than TracIn.

## 1 Introduction

Recent years have witnessed *large language models (LLMs)* demonstrating remarkable abilities in absorbing vast knowledge from extensive text corpora, yielding impressive advancements in NLP tasks such as question answering (QA). However, these models often produce seemingly coherent yet unfounded outputs, known as ‘hallucinations’ (Agrawal et al., 2023), posing risks in high-stake scenarios such as healthcare and finance, where reliability is of paramount importance (Master of Code, 2023). This critical challenge has motivated research on *fact tracing* (Akyürek et al., 2022), which aims to identify the training data that

serves as the knowledge source for LLMs’ generation. Striving to provide a pathway to understanding and mitigating the issue of hallucination, Akyürek et al. (2022) proposed a benchmark for fact tracing, formulating it as a challenging task that involves searching for training data that has fact-support correspondence (i.e., supportiveness) with given queries. Current methods, however, tend to miss the mark and overly rely on similarity measures between individual training samples and the target query, such as gradient similarity (Pruthi et al., 2020; Koh and Liang, 2017), embedding similarity (Rajani et al., 2020), or lexical similarity (Robertson et al., 1995; Lv and Zhai, 2011). As a natural result, these approaches may fail to differentiate between samples that merely look similar and those that actually contain the supporting information sought by the query—even in considerably simple cases. This prominent issue limits their effectiveness in identifying supportive training examples, preventing them from being effective in broader use cases (Akyürek et al., 2022). Besides, some of these methods, such as Pruthi et al. (2020); Koh and Liang (2017), carry a significant computational overhead in analyzing a given query. Providing intellectual inspiration for research exploration, nonetheless, its computational demand can be unaffordable for most practical scenarios. Despite soaring interest in this emerging problem, current research still falls short of the critical need by a large margin.

We summarize the **desiderata** for fact-tracing methods as the following:

- ◇ **D-i. Effective and Accurate.** For a target query, fact-tracing methods need to identify *all* supporting facts in the training corpus and achieve both high precision and recall simultaneously.

- ◊ **D-ii. Computationally Tractable.** Fact-tracing methods need to be scalable with both the number of queries and the number of training samples to be examined.
- ◊ **D-iii. Practically Robust.** Fact-tracing prioritizes general-purposed, principled methods that are plausible for deployment and transferable between use cases.

Current methods all miss one or more of these principles. Specifically, gradient-similarity-based methods (Pruthi et al., 2020; Koh and Liang, 2017) are notoriously computationally demanding (D-i, D-ii). Also, gradients are considerably susceptible to noises, rendering their performance rather unstable even with extensive hyper-parameter tuning (Akyürek et al., 2022; Park et al., 2023) (D-i, D-iii). Lexical-similarity-based methods (Robertson et al., 1995; Lv and Zhai, 2011) are typically faster, but relying on queries and samples with supporting facts being similarly phrased. This assumption is not necessarily true in realistic use cases (D-iii). Table 4 shows that the performance for such methods may drop a large margin under slight rephrasing of the text (D-i). Therefore, these methods are neither practical nor reliable (as illustrated in Sec. 5.2).

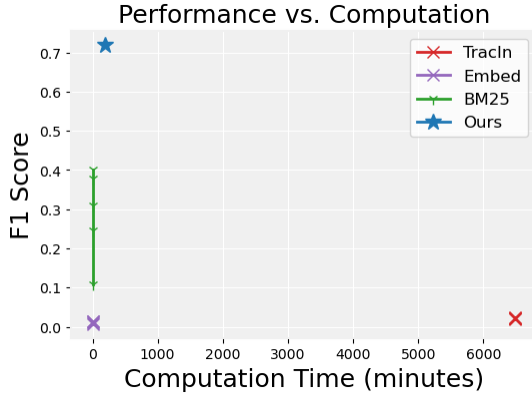


Figure 1: FASTTRACK achieves the best tradeoffs between fact tracing efficacy and efficiency. The x-axis is the computational time of evaluating 100 queries using a 10k corpus, and the y-axis is the tracing performance when using top-k thresholds (if applicable). TDA methods yield consistently low performance across top-k thresholds, making them look like dots in the plot.

Determining whether a training example supports a factual statement in a query demands reasoning abilities beyond sample similarities where support for a factual assertion often arises through the

inference of connections among related pieces of information. The dilemma with these approaches is that *no single representation works in all cases* and the similarity in these pre-defined spaces may easily fail to capture the nuance of *supportiveness* effectively. Inspired by the recent advancement in LLM’s abilities in natural language understanding (NLU), a natural idea is to directly evaluate the supportiveness between each training sample and the target query using an LLM. Unprecedented in-context learning (ICL) capabilities make these models notably versatile and easily adaptable to novel cases with minimal customization, effectively bridging the realistic gap between fact-tracing methods and real-world scenarios. Admittedly, our preliminary investigation shows that this idea indeed enhances the efficacy in the identification of supportive training samples to an impressive extent. Nevertheless, this idea faces immediate challenges when applied to a practical-sized training corpora: traversal evaluation for all training sample-query pairs requires a massive number of queries to the LLM, unaffordable in both computation time and costs, hindering it from being practically useful.

To address this dilemma, we propose FASTTRACK, which is a two-stage scheme decomposed into offline and online components. In the first stage, we build semantic indexes for the training corpus through hierarchical clustering. Such process is completely offline and only need to be run once. During online stage, these pre-built semantic indexes facilitate the retrieval of relevant clusters for any given query, significantly reducing the search range. FASTTRACK then runs a fine-grained examination by employing a LLM to evaluate the supportiveness of training data in the retrieved clusters. While prior work (Akyürek et al., 2022) requires careful selection of small candidate set of size around 500 for practical evaluation, FASTTRACK enables a balance between computational feasibility and fine-grained analysis. This enables it to accommodate large corpus of size 10k or even 100k, while ensuring both satisfactory efficiency and efficacy (high precision and recall).

