When Claims Evolve: Evaluating and Enhancing the Robustness of Embedding Models Against Misinformation Edits

Jabez Magomere,[1] Emanuele La Malfa,[1,2] Manuel Tonneau,[1,3,4]
Ashkan Kazemi,[5] Scott A. Hale[1,5]
[1]University of Oxford, [2]Alan Turing Institute, [3]World Bank,
[4]New York University, [5]Meedan
jabez.magomere@keble.ox.ac.uk

Abstract

Online misinformation remains a critical challenge, and fact-checkers increasingly rely on claim matching systems that use sentence embedding models to retrieve relevant fact-checks. However, as users interact with claims online, they often introduce edits, and it remains unclear whether current embedding models used in retrieval are robust to such edits. To investigate this, we introduce a perturbation framework that generates valid and natural claim variations, enabling us to assess the robustness of a wide-range of sentence embedding models in a multi-stage retrieval pipeline and evaluate the effectiveness of various mitigation approaches. Our evaluation reveals that standard embedding models exhibit notable performance drops on edited claims, while LLM-distilled embedding models offer improved robustness at a higher computational cost. Although a strong reranker helps to reduce the performance drop, it cannot fully compensate for first-stage retrieval gaps. To address these retrieval gaps, we evaluate train- and inference-time mitigation approaches, demonstrating that they can improve in-domain robustness by up to 17 percentage points and boost out-of-domain generalization by 10 percentage points. Overall, our findings provide practical improvements to claim-matching systems, enabling more reliable fact-checking of evolving misinformation.

1 Introduction

The spread of misinformation, false or inaccurate information, is considered to be one of the biggest short-term threats to the cohesion within and between nations (WEF, 2025). While misinformation spreads fast (Vosoughi et al., 2018), manual verification and debunking of false claims takes time.

To speed up the fact-checking process, a growing body of research has focused on developing tools to help support human fact-checkers (Nakov et al., 2021a; Guo et al., 2022; Das et al., 2023).

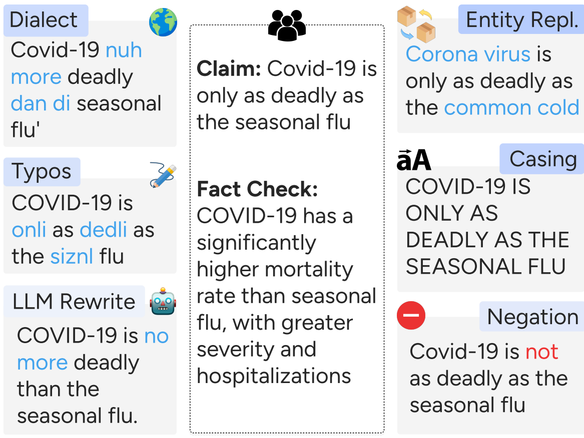

Figure 1: **Examples of user-informed misinformation edits observed in fact-checking**. Users frequently introduce subtle edits to online misinformation claims that preserve the corresponding fact-check. However, it remains unclear whether sentence embedding models are robust to these edits.

One such approach is **claim matching**, the task of determining whether a given claim has already been fact-checked or retrieving relevant fact-checks (Shaar et al., 2020; Sheng et al., 2021; Shaar et al., 2022; Kazemi et al., 2021). Claim matching has shown strong potential in reducing the time human fact-checkers spend verifying repeated or previously debunked claims, by ensuring that once a costly human fact-check is written, it can be linked to all relevant instances of that claim to provide context to others encountering it. However, current claim matching approaches are typically trained on datasets composed of previously fact-checked claims (Barrón-Cedeno et al., 2020; Nakov et al., 2022), excluding the vast majority of unverified claims in the wild. This limits models' exposure to the noisy, diverse, and evolving nature of misinformation as it appears in real-world contexts. As a result, these models may perform well in controlled, in-domain evaluations but generalize poorly to settings characterized by unpredictable

Findings of the Association for Computational Linguistics: ACL 2025, pages 22374–22404

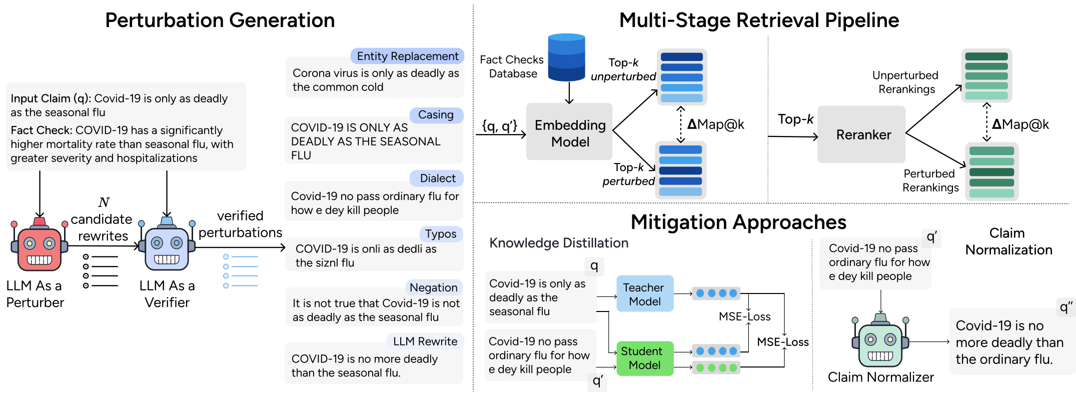

Figure 2: **An Overview of Our Approach**: (1) We generate candidate rewrites using an *LLM As a Perturber*, followed by another *LLM As a Verifier* of these perturbations. (2) Next, we assess the robustness of various embedding models within a multi-stage retrieval pipeline. (3) Finally, we examine the impact of train- and inference-time mitigation approaches on improving robustness for weaker embedding models.

user behaviour.

As users interact with and share misinformation online, they often introduce various edits to claims, such as replacing entities, introducing common social-media slang, or rewriting a claim in a different dialect (Kazemi et al., 2023; Quelle et al., 2025) as shown in Figure 1. While these edits may be subtle and not affect the matching fact-check, Yan et al. (2022) show that **mutated claims are more likely to spread widely on social media compared to non-mutated claims**. Thus, current claim-matching methods need to be robust to user claim edits to curb the spread of misinformation effectively.

At the core of claim matching tasks is the reliance on sentence embedding models to represent input claims and fact-checks during retrieval. There has been growing interest in developing general-purpose sentence embeddings designed to handle a wide range of tasks (Lee et al., 2024; Su et al., 2023). However, their effectiveness in task-specific applications remains uncertain, as different domains demand varying notions of similarity (Ghafouri et al., 2024). Fact-checking is one such domain where fact-checkers need to retrieve sentences that may be evaluated under the same factual basis, even if they are not strictly semantically equivalent, a nuance that general-purpose embeddings mostly trained on semantic textual similarity datasets (Cer et al., 2017) may struggle to capture.

To tackle the challenge of embedding model brittleness in real-world settings, we study **the robustness of sentence embedding models to user-driven misinformation edits** and **propose mitigation strategies to improve robustness**. We introduce a taxonomy of real-world edits drawn from fact-checking scenarios and evaluate model robustness on the downstream task of claim matching (Figure 2). Our analysis shows that widely used embedding models are highly sensitive to user-informed misinformation edits, and that even strong rerankers fail to fully mitigate these weaknesses. However, we demonstrate that both train-time and inference-time mitigation approaches can significantly improve in-domain and out-of-domain robustness. Based on these findings, our key contributions are as follows:[1]

- We propose a novel perturbation framework for generating valid and natural claim edits, resulting in a new dataset for evaluating robustness of sentence embedding models.
- We conduct a comprehensive evaluation of a wide range of embedding models, including BERT-based, T5-based, and LLM-distilled in a multi-stage retrieval pipeline comprising first-stage retrieval followed by reranking.
- We assess the effectiveness of both train-time and inference-time mitigation approaches in improving the robustness of weaker embedding models, providing insights into their trade-offs and effectiveness.

2 Related Work

Robustness in Misinformation Detection Robustness in NLP is a well-studied area, with research focusing on developing adversarial attacks that can fool models and compromise real-world deployments (Malfa and Kwiatkowska, 2022; Wang

[1]Code and datasets are available for research purposes.

et al., 2022; Goyal et al., 2023). In the context of misinformation, prior work has explored adversarial robustness in text classifiers for fake news detection (Zhou et al., 2019; Le et al., 2020; Flores and Hao, 2022), veracity prediction (Thorne et al., 2019; Hidey et al., 2020), and propaganda detection (Przybyła et al., 2024). However, fact checkers have evidenced a lack of transparency and credibility on automated fact-checking approaches that assign factuality labels to claims (Glockner et al., 2022; Das et al., 2023; Procter et al., 2023). In contrast, the robustness of *claim matching*—a core information retrieval task—remains underexplored, with prior research focusing on improving retrieval performance (Barrón-Cedeno et al., 2020; Sheng et al., 2021; Nakov et al., 2021b, 2022). Most studies on robustness in misinformation rely on adversarial techniques, such as character- and word-level substitutions (e.g., synonym replacements: Jin et al., 2019; Li et al., 2020; Garg and Ramakrishnan, 2020), to perturb input claims. However, the practicality of these techniques has been questioned, as recent studies show that such attacks can mislead NLP models but have little impact on human readers (Morris et al., 2020a; Dyrmishi et al., 2023a; Chiang and Lee, 2023). There is also a well-known trade-off between how realistic a local adversarial attack is and its plausibility for humans (Malfa and Kwiatkowska, 2022). This issue is even more critical in misinformation contexts, where the goal is to not only fool a model but also to generate believable claims and spread content among human users; therefore, the perturbations should be both valid and natural (Dyrmishi et al., 2023a). In contrast to adversarial robustness which focuses on exploiting model vulnerabilities, our work focuses on *realistic user edits*: natural variations in misinformation that arise organically, either intentionally or unintentionally, as claims spread online.

Evaluating Embedding Model Robustness Several studies have examined the robustness of embedding models across different settings. Rafiei Asl et al. (2024) evaluate robustness of embedding models adversarially using TextAttack (Morris et al., 2020b) and propose using gradient-based (Goodfellow et al., 2014) perturbed embeddings to enhance robustness. Chiang et al. (2023) examine how different sentence encoders represent sentence pairs with high lexical overlap, showing that training datasets influence similarity notions. Closely related to our work, Nishimwe et al. (2024) assess LASER (Heffernan et al., 2022) against user-generated content and apply knowledge distillation (Reimers and Gurevych, 2020) to improve robustness. However, prior work largely evaluates embeddings in isolation, despite their role in larger systems: e.g., candidate selection in multi-stage retrieval (Nogueira et al., 2020) or retrieval-augmented generation (RAG) pipelines (Fan et al., 2024). The impact of representation differences on end-to-end system performance remains unclear. For example, we examine whether the robustness of embedding models in first-stage retrieval meaningfully affects the overall performance of a multi-stage retrieval system when a strong reranker is applied, addressing this gap.

3 Data and Methods

3.1 Background

3.1.1 Task Definition

Following Shaar et al. (2020), we define the task of claim matching as a *learning-to-rank problem*. Given a check-worthy input claim q and a set of verified claims or fact-checks $V = \{v_1, v_2, \ldots, v_n\}$, the objective is to learn a ranking function f that orders the verified claims such that those that can help verify the input claim q are ranked higher. f returns a ranked list $[v_{\pi(1)}, v_{\pi(2)}, \ldots, v_{\pi(n)}]$, where π is a permutation of indices that sorts the claims in descending order of relevance to q. Model performance on this task is evaluated using Mean Average Precision (MAP) score truncated to rank k (MAP@k). We compare claims with a sentence embedding model θ that encodes the input claim and fact-checks into dense vector representations. These embeddings are then utilized in learning the ranking function f.

3.1.2 Problem Formulation

We consider a scenario where a social media user u edits an original claim q to produce a perturbed claim q'. The edits must ensure that q' is both ***valid***, remaining applicable to the fact-check v and conveying the same claim as q and ***natural***, meaning that q' is something a social media user might write. These edits are modeled as a set of transformations $\mathcal{T}$, each applied with a noise level b drawn from a budget $B_t = [b^t_{\text{low}}, b^t_{\text{high}}]$ for $t \in \mathcal{T}$. We vary the noise levels to capture the spectrum of user edits, from minimal edits (low noise) to substantial rewording (high noise). A set of constraints $\mathcal{C}$ enforces the validity and naturalness of q'. Formally, this process is described by

$$q' = t_b(q) \quad \text{with} \quad \mathcal{C}(q, q', v) = \text{True} \qquad (1)$$

where given an input claim q, the *baseline edit* is defined as $q^t_{\text{base}} = t_{b^t_{\text{low}}}(q)$ and the *worst-case edit* as $q^t_{\text{worst}} = t_{b^t_{\text{high}}}(q)$. This formulation systematically examines how varying noise levels within B_t impact sentence embedding models in claim matching tasks.

3.1.3 Perturbations as Claim Edits

We introduce a taxonomy of common misinformation edits observed in real-world fact-checking, which serves as a template for the transformations $t \in \mathcal{T}$. The applied transformations and noise levels are as follows:

Casing A subtle transformation observed in claims involves changes in casing, such as uppercasing an entire claim or parts of it to add emphasis. In other cases, casing differences may be artefacts of the noisy nature of social media text (Ritter et al., 2011; Baldwin et al., 2015). To capture casing variations, we apply `TrueCasing` (Lita et al., 2003)[2] to the original claim as the minimum amount of noise (b_{low}) and `UPPERCASE` the entire input claim as the maximum noise level (b_{high}).

Typos User edits to a claim may introduce spelling errors, common social media abbreviations and slang (Van Der Goot et al., 2018; Sanguinetti et al., 2020). We define a transformation budget based on the *Levenshtein* edit distance between the unperturbed and perturbed input claims. The baseline budget, b_{low}, corresponds to the perturbed claim with the smallest edit distance, while the worst-case budget, b_{high}, represents the largest edit distance to the unperturbed claim.

Negation Social media users may negate claims, for instance as part of counterspeech (Gligoric et al., 2024), though the underlying claim to be verified remains unchanged. To evaluate embedding models on such cases, we apply negation transformations to original claims, defining the minimum noise level (b_{low}) as a single negation and the maximum noise level (b_{high}) as a double negation. For example, the claim "Covid-19 is only as deadly as the seasonal flu" with a single negation becomes "Covid-19 is not as deadly as the seasonal flu". With double negation the same claim becomes "It is not true that Covid-19 is not as deadly as the seasonal flu".

Entity Replacement One common pattern in mutated claims is the substitution of entities in the original text with synonymous references, such as replacing a well-known person's name with a nickname or changing entities to tailor a claim to a new audience; for example, changing "According to a Harvard study ..." to "According to an Oxford study ..." while retaining the original meaning. We define the transformation budget as the number of named entities to be replaced in a claim, with $b_{\text{low}} = 1$, meaning at most one entity is replaced, and $b_{\text{high}} = All$, meaning all entities are replaced.

LLM rewrites Chen and Shu (2024) demonstrate that LLMs can be used to rewrite a claim, either to conceal authorship or to make the claim more deceptive and harder to detect. We prompt an LLM to rewrite an original claim, setting the transformation budget based on the *Levenshtein* distance, where b_{low} is the rewrite with the fewest changes, and b_{high} is the rewrite with the most changes.

Dialect Changes A social media user may rewrite a claim in a different dialect to fit a specific target audience. However, most NLP models are primarily trained on American and British English (Longpre et al., 2024; Tonneau et al., 2024a), leading to notable performance drops when applied to other English dialects (Jurgens et al., 2017; Ziems et al., 2023; Tonneau et al., 2024b). To study the effect of dialectal variations, we rewrite claims in *African American English*, *Nigerian Pidgin English*, *Singlish* (an English variant spoken in Singapore) and *Jamaican Patois*, an English-based creole spoken in the Caribbean.[3]

3.2 Experimental Setup

3.2.1 Perturbation Generation Framework

To generate perturbations that meet the constraints outlined in Section §3.1.2, we implement a two-stage, multi-prompting framework leveraging two separate LLMs: one for perturbing input claims and another for verifying them. In the **perturbation step**, the *perturber* LLM takes an input claim q and its corresponding fact-check v as inputs to generate an initial set of N candidate rewrites intended to satisfy the transformation requirements and constraints. To select valid perturbations, we implement a **verification step**, where another LLM

[2] A preprocessing technique that converts all-lowercase text to properly capitalized text and introduces variations in text.

[3] We treat all dialects equal and assume similar noise levels.

assumes the role of *verifier* and checks whether the generated rewrites adhere to all constraints. Each perturbation is assigned a binary label (1 for valid, 0 for invalid), and invalid perturbations are discarded. We set the initial candidate size N to 5 and use `GPT4o` as both the *perturber* and *verifier*. A human evaluation of the perturbation framework shows that the *perturber* applies perturbations with 96.70% accuracy, while the *verifier* achieves 100% precision in selecting valid perturbations. However, the verifier's lower recall of 55.67% suggests that it may be overly conservative, potentially filtering out some valid instances. Further details on the human evaluation and prompts are provided in Sections §B and §I, respectively.

3.2.2 Datasets

We apply perturbations to two claim-matching datasets: *CheckThat22* (Nakov et al., 2022) and the dataset proposed by Kazemi et al. (2022), referred thereafter as *FactCheckTweet*. *CheckThat22* matches tweets with short, verified claims, whereas *FactCheckTweet* matches claims with full fact-check articles. We apply perturbations to the test splits only (see Appendix §A). To assess generalization i.e. ability to improve performance on novel, out-of-distribution claims, we test our mitigation approaches on an out-of-domain (*OOD Dataset*) composed of novel claim–fact-check pairs provided by Meedan[4] and primarily sourced through five fact-checking organizations. Table 1 summarizes the descriptive statistics of these datasets.

Dataset	# Claim	# FC	Claim Len	FC Len
CheckThat22	2,592	14,231	42.3	19.2
FactCheckTweet	3,578	59,951	37.9	682.1
OOD Dataset	721	1,261	150.1	80.4

Table 1: Statistics of datasets in this work. FC denotes target fact-checks; lengths are measured in words.

3.2.3 Multi-Stage Retrieval Pipeline

To evaluate the robustness of various embedding models, we adopt a common multi-stage retrieval pipeline (Nogueira and Cho, 2019; Nogueira et al., 2019a; Ma et al., 2024) consisting of a computationally efficient *retriever* model to return top-j relevant fact-checks followed by a more computationally intensive *reranker* that refines the retrieved candidates to improve ranking quality. We describe the models we evaluate for each stage below:

[4]The non-profit Meedan supports fact-checkers running misinformation tiplines on WhatsApp and other platforms.

Retrievers In the retrieval stage, we evaluate embedding models with varying dimensions, pre-training approaches, and backbone architectures. This includes: **BERT-based embedding models** (Devlin et al., 2019; Reimers and Gurevych, 2019): `all-MiniLM-L12-v2`, `all-mpnet-base-v2`, and `all-distilroberta-v1`. **T5-based embedding models** (Ni et al., 2022; Raffel et al., 2020): `sentence-t5-base`, `sentence-t5-large`, `instructor-base`, and `instructor-large`—instruction-finetuned embeddings based on GTR models (Su et al., 2023). **Decoder-only LLM-distilled embeddings**: `SFR-Embedding-Mistral` (Meng et al., 2024) and `NV-Embed-v2` (Lee et al., 2025),[5] both initialized from `Mistral-7B`. To study the effect of finetuning, we finetune `all-mpnet-base-v2` and `sentence-t5-large` on the *CheckThat22* training set, and denote the finetuned models respectively as `all-mpnet-base-v2-ft` and `sentence-t5-large-ft`. We also implement **BM25 as a baseline** to explore the robustness of lexical-based sparse retrieval approaches. We provide further details on embedding models and finetuning in §C of the Appendix.