**Our contributions are summarized as follows:**

- We propose a novel two-stage pipeline FASTTRACK and show it is easily adaptable without needing to train a model. (*meets D-iii*)
- We evaluate FASTTRACK’s performance on various datasets with baseline methods. FAST-

TRACK achieves notable F1 scores of 0.72 on FTRACE-TREx and 0.91 on VITAMINC, more than **doubling** the performance of the best existing methods. (*meets D-i*)

- We show FASTTRACK to offer a substantial edge in efficiency, being **33 $\times$  faster** than the TDA method TRACIn for a corpus of 10k samples, and readily applicable to larger datasets with more than **100k samples**. (*meets D-ii*)

## 2 Related Work

**Training Data Attribution (TDA).** TDA aims to trace model predictions back to the training examples that responsible for these predictions, which shares a similar goal with fact tracing. Prior work (Akyürek et al., 2022) proposes to use two main types of TDA methods as baselines: gradient-based and embedding-based attributions. Gradient-based methods, such as TRACIn (Pruthi et al., 2020), estimate the attribution score of training data on predictions by calculating the cosine similarity between the gradients of the training data and the query. Embedding-based methods employs the model’s internal representations to determine the relevance of training examples to a given test prediction (Rajani et al., 2019). The attribution score is defined as a cosine product of hidden representations.

To retrieve supporting training data for a given query  $z_{\text{query}}$ , one need to score every training data and rank them by their influence score. As it could be computationally infeasible for gradient-based TDA scoring all training data in large datasets, Akyürek et al. (2022) only evaluates on carefully selected small subsets (i.e., around 500) for each query. This limitation motivates us to design a framework that is both more computationally efficient and more effective.

**Information Retrieval (IR).** IR focuses on retrieving relevant documents in a large collection given specific queries (Izacard et al., 2021). Though not originally designed for fact tracing task, prior work (Akyürek et al., 2022) found it effective and outperforms principled TDA methods by a large margin. IR splits into two categories: term-frequency-based methods like BM25(Thakur et al., 2021; Zhou et al., 2022), which score each training data base on the token overlap with the given query, inversely weighted with the frequency of such tokens, and neural network-based methods (Izacard et al., 2021; Ni et al., 2021), which, despite their

advanced capabilities, often require extensive manual annotations, making them less suited for fact tracing due to the absence of necessary annotations. Recent attempts to adapt neural methods through zero-shot learning have not matched BM25’s performance (Thakur et al., 2021; Zhou et al., 2022). Therefore, following prior work, we select BM25 as the baseline for fact tracing due to its superior retrieval quality without the need for annotated data.

All of the methods above focus on *relevance* while neglecting the *supportiveness* of the connection between training data and the query. In this paper, we introduce FASTTRACK, the first supportiveness-aware approach for fact tracing, offering substantial benefits in real scenarios where training data may contain conflicting information.

## 3 Methodology

Fact tracing aims to identify knowledge source of a particular query. While similar to TDA, it focuses more on the fact-support correspondance between training data and query. This distinction is crucial: existing methods often retrieve relevant examples but fail to provide factual support, misaligning with the objective. The strong capability of LLMs such as ChatGPT makes it a perfect solution to provide justification based on ‘supportiveness’. However, directly doing pair-level comparison could be very time-consuming: Given a corpus of size  $N$  and  $m$  queries, the computation complexity is  $\mathcal{O}(mN)$ .

In this section, we introduce an original two-stage framework FASTTRACK, as illustrated in Figure 2. In the first stage, FASTTRACK leverages a recursive clustering scheme to mine the semantic structure in the training corpus, which enables a coarse matching for a given query. This significantly refines the search range, making it feasible to perform a fine-grained examination of each candidate training examples in the second stage.

### 3.1 Semantic Clustering

The goal of the first stage is to create semantically meaningful indexes in an offline setting. This one-time process allows for the efficient utilization of these indexes in subsequent online stages, eliminating the need for re-computation. In this paper, we propose to employ a simple hierarchical clustering process over training data embeddings to recover underlying tree structures of the data. This process reorganize the entire training corpus into a more structure format, laying the groundwork for more

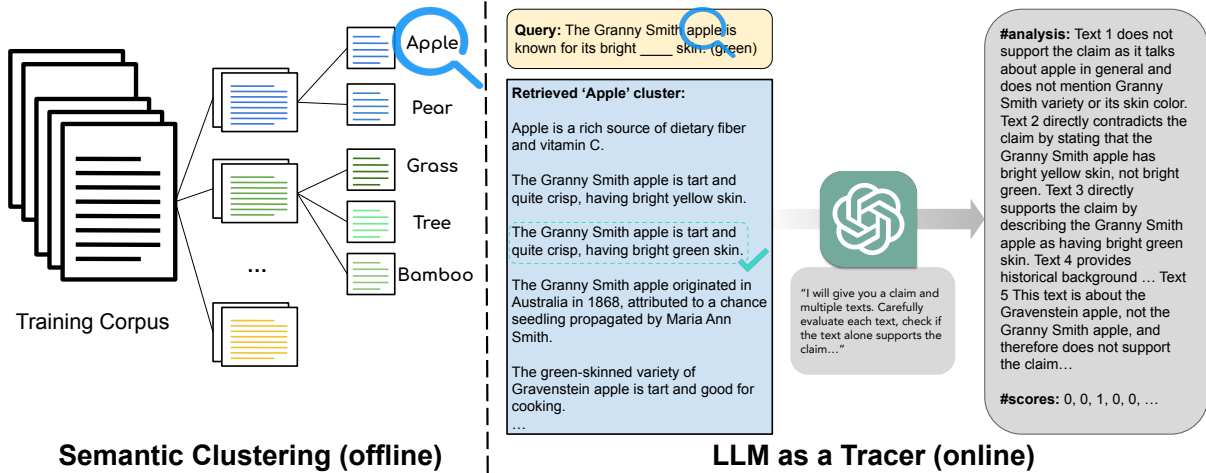


Figure 2: Illustration of FASTTRACK workflow. Stage 1, which is completely offline, reorganizes the training corpus into a semantic tree for easier navigation; Stage 2 retrieves relevant clusters using fuzzy keyword matching, then employs LLMs to assess candidate samples, retrieving those with a score of 1.

effective data navigation and retrieval. We first apply k-means clustering on the sample embeddings to mine its semantic structure. The clustering is conducted recursively where larger clusters will be further clustered until the size of all the clusters is within a certain threshold.

The key of our method lies in transcending the limitations of conventional clustering algorithms, which typically do not assign semantically meaningful labels to each cluster. By harnessing the power of Large Language Models (LLMs), FASTTRACK assigns a carefully selected set of keywords to each cluster, serving as its semantic label. This strategic integration not only renders the clustering outcomes interpretable but also significantly simplifies the process of navigating through the corpus in response to specific queries. We note that such semantic clustering only need to be applied once offline, effectively allowing us to leverage the massive amount of compute in pre-training for free.

### 3.2 LLM as a Sample-Level Tracer

With the structured and semantically meaningful clusters, we can now online process each query for fact tracing efficiently. The first step is to retrieve relevant clusters for a given query. A simple example for such cluster-level retrieval is to apply fuzzy match<sup>1</sup> to identify those clusters that shared similar keywords as the query. Furthermore, the efficacy of clustering can be enhanced through ensemble of different clustering outcomes, as detailed in Table 2.

<sup>1</sup><https://github.com/seatgeek/thefuzz>

Now, with the retrieved clusters, the second step is to identify the groundtruth supporting data from this narrowed pool. We frame this stage as a binary verification problem: given a specific query, we classify each candidate training example into two categories based on its ‘supportiveness’. An example is considered ‘grounding’ if it supports the query. A direct way to perform such classification is to instruct the LLM to evaluate a single training example against a query for supportiveness, assigning a score of 1 for supportiveness and 0 otherwise. Although effective, this one-at-a-time scoring method can still be computationally and financially costly. To further enhance efficiency and speed up the process, we devised the prompting strategy to evaluate a batch of training data in a single inference run. This batch processing approach significantly cuts down the time required for evaluations, reducing the number of necessary inferences by a factor of  $b$ , where  $b$  is the number of candidate examples in a batch. The example prompt used in our experiments can be found in Appendix F. Following the LLM’s evaluation, examples that are assigned a score of 1, indicating supportiveness, are systematically retrieved. The detailed workflow of FASTTRACK is presented in Algorithm 1.

## 4 Experimental Setup

### 4.1 Datasets

**FTRACE-TREx.** The FTRACE-TREx dataset is proposed by (Akyürek et al., 2022), with 27k queries created using LAMA (Petroni et al., 2019) and 1M masked training examples extracted from



TREx (Elsahar et al., 2018) as the attribution set. Each training example is a cloze-style sentence with either the subject or object masked. The groundtruth training example for each query is defined as the examples that express the same fact, regardless of the masking position. To address the computational overhead, Akyürek et al. (2022) proposes to construct a small, separate candidate set for each query (around 500). We follow a similar setup, but use a **larger, fixed** candidate pool to better reflect real-world scenarios: we randomly sample 100 queries from the entire query set for evaluation, and build the candidate pool by including all the corresponding groundtruth, supplementing with random samples to form a corpus of **10k**.

**VITAMINC.** We incorporate the VITAMINC dataset (Schuster et al., 2021) as a means to evaluate fact tracing methods’ ability to mirror real scenarios where training corpus of LMs containing contradictions or misinformation. The VITAMINC dataset is built based on factual revisions to Wikipedia: each single factual revision yields a contrastive pair of contexts, where one context refutes the given claim and the other supports it. The original VITAMINC dataset presented each entry in the format of *claim*, *evidence*, and *label*, where the label indicates if the evidence ‘SUPPORTS’, ‘REFUTES’, or provide ‘NOT ENOUGH INFO’ to the evidence. To use it for fact tracing purposes, we build the attribution set by collecting 10k unique pieces of evidence (acting as training data). Then the query set is built by collecting corresponding claims that can be supported by the evidence.<sup>2</sup>

## 4.2 Baselines

Following Akyürek et al. (2022), we compare our method FASTTRACK with TDA methods (i.e., TRACIN, EMBED) and the most representative IR method (i.e., BM25).

**TRACIN.** TRACIN (Pruthi et al., 2020) is a recent gradient-based TDA method that has demonstrated strong empirical results and tractability. Following the setup of Akyürek et al. (2022), we use an optimized version of TRACIN by rescaling gradients with Adafactors accumulators, applying unit-normalization to the gradients, and selecting the

best-performing layer. Data in FTRACE-TREx are cloze-style examples, hence we finetune an MT5 model (Xue et al., 2021) following Akyürek et al. (2022) to predict the masked tokens. We note that gradient similarity is only meaningful when query and training data have the same question-answer construction, and it is difficult to construct the VITAMINC dataset in this way. Hence, we omit the evaluation of TRACIN on VITAMINC dataset.

**EMBED.** Embedding-based similarity is another popular branch for fact tracing tasks. Here we refer to Equation 2 as baseline EMBED. For FTRACE-TREx dataset, we use the same finetuned MT5 model as for TRACIN, selecting the best-performing layer as the final result. For the VITAMINC dataset, we finetune a BERT model (Kenton and Toutanova, 2019) on our constructed attribution set.

**BM25.** We use a publicly available implementation of BM25 (Lv and Zhai, 2011) as our baselines<sup>3</sup>. We tokenize queries and training examples by space, removing any masked tokens. We proceed with the default settings for all hyperparameters, ensuring a standardized approach for our baseline comparisons.

## 4.3 Tracing Performance Evaluation

TDA methods and BM25 score a given test query against every training example and then sort all examples based on their scores. This results in a top-k precision and recall performance measurement, where the k is the threshold of taking the top k ranked examples as the retrieved supporting training data (Akyürek et al., 2022). In contrast, our method directly retrieves the supporting training data without ranking. To enable a unified comparison, we use F1 score as the main metric. We report the best-performing F1 score and the corresponding precision and recall for each method.

# 5 Empirical Results

## 5.1 Overall Performance

We first evaluate the overall performance of different methods on FTRACE-TREx and VITAMINC datasets in Table 1. Hyperparameters for all methods are presented in Appendix C.

Fact tracing is a challenging task. Previous work (Akyürek et al., 2022) proposes several techniques to optimize TDA methods but found that even

<sup>2</sup>Due to the labeling format of the original dataset, some claims may have more than one supporting evidence but we do not know. To address such an issue, we manually inspect 100 queries for their groundtruth data and use these queries for evaluation. We provide the data we manually inspect along with this submission.

<sup>3</sup><https://pypi.org/project/rank-bm25/>

Table 1: Comparison of fact tracing performance. We present the best F1 scores among top-k for each method; precisions and recalls are chosen at the threshold lead to optimal F1 score. Among all methods, FASTTRACK performs the best. \*The last row gives the upper bound performance achievable in the first cluster-level retrieval stage.

|              | FTRACE-TREx |             |             | VITAMINC    |             |             |
|--------------|-------------|-------------|-------------|-------------|-------------|-------------|
|              | F1          | Precision   | Recall      | F1          | Precision   | Recall      |
| TRACIN       | 0.02        | 0.19        | 0.01        | -           | -           | -           |
| EMBED        | 0.01        | 0.08        | 0.01        | 0.48        | 0.54        | 0.46        |
| BM25         | 0.40        | 0.49        | 0.52        | 0.55        | 0.59        | 0.53        |
| <b>Ours</b>  | <b>0.72</b> | <b>0.81</b> | <b>0.69</b> | <b>0.91</b> | <b>0.88</b> | <b>0.98</b> |
| <b>Ours*</b> | 0.86        | 0.92        | 0.83        | 1.00        | 1.00        | 1.00        |

BM25 with no tuning outperforms TDA, and all these methods are far from perfect. In Table 1 we show similar findings, where TRACIN and EMBED resulted in F1 score lower than 0.1 on FTRACE-TREx dataset. We also observe that TRACIN’s performance is highly dependent on the chosen model checkpoint. Specifically, the performance noted in our main results table was achieved using the final 80k-step checkpoint, with earlier checkpoints yielding even weaker outcomes (as shown in Appendix E).