Rerankers With multiple combinations of embedding models ($m = 12$), perturbation types ($p = 14$), and rerankers ($r = 7$)—totalling $m \times p \times r = 1,176$ cases, we evaluate a wide range of rerankers and select the best-performing one for further experiments. While `RankGPT` (Sun et al., 2023) using `GPT4o` achieves the highest performance, we select `bge-reranker-v2-gemma` (Chen et al., 2024) for its comparable accuracy and significantly lower inference cost. See §D.1 in the Appendix for reranker evaluations.

4 Results

4.1 RQ1: How do perturbations affect the robustness of embedding models in first-stage retrieval?

We begin by evaluating the robustness of embedding models used for first-stage retrieval under various perturbation types. For each embedding model and perturbation type, we compute the Δ*retrieval gap* by comparing the MAP@k performance difference between the unperturbed and

[5]`NV-Embed-v2` was the top-performing model on MTEB (Muennighoff et al., 2022) at the time of this study. `https://huggingface.co/spaces/mteb/leaderboard`

Stage	Model	Casing: *TrueCase*	Casing: *Upper*	Typos: *Least*	Typos: *Most*	Negation: *Shallow*	Negation: *Double*	Entity Replacement: *Atleast 1*	Entity Replacement: *All*	LLM Rewrite: *Least*	LLM Rewrite: *Most*	Dialect: *AAE*	Dialect: *Jamaican*	Dialect: *Pidgin*	Dialect: *Singlish*
	CheckThat22														
First-Stage Retrieval	`BM25`	+0.0	+0.0	-15.2	-15.0	+0.5	-2.6	-2.7	-12.4	-0.8	-0.2	-9.2	-6.7	-5.0	-1.6
	`all-distilroberta-v1`	-0.4	-35.4	-15.8	-13.9	+1.8	+2.4	-3.0	-5.4	+3.5	+5.3	-4.4	-11.4	-6.6	-0.7
	`all-MiniLM-L12-v2`	+0.0	+0.0	-8.2	-13.6	+3.5	+4.4	-2.9	-2.5	+3.1	+4.4	-6.7	-10.0	-5.8	+0.2
	`all-mpnet-base-v2`	+0.0	+0.0	-8.3	-8.4	+0.2	-4.2	-3.1	-9.8	+1.0	+1.8	-3.8	-7.5	-8.5	-1.3
	`all-mpnet-base-v2-ft`	+0.0	+0.0	-8.6	-12.8	-1.4	-4.2	-1.8	-8.6	-1.3	-1.7	-3.8	-4.0	-8.5	-3.2
	`sentence-t5-base`	-2.5	-15.8	-2.9	-7.1	-21.4	-12.2	-3.3	-5.7	+2.3	+2.0	-0.9	-8.2	-5.5	-0.1
	`sentence-t5-large`	-1.0	-9.9	-2.1	-5.4	-27.2	-14.7	-0.9	-5.4	+1.8	+1.7	+0.8	-3.4	-3.2	+2.0
	`sentence-t5-large-ft`	-0.6	-9.7	-1.8	-3.6	-11.7	-6.6	-2.5	-1.6	+2.2	+1.0	+1.5	-2.4	-3.7	+1.2
	`instructor-base`	-1.1	-9.4	-4.4	-3.2	-3.7	-3.8	-1.7	-8.0	+0.1	-0.8	-1.1	-4.0	-4.3	-1.5
	`instructor-large`	-0.7	-4.5	-1.1	-1.7	-4.1	-4.9	+0.1	-2.5	+0.0	-0.9	-0.9	-2.5	-4.1	-2.0
	`SFR-Embedding-Mistral`	-0.6	-0.6	-0.7	-0.5	-1.2	-2.9	-0.8	-1.1	+0.1	+0.7	-1.9	-0.9	-1.2	-2.2
	`NV-Embed-v2`	+0.0	-0.6	+0.7	+-0.0	-0.5	-2.4	+0.2	-0.3	-0.1	-0.2	-1.0	-0.2	-0.9	-1.2
Reranking Recovery	`BM25`	+10.9	+10.7	+22.9	+20.0	+6.1	+7.2	+12.7	+18.5	+9.8	+9.3	+14.4	+15.1	+14.0	+13.1
	`all-distilroberta-v1`	+20.0	+30.4	+26.5	+24.7	+13.4	+11.8	+21.4	+16.4	+15.8	+14.0	+20.1	+25.9	+20.9	+17.5
	`all-MiniLM-L12-v2`	+18.7	+19.4	+19.3	+25.2	+11.9	+7.0	+22.9	+19.3	+13.6	+14.4	+21.8	+23.1	+18.4	+19.6
	`all-mpnet-base-v2`	+18.7	+19.4	+20.6	+21.1	+12.3	+13.5	+21.0	+25.4	+15.5	+14.9	+21.2	+22.4	+21.3	+17.4
	`all-mpnet-base-v2-ft`	+5.9	+6.6	+10.0	+16.5	+0.9	+5.3	+9.4	+11.3	+6.4	+6.7	+9.2	+8.8	+14.3	+10.2
	`sentence-t5-base`	+14.9	+22.7	+14.9	+18.7	+22.6	+16.0	+15.4	+20.5	+10.1	+10.4	+14.3	+19.9	+18.3	+13.5
	`sentence-t5-large`	+13.6	+19.9	+15.2	+18.6	+23.6	+16.8	+15.5	+20.6	+13.8	+12.9	+15.5	+16.4	+14.2	+13.4
	`sentence-t5-large-ft`	+5.3	+14.1	+6.1	+7.7	+8.1	+6.6	+7.7	+8.4	+3.4	+4.0	+3.3	+6.3	+9.1	+4.4
	`instructor-base`	+5.5	+12.3	+6.1	+6.2	+0.9	+3.1	+7.0	+14.2	+4.2	+5.0	+5.6	+8.5	+8.9	+6.3
	`instructor-large`	+2.2	+5.8	+1.0	+2.2	-1.1	+0.5	+1.8	+3.6	+0.2	+2.3	+1.2	+3.4	+4.9	+3.9
	`SFR-Embedding-Mistral`	+0.7	+1.9	-1.2	-0.5	-4.3	-1.8	+2.0	+2.1	-0.3	-0.7	+1.2	+0.0	+1.4	+3.3
	`NV-Embed-v2`	-2.2	-0.7	-4.2	-2.3	-8.2	-3.1	-2.1	-2.5	-2.8	-1.7	-1.8	-2.9	-1.8	-0.3
Overall Pipeline	`BM25`	-0.2	-0.4	-3.5	-5.9	-3.1	-5.5	-1.2	-3.2	-1.1	-1.0	-5.9	-2.6	-2.3	+0.8
	`all-distilroberta-v1`	-0.8	-25.5	-9.4	-9.2	-5.2	-5.8	-2.5	-7.4	-1.1	-1.0	-4.7	-6.0	-6.1	-3.6
	`all-MiniLM-L12-v2`	-0.8	-0.1	-8.7	-7.6	-3.5	-5.3	-0.6	-2.4	-1.2	+1.0	-4.4	-6.1	-6.5	+0.6
	`all-mpnet-base-v2`	-0.6	+0.1	-5.9	-4.7	-6.7	-7.5	-0.3	-1.3	-1.9	-1.7	-1.8	-4.2	-6.3	-2.4
	`all-mpnet-base-v2-ft`	-0.6	+0.1	-3.8	-1.7	-6.3	-5.9	+1.0	-2.2	-0.5	-0.6	-1.0	-1.4	-0.7	+0.6
	`sentence-t5-base`	-0.7	-6.2	-1.5	-1.5	-11.2	-8.9	-1.6	-0.3	-0.6	-0.7	+0.1	-2.0	-0.5	+0.2
	`sentence-t5-large`	-2.1	-4.8	-2.3	-2.0	-18.5	-12.4	-1.7	-1.5	+1.0	+0.0	+1.6	-1.9	-4.0	+0.7
	`sentence-t5-large-ft`	-0.1	-0.4	-1.4	-1.3	-9.2	-6.8	-0.2	+0.2	+1.1	+0.6	+-0.0	-1.5	+0.5	+0.7
	`instructor-base`	-0.8	-2.3	-3.3	-2.1	-7.7	-6.3	+0.2	-0.9	-0.4	-0.5	-0.8	-1.4	-0.8	-0.6
	`instructor-large`	-0.8	-0.9	-2.4	-1.8	-7.5	-8.4	-0.5	-0.8	-1.0	+0.1	-1.5	-1.5	-1.1	+0.1
	`SFR-Embedding-Mistral`	-1.0	+0.2	-2.2	-1.1	-6.4	-6.7	+0.1	+0.9	-0.9	-0.7	-1.8	-2.2	-0.9	+0.1
	`NV-Embed-v2`	-0.6	+0.4	-2.6	-1.2	-7.6	-6.3	-0.6	+0.0	-1.1	-0.1	-1.4	-2.2	-1.3	-0.3
	FactCheckTweet														
First-Stage Retrieval	`BM25`	+0.67	+0.00	-9.16	-11.91	+2.39	+5.89	-2.60	-19.17	+2.57	-2.11	-6.71	-10.22	-11.09	-2.39
	`all-distilroberta-v1`	-1.30	-23.85	-7.86	-13.42	+5.04	+10.50	+0.85	-11.12	+4.31	+1.39	-6.19	-8.71	-12.79	-7.15
	`all-MiniLM-L12-v2`	+0.00	+0.00	-13.57	-17.58	+2.67	+5.10	-3.62	-11.33	+2.16	-2.92	-7.14	-15.27	-14.08	-9.86
	`all-mpnet-base-v2`	+0.00	+0.00	-9.20	-11.21	-0.42	-0.77	-3.55	-3.71	-6.06	-9.19	-6.87	-8.56	-8.64	-9.11
	`all-mpnet-base-v2-ft`	+0.00	+0.00	-6.31	-8.15	+3.76	-1.28	-4.08	-12.65	+2.69	+1.61	-4.01	-5.84	-6.97	-3.85
	`sentence-t5-base`	-0.11	-8.68	-6.79	-8.32	-11.81	-11.60	+1.90	-2.13	-5.56	-4.55	-2.45	-9.27	-8.61	-4.19
	`sentence-t5-large`	-0.01	-6.23	-6.28	-7.57	-16.71	-5.94	-0.03	-11.90	-3.73	-6.12	-5.51	-6.64	-8.15	-5.30
	`sentence-t5-large-ft`	-0.98	-5.32	-6.04	-6.85	-7.17	-1.05	-0.19	-11.31	-3.34	-2.34	-3.50	-4.12	-6.49	-4.16
	`instructor-base`	-0.74	-8.27	-4.69	-8.27	+2.84	+0.18	-1.76	-10.37	-2.45	-4.40	-2.50	-4.66	-7.32	-6.03
	`instructor-large`	-0.40	-3.40	-4.61	-5.68	-1.09	-4.71	-2.33	-11.84	-0.02	-2.75	-2.64	-2.22	-3.25	-5.73
	`SFR-Embedding-Mistral`	-0.87	-0.40	+0.26	-4.22	+0.91	+2.59	-1.53	-9.36	-0.25	-0.80	-3.13	+0.15	-1.74	+1.44
	`NV-Embed-v2`	-1.34	-1.08	-1.39	-3.00	+0.12	-1.19	-1.34	-12.95	-2.08	-2.56	-3.19	-3.23	-3.36	-5.09
Reranking Recovery	`BM25`	+4.90	+5.81	+12.48	+11.32	+5.62	+6.08	+12.02	+13.51	+9.73	+12.17	+11.87	+15.82	+17.19	+8.41
	`all-distilroberta-v1`	+11.96	+16.04	+19.17	+14.79	+14.00	+15.78	+13.32	+12.99	+14.50	+15.37	+14.73	+22.34	+21.46	+16.21
	`all-MiniLM-L12-v2`	+12.00	+12.88	+19.30	+17.98	+14.57	+11.65	+17.59	+13.48	+13.05	+19.75	+19.98	+28.30	+18.27	+20.96
	`all-mpnet-base-v2`	+11.86	+12.41	+15.40	+18.35	+17.12	+17.39	+19.57	+15.43	+16.95	+22.70	+19.22	+18.41	+16.27	+17.29
	`all-mpnet-base-v2-ft`	+10.91	+11.49	+11.98	+12.14	+9.36	+14.69	+17.68	+13.66	+10.65	+13.94	+13.11	+15.25	+17.50	+13.42
	`sentence-t5-base`	+13.31	+17.32	+18.16	+16.99	+18.20	+22.56	+11.86	+12.26	+19.35	+17.23	+15.88	+23.02	+20.56	+19.53
	`sentence-t5-large`	+14.10	+18.15	+20.92	+15.65	+23.16	+19.55	+16.20	+18.94	+18.41	+20.15	+19.15	+19.33	+18.91	+19.23
	`sentence-t5-large-ft`	+10.16	+11.92	+14.91	+13.07	+13.44	+14.25	+15.44	+12.97	+14.50	+12.54	+13.88	+16.42	+18.49	+11.77
	`instructor-base`	+9.86	+12.17	+11.41	+9.51	+6.93	+10.69	+15.38	+12.80	+13.08	+16.61	+10.08	+13.45	+16.43	+11.59
	`instructor-large`	+7.98	+8.41	+14.71	+10.25	+10.92	+13.66	+15.38	+15.92	+10.72	+14.11	+9.49	+11.92	+12.40	+15.40
	`SFR-Embedding-Mistral`	+6.68	+6.64	+9.04	+8.67	+7.51	+8.35	+12.32	+12.00	+11.69	+12.12	+13.80	+14.58	+14.28	+9.24
	`NV-Embed-v2`	+2.30	+2.40	+4.16	+0.54	+2.50	+5.72	+6.72	+4.04	+6.59	+7.03	+2.09	+4.29	+3.85	+4.12
Overall Pipeline	`BM25`	+0.11	+0.35	-4.04	-7.95	+1.55	+3.31	-0.28	-19.10	+4.56	+2.26	-1.83	-1.78	-0.89	-0.98
	`all-distilroberta-v1`	-1.85	-20.33	-6.93	-16.88	+4.87	+9.32	-3.71	-18.23	+1.09	-1.22	-7.62	-2.91	-7.48	-7.09
	`all-MiniLM-L12-v2`	-0.25	+0.63	-8.95	-14.28	-0.29	+3.37	-4.88	-19.25	+1.22	+2.33	-1.14	-1.30	-9.80	-2.88
	`all-mpnet-base-v2`	+0.11	+0.66	-7.50	-6.55	+4.23	+5.91	-3.93	-10.48	-2.50	-0.12	+0.21	-2.51	-4.50	-3.96
	`all-mpnet-base-v2-ft`	-0.18	+0.40	-5.91	-7.59	+0.86	+4.59	-3.93	-9.96	+1.44	+3.55	-1.84	-1.44	-0.41	-1.37
	`sentence-t5-base`	-0.29	-4.84	-0.44	-3.13	-5.75	+1.10	-0.80	-3.90	+4.02	+2.84	+4.06	+4.19	+2.57	+5.97
	`sentence-t5-large`	-0.50	-2.67	-0.57	-7.13	-11.83	-1.27	-3.37	-6.13	-1.06	-1.96	-1.73	-2.98	-4.61	-1.44
	`sentence-t5-large-ft`	+0.35	-2.24	-2.04	-4.69	-5.24	+4.73	+0.81	-11.86	+0.55	-0.56	+1.23	+2.84	+2.85	-1.55
	`instructor-base`	+0.77	-4.45	-3.17	-8.65	-0.61	+4.45	-0.98	-11.37	+2.19	+3.71	-0.81	+0.51	+0.71	-2.83
	`instructor-large`	-0.96	-3.52	-1.70	-7.23	-1.90	+1.75	-1.42	-11.71	-0.78	-0.21	-3.89	-1.14	-1.59	-1.06
	`SFR-Embedding-Mistral`	-0.21	+0.22	+0.83	-4.02	+0.12	+5.66	+1.45	-10.21	+2.73	+2.54	-0.80	+2.36	+1.08	-0.78
	`NV-Embed-v2`	+0.22	+0.58	+0.63	-4.59	+0.61	+4.48	+0.77	-11.68	+1.64	+1.58	-1.27	+0.91	+0.31	-1.14

Table 2: Effect of perturbations on ***CheckThat22*** (top) and ***FactCheckTweet*** (bottom). **First-Stage Retrieval** shows the Δ*retrieval gap* between unperturbed and perturbed inputs sets. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-50 candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@20 is used for all comparisons. Models finetuned are indicated with the postfix `ft`.

perturbed sets for a given value of k. Results for *CheckThat22* and *FactCheckTweet* with $k = 20$ are shown in Table 2. We also provide results for additional values of k, which exhibit similar trends, in Appendix §F. From these results, we make the following observations:

LLM-distilled embeddings exhibit superior robustness. Our analysis shows a smaller Δ*retrieval gap* for `SFR-Embedding-Mistral` and `NV-Embed-V2`, both based on `Mistral-7B` (Jiang et al., 2023). We hypothesize that this robustness stems from the increased representational capacity afforded by the larger scale of the backbone model from which the embeddings are derived, a characteristic shown to enhance robustness to adversarial inputs (Howe et al., 2024). This robustness also extends to dialectical variations and social-media typos, which may be due to the increasing importance of social media data in the training corpora of recent generative models (Longpre et al., 2024; Miranda et al., 2024; Penedo et al., 2024). In contrast, BERT-based and T5-based models are primarily trained on less diverse, standardised corpora, making them less robust to lexical variations.

NLI-Fine-Tuned embedding models such as Sentence-T5 demonstrate low robustness to negation. Sentence embedding models primarily trained on natural language inference (NLI) datasets, such as T5 (Raffel et al., 2020), exhibit a large Δ*retrieval gap* under negation. This may stem from treating contradiction hypotheses as hard-negatives during training with contrastive loss (Ni et al., 2022), making the models highly sensitive to negation (Chiang et al., 2023). In contrast, models trained on more diverse datasets (e.g., `all-distilroberta-v1`, `all-MiniLM-L12-v2`, `all-mpnet-base-v2`) demonstrate greater robustness to *negation* and, in some cases, even show improvements. BM25, meanwhile, remains robust to negation, as it is affected minimally by small lexical changes. Finally, *double negation* rewrites claims in a form closely aligned with fact-checks (e.g., "It is not true ..."), and may lead to retrieval improvements for some embedding models as observed on *FactCheckTweet*.

Embedding models with case-sensitive tokenizers struggle with casing variations. Embedding models such as `all-distilroberta-v1` and `T5`-based models with case-sensitive tokenizers, which distinguish between tokens like `apple` and `Apple`, show significant retrieval performance gaps on casing-perturbed claims. In contrast, embedding models with case-insensitive tokenizers exhibit robustness to casing differences, showing minimal to no retrieval gaps. While this behavior is expected for case-sensitive models, in retrieval tasks where casing differences are irrelevant, applying casing normalization or using a case-insensitive model can be an effective solution.

4.2 RQ2: Can a Strong Reranker Recover the Retrieval Gap?

In Section §4.1, we observe that perturbations introduce significant retrieval gaps in first-stage retrieval. However, it remains unclear whether a strong reranker can mitigate these gaps and improve ranking quality. To test this, we apply a reranker to the top-j retrieved candidates and measure the Δ*recovery gap*, which quantifies the MAP@k improvement on the perturbed set after reranking. We rerank the top $j = 50$ candidates, as this balances efficiency and performance, achieving the highest MAP@5 across evaluated values ($j \in \{5, 10, 20, 50, 100, 200\}$) (see §D.2). We report this gap for $k = 20$ and $j = 50$ on *CheckThat22* and *FactCheckTweet* using the reranker `bge-reranker-v2-gemma` in Table 2. Additional results for different k values are provided in §F of the Appendix. Across different values of k and evaluation datasets, we observe that:

Reranking improves weaker embedding models but can negatively impact stronger models in short-target tasks. Across both datasets, we observe that reranking is largely beneficial for embedding models that are less robust in first-stage retrieval. However, in *CheckThat22*, which contains shorter targets (i.e., short fact-check statements), reranking reduces performance for stronger embedding models, as shown by a drop in MAP@20 scores. This decline may stem from noise i.e., inclusion of additional, potentially irrelevant or low-quality candidates introduced when reranking a larger set of top-50 candidates, which can distort the high-quality rankings produced by stronger models. For stronger embedding models, the most relevant candidates are often already ranked highly and expanding the candidate set can introduce noise. When using stronger embedding models in first-stage retrieval, reranking a smaller candidate subset or skipping reranking altogether might mitigate this issue. For *FactCheckTweet*, the reranker processes the full fact-check article due to its longer context window, unlike the paragraph-based splitting used during retrieval. This allows the reranker to leverage the full fact-check's information, which we hypothesize contributes to its more pronounced reranking gains, even for stronger models, compared to *CheckThat22*.