**Takeaway:** FASTTRACK delivers impressive tracing performance, yielding both high precision and recall, improving the F1 score by **>80%** compared to the best-performing baseline BM25.

All baseline methods retrieve training examples based on their ‘relevance’ to the given query, which could violate the goal of fact tracing. This discrepancy becomes evident in real-world scenarios, where datasets, unlike the scientifically accurate and consistent ones often evaluated in prior research, contain conflicting information. Our evaluation on VITAMINC dataset reveals that such methods yield low precision due to their relevance-focused logic. Notably, FASTTRACK significantly outperforms all baselines, achieving an F1-score of 0.91, demonstrating its effectiveness in accurately identifying grounding training data for queries.

**Takeaway:** FASTTRACK not only excels in fact-tracing performance but also offers the optimal balance between computational speed and effectiveness. It outperforms competitors significantly, running **33 times faster** than TRACIN in evaluating 100 queries (Figure 1).

## 5.2 Failure Analysis

In this section, we qualitatively examine some failure examples of different tracing methods to shed light on the future direction of fact tracing.

**When does BM25 fail?** BM25 operates based on token overlap, and retrieves examples with high lexical similarity to the query, regardless of their factual consistency. As shown in the example below, while the first retrieved example is correct, the second contradicts the query, and the third is entirely unrelated.

**Query:** Alloy Digital’s network has a monthly reach of more than 100 million unique visitors.

**BM25 Retrieved:**

**Rank-1:** Defy Media: According to comScore, Alloy Digital’s network reaches over 221 million unique visitors each month, including more than half of the aged 12-34 internet users.

**Rank-2:** According to comScore, Alloy media platforms reach over 95 million unique visitors each month, including over half of the age 12-34 internet users.

**Rank-3:** The franchise has sold more than 26 million units worldwide with the release of 2018 ’s installment.

BM25’s performance can be poor even when there are no such data conflicts. We further conduct experiment on FTRACE-TREx dataset where we paraphrase each query using an open-sourced paraphraser<sup>4</sup>. The performance of BM25 before and after paraphrasing is shown in Table 4, where both precision and recall drop by a wide margin.

**When do TDA methods fail?** TRACIN conducts a first-order approximation and uses the dot product of the model’s gradients between each train-test sample pair to measure this contribution. However, we find its actual performance is fragile and can be affected by a number of factors.

1) TRACIN’s performance is highly dependent on having the exact same construct of question-answer pairs. LMs for QA tasks typically use an encoder-decoder architecture, such as T5/MT5. The gradient is calculated with respect to the loss

<sup>4</sup>[https://huggingface.co/humarin/chatgpt\\_paraphraser\\_on\\_T5\\_base](https://huggingface.co/humarin/chatgpt_paraphraser_on_T5_base)

of the word/token being predicted. However, gradient similarity between a train-test sample pair is only meaningful when these are the same QA questions with identical question-answer pairs. In other words, even for sample pairs where the texts are the same, if the construction of question-answer is different, the loss and gradient may be unrelated. This aligns with our evaluation results: we find that TRACIN cannot identify those groundtruth training examples with supporting facts but having different QA construction. This results in arbitrarily poor performance on some queries, as the cosine similarity between gradients - which are high-dimensional vectors - can be dominated by unrelated factors and fail to capture the actual correlation between samples.

2) TRACIN *tends to retrieve sentence with the same masked token*. Such finding has also been observed in (Akyürek et al., 2022). This likely occurs because the same masked token produces similar training gradients.

**Query:** Comptroller of Maryland is a legal term in \_\_\_\_\_. (Maryland)

**TRACIN Retrieved:**

**Rank-1:** The \_\_\_\_\_ Comptroller election of 2010, was held on November 2, 2010. (Maryland)

**Rank-2:** It is found in Alabama, Florida, Louisiana, \_\_\_\_\_, Mississippi, North Carolina and Virginia. (Maryland)

As illustrated in the example above, the top-ranked retrieved example is correct, where the training example and query share the same masked target token. However, the second retrieved example does not provide any relevant fact, only the masked token to predict is the same.

The other TDA method evaluated in this paper, EMBED, relies on hidden space similarity search. The dilemma for this approach is that *no single representation works for all tasks* (Vaze et al., 2023), which is more pronounced in these QA problems. The similarity of text pairs could be measured from different perspectives and the one that is best captured does not necessarily focus on the "supporting fact". Another major issue with this approach is that similar texts always receive similar scores, rendering the results end up in clumps. If the front-running clump is wrong, all samples in the clump are wrong, yields zero top-k accuracy. For example, for the same query "*Comptroller of Maryland is a legal term in <MASK>*", the top 3 retrieved

examples of EMBED are:

**Rank-1:** the Mayor of \_\_\_\_\_. (Moscow)

**Rank-2:** Embassy in Cyprus is located in \_\_\_\_\_. (Nicosia)

**Rank-3:** He served on the \_\_\_\_\_ of Edmonton. (town council)

These retrieved examples, to varying degrees, relate to the query by involving 1) public offices and elected officials, 2) political or geographical entities, and 3) individuals with governmental roles. In fact, The groundtruth example belongs to a similar category. Yet, embedding similarity cannot detect fact-support correspondence between samples and cannot distinguish different levels of sample similarities.

## 6 Ablation Study and Analysis

**In-depth Analysis of FASTTRACK.** The first stage of FASTTRACK- cluster level retrieval - decides the performance upper bound of our methods. If relevant clusters are not identified during this phase, it becomes impossible to recover them in the later stage. We report the upper bound performance achievable in the last row of Table 1, to reveal the limitation origins from the first stage. Specifically, this upper bound assumes perfect accuracy in the second stage, meaning if the correct cluster is identified, we achieve 100% precision and recall on this cluster. As shown in Table 1, the upper bound of FASTTRACK has a short gap to perfect. The precision is 0.92 while the recall is only 0.83. Such failure origins from the first stage could come from two sides:

1) *clustering algorithm*. The clustering algorithms group data with similar embedding together. Although in general, we observe that groundtruth training data for a specific query usually falls within 4 clusters on average, which means the clustering algorithm successfully groups relevant training data into the same cluster, there still exists the case that for some clusters the groundtruth training data is the minority. In such case, groundtruth data could be ignored when assigning the cluster semantically meaningful keywords, making this cluster hard to retrieve. In practice, this can be improved by using an ensemble - we observe that an ensemble of three yields a performance upper bound of precision 0.92, recall 0.83; while single clustering yields an upper bound of precision 0.81, recall 0.65.

2) *cluster retrieval method*. We currently em-

Table 2: Upperbound performance of FASTTRACK when using single and ensemble embeddings on FTRACE-TREx.

|                  | Single | Two-Ensemble | Three-Ensemble |
|------------------|--------|--------------|----------------|
| <b>Precision</b> | 0.81   | 0.89         | 0.92           |
| <b>Recall</b>    | 0.65   | 0.78         | 0.83           |

ploy simple fuzzy matches to capture clusters that share similar keywords as the query. However, the training data may present the query on a different surface. Future studies could leverage more advanced tools to enhance the process.

Table 1 shows that there exists a gap between performance upper bound and final performance. This gap comes from ChatGPT’s limitation, where it misclassified a few examples. We show two interesting types of misclassification here:

**Query:** President of the Executive Yuan is a legal term in \_\_\_\_\_. (Taiwan)

**False negative examples (mask removed):**

1. He has interviewed financial services regulators including Sean Chen (politician), the Premier of Taiwan, when he was the Chairman of the Financial Supervisory Commission (Republic of China) of Taiwan and negotiated the financial Memorandum of Understanding with China.
2. Hsich Tung-min was the ninth Governor of Taiwan Province (1972-1978) and the sixth and first local Taiwanese Vice President of the Republic of China (1978-1984) under President Chiang Ching-Kuo.

**GPT-4 analysis:**

The term "President of the Executive Yuan" is not mentioned in any of the texts. The texts mention various political positions in Taiwan, such as the Premier of the Republic of China and the President of Taiwan, but none of them refer to the President of the Executive Yuan. Therefore, it cannot be inferred from the texts that "President of the Executive Yuan" is a legal term in Taiwan.

In the above example, GPT-4 did not recognize that the 'Executive Yuan's leader is the 'Premier of Taiwan', indicating a gap in connecting related concepts. The second failure example appears to be a labeling error. Another example is that GPT-4 struggles with complex logical reasoning involving dates; for instance, it incorrectly equates the information from different dates, focusing merely on numerical comparisons (see Appendix E). Failure cases at this stage mainly stem from LLM's own bottleneck. These challenges represent a significant area of ongoing research and are beyond the scope of our current study. We acknowledge these limitations and suggest them as critical avenues for future investigation to enhance the capabilities and

Table 3: Performance of BM25 and FASTTRACK when dealing with different corpus size. Both of the methods encounter a slight performance drop, while FASTTRACK is still  $1.66\times$  better than BM25.

|              | VITAMINC-10k |           |        | VITAMINC-100k |           |        |
|--------------|--------------|-----------|--------|---------------|-----------|--------|
|              | F1           | Precision | Recall | F1            | Precision | Recall |
| <b>BM25</b>  | 0.55         | 0.59      | 0.53   | 0.53          | 0.56      | 0.50   |
| <b>Ours</b>  | 0.91         | 0.88      | 0.98   | 0.88          | 0.85      | 0.92   |
| <b>Ours*</b> | 1.00         | 1.00      | 1.00   | 0.95          | 0.95      | 0.95   |

applications of LLMs.

**Embeddings Schemes.** We use SentenceTransformer<sup>5</sup> as the embedding model to perform clustering in our main evaluation. To test the sensitivity of FASTTRACK on different choices of embeddings, we also test some state-of-the-art embedding models such as Cohere Embed v3<sup>6</sup> and Mistral-Embed<sup>7</sup>. As shown in Table 6, FASTTRACK consistently achieves comparable top-performance upperbounds across various embedding models, underscoring its adaptability to different embedding choices.

**Corpus Size.** Moving forward, we aim to tackle a more challenging scenario: we use the same query set of VITAMINC, but augment the attribution set with additional non-relevant examples until the total reaches 100k. This setting is designed to evaluate our method's robustness in scenarios that better resemble real-world applications. As shown in Table 7, both methods exhibit a slight decline in performance, yet FASTTRACK consistently outperforms BM25 by a significant margin. BM25's performance drop is ascribed to the inclusion of new examples that exhibit high lexical overlap with the queries, while our method, mainly stems from the clustering stage, where the clustering logic has been impacted after a more diverse sample are included. We leave a detailed analysis in Appendix E.

## 7 Conclusion

In this paper, we introduced FASTTRACK, a pioneering two-stage framework designed to address the shortcomings in current fact tracing methodologies, particularly their neglect of the 'supportiveness' of evidence. FASTTRACK substantially improves the tracing performance by more than 100% in F1 score, and offers computational effi-

<sup>5</sup><https://www.sbert.net>

<sup>6</sup><https://txt.cohere.com/introducing-embed-v3/>

<sup>7</sup><https://docs.mistral.ai/api/>



ciency, capable of handling large datasets up to 100k in size. We also provide a thorough analysis of each tracing method to shed light on future direction in fact tracing. It's important to note that the current performance bottleneck primarily stems from the limitations of GPT models. Therefore, future efforts could focus on fine-tuning a model specifically for tracing purposes.

## Limitations

While our proposed method, FASTTRACK, has shown considerable success, it's important to acknowledge that its performance is ultimately constrained by the capabilities of GPT models. Thus, future work could explore techniques to fine-tune a LLM specifically targeting tracing purpose. Another limitation of FASTTRACK is its capacity to process only a limited number of training examples in each batch. This presents an opportunity for future improvements by incorporating techniques that can handle longer contexts. By doing so, it may be possible to decrease the necessity for multiple inferences, thereby optimizing the process.

## Ethics Statement

Our research advances the accuracy and efficiency of fact tracing techniques, elucidating the connections between the training data of LLMs and their generated assertions. Our method ensures data privacy integrity, as it solely utilizes publicly accessible data samples, and is not designed for the inadvertent generation of unintended content by LLMs. While mindful of the potential for redistributing web data to inadvertently disseminate misinformation, we foresee no other ethical concerns with our methods.