4.3 RQ3: To what extent do perturbations impact the overall pipeline's performance?

We observe that a strong reranker is helpful in recovering the retrieval gap discussed in Section §4.2. However, whether it fully compensates for this gap remains to be determined. Here, we examine the end-to-end pipeline (i.e., retrieval plus reranking), by comparing performance on perturbed vs. unperturbed sets. This analysis reveals the Δ*overall gap*, which is the MAP@k difference between the unperturbed and perturbed sets for the full pipeline of retrieval and reranking the *top-j* candidates. We show the results in Table 2 for $k = 20, j = 50$, and provide additional results for other k values in §F of the Appendix. We observe that:

A strong reranker helps, but cannot fully compensate for first-stage retrieval gaps. A strong reranker proves effective in reducing the retrieval gap particularly for less robust embedding models. However, a notable end-to-end performance gap persists, indicating that a strong reranker alone is not sufficient. Although we select a SoTA reranker, we hypothesize that it may also exhibit brittleness to some of the perturbations, which could reduce its effectiveness in reranking and contribute to the observed performance drop. For instance, under the *negation* perturbation on *CheckThat22*, the reranker negatively impacts the overall performance of embedding models that initially performed well in first-stage rankings. Additionally, weaker embedding models display significant deficiencies in handling *typos* and *dialect* perturbations. Similar limitations are observed with case-sensitive embedding models, where reranking fails to fully recover from retrieval gaps caused by *casing* perturbations. For *entity replacements*, reranking helps reduce the retrieval gap on *CheckThat22*, but offers little benefit on *FactCheckTweet*. We hypothesize that entity swaps have a smaller impact when matching against shorter targets, as in *CheckThat22*, compared to longer targets, where the effects of entity changes are more severe.

***LLM Rewrites* improves overall performance** as it acts a form of query rewriting or naïve query expansion, a technique shown to improve performance in retrieval tasks (Nogueira et al., 2019b; Kazemi et al., 2023; Weller et al., 2024).

4.4 RQ4: How effective are mitigation approaches at improving robustness for weaker embedding models, and what trade-offs do they entail?

LLM-distilled embeddings are more robust to misinformation edits (Section §4.1), but they come with the trade-off of increased storage requirements and slower inference, making them less ideal for first-stage retrieval. In contrast, smaller models are computationally efficient but exhibit weaker robustness. In this section, we explore whether train-time and inference-time approaches can improve the robustness of weaker embedding models on *typos*, *negation*, *dialect changes*, and *entity replacement*, perturbations that remain challenging even for a strong reranker. We test the approaches on the *CheckThat22* dataset, which we refer to as the in-domain dataset, as we use its training and development sets. We also evaluate how well these improvements generalize to an out-of-domain (OOD) dataset consisting of novel claims not seen during training on *CheckThat22*. All mitigations are applied only to the first-stage retrieval, and we report retrieval performance for this stage.[6]

Model	*Typos* U	*Typos* P	*Neg.* U	*Neg.* P	*Entity R.* U	*Entity R.* P	*Dialect* U	*Dialect* P
`mpnet (baseline)`	0.76	0.68	0.78	0.74	0.76	0.66	0.74	0.65
`mpnet-robust`	0.83	0.73	0.84	0.81	0.83	0.77	0.81	0.80
`mpnet-ft`	0.90	0.77	0.90	0.85	0.89	0.80	0.88	0.80
`mpnet-robust-ft`	0.89	0.78	0.88	0.84	0.90	0.81	0.87	0.84
`mpnet+CN`	0.77	0.78	0.80	0.79	0.76	0.75	0.74	0.73
`mpnet-robust+CN`	0.83	0.83	0.89	0.82	0.84	0.84	0.82	0.82
`mpnet-ft+CN`	0.89	**0.89**	0.85	**0.87**	0.89	0.87	0.88	0.86
`mpnet-robust-ft+CN`	0.90	0.88	0.90	0.85	0.89	**0.89**	0.89	**0.87**
`NV-Embed-v2`	0.98	0.98	0.97	0.94	0.97	0.97	0.97	0.96
`NV-Embed-v2 + CN`	0.98	0.97	0.97	0.93	0.97	0.97	0.97	0.97

Table 3: **Effect of mitigation approaches** on *typos*, *negation*, *entity replacement* and dialect (*Pidgin*) on ***CheckThat22* (In-Domain)**. `mpnet` refers to `all-mpnet-base-v2`, with suffixes denoting mitigation approaches: `robust` (KD approach), `ft` (task finetuning), and `+CN` (claim normalization). Columns marked `U` and `P` indicate MAP@20 performance on the unperturbed and perturbed sets, respectively.

For train-time interventions, we examine **finetuning** on the *CheckThat22* training set and a **knowledge distillation (KD)** approach inspired by Reimers and Gurevych (2020), which aims to reduce representation differences between unperturbed and perturbed input claims. For KD, we generate parallel sentence pairs using our perturbation generation framework (§3.2.1), applied to the *CheckThat22* training set, yielding 11,593 unperturbed–perturbed claim pairs. We further expand this set by pairing different perturbations of the same claim (e.g., typo and dialect), resulting in 70,954 claim pairs. Details of the training setup are provided in §C.1.2. As an inference-time intervention, we investigate a **claim normalization** approach proposed by Sundriyal et al. (2023), where input claims are rewritten into a standard form using `GPT4o` prior to retrieval.[7] Finally, we evaluate the effect of combining multiple approaches. We select `all-mpnet-base-v2` as a representative weaker—yet computationally efficient—embedding model, as it is widely used for retrieval

[6]We assume that improving the first-stage retrieval will subsequently improve reranking performance.

[7]The prompt used is shown in §I.

Model	MAP@1	MAP@5	MAP@20	MAP@50
`mpnet (baseline)`	0.4327	0.5129	0.5254	0.5269
`mpnet-robust`	0.4508	0.5259	0.5390	0.5411
`mpnet-ft`	0.5076	0.5853	0.5982	0.5998
`mpnet-robust-ft`	0.5090	0.5851	0.5991	0.6006
`mpnet+CN`	0.4938	0.5696	0.5799	0.5816
`mpnet-robust+CN`	0.5076	0.5765	0.5888	0.5911
`mpnet-ft+CN`	**0.5368**†	**0.6109**†	**0.6213**†	**0.6233**†
`mpnet-robust-ft+CN`	0.5257	0.6046	0.6163	0.6185
`NV-Embed-v2`	0.5520	0.6306	0.6402	0.6418
`NV-Embed-v2+CN`	**0.5576***	**0.6375***	**0.6464***	**0.6479***

Table 4: **Effect of Mitigation Approaches on an Out-of-Domain (OOD) Dataset**. `mpnet` denotes `all-mpnet-base-v2`, with `-ft` for task finetuning, `-robust` for the KD approach, and `+CN` indicating claim normalization. † indicates the model with the best improvement over the baseline, while * indicates the model with the best overall performance.

tasks.[8] Results are shown in Table 3 and Table 4 for the test split of *CheckThat22* and the *OOD Dataset*, respectively. We observe that:

Finetuning, KD and claim normalization all improve robustness for weaker embedding models and can be applied independently or jointly. KD and finetuning are *train-time* interventions that update model weights to better handle differences between unperturbed and perturbed input claims. KD relies on synthetically generated perturbed input claims that are scalable and inexpensive to produce (Shliselberg et al., 2024; Liu et al., 2024). In contrast, finetuning requires *claim-factcheck* pairs, which are costly to annotate. On the other hand, claim normalization is an *inference-time* intervention that leverages the innate representational capabilities of weaker embedding models by standardizing perturbed input claims into a form that is easier to represent, without having to update model weights. Although claim normalization is appealing for its "train-free," inference-time application, its reliance on a strong normalizer (often an LLM) may add a computational overhead to the overall retrieval pipeline, making robust adaptation through knowledge distillation a scalable alternative for scenarios where train-time interventions are feasible.

We expect a significant drop in retrieval performance across all approaches for novel claims in the OOD dataset as they are (a) taken from a different time period, (b) cover different topics, and (c) come from a different platform, with different characteristics than the tweets in *CheckThat22* and *FactCheckTweet* (e.g., longer length). We observe such a drop. Nonetheless, **all mitigation approaches improve performance relative to the baseline and combining them yields the largest improvement of 10 percentage points** similar to the in-domain setting. This highlights the effectiveness of the mitigation approaches in generalizing to novel out-of-domain claims.

5 Discussion and Conclusion

Real-world claim matching systems must handle not only factual but also edited claims, which often arise as users engage with misinformation online. Embedding models used in these systems must therefore be robust to such user-informed misinformation edits. However, standard benchmarks like MTEB (Muennighoff et al., 2022; Enevoldsen et al., 2025), primarily evaluate sentence embedding models on clean, generalist tasks and fail to capture this challenge. To address this gap, we introduce a perturbation generation framework that produces natural and valid claim edits. This framework creates a more realistic evaluation test bed for sentence embedding models. Although developed for claim matching, these perturbations have broader relevance and can help select embedding models for other downstream tasks that involve user-generated inputs. Our evaluation shows that widely-used embedding models are highly sensitive to user edits, while LLM-distilled embeddings offer greater robustness. Reranking helps, but even strong rerankers cannot fully recover performance under perturbation. To address the retrieval gaps, we show that train- and inference-time approaches can improve embedding model robustness and generalize to out-of-domain settings with different trade-offs. Knowledge distillation and finetuning are train-time interventions that require generating synthetic data or collecting costly human annotated data with additional computational costs for training and re-embedding fact-checks. On the other hand, claim normalization is a "train-free" inference-time mitigation that depends on an LLM and may introduce significant latency and cost to the overall pipeline. Encouragingly, prior work (Ma et al., 2023) shows that rewriting capabilities can be distilled into smaller models, offering a practical compromise. Overall, our study offers practical insights for improving claim matching systems, ultimately empowering human fact-checkers to retrieve relevant fact-checks with greater reliability.

[8] `all-mpnet-base-v2` had 33.4M downloads in the past month at the time of this study.

Limitations

While our study makes meaningful strides in evaluating the robustness of embedding models to real-world user edits and exploring mitigation strategies, we acknowledge some limitations:

Dataset Coverage Our evaluation relies on claim matching datasets from fact-checking organizations. Although these datasets provide a useful starting point, they represent only a fraction of the broader misinformation landscape, as many claims in the wild go unchecked. As such, the taxonomy of common edits proposed in this work is representative but not exhaustive, as it is limited to claims encountered by fact-checkers. This limitation is evident in our out-of-domain evaluations, where models experience notable performance degradation when faced with novel claims spanning different timelines, topics, and platforms. Nonetheless, our proposed interventions generalize well to the out-of-domain setting, improving the performance of baseline models by an average of 10 percentage points.

Language Scope Our analysis is limited to English-language claims, even though misinformation is a multilingual phenomenon (Kazemi et al., 2021; Quelle et al., 2025). We believe that our approach could be extended to other languages, and pursuing this extension is an important direction for future work.

Experimental Coverage Although our experiments cover a diverse range of settings, evaluating 12 embedding models, 7 rerankers, and 14 perturbation types across various MAP@k values ($k \in \{5, 10, 20, 50\}$) and different top-j reranking candidates ($j \in \{5, 10, 20, 50, 100, 200\}$)—this selection is not exhaustive. Alternative configurations may yield different robustness outcomes. We evaluate widely used embedding models that represent various pre-training approaches, backbone architectures and embedding dimension sizes. However, we note that other types of embedding models, such as unsupervised embeddings (Gao et al., 2021) or contextual document embeddings (Morris and Rush, 2025), might behave differently. We also do not evaluate closed-source embedding models (e.g., `text-embedding-ada-002` from OpenAI[9] or `Embed v3` from Cohere[10]) due to the inference costs involved. While we test different rerankers and select the best-performing one, we acknowledge that the reranker we select could also exhibit some brittleness to the perturbations, and we believe that evaluating rerankers against the same perturbation set would be a valuable direction for future work.

[9] https://platform.openai.com/docs/guides/embeddings

[10] https://cohere.com/blog/introducing-embed-v3

Ethical Considerations

We acknowledge the ethical responsibilities inherent in misinformation research given its real-world impact. Our perturbation generation framework relies on widely available LLMs, and our prompts are publicly released for reproducibility; however, this introduces the risk of misuse, whether inadvertently by non-experts or deliberately by malicious actors to generate misinformation or create rewritings that real-world claim matching systems struggle to identify. For a more detailed discussion on the ethical risks associated with LLM-generated misinformation, see Chen and Shu (2024). We also note that the synthetically perturbed claims generated by our work may contain fictional or inaccurate information; as such, these claims should not be used to train models that learn facts from data. Notably, during our claim normalization experiments, `GPT-4o` refused to process two claims containing sensitive political or religious statements. This incident highlights the critical need for automated systems to differentiate between the use of misinformation (deliberately propagating false information) and the mention of misinformation (discussing or referencing sensitive topics without endorsing them) (Gligoric et al., 2024).

Acknowledgments

We thank Meedan for providing the datasets used in this project. JM is supported by the Rhodes Scholarship. SH is supported by a grant from The Alan Turing Institute for the project entitled "Effective discovery, tracking, and response to mis- and disinformation" and the ESRC Digital Good Network (grant reference ES/X502352/1). We are grateful to Samuel Mensah and Elena Kochkina for their helpful feedback on drafts of this paper.

References

Timothy Baldwin, Marie-Catherine De Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings* *of the workshop on noisy user-generated text*, pages 126–135.

Alberto Barrón-Cedeno, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, Fatima Haouari, Nikolay Babulkov, Bayan Hamdan, Alex Nikolov, et al. 2020. Overview of checkthat! 2020: Automatic identification and verification of claims in social media. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11*, pages 215–236. Springer.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Canyu Chen and Kai Shu. 2024. Can LLM-generated misinformation be detected? In *The Twelfth International Conference on Learning Representations*.

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

Cheng-Han Chiang and Hung-yi Lee. 2023. Are synonym substitution attacks really synonym substitution attacks? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1853–1878, Toronto, Canada. Association for Computational Linguistics.

Cheng-Han Chiang, Hung-yi Lee, Yung-Sung Chuang, and James Glass. 2023. Revealing the blind spot of sentence encoder evaluation by HEROS. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 289–302, Toronto, Canada. Association for Computational Linguistics.

Anubrata Das, Houjiang Liu, Venelin Kovatchev, and Matthew Lease. 2023. The state of human-centered nlp technology for fact-checking. *Information processing & management*, 60(2):103219.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Salijona Dyrmishi, Salah Ghamizi, and Maxime Cordy. 2023a. How do humans perceive adversarial text? a reality check on the validity and naturalness of word-based adversarial attacks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8822–8836, Toronto, Canada. Association for Computational Linguistics.

Salijona Dyrmishi, Salah Ghamizi, and Maxime Cordy. 2023b. How do humans perceive adversarial text? A reality check on the validity and naturalness of word-based adversarial attacks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8822–8836, Toronto, Canada. Association for Computational Linguistics.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Veysel Çağatan, Akash Kundu, Martin Bernstorff, Shitao Xiao, Akshita Sukhlecha, Bhavish Pahwa, Rafał Poświata, Kranthi Kiran GV, Shawon Ashraf, Daniel Auras, Björn Plüster, Jan Philipp Harries, Loïc Magne, Isabelle Mohr, Dawei Zhu, Hippolyte Gisserot-Boukhlef, Tom Aarsen, Jan Kostkan, Konrad Wojtasik, Taemin Lee, Marek Suppa, Crystina Zhang, Roberta Rocca, Mohammed Hamdy, Andrianos Michail, John Yang, Manuel Faysse, Aleksei Vatolin, Nandan Thakur, Manan Dey, Dipam Vasani, Pranjal A Chitale, Simone Tedeschi, Nguyen Tai, Artem Snegirev, Mariya Hendriksen, Michael Günther, Mengzhou Xia, Weijia Shi, Xing Han Lù, Jordan Clive, Gayatri K, Maksimova Anna, Silvan Wehrli, Maria Tikhonova, Henil Shalin Panchal, Aleksandr Abramov, Malte Ostendorff, Zheng Liu, Simon Clematide, Lester James Validad Miranda, Alena Fenogenova, Guangyu Song, Ruqiya Bin Safi, Wen-Ding Li, Alessia Borghini, Federico Cassano, Lasse Hansen, Sara Hooker, Chenghao Xiao, Vaibhav Adlakha, Orion Weller, Siva Reddy, and Niklas Muennighoff. 2025. MMTEB: Massive multilingual text embedding benchmark. In *The Thirteenth International Conference on Learning Representations*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Lorenzo Jaime Yu Flores and Yiding Hao. 2022. An adversarial benchmark for fake news detection models. *arXiv preprint arXiv:2201.00912*.

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

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Vahid Ghafouri, Jose Such, and Guillermo Suarez-Tangil. 2024. I love pineapple on pizza != I hate pineapple on pizza: Stance-aware sentence transformers for opinion mining. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21046–21058, Miami, Florida, USA. Association for Computational Linguistics.

Kristina Gligoric, Myra Cheng, Lucia Zheng, Esin Durmus, and Dan Jurafsky. 2024. NLP systems that can‘t tell use from mention censor counterspeech, but teaching the distinction helps. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5942–5959, Mexico City, Mexico. Association for Computational Linguistics.

Max Glockner, Yufang Hou, and Iryna Gurevych. 2022. Missing counter-evidence renders NLP fact-checking unrealistic for misinformation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5916–5936, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Shreya Goyal, Sumanth Doddapaneni, Mitesh M Khapra, and Balaraman Ravindran. 2023. A survey of adversarial defenses and robustness in nlp. *ACM Computing Surveys*, 55(14s):1–39.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2101–2112, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Christopher Hidey, Tuhin Chakrabarty, Tariq Alhindi, Siddharth Varia, Kriste Krstovski, Mona Diab, and Smaranda Muresan. 2020. DeSePtion: Dual sequence prediction and adversarial examples for improved fact-checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8593–8606, Online. Association for Computational Linguistics.

Nikolaus Howe, Ian McKenzie, Oskar Hollinsworth, Michał Zajac, Tom Tseng, Aaron Tucker, Pierre-Luc Bacon, and Adam Gleave. 2024. Effects of scale on language model robustness. *Preprint*, arXiv:2407.18213.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

David Jurgens, Yulia Tsvetkov, and Dan Jurafsky. 2017. Incorporating dialectal variability for socially equitable language identification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 51–57, Vancouver, Canada. Association for Computational Linguistics.

A Kazemi, Z Li, V Peréz-Rosas, SA Hale, and R Mihalcea. 2022. Matching tweets with applicable fact-checks across languages. In *CEUR Workshop Proceedings*.

Ashkan Kazemi, Artem Abzaliev, Naihao Deng, Rui Hou, Scott Hale, Veronica Perez-Rosas, and Rada Mihalcea. 2023. Query rewriting for effective misinformation discovery. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 398–407, Nusa Dua, Bali. Association for Computational Linguistics.

Ashkan Kazemi, Kiran Garimella, Devin Gaffney, and Scott Hale. 2021. Claim matching beyond English to scale global fact-checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4504–4517, Online. Association for Computational Linguistics.

Thai Le, Suhang Wang, and Dongwon Lee. 2020. Malcom: Generating malicious comments to attack neural fake news detection models. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 282–291. IEEE.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Nv-embed: Improved techniques for training llms as generalist embedding models. *Preprint*, arXiv:2405.17428.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R. Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, Yi Luan, Sai Meher Karthik Duddu, Gustavo Hernandez Abrego, Weiqiang Shi, Nithi Gupta, Aditya Kusupati, Prateek Jain, Siddhartha Reddy Jonnalagadda, Ming-Wei Chang, and Iftekhar Naim. 2024. Gecko: Versatile text embeddings distilled from large language models. *Preprint*, arXiv:2403.20327.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lucian Vlad Lita, Abe Ittycheriah, Salim Roukos, and Nanda Kambhatla. 2003. Truecasing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 152–159.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, et al. 2024. Best practices and lessons learned on synthetic data for language models. *arXiv preprint arXiv:2404.07503*.

Shayne Longpre, Nikhil Singh, Manuel Cherep, Kushagra Tiwary, Joanna Materzynska, William Brannon, Robert Mahari, Manan Dey, Mohammed Hamdy, Nayan Saxena, et al. 2024. Bridging the data provenance gap across text, speech and video. *arXiv preprint arXiv:2412.17847*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Emanuele La Malfa and Marta Kwiatkowska. 2022. The King Is Naked: On the Notion of Robustness for Natural Language Processing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11047–11057. Section: AAAI Technical Track on Speech and Natural Language Processing.

Rui Meng, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. Sfr-embedding-mistral:enhance text retrieval with transfer learning. Salesforce AI Research Blog.

Brando Miranda, Alycia Lee, Sudharsan Sundar, Allison Casasola, and Sanmi Koyejo. 2024. Beyond scale: The diversity coefficient as a data quality metric for variability in natural language data. *Preprint*, arXiv:2306.13840.

John Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3829–3839, Online. Association for Computational Linguistics.

John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020b. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909*.

John Xavier Morris and Alexander M Rush. 2025. Contextual document embeddings. In *The Thirteenth International Conference on Learning Representations*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021a. Automated Fact-Checking for Assisting Human Fact-Checkers. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4551–4558. International Joint Conferences on Artificial Intelligence Organization.

Preslav Nakov, Giovanni Da San Martino, Firoj Alam, Shaden Shaar, Hamdy Mubarak, and Nikolay Babulkov. 2022. Overview of the CLEF-2022 CheckThat! lab task 2 on detecting previously fact-checked claims. In *Working Notes of CLEF 2022—Conference and Labs of the Evaluation Forum*, CLEF '2022, Bologna, Italy.

Preslav Nakov, Giovanni Da San Martino, Tamer Elsayed, Alberto Barrón-Cedeno, Rubén Míguez, Shaden Shaar, Firoj Alam, Fatima Haouari, Maram Hasanain, Nikolay Babulkov, et al. 2021b. The clef-2021 checkthat! lab on detecting check-worthy claims, previously fact-checked claims, and fake news. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*, pages 639–649. Springer.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Lydia Nishimwe, Benoît Sagot, and Rachel Bawden. 2024. Making sentence embeddings robust to user-generated content. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 10984–10998, Torino, Italia. ELRA and ICCL.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. corr abs/1901.04085 (2019). *URL: http://arxiv. org/abs/1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online. Association for Computational Linguistics.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019a. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. RankZephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv:2312.02724*.

Rob Procter, Miguel Arana-Catania, Yulan He, Maria Liakata, Arkaitz Zubiaga, Elena Kochkina, and Runcong Zhao. 2023. Some observations on fact-checking work with implications for computational support. *arXiv preprint arXiv:2305.02224*.

Piotr Przybyła, Alexander Shvets, and Horacio Saggion. 2024. Verifying the robustness of automatic credibility assessment. *Natural Language Processing*, page 1–29.

Dorian Quelle, Calvin Yixiang Cheng, Alexandre Bovet, and Scott A. Hale. 2025. Lost in translation: Using global fact-checks to measure multilingual misinformation prevalence, spread, and evolution. *EPJ Data Science*, 14(1).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Javad Rafiei Asl, Prajwal Panzade, Eduardo Blanco, Daniel Takabi, and Zhipeng Cai. 2024. RobustSentEmbed: Robust sentence embeddings using adversarial self-supervised contrastive learning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3795–3809, Mexico City, Mexico. Association for Computational Linguistics.

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

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1524–1534.

Manuela Sanguinetti, Bosco Cristina, Cassidy Lauren, Cetinoglu Ozlem, Cignarella Alessandra Teresa, Lynn Teresa, Rehbein Ines, Ruppenhofer Joseph, Seddah Djamé, Zeldes Amir, et al. 2020. Treebanking user-generated content: A proposal for a unified representation in universal dependencies. In *Proceedings of the 12th language resources and evaluation conference*, pages 5240–5250. ELRA, Language Resources Association.

Shaden Shaar, Nikolay Babulkov, Giovanni Da San Martino, and Preslav Nakov. 2020. That is a known lie: Detecting previously fact-checked claims. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3607–3618, Online. Association for Computational Linguistics.

Shaden Shaar, Nikola Georgiev, Firoj Alam, Giovanni Da San Martino, Aisha Mohamed, and Preslav Nakov. 2022. Assisting the human fact-checkers: Detecting all previously fact-checked claims in a document. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2069–2080, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Qiang Sheng, Juan Cao, Xueyao Zhang, Xirong Li, and Lei Zhong. 2021. Article reranking by memory-enhanced key sentence matching for detecting previously fact-checked claims. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5468–5481, Online. Association for Computational Linguistics.

Michael Shliselberg and Shiri Dori-Hacohen. 2022. Riet lab at checkthat!-2022: Improving decoder based re-ranking for claim matching. In *CLEF (Working Notes)*, pages 671–678.

Michael Shliselberg, Ashkan Kazemi, Scott A Hale, and Shiri Dori-Hacohen. 2024. SynDy: Synthetic dynamic dataset generation framework for misinformation tasks. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2801–2805.

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

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.

Megha Sundriyal, Tanmoy Chakraborty, and Preslav Nakov. 2023. From chaos to clarity: Claim normalization to empower fact-checking. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6594–6609, Singapore. Association for Computational Linguistics.

Manveer Singh Tamber, Ronak Pradeep, and Jimmy Lin. 2023. Scaling down, litting up: Efficient zero-shot listwise reranking with seq2seq encoder-decoder models. *arXiv preprint arXiv: 2312.16098*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Evaluating adversarial attacks against multiple fact verification systems. 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 2944–2953, Hong Kong, China. Association for Computational Linguistics.

Manuel Tonneau, Diyi Liu, Samuel Fraiberger, Ralph Schroeder, Scott Hale, and Paul Röttger. 2024a. From languages to geographies: Towards evaluating cultural bias in hate speech datasets. In *Proceedings of the 8th Workshop on Online Abuse and Harms (WOAH 2024)*, pages 283–311, Mexico City, Mexico. Association for Computational Linguistics.

Manuel Tonneau, Pedro Quinta De Castro, Karim Lasri, Ibrahim Farouq, Lakshmi Subramanian, Victor Orozco-Olvera, and Samuel Fraiberger. 2024b. NaijaHate: Evaluating hate speech detection on Nigerian Twitter using representative data. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9020–9040, Bangkok, Thailand. Association for Computational Linguistics.

Rob Van Der Goot, Rik van Noord, and Gertjan van Noord. 2018. A taxonomy for in-depth evaluation of normalization for user generated content. In *Eleventh International Conference on Language Resources and Evaluation*, pages 684–688. European Language Resources Association (ELRA).

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *science*, 359(6380):1146–1151.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2022. Measure and improve robustness in NLP models: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4569–4586, Seattle, United States. Association for Computational Linguistics.

WEF. 2025. Global risks report 2025: Conflict, environment and disinformation top threats. Accessed: 2025-02-07.

Orion Weller, Kyle Lo, David Wadden, Dawn Lawrie, Benjamin Van Durme, Arman Cohan, and Luca Soldaini. 2024. When do generative query and document expansions fail? a comprehensive study across methods, retrievers, and datasets. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1987–2003, St. Julian's, Malta. Association for Computational Linguistics.

Muheng Yan, Yu-Ru Lin, and Wen-Ting Chung. 2022. Are mutated misinformation more contagious? A case study of Covid-19 misinformation on Twitter. In *Proceedings of the 14th ACM Web Science Conference 2022*, pages 336–347.

Zhixuan Zhou, Huankang Guan, Meghana Moorthy Bhat, and Justin Hsu. 2019. Fake news detection via nlp is vulnerable to adversarial attacks. *arXiv preprint arXiv:1901.09657*.

Caleb Ziems, William Held, Jingfeng Yang, Jwala Dhamala, Rahul Gupta, and Diyi Yang. 2023. Multi-value: A framework for cross-dialectal english nlp. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Association for Computational Linguistics.

A Dataset Statistics

We report the details of each dataset used in our experiments in this section. The *CheckThat22* dataset (Nakov et al., 2021b) is available under a permissible research-use license.[11] Similarly, the *FactCheckTweet* dataset (Kazemi et al., 2022) can be accessed for research purposes.[12] The *OOD Dataset* is provided by Meedan, primarily sourced through five fact-checking organizations operating tiplines on WhatsApp, and our use is consistent with its intended use.

A.1 Dataset Splits

For all experiments, we use only the English subsets of the datasets. In the robustness experiments, we apply perturbations to the test set only. *CheckThat22* provides a predefined train/dev/test split, whereas *FactCheckTweet* does not. We therefore partition the latter into 80% training, 10% development, and 10% test splits. The train and development splits from *CheckThat22* are used for finetuning the embedding models or improving robustness through the teacher-student knowledge distillation approach. For the *OOD Dataset*, we treat the full split as an evaluation set and do not partition it. Table 5 shows the splits for each dataset.

Dataset	Train	Dev	Test
CheckThat22	1,195	1,195	202
FactCheckTweet	2,504	537	537
OOD Dataset	-	-	721

Table 5: Dataset splits for training, development, and testing.

A.2 Perturbed Claims Statistics

We report the number of valid perturbed input claims for each perturbation type in Table 6. All perturbations are applied using the framework detailed in Section 3.2.1, except for Casing, which we apply using the `TrueCase` Python library.[13] While perturbations are initially applied to the full test sets, the number of valid claims varies due to differences in the selection criteria introduced by the LLM as a verifier. **When comparing against unperturbed claims, we use only the same subset and not the entire original test set.** Overall, perturbations applied at the maximum noise level (b_{high}) result in more invalid perturbations, which are filtered out in our generation pipeline.

Perturbation	Type	*CheckThat22*	*FactCheckTweet*
Casing	*TrueCase*	202	537
	UpperCase	202	537
Typos	*Least*	192	443
	Most	189	441
Negation	*Shallow*	181	405
	Double	148	308
Entity Rep.	*At least 1*	181	327
	All	110	164
LLM Rewrite	*Least*	185	373
	Most	185	371
Dialect	*AAE*	195	449
	Jamaican	192	443
	Pidgin	192	449
	Singlish	195	449

Table 6: Valid perturbed input claims for each perturbation type across *CheckThat22* and *FactCheckTweet* datasets.

We report the degree of lexical overlap between the unperturbed and perturbed input claims for the *CheckThat22* and *FactCheckTweet* datasets in Table 7 and Table 8, respectively. Following the approach of Chiang et al. (2023), we compute the ROUGE F1 scores (**R1**, **R2**, **RL**) (Lin, 2004), where R1 measures unigrams, R2 measures bigrams, and RL measures the longest common subsequence. We use the `rouge`[14] package for this purpose. A higher ROUGE score indicates greater lexical similarity. We also report the normalized Levenshtein distance between the unperturbed and perturbed input pairs, obtained by calculating the Levenshtein distance between the pair and dividing by the length of the longer sequence. We tokenize the sentences using `bert-base-uncased` and calculate the distance between the token IDs. A lower Levenshtein distance implies higher lexical overlap. Across all perturbation types, we observe that the worst-case edits—those applied with the maximum noise level (b_{high})—result in perturbed claims that have less lexical overlap with their unperturbed counterparts than do the edits applied with the minimum noise level (b_{low}). This difference in lexical overlap enables us to capture a wide spectrum of user edits.

[11] https://gitlab.com/checkthat_lab/clef2022-checkthat-lab/clef2022-checkthat-lab.git

[12] https://lit.eecs.umich.edu/publications.html

[13] https://github.com/daltonfury42/truecase

[14] https://pypi.org/project/rouge/

Perturbation	Type	R1	R2	RL	Lev.
Casing	*TrueCase*	90.20	83.02	90.20	0.00
	UpperCase	0.00	0.00	0.00	0.00
Typos	*Least*	60.47	31.46	60.47	0.48
	Most	41.67	10.81	38.89	0.68
Negation	*Shallow*	56.34	46.58	53.52	0.65
	Double	17.78	12.50	17.78	0.90
Entity Rep.	*Atleast 1*	98.00	94.23	98.00	0.05
	All	87.50	74.75	87.50	0.14
LLM Rewrite	*Least*	58.14	30.77	53.49	0.52
	Most	41.46	16.47	41.46	0.66
Dialect	*AAE*	39.17	17.19	37.97	0.63
	Jamaican	34.33	12.98	32.98	0.68
	Pidgin	34.49	13.32	33.00	0.68
	Singlish	40.28	17.81	38.51	0.65

Table 7: Degree of lexical overlap between unperturbed and perturbed input claims for the ***CheckThat22*** dataset. The table reports ROUGE F1 scores (R1, R2, RL), and the normalized Levenshtein distance (Lev.). Higher ROUGE scores and lower Levenshtein distances indicate higher lexical overlap.

Perturbation	Type	R1	R2	RL	Lev.
Casing	*TrueCase*	81.25	66.67	81.25	0.00
	UpperCase	6.25	0.00	6.25	0.00
Typos	*Least*	42.86	30.77	42.86	0.55
	Most	42.86	15.38	35.71	0.95
Negation	*Shallow*	96.77	89.66	96.77	0.11
	Double	77.78	66.67	77.78	0.31
Entity Rep.	*At least 1*	93.75	93.33	93.75	0.21
	All	46.67	14.29	46.67	0.63
LLM Rewrite	*Least*	48.48	19.35	48.48	0.58
	Most	43.75	20.00	37.50	0.86
Dialect	*AAE*	33.92	12.11	32.29	0.71
	Jamaican	29.52	9.42	27.92	0.75
	Pidgin	30.37	10.25	28.78	0.74
	Singlish	35.45	14.15	33.50	0.71

Table 8: Degree of lexical overlap between unperturbed and perturbed input claims for the ***FactCheckTweet*** dataset. The table reports ROUGE F1 scores (R1, R2, RL) and the normalized Levenshtein distance (Lev.). Higher ROUGE scores and lower Levenshtein distances indicate higher lexical overlap.

B Human Evaluation

We conduct a human evaluation to assess the effectiveness of our perturbation generation framework in producing ***valid*** and ***natural*** perturbed claims. Our evaluation follows a similar annotation task to that of Dyrmishi et al. (2023b), which assessed adversarial examples in human settings. We randomly sample 20 unique perturbed input claims from three perturbation types—*typos*, *entity replacement*, and *dialect*. Half of these claims are labeled as valid (1) and the other half as invalid (0) by the *LLM as a Verifier*. We then present both the unperturbed and perturbed claims, along with the annotation instructions provided below, to three annotators, all of whom are authors. No compensation was paid to the author-annotators.

Annotation Instructions

Your task: For each perturbed claim, evaluate the following criteria:

1. Perturbation Accuracy Does the input claim contain the `perturbation_type`?
☐ Yes ☐ No

2. Validity and Naturalness

1. Does the fact-check apply to it, i.e., is the fact-check helpful in verifying the perturbed claim? ☐ Yes ☐ No
2. Does it convey the same main claim as the unperturbed claim? ☐ Yes ☐ No
3. Does the perturbed claim feel like something a typical social media user might write? ☐ Yes ☐ No

We first assess whether the perturbed claim contains the intended perturbation type compared to the original. We define this as perturbation accuracy, measured as the proportion of instances in which the LLM as a Perturber successfully applies the intended change. Next, we evaluate validity (and naturalness) by collecting binary labels from three independent annotators, using an all-or-nothing rule where a positive label is assigned only if all annotators agree. For validity, we report precision on the positive class, reflecting the reliability of the LLM as a Verifier in identifying valid perturbations. Table 9 shows that our framework achieves a perturbation accuracy of 96.70%, indicating that the intended perturbation was applied in nearly all cases. The verifier attained a 100% precision on the positive class, meaning that every claim labelled as valid by the model was confirmed by human annotators. However, the overall recall score for validity is lower (55.67%), suggesting that while the verifier is highly precise, it may be overly conservative, resulting in some valid instances getting filtered out.

Type	# Count	Acc (%)	Valid (Pr.)	Valid (Re.)	Valid (F1)
Typos	20	100	100	52.63	68.97
Entity Rep.	20	90	100	58.82	72.07
Dialect	20	100	100	55.56	71.43
Total	60	96.70	100	55.67	70.82

Table 9: **Perturbation Evaluation: Acc (%)** indicates the proportion of instances where the perturbation was correctly applied by the *LLM As a Perturber*. **Valid (Pr.)** represents the precision on valid instances, **Valid (Re.)** is the recall on valid instances, and **Valid (F1)** is the overall F1 score.

C Embedding Models Details

The list of all the embedding models evaluated in this study; their parameter count and embedding dimensions are shown in Table 10. We use all embedding models off-the-shelf with default hyperparameters, utilizing the `SentenceTransformers` library.[15] Since the *FactCheckTweet* dataset contains longer articles and most embedding models support only up to 512 tokens, we split articles into paragraphs and compute similarity between each paragraph and the input text, following Kazemi et al. (2021). This approach is also applied to LLM-based embeddings, despite their support for longer sequences, to ensure fair comparison. To run the BM25 evaluation, we used the `rank_bm25` implementation.[16]

C.1 Training Setup and Hyperparameters

C.1.1 Finetuning Embedding Models

We investigate the robustness of finetuned embedding models by finetuning `all-mpnet-base-v2` and `sentence-t5-base` on the *CheckThat22* training split, which consists of $1,195$ input claims. We follow a similar approach to Shliselberg and Dori-Hacohen (2022) and finetune the embedding models using contrastive learning with the Multiple Negatives Ranking (MNR) loss. MNR loss optimizes the similarity between positive claim–factcheck pairs while reducing the similarity of negative pairs within a batch. It is defined as:

$$-\frac{1}{n}\sum_{i=1}^{n}\log\frac{\exp(\text{sim}(a_i,p_i))}{\sum_j \exp(\text{sim}(a_i,p_j))} \quad (2)$$

where $f(x)$ and $f(y)$ are the embedding representations of x (input claim) and y (factcheck), respectively. To improve generalization, we introduce "hard negatives" by using BM25 to generate a ranked list of candidate negatives and selecting the top-ranked negative factcheck. We use the training scripts provided by Shliselberg and Dori-Hacohen (2022)[17] with the default hyperparameters: `learning rate` = $5e-6$, `batch size` = 6, `max length` = 128, `temperature` = 0.1, and `epochs` = 1.

Embedding Model	# Params	d_{emb}
`all-MiniLM-L12-v2`	82M	384
`all-distilroberta-v1`	82M	768
`all-mpnet-base-v2`	110M	768
`sentence-t5-base`	110M	768
`sentence-t5-large`	335M	768
`hkunlp/instructor-base`	86M	768
`hkunlp/instructor-large`	335M	768
`nvidia/NV-Embed-v2`	7B	4,096
`Salesforce/SFR-Embedding-Mistral`	7B	4,096

Table 10: Embedding models with their parameter counts (# Params) and embedding dimensions (d_{emb}). Model names are hyperlinked.