Committed to the ethos of open science, our study champions reproducibility, transparency, and the facilitation of further inquiry. To this end, we will grant unrestricted access to all materials related to our research. This includes a comprehensive, meticulously documented repository encompassing all scripts, models, and codes necessary for preprocessing and evaluation, allowing for the full replication of our experiments. Beyond making these resources available, we are dedicated to their ongoing maintenance and to providing timely support for any inquiries or clarifications. This commitment underlines our dedication to fostering a collaborative, inclusive research community.

## References

- Ayush Agrawal, Lester Mackey, and Adam Tauman Kalai. 2023. Do language models know when they're hallucinating references? *arXiv preprint arXiv:2305.18248*.
- Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Towards tracing knowledge in language models back to the training data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2429–2446.
- Hady Elsahar, Pavlos Vougiouklis, Arslan Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.
- Hoang Anh Just, Feiyang Kang, Jiachen T Wang, Yi Zeng, Myeongseob Ko, Ming Jin, and Ruoxi Jia. 2023. Lava: Data valuation without pre-specified learning algorithms. *arXiv preprint arXiv:2305.00054*.
- Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.
- Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.
- Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.
- Yuanhua Lv and ChengXiang Zhai. 2011. Lower-bounding term frequency normalization. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 7–16.
- Master of Code. 2023. Hallucinations in llms: What you need to know before integration. <https://masterofcode.com/blog/hallucinations-in-llms-what-you-need-to-know-before-int>. Accessed: 2024-02-15.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

- Shi Na, Liu Xumin, and Guan Yong. 2010. Research on k-means clustering algorithm: An improved k-means clustering algorithm. In *2010 Third International Symposium on intelligent information technology and security informatics*, pages 63–67. Ieee.
- Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.
- Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. 2023. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*.
- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. [Language models as knowledge bases?](#) 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.
- Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930.
- Nazneen Fatema Rajani, Ben Krause, Wengpeng Yin, Tong Niu, Richard Socher, and Caiming Xiong. 2020. Explaining and improving model behavior with k nearest neighbor representations. *arXiv preprint arXiv:2010.09030*.
- Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.
- Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.
- Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. [Get your vitamin C! robust fact verification with contrastive evidence.](#) In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.
- Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.
- Sagar Vaze, Andrea Vedaldi, and Andrew Zisserman. 2023. No representation rules them all in category discovery. *arXiv preprint arXiv:2311.17055*.
- Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.
- Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. [mT5: A massively multilingual pre-trained text-to-text transformer.](#) In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.
- Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Lan Luo, Ke Zhan, Enrui Hu, Xinyu Zhang, Hao Jiang, Zhao Cao, Fan Yu, Xin Jiang, Qun Liu, and Lei Chen. 2022. [Hyperlink-induced pre-training for passage retrieval in open-domain question answering.](#) In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7135–7146, Dublin, Ireland. Association for Computational Linguistics.

## A Algorithm of FASTTRACK

---

### Algorithm 1: FASTTRACK Workflow

---

**Input:** Query set  $Q$ , training corpus  $D$ , instruction prompt for keyword assignment  $Inst_{key}$ , instruction prompt for supportiveness evaluation  $Inst_{eval}$

**Output:** Retrieved Samples  $D_{sel}$

```

/* Stage 1: Semantic Clustering (Offline) */
1  $D_{emb} \leftarrow SentenceTransformer(D)$ 
  Leaf Clusters  $C = \{c_0, c_1, \dots, c_{n-1}\} \leftarrow$ 
  Hierarchical clustering on  $D_{emb}$  using
  k-Means (k=10)
2 Semantic Labels  $J = \{j_0, j_1, \dots, j_{n-1}\} \leftarrow$ 
  LLM( $\{c_0, c_1, \dots, c_{n-1}\}, Inst_{key}$ )
/* Stage 2: Tracing (Online) */
3 for each query  $q \in Q$  do
4    $D_q \leftarrow \{\}$ 
5    $C_{sel} \leftarrow fuzzymatch(q, J, C)$ 
6    $Batches \leftarrow$  partition  $C_{sel}$  into batches
     of size  $b$ 
7   for each batch  $B \in Batches$  do
8      $S_B \leftarrow LLM(q, B, Inst_{eval})$ 
9      $D_q \leftarrow D_q \cup \{z \mid z \in B, s_i = 1\}$ 
10  end
11  $D_{sel} \leftarrow D_{sel} \cup D_q$ 
12 end

```

---

## B Extended Related Work

**Training Data Attribution (TDA).** TDA aims to trace model predictions back to the training examples that responsible for these predictions. As it shares a similar goal with fact tracing, prior work (Akyürek et al., 2022) proposes to use two popular families of TDA methods as baselines. Gradient-based attribution, for instance, focuses on quantifying the direct influence a particular training example  $z$  has on the loss at a test example  $z_{query}$ , when using a model parameterized by  $\theta$ . A notable technique within this category is TRACIn (Pruthi et al., 2020). It employs a first-order Taylor approximation to estimate the loss change on  $z_{query}$  when taking a gradient step on training example  $z$  at time  $t$ . The resulting attribution score is simply a dot product of gradients at a particular step  $t$ :

$$\mathcal{I}_t(z, z_{query}) = \nabla_{\theta} L(z_{query}, \theta_t)^\top \nabla_{\theta} L(z, \theta_t) \quad (1)$$

Embedding-based attribution employs the model’s internal representations to determine the relevance of training examples to a given test prediction (Rajani et al., 2019). The attribution score is defined as a cosine product of hidden representations:

$$\mathcal{I}(z, z_{query}) = \frac{LM_{inter.}(z)^\top LM_{inter.}(z_{query})}{\|LM_{inter.}(z)\| \|LM_{inter.}(z_{query})\|} \quad (2)$$

To retrieve supporting training data for a given query  $z_{query}$ , one need to score every training data and rank them by their influence score. However, TDA methods fail to justify groundness and often perform worse than simple IR baseline (i.e., BM25) (Akyürek et al., 2022). Moreover, it could be computationally infeasible for gradient-based TDA scoring all training data in large datasets, and relies on evaluation on carefully selected smallsubset (i.e., around 500) for each query. This limitation motivates us to design a framework that is both more computationally efficient and more effective.