C.1.2 Teacher-Student Knowledge Distillation

We provide the details of the knowledge-distillation approach for improving the robustness of weaker embedding models in this section. This approach depends on parallel sentence pairs $((s_1, t_1), \ldots, (s_n, t_n))$ where s_i is a source sentence and t_i the target sentence. In our case, s_i and t_i are variations of the same claim. We obtain these parallel sentences by generating perturbations of the *CheckThat22* training split for *typos*, *negation*, *dialect*, and *entity replacement*, resulting in an initial set of $11,593$ pairs. We then expand this list by pairing instances of the same claim (e.g., *dialect*

[15] `https://sbert.net/`

[16] `https://github.com/dorianbrown/rank_bm25`

[17] `https://github.com/RIET-lab/GenerativeClaimMatchingPipeline/blob/main/src/dynamicquery/candidate_selection/train_sentence_model.py`

Source Sentence (s_i)	Target Sentence (t_i)	Pairing (s_i vs t_i)
The US drone attack on #Soleimani caught on camera. #IranUsa	Dem catch US drone strike on Soleimani pon camera, yah. #IranUsa	*Unperturbed* vs. *Jamaican Patois*
The American drone strike on #Qassem caught on camera. #TehranWashington	Dem capture US drone attack wey hit Soleimani for cam. #IranUsa	*Entity Replacement* vs. *Nigerian Pidgin*
It is not false that the US drone attack on #Soleimani was caught on camera.	Wah, US drone attack on Soleimani kenna caught on video leh. #IranUsa	*Negation* vs. *Singlish*
US drone strike on Soleimani captrd on video. #IranUsa.	US drone strike pon Soleimani seen on di camera, seen. #IranUsa	*Typos* vs. *Jamaican Patois*

Table 11: Examples of parallel sentences generated using various perturbations.

vs. *typos*), yielding a larger dataset of $70,954$ pairs. Examples of parallel sentences are shown in Table 11. We adopt the same training approach as in Reimers and Gurevych (2020) where we train a student model $\hat{M}$ such that

$$\hat{M}(s_i) \approx M(s_i) \quad \text{and} \quad \hat{M}(t_i) \approx M(s_i).$$

For a given minibatch B, we minimize the mean-squared loss,

$$\mathcal{L}_{\text{MSE}} = \frac{1}{|B|} \sum_{j \in B} \left[\left(M(s_j) - \hat{M}(s_j) \right)^2 + \left(M(s_j) - \hat{M}(t_j) \right)^2 \right]$$

We use `all-mpnet-base-v2` as both the student ($\hat{M}$) and teacher (M) model. To train the model, we use the script provided by Reimers and Gurevych (2020)[18] with the following hyperparameters: `learning rate` = $2e-5$, `batch size` = 64, `max length` = 256, and epochs = 20.

D Rerankers Evaluation

D.1 Selecting a strong reranker

Given the many possible combinations between embedding models and rerankers, we evaluate a wide range of reranker models to select a strong-performing one for further evaluation. In our study, we consider several types of rerankers: **pairwise rerankers**, such as `ms-marco-MiniLM-L-6-v2` and `monot5-3b` (Nogueira et al., 2020); **listwise rerankers**, such as `LiT5-Distill` (Tamber et al., 2023); and **LLM-based rerankers**, including `bge-reranker-v2-gemma` (Chen et al., 2024), `rank_zephyr_7b_v1_full` (Pradeep et al., 2023), and `RankGPT` (Sun et al., 2023) (for which we use `GPT4o` as the reranker). We evaluate these rerankers using the first-stage retrieval rankings from `all-mpnet-base-v2` on the unperturbed *CheckThat22* test set, setting the number of candidates, i.e., *top-j*, to 20. A value of 20 is chosen as it is a feasible size that fits well within the context window for `RankGPT`. We use `RankLLM`[19] implementation for the different reranker models.

Table 12 presents the performance results for the different rerankers. RankGPT, using GPT-4o as the reranker, achieves the best performance. However, its results are comparable to the `bge-reranker-v2-gemma` model, based on `gemma-2b`. Given that `bge-reranker-v2-gemma` is lightweight and can be run locally with lower latency and cost, we select it as a representative SoTA reranker over GPT-4o, which increases latency in production retrieval systems (Sun et al., 2023).

[18] https://github.com/UKPLab/sentence-transformers/blob/master/examples/training/multilingual/make_multilingual.py

[19] https://github.com/castorini/rank_llm

Model	MAP@1	MAP@5	MAP@10	MAP@20
`RankGPT(GPT4o)`	**0.908**	**0.919**	**0.919**	**0.919**
`bge-reranker-v2-gemma`	0.908	0.918	0.918	0.918
`monot5-3b-msmarco-10k`	0.897	0.912	0.912	0.912
`LiT5-Distill-large-v2`	0.876	0.902	0.902	0.902
`bge-reranker-v2-m3`	0.870	0.891	0.892	0.892
`ms-marco-MiniLM-L-6-v2`	0.854	0.883	0.884	0.884
`rank_zephyr_7b_v1_full`	0.778	0.845	0.846	0.846

Table 12: Performance comparison of rerankers on *CheckThat22* (unperturbed test set). The table reports Mean Average Precision (MAP@k) at different cutoffs ($k = 1, 5, 10, 20$) for rerankers applied to first-stage top-20 candidates from `all-mpnet-base-v2`.

D.2 Selecting the optimal *top-j* candidates for reranking

To determine the optimal number of candidates for reranking (*top-j*), we evaluate MAP@5 across different values of $j \in \{5, 10, 20, 50, 100, 200\}$, on the *CheckThat22* unperturbed test set. We use MAP@5 to ensure comparability across different values of *j*. We use two rerankers: `cross-encoder/ms-marco-MiniLM-L-6-v2` and `bge-reranker-v2-gemma`, applied to first-stage retrieval rankings from `all-mpnet-base-v2`. Figure 3 presents the reranking performance of cross-encoder/ms-marco-MiniLM-L-6-v2, while Figure 4 shows that of bge-reranker-v2-gemma. The cross-encoder/ms-marco-MiniLM-L-6-v2 achieves its highest MAP@5 at $j = 50$, whereas the bge-reranker-v2-gemma shows minimal improvement beyond this point. Based on this, we set *top-j* to 50 for the reranking experiments, balancing performance and efficiency.

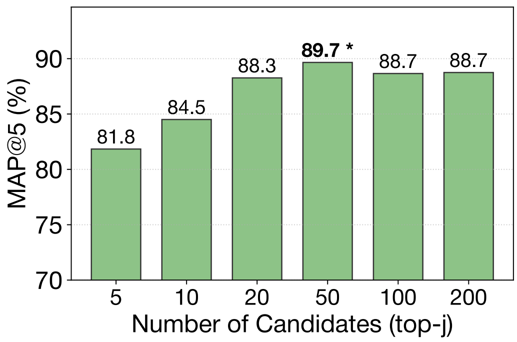

Figure 3: MAP@5 performance across different *top-j* values for **cross-encoder/ms-marco-MiniLM-L-6-v2** on *CheckThat22*. The highest MAP@5 is achieved at $j = 50$.

E Compute Requirements

All embedding model inferences, reranker fine-tuning, and evaluations are conducted on a single NVIDIA A100 80GB GPU.

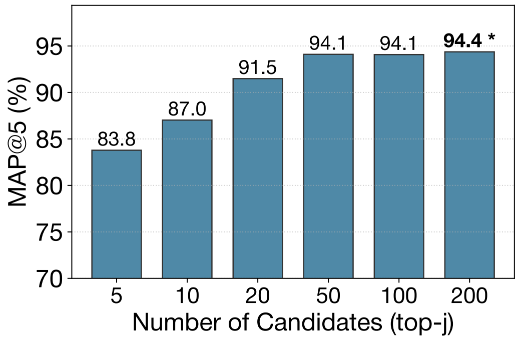

Figure 4: MAP@5 performance across different *top-j* values for **bge-reranker-v2-gemma** on *CheckThat22*. Reranking improves up to $j = 50$, after which further gains are minimal.

F Additional Results

F.1 Additional Results Across Different MAP@*k* on *CheckThat22*

We provide robustness evaluation results across different MAP@k values, $k \in \{5, 10, 50\}$, for the *CheckThat22* dataset. Table 13 reports the evaluation at MAP@*5*, Table 14 at MAP@*10*, and Table 15 at MAP@*50*, with reranking applied to the top 50 candidates in all cases. We observe minimal performance variations across different MAP@k values on the *CheckThat22* dataset, indicating that the robustness results hold across various MAP@k values.

F.2 Results for *FactCheckTweet*

We report evaluation results across different MAP@k values, $k \in \{5, 10, 20, 50\}$, for the *FactCheckTweet* dataset. Table 16 presents the evaluation at MAP@5, Table 17 at MAP@10 and Table 18 at MAP@50, with reranking applied to the top 50 candidates in all cases.

G Mitigation Approaches Comparison

We compare our mitigation approaches on the Robust-LASER (`RoLASER`) model proposed by Nishimwe et al. (2024), who use a similar teacher–student knowledge distillation approach to improve the robustness of embedding models for user-generated content (UGC). In contrast to our work, they focus only on a subset of perturbations—specifically, those in which UGC resembles the *Typos* perturbation type—and generate an artificial UGC dataset using rule-based methods. For the KD approach, we compare the performance of the original `LASER` model with that of Robust-LASER (`RoLASER`), the adapted model. We also

		Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
Stage	Model	*TrueCase*	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	-0.06	+0.00	-15.33	-15.19	+0.57	-2.59	-2.79	-12.83	-0.85	-0.25	-9.18	-6.64	-5.03	-1.71
	all-distilroberta-v1	-0.64	-35.92	-16.09	-13.80	+1.75	+2.15	-2.97	-5.09	+3.70	+5.44	-4.50	-12.01	-6.71	-0.87
	all-MiniLM-L12-v2	+0.00	+0.00	-8.65	-13.77	+3.94	+4.83	-3.19	-2.63	+3.34	+4.77	-6.67	-10.02	-5.34	+0.49
	all-mpnet-base-v2	+0.00	+0.00	-8.87	-8.74	+0.01	-4.03	-3.52	-10.11	+0.76	+1.96	-3.85	-7.60	-8.36	-1.38
	all-mpnet-base-v2-ft	+0.00	+0.00	-8.65	-13.25	-1.05	-4.13	-1.73	-9.04	-1.44	-1.69	-4.09	-4.23	-8.88	-3.29
	sentence-t5-base	-2.81	-16.05	-3.09	-7.11	-21.94	-12.49	-3.34	-6.06	+2.41	+2.11	-1.09	-8.40	-5.72	+0.09
	sentence-t5-large	-0.78	-9.95	-1.94	-5.25	-28.08	-14.80	-0.81	-5.40	+2.07	+1.57	+0.86	-3.53	-3.06	+1.91
	sentence-t5-large-ft	-0.59	-10.12	-2.07	-4.17	-12.14	-6.76	-2.90	-1.41	+1.99	+1.01	+1.44	-2.61	-3.82	+1.09
	instructor-base	-1.11	-9.83	-4.35	-3.59	-3.83	-3.75	-1.76	-8.33	-0.15	-0.95	-1.26	-4.45	-4.89	-1.52
	instructor-large	-0.70	-4.65	-1.18	-1.76	-4.19	-5.07	+0.00	-2.80	+0.09	-0.79	-0.84	-2.57	-4.64	-2.12
	SFR-Embedding-Mistral	-0.62	-0.56	-0.76	-0.62	-1.37	-2.87	-0.64	-1.15	+0.09	+0.68	-2.05	-0.76	-1.15	-2.21
	NV-Embed-v2	+0.00	-0.62	+0.61	-0.04	-0.50	-2.51	+0.23	-0.38	-0.09	-0.27	-1.07	-0.30	-1.13	-1.37
Reranking Recovery	BM25	+11.49	+11.29	+23.63	+20.86	+6.31	+7.52	+13.25	+19.50	+10.18	+9.68	+14.89	+15.52	+14.55	+13.66
	all-distilroberta-v1	+21.11	+31.78	+27.63	+25.36	+14.38	+12.73	+22.37	+16.76	+16.33	+14.64	+20.90	+27.21	+21.76	+18.38
	all-MiniLM-L12-v2	+19.60	+20.29	+20.56	+26.21	+12.40	+7.26	+24.08	+20.38	+14.05	+14.68	+22.64	+23.98	+18.78	+20.19
	all-mpnet-base-v2	+19.59	+20.31	+21.88	+22.22	+13.17	+13.81	+22.00	+26.56	+16.58	+15.49	+22.11	+23.39	+22.05	+18.24
	all-mpnet-base-v2-ft	+6.19	+6.87	+10.39	+17.12	+0.85	+5.41	+9.54	+12.02	+6.83	+6.95	+9.79	+9.26	+14.98	+10.58
	sentence-t5-base	+15.68	+23.48	+15.62	+19.29	+23.71	+16.66	+15.99	+21.57	+10.56	+10.73	+15.03	+20.69	+18.97	+13.93
	sentence-t5-large	+14.26	+20.68	+15.94	+19.18	+25.30	+17.67	+16.23	+21.47	+14.30	+13.68	+16.25	+17.22	+14.81	+14.26
	sentence-t5-large-ft	+5.40	+14.62	+6.48	+8.41	+8.51	+6.91	+8.31	+8.69	+3.60	+4.11	+3.53	+6.69	+9.36	+4.64
	instructor-base	+5.68	+13.03	+6.22	+6.68	+1.07	+2.98	+7.31	+14.88	+4.57	+5.30	+5.99	+9.23	+9.75	+6.59
	instructor-large	+2.12	+6.01	+1.11	+2.33	-1.35	+0.50	+1.93	+4.10	+0.31	+2.19	+1.18	+3.63	+5.51	+4.19
	SFR-Embedding-Mistral	+0.92	+1.92	-1.11	-0.45	-4.45	-1.97	+2.00	+2.45	-0.18	-0.59	+1.50	+0.09	+1.46	+3.50
	NV-Embed-v2	-2.29	-0.74	-4.18	-2.36	-8.51	-3.29	-2.21	-2.37	-2.80	-1.65	-1.79	-2.89	-1.78	-0.15
Overall Pipeline	BM25	-0.25	-0.40	-3.47	-5.86	-3.16	-5.69	-1.20	-3.21	-1.08	-0.99	-5.88	-2.63	-2.26	+0.77
	all-distilroberta-v1	-0.77	-25.37	-9.39	-9.17	-5.08	-5.82	-2.53	-7.49	-1.08	-1.04	-4.68	-5.95	-6.09	-3.56
	all-MiniLM-L12-v2	-0.81	-0.12	-8.71	-7.61	-3.45	-5.41	-0.69	-2.45	-1.15	+0.90	-4.40	-6.12	-6.47	+0.56
	all-mpnet-base-v2	-0.60	+0.12	-5.89	-4.66	-6.71	-7.60	-0.32	-1.30	-1.87	-1.76	-1.79	-4.24	-6.34	-2.35
	all-mpnet-base-v2-ft	-0.60	+0.08	-3.85	-1.81	-6.36	-6.14	+0.92	-2.22	-0.54	-0.68	-0.98	-1.37	-0.69	+0.60
	sentence-t5-base	-0.72	-6.16	-1.50	-1.46	-11.24	-9.04	-1.61	-0.31	-0.63	-0.77	+0.13	-2.00	-0.48	+0.26
	sentence-t5-large	-2.09	-4.84	-2.28	-2.07	-18.45	-12.40	-1.70	-1.45	+0.99	-0.14	+1.62	-1.87	-3.98	+0.68
	sentence-t5-large-ft	-0.08	-0.39	-1.48	-1.38	-9.43	-6.91	-0.18	+0.31	+1.08	+0.60	-0.02	-1.50	+0.48	+0.75
	instructor-base	-0.84	-2.20	-3.34	-2.20	-7.77	-6.59	+0.12	-0.92	-0.39	-0.45	-0.87	-1.43	-0.82	-0.53
	instructor-large	-0.91	-0.97	-2.47	-1.88	-7.99	-8.76	-0.62	-0.69	-1.00	+0.00	-1.60	-1.48	-1.10	+0.13
	SFR-Embedding-Mistral	-0.97	+0.08	-2.22	-1.16	-6.79	-6.98	+0.06	+0.92	-0.90	-0.72	-1.79	-2.20	-0.95	+0.06
	NV-Embed-v2	-0.59	+0.34	-2.67	-1.23	-7.84	-6.67	-0.68	+0.05	-1.11	-0.14	-1.36	-2.19	-1.38	-0.27

Table 13: Effect of perturbations on ***CheckThatTweet*** (MAP@5). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@5})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@5 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

		Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
Stage	Model	*TrueCase*	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	+0.01	+0.00	-15.32	-15.19	+0.42	-2.62	-2.81	-12.48	-0.82	-0.32	-9.28	-6.84	-5.17	-1.75
	all-distilroberta-v1	-0.34	-35.57	-16.02	-14.02	+1.93	+2.41	-2.79	-5.31	+3.61	+5.68	-4.34	-11.44	-6.63	-0.69
	all-MiniLM-L12-v2	+0.00	+0.00	-8.43	-13.72	+3.44	+4.47	-2.96	-2.73	+3.08	+4.44	-6.99	-10.17	-5.94	+0.09
	all-mpnet-base-v2	+0.00	+0.00	-8.08	-8.31	+0.25	-4.04	-3.00	-9.55	+1.08	+2.00	-3.70	-7.38	-8.31	-1.21
	all-mpnet-base-v2-ft	+0.00	+0.00	-8.60	-13.05	-1.27	-4.29	-1.83	-8.72	-1.34	-1.65	-3.88	-4.09	-8.64	-3.29
	sentence-t5-base	-2.55	-15.89	-2.88	-7.17	-21.48	-12.29	-3.41	-5.56	+2.40	+1.93	-0.91	-8.29	-5.55	-0.03
	sentence-t5-large	-0.82	-9.83	-1.98	-5.37	-27.14	-14.62	-0.82	-5.39	+2.00	+1.81	+0.92	-3.29	-3.14	+2.05
	sentence-t5-large-ft	-0.59	-9.72	-1.93	-3.64	-11.88	-6.73	-2.54	-1.73	+2.19	+0.94	+1.52	-2.40	-3.86	+1.06
	instructor-base	-1.08	-9.35	-4.30	-3.31	-3.57	-3.80	-1.56	-8.22	+0.21	-0.74	-1.03	-4.01	-4.22	-1.48
	instructor-large	-0.76	-4.51	-1.10	-1.81	-4.11	-4.87	+0.12	-2.49	+0.04	-0.85	-0.84	-2.49	-4.19	-2.05
	SFR-Embedding-Mistral	-0.63	-0.59	-0.68	-0.50	-1.32	-2.98	-0.83	-1.15	+0.09	+0.73	-1.95	-0.87	-1.26	-2.23
	NV-Embed-v2	+0.00	-0.62	+0.68	-0.04	-0.50	-2.44	+0.23	-0.29	-0.09	-0.19	-1.07	-0.25	-0.95	-1.23
Reranking Recovery	BM25	+10.94	+10.80	+23.11	+20.34	+6.17	+7.29	+12.82	+18.65	+9.86	+9.44	+14.56	+15.27	+14.24	+13.26
	all-distilroberta-v1	+20.31	+30.93	+27.03	+25.04	+13.63	+12.11	+21.62	+16.56	+15.96	+14.00	+20.30	+26.20	+21.24	+17.76
	all-MiniLM-L12-v2	+18.87	+19.55	+19.62	+25.51	+12.13	+7.01	+23.18	+19.79	+13.77	+14.53	+22.25	+23.41	+18.67	+19.87
	all-mpnet-base-v2	+19.07	+19.80	+20.63	+21.33	+12.54	+13.62	+21.25	+25.56	+15.78	+15.03	+21.43	+22.62	+21.46	+17.61
	all-mpnet-base-v2-ft	+5.99	+6.67	+10.13	+16.76	+0.94	+5.39	+9.49	+11.48	+6.51	+6.75	+9.38	+8.91	+14.53	+10.37
	sentence-t5-base	+15.09	+22.93	+14.99	+18.93	+22.90	+16.21	+15.61	+20.61	+10.14	+10.60	+14.44	+20.16	+18.38	+13.65
	sentence-t5-large	+13.79	+20.11	+15.44	+18.87	+23.88	+16.99	+15.67	+21.01	+13.89	+13.12	+15.74	+16.59	+14.43	+13.67
	sentence-t5-large-ft	+5.37	+14.13	+6.27	+7.81	+8.35	+6.83	+7.75	+8.69	+3.36	+4.10	+3.32	+6.29	+9.29	+4.55
	instructor-base	+5.63	+12.38	+6.13	+6.45	+0.90	+3.16	+7.09	+14.61	+4.17	+5.06	+5.72	+8.61	+8.97	+6.38
	instructor-large	+2.19	+5.89	+1.02	+2.31	-1.09	+0.32	+1.82	+3.68	+0.23	+2.24	+1.12	+3.42	+5.02	+4.00
	SFR-Embedding-Mistral	+0.77	+1.86	-1.20	-0.51	-4.25	-1.71	+2.00	+2.14	-0.27	-0.68	+1.22	+0.02	+1.48	+3.35
	NV-Embed-v2	-2.21	-0.67	-4.25	-2.30	-8.18	-3.02	-2.13	-2.46	-2.80	-1.67	-1.79	-2.95	-1.87	-0.29
Overall Pipeline	BM25	-0.25	-0.40	-3.47	-5.86	-3.07	-5.49	-1.20	-3.21	-1.08	-0.99	-5.88	-2.63	-2.26	+0.77
	all-distilroberta-v1	-0.85	-25.45	-9.39	-9.17	-5.17	-5.82	-2.53	-7.49	-1.08	-0.98	-4.68	-5.95	-6.09	-3.56
	all-MiniLM-L12-v2	-0.81	-0.12	-8.71	-7.61	-3.45	-5.41	-0.60	-2.45	-1.15	+0.97	-4.40	-6.12	-6.47	+0.56
	all-mpnet-base-v2	-0.60	+0.12	-5.89	-4.66	-6.71	-7.53	-0.25	-1.30	-1.87	-1.70	-1.79	-4.24	-6.34	-2.35
	all-mpnet-base-v2-ft	-0.60	+0.08	-3.85	-1.75	-6.27	-6.04	+1.00	-2.22	-0.54	-0.61	-0.98	-1.37	-0.69	+0.60
	sentence-t5-base	-0.71	-6.21	-1.50	-1.46	-11.20	-8.93	-1.61	-0.31	-0.68	-0.70	+0.08	-2.00	-0.53	+0.21
	sentence-t5-large	-2.09	-4.77	-2.28	-2.01	-18.45	-12.40	-1.70	-1.45	+0.99	+0.03	+1.62	-1.87	-3.98	+0.68
	sentence-t5-large-ft	-0.01	-0.38	-1.42	-1.33	-9.13	-6.83	-0.26	+0.17	+1.09	+0.58	-0.03	-1.58	+0.48	+0.74
	instructor-base	-0.75	-2.26	-3.28	-2.06	-7.66	-6.34	+0.19	-0.92	-0.39	-0.46	-0.79	-1.43	-0.82	-0.59
	instructor-large	-0.78	-0.83	-2.36	-1.83	-7.52	-8.66	-0.48	-0.69	-1.00	+0.11	-1.53	-1.48	-1.02	+0.13
	SFR-Embedding-Mistral	-0.97	+0.15	-2.22	-1.10	-6.44	-6.72	+0.06	+0.92	-0.90	-0.67	-1.79	-2.20	-0.87	+0.06
	NV-Embed-v2	-0.56	+0.36	-2.67	-1.18	-7.56	-6.32	-0.60	+0.05	-1.16	-0.14	-1.41	-2.19	-1.35	-0.32

Table 14: Effect of perturbations on ***CheckThat22*** (MAP@10). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@10})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@10 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

evaluate the effect of *claim normalization* using the same setup as in our approach. The results for this experiment are shown in Figure 5.

Although the mitigation approaches are not directly comparable here due to the use of different embedding models (i.e., `LASER` vs.

Stage	Model	Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
		TrueCase	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	+0.01	+0.00	-14.93	-14.88	+0.42	-2.57	-2.76	-12.42	-0.80	-0.26	-9.15	-6.71	-5.03	-1.67
	all-distilroberta-v1	-0.35	-35.26	-15.70	-13.92	+1.82	+2.32	-2.93	-5.31	+3.46	+5.29	-4.38	-11.30	-6.63	-0.67
	all-MiniLM-L12-v2	+0.00	+0.00	-8.18	-13.46	+3.61	+4.43	-2.89	-2.44	+3.11	+4.52	-6.66	-9.84	-5.68	+0.33
	all-mpnet-base-v2	+0.00	+0.00	-8.30	-8.21	+0.16	-4.15	-3.11	-9.74	+0.90	+1.81	-3.84	-7.40	-8.47	-1.30
	all-mpnet-base-v2-ft	+0.00	+0.00	-8.51	-12.65	-1.33	-4.24	-1.80	-8.55	-1.31	-1.70	-3.83	-4.01	-8.43	-3.17
	sentence-t5-base	-2.45	-15.66	-2.84	-6.99	-21.12	-12.07	-3.24	-5.62	+2.34	+1.98	-0.79	-7.96	-5.38	-0.01
	sentence-t5-large	-0.95	-9.79	-2.00	-5.38	-27.10	-14.48	-0.82	-5.38	+1.88	+1.72	+0.88	-3.23	-3.17	+2.05
	sentence-t5-large-ft	-0.57	-9.65	-1.84	-3.63	-11.56	-6.56	-2.50	-1.54	+2.18	+1.00	+1.46	-2.40	-3.61	+1.15
	instructor-base	-1.07	-9.36	-4.35	-3.22	-3.70	-3.83	-1.61	-8.00	+0.10	-0.79	-1.04	-4.03	-4.29	-1.53
	instructor-large	-0.68	-4.49	-1.09	-1.68	-4.09	-4.85	+0.09	-2.50	+0.05	-0.84	-0.86	-2.47	-4.08	-1.97
	SFR-Embedding-Mistral	-0.60	-0.57	-0.68	-0.50	-1.23	-2.86	-0.83	-1.07	+0.13	+0.73	-1.95	-0.85	-1.17	-2.18
	NV-Embed-v2	+0.00	-0.62	+0.68	-0.04	-0.50	-2.40	+0.23	-0.29	-0.09	-0.19	-1.04	-0.25	-0.95	-1.18
Reranking Recovery	BM25	+10.78	+10.65	+22.55	+19.87	+6.09	+7.18	+12.64	+18.50	+9.71	+9.26	+14.28	+14.99	+13.95	+13.03
	all-distilroberta-v1	+19.85	+30.16	+26.29	+24.55	+13.24	+11.75	+21.28	+16.20	+15.64	+13.92	+19.90	+25.60	+20.79	+17.29
	all-MiniLM-L12-v2	+18.63	+19.31	+19.12	+24.99	+11.76	+6.91	+22.84	+19.19	+13.51	+14.21	+21.68	+22.84	+18.18	+19.39
	all-mpnet-base-v2	+18.60	+19.33	+20.42	+20.86	+12.14	+13.26	+20.86	+25.26	+15.49	+14.76	+21.12	+22.18	+21.15	+17.25
	all-mpnet-base-v2-ft	+5.88	+6.57	+9.96	+16.28	+0.88	+5.31	+9.35	+11.17	+6.41	+6.72	+9.23	+8.73	+14.21	+10.15
	sentence-t5-base	+14.85	+22.58	+14.80	+18.58	+22.37	+15.86	+15.28	+20.38	+10.09	+10.38	+8.78	+19.67	+18.09	+13.47
	sentence-t5-large	+13.60	+19.75	+15.14	+18.55	+23.49	+16.57	+15.37	+20.57	+13.69	+12.90	+15.43	+16.19	+14.11	+13.32
	sentence-t5-large-ft	+5.26	+14.00	+6.09	+7.71	+7.94	+6.59	+7.67	+8.36	+3.33	+3.98	+3.33	+6.24	+8.98	+4.38
	instructor-base	+5.45	+12.24	+6.05	+6.22	+0.84	+3.09	+6.96	+14.16	+4.15	+4.97	+5.57	+8.46	+8.86	+6.30
	instructor-large	+2.13	+5.84	+0.99	+2.14	-1.15	+0.44	+1.78	+3.57	+0.24	+2.24	+1.16	+3.37	+4.89	+3.89
	SFR-Embedding-Mistral	+0.74	+1.84	-1.20	-0.51	-4.34	-1.75	+2.04	+2.06	-0.31	-0.68	+1.22	+0.00	+1.38	+3.30
	NV-Embed-v2	-2.21	-0.67	-4.21	-2.30	-8.18	-3.03	-2.08	-2.46	-2.76	-1.67	-1.78	-2.95	-1.84	-0.34
Overall Pipeline	BM25	-0.25	-0.40	-3.47	-5.86	-3.07	-5.49	-1.20	-3.21	-1.08	-0.99	-5.88	-2.63	-2.26	+0.77
	all-distilroberta-v1	-0.85	-25.45	-9.39	-9.17	-5.17	-5.77	-2.53	-7.44	-1.08	-0.98	-4.68	-5.95	-6.09	-3.56
	all-MiniLM-L12-v2	-0.81	-0.12	-8.71	-7.61	-3.45	-5.34	-0.60	-2.45	-1.15	+0.97	-4.40	-6.12	-6.47	+0.56
	all-mpnet-base-v2	-0.60	+0.12	-5.89	-4.66	-6.71	-7.53	-0.25	-1.26	-1.87	-1.70	-1.79	-4.24	-6.34	-2.35
	all-mpnet-base-v2-ft	-0.60	+0.08	-3.85	-1.75	-6.27	-5.92	+1.00	-2.22	-0.54	-0.61	-0.98	-1.37	-0.69	+0.60
	sentence-t5-base	-0.71	-6.18	-1.50	-1.46	-11.20	-8.93	-1.61	-0.31	-0.64	-0.70	-5.31	-2.00	-0.49	+0.21
	sentence-t5-large	-2.09	-4.77	-2.28	-2.01	-18.45	-12.35	-1.66	-1.45	+0.99	+0.03	+1.62	-1.87	-3.98	+0.68
	sentence-t5-large-ft	-0.05	-0.40	-1.42	-1.33	-9.18	-6.80	-0.22	+0.17	+1.09	+0.56	-0.03	-1.54	+0.46	+0.69
	instructor-base	-0.75	-2.26	-3.28	-2.06	-7.66	-6.29	+0.19	-0.92	-0.39	-0.46	-0.79	-1.43	-0.82	-0.55
	instructor-large	-0.77	-0.88	-2.36	-1.83	-7.52	-8.43	-0.52	-0.76	-1.05	+0.07	-1.53	-1.48	-1.06	+0.08
	SFR-Embedding-Mistral	-0.97	+0.15	-2.22	-1.10	-6.44	-6.64	+0.11	+0.92	-0.90	-0.67	-1.79	-2.20	-0.87	+0.06
	NV-Embed-v2	-0.56	+0.36	-2.64	-1.18	-7.56	-6.29	-0.56	+0.05	-1.12	-0.14	-1.36	-2.19	-1.32	-0.32

Table 15: Effect of perturbations on ***CheckThat22*** (MAP@50). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@50})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@50 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

Stage	Model	Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
		TrueCase	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	+0.70	+0.00	-10.09	-12.40	+2.73	+6.28	-2.68	-19.95	+2.35	-2.33	-6.97	-11.03	-11.50	-2.72
	all-distilroberta-v1	-1.01	-23.72	-7.28	-12.91	+6.03	+10.71	+1.12	-10.38	+4.97	+2.07	-5.60	-8.93	-12.09	-6.56
	all-MiniLM-L12-v2	+0.00	+0.00	-13.69	-17.66	+3.31	+5.26	-3.00	-10.85	+1.93	-3.06	-6.67	-15.52	-13.59	-10.06
	all-mpnet-base-v2	+0.00	+0.00	-9.67	-11.56	-1.25	-0.99	-3.69	-3.55	-5.85	-9.54	-7.14	-8.56	-9.24	-9.09
	all-mpnet-base-v2-ft	+0.00	+0.00	-6.70	-8.46	+3.57	-1.73	-4.74	-12.70	+2.40	+1.41	-4.60	-6.05	-7.53	-4.16
	sentence-t5-base	-0.01	-8.70	-7.03	-8.41	-11.47	-11.80	+1.68	-1.58	-5.80	-4.56	-3.10	-10.05	-9.62	-4.94
	sentence-t5-large	+0.19	-6.29	-6.27	-7.83	-16.40	-5.87	-0.26	-11.94	-3.72	-6.39	-5.57	-6.70	-8.13	-4.78
	sentence-t5-large-ft	-0.70	-5.43	-5.80	-6.44	-7.06	-0.94	-0.32	-11.42	-3.60	-2.41	-3.47	-4.42	-6.60	-4.48
	instructor-base	-0.87	-8.21	-4.76	-8.62	+2.83	-0.19	-2.31	-10.49	-2.58	-4.34	-2.62	-5.07	-7.97	-6.64
	instructor-large	-0.43	-3.44	-4.45	-5.88	-1.22	-4.64	-2.55	-12.02	+0.03	-3.07	-2.34	-2.61	-3.41	-5.99
	SFR-Embedding-Mistral	-0.84	-0.43	+0.18	-4.42	+0.88	+2.21	-1.52	-9.45	-0.23	-0.59	-3.06	+0.14	-1.63	+1.82
	NV-Embed-v2	-1.37	-1.01	-1.18	-2.50	+0.04	-0.99	-1.52	-13.72	-1.67	-2.19	-2.94	-3.05	-2.91	-4.80
Reranking Recovery	BM25	+5.26	+6.18	+13.90	+12.25	+5.75	+6.51	+12.71	+14.81	+10.56	+12.96	+12.39	+17.09	+17.99	+9.02
	all-distilroberta-v1	+12.44	+16.97	+20.22	+15.74	+14.30	+16.94	+13.99	+13.91	+15.22	+16.30	+15.73	+24.27	+22.42	+17.19
	all-MiniLM-L12-v2	+12.55	+13.41	+20.38	+18.90	+15.28	+12.36	+18.16	+14.75	+14.21	+20.91	+20.83	+29.85	+19.10	+22.37
	all-mpnet-base-v2	+12.54	+13.05	+16.53	+19.34	+18.58	+18.13	+20.79	+16.07	+17.33	+23.76	+20.07	+19.27	+17.79	+18.00
	all-mpnet-base-v2-ft	+11.42	+11.91	+12.53	+12.50	+9.70	+15.60	+18.99	+13.93	+11.14	+14.44	+13.59	+15.45	+18.34	+13.94
	sentence-t5-base	+13.92	+18.13	+19.21	+17.69	+18.83	+23.59	+12.98	+12.92	+20.27	+17.93	+17.18	+24.67	+22.42	+20.97
	sentence-t5-large	+14.68	+19.06	+22.10	+17.10	+24.05	+20.15	+17.20	+19.75	+19.26	+21.28	+20.29	+20.50	+20.01	+19.72
	sentence-t5-large-ft	+10.49	+12.74	+15.66	+13.53	+14.61	+15.14	+16.87	+14.70	+15.76	+13.58	+14.76	+17.73	+19.71	+12.90
	instructor-base	+10.52	+12.73	+12.07	+10.54	+7.52	+11.63	+16.46	+14.29	+13.91	+17.34	+10.57	+14.55	+17.66	+12.64
	instructor-large	+8.34	+8.91	+15.23	+11.47	+11.79	+14.25	+16.30	+17.51	+11.55	+15.22	+9.42	+12.90	+12.98	+16.06
	SFR-Embedding-Mistral	+6.96	+6.96	+9.77	+9.57	+8.20	+8.91	+12.76	+11.72	+12.17	+12.41	+14.58	+15.78	+15.58	+9.81
	NV-Embed-v2	+2.37	+2.40	+4.45	+0.56	+3.11	+5.87	+7.50	+5.16	+6.67	+7.25	+2.02	+4.57	+3.87	+4.18
Overall Pipeline	BM25	+0.06	+0.28	-3.84	-7.80	+1.73	+3.74	-0.51	-19.34	+4.74	+2.40	-1.74	-1.49	-0.68	-0.87
	all-distilroberta-v1	-1.86	-20.03	-6.91	-17.01	+4.89	+9.69	-3.85	-18.20	+0.88	-1.22	-7.42	-2.63	-7.22	-6.92
	all-MiniLM-L12-v2	-0.27	+0.59	-8.94	-14.38	-0.23	+3.74	-5.02	-19.13	+1.23	+2.43	-0.71	-1.04	-9.37	-2.57
	all-mpnet-base-v2	+0.20	+0.71	-7.36	-6.45	+4.54	+6.41	-3.81	-10.25	-2.58	-0.08	+0.33	-2.21	-4.04	-3.68
	all-mpnet-base-v2-ft	-0.19	+0.31	-5.65	-7.44	+1.14	+4.91	-3.85	-9.64	+1.61	+3.83	-1.82	-1.29	+0.00	-1.02
	sentence-t5-base	-0.23	-4.71	-0.51	-3.41	-5.81	+1.41	-0.80	-4.02	+4.02	+2.84	+4.06	+4.40	+2.78	+6.00
	sentence-t5-large	-0.47	-2.58	-0.46	-7.02	-11.68	-1.39	-3.37	-5.87	-1.19	-2.11	-1.63	-2.89	-4.48	-1.41
	sentence-t5-large-ft	+0.29	-2.18	-2.21	-4.99	-5.10	+4.80	+0.67	-11.83	+0.49	-0.65	+1.20	+3.02	+3.02	-1.67
	instructor-base	+0.79	-4.35	-3.04	-8.42	-0.38	+4.97	-1.06	-10.71	+2.26	+3.87	-0.51	+1.05	+1.23	-2.46
	instructor-large	-0.96	-3.40	-1.79	-6.97	-1.90	+1.89	-1.52	-11.80	-0.65	-0.17	-4.10	-1.00	-1.61	-1.11
	SFR-Embedding-Mistral	-0.22	+0.18	+1.07	-3.73	+0.39	+5.92	+1.36	-10.74	+2.82	+2.63	-0.82	+2.64	+1.60	-0.71
	NV-Embed-v2	+0.25	+0.64	+0.74	-4.47	+0.65	+4.85	+0.80	-12.16	+1.52	+1.56	-1.52	+1.04	+0.36	-1.22

Table 16: Effect of perturbations on ***FactCheckTweet*** (MAP@5). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@5})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@5 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

`all-mpnet-base-v2`), we observe that *claim normalization* is generally beneficial across all perturbation types. Additionally, `RoLASER` outperforms the standard `LASER` model on typos and shows marginal improvement on dialect perturbations. We hypothesize that this occurs because `RoLASER` is