**Information Retrieval (IR).** IR focuses on retrieving relevant documents in a large collection given specific queries (Izacard et al., 2021). Though not theoretically justified for fact tracing task, prior work (Akyürek et al., 2022) found it could serve as a possible solution and outperforms principled TDA methods by a large margin. There are two branches of IR methods: term-frequency based (Mikolov et al., 2013; Lv and Zhai, 2011; Robertson et al., 1995) and neural network based (Karpukhin et al., 2020; Xiong et al., 2020; Izacard et al., 2021; Ni et al., 2021). A classic example of the former one is BM25 (Robertson et al., 1995; Lv and Zhai, 2011), which represents the best performing variant of lexical similarity-based IR methods. When using BM25 for fact tracing, one can treat examples as a bag of words, and score each training data base on the token overlap with the given query, inversely weighted with the frequency of such tokens. On the other hand, neural network-based methods often require labor-intensive annotations on query-document pairs (Karpukhin et al., 2020; Xiong et al., 2020). This making them impractical in the fact tracing scenario where such annotations are not available. While some recent works (Izacard et al., 2021; Ni et al., 2021) propose to over-

come the limitation using zero-shot learning, they usually results in an inferior retrieval quality, even worse than a non-parametric BM25 (Thakur et al., 2021; Zhou et al., 2022). Thus, we follow Akyürek et al. (2022) and choose BM25 as IR baseline for fact tracing.

Similar to TDA methods, IR methods also focus on *relevance* while neglecting the *supportiveness* of the connection between training data and the query. In this paper, we introduce FASTTRACK, the first supportiveness-aware approach for fact tracing, offering substantial benefits in scenarios where training data contains conflicting information, such as time-sensitive facts in news reports.

## C Baseline Implementation Details

**TRACIN** We follow the setup of Akyürek et al. (2022), optimize TRACIN by recaling gradient with Adafactor accumulators, applying unit-normalization to the gradients and selecting the best performing layer, which is first encoder layer. In practice, it is found that aggregating over multiple checkpoints often does not lead to improved performance but raises the computational burden. Thus, it is preferred to just use a single checkpoint that gives the best results (Just et al., 2023). For FTRACE-TREx dataset, we use a MT5 model finetuned on it for 80k steps.

**EMBED** We use the same MT5 checkpoint as for TRACIN on FTRACE-Trex dataset. For VITAMINC dataset, we finetune a Bert model by randomly masking some tokens of the training data. We observe that best performing layer is the last encoder layer of Bert and use results of this layer as the final results.

## D FASTTRACK Implementation Details

A detailed algorithm of FASTTRACK is given in 1. When perform hierarchical clustering, we employ k-means (Na et al., 2010). The clustering process is applied recursively: clusters with more than  $c$  samples will be clustered again until it contains less than  $c$  samples. We use  $c = 100, k = 10$  for all experiments.

For keyword assignment, we set temperature to be 0 and output length to be 256 when calling GPT-4 api. For the batch process at Stage 2, we use a batch size  $b = 15$ . We set temperture to be 0 and output length to be 1024. Prompts used can be found in Section F.

## E More Results

**BM25’s performance before and after paraphrasing queries** BM25 operates based on token overlap, and retrieves examples with high lexical similarity to the query, regardless of their factual consistency. Its performance can be poor even when there are no such data conflicts. We further conduct experiment on FTRACE-TREx dataset where we paraphrase each query using an open-sourced paraphraser<sup>8</sup>. The performance of BM25 before and after paraphrasing is shown in Table 4, where both precision and recall drop by a wide margin.

### TRACIN’s performance when using different model checkpoints

**FASTTRACK’s performance with larger corpus size.** Akyürek et al. (2022) benchmark tracing methods only on a curated small candidate set with about 500 examples for each query. In contrast, our benchmark has assessed the efficacy and efficiency of our method on a significantly larger corpus, containing 10k instances. Moving forward, we aim to tackle a more challenging scenario: we use the same query set of VITAMINC, but augment the attribution set with additional non-relevant examples until the total reaches 100k. This setting is designed to evaluate our method’s robustness in scenarios that better resemble real-world applications. As shown in Table 7, both methods exhibit a slight decline in performance, yet FASTTRACK consistently outperforms BM25 by a significant margin.

BM25’s performance drop is ascribed to the inclusion of new examples that exhibit high lexical overlap with the queries, thereby impairing BM25’s effectiveness. For FASTTRACK, the performance drop is primarily observed in the initial stage, where some clusters are not successfully retrieved. This results from the clustering logic changes after more diverse sample are included. Specifically, if a cluster contains only a few groundtruth examples, these may be overlooked during the semantic labeling process, leading to false negative retrieval. Despite this minor reduction in performance, FASTTRACK consistently outperforms BM25 by a significant margin.

**Failure example of FASTTRACK.** Example below shows that GPT-4 struggles with complex logi-

<sup>8</sup>[https://huggingface.co/humarin/chatgpt\\_paraphraser\\_on\\_T5\\_base](https://huggingface.co/humarin/chatgpt_paraphraser_on_T5_base)



Table 4: BM25’s performance before and after paraphrasing queries in FTRACE-TREx dataset. Notably, BM25 exhibits a 21 percentage point drop in precision, while FASTTRACK maintains consistent performance, achieving a precision of 0.81 and a recall of 0.69.

|               | <b>Top-1</b>     |               | <b>Top-10</b>    |               | <b>Top-25</b>    |               |
|---------------|------------------|---------------|------------------|---------------|------------------|---------------|
|               | <b>Precision</b> | <b>Recall</b> | <b>Precision</b> | <b>Recall</b> | <b>Precision</b> | <b>Recall</b> |
| <b>Before</b> | 0.83             | 0.06          | 0.66             | 0.36          | 0.49             | 0.52          |
| <b>After</b>  | 0.62             | 0.05          | 0.48             | 0.28          | 0.38             | 0.42          |

Table 5: TRACIN’s performance using checkpoints at gradient steps 30k and 80k

|            | <b>Top-1</b>     |               | <b>Top-10</b>    |               | <b>Top-25</b>    |               |
|------------|------------------|---------------|------------------|---------------|------------------|---------------|
|            | <b>Precision</b> | <b>Recall</b> | <b>Precision</b> | <b>Recall</b> | <b>Precision</b> | <b>Recall</b> |
| <b>30k</b> | 0.10             | 0.003         | 0.02             | 0.01          | 0.01             | 0.01          |
| <b>80k</b> | 0.19             | 0.01          | 0.05             | 0.02          | 0.03             | 0.03          |

cal reasoning involving dates; for instance, it incorrectly equates the information from different dates, focusing merely on numerical comparisons.