Stage	Model	Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
		TrueCase	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	+0.68	+0.00	-9.31	-12.33	+2.51	+5.93	-2.92	-19.50	+2.65	-2.40	-6.71	-10.42	-11.27	-2.54
	all-distilroberta-v1	-1.16	-23.78	-7.85	-13.07	+5.20	+10.70	+0.76	-10.80	+4.58	+1.71	-6.14	-9.01	-12.74	-7.25
	all-MiniLM-L12-v2	+0.00	+0.00	-13.68	-17.92	+2.50	+4.88	-3.86	-11.39	+1.95	-3.08	-7.41	-15.43	-14.19	-10.04
	all-mpnet-base-v2	+0.00	+0.00	-9.23	-11.11	-0.54	-0.73	-3.70	-3.47	-6.08	-9.48	-6.91	-8.48	-8.62	-9.08
	all-mpnet-base-v2-ft	+0.00	+0.00	-6.56	-8.13	+3.66	-1.75	-4.23	-12.79	+2.69	+1.63	-4.17	-5.96	-6.92	-3.93
	sentence-t5-base	-0.21	-8.79	-7.04	-8.43	-12.15	-11.65	+1.72	-2.50	-5.65	-4.84	-2.76	-9.67	-9.04	-4.34
	sentence-t5-large	-0.15	-6.34	-6.58	-7.68	-16.75	-6.07	-0.01	-12.12	-4.07	-6.37	-5.87	-6.92	-8.38	-5.52
	sentence-t5-large-ft	-0.84	-5.29	-5.98	-6.63	-7.01	-1.04	-0.15	-11.21	-3.08	-2.22	-3.39	-3.97	-6.67	-3.96
	instructor-base	-0.80	-8.32	-4.70	-8.09	+3.16	+0.20	-1.72	-10.39	-2.43	-4.41	-2.37	-4.60	-7.07	-5.90
	instructor-large	-0.45	-3.56	-4.42	-5.56	-1.23	-4.46	-2.09	-11.63	-0.03	-2.84	-2.47	-2.09	-3.18	-5.80
	SFR-Embedding-Mistral	-0.88	-0.45	+0.41	-4.22	+1.03	+2.40	-1.84	-9.54	-0.20	-0.87	-3.04	+0.22	-1.62	+1.36
	NV-Embed-v2	-1.32	-1.09	-1.46	-2.87	+0.29	-1.09	-1.26	-12.71	-1.94	-2.52	-3.13	-3.09	-3.37	-5.00
Reranking Recovery	BM25	+4.89	+5.88	+12.73	+11.88	+5.62	+6.12	+12.49	+13.79	+9.82	+12.63	+11.93	+16.11	+17.40	+8.60
	all-distilroberta-v1	+12.13	+16.30	+19.65	+14.86	+14.23	+15.92	+13.86	+13.24	+14.72	+15.61	+15.08	+23.09	+21.76	+16.75
	all-MiniLM-L12-v2	+12.13	+12.99	+19.66	+18.53	+14.87	+12.00	+18.00	+14.04	+13.44	+20.08	+20.42	+28.67	+18.54	+21.32
	all-mpnet-base-v2	+12.03	+12.58	+15.82	+18.64	+17.60	+17.70	+20.22	+15.75	+17.38	+23.39	+19.57	+18.72	+16.59	+17.59
	all-mpnet-base-v2-ft	+11.07	+11.62	+12.47	+12.22	+9.48	+15.23	+18.00	+13.66	+10.88	+14.14	+13.36	+15.57	+17.58	+13.61
	sentence-t5-base	+13.60	+17.64	+18.55	+17.19	+18.63	+22.76	+12.04	+12.51	+19.53	+17.62	+16.27	+23.51	+21.07	+19.70
	sentence-t5-large	+14.44	+18.45	+21.40	+15.93	+23.44	+19.75	+16.48	+19.30	+18.90	+20.56	+19.72	+19.80	+19.35	+19.66
	sentence-t5-large-ft	+10.26	+12.12	+15.36	+13.26	+13.80	+14.62	+15.74	+13.33	+14.69	+12.82	+14.12	+16.73	+19.09	+11.95
	instructor-base	+10.11	+12.47	+11.78	+9.61	+7.07	+10.93	+15.57	+13.45	+13.40	+16.97	+10.21	+13.87	+16.61	+11.88
	instructor-large	+8.11	+8.71	+14.84	+10.39	+11.18	+13.79	+15.44	+16.18	+10.94	+14.34	+9.59	+12.21	+12.59	+15.73
	SFR-Embedding-Mistral	+6.78	+6.78	+9.10	+8.82	+7.45	+8.59	+12.48	+12.37	+11.85	+12.24	+13.93	+14.82	+14.34	+9.54
	NV-Embed-v2	+2.27	+2.49	+4.30	+0.53	+2.33	+5.92	+6.75	+4.04	+6.59	+7.02	+2.07	+4.22	+3.92	+4.08
Overall Pipeline	BM25	+0.10	+0.41	-4.02	-7.89	+1.58	+3.17	-0.19	-19.24	+4.63	+2.33	-1.79	-1.78	-0.88	-0.95
	all-distilroberta-v1	-1.82	-20.27	-6.89	-16.89	+4.92	+9.40	-3.71	-18.20	+1.03	-1.22	-7.56	-2.81	-7.47	-7.00
	all-MiniLM-L12-v2	-0.25	+0.61	-8.90	-14.27	-0.38	+3.39	-4.88	-19.13	+1.16	+2.27	-1.07	-1.13	-9.73	-2.80
	all-mpnet-base-v2	+0.10	+0.65	-7.46	-6.51	+4.24	+5.96	-3.93	-10.36	-2.43	-0.05	+0.28	-2.36	-4.41	-3.87
	all-mpnet-base-v2-ft	-0.10	+0.46	-5.80	-7.61	+0.88	+4.57	-4.01	-10.15	+1.57	+3.68	-1.82	-1.31	-0.34	-1.32
	sentence-t5-base	-0.29	-4.84	-0.44	-3.18	-5.75	+1.10	-0.80	-4.02	+4.02	+2.84	+4.06	+4.19	+2.57	+5.90
	sentence-t5-large	-0.53	-2.71	-0.57	-7.13	-11.83	-1.27	-3.37	-6.22	-1.06	-1.96	-1.73	-3.02	-4.61	-1.44
	sentence-t5-large-ft	+0.34	-2.26	-1.93	-4.70	-5.12	+4.73	+0.81	-12.04	+0.62	-0.54	+1.23	+2.95	+2.92	-1.51
	instructor-base	+0.76	-4.41	-3.11	-8.68	-0.64	+4.42	-0.98	-11.18	+2.19	+3.71	-0.87	+0.69	+0.83	-2.72
	instructor-large	-0.90	-3.41	-1.69	-7.27	-2.01	+1.79	-1.40	-11.76	-0.64	-0.14	-3.78	-0.89	-1.50	-0.97
	SFR-Embedding-Mistral	-0.15	+0.29	+0.97	-3.94	+0.33	+5.72	+1.45	-9.91	+2.99	+2.63	-0.60	+2.60	+1.24	-0.59
	NV-Embed-v2	+0.31	+0.76	+0.68	-4.50	+0.68	+4.42	+0.81	-11.52	+1.84	+1.67	-1.11	+1.10	+0.51	-0.97

Table 17: Effect of perturbations on ***FactCheckTweet*** (MAP@10). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@10})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@10 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

Stage	Model	Casing		Typos		Negation		Entity Replacement		LLM Rewrite		Dialect			
		TrueCase	*Upper*	*Least*	*Most*	*Shallow*	*Double*	*Atleast 1*	*All*	*Least*	*Most*	*AAE*	*Jamaican*	*Pidgin*	*Singlish*
First-Stage Retrieval	BM25	+0.66	+0.00	-9.09	-11.87	+2.51	+5.94	-2.52	-19.21	+2.63	-2.12	-6.71	-10.22	-11.09	-2.39
	all-distilroberta-v1	-1.24	-23.82	-7.84	-13.44	+4.97	+10.45	+0.72	-11.20	+4.39	+1.42	-6.19	-8.71	-12.79	-7.15
	all-MiniLM-L12-v2	+0.00	+0.00	-13.63	-17.49	+2.53	+4.95	-3.58	-11.44	+1.98	-2.90	-7.14	-15.27	-14.08	-9.86
	all-mpnet-base-v2	+0.00	+0.00	-9.31	-11.11	-0.57	-0.75	-3.69	-3.56	-6.04	-9.15	-6.87	-8.56	-8.64	-9.11
	all-mpnet-base-v2-ft	+0.00	+0.00	-6.44	-8.20	+3.60	-1.45	-4.18	-12.75	+2.54	+1.52	-4.01	-5.84	-6.97	-3.85
	sentence-t5-base	-0.13	-8.68	-6.72	-8.23	-11.79	-11.50	+1.84	-2.18	-5.37	-4.51	-2.45	-9.27	-8.61	-4.19
	sentence-t5-large	-0.08	-6.23	-6.27	-7.61	-16.70	-5.91	-0.15	-12.02	-3.76	-6.14	-5.51	-6.64	-8.15	-5.30
	sentence-t5-large-ft	-0.91	-5.29	-6.01	-6.81	-7.19	-0.95	-0.24	-11.24	-3.29	-2.34	-3.50	-4.12	-6.49	-4.16
	instructor-base	-0.72	-8.25	-4.76	-8.30	+2.80	+0.22	-1.63	-10.23	-2.43	-4.30	-2.50	-4.66	-7.32	-6.03
	instructor-large	-0.46	-3.48	-4.54	-5.72	-1.13	-4.53	-2.27	-11.65	-0.08	-2.84	-2.64	-2.22	-3.25	-5.73
	SFR-Embedding-Mistral	-0.84	-0.38	+0.28	-4.24	+0.90	+2.53	-1.49	-9.23	-0.24	-0.76	-3.13	+0.15	-1.74	+1.44
	NV-Embed-v2	-1.32	-1.05	-1.46	-2.98	+0.09	-1.19	-1.36	-12.81	-2.05	-2.59	-3.19	-3.23	-3.36	-5.09
Reranking Recovery	BM25	+4.85	+5.76	+12.35	+11.25	+5.55	+6.03	+11.87	+13.50	+9.66	+12.17	+11.87	+15.85	+17.22	+8.41
	all-distilroberta-v1	+11.74	+15.84	+18.96	+14.62	+13.86	+15.57	+13.18	+12.80	+14.23	+15.15	+14.73	+22.34	+21.46	+16.21
	all-MiniLM-L12-v2	+11.79	+12.66	+19.11	+17.64	+14.44	+11.57	+17.30	+13.41	+12.98	+19.47	+19.98	+28.30	+18.27	+20.96
	all-mpnet-base-v2	+11.66	+12.20	+15.31	+18.06	+17.01	+17.18	+19.39	+14.94	+16.73	+22.45	+19.22	+18.41	+16.27	+17.29
	all-mpnet-base-v2-ft	+10.78	+11.34	+11.91	+11.96	+9.24	+14.67	+17.53	+13.35	+10.54	+13.77	+13.11	+15.25	+17.50	+13.42
	sentence-t5-base	+13.10	+17.09	+17.94	+16.77	+18.03	+22.36	+11.77	+12.11	+19.02	+17.06	+15.88	+23.02	+20.56	+19.53
	sentence-t5-large	+13.93	+17.91	+20.68	+15.46	+22.88	+19.25	+16.06	+18.70	+18.17	+19.90	+19.15	+19.33	+18.91	+19.23
	sentence-t5-large-ft	+9.97	+11.77	+14.85	+12.97	+13.31	+14.05	+15.37	+12.83	+14.36	+12.44	+13.93	+16.44	+18.52	+11.77
	instructor-base	+9.74	+12.04	+11.39	+9.49	+6.89	+10.64	+15.11	+12.54	+13.01	+16.46	+10.08	+13.47	+16.46	+11.62
	instructor-large	+7.91	+8.35	+14.55	+10.23	+10.86	+13.47	+15.23	+15.61	+10.71	+14.12	+9.54	+11.95	+12.43	+15.42
	SFR-Embedding-Mistral	+6.65	+6.62	+9.00	+8.71	+7.55	+8.33	+12.28	+11.83	+11.64	+12.05	+13.85	+14.60	+14.31	+9.24
	NV-Embed-v2	+2.28	+2.41	+4.20	+0.58	+2.56	+5.78	+6.70	+4.01	+6.53	+7.04	+2.16	+4.37	+3.91	+4.18
Overall Pipeline	BM25	+0.10	+0.35	-4.05	-7.93	+1.58	+3.31	-0.28	-19.04	+4.59	+2.29	-1.86	-1.78	-0.89	-1.00
	all-distilroberta-v1	-1.85	-20.33	-6.93	-16.88	+4.89	+9.32	-3.71	-18.23	+1.09	-1.22	-7.62	-2.91	-7.48	-7.09
	all-MiniLM-L12-v2	-0.25	+0.62	-8.95	-14.28	-0.29	+3.37	-4.88	-19.18	+1.22	+2.33	-1.14	-1.30	-9.80	-2.88
	all-mpnet-base-v2	+0.11	+0.66	-7.47	-6.52	+4.26	+5.91	-3.93	-10.41	-2.50	-0.12	+0.21	-2.51	-4.50	-3.96
	all-mpnet-base-v2-ft	-0.16	+0.39	-5.89	-7.59	+0.89	+4.64	-3.93	-9.96	+1.44	+3.55	-1.84	-1.44	-0.41	-1.37
	sentence-t5-base	-0.29	-4.85	-0.44	-3.13	-5.75	+1.10	-0.76	-3.90	+4.02	+2.84	+4.06	+4.19	+2.57	+5.97
	sentence-t5-large	-0.50	-2.67	-0.57	-7.13	-11.83	-1.27	-3.37	-6.13	-1.06	-1.96	-1.73	-2.98	-4.61	-1.44
	sentence-t5-large-ft	+0.35	-2.23	-2.01	-4.69	-5.26	+4.73	+0.82	-11.85	+0.55	-0.56	+1.26	+2.87	+2.85	-1.57
	instructor-base	+0.77	-4.47	-3.17	-8.62	-0.58	+4.49	-1.02	-11.35	+2.20	+3.72	-0.84	+0.54	+0.72	-2.82
	instructor-large	-0.94	-3.52	-1.73	-7.23	-1.90	+1.82	-1.41	-11.78	-0.74	-0.18	-3.89	-1.13	-1.61	-1.09
	SFR-Embedding-Mistral	-0.20	+0.23	+0.83	-3.97	+0.21	+5.67	+1.45	-10.22	+2.76	+2.57	-0.76	+2.36	+1.09	-0.81
	NV-Embed-v2	+0.24	+0.64	+0.59	-4.55	+0.64	+4.47	+0.80	-11.70	+1.64	+1.58	-1.31	+0.88	+0.26	-1.19

Table 18: Effect of perturbations on ***FactCheckTweet*** (MAP@50). **First-Stage Retrieval** shows the *retrieval gap* between unperturbed and perturbed inputs: $\Delta(\text{MAP@50})_{\text{retrieval}}$. **Reranking Recovery** measures the Δ*recovery gap* for the perturbed input set before and after reranking top-*50* candidates, with improvements highlighted in green and drops in red. **Overall Pipeline** shows the Δ*overall gap* for combined retrieval and reranking between unperturbed and perturbed inputs. Colors indicate positive or negative deltas compared to unperturbed. MAP@50 is used for all comparisons. Models finetuned on the task are indicated with the postfix `ft`.

trained exclusively on user-generated content and not on other variations (e.g., *dialect* or *entity replacements*), which limits its generalizability to perturbation types not included in the augmentation set. Finally, we note that the LASER models generally perform poorly on this task compared to the other embedding models we evaluate.

H Scaling Laws and Performance Drop

We report an analysis of the performance gap as reported in Table 2, and the models' size and embedding used to represent the sentences at each stage. Figures 6, 9 reports the correlation for the First-Stage Retrieval (embedding and model size); Figures 7 and 10 that for Reranking Recovery (embedding and model size); Figures 8 and 11 the Overall Pipeline (embedding and model size). We excluded BM25 from the analysis as there is no proper notion of embedding dimension for that technique. In general, we do not observe any significant correlation between the embedding dimension and the performance gap.

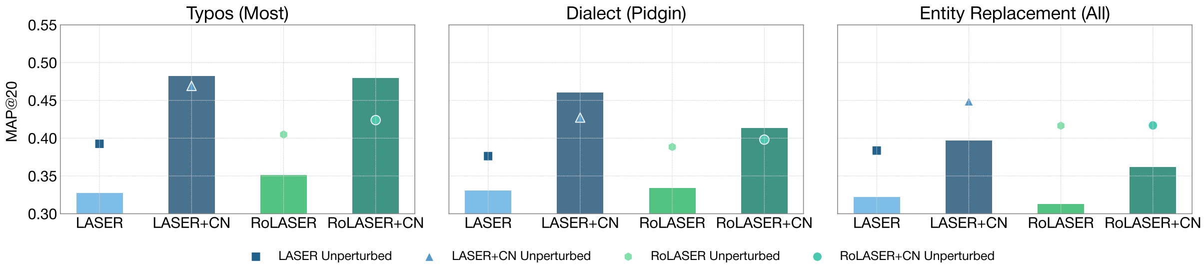

Figure 5: Effect of mitigation approaches on worst-case perturbations for *typos*, *dialect*, and *entity replacement*. For each embedding model, markers represent performance on the unperturbed input set, while the bars indicate performance on the perturbed set. RoLASER is a robust-LASER model adapted for user-generated content (Nishimwe et al., 2024), while models denoted with +CN incorporate *claim normalization*.

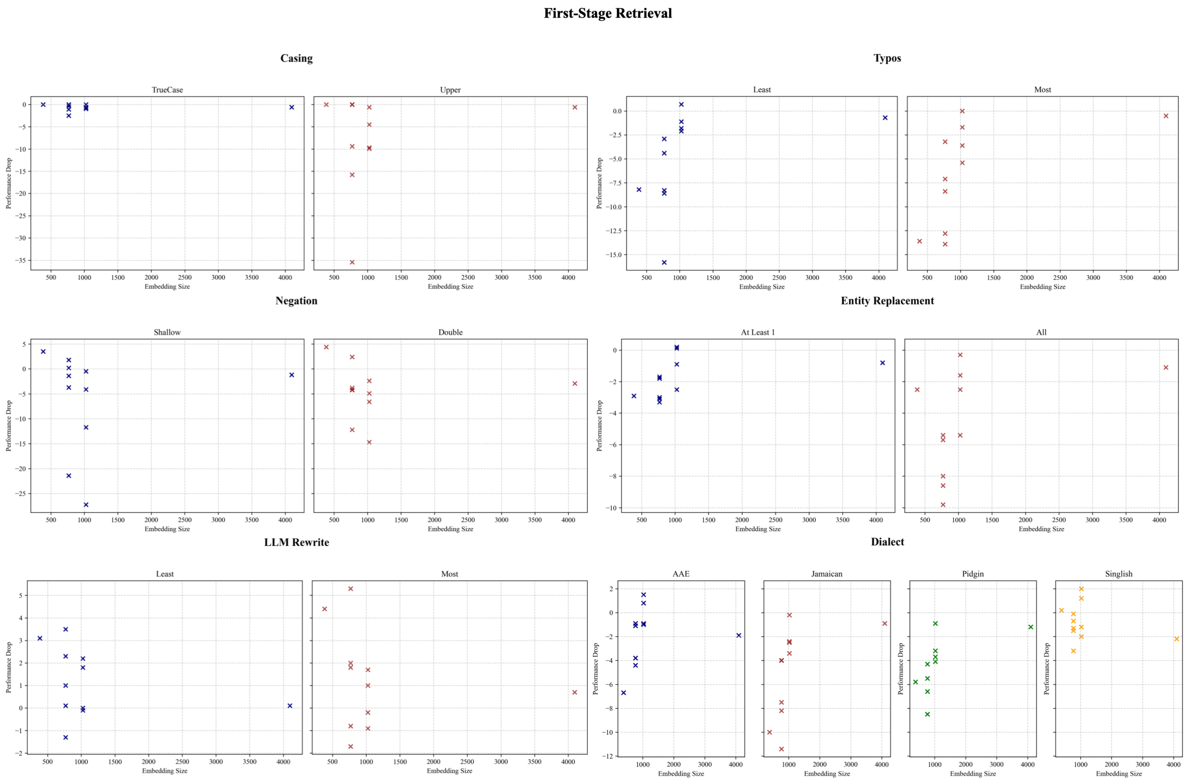

Figure 6: Correlation between a model embedding's size and the performance gap for the First-Stage Retrieval as reported in Table 2.

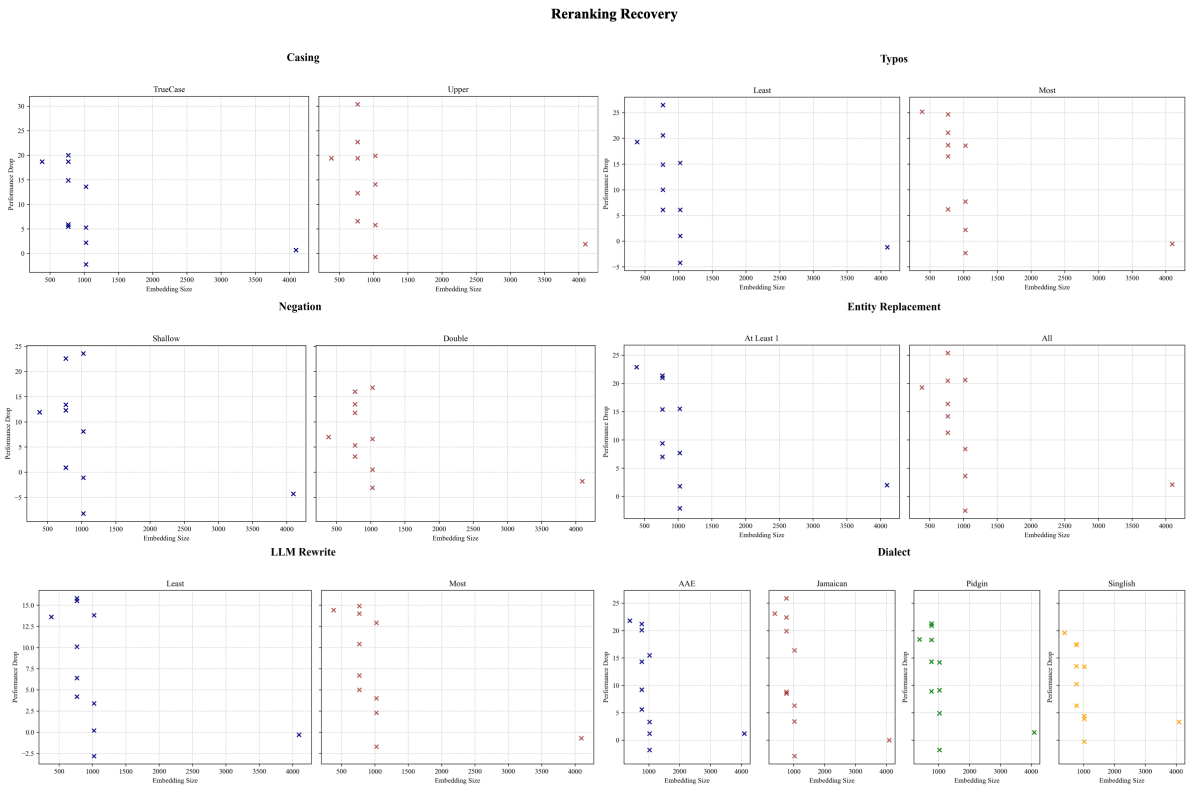

Figure 7: Correlation between a model embedding's size and the performance gap for the Reranking Recovery as reported in Table 2.

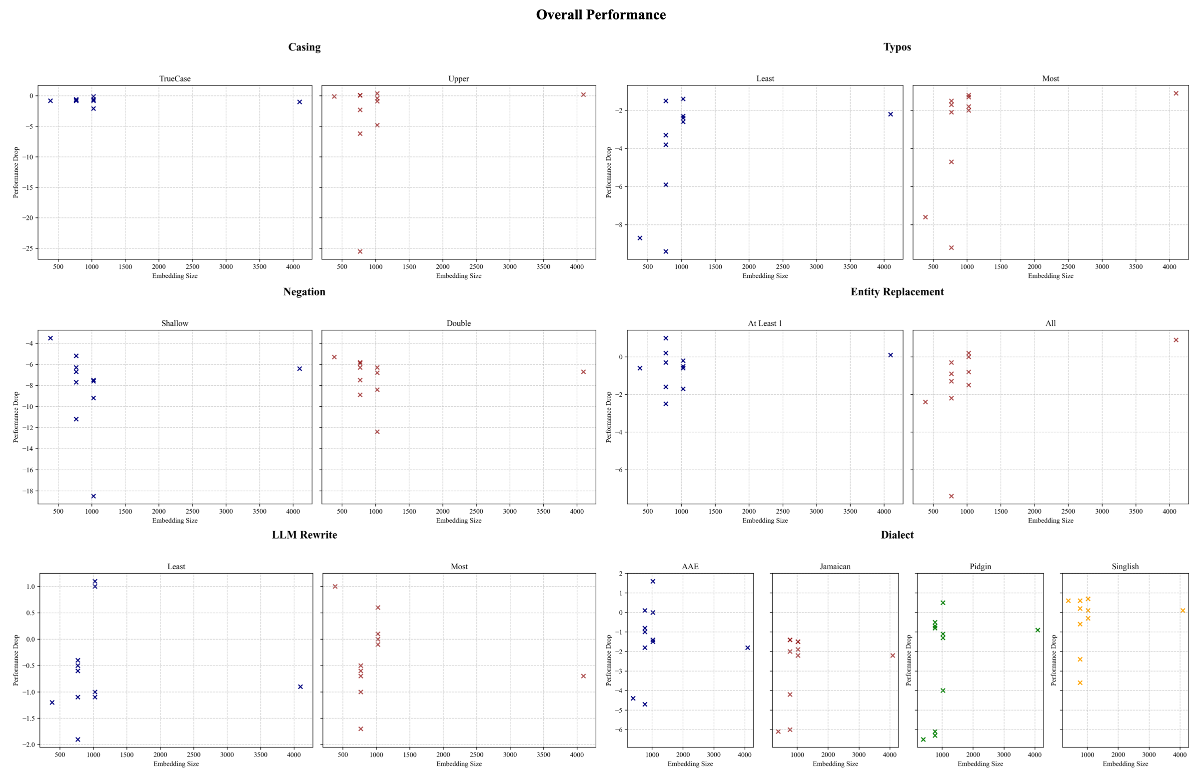

Figure 8: Correlation between a model embedding's size and the performance gap for the Overall Performance as reported in Table 2.

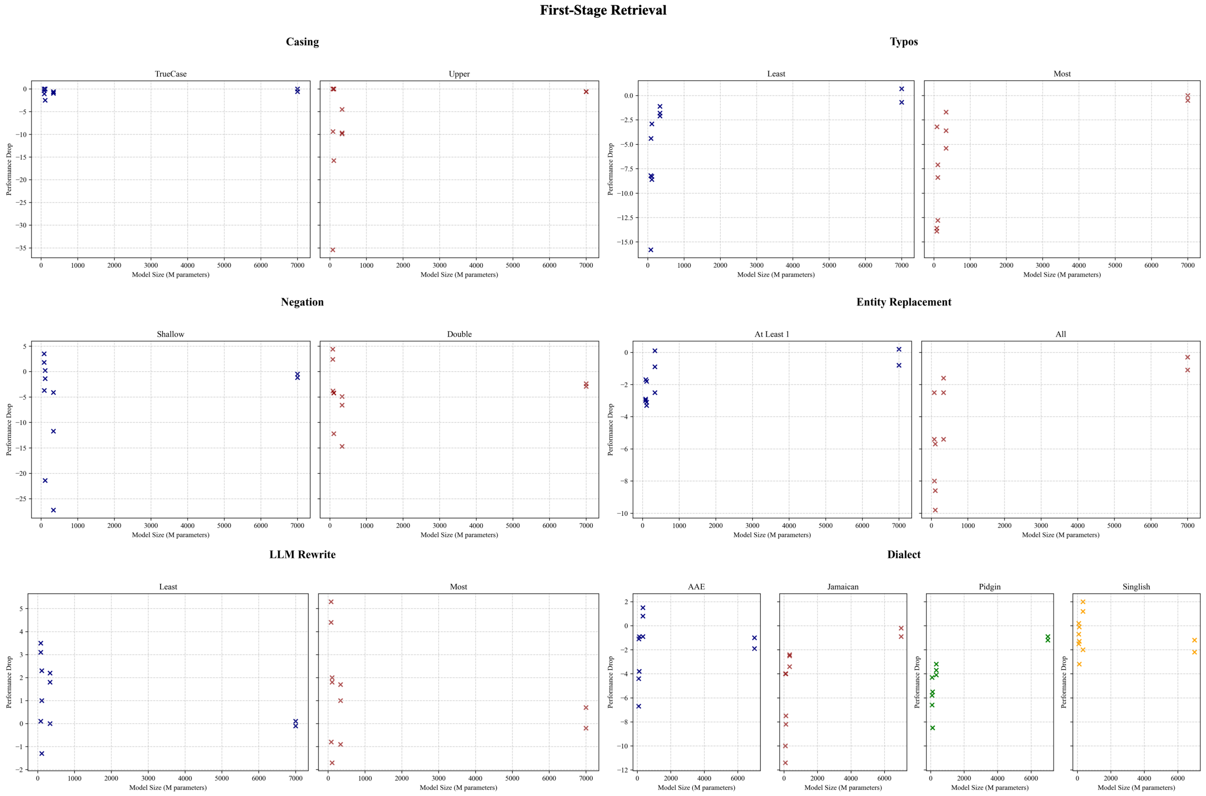

Figure 9: Correlation between model size (millions of parameters) and the performance gap for the First-Stage Retrieval as reported in Table 2.

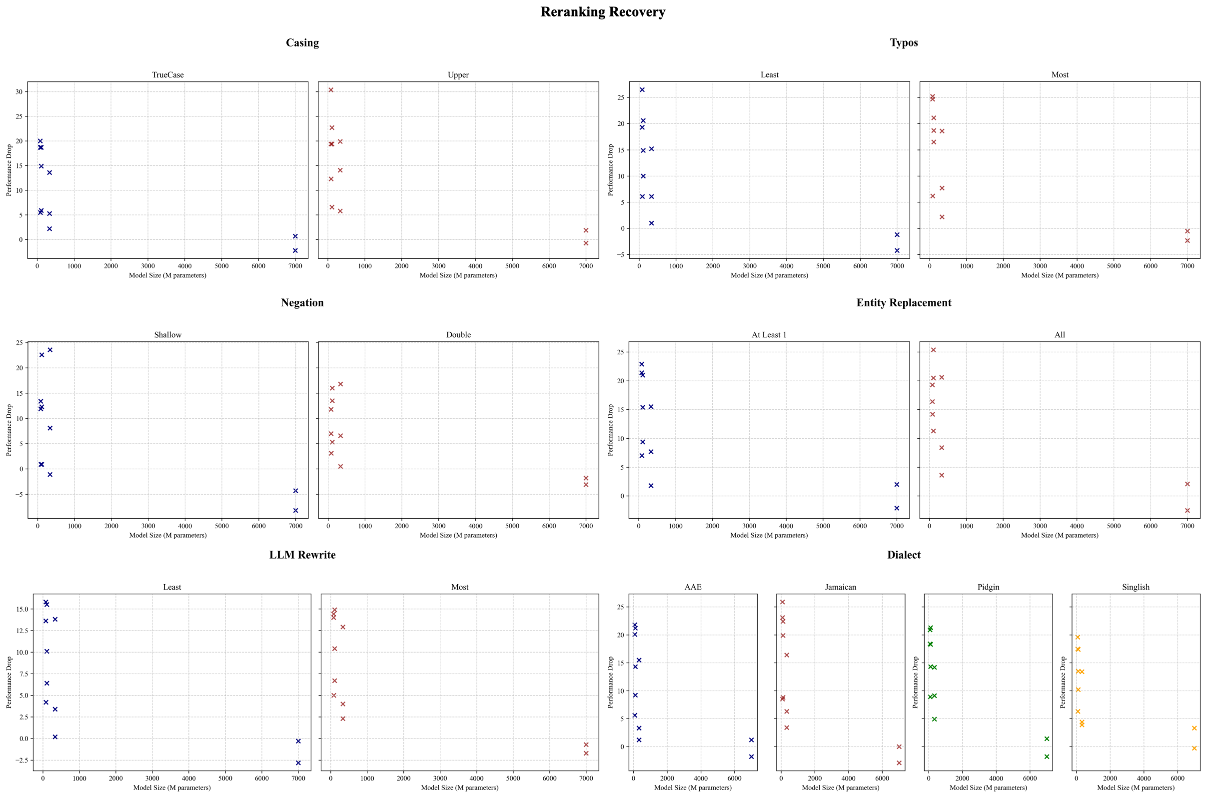

Figure 10: Correlation between model size (millions of parameters) and the performance gap for the Reranking Recovery as reported in Table 2.

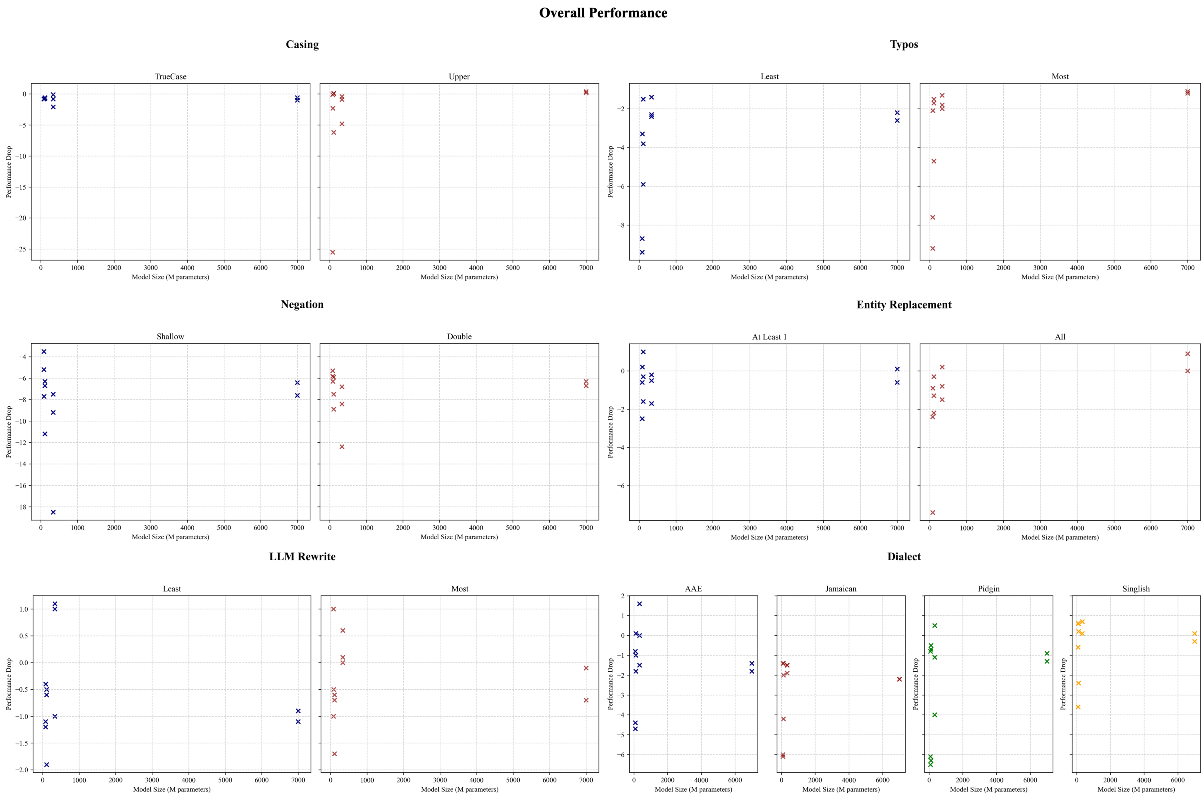

Figure 11: Correlation between model size (millions of parameters) and the performance gap for the Overall Performance as reported in Table 2.

I Prompts

We provide examples of prompts used in the perturbation generation framework, including the *LLM as a perturber* and *LLM as a verifier* prompts, as well as the *claim normalization* prompt used in the mitigation approaches. For brevity, we only present prompts for *Dialect* perturbation and verification; all other prompts are available in the repository. To encourage diversity in the generated perturbations, we set the generation temperature to 0.9, while for verification and claim normalization, we use a temperature of 0 to ensure reproducibility.

Dialect Perturbation

You are now a social media user tasked with rewriting a given tweet in different English dialects. You will receive two inputs:

1. A claim (original tweet)
2. A fact-check that supports or refutes the claim

Your task: Rewrite the given tweet in as many different ways as possible using the following dialects: African American Vernacular English, Nigerian Pidgin English, Singlish (Singapore English), Jamaican Patois
Ensure that:

- The fact-check remains applicable to the rewritten tweets.
- The rewritten tweets convey the same main claim as the original.
- The rewritten tweets read naturally and do not appear suspiciously altered.

Example:

Original tweet: "Biden signed an executive order today banning the term 'China virus'."

Fact-check: "President Joe Biden issued an executive order in January 2021 banning the term 'China virus.' "

Possible rewrites:

- Biden done signed an order today, no more sayin' 'China virus.'
- Biden don sign executive order today, e no go dey call am 'China virus' again.
- Wah, Biden just sign order today, cannot call 'China virus' liao.
- Biden sign one order today, fi stop di use a 'China virus.'

Response Format:

- Rewritten Tweet 1: [Your first rewritten version]
- Rewritten Tweet 2: [Your second rewritten version]
- . . . (and so on)

Inputs:

Tweet: {claim}

Fact Check: {fact_check}
Generate your response in the specified string format.

Dialect Verification

You are now a fact checker tasked with verifying whether a fact-check is applicable to a list of rewritten tweets. You will receive three inputs:

1. Fact-check: A statement supporting or refuting a claim.
2. Original Tweet: The source tweet conveying the claim.
3. Rewritten Tweets: A list of tweets rewritten based on the original tweet in different English dialects.

Your task: For each rewritten tweet, evaluate:

- Does the fact-check apply to it (i.e., is the fact-check helpful in verifying the rewritten tweet)?
- Does it convey the same main claim as the original tweet?
- Does the rewritten tweet read naturally, as if written by a typical social media user?

Your output: For each rewritten tweet, provide a binary label indicating whether the constraints above are satisfied.
Example:

Fact-check: "President Joe Biden issued an executive order in January 2021 banning the term 'China virus."'

Original tweet: "Biden signed an executive order today banning the term 'China virus'."

Rewritten tweets:
• Biden done signed an order today, no more sayin' 'China virus.'
• Biden don sign executive order today, e no go dey call am 'China virus' again.
• Wah, Biden just sign order today, cannot call 'China virus' liao.
• Biden sign one order today, fi stop di use a 'Bejing virus.'
Output:

```
{
    "labels": [1, 1, 1, 0]
}
```

Inputs:

Fact Check: {fact_check}

Original Tweet: {claim}

Rewritten Tweets: {rewrites}

Generate your response in the specified JSON format.

Claim Normalization

You will be provided with a noisy input claim from a social media post. The input claim may contain informal language, typos, abbreviations, double negations, and dialectal variations. Your task is to normalise the claim to a more formal and standardised version while preserving the original meaning.

Ensure that:

- The normalised claim conveys the same main claim as the original.

Let's see an example:

Noisy Claim: "Wah, Biden just sign order today, cannot call 'China virus' liao"

Normalised Claim: "President Joe Biden issued an executive order today banning the term 'China virus."'

Noisy Claim: "Soros son sez he and dad pickd Harris 4 VP after pic interview!"

Normalised Claim: "George Soros son revealed that he and his father chose Kamala Harris as the Vice President after a picture interview."

Noisy Claim: "It is not untrue that President-elect Joe Biden's German shepherd, Major, is set to become the first shelter dog in the White House."

Normalised Claim: "President-elect Joe Biden's German shepherd, Major, is set to become the first shelter dog in the White House."

Response Format:

Normalised Claim: [Your normalised claim]

Inputs:

Noisy Claim: {claim}

Generate your response in the specified string format.