**Query:** As of 14 March , there have been more than 20 confirmed COVID-19 cases in Kerala .

**False positive examples:**

1: 2020 coronavirus pandemic in Kerala: As of 28th March 2020 , there are 182 confirmed cases of the virus and more than 1,15,000 people are under surveillance in the state .

2: 2020 coronavirus pandemic in Kerala: As of 29th March 2020 , there are 202 confirmed cases of the virus and more than 1,41,000 people are under surveillance in the state .

**GPT-4 analysis:**

Text 1: States that "As of 28th March 2020, there are 182 confirmed cases of the virus in Kerala." This text supports the claim because if there were 182 cases by the 28th of March, there must have been more than 20 cases by the 14th of March.

Text 2: States that "As of 29th March 2020, there are 202 confirmed cases of the virus in Kerala." This text also supports the claim because if there were 202 cases by the 29th of March, there must have been more than 20 cases by the 14th of March.

## F Prompting Templates

Table 6: Upperbound performance of FASTTRACK using different clustering algorithms on FTRACE-TREx. Different embedding models do not bring much effects on the performance upperbound, demonstrating the robustness of FASTTRACK on the choice of embeddings. *\*We list GPU time if it is an open-sourced model deployed on our server and costs if it is accessed through queries to the API.*

| Embedding Scheme       | Precision   | Recall      | Time/Costs* |
|------------------------|-------------|-------------|-------------|
| Sentence-Transformer   | 0.81        | <b>0.65</b> | 0.16 min    |
| Cohere Clustering      | 0.80        | 0.63        | \$ 0.04     |
| Cohere Classification  | 0.75        | 0.57        | \$ 0.04     |
| Cohere Search-Query    | 0.79        | 0.60        | \$ 0.04     |
| Cohere Search-Document | <b>0.82</b> | 0.56        | \$ 0.04     |
| Mistral-Embed          | 0.69        | 0.53        | \$0.05      |

Table 7: Performance of BM25 and FASTTRACK when dealing with different corpus size. Both of the methods encounter a slightly performance drop, while FASTTRACK still  $1.66\times$  better than BM25.

|              | VITAMINC-10k |           |        | VITAMINC-100k |           |        |
|--------------|--------------|-----------|--------|---------------|-----------|--------|
|              | F1           | Precision | Recall | F1            | Precision | Recall |
| <b>BM25</b>  | 0.55         | 0.59      | 0.53   | 0.53          | 0.56      | 0.50   |
| <b>Ours</b>  | 0.91         | 0.88      | 0.98   | 0.88          | 0.85      | 0.92   |
| <b>Ours*</b> | 1.00         | 1.00      | 1.00   | 0.95          | 0.95      | 0.95   |

Figure 3: Prompt template for keyword assignment.

**Prompt:**

Analyze the following group of sentences and identify 5 to 10 of phrases that capture the main topics, focusing on the key entities.

Group of sentences: [cluster texts]

Output the keywords in the following format: #keywords: your keywords here. [note: Extract 5 to 10 keywords only. The keywords should capture the main topic and seperated by comma.]

Figure 4: Prompt template for supportiveness evaluation on FTRACE-TREx dataset.

**Prompt:**

I will give you a claim and multiple texts. Carefully evaluate each text, check if the text supports the claim.

For example,

Fact: Member of the Scottish Parliament is a legal term in Scotland.

Group of Texts:

Text 1: Dennis Robertson is a Scottish politician, and has been an Member of the Scottish Parliament (MSP) for Aberdeenshire West since 2011, after defeating the Liberal Democrat incumbent, Mike Rumbles, by a majority of 4,112 votes.

Text 2: The West Lothian question, also known as the English question, refers to whether MPs from Northern Ireland, Scotland and Wales, sitting in the House of Commons of the United Kingdom, should be able to vote on matters that affect only England, while MPs from England are unable to vote on matters that have been devolved to the Northern Ireland Assembly, the Scottish Parliament and the Welsh Assembly.

#analysis: A "legal term" refers to a term or expression that is associated with or used in the formal context of a particular country. Text1 mentions that "Member of the Scottish Parliament (MSP)" is in Scotland; because we can infer that member of the scottish parliament is used in the formal political context of Scotland, it implicitly establishes that Member of the Scottish Parliament is a legal term in Scotland. Text2 mentions the Scottish Parliament but does not state that "Member of the Scottish Parliament" is a legal term used within the context of the Scotland.

#scores: 1, 0

Fact: [query]

Group of Texts: [indexed candidate training data]

Now evaluate each text carefully in the group and output in the following format:

#analysis: your analysis here.

#scores: your scores here. (score each text 0 or 1 according to the analysis.)

Figure 5: Prompt template for supportiveness evaluation on VITAMINC dataset.

**Prompt:**

I will give you a claim and multiple texts. Carefully evaluate each text, check if the text supports the claim.

For example,

Claim: Black Mass earned less than \$ 43.6 million in North America as of 22 April 2020.

Group of Texts:

Text 1: Black Mass has grossed less than \$ 42.6 million in North America as of 22 April 2020.

Text 2: Black Mass has grossed less than \$ 42.6 million in North America as of 12 April 2020.

Text 3: Black Mass has grossed more than \$ 42.6 million in North America as of 22 April 2020.

Text 4: Black Mass has grossed more than \$ 43.6 million in North America as of 22 April 2020.

#analysis:

Text 1: States that "Black Mass has grossed less than \$42.6 million in North America as of 22 April 2020." This text supports the claim because if it grossed less than \$42.6 million, it also grossed less than \$43.6 million.

Text 2: States that "Black Mass has grossed less than \$42.6 million in North America as of 12 April 2020." This text does not directly support or refute the claim because it provides information as of 12 April. Without specific information on the movie's earnings trends or events that might have affected its box office performance between 12 April and 22 April 2020, it is impossible to determine whether the gross was less than \$43.6 million as of 22 April.

Text 3: States that "Black Mass has grossed more than \$42.6 million in North America as of 22 April 2020." This text does not directly support or refute the claim because it does not provide enough information to determine whether the gross was less than \$43.6 million. It only indicates that the gross was more than \$42.6 million, which could still be less than \$43.6 million.

Text 4: States that "Black Mass has grossed more than \$43.6 million in North America as of 22 April 2020." This text does not support the claim because it directly contradicts it, indicating that the gross was more than \$43.6 million.

#scores: 1, 0, 0

Fact: [query]

Group of Texts: [indexed candidate training data]

Now evaluate each text carefully in the group and output in the following format:

#analysis: your analysis here.

#scores: your scores here. (score each text 0 or 1 according to the analysis.